UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Identifying and diagnosing video streaming performance issues

Georgios Dimopoulos

Identifying and Diagnosing Video Streaming Performance Issues

Georgios Dimopoulos

Advisors: Pere Barlet-Ros, Ilias Leontiadis

Ph.D. in Computer Science

Universitat Polytècnica de Catalunya BarcelonaTech
Department of Computer Architecture

Barcelona, September 2017

To Iró

Thesis advisors: Pere Barlet-Ros, Ilias Leontiadis Georgios Dimopoulos

Identifying and Diagnosing Video Streaming Performance Issues

Abstract

On-line video streaming is an ever evolving ecosystem of services and technologies, where content providers are on a constant race to satisfy the users' demand for richer content and higher bitrate streams, updated set of features and cross-platform compatibility. At the same time, network operators are required to ensure that the requested video streams are delivered through the network with a satisfactory quality in accordance with the existing Service Level Agreements (SLA).

However, tracking and maintaining satisfactory video Quality of Experience (QoE) has become a greater challenge for operators than ever before. With the growing popularity of content engagement on handheld devices and over wireless connections, new points-of-failure have added to the list of failures that can affect the video quality. Moreover, the adoption of end-to-end encryption by major streaming services has rendered previously used QoE diagnosis methods obsolete.

In this thesis, we identify the current challenges in identifying and diagnosing video streaming issues and we propose novel approaches in order to address them. More specifically, the thesis initially presents methods and tools to identify a wide array of QoE problems and the severity with which they affect the users' experience. The next part of the thesis deals with the investigation of methods to locate under-performing parts of the network that lead to drop of the delivered quality of a service.

In this context, we propose a data-driven methodology for detecting the under performing areas of cellular network with sub-optimal Quality of Service (QoS) and video QoE.

Thesis advisors: Pere Barlet-Ros, Ilias Leontiadis Georgios Dimopoulos

Moreover, we develop and evaluate a multi-vantage point framework that is capable of diagnosing the underlying faults that cause the disruption of the user's experience. The last part of this work, further explores the detection of network performance anomalies and introduces a novel method for detecting such issues using contextual information. This approach provides higher accuracy when detecting network faults in the presence of high variation and can benefit providers to perform early detection of anomalies before they result in QoE issues.

Thesis advisors: Pere Barlet-Ros, Ilias Leontiadis Georgios Dimopoulos

RESUM

La distribució de vídeo online és un ecosistema de serveis i tecnologies, on els proveïdors de continguts es troben en una cursa continua per satisfer les demandes creixents del usuaris de més riquesa de contingut, velocitat de transmissió, funcionalitat i compatibilitat entre diferents plataformes. A la vegada, els operadors de xarxa han d'assegurar que els continguts demandats són entregats a través de la xarxa amb una qualitat satisfactòria segons els acords existents de nivell de servei (en anglès Service Level Agreement o SLA).

Tanmateix, el monitoratge i el manteniment d'un nivell satisfactori de la qualitat d'experiència (en anglès Quality of Experience o QoE) del vídeo online ha esdevingut un repte més gran que mai per als operadors. Donada la creixent popularitat del consum de contingut amb dispositius mòbils i a través de xarxes sense fils, han aparegut nous punts de fallada que s'han afegit a la llista de problemes que poden afectar a la qualitat del vídeo transmès. Addicionalment, l'adopció de sistemes d'encriptació extrem a extrem, per part dels serveis més importants de distribució de vídeo online, ha deixat obsolets els mètodes existents de diagnòstic de la QoE.

En aquesta tesi s'identifiquen els reptes actuals en la identificació i diagnòstic dels problemes de transmissió de vídeo online, i es proposen noves solucions per abordar aquests problemes. Més concretament, inicialment la tesi presenta mètodes i eines per identificar un conjunt ampli de problemes de QoE i la severitat amb la que aquests afecten a la experiència dels usuaris. La següent part de la tesi investiga mètodes per localitzar parts de la xarxa amb un rendiment baix que resulten en una disminució de la qualitat del servei ofert.

En aquest context es proposa una metodologia basada en l'anàlisi de dades per detectar àrees de la xarxa mòbil que ofereixen un nivell subòptim de qualitat de servei (en anglès Quality of Service o QoS) i QoE. A més, es desenvolupa i s'avalua una solució basada en múltiples punts de mesura que és capaç de diagnosticar els problemes subjacents que causen l'alteració de l'experiència d'usuari. L'última part d'aquest treball explora addicionalment la detecció d'anomalies de rendiment de la xarxa i presenta un nou mètode per detectar aquestes situacions utilitzant informació contextual. Aquest enfoc proporciona una major precisió en la detecció de fallades de la xarxa en presencia d'alta variabilitat i pot ajudar als proveïdors a la detecció precoç d'anomalies abans de que es converteixin en problemes de QoE.

Thesis advisors: Pere Barlet-Ros, Ilias Leontiadis Georgios Dimopoulos

RESUMEN

La distribución de vídeo online es un ecosistema de servicios y tecnologías, donde los proveedores de contenidos se encuentran en una carrera continua para satisfacer las demandas crecientes de los usuarios de más riqueza de contenido, velocidad de transmisión, funcionalidad y compatibilidad entre diferentes plataformas. Asimismo, los operadores de red deben asegurar que los contenidos demandados son entregados a través de la red con una calidad satisfactoria según los acuerdos existentes de nivel de servicio (en inglés Service Level Agreement o SLA).

Sin embargo, la monitorización y el mantenimiento de un nivel satisfactorio de la calidad de experiencia (en inglés Quality of Experience o QoE) del vídeo online se ha convertido en un reto mayor que nunca para los operadores. Dada la creciente popularidad del consumo de contenido con dispositivos móviles y a través de redes inalámbricas, han aparecido nuevos puntos de fallo que se han añadido a la lista de problemas que pueden afectar a la calidad del vídeo transmitido. Adicionalmente, la adopción de sistemas de encriptación extremo a extremo, por parte de los servicios más importantes de distribución de vídeo online, ha dejado obsoletos los métodos existentes de diagnóstico de la QoE.

En esta tesis se identifican los retos actuales en la identificación y diagnóstico de los problemas de transmisión de vídeo online, y se proponen nuevas soluciones para abordar estos problemas. Más concretamente, inicialmente la tesis presenta métodos y herramientas para identificar un conjunto amplio de problemas de QoE y la severidad con los que estos afectan a la experiencia de los usuarios. La siguiente parte de la tesis investiga métodos para localizar partes de la red con un rendimiento bajo que resultan en una disminución de la calidad del servicio ofrecido.

En este contexto, se propone una metodología basada en el análisis de datos para detectar áreas de la red móvil que ofrecen un nivel subóptimo de calidad de servicio (en inglés Quality of Service o QoS) y QoE. Además, se desarrolla y se evalúa una solución basada en múltiples puntos de medida que es capaz de diagnosticar los problemas subyacentes que causan la alteración de la experiencia de usuario. La última parte de este trabajo explora adicionalmente la detección de anomalías de rendimiento de la red y presenta un nuevo método para detectar estas situaciones utilizando información contextual. Este enfoque proporciona una mayor precisión en la detección de fallos de la red en presencia de alta variabilidad y puede ayudar a los proveedores a la detección precoz de anomalías antes de que se conviertan en problemas de QoE.

Acknowledgments

I would like to extend my gratitude and appreciation to my advisors Dr. Pere Barlet-Ros and Dr. Ilias Leontiadis without whom this dissertation would not have been possible. Your guidance and support helped me to always stay motivated and focused, while your knowledge and expertise helped me greatly improve my work and my research skills.

I cannot thank enough Dr. Dina Papagiannaki for giving me the opportunity to fulfill my PhD in collaboration with Telefonica Research, but also for all her invaluable guidance and solid advice during my time there.

I would also like to thank Prof. Josep Solé Pareta for encouraging me to join the doctorate program and for his persistent efforts to help me overcome countless administrative issues since day one.

To Prof. Constantine Dovrolis, I would like to express my gratitude for his hospitality during my internship in GeorgiaTech and for his guidance and support throughout and after my stay there.

Finally, all my love goes to my family for never stop believing in me and to my wife for sticking by me through thick and thin. I would not be where I am now if it wasn't for you.

Last but certainly not least, a special thanks to Ranjeet, Maziar and Mario for all the laughs and talks we had during our coffee and lunch breaks.

Contents

List of Figures

List of Tables

List of Acronyms

AP	Access Point
BDP	Bandwidth Delay Product
BIF	Bytes In Flight
CAD	Contextual Anomaly Detection
CDN	Content Distribution Network
CNAME	Canonical NAME
CUSUM	CUmulative SUM control chart
DASH	Dynamic Adaptive Streaming over HTTP
DNS	Domain Name System
DPI	Deep Packet Inspection
DSL	Digital Subscriber Line
DTW	Dynamic Time Warping
FCC	Federal Communications Commission
FLV	Flash Video
FQDN	Fully Qualified Domain Name

FTP	File Transfer Protocol
HAS	HTTP Adaptive Streaming
HSDPA	High Speed Downlink Packet Access
HSUPA	High Speed Uplink Packet Access
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IQR	Inter-Quartile Range
ISP	Internet Service Provider
kNN	k Nearest Neighbors
KPI	Key Performance Indicator
LAN	Local Area Network
MME	Mobility Management Entity
MOS	Mean Opinion Score
MP4	MPEG-4
NIC	Network Interface Card
PCA	Principal Component Analysis
PQSR	Presentation Quality Switch Rate
PSO	Particle Swarm Optimization
QoE	Quality of Experience
QoS	Quality of Service

RAB	Radio Access Bearer
RCA	Root Cause Analysis
RNC	Radio Network Controler
RRC	Radio Resource Control
RSSI	Received Signal Strength Indicator
RTT	Round Trip Time
SIM	Susbscriber Identity Module
SPI	Sector Priority Index
SVN	Support Vetctor Machine
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
URI	Unique Resource Identifier
VoD	Video on Demand
VoIP	Voice Over IP
WAN	Wide Area Network
WLAN	Wireless Local Area Network

1 introduction

On-line video streaming has become more popular and ubiquitous than ever before. There is a constantly growing number of users who are engaging video content from mobile devices and Cisco predicts that mobile video will increase 11-fold by 2020, accounting for 75% of total mobile data traffic [1]. At the same time, video streaming services are constantly enriching their libraries with new and higher bitrate content.

The increasing popularity of video streaming among users has led to demands for higher bandwidth availability, while maintaining a satisfactory Quality of Experience (QoE). To meet these requirements, operators are forced to constantly provision [2] their fixed and mobile access infrastructure in order to accommodate the high traffic volumes that are generated from video streaming.

At the same time, continuous monitoring of the different network segments for performance issues is necessary to ensure that faults and problematic links are identified as soon as possible. In this way, any problems that affect the users can be properly addressed before they start to impact their received QoE.

In this chapter we discuss in detail the reasons that motivated the different stages of this work and the challenges that had to be overcome. Next, we analyze the major contributions that this thesis makes and finally we present the details of the organization and the structure of this manuscript.

1.1 Motivations and Challenges

Tracking and maintaining satisfactory QoE for video streaming services is becoming a greater challenge for network operators than ever before. On one hand, many streaming services have introduced over the recent years modern video optimization technologies such as bitrate adaptation and video pacing as well as high resolution encoding allowing up to 4k vertical resolutions. At the same time, operators are required to deal with the currently growing trend among users to stream video content on mobile devices, that is causing a demand for higher bandwidth availability and better provisioning throughout the network infrastructure.

Although in the current state-of-the-art, there are multiple works which have introduced methodologies to measure the video streaming QoE based on the player [3], browser [4, 5] or the device [6] instrumentation, they rely on modifying the user's device in the form of either custom video players, browser plug-ins or a pre-installed tool-set for monitoring the video stream for performance issues. Such solutions are in general not preferable given that the device instrumentation requires the user's consent and action and interferes with the video delivery mechanisms which clearly hinder the scalability and accuracy of such systems. As a result, there has been an increased interest by the research community as well as the industry for novel methods and tools that can identify video QoE issues from passive network measurements.

These limitations have given rise to video QoE measurement approaches that use Deep Packet Inspection (DPI) [7, 8] to extract video metadata and QoE information from network packets. However, these methods are quickly becoming obsolete as popular demand for privacy has led many content providers to adopt end-to-end encryption, leaving network operators with only a handful of indicators for identifying QoE issues. For this reason, video QoE models must now be developed with performance metrics that are extracted from lower layers of the network.

Nevertheless, the detection of QoE issues is only the first step towards improving the quality of the service that is delivered to the users in a network. By itself, it does not provide any information regarding the underlying cause and what actions are needed to properly address these issues and prevent future ones from occurring. To achieve this it is necessary to understand from which vantage points (VP) along the data path it is necessary to install measurement probes and which performance metrics are correlated with video QoE issues.

In the current literature, many works have investigated the correlations between different QoE issues and user engagement [9, 10], while in [11] user behaviour is correlated with startup delay, redirections and server response time. Moreover, Schatz et al. [12] used passive network measurements at ISP-based VPs to infer the rebuffering frequency and duration. However, to the extent of our knowledge, none of the related works investigate the impact that different performance faults have on the user's experience and which are the important metrics that can be used to detect them. Furthermore, the literature does not study which VPs can be utilized to better identify where in the network and what type of fault has affected the video QoE.

One of the main challenges involved in this area of research, is that due to the heterogeneity of the devices and networks between the content server and the client, it is often difficult to pin-point the offending part of the network where the video QoE issues occur and to identify the root cause that is behind them. Apart from typical network-related problems such as delay, congestion or limited bandwidth, video streaming on mobile devices may also suffer from the device's hardware limitations, high load on the endpoints and problems in the wireless medium.

To properly troubleshoot the underlying faults that cause the degradation of the users' experience, providers are required to deploy measurement probes in multiple VP along the data path to collect and analyze all the necessary performance metrics. One important challenge to overcome however with this approach, is to determine the minimum number of VPs and KPIs that are required to get an accurate QoE estimation. In this way, it will be possible to monitor and troubleshoot issues using a minimal set of features and instrumented devices.

The growing popularity of video engagement over cellular connections in conjunction with the availability of higher bit-rate streams, has led to a constantly increasing demand for bandwidth. To cope with the demand and ensure that the requested service is delivered with satisfactory quality, mobile operators are forced to constantly optimize and upgrade their infrastructure. However, resources need to be allocated in the under-performing parts of the network after careful planning and analyzing in detail the performance down to the customer and the cellular sector level.

Currently, operators monitor the performance of their networks by complementing passive measurements with drive tests and controlled field experiments for fine-grained benchmarking and root cause analysis [13–17]. However, these approaches are costly, require a lot of resources and cannot be relied upon for a complete and uninterrupted view of the entire network's performance. At the same time, they do not provide a clear and straightforward correlation between performance metrics and the user's QoE.

As a result, there is a need for way to combine multiple performance metrics from the different network elements into a single metric that can be easily mapped to a QoE index. Such a method will significantly simplify the process of identifying the per-customer and per-sector QoE in large-scale networks with millions of users.

Such a task can often be daunting since it involves dealing with large volumes of traffic and numerous KPIs that are periodically collected from network radio access elements (i.e., sectors, towers, and controllers) to monitor both wireless channels and backhaul performance. There is a set of newly introduced factors that affect the user's experience, such as hardware and software changes (i.e. new mobile devices, new multi-media codecs) and new network technologies (i.e. LTE), while network operators are forced to redefine older thresholds for well known performance indicators when monitoring the performance of video streaming services.

The aforementioned approaches in this part can serve very well when dealing with issues that are already affecting the users, but they are not adequate for preventing future problems from occurring when the capacity of a network segment is exceeded again. For this reason, providers need to adopt reliable methods for detecting anomalies in the performance of a network as they occur and troubleshoot them before they escalate into QoE impairments.

However, the accurate detection without raising false alarms can become a challenging task when there is high variance in the traffic. The performance of an entity or a segment in a network can be identified as anomalous, if its behavior deviates significantly from a predefined normal profile.

Popular methods in the state of the art, define the normal profile based either on the previous behavior of a target sequence e.g. CUSUM [18], or the variance of the entire dataset e.g. PCA [19]. These methods may fall short when individual paths are characterized by natural high variance or when different normal profiles can be found when examining different regions of the network. Behaviors like these can be observed real-life scenarios such as the increased latency in a path of an ISP's core network during peak hours or the high packet loss due to an outage over a wide area caused by a natural disaster.

To minimize the probability of undetected anomalies or false alarms in these cases, it is necessary to perform anomaly detection while taking into consideration the behavior of the context that a measured entity or path belongs to. A context corresponds to a peer group where the members have similar behavior with the target in the same time window.

1.2 Thesis Overview and Contributions

The work that is presented thesis makes several important contributions. In more detail, the first two chapters present two novel methodologies for monitoring the QoE of video streaming sessions from clear-text and encrypted traffic respectively. Next, we introduce a a multi-vantage point system for detecting video QoE issues and identifying their root cause. In the following part of the thesis we introduce a data-driven methodology that allows to combine multiple performance metrics from a cellular network into a single metric that can be used to measure the QoE of a service. Finally, we present a novel approach for detecting network performance anomalies using contextual information.

1.2.1 Measuring Video QoE from Clear-text Traffic

There is currently a high demand for non-intrusive and scalable video QoE monitoring solutions that can be seamlessly deployed on large-scale networks to accurately report quality impairments without the requirement for software or hardware modifications. Towards this end, the first part of the thesis investigates the extraction of QoE information from clear-text network traffic using strictly passive measurements.

Specifically, Chapter 3 presents a novel approach that allows the extraction of video streaming performance and user experience related metrics by means of passive network measurements. Unlike active methods presented in previous works [3–6] that require the modification of the player and/or additional client-side software, our solution only relies on the analysis of the video traffic between the clients and the content server. As a result, the proposed approach is much more scalable and more reliable as it does not interfere with the video delivery mechanisms.

One of the most notable contributions that the work in Chapter 3 makes, is the methodology for identifying, reverse-engineering and extracting YouTube QoE performance metrics directly from the metadata of the HTTP traffic that is generated during a video session. This makes the passive measurement of video QoE simpler and more detailed than previously done. To our knowledge, this is the first work to reverse engineer these requests, which provide a simple mechanism to track YouTube sessions passively and accurately

1.2.2 Measuring Video QoE from Encrypted Traffic

However, recent developments in video streaming technologies and services such as the adoption of adaptive streaming and end-to-end traffic encryption threatens to render previously available QoE models and detection tools obsolete. Therefore, in Chapter 4 we explore new methodologies that enable the video QoE monitoring in the presense of encrypted traffic and the latest streaming technologies.

The work presented in Chapter 4 is the first one, to the extent of our knowledge, to provide a solution that can identify from encrypted data multiple types of QoE issues that occur in traditional but also modern streaming technologies, as well as the severity with witch these issues affect the user's experience. Moreover, it is the first approach of its kind that was fully developed and tested with more than 390,000 video sessions from the mobile network of a large provider with more than 10M customers. We demonstrate that the models we developed can identify quality issues from unencrypted data with accuracies between 78% and 93.5% and from encrypted traffic with accuracies between 76% and

91.8%.

Moreover, this work provides important insights about the information that can be extracted from encrypted traffic. Our results indicate that i) passive measurements from a single vantage point are enough to accurately detect the key factors that affect the users' experience ii) we discuss on the features that are the most significant for detecting each particular problem iii) we demonstrate that client instrumentation is not required.

The contributions of Chapter 4 extend and complement the contributions made by Chapter 3. The work in Chapter 3 deals with the detection of the QoE from clear-text video streams which is developed using fixed-access traffic, whereas Chapter 4 presents a novel methodology for video QoE detection from encrypted traffic that was generated by mobile devices over a large-scale cellular network.

When compared to previous related works [20–22], aside from being compatible with encrypted traffic, our approach can detect all issues that affect the video streaming quality with significantly higher accuracy but also report with finer detail the severity with which the user was affected.

1.2.3 Identifying the Root Cause of Video Streaming Issues

Video streaming on mobile devices is prone to a multitude of faults and although well established video QoE metrics such as stall frequency are a good indicators of the problems perceived by the user, they do not provide any insights about the nature of the problem nor where it has occurred. Quantifying the correlation between the aforementioned faults and the users' experience is a challenging task due the large number of variables and the numerous points-of-failure.

To address this problem, Chapter 5 presents a distributed diagnostic framework for video streaming issues. The framework can use a wide variety of network and hardware measurements collected at one or more vantage points along the video data path to identify and pinpoint the root cause of detected QoE problems.

Although there is substantial number of works in the related literature on path diagnosis [23, 24] and on correlating network QoS metrics with the video streaming performance [6, 9–12, 25], the current state-of-the-art does not offer providers and users with a proper tool-set that is capable of identifying, locating and troubleshooting network faults that result in video streaming issues. Therefore, the proposed framework that can perform each of the aforementioned tasks with high accuracy, is the main contribution of this work.

1.2.4 Combining Performance Metrics to Better Capture Per-Sector QoE in Mobile Networks

Mobile network operators are required to constantly monitor the performance of different segments of their network down to the cellular tower level in order to better provision the infrastructure and deploy upgrades according to the requirements of an area. To address the aforementioned problems, the next part of the thesis contributes a data-driven framework that improves the QoE monitoring flexibility with respect to the current state of the art solutions.

In the work presented in Chapter 6, we leverage data provided by a large mobile operator serving more than 10 million subscribers, and we extensively study how to combine KPIs to create rankings that better capture under-performing cellular sectors. Furthermore, we target to capture individual QoE components that affect users' experience such as web throughput, latency and video stalls. Our results indicate that the resulting ranking correlates three times more than the currently used method, hence offering a better vision on under-performing sectors and their relation to user experience.

The methodology to combine performance indicators into a single metric has been established by equipment vendors and operators based on logical decisions, Service Level Agreements (SLAs), and controlled experiments such as drive tests and field tests [14, 26–29]. However, it remains an empirical approach, funded on deep domain knowledge fine-tuned over the years. In contrast, our novel data-driven methodology that builds upon the already collected sector performance metrics and bridges them with different QoE metrics. By doing so, the system empowers operators with an automated methodology that provides better visibility on underperforming sectors. Moreover, our results indicate that the currently used solution that is based on thresholding is sub-optimal to identify critical sectors

1.2.5 Detecting Network Performance Issues with Contextual Anomaly Detection

Instead of retroactively attending to already existing QoE issues, it is often more valuable for operators to be able to prevent such issues from occurring altogether by early detecting performance anomalies in their network.

To address this problem, Chapter 7 proposes the use of the Contextual Anomaly Detection (CAD) methodology which allows a more accurate detection of anomalies based on contextual information, while at the same time minimizing the false alarm rate. This solution can directly benefit operators by reducing the number of generated support tickets and calls since individual performance issues can be identified and addressed as soon as they occur and before they affect a larger part of the network.

Specifically, this chapter makes the following important contributions. First, we introduce a novel methodology for detecting contextual network performance anomalies. Next, we present the benefits of detecting anomalies in network measurements using CAD and proposes methods to improve the state-of-the art algorithm in terms of accuracy and scalability. Finally, we evaluate the algorithm's accuracy when identifying the contextual information of a sequence and when detecting the anomalous sequences with both synthetic and real data.

1.3 Thesis Organization

This section describes the organization of this thesis dissertation. Chapter 2 provides important background related to the video streaming technologies and the QoE issues that affect them. It also discusses the methods for extracting related QoE metrics from encrypted and non-encrypted traffic and describes in detail the datasets that are used throughout this manuscript. Chapter 3, describes a novel method for tracking the quality of video streaming sessions without a requirement for client-side instrumentation. All the important statistical information about the status of the playback are obtained by reverse engineering the metrics in the related network traffic that is captured from the network of a large university campus. This work is is based on the conference paper published in [7]. Chapter 4 presents a methodology for detecting video streaming QoE issues from encrypted traffic. The methodology includes the development of predictive models for detecting different levels of QoE degradation that is caused by three key influence factors, i.e. stalling, the average video quality and the quality variations. The models are then evaluated on the production network of a large scale mobile operator, where we show that despite encryption our methodology is able to accurately detect QoE problems with 72%-92% accuracy, while even higher performance is achieved when dealing with clear-text traffic. The method and the related results have been published in a conference paper in [30]. Chapter 5 introduces a multi-VP framework for diagnosing and performing root-cause analysis on video streaming issues. The framework is capable of identifying, localizing and troubleshooting the faults that affect the received video quality. The framework is initially evaluated in a controlled lab testbed where a variety of faults are induced manually. Next, it is deployed and tested in a semi-controlled environment and then in the wild where faults occur naturally. The related conference publication can be found in [31]. The following chapter studies how to bridge sector KPIs to reflect Quality of Experience (QoE) ground truth measurements, namely throughput, latency and video streaming stall events. We leverage one month of data collected in the operational network of mobile network operator serving more than 10 million subscribers. We extensively investigate up to which extent adopted methodologies efficiently capture QoE. This work was published in a conference paper in [32]. Chapter 6 presents a novel methodology for detecting performance anomalies based on contextual information. The proposed method is compared with the state of the art and is evaluated with high accuracy on both synthetic and real network traffic. The related journal paper under submission can be found in [33]. Finally, Chapter 8 presents in detail the related work, while Chapter 9 concludes the thesis and introduces some ideas for future work.

2 Background

2.1 How Video Streaming Works?

For many content providers HTTP has become the preferred protocol for video delivery over the last few years. HTTP streaming combines advantages such as firewall pass-through and easy network address translation, but also the benefits of TCP, i.e. congestion control mechanisms and reliable packet delivery.

2.1.1 Traditional HTTP Video Streaming

In traditional HTTP video streaming, the video is downloaded as a single continuous file which represents a single quality setting. Moreover, video buffering is employed as an additional measure to compensate for jitter and short-term bandwidth variations.

Typically, each video session can be divided into two buffering phases, i.e. the start-up phase and the steady state [34]. During the start-up phase the player will download the first part of the video as fast as possible to quickly fill the buffer and minimize the initial delay before the playback begins.

Once the buffer has been filled up to a specific threshold and the playback has started, the video session goes into the steady state. This phase is characterized by *ON-OFF* cycles, also referred to as pacing, where the download is paused as soon as the buffer has been filled and resumes when it is reaching depletion.

2.1.2 HTTP Adaptive Streaming (HAS)

In contrast to traditional streaming, HAS videos are split on the server in multiple segments, each one corresponding to a few seconds of playback time. Each segment is encoded in a range of different quality profiles which are defined by the content provider.

Instead of requesting the entire video, the player performs HTTP requests to fetch consecutive segments. The quality profile of the next segment is determined as a function of the throughput with which the previous segment was downloaded and the available seconds of playback in the buffer. In this way, the representation of the video can change dynamically to adapt to changes in the network and minimize stalls.

2.2 What Factors Affect the Video QoE?

2.2.1 Initial Delay

The initial delay refers to the time spent from the moment the user requests the video until the playback begins. This delay has two components, the network delay and the initial buffering delay. The former consists of the resolution delay, the redirection delay and the initiation delay. The aforementioned delays are shown in Figure 2.2.1, that illustrates a timeline of the events that take place during a YouTube video session.

The resolution delay is caused by the time required to perform the DNS resolution of the video server's address. The redirection delay is a result of the content provider's CDN load balancing mechanism, that will redirect the video request to the preferred server which is selected based on criteria such as proximity and load. The buffering delay is a result of the

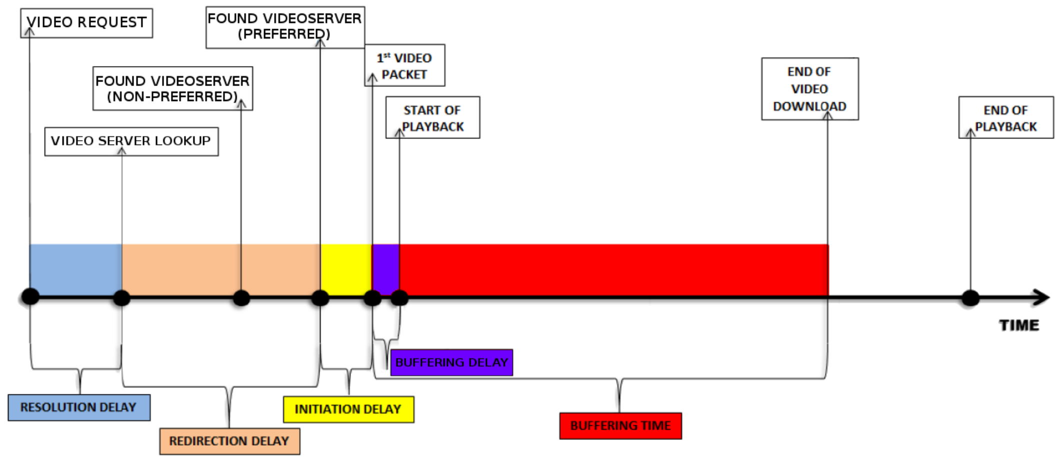


Figure 2.2.1: Initial delays during the pre-loading phase.

time required to preload the buffer with sufficient video data to allow a smooth playback. This latency depends on factors such as the size of the buffer on the user's device and the preload threshold set by the streaming service, but also on the speed with which the data arrives at the client.

It is interesting to mention at this point, that two independent studies by Mok et al. [35] and Etoh et al. [36], both found that the initial delays have the lowest impact on the QoE as users tend to be more tolerant to longer initial delays than other impairments such as stalls or quality switches. For this reason these delays are not taken into account when developing the methodology for Chapter 4, in order to focus on the other impairment types that have greater impact on the user's experience.

2.2.2 Stalls

Whenever the network throughput is not sufficient for the content to be downloaded faster than the rate that it is consumed, the buffer is depleted and the playback is forced to pause until more data are downloaded and the buffer is filled again. Hoßfeld et al. [37] showed that not only the frequency but also the duration of the playback stalls which occur due to buffer outages, have a high correlation with poor QoE. Specifically, the authors conclude that a video with 2 stalls of 3 seconds duration each, will lead to significantly lower Mean

Opinion Score (MOS). Moreover, Mok et al. [3] found that the rebuffering frequency has the highest impact on QoE and that a medium rebuffering frequency can result in a MOS lower by 2 points. In the work presented in this thesis, we measure the stalls using the *Rebuffering Ratio* which is expressed as the time spent stalling over the total duration of the video session.

2.2.3 Average Representation Quality

The average quality can be applied only to HAS video sessions, since only in these cases quality representation changes may occur. It is calculated as the average of all the individual qualities of the segments which belong to a video session. Multiple related works have shown a high correlation between the video representation quality and the user's QoE. In one of these studies [38], the subjective experiments performed in mobile networks have shown that video streams with higher quality representations are linked to better overall QoE.

2.2.4 Representation Quality Variation

Another factor that affects the QoE of adaptive video streaming, is the changes in quality variation. The variation in this case has two dimensions, the frequency of the changes and their amplitude. The frequency is the absolute number of changes that occurred in a video session, while the amplitude corresponds to the difference in magnitude between two consecutive qualities. In [39], the authors investigate how the representation switching amplitude and the switching frequency affect the QoE. Their results show that the switching amplitude has a very high impact on the user experience.

2.3 What Factors Affect the Video Streaming Performance?

In a typical video streaming session on a mobile device from a popular service such as YouTube, the video data is downloaded from a content server in a Content Distribution Network (CDN). As shown in Figure 2.3.1, the video stream is first transferred through Internet Backbone links to the client's ISP. Next, the data is downloaded to the mobile device over a broadband link connected to a home gateway/Access Point or a cell tower depending on the client's connection type.

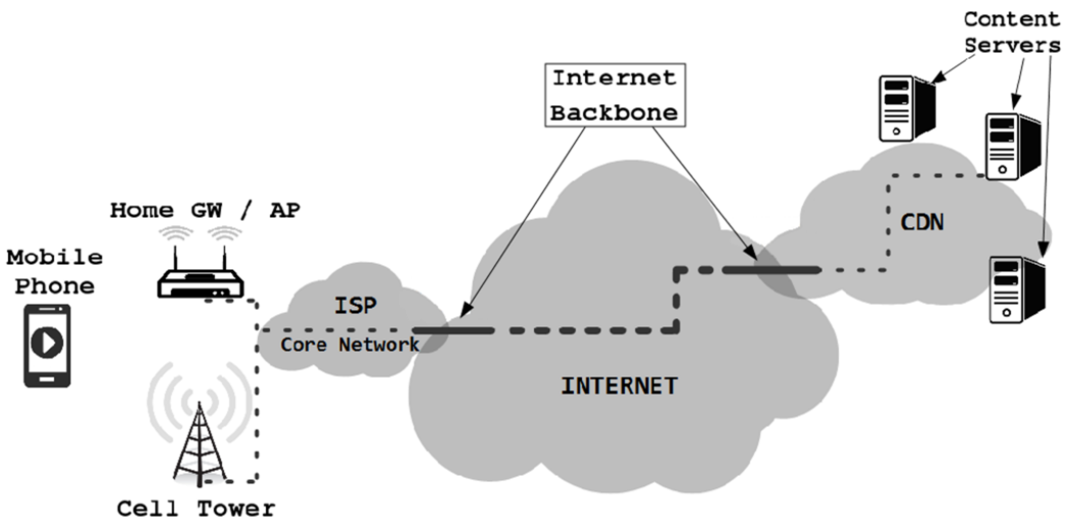


Figure 2.3.1: Video data path in a real world scenario.

In each hop along the data path, different types of failures may take place that can affect the performance of video streaming services and lead to QoE degradation. Such failures can occur in one or more layers of the devices that are responsible for propagating the data between the server and the client. Below are listed in more detail, the most prominent faults that can impact the video data delivery can originate from the hardware, the link/physical and/or the transport layer.

2.3.1 Hardware Faults

Hardware faults related to limited system resources such as high CPU utilization and low memory availability can affect network devices such access points, routers, switches, servers but also the end user's device. Such issues can cause increased processing times that lead to high latency when they occur on network devices and to video stalling when the user's device is affected.

2.3.2 Transport Layer Faults

HTTP video streams are typically delivered over the Internet using a transport protocol such as TCP or UDP. However, transport protocols can suffer from performance issues, e.g. latency, jitter and packet loss. These have a direct impact on the available throughput that the problematic link can deliver and in turn impact the initial delay of a video session and the rate at which the video data fills the playout buffer on the client, increasing in this way the chance of stalling.

2.3.3 Link/Physical Layer Faults

Portable devices are becoming more and more popular among users for streaming video content. These devices rely on wireless connectivity over WLANs or cellular connections to access and download the remote content. However, the wireless medium can suffer from an array of faults such as congestion, low signal reception and frequency interference. These issues contribute to the increase of loss and the reduction of the available bandwidth, wich in turn can result in delay and stalling.

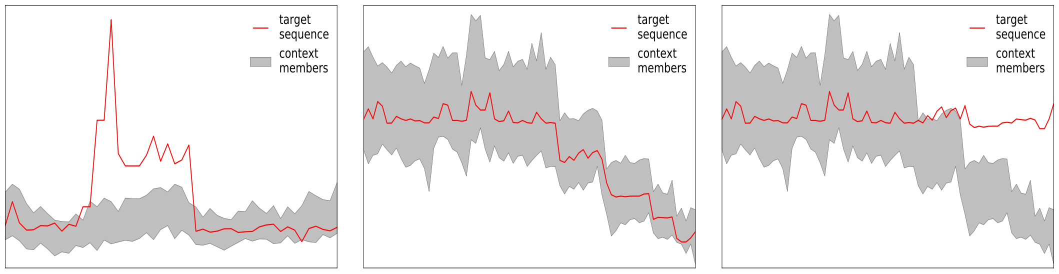
2.4 Network Performance Anomaly Detection

The performance of an entity or a segment in a network can be identified as anomalous, if its behavior deviates significantly from a predefined normal profile. Popular methods in the state of the art, define the normal profile based either on the previous behavior of a target sequence e.g. CUSUM [18], or the variance of the entire dataset e.g. PCA [19].

However, these approaches may fall short when individual paths are characterized by natural high variance or when different normal profiles can be found when examining different regions of the network. Behaviors like these can be observed real-life scenarios such as the increased latency in a path of an ISP's core network during peak hours or the high packet loss due to an outage over a wide area caused by a natural disaster.

To minimize the probability of undetected anomalies or false alarms in these cases, it is necessary to perform anomaly detection while taking into consideration the behavior of the context that a measured entity or path belongs to. A context corresponds to a peer group where the members have similar behavior with the target in the same time window.

The graphs in Figure 2.4.1 show real examples of a performance metric of a target in time series format (red), while the grey area shows the sequences that form the context.



(a) (b) (c)

Figure 2.4.1: Examples where the target sequence follows its context (b) and deviates from it (a), (c).

In Figures 2.4.1(a) and 2.4.1(c) the target sequences deviate from their context while in 2.4.1(b) the target follows the context's behavior. Previous detection methods may identify the red sequences in 2.4.1(a) and 2.4.1(b) as anomalous but not 2.4.1(c). However, if we take the context's behavior into consideration, 2.4.1(a) and 2.4.1(c) should be detected as anomalous but 1b should not.

To address this problem, we propose the use of the Contextual timeseries Anomaly Detection (CAD) methodology which allows the detection of anomalies based on contextual information with higher accuracy, while at the same time minimizing the false alarm rate. This solution can directly benefit operators by reducing the number of generated support tickets and calls since individual performance issues can be identified and addressed as soon as they occur and before they affect a larger part of the network.

3 Measuring Video Streaming QoE from Clear-text Traffic

3.1 Introduction

In this chapter, we analyze the YouTube video streaming service and the traffic generated from its usage. The purpose of this study is to identify by strictly using passive measurements the information that can be used as metrics or indicators of the progress of individual video sessions and to estimate the impact of these metrics in the user experience.

The proposed approach includes a novel method to track the progress of the video playback that in contrast to previous works, does not require instrumentation of the video player neither browser-based plug-ins. Instead, we extract important statistical information about the status of the playback by reverse engineering the metrics in related HTTP requests that are generated during playback. For the purpose of collecting these metrics, a tool was developed to perform YouTube traffic measurements by means of passive network monitoring in a large university campus network.

The analysis of the obtained data revealed that the most important source of initial delay in the video sessions is the redirection delay. The download rate analysis showed that YouTube initially uses a fast-start mechanism to quickly fill the video buffer and then switches to limited download rate in order to minimize unecessary data transfers. Further analysis revealed that the video advertisements and re-buffering events have the highest impact on the user experience, resulting in earlier video abandonment.

An important contribution of our approach is the discovery of the possibility to extract detailed information about user sessions from the statistical "s" requests, that will be presented in detail in Section 3.2.2. This allowed easier passive measurement of sessions and in more detail than previously done.

3.2 Methodology

In this section we distinguish two categories of measured variables. In the first category we measure the different processes in the video server selection mechanism which are responsible for inserting delay in the video session. Each delay is measured by comparing the time-stamps of the related HTTP events as shown in Figure 3.2.1. In the second category we present the set of parameters that we extract directly from the HTTP requests containing statistical information and those that are inferred from the later with simple calculations.

Both sets of parameters are of high importance with regards to the QoE a user experiences. On one hand, the initial delays provide an insight into the time the user has to wait until the video session is initialized and the playback begins. Long delays are indicators of poor performance of the YouTube video delivery mechanism due to factors such as multiple re-directions and congestion of the servers, and strongly affect the perceived quality of experience for the user.

On the other hand, parameters from the HTTP traffic are extracted throughout the entire duration of the video session and therefore can be used to monitor each part of a video session. Apart from returning important metadata about the video, they help us identify events such as stalls in the playback due to buffer depletion, the download rate at which the video was delivered to the client as well as the percentage of the total video that was watched. Another significant parameter derived from the HTTP parameters is the Viewed Ratio. This metric is used to indicate the number of abandoned video sessions and at which point of the playback they were abandoned. It is strongly related to the viewer's QoE given that users frustrated from poor performance tend to abandon their video sessions.

3.2.1 Measuring the Initial Delays

The first set of parameters that we measure, are those related to the delay introduced from the beginning of the video session, until the first video packet arrives at the client side. During this period, all the required elements of the web page are delivered to the user's browser and at the same time the preferred video server is located. In order to identify the preferred video server, the load-balancing mechanisms of the YouTube CDN attempt to make the best selection among those servers that are closer to the viewer and are not suffering from high load. This process consists of three sub-processes, each responsible for introducing delay in the video session as shown in Figure 3.2.1.

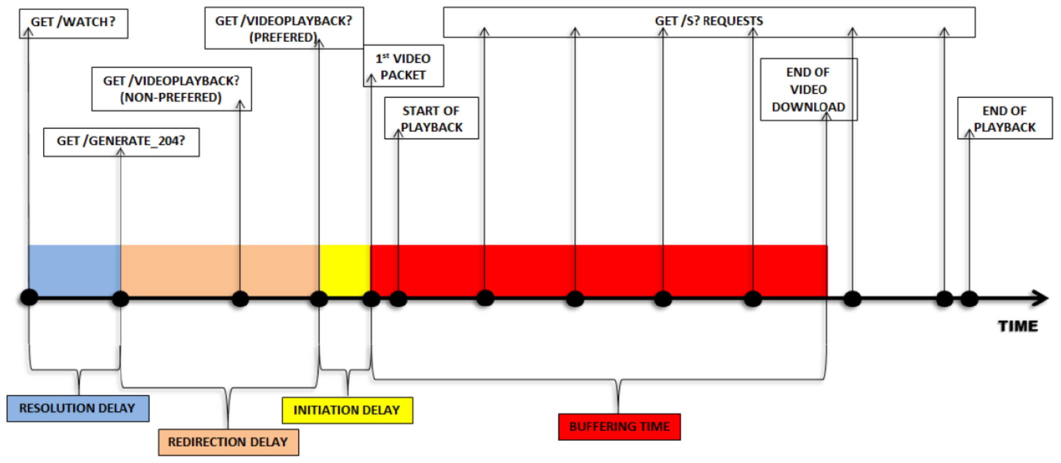


Figure 3.2.1: Important events in a YouTube video session

The first part of the video server selection is the "Resolution" phase that begins along with the video session by opening a YouTube video link and ends when the "generate_204" request is made. The "generate_204" is an HTTP GET request whose URI begins with the string "/generate_204?" and it is responsible for the DNS resolution of the video server. It is identical to the request that will be made in the next phase to get the video from the video server, with the difference that it returns a "204 No Content" from the server. This response indicates that the address of the video server was successfully resolved, however no video data will be delivered yet [40].

The following section of the video selection process corresponds to the "Redirection" phase. In this part, a "videoplayback" request is generated by the browser to start downloading the video. If the server responds with a code "302 Found", the requested video is located in this server but the client needs to be redirected to a more preferable server. At this point, multiple re-directions may occur until the best video server is located. If the server is the preferred, it responds with "200 OK" and the video will shortly start downloading.

The final phase that is responsible for generating delay in this process is the "Initiation" phase. This period marks the time required from the generation of "videoplayback" request to the preferred video server, until the arrival of the first video packet at the client side. The Initiation Delay that derives from this part, indicates the time that the server needs to process the request and start delivering the video.

Another important metric concerning the progress of a video session is the buffering time that corresponds to the time needed to deliver the entire video to the user. Until recently, it was possible to compare the buffering time with the duration of the video and infer buffer outage events that force the playback to halt in a re-buffering state until enough video data has been delivered in order to resume playback as done in [6]. However, lately YouTube modified the mechanism of delivering video data. Currently, the video download is paused when the buffer holds enough data to continue playback and it resumes when the buffer is close to depletion. As a consequence, if a user pauses the video playback, then the video download will also pause as soon as the buffer is filled and will not resume unless the playback resumes. Hence, the duration of the video download does not correspond to the buffering time and therefore it cannot be used an indicator of re-buffering events.

3.2.2 Measuring the QoE during playback

Moreover, after the beginning of a video playback a plethora of HTTP events are generated to maintain and update the video session (or occasionally deliver advertisement content). The most intriguing of these events are the "s" GET HTTP requests that can be identified from the "/s?" marking the beginning of the request URI, followed by a long line of parameters and their corresponding values. They are generated on short intervals of a few seconds as soon as the playback begins, while the generation stops if the video is paused by the user. The "s" requests are made to servers under the "s.youtube.com" domain. This particular domain's CNAME record points to the domain "video-stats.l.google.com" that is used Google (owner of YouTube), to collect statistical information from its services.

At this point due to the fact that neither YouTube nor previous studies have documented the meaning and usage of the parameters involved in the "s" requests, we proceeded to reverse engineer the Flash player object which automatically generates the aforementioned requests.

After reverse engineering and analyzing the player, we verified the statistical role of the parameters involved. Among those, the ones with the greatest significance for our study are the "bd" and "bt" that stand for bytes downloaded and bytes time. The bytes downloaded is the count of video data bytes the client received since the previous "s" request and the bytes time returns the time the later amount of bytes was required to be delivered. Therefore, from the ratio of these two parameters we can derive the download rate between the current "s" request and the previous one.

At this point it is worth to mention that to the best of our knowledge, the results obtained from reverse engineering the YouTube Flash player have not been published before.

Another two parameters that hold useful information are the "rt" that represents the time the "s" request was generated relative to the beginning of playback and "len" that indicates the duration of the video in seconds. By comparing the values of these variables we can infer whether the full length of a video was watched by the user or if it was abandoned earlier. We accomplish this by comparing the value of the last "rt" generated plus a timeout, against the length of the video. If the calculated value is smaller than the video length we conclude that the video was not watched to its entirety. In the other case, the complete video was watched.

Moreover "fmt" is an in-URL parameter of the "s" requests, used to specify the quality of the watched video in the form of a numerical identifier. The value of "fmt" can either be constant within a video session or vary if the user selects a different video quality during playback. In Table 3.2.1 the most popular video formats in our data-set are displayed along with their characteristics. The "B/s" column indicates the bit-rate requirements of each format.

fmt	**Characteristics**	**B/s**	**fmt**	**Characteristics**	**B/s**
5	240p FLV	40960	37	1080p MP4	792576
18	360p MP4	94208	43	360p WebM	118784
22	720p MP4	408576	44	480p WebM	163840
34	360p FLV	118784	46	1080p WebM	n/a
35	480p FLV	163840			

Table 3.2.1: fmt identifier with corresponding video characteristics

The last of the statistical parameters examined in this work is the "pd" which stands for player delay. The player delay is one of the most important variables identified during our analysis of YouTube traffic because it illustrates the time the playback was stalled due to re-buffering events throughout the video playback. In this chapter we will continue the "pd" abbreviation for the player delay but for clarity it will be referred to as re-buffering delay. Unlike the other aforementioned parameters that are returned every time an "s" request is generated, the re-buffering delay is only present if holds a non-zero value.

The importance of the re-buffering delay is clear since it provides us not only with information about the progress and performance of different parts of the video session but also with details about re-buffering events which are known to heavily influence user experience [41].

Finally, in this section we also define the Viewed Ratio parameter, as the ratio of the watched part of the video over the total duration of the video. To calculate the duration of the watched part of the video, we measure the time the last "s" request of the video session was generated through the value of "rt" and allow an additional time-out period of 10 seconds. If the calculated time is smaller than the video duration, then we conclude that the video was abandoned.

As the reader will observe, “s” requests can be used to extract very important parameters of YouTube video sessions. To our knowledge, this is the first work to reverse engineer these requests, which provide a simple mechanism to track YouTube sessions passively and accurately.

3.3 The Dataset

Our measurement scenario included seven continuous days of data collection from the campus network of UPC. The period of one week was selected in order to measure traffic under various time and day-of-the-week conditions such as network peak load periods. To perform the data collection, we developed a module able to identify and extract parameters related to YouTube traffic. The module was used as an extension of the CoMo passive network measurement platform [42], which was in turn installed on a dedicated machine, capable of capturing all incoming and outgoing traffic from the campus network over a full-duplex Gigabit Ethernet link. In Table 3.3.1 we provide important metrics from the obtained data set, while more information on the network and the related traffic can be found in the work of Sanjuas-Cuxart et al. [43].

# of video sessions	62778
Measurement period	26 Nov - 3 Dec 2012
Unique Video IDs	54847
# of sessions containing adv.	7423 (11.82%)
# of ads skipped	3719 (50.1%)
adv. time watched (mean)	21.31 sec
adv. full duration (mean)	35.29 sec
video duration (mean)	490.6 sec
session duration (mean)	172 sec

Table 3.3.1: Data set metrics

3.4 Results

In this section we initially measure the different parameters that can have an impact on QoE and later we proceed to estimate their impact on the QoE perceived by the end user.

3.4.1 Initial Delays

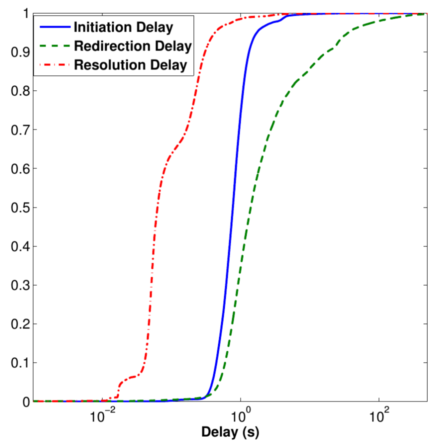


Figure 3.4.1: ECDF plots of the Initiation, Redirection and Resolution Delays

In Figure 3.4.1 the empirical CDF plots of the three initial delays are depicted. In more detail, we can see that the Resolution Delay contributes the least in the total delay before the video playback start, as it is the smallest in magnitude. Although there are some irregularities in the distribution of the Resolution Delay, the vast majority of its values are below 400 ms and approximately the 63% of all the values are lower than 90 ms. The variance that are observed around the 100 ms and 30 ms values can be associated with the different response and processing times presented by different video servers.

The distribution of the Initiation Delay shown on Figure 3.4.1, shows that 95% of the values are between 350 and 1100 ms. This distribution is more uniform than the previous and the overall variation of the values is the smallest among the three delays. Given that the Initiation Delay measures the time required to deliver the HTML code and different scripts from one server over a single TCP connection, the tight and uniform distribution of the Initiation Delay agrees with the fact that YouTube video pages share the same structure and are delivered from the same servers in a given geographical area.

On the other hand, the greatest contribution to the start-up delay comes from the Redirection Delay. This is attributed to some sessions suffering from many re-directions until the preferred video server is located. Each redirection that takes place may add a significant amount of delay to the particular part of the video session.

However this part of the session may also be further delayed from in-line video advertisements that are presented to the viewer before the beginning of the playback of the desired video. These advertisements are either non-skippable short clips with duration in the range of 10 seconds, or longer clips that can be skipped by the user after 4 seconds.

It is important to notice that the presented results concerning the Initiation delay are coherent with the corresponding Processing Time that can be found in [40]. Additionally, the Redirection Delay is not equivalent but can be considered analogous to the Startup Latency in the same paper. The ECDF plots of the two parameters are similar with the difference that Redirection Delay is overall larger. This can be attributed to the introduction of the advertisements in the video sessions in our data set, which heavily affect the Redirection Delay.

3.4.2 Download Rate

Figures 3.4.2 and 3.4.3 plot the empirical CDF of the Download Rate for four different video formats. In all the cases the 1st "s" request reports that the download rate has a high value, while in the following requests the download rate is reducing and after the 4th "s" request is converging to a certain value. These results indicate that YouTube is making use of a fast-start mechanism to fill as quickly as possible an initial buffer of large size. As soon as it is full, then the rate at which video data is sent to the browser gradually reduces, to reach to the minimum required download rate that will allow a smooth playback of a video with the given format. As mentioned earlier, the required data-rate for each format can be seen in Table 3.2.1.

The fast fill of a larger initial buffer serves the purpose of initiating the playback as soon

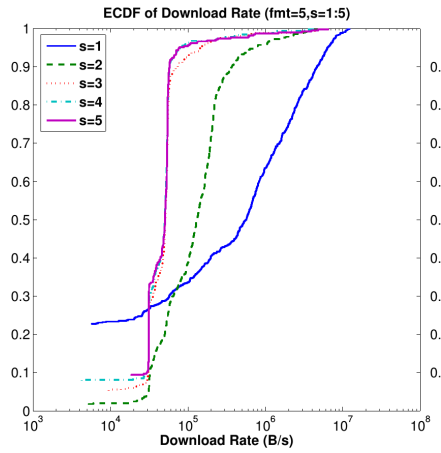


Figure 3.4.2: ECDF plots of the Download Rate per "s" request for fmt=5, 18

as possible, in order to reduce the initial delay and maintain playback in case of insufficient bandwidth at the client side. The second part of this mechanism limits the download rate to a minimum in order to minimize the unnecessary download video data in case the user skips a part of a video or aborts the video session.

3.4.3 Advertisements

Another important aspect of the YouTube video sessions are the video advertisements that are occasionally presented to the viewer before the playback of the requested video. Small advertisements with duration around 10 seconds cannot be skipped, while larger video ads allow the user to skip them after watching 4 seconds of content. Since the delivery of the advertisements occurs before the delivery of the requested video, they heavily affect the Redirection Delay when they are present. In our measurements with the parameter "Ad Duration" we calculate the time the advertisement was watched regardless to its duration so that we can accurately monitor how it affects the video session.

In Figure 3.4.4 we show the distribution of the ratio between the Ad Duration time and the Watched Video time. This ratio indicates the magnitude of the advertisement time compared to the video time. In the figure we observe that only approximately 12% of the video sessions included advertisement videos and there is a 2% where the advertisement

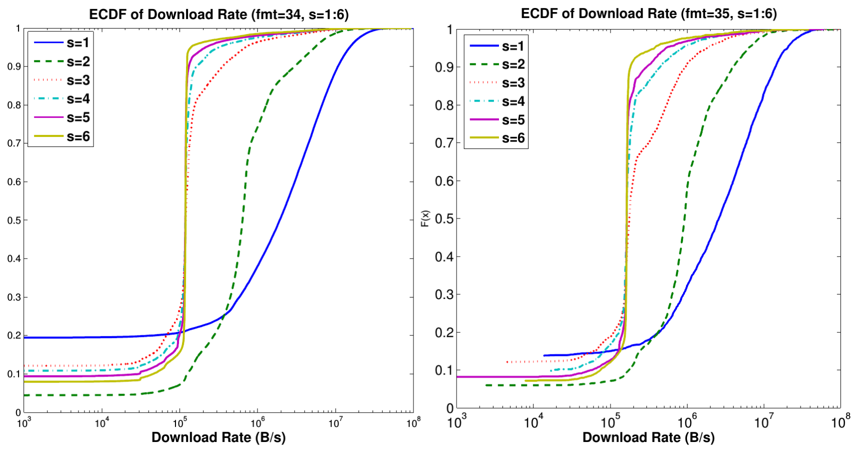


Figure 3.4.3: ECDF plots of the Download Rate per "s" request for fmt=34, 35

lasted longer than the video playback.

3.4.4 RE-BUFFERING DELAY

In Figure 3.4.5 we present the distribution of the re-buffering delay Sum over the complete data-set and the distribution of the parameter per "s" request. The re-buffering delay Sum is calculated from the addition of the individual re-buffering delay values introduced at each video session. The interesting observations in this figure, is the lack of re-buffering delay for the 73% of all the video sessions, while for the rest the values remain under 1 second with the exception of approximately 5% of the sessions. In the 5% of the video sessions with re-buffering delay over 1 second, the total delay can climb up to 60 seconds. To get a better insight of how the re-buffering delay evolves during the video playback, we investigated the values it takes per "s" request across all sessions. This information for the first 5 "s" requests of every video is shown in the ECDF plot in 3.4.5. The selection of the first 5 requests gives us a view of the first one minute of video playback. In the plot we can see that the re-buffering delay takes larger values in the first few seconds of the playback and in the 2nd "s" request it reduces by a small amount. However, in the following 3 requests the sessions there is an increase in the sessions where re-buffering delay is zero and for the complementary sessions, the value of re-buffering delay is decreased significantly. The above findings illustrate that users are more likely to experience large delay due to buffer

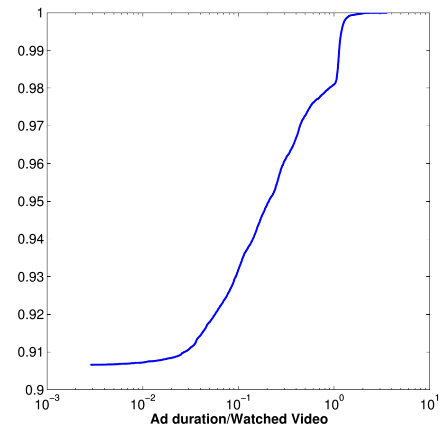


Figure 3.4.4: ECDF of the ratio between advertisement duration and video playback duration

depletion within the first thirty seconds of playback.

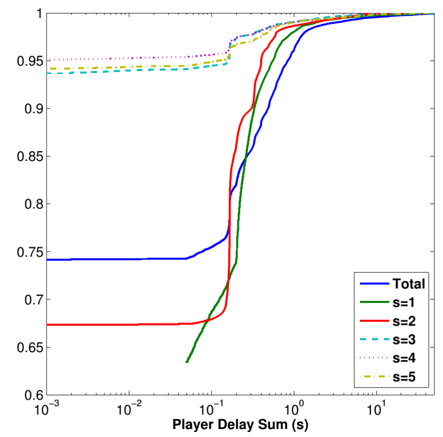


Figure 3.4.5: ECDF of re-buffering delay Sum for the complete data-set and per "s" request

3.4.5 QoE estimation

In this part we estimate the impact of the different metrics presented so far on the QoE of the user by measuring the abandoned video sessions. As mentioned in the Methodology section, the parameter that we have at our disposal for that purpose is the Viewed Ratio parameter which is calculated from the “rt” parameters of the “s” requests and the video duration. The plot in Figure 3.4.6 shows the ECDF plot of the Viewed Ratio Parameter. In this figure we can observe that only 40% of the video sessions in the data-set were completed and 40% where not watched over half of their duration. These results are similar to those presented in the corresponding section of [40].

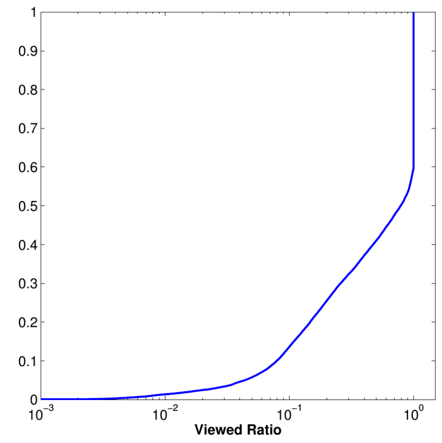


Figure 3.4.6: ECDF plot of Viewed Ratio

It is necessary to point out that a user may not watch a video to its full length due to lack of interest for the content. However, we illustrate in this section that the Viewed Ratio is strongly affected by parameters that indicate delay, stalling of the playback and the existence of advertisements. Therefore, we show through our findings that the Viewed Ratio can be effectively used as a metric of users’ QoE.

The three ECDF plots in Figure 3.4.7 show the relation of each of the initial delays with the Viewed Ratio. In each plot the corresponding delay is split into three classes in order to demonstrate how the Viewed Ratio is affected by the delay values in each class. In all three cases the diagrams indicate that when the respective delay is increasing the Viewed Ratio is decreasing. This means that users abandon sooner a video when they experience higher values of delay due to poorer QoE. Although in this figure we illustrate results for videos with medium length, the same observations can be made for larger or smaller video duration.

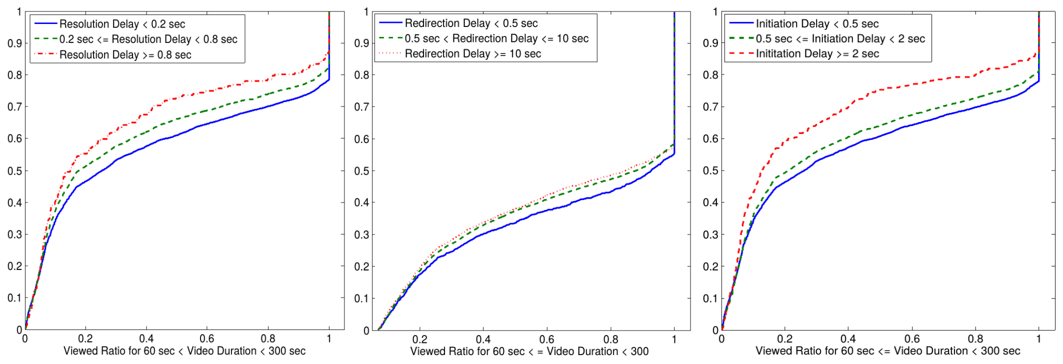


Figure 3.4.7: ECDF plots of Viewed Ratio for the Initial Delays.

In Figure 3.4.8 we show three ECDF plots, one for small videos with duration smaller than 1 minute, one for medium duration between 1 and 5 minutes and one for large videos lasting more than 5 minutes. In all the plots we examine how the Viewed Ratio is affected with different values of the re-buffering delay. In particular we examine in all three plots the following four cases: a) the total re-buffering delay per session is smaller than 0.5 seconds, b) between 0.5 and 1 second, c) between 1 and 2 seconds and finally d) the total re-buffering delay is larger than 2 seconds.

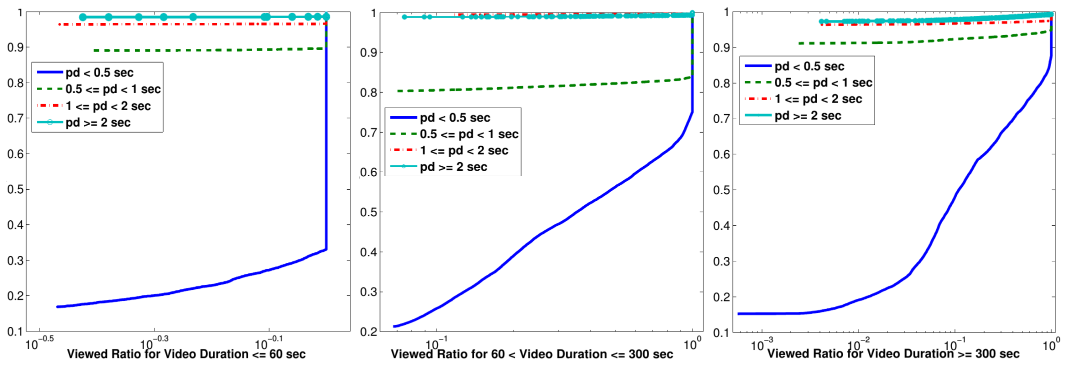


Figure 3.4.8: ECDF plots of Viewed Ratio for i) small, ii) medium and iii) large videos with 4 classes of re-buffering delay in each case.

Similar observations can be made for all the graphs in Figure 3.4.8. When the re-buffering delay increases then the number of abandoned video sessions is also increasing and when the re-buffering delay is larger than 2 seconds the vast majority of the video sessions are not completed. Therefore, we can see a strong correlation between the delay during a video playback and the amount of video the viewers watch. When a video session suffers from larger re-buffering delay, more and longer stalls in the playback are taking place due to re-buffering events and therefore the user's perceived QoE is degrading to a larger scale, causing the user to abandon the session earlier.

In order to investigate how the user's QoE is affected by the existence of video advertisements in the beginning of a session, we plot Figure 3.4.9. Here, we only take into account video sessions with advertisements. From these videos we distinguish those where the advertisement was skipped and those that was not skipped. This information is derived from the comparison of the full duration of the ad video against the time the user watched the ad. We assume that shorter watched times than the actual duration of the ad correspond to skipped advertisements. Hence, from the ECDF plots of Figure 3.4.9 we conclude that users who did not or could not skip the advertisement video, abandoned the video session earlier while less users watched the entire video. The implication of the later observation is that sessions where the ads were not skipped, resulted in poorer QoE for the users and led them to eventually abandon the entire video session sooner.

When making an overall comparison of the figures presented in this subsection, we observe that from all the studied parameters the one with the greatest impact on Viewed Ratio is the re-buffering delay as can be seen in Figure 3.4.8. The increase of re-buffering delay is causing a much greater increase in video abandonment than the increase of initial delay or the introduction of non skipped ads. As a result, larger re-buffering delay during a video session can cause greater degradation of QoE than the other two cases. This is observation is logical due to the fact that re-buffering delay is linked to buffer depletion that in turn causes stalls during the playback and strongly affects the user's experience.

3.5 Chapter Summary

In this chapter we studied the sub-processes of YouTube's video delivery mechanism and identified the Redirection Delay as the greatest contributor in initial delay. Moreover, our main finding in this work are the "s" requests which are used by YouTube to transfer sta-

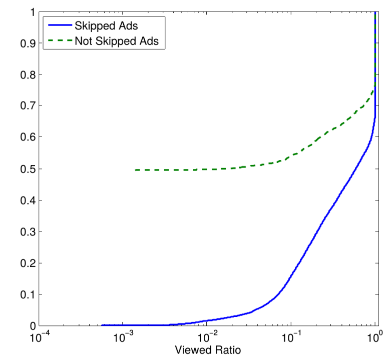


Figure 3.4.9: ECDF of Viewed Ratio for sessions with skipped Ads and Not skipped Ads.

tistical information. The analysis of these requests, helped us conclude that there are parameters involved which can be used to extract important information about anomalies throughout the video playback, such as re-buffering events and abandonment rate that are strongly correlated with the quality of experience of the user. In addition, we measured the impact that advertisement videos have on the initial delay and the video abandonment rate and showed that users are more likely to abort a video when the advertisements are not skipped. Finally, from our study we concluded that the re-buffering delay is the parameter with the highest impact on the user's perceived QoE that eventually causes viewers to abandon the watched videos.

4
Measuring Video Streaming QoE from Encrypted Traffic

4.1 Introduction

This chapter complements and extends the work that was presented in the previous chapter. However, in contrast to the previous methodology for monitoring the video streaming QoE from clear-text traffic, this part of the thesis presents a novel methodology for detecting video streaming QoE issues from both clear-text and encrypted traffic. In more detail, we develop predictive models for detecting different levels of QoE degradation that is caused by three key influence factors, i.e. stalling, the average video quality and the quality variations. The models are then evaluated on the production network of a large scale mobile operator, where we show that despite encryption our methodology can accurately detect different types of QoE problems with 72%-92% accuracy, while even higher performance is achieved when dealing with clear-text traffic.

4.1.1 Problem Statement

Adaptive streaming and encryption are nowadays the default technologies used by the majority of the popular content providers. The widespread adoption of these new technologies has given rise to a new set of challenges for identifying video QoE issues and has rendered previous solutions obsolete.

Deep Packet Inspection (DPI) solutions for extracting quality metrics, such as the video resolution and stall characteristics [12], [7], do not work anymore with encrypted traffic. Moreover, adaptive quality switching has introduced new factors that affect the user's experience, i.e. quality switching amplitude and frequency. However, these factors were not included in previous models for video QoE.

These changes in video streaming technologies, have caused a high demand, not only by network operators but also the by research community, for updated tools and methods for detecting and quantifying quality issues. Towards this end, this work aims to provide new methods for assessing the different types of impairments that affect the users' QoE from encrypted traffic.

Although many services have already made the migration towards adaptive streaming, their platforms continue to maintain backward compatibility with traditional static streaming. Therefore, one of the main challenges in this work, is to provide a solution which will be compatible with current but also previous video streaming technologies.

Moreover, with end-to-end encryption enabled, a great part of the metrics that were previously available in the network traffic for detecting QoE issues is now becoming inaccessible. For this reason, one of our challenges is to identify the right metrics from the limited amount of information that is provided by encrypted traffic and build the models to detect quality impairments. In order to accomplish that, we need to reverse engineer the video services and rely on machine learning and time series analysis.

Finally, in order to preserve the user's privacy but at the same to make our solution as generalizable as possible, we focus on developing a methodology that will be capable of detecting problems from network traffic alone and will not depend on the instrumentation of devices or video players and therefore it can be easily deployed by operators.

4.2 Dataset

The set which is presented in this section is constructed from unencrypted data that contains the ground truth for the QoE impairments of each video session. This information is then used to create the predictive models for identifying each impairment type. We then move to a set of encrypted data to validate the previously constructed models using controlled experiments.

4.2.1 Weblogs

The data is collected from a web proxy that is deployed on the cellular network of a large European provider. The proxy is capable of registering all unencrypted HTTP traffic including IP-port tuples, URI's, object sizes, transaction times, request time-stamps and more. Moreover, each log is annotated with a set of transport layer performance metrics, i.e. bandwidth-delay product (BDP), bytes-in-flight (BIF), packet loss, packet retransmissions and RTT. The BDP is equal to the link's capacity divided by its round-trip delay and represents the maximum amount of bytes that can be transferred by the link at any given time.

The dataset is created from YouTube traffic weblogs which are collected over a period of 45 days spanning from February to April 2016. From all the HTTP traffic that is generated by the service, we keep the weblogs that correspond to video and audio segment downloads and the signalling exchanged between the video player and the service during playback.

All the data is anonymized before the extraction by removing all private information such as user agents, subscriber and handset identifiers, MAC and IP addresses and so on. The only identifier which is preserved is the unique 16-character video session ID which is generated by YouTube. This parameter is described in more detail in Section 4.2.2.

We find that YouTube is the most suitable candidate among the currently popular video streaming services for developing and evaluating our methodology. The main reasons for this are i) the service's huge popularity which allows the generation of a very rich dataset in a short time window, ii) the diversity of the provided content in terms of video formats, qualities and durations, iii) its popularity among mobile users and iv) the adoption of modern technologies i.e. HTTP Adaptive Streaming (HAS), HTML-based video playback and pacing.

Moreover, most of the popular video sharing services are currently following YouTube's streaming paradigm, adopting adaptive streaming, a variety of supported codecs and HTML-based players.

Note that although Google has in the recent years deployed HTTPS for all of its services including YouTube, we can still observe significant amount of video sessions in clear-text HTTP in our dataset. This is attributed to the fact that many users use legacy devices or players that either do not support TLS encryption or do not have it enabled by default.

Nevertheless, we verified through experiments in the lab that apart from the encryption which is enabled by default, the delivery mechanism and overall behaviour of the app remains the same with newer devices with modern browsers and the latest version of the app.

In the weblogs, each segment download is generated from the client with a separate HTTP request and therefore we obtain a new entry for each new video chunk. From the list of metrics mentioned above, we also compute the *chunk size* and the *chunk time* that indicates the time when a video chunk arrives at the client, since in our experiments we found they bring relevant information to model the QoE impairments. The complete list of the metrics extracted from the traffic can be found in Table 4.2.1.

The final set consists of approximately 390,000 unique video sessions. However, only 3% of these are adaptive streaming sessions. This imbalance is expected since we are able to observe traffic from mainly legacy devices and video players which do not support the more recently adopted adaptive technology.

For the methodology of the stall detection we take the entire dataset, while for the development of the average representation and the representation quality switch detection we only keep the videos that made use of adaptive streaming.

Network Features	**Ground Truth (URI)**
minimum RTT	chunk resolution
average RTT	stall count
maximum RTT	stall duration
Bandwidth-delay product	video session ID
average bytes-in-flight	
maximum bytes-in-flight	
% packet loss	
% packet retransmissions	
chunk size	
chunk time	

Table 4.2.1: Metrics that we extract from the operator's web logs (left column) and the ones that are reverse engineered from the request URIs (right column). The features (left) are available for encrypted and non-encrypted flows whereas the ground truth is only available for non-encrypted sessions.

4.2.2 Ground Truth

From the meta-data that are passed as parameters in the URIs of the HTTP requests we are able to collect the ground truth that will be used in the evaluation phase. In more detail, these parameters carry three main types of statistics, i.e. generic device and player stats, content stats and playback stats [7].

The generic stats include information about the user's device such as OS, locale, screen resolution, player type and so on. One of the most important parameters here, is the unique video session ID. This ID is a 16-character hash that is randomly generated and it is unique to each session. We use it to identify and group together all the weblogs that belong to the same video session.

The content stats are extracted from the HTTP requests for downloading the individual video segments. One of the the parameters in this group is the 'content type', which indicates if the segment contains video or audio content and the multimedia container that was used to encode it, e.g. MP4, FLV or WebM. 'Itag' is another parameter which is used to specify the bit-rate, frame-rate and resolution of the segment, which we use to obtain the ground truth for the changes in representation quality throughout the session.

Finally, the playback statistics are included in the statistical reports that are periodically sent from the player to Google servers during the playback. Each report contains information that summarizes the progress of the playback since the previous report was generated. Different flags are used in the reports to specify if the video has successfully loaded, if the playback has started, paused or stopped and if there was a stall and how long it lasted. These indicators allow us to discover if a video was played throughout or abandoned and more important, identify the frequency and duration of stalls.

Out of the information that is available in the unencrypted data, we only use the chunk resolution, the stall count and duration and the video session ID (Table 4.2.1).

These features will be used as the ground truth for training the detection models in Section 4.3. After the completion of training phase, the access to the ground truth from unencrypted traffic will no longer be required and even if YouTube removes this information or deploys encryption for all sessions, the methodology will still be applicable.

4.2.3 Data Preparation

Before starting the analysis, we ensure that any logs that correspond to cached and/or compressed content by the proxy are removed from the dataset.

Next, after the ground truth for the stalls and representation switches is extracted, all the logs that belong to the same video session are identified by the common session ID and are then grouped together.

Thus, each entry in the dataset corresponds to a unique video session which includes information about the total number of stalls and their duration, as well as the characteristics of each chunk such as the quality representation, size, download time-stamp, but also the transport layer statistics like RTT, loss, re-transmissions, BDP and bytes-in-flight for each chunk download.

4.3 Building the Detection Framework

Our approach involves first the development and testing of the detection framework with unencrypted data. As soon as we verify that the constructed models can leverage the cleartext dataset, we can proceed to test the framework with data from encrypted video streams.

As mentioned in Section 2, there are three main types of impairments which may cause the degradation of poor video QoE, the frequency and duration of stalls, the session's Average Representation Quality and the Representation Quality Variation [38].

The initial delay is not considered as part of our video QoE model given its small contribution on the overall user experience as explained in 2.

In this section we describe the process of identifying from the limited number of metrics that are offered by the encrypted traffic, those that are the most significant for creating predictive models to detect each of the three types of impairments. An important part of this process is the feature construction that allows the generation of new more powerful features from the already existing ones.

Next, we show that there is a different set of metrics that better describes each type of impairment and contributes more information to the detection model.

In order to generate predictive models for detecting the level of stalling and the average representation, we use Machine Learning (ML) and in particular the Random Forest algorithm and 10-fold cross-validation.

Classification is preferred over regression given that we divide the data in discrete classes in both scenarios and the models are required to identify in which class each video session belongs based on the amount of stalling or the level of the average representation.

4.3.1 Stall Detection

Feature Construction

From the traffic features described in Section 4.2 (Table 4.2.1), we generate summary statistics, i.e. max, min, mean, standard deviation, 25th, 50th and 75th percentiles for each of the metrics, resulting in 70 new metrics.

Among all the performance metrics that we take into consideration, the chunk size is one of the most important for detecting stalls. If we take an example of a video session were stalling has occurred (Figure 4.3.1), we can see the significant changes in the chunk size when the two events take place at the third and the seventeenth second of the video session.

More specifically, whenever there is an outage on the player's buffer that results in a stall, the player will request small chunks which can be downloaded much faster so that the buffer will be filled as soon as possible and the video playback can resume. Then the size of the chunks will gradually increase and remain at a maximum value during the steady state as long as no further issues occur.

Therefore, we understand that we can significantly improve the accuracy of the stall detection model by including the sizes of the chunks in our feature set.

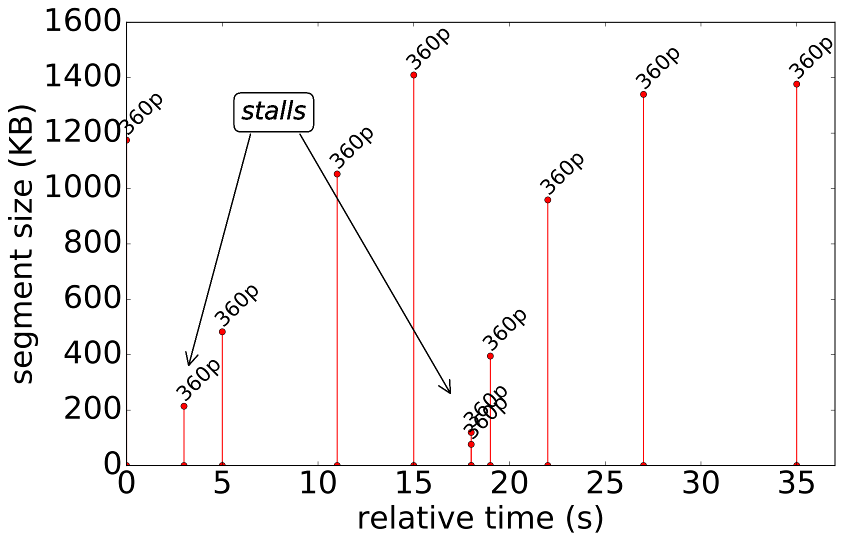


Figure 4.3.1: Changes in chunk sizes in a video session with stalls.

After all the required features have been generated, the dataset is then split into sessions without stalls and sessions where at least one stall has occurred. The information regarding the number of stalls observed during a video session and their duration, is the ground truth which is extracted from the meta-data of URIs as mentioned in Section 4.2.

Figure 4.3.2 (left) illustrates the distribution of the number of stalls that occurred per video session. We observe that 12% of all the sessions have suffered from rebuffering events, while about 8% was affected by more than 1 event.

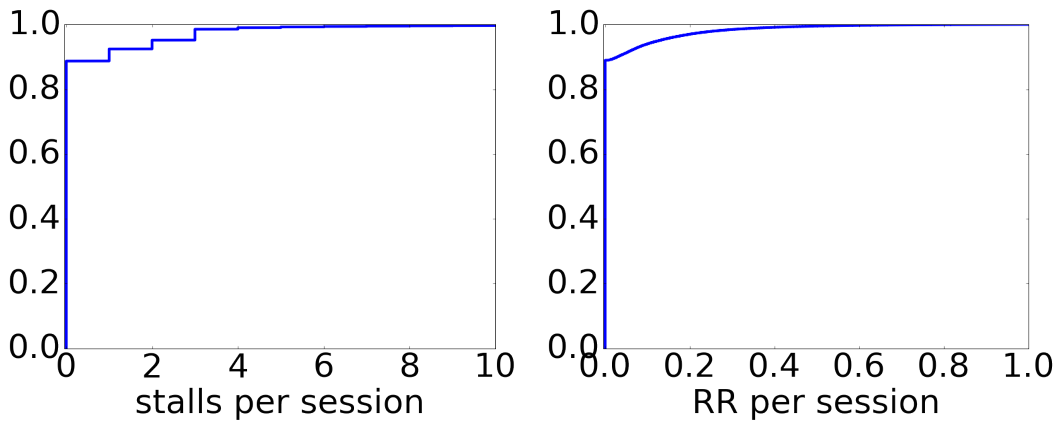


Figure 4.3.2: ECDF of number of stalls (left) and rebuffering ratio (right) per session

Labelling

Next, we use the information from the ground truth to label the data and create a predictive model. To do this, first we calculate the re-buffering ratio (RR) for each video session as the ratio of the sum of the duration t_{stall_k} of each of the total K stalls over the duration of the entire session t_{total} (eq. 4.1)

$$RR = \frac{\sum_{k=1}^{K} t_{stall\_k}}{t_{total}} \tag{4.1}$$

The sessions are then labelled according to the rule below. The definition of three levels of stalling, i.e. no stalling, mild and severe, allows a more detailed view of the degree to which the stalls affect the user.

$$Stall\ labels: \begin{cases} \text{``}no\ stalling: & RR = 0 \\ \text{``}mild\ stalling: & 0 > RR \geq 0.1 \\ \text{``}severe\ stalling: & RR > 0.1 \end{cases}$$

The RR threshold for distinguishing mild and severe stalling is set to 0.1, since in their work [44] Krishnan et al. have shown that when the RR is over 0.1, the severity of the stalling causes such a quality degradation that leads the users to abandon the video.

Figure 4.3.2 (right) shows the distribution of the RR per video session. We can observe that the sessions with RR equal or greater thatn 0.1 correspond to approximately 10% of the distribution.

Feature Selection

We then proceed to apply Feature Selection (FS) using the Correlation-based Feature Subset Selection (CfsSubsetEval) with the Best First search algorithm to reduce the number of features from 70 to the following four, BDP mean, packet re-transmissions max, chunk size min and the chunk size standard deviation.

The output of the feature selection algorithm reveals that there are three important factors that are correlated with stalling, BDP which is equivalent to throughput, number of retransmissions and chunk size. The limited throughput and increased number of retransmitted packets are QoS metrics which are performance indicators of congested networks and/or networks with limited bandwidth where stalling is more likely to occur.

Table 4.3.1 shows the gain of each of the features that were obtained after FS was applied and their respective information gains. The information gain represents the contribution of each feature in the construction of the predictive model. Features with higher information gain have a higher correlation with the problems that we want the model to detect and are used more frequently by the classifier.

The higher gains for the minimum and standard deviation of the chunk size indicate that both these features carry important information for detecting if a video suffered from stalls or not. Smaller chunk sizes correspond to lower quality streams that are frequently selected by the user or the adaptive algorithm in the presence of poor network conditions and limited bandwidth.

On the other hand, larger deviation of the size of chunks is related to sudden changes in the network's performance that in turn lead to quality switches during playback. In both cases the videos which are streamed under these conditions are more prone to stalling due to buffer outages.

The BDP and number of packet retransmissions have a more clear and direct correlation to low bandwidth and congestion scenarios where the speed at which the video buffer is filled is limitted and therefore there is a much higher probability of stalling. These metrics can be beneficial specially for cases of traditional streaming where the video is downloaded over a single connection.

info. gain	feature
0.45	chunk size minimum
0.25	chunk size std. deviation
0.18	BDP mean
0.12	packet retransmissions max

Table 4.3.1: Features and respective gains for the stall detection model.

Training and Testing the Predictive Model

In order to avoid biasing the results during the test phase, we balance the number of instances among the three classes before training the classifier. The instances in the classes are then restored to their original numbers for testing.

Overall, the classifier is able to make predictions with 93.5% accuracy. The proposed stall detection model is a significant improvement over previous approaches [20] where the achieved accuracy was approximately 84% for a binary classification. In contrast, our model not only achieves much higher accuracy but it also can predict the severity of the stalling that affected the user.

The output of the test phase of the model in terms of True Positives (TP), False Positives (FP), Precision and Recall can be found in Table 4.3.2, while the corresponding confusion matrix is shown in Table 4.3.3.

Precision is calculated as the ratio of TP over TP and FP and corresponds to the accuracy with which a certain problem is predicted. Recall is equal to the ratio of TP divided by the total instances in this class and measures the models's ability to correctly identify the QoE issue of a video session from the data set.

From the confusion matrix we can see that the classification errors occur between instances without stalls and those with mild stalls but also between mild and severe. However, significantly fewer misclassifications happen between the severe and "no stall" classes.

Therefore, it is straightforward that the errors occur due to the classifier's inability to correctly identify marginal cases where the RR is close to the RR thresholds we defined for labelling the instances. Hence, instances with RR slightly over zero can be falsely predicted as healthy sessions without stalls and thus increasing the number of FP. The same applies for cases where the RR is marginally over 10%, which can be identified as mildly problematic and vice versa.

In more detail, although some marginal instances belong to different classes, they often have similar characteristics, such as throughput delay and loss. The similarity between instances of different classes can cause confusion to the classifier resulting to the generation of FP.

From Table 4.3.2, we can see that the healthy sessions are predicted with higher Precision and Recall when compared to the other two classes. Moreover, the confusion matrix in Table 4.3.3 indicates that very few sessions have been misclassified as mildly or severely problematic.

These indicators show that healthy video sessions are streamed in significantly better network conditions as opposed to the problematic ones. This is translated to higher BDP and close to zero packet retransmissions for the vast majority of the instances. Additionally, healthy conditions allow higher quality streams with fewer or no quality switches. The combination of these characteristics allow the algorithm to easily distinguish healthy videos from problematic ones.

The separation of problematic sessions can be more challenging however, which can be verified from respective values in the confusion matrix. Here, in contrast to the healthy cases, there is a much higher number of misclassifications between the videos with mild stalls and those with severe stalls. In these cases, the chunk size often is not sufficient to indicate the amount of stalling. The reason for this is that frequently the minimum video quality is already selected due to limited bandwidth and therefore the minimum chunk size or its standard deviation will not contribute significant information for detecting the amount of stalling that took place during a video session.

Class	TP Rate	FP Rate	Precision	Recall
no stalls	0.977	0.111	0.965	0.977
mild stalls	0.809	0.035	0.816	0.809
severe stalls	0.793	0.009	0.887	0.793
weighted avg.	0.935	0.09	0.934	0.935

Table 4.3.2: Classifier's output for the stall detection model

original label	predicted label		
	no stalls	mild stalls	severe stalls
no stalls	97.76%	2.06%	0.18%
mild stalls	14.7%	80.9%	4.4%
severe stalls	4.2%	16.5%	79.3%

Table 4.3.3: Stall detection confusion matrix

4.3.2 Average Representation Detection

Feature Construction

In order to detect the average representation of videos with higher accuracy, in addition to the 10 features that are already available in the dataset, we construct five new ones, i.e. the chunk average size, the chunk size delta, the chunk time delta, the average throughput and the throughput cumulative sum. The chunk resolution is only used for the ground truth and labelling of the instances and not for the construction of the predictive model. Hence, we have a total of 14 features from which we extract the following statistics, minimum, mean, maximum, std. deviation and 5th, 10th, 15th, 20th, 25th, 50th, 75th, 80th, 85th, 90th and 95th percentiles. As a result, the total number of features we end up with is equal to 210.

The chunk average size is calculated from the sizes of all the individual chunks in a video. The size of a chunk has a strong correlation with the respective quality of the video segment. The chunk size delta represents the difference in the size of consecutive chunks while the chunk time delta corresponds to the inter-arrival time of video chunks. These parameters are indicators of representation switches which in turn affect the average representation of the session and will be discussed in more detail in Section 4.3.3.

Figure 4.3.3 presents a video session with a representation switch from 144p to 480p. Each point in the plot represents a video chunk, while the labels above the points indicate the segments' resolutions. The x axis corresponds to the video session relative time and the y axis to the size of the video segments. In this example there is a representation switch from 144p to 480p at $t = 22$ of the time line. This is translated to a significant increase for both chunk Δt and chunk $\Delta size$, which indicates that they can be relevant indicators of quality switches.

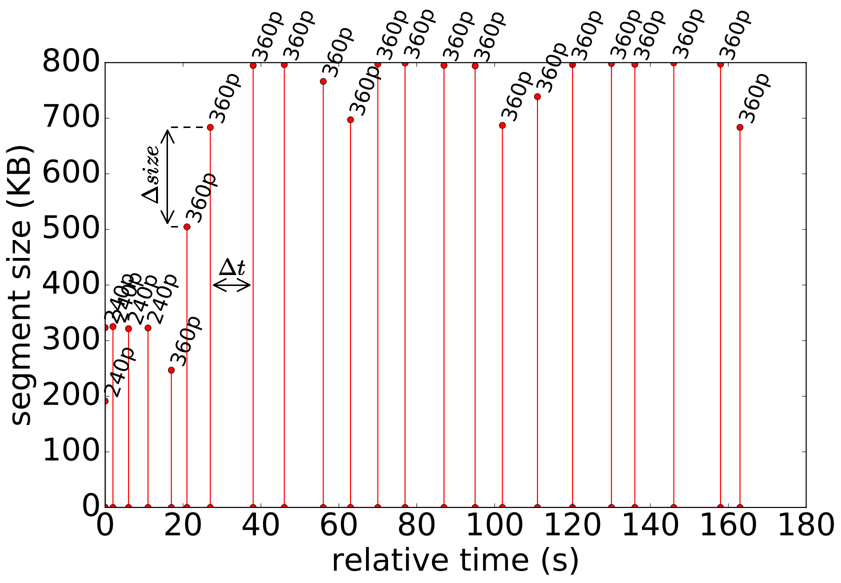


Figure 4.3.3: Δt and $\Delta size$ in a video session with a representation switch

The average throughput is calculated from the individual throughputs of all the chunks, while the cusum is their cumulative sum. The later is used as an indicator of variations in throughput.

Labelling

For the detection of the average representation of a video session, it is necessary to categorize the videos in three main categories based on their average resolution, low (LD), standard (SD) and high definition (HD). Given that in our dataset all the observed resolutions take only a few standard values, i.e. 144p, 240p, 360p, 480p, 720p and 1080p, we label all videos with resolutions 144p and 240p as LD, 360p and 480p as SD and all videos with higher resolution as HD.

In the dataset 57% of the videos have LD average quality, 38% have SD quality and only 5% have HD. This is an expected finding in our case where videos are streamed using limited mobile data plans and on handheld devices that often come whith smaller screens which leads users to opt for LD and SD video qualities.

However, we need to also account for cases where there are representation changes during the playback. For these videos, we calculate the average representation μ from the resolutions of all the segments. We proceed to label the instances in the dataset following the rule below for calculating the Representation Quality RQ.

$$RQ = \begin{cases} HD: & \mu > 480 \\ SD: & 480 \geq \mu \geq 360 \\ LD: & \mu < 360 \end{cases}$$

Feature Selection

The FS is again performed with the aid of CfsSubsetEval and Best First. After the selection there are 15 features remaining out of the initial 210. These features are listed in Table 4.3.4, ranked by their respective information gain.

We observe that statistics derived from the chunk size are the ones with the highest rank and represent the vast majority of the 15 features. This is a meaningful and expected result since the chunk sizes are highly correlated with the different representation qualities.

Moreover, the list of features also contains the BDP and the BIF which are proportional to the amount of bytes that can be delivered by the network but also the throughput cusum which is related to the throughput variations throughout the video session.

info. gain	feature
0.41	chunk size 75%
0.39	chunk size 85%
0.38	chunk size 90%
0.37	chunk size 50%
0.33	chunk size max
0.32	chunk avg size mean
0.22	BIF avg max
0.21	cumsum throughput min
0.2	chunk $\Delta size$ max
0.19	chunk size std
0.16	chunk $\Delta size$ std
0.15	chunk Δt 25%
0.06	BDP 90%
0.05	BIF maximum min
0.03	RTT minimum min

Table 4.3.4: Features used for the Average Representation detection.

Training and Testing the Predictive Model

The model to predict the average representation quality is again built using ML and Random Forest. The training is done with balanced classes and then the trained model is tested on the entire set. The obtained overall accuracy in this case is 84.5%. The accuracy for each class is provided in Table 4.3.5 and the corresponding confusion matrix in Table 4.3.6.

Class	TP Rate	FP Rate	Precision	Recall
LD	0.9	0.206	0.845	0.9
SD	0.768	0.106	0.82	0.768
HD	0.756	0.003	0.945	0.756
weighted avg.	0.841	0.156	0.841	0.841

Table 4.3.5: Classifier's output for the average representation model

The accuracies in the later table reveal that our model is able to predict the average quality of LD videos with very high accuracy but with slightly reduced accuracy in the case of

original label	**predicted label**		
	LD	SD	HD
LD	90%	9.9%	0.1%
SD	22.7%	76.8%	0.5%
HD	6.8%	18.2%	75%

Table 4.3.6: Average representation confusion matrix

SD and HD videos. Nevertheless, the overall but also the individual accuracies remain in high levels, which verify the model's good performance.

When further investigating the accuracy loss however, we identify that its caused by the increased number of misclassifications that occur in the SD and HD classes. More specifically, a considerable amount of SD video sessions is falsely detected as LD, while 18% of HD videos are identified as SD.

This behavior is attributed to the quality downscales that happen during a video session. As a result one part of the video is streamed in higher quality and the part after the downscale is streamed with lower quality. The differences in chunk sizes between the two qualities of a session lead to the incorrect classification of the video. Of course the effects of this phenomenon cannot be observed for LD videos since there is no lower quality to downgrade to and chunk sizes remain consistent throughout the session.

4.3.3 Representation Quality Switch Detection

Adaptive streaming can adjust the representation of the video during playback in order to compensate for changes in the network conditions and reduce the likelihood of playback buffer outages that lead to stalls. The duration and frequency of the representation changes, also known as Presentation Quality Switch Rate (PQSR), as well as the amplitude of the switch can have a negative impact on the perceived QoE.

Filtering

During the start-up phase, many content providers employ a fast start mechanism that allows them to fill the playout buffer and start the playback as fast as possible, effectively reducing the start-up delay. This short initial part of a video session may have very different characteristics in terms of segment sizes, inter-segment arrival times and throughput when compared to the much longer steady phase.

To reduce the noise introduced by the start-up phase in the detection of resolution variations, we remove the first ten seconds of all video sessions in our dataset. Given that this initial section represents a very small fraction of the entire video session (the average session duration is approximately 180 seconds), we can safely remove it to reduce the noise introduced by the start-up phase while maintaining more than 95% of the session.

Labelling

In order to build a model for quality switching detection, it is necessary to first quantify the switches in terms of frequency and amplitude. To this end, we define two metrics, the time spent in each representation t_r, the frequency of representation switches F and the switch amplitude A.

The switching frequency F is simply calculated as the total number of switches that were observed in a video. The lower the value this metric has, the better the quality of the corresponding video is.

Finally, equation 4.2 which is based on the work of Yin et al.[45], expresses the switch amplitude A as the normalized sum of all the amplitudes of representation switches between consecutive segments r_k and r_{k+1}. Again, A is analogous to the degradation of QoE since large representation changes which lead to poor QoE will return higher values of A.

$$A = \frac{1}{K-1} \sum_{k=1}^{K-1} |r_{k+1} - r_k| \tag{4.2}$$

The two metrics are then combined to a single indicator of the representation variation Var using linear combination. Next, each instance in the dataset is classified in one of three main categories, no variation, mild variation and high variation, based on the value of Var.

Change Detection

During the study of the sessions with many representation changes, we observe that whenever the adaptive algorithm enforces a change in the representation of the video, a new start-up phase is initiated for the new representation. During this phase, the size and inter-arrival times of the segments are reduced significantly until a certain threshold in the play-out buffer has been reached and the video download returns to the steady phase.

In the video session in Figure 4.3.3, we can see there is a steady state in terms of size and inter-arrival times for the first quality. When the representation switch occurs however, the chunk time delta and size delta are gradually increasing until a steady state is reached again.

Therefore, for the purpose of more accurately capturing the representation changes we use the two features that were used in section 4.3.2, the segment size delta $\Delta size$ and segment time delta Δt.

The most suitable approach to detect representation changes, is to perform a time-series analysis. This method allows the identification of abrupt changes in the values of different metrics in the dimension of time that are correlated with the switches of representations.

In more detail, our analysis of video sessions with quality switches showed that whenever a change in resolution takes place, a new start-up phase is initiated in order to fill the buffer with data from the new representation as fast as possible. This phase is characterized by video segments with small sizes and small inter-arrival times which will increase gradually until the steady state is reached once again.

We find that the metric which better captures the changes in both the size and the inter-arrival of the video segments, is the product $\Delta size \times \Delta t$. Specifically, the multiplication of the two parameters will combine but at the same time emphasize the effects of each one. Therefore, for each video session in the dataset, we calculate a new time series where each point corresponds to the aforementioned product.

While there are many tools and algorithms for detecting abrupt changes in a time series, we find that the most suitable for the purposes of this work is the Cumulative Sum Control Chart (CUSUM) which was developed by E.S. Page [46].

CUSUM is a change detection monitoring technique which allows the detection of shifts from the mean of a given sample of points in a time series. When a point exceeds an upper or lower threshold then a change is found. In our case, instead of thresholds we use the standard deviation of the output of the change detection algorithm. The standard deviation is capable of capturing the magnitude of the changes that occurred and is an indicator of high variance.

Figure 4.3.4, shows the distributions of the standard deviation of the change detection output for sessions with and without variance. We observe that there is significant separation between the two distributions and by defining a threshold at value 500 on the horizontal axis, we are capable of correctly identifying 78% of the sessions without variance and 76% of those that have representation variations.

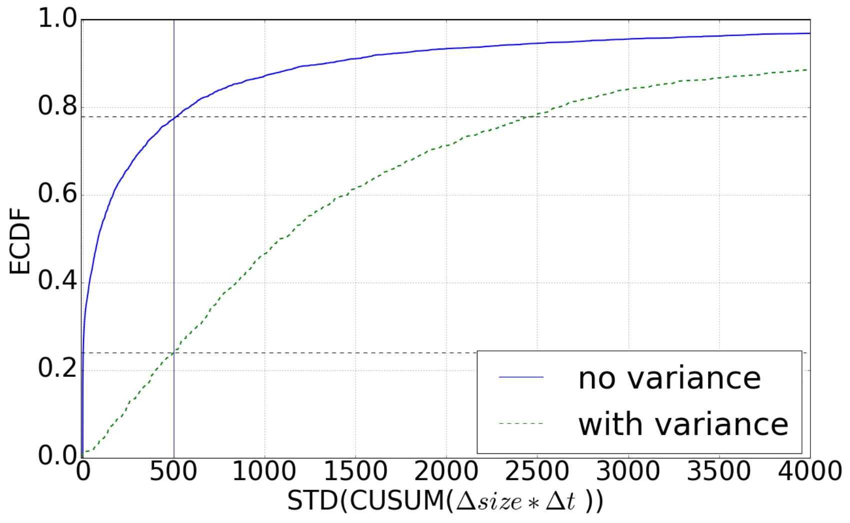


Figure 4.3.4: CDF of change detection output for videos with and without resolution changes.

Apart from the time-series analysis, ML was also considered to develop a model for the detection of representation switches. However, it did not perform as well as the proposed methodology did and for this reason that approach was not considered.

4.4 Evaluation with Encrypted Traffic

In this section we present and discuss the findings from the evaluation of the models that were developed in Section 4.3 with encrypted data. This step is important for verifying that the proposed methodology can perform with similar accuracy when dealing with encrypted traffic.

4.4.1 Ground Truth

For the collection of the encrypted traffic, we developed an Android application which is responsible for automatically launching YouTube videos which are randomly selected from the list of the 100 most popular videos on the website [47]. All videos are played using the latest version of the stock YouTube app for Android, where encryption is enabled by default.

Apart from handling the playback of videos, the app has also the capability to extract performance measurements related to the video that is being played. In more detail, by accessing the device's log, it can identify and log the playback status of a video, i.e. if the playback has started, paused, stopped or if a stall has occurred. Therefore, we do not only detect if the video was watched throughout its full length or abandoned earlier, but also identify any stalling events and their duration. This information is used as the ground truth for labeling the data and evaluating the accuracy of the stall detection model.

In order to capture the ground truth related to the representation quality switches we need access to the metadata in the HTTP requests that are responsible for the download of the individual video chunks. However, these requests are encrypted by default by the YouTube application and the required information cannot be captured by means of traffic monitoring.

Although solutions such as Man-in-the-middle (MITM) proxies are common in such use cases for decrypting the traffic generated by the device, we believe that they are not practical since they alter the path between the client and the server, but also change the encryption scheme by establishing two separate TLS connections instead of one.

To make sure that the ground truth for the quality switches is obtained without tampering with the encryption scheme or the traffic between the player and the content server, we reverse engineer the YouTube application and pinpoint the method which is responsible for constructing and performing HTTP requests. Our application then 'hooks' each invocation of this method and extracts its result, which in this case is the full URL of the HTTP request. The URL is then parsed to extract the required ground truth.

Finally, our app will periodically aggregate and send the collected information from the videos to a remote server. The local copy of this information is then deleted from the device to free up space.

4.4.2 Dataset

Next, the app was installed on a Samsung Galaxy S2 device with a SIM card with an unlimited 3G data plan. The instrumented phone was given to a user who was instructed to carry it at all times for a period of 25 days. The user was motivated to launch the application when moving to increase the probability of QoE issues.

As a result, we generated a dataset for the ground truth and a dataset from the encrypted traffic corresponding to 722 video sessions. Each entry in the ground truth dataset corresponds to a unique segment and the video session ID which the segment belongs to, the timestamp that marks the beginning of the chunk download, a field to indicate if it is an audio or video segment, the total number and duration of the stalls observed in the session and finally its quality representation.

The encrypted traffic data is collected again from the proxy in the form of weblogs. However, since the flows are encrypted, information such as the session ID, the stall characteristics and the quality level of each chunk are not available. Therefore, we only extract the timestamp of the HTTP request, the server IP address and port, the size of the requested object and the TCP statistics which were described in detail in Section 4.2.1.

Although the session ID is available in the ground truth dataset and it is used to group the video segment statistics in unique sessions, this parameter is missing from the encrypted data. Even so, we find that it is possible to identify the encrypted segments that belong to the same session and group them together.

To achieve this we go through the following steps:

- Identify the traffic that corresponds to a single subscriber and remove all requests that do not belong to YouTube by filtering out those that have domain names not related to the service.

- Next, we look for the unique HTTP traffic patterns that take place at the beginning of a new video session but also after the completion of the playback. These include requests to m.youtube.com and i.ytimg.com which are responsible for downloading multiple web objects such as HTML, scripts and images to construct the video's web page.

- Longer periods without traffic that correspond to the time between consecutive sessions are identified in order to clearly define the beginning and ending of each session.

This methodology has high accuracy as it successfully identified the vast majority of the sessions that were launched during the entire period of the measurements. However, it can be limited in scenarios were the same subscriber launches multiple videos in parallel and not sequentially. Although such cases are quite rare, it can be challenging to identify the segments that belong to the same video session.

Then the two datasets can be easily joined by matching the respective timestamps and the chunk count per session. As a result, the final dataset contains the same metrics that were described in the left column of Table 4.2.1. Having the exact same set of features in both datasets is necessary to allow the evaluation of the trained models that were created in the previous section with the new data from the encrypted traffic.

4.4.3 Dataset Comparison

In this section we characterize the two datasets and make a comparison of the key features. This will help verify that the encrypted YouTube service behaves similarly to the unecrypted and the model built for plain traffic works for encrypted traffic as well.

More specifically, in Figure 4.4.1 we present the distributions of the segment size (left) for encrypted and clear-text. The right figure shows the comparison between the two distributions for the segment inter-arrival times.

In the case of the segment size, there is a significant overlap between the two distributions. This indicates that there is a common pattern with respect to the downloaded chunk sizes of the videos in both datasets which can be translated to videos streamed with similar qualities. Only 10% of the segments were larger than 1MB which can be found in HD videos, while the majority of the segment sizes are concentrated at or below 500KB which corresponds to SD video quality.

The distributions for the segment inter-arrival times also have very common characteristics. However, 60% of the encrypted chunks have slightly lower values in comparison with the respective unencrypted data. The shorter times between chunks are indicative of lower bandwidth availability that results in faster depletion of the playout buffer and a more frequent request of new segments. This observation is expected since a large part of the encrypted videos was downloaded while the user was commuting where network conditions can significantly deteriorate.

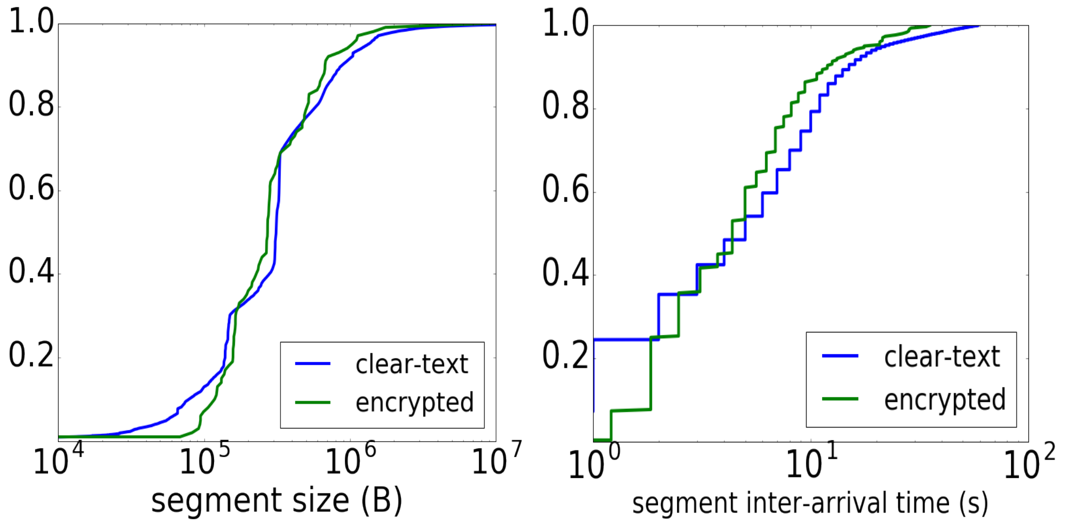


Figure 4.4.1: CDF of the segment size (left) and segment inter-arrival time (right) for encrypted and unencrypted traffic.

4.4.4 Stall Detection

Before evaluating the model for detecting stalls, we repeat the feature construction process described in Section 4.3.1. However, an automated feature selection like the one employed in the previous section is no longer necessary since we already know the important features that are required to make predictions and the rest are safely removed. Next, the trained model from Section 4.3.1 is directly tested with encrypted traffic.

The resulting accuracy is 91.8% which corresponds to only 1.7% lower performance than the evaluation with unencrypted data. Nevertheless, this is still an excellent result which demonstrates that the training set that we used created a very accurate model that can be applied to encrypted traffic with equal success.

Table 4.4.1 shows the evaluation results in terms of Precision and Recall and Table 4.4.2 the corresponding confusion matrix. Here we can see that the performance has improved for the videos without stalls, it remained roughly the same for sessions affected by mild stalling but has decreased for the case of videos with severe stalls.

Class	**TP Rate**	**FP Rate**	**Precision**	**Recall**
no stalls	0.97	0.19	0.96	0.97
mild stalls	0.75	0.04	0.79	0.75
severe stalls	0.64	0.02	0.6	0.54
weighted avg.	0.92	0.16	0.92	0.92

Table 4.4.1: Classifier's output for the stall detection evaluation

original label	**predicted label**		
	no stalls	mild stalls	severe stalls
no stalls	97.2%	2.5%	0.3%
mild stalls	18.6%	75.2%	6.2%
severe stalls	2%	32.4%	65.6%

Table 4.4.2: Stall detection confusion matrix

The detection of non-problematic videos is done with higher accuracy than the one observed in Section 4.3 because there is smaller diversity in the network conditions where the healthy sessions occur. This is attributed to the fact that the majority of these sessions are generated when the user is static either at the office or at home, where the network conditions have a constant performance and as a result, the classifier can more easily identify that these sessions did not have any issues.

The main source of the overall accuracy loss in this evaluation however, is the class of videos with sever stalls. From the confusion matrix it is apparent that this is a result of the increased number of videos with severe stalls that were falsely detected as mild stalls. This is a problem that was also observed to a lesser extent in the training and evaluation with the unencrypted dataset (Section 4.3.1).

Although the low performance for the severe stalls class is attributed to the same reasons that were described in the previous section, the further decrease in accuracy originates from the fact that in the new dataset most of the sessions with severe stalls have a Rebuffering Ratio slightly higher than 0.1. Remember that 0.1 is the borderline that was defined to separate sessions with mild and severe stalls. Therefore, it becomes more difficult for the classifier to distinguish to which class these videos belong to.

4.4.5 Average Representation Detection

The evaluation of the second model for the detection of the average representation is done following the same process as previously. The extended set of features is generated by means of feature construction, followed by the manual removal of the features which do not contribute to the model. This results in the same 15 parameters that were presented in Table 4.3.4.

The evaluation is performed with the same approach as previously, where the encrypted dataset is used as a test set for the trained model. The process returns an overall accuracy equal to 81.9% which is approximately 2.5% less than the respective result we got when using the unencrypted dataset in Section 4.3.2. Again, this is an overall good indicator that the model can perform the detection with almost equally good accuracy when dealing with encrypted traffic.

In Tables 4.4.3 and 4.4.4, we can see more details regarding the performance of the evaluation per label. Specifically, although the detection of LD and SD videos is done with slightly reduced accuracy, we still get satisfactory performance as we can see from the Precision and Recall values. If we look at the confusion matrix below however, we observe that there is an increase in the LD videos which were misclassified as SD. This is attributed to the fact that in the current dataset the number of 240p videos in the LD category is significantly higher than the 144p. This causes a shift in the distribution of the average quality for this category toward the higher end, which in turn causes the incorrect classification of a percentage of these videos as SD.

Another reason behind the reduction of the accuracy is the reduced detection capabilities for the HD videos. In this case, the Precision and Recall for this class have both reduced significantly. At the same time, from the confusion matrix we see that a significant amount of videos have been incorrectly identified as SD quality. This poor performance is a result of the very small number of videos that are available in the HD class. When combined with the also relatively small number of HD videos that were used to train the model, this results in a class where the training and testing was done with small number of samples and therefore reduced detection capabilities for this class.

This problem can be easily alleviated by introducing a training set that is much richer in HD videos. This will allow the creation of a predictive model which will be based on a more diverse dataset that will be capable of a more accurate detection of the average quality of HD videos with different characteristics.

Class	**TP Rate**	**FP Rate**	**Precision**	**Recall**
LD	0.845	0.203	0.853	0.845
SD	0.789	0.157	0.775	0.789
HD	0.513	0.003	0.641	0.513
weighted avg.	0.819	0.183	0.819	0.819

Table 4.4.3: Accuracies from the evaluation for the average representation detection

original label	**predicted label**		
	LD	SD	HD
LD	84.5%	15.4%	0.1%
SD	20.4%	78.9%	0.7%
HD	15%	33.75%	51.25%

Table 4.4.4: The confusion matrix from the average representation evaluation

4.4.6 Representation Quality Switch Detection

The last phase of the evaluation is done for detecting quality switches. In this case, there is no trained model that can be directly applied to the encrypted data. In contrast, the methodology relies on the detection of changes that happen in the time intervals between segment downloads and the difference in size between consecutive segments.

In this evaluation there is no requirement for feature construction or feature selection. We only need to calculate the time series of the products $\Delta size \times \Delta t$ for each video in the dataset which is going to be used as input for the change detection algorithm. Next, we apply the change detection on each session and from that we take the standard deviation.

In order to validate the methodology from Section 4.3.3, we use the same value that was proposed in that section as a threshold for the standard deviation of the change detection output.

$$STD(CUSUM(\Delta size \times \Delta t)) = 500 \tag{4.3}$$

According to the proposed methodology, all sessions below the threshold should represent approximately 78% of the sessions without quality switches and the sessions above the threshold should represent 76% of the sessions with quality switches (Figure 4.3.4).

Next, the dataset is split into two parts, i.e. the sessions with score below the threshold and those with a score above it. From the ground truth from the encrypted data, we are able to evaluate if the predefined threshold allows the detection of variance with accuracy equal to the one demonstrated in Section 4.3.3.

Our analysis reveals that the first part of the dataset consists of 76.9% of videos without any quality change, while in the second part we find 71.7% of the sessions with quality switches. These accuracies are lower by 1.1% and 4.3% respectively as compared to the results from the evaluation with unencrypted data.

The decrease in accuracy for detecting videos with quality switches indicates that the encrypted data consists of videos where the average quality variance is smaller than the one that was observed in the previous section. As a result, the distribution of (4.3) shifted towards the smaller values and after the threshold was applied, lower percentage of problematic sessions was correctly identified.

4.5 Limitations

The methodology presented in this chapter was developed using information from YouTube video sessions that were streamed with the service's current configuration. However, the predictive power of the models responsible for detecting QoE impairments can be limited in the case YouTube changes its video delivery scheme. In such a scenario, the models that were affected by the changes need to be trained and evaluated again with an updated dataset.

Moreover, we do not study the evaluation of the methodology with other video streaming services in order to verify to what extent this approach can be generalized. However, our analysis of other popular video streaming services such as Vevo, Vimeo, Dailymotion and so on, has revealed that they have adopted the same technologies that YouTube is using for content delivery such as adaptive streaming, rate limiting, wide range of codecs and qualities and HTML5-based playback. This common set of characteristics is a strong indicator that our methodology can be generalized to a number of other streaming services and motivates us to include it in the future steps of this work.

4.6 Chapter Summary

In this chapter we presented a novel framework for detecting from encrypted traffic the 3 key factors that impact both adaptive and classical video streaming QoE, i.e. stalls, average quality and quality switching.

Next, we demonstrated through evaluations on encrypted and unencrypted traffic, that the proposed models can detect different levels of impairments with accuracies as high as 93.5%.

One of the main findings of the work is that the changes in size and inter-arrival times of video segments are among the most important indicators of quality impairments. The incorporation of these features in our detection framework resulted in significant improvements in accuracy.

We showed that the framework can perform very well on a real production network using a few key performance metrics from a single vantage point and without the requirement of instrumented clients or additional vantage points, so it can easily be deployed by network operators. The trained models can be then directly applied on the passively monitored traffic and report issues in real time.

5
Identifying the Root Cause of Video Streaming Issues

After having established working methodologies for monitoring the video quality from clear-text and encrypted traffic in Chapters 3 and 4, in this Chapter we investigate how to identify the underlying faults that lead to video QoE issues and how to locate the offending part of the network. Towards this end, we develop a framework for diagnosing the root cause of mobile video QoE issues with the aid of machine learning. Our solution can take advantage of information collected at multiple vantage points between the video server and the mobile device to pinpoint the source of the problem. Moreover, our design works for different video types (e.g., bitrate, duration, ...) and contexts (e.g., wireless technology, encryption, ...) After training the system with a series of simulated faults in the lab, we analyzed the performance of each vantage point separately and when combined, in controlled and real world deployments. In both cases we find that the involved entities can independently detect QoE issues and that only a few vantage points are required to identify a problem's location and nature.

5.1 Approach and Challenges

Approach - Machine learning has been widely used to help solve complex problems, including fault diagnosis and analyzing the QoE of video, so it is a natural starting point for our work. However, ML alone is not sufficient. For example, based on data collected on the mobile phone, it may be possible to learn that poor QoE is caused by "low bandwidth", but it will not be possible to identify what network is at fault, e.g., the wireless link versus the Internet backbone.

To help both identify and pinpoint failures that may cause QoE issues during playback, we propose to place measurement probes at multiple vantage points (VPs) along the path. Collecting data at multiple points will provide a system-wide view that can help us isolate performance metrics for different segments and devices. In an ideal world, we would be able to collect measurements on every device, but this is not practical so we focus on a scenario with three VPs, the two endpoints and the wireless AP.

Challenges - While there has been some research on diagnosing video QoE, as discussed in Section 8, we are not aware of any work that uses a combination of network and hardware metrics collected across multiple VPs. This novel approach introduces a number of challenges.

First, the amount of data that can be collected across the vantage points is overwhelming. We use feature construction and selection to identify the most useful features (Section 5.2.2).

Second, providers offer video with different quality, encoding, and duration, and also use a variety of video delivery mechanisms such as static or adaptive streaming, pacing and so on. Hence, the system needs to be agnostic to the details of both the video itself but also how it is delivered. Our solution to this problem is to normalize the metrics we collect (Section 5.3).

Third, while we found that a multiple vantage point solution is very effective, it may not always be possible to obtain data from all vantage points of interest for reasons such as security or privacy concerns. To address this challenge, we designed our system so it can also diagnose problems, albeit with lower accuracy, if it lacks data from some vantage points. We also identify for each vantage point how much its measurements contribute to the diagnosis (Section 5.4).

Finally, there is a growing demand for encrypted content delivery, even for video. This poses a challenge for systems that rely on packet inspection or HTTP traffic analysis. Our design avoids such analysis so it can be compatible with encrypted traffic. We similarly avoid dependencies on any particular wireless technology to ensure wide applicability.

5.2 System Design

The proposed framework consists of one or more *probes* along the data delivery path that provide a number of performance metrics, and a QoE *detection system* that constructs the necessary features from these metrics and applies machine learning algorithms in order to extract the root cause.

5.2.1 Probes and Metrics

Ideally, each network device along the data path may provide performance indicators. However in practice this is not feasible as it involves the cooperation of too many parties and hardware vendors. Therefore, in our approach we focus on three key vantage points: *i) the mobile device, ii) the connection gateway (e.g., wireless router) and iii) the content server*. These three points can capture issues at the boundaries of each of the three segments in the video delivery path: the user, the ISP and the content provider.

The probes collect performance metrics from all relevant layers, i.e. application, hardware, transport and link layers, in order to capture the related faults which are described in detail in Section 2.3.

Application layer: At the mobile probe, we capture statistics concerning the QoE of video playback from the mobile OS irrespectively of the video application (our implementation is done on Android). These metrics include the video startup delay, video stalls, frame skips, the status of the buffer, video bit-rates, etc. These are used to construct an estimated *Mean Opinion Score* ([6]) that represents the QoE ground-truth. Notice that while these metrics can indicate the existence of poor QoE, we are not including them as features in the classifier, i.e., they are only used to provide the labeled QoE ground-truth.

OS/Hardware Layer: The hardware metrics provide information about the available resources and the connectivity state at each of the three VPs. For that purpose, we monitor the percentage of load, CPU utilization, the amount of free system memory and so on. At the end of a video flow, aggregated information about each feature is returned (e.g., average, minimum, maximum, standard deviation of CPU usage).

Transport layer: A set of 113 network metrics are collected per flow, including RTT, number of packets, flow duration, window size, out-of-order and re-transmitted packets, etc. These metrics are collected on all probes for each of network interface using `tstat` [48]. Extensive documentation of these metrics can be found in [49].

Link/Physical layer: For each of the available network interfaces (NICs) the probes extract information about the utilization, bandwidth, and dropped or retransmitted packets. In addition, for wireless links (WiFi/3G), the radio technology, the advertised rate and signal strength information (RSSI) for each of the connected devices is monitored.

Similarly to the OS/hardware metrics, an aggregated set is calculated for the conditions of each NIC during a video flow. For instance, the average/minimum RSSI or the number of disconnections/handovers during the flow is returned.

Our proposed multi-VP approach enables each entity with a deployed probe to diagnose problems within its own proximity, separately without requiring information from other contributors. This way, providers or users are not limited by common privacy concerns or collaboration restrictions. However, combining information from all three entities can improve root cause analysis accuracy.

5.2.2 Detection System

Our system uses machine learning to learn the correlations between performance and QoE metrics and to create a model for detecting and characterizing the root cause of playback problems. Before applying the ML tools, we employ two techniques, Feature Construction (FC) and Feature Selection (FS) that help improve the classifier's performance.

Feature Construction aims in making the system more agnostic to the specifics of each scenario, i.e., video type, streaming techniques and network technology in our case. With this method, our objective is to make the prediction model as generalizable as possible so that it can be successfully applied for different devices, video players and video services but also for different network conditions.

To achieve this task, we normalize the features that depend on any of the aforementioned variables. Specifically, we normalize all the parameters which are expressed in bytes or packets with the respective total number of bytes or packets of the entire session. The list of normalized features includes among others, the number of data packets, data bytes, re-transmitted packets, re-transmitted bytes and out of order packets [49]. The same approach is applied for the video duration which is normalized with the total duration of the video session.

Furthermore, we calculate the uplink and downlink utilization of each device's NIC by dividing the average transfer rate of a video session by the maximum transfer rate observed for this NIC in the entire dataset. In this way, the utilization takes values between zero and one.

The RSSI is collected in one second intervals and then the average, maximum and minimum values are calculated for the entire session. For our analysis we keep the average value only as we observed that it has better predictive capabilities as compared to the maximum and minimum.

Feature Selection: To increase the performance of the algorithm in terms of both accuracy and execution time, it is important to significantly reduce the feature space size. The reduction of the number of features used to train the algorithm, minimizes the over-fitting problem that is either caused by multiple features with little or no predictive power, or by features that contribute the same information to the prediction. After experimenting with different FS algorithms, we find that the Fast Correlation-Based Filter algorithm is the most efficient in identifying a minimal set of features with high predictive power.

After applying FS, the number of features is reduced from 354 to 22 (Table 5.2.1). Among the remaining features, those with higher weights were the utilization of the interfaces, the 3 hardware metrics from the mobile device: the free memory, the CPU utilization and the RSSI. In section 5.4.4 we discuss how much each of these features contributes to identify-

mobile CPU utilization	mobile bytes retransmitted
mobile free memory	router out-of-order pkts
mobile RSSI	server avg RTT
mobile downlink utilization	mobile first packet arrival
router downlink utilization	router first packet arrival
server uplink utilization	server max window size
mobile pkts retransmitted	mobile min MSS
server min MSS	mobile max RTT
server video data pkts	router video data pkts
mobile max window size	router reordered pkts
router avg RTT	router max RTT

Table 5.2.1: Features after Feature Selection.

ing individual problems and the improvements resulting from both FS and FC.

Machine Learning: For the data processing and analysis we use version 3.6.10 of Weka. Our classifier of choice for the data analysis is J48 which is an implementation of the popular C4.5 algorithm. The training and testing of the algorithm is performed using the 10-fold cross-validation method.

C4.5 and Decision Trees in general, are known to perform well with noisy data. Therefore, they are a suitable solution for building our predictive model since we intent to train and test it on network data where noise is induced by background variations. We further discuss background variations in Section 5.3.2.

Decision Trees outperformed other algorithms like Naive Bayes and Support Vector Machines which we also evaluated with our datasets. Given that the datasets from our experiments consist of a large number of features that often have a non-linear relation between them, decision trees are well suited for our predictive model since their performance is not affected by such non-linear relations, while their hierarchical structure fits well with our troubleshooting approach.

Moreover, data collected from real networks can be noisy due to background variations generated by multiple sources. For that reason, Decision Trees are a good solution since they cope well with noisy data.

Finally, another advantage of using C4.5 is that contrary to algorithms such as SVMs, the model is not a black box. The constructed tree can be visualized and interpreted. This can greatly simplify and improve the feature selection process and help optimize the performance of the model.

5.3 Collecting the Ground Truth

In order to build and train our ML model, we need to collect the ground-truth: a set of labeled good and problematic video instances with a known root-cause. This data set will be used to train our model and also for controlled experiments. Finally, we will use our model in real-world experiments to evaluate how well it can cope with the added noise and complexity of the Internet. To achieve this, we implemented a testbed infrastructure with four components i) a realistic hardware setup with multiple simulated mobile devices and backbone connections, ii) background workloads for generating constant variations, iii) induced impairments that will simulate a specific scenario/label and iv) associate mean opinion score to the collected measurements. To make the generated model as realistic as possible, the settings of each component (e.g., loss rate, link speeds, load, etc) is based on distributions that were derived from traces that were acquired from a network within a large European ISP.

5.3.1 Setup

Simulated Problem	**Tools/Method**	**Settings/Comments**
LAN Shaping	`tc`, `netem`	LAN: BW cap=1-70Mb/s, 1ms delay, 0% loss
WAN Sahping	`tc`, `netem`	DSL: BW cap=7.8Mb/s, 50ms delay, 0.75% loss Mobile: BW cap=5.22Mb/s, 100ms delay, 1.4% loss
LAN Congestion	`iperf`	UDP traffic from wired client to the router
WAN Congestion	`iperf`	UDP traffic from wired client to the server
Mobile Load	`stress`	CPU, memory, IO, disk workloads
Poor Signal Reception	distance, attenuation	Reduced SNR and data rate
WiFi Interference	external interference source	Transmission on the same frequency

Table 5.3.1: List of simulated problems, used tools and configurations

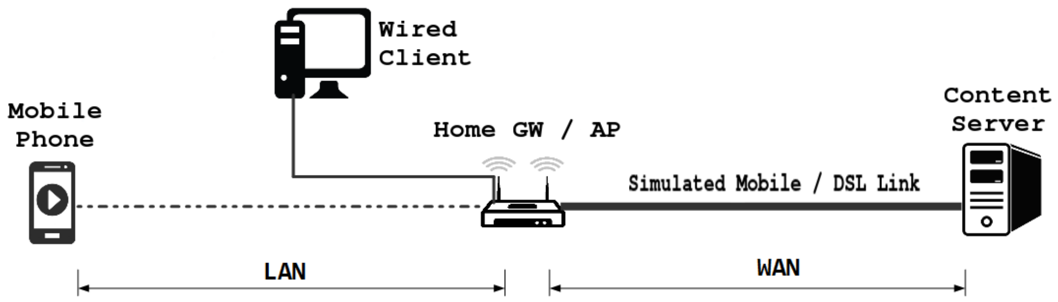


Figure 5.3.1: Device setup in the testbed.

We set-up a simple testbed with a video server, a router/AP (Access Point) and three different Android devices. The phones are connected to the Wireless LAN of the AP and the server is connected via an Ethernet cable to the router. In addition to the devices that are necessary to deliver the video, other wired and wireless devices are available in order to generate background traffic on the network segments and interference on the wireless link (Figure 5.3.1).

We use an Apache server to deliver the video. We downloaded the videos from the top 100 most viewed list [50] from YouTube to the server in either Standard or High Definition to ensure the diversity of the video collection. A Netgear WNDR3800 running OpenWRT was used as a router/AP. It was configured to operate at 5 GHz after verifying that there are no surrounding sources of interference. Three types of Android devices were used as mobile clients: Samsung Galaxy S II, Nexus S, and Nexus 5. The devices are instrumented with our developed application, which is responsible for performing HTTP video requests to the server and opening the returned video stream using the default Android media player.

	BW	delay	loss
DSL	7.8Mbit/s	50±20ms	0.75±0.5%
Mobile	5.22Mbit/s	100±30ms	1.4±1%

Table 5.3.2: Configuration of simulated links

In order to make the generated model as realistic as possible, `tc` and `netem` are used to simulate a DSL and a mobile link with the settings shown in Table 5.3.2. The delay and loss for both configurations follow a normal distribution within the indicated ranges. As mentioned in the previous part, these settings were obtained by analyzing traces from a real deployment in a large ISP.

5.3.2 Background Variations

To recreate realistic network conditions, we introduce synthetic competing traffic workloads of different patterns. These background variations are based on real world network traces and will aid in training the algorithm for successful deployment in the real world. This is done using the D-ITG generator [51], which supports traffic generation based on different applications such as Telnet, FTP, gaming, VoIP and more. We also use ApacheBench to create a realistic load on the server.

5.3.3 Simulated Problems

In order to generate the dataset that contains various levels of QoE, we iterate through a set of scenarios in which we stream a randomly picked video and artificially induce a problem with varied intensity. Apart from background variations, we use problems in three categories: networking, device hardware and wireless medium issues. The list of simulated problems, the used methodology, and the specific configurations can be found in Table 5.3.1.

Shaping and Congestion. To simulate LAN congestion, we use multiple `iperf` instances to transmit UDP traffic between the wired LAN client and the router; for WAN congestion we generate traffic with the same method but between the server and the router.

For traffic shaping, different bandwidth, delay and loss restrictions are applied to the corresponding link. The LAN is shaped based on the data rates offered by common 802.11 standards such as a, b, g and n that are capable of providing rates per stream ranging from 1 up to 70Mbit/s. For the WAN shaping we set different restrictions for mobile and DSL connections (Table 5.3.1).

Mobile Load. This category examines cases where the high load on the device hardware does not allow the proper decoding and playback of the video. The load simulation is performed with the workload generator tool `stress` that allows CPU, I/O, memory and disk workload generation.

Poor Wireless Signal Reception. We simulate poor signal reception by placing the phone far from the AP and by attenuating the transmitted signal at the AP. As a result, there is degradation in the wireless link's SNR and the available data rate.

WiFi Interference. This scenario, involves creating interference on the wireless channel from external sources. In real use cases, interference can be caused by near-by devices transmitting or receiving on the same frequency range. In our experiments interference is created by generating large traffic workloads on an adjacent second WLAN operating on the same channel as the AP we use for measurements.

5.3.4 MOS-based Labeling

Before performing the analysis, the instances in the dataset need to be labeled with the QoE ground truth so they can be used for training and evaluation of the classifier. QoE labeling has to express the quality of the video session in terms of user satisfaction so that we can correlate problematic videos with the QoE.

For that purpose, we convert application performance metrics such as startup delay and the frequency and duration of stalls to Mean Opinion Score (MOS) ratings based on the work of Mok et al. [3] who derived an equation for calculating the MOS from performance metrics by means of regression analysis. Based on the obtained scores, we label instances with MOS greater than 3 as 'good', instances with scores between 2 and 3 as 'mild' and those with MOS lower than 2 are labeled as 'severe'.

For the detection of the location of the problem, we create six new labels based on the combination of the segment that the issue occurs and its severity. For the evaluation of the algorithm when detecting the exact problem, we label problematic instances according to the type of the fault.

5.4 Evaluation

In this section, we evaluate the system's performance in the controlled environment described in Section 5.3 for detecting the existence of problems, detecting the problem's location and for identifying the exact problem. Later we will examine if the resulting model is robust enough to detect problems in the real deployment.

The training and testing of the algorithm in all the evaluation scenarios is performed using 10-fold cross-validation. We present the system's performance in terms of overall accuracy, defined as the percentage of correctly predicted instances, i.e., the number of True Positives (TP) and True Negatives (TN) over the total number of instances. In addition, we also use the Precision and Recall metrics. *Precision* is expressed by the ratio of TP over TP and False Positives (FP) and represents the accuracy a certain class is predicted. *Recall* is the ratio of TP divided by the total instances in this class and it measures the classifier's ability to correctly identify the desired classes from the data set. In simple terms, for a root-cause *c* (e.g., low RSSI), high precision means that the framework did not miss-classify other problems as *c*, while high recall means that it found most of the instances that exhibited *c* and, therefore, has a high probability of detecting this issue.

The collected dataset consists of 354 metrics including network metrics, the total number of rebuffering events, device CPU and memory utilization and the RSSI. Note that the rebuffering events are only used for labeling the instances and not as a feature. Overall, there are 3919 instances in total out of which 3125 are labeled as good, 450 as mild and 344 as severe.

5.4.1 Who Can Detect the Existence of a Problem?

First, we examine which of the VPs (or which combination of them) is performing better when identifying the *existence* of a problematic video flow. For that reason, we aggregate all labels into three categories: 'good', 'mild' or 'severe', as discussed in section 5.3.

As observed in Fig 5.4.1, each one of the vantage points can independently discover problematic sessions with similar accuracy: for the mobile it is 88.1%, for the router 86.4% and for the server 85.6%. Finally, when combining the measurements from all the vantage points, the performance slightly improves to 88.8%. We observe that the mobile phone achieves performance that is as good as the combination of all three vantage points as it is in the position to measure both local (e.g., CPU/RSSI) and remote (e.g., server load, network) issues.

Moreover, although the other two VPs achieve more than 85% accuracy in detecting good instances, they have significant problems to discern between mild and severe problems. In more detail, the system's poor performance for mild problem detection is correlated to the high number of false negatives where the problems are identified as severe and the false negatives where they are labeled as healthy.

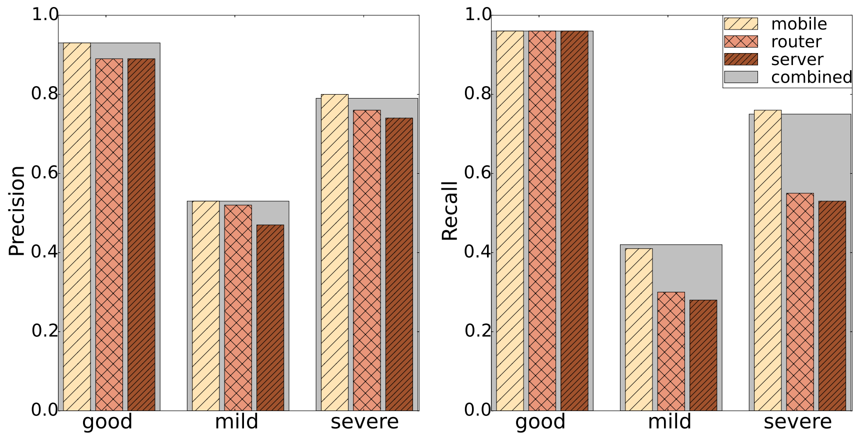


Figure 5.4.1: Precision (left) and Recall (right) for problem detection in controlled experiments.

Takeaway: The existence of healthy sessions can be identified with high accuracy from each entity independently. ISPs and content providers can identify that there was a problem but they cannot be certain about its severity in terms of impact on the users' QoE. Moreover, we find that instrumentation closer to the mobile terminals where the majority of the problems occur yields higher performance.

5.4.2 Who Can Detect a Problem's Location?

Apart from the existence of a problem, it is important for each entity to understand if the fault is within their network or who is to blame when there is an issue. For that reason, we aggregate labels into three categories based on the *location* of the problem: mobile device, LAN and WAN.

What is interesting however, is the ability of the server VP to localize problems in the LAN segment. Specifically, the server shows almost equal performance to the router VP for detecting LAN problems. To better understand why this is the case, we inspect the features that contribute most to detecting LAN problems. We find that for both VPs the same features, RTT, first packet arrival delay and the number of retransmissions, are ranked highest for LAN problem detection.

We also evaluated the benefits of using VP pairs for location detection. However, we did not observe any significant improvement in accuracy nor any intriguing result.

Takeaway: Content providers who deploy such a system have the ability to identify if a problem has occurred on the ISP's network. This information is useful to content providers for spotting congested or under-provisioned ISP networks and pursue better peering agreements with ISPs in order to minimize bottlenecks. ISPs can also identify whether the issue has originated within their own network or the user's LAN if the home router is instrumented.

Finally, an instrumented application or an instrumented phone can provide valuable information to the users to identify whether their home network, their ISP or the content provider is to blame for poor QoE and it can significantly improve the accuracy of the other entities if the measurements are shared.

5.4.3 Who Can Detect the Exact Problem?

Next, we trained and evaluated the algorithm using all the labels of problematic scenarios that are available in our dataset, allowing us to assess the accuracy with which the classifier can detect the *exact root cause* behind the problem experienced by the user. The overall accuracy for detecting the exact problem, is 88.18% for the mobile VP, 85.74% for the router, 84.2% for the server and 88.95% for all three VPs. These numbers demonstrate the system's high performance when carrying out the task of identifying the root cause behind video QoE issues.

However, while the overall accuracy is good, we observe that certain vantage points exhibit difficulties in discerning certain problems. Figure 5.4.2 shows the different accuracy with which each issue is predicted, while Table 5.4.1 provides insights about the 3 metrics with the highest prediction power for each label (notice that there are cases where only two or even only one metric make significant contribution).

	WAN **CONGESTED**	**WAN** **SHAPED**	**LAN** **CONGESTED**	**LAN** **SHAPED**	**MOBILE** **LOAD**	**LOW** **RSSI**	**WIFI** **IN/FERENCE**
M	1st pkt arrival M RTT M out-of-ord. pkts	1st pkt arrival M out-of-ord. pkts	M RTT M Util. M Bytes re-TX	1st pkt arrival M Util. M RTT	CPU MEM M RTT	RSSI M RTT MEM	RSSI M pkts re-TX M Util.
R	R RTT R 1st pkt arrival R Util.	R out-of-ord. pkts R RTT R 1st pkt arrival	R RTT R 1st pkt arrival R Util.	R RTT R 1st pkt arrival R pkts reordered	R RTT	R 1st pkt arrival R out-of-ord. pkts R RTT	R RTT R 1st pkt arrival
S	S Util. S data pkts S RTT	S Util. S data pkts S win size	S data pkts S Util. S RTT	S data pkts S Util. S win size	S RTT	S data pkts S RTT	S RTT S Util.
C	R Util. S Util. M RTT	S pkts re-TX S Util. R RTT	R Util. R 1st pkt arrival M RTT	M RTT R Util. M 1st pkt arrival	CPU MEM M RTT	RSSI CPU MEM	RSSI M pkts re-TX

Table 5.4.1: Feature ranking for exact problem detection (M=mobile, R=router, S=server, C=combined)

Furthermore, we observe the high accuracy with which WiFi interference and low RSSI related problems are predicted. From the figure it is clear that all the VPs in the system perform very well when detecting sessions which suffer from severe problems in the wireless medium.

However, the detection of mild interference from the router and the server is done with much lower accuracy. Given that these two VPs don't have RSSI information, they are unable to distinguish the small variations caused by mild interference. As a result, we observe from the classifier's output that a large number of mild interference instances are predicted as healthy witch causes the particular label to be predicted with lower accuracy.

More information that help understand this behavior can be found in Table 5.4.1, where for the router and the server, the highest ranked features are RTT and the first packet arrival delay which do not offer much information about the performance of the wireless medium.

Table 5.4.1 also offers interesting insights in the features with the highest prediction power for mobile load cases. Specifically, when the mobile VP is used, CPU, memory and RTT are the most important for detecting the problem. However, for the router and server the highest ranked feature is the RTT which has an obviously very little information regarding the load of the device. This can be reflected in the low performance of these two VPs for identifying mobile load problems.

Specifically, for network related issues the important metrics are the interface utilization, the RTT and the number of video packets. In wireless medium problem detection the greatest contribution is made from the RSSI when the mobile VP is used and from RTT for the other two VPs.

The router and server VPs have very poor detection capabilities for mobile load issues since the significant features in this case are the device CPU and memory load. The very low accuracy for these problems by the router and server VPs, is a result of the high number of instances that are detected as healthy which in turn has an impact on the number of false positives.

Apart from the mobile load, there are also other cases in Figure 5.4.2 such as mild WAN congestion and shaping where both the server and the router VPs show lower detection capabilities. This poor performance is attributed to the large number of miss-classifications of these faults as either LAN congestion and shaping or healthy and thus increasing the number of FP and FN.

However, for the case of the mobile load and WAN congestion, we find that the combined use of the three VPs significantly improves the detection performance. These findings can motivate ISPs and content providers to pursue collaborations in order to improve the performance.

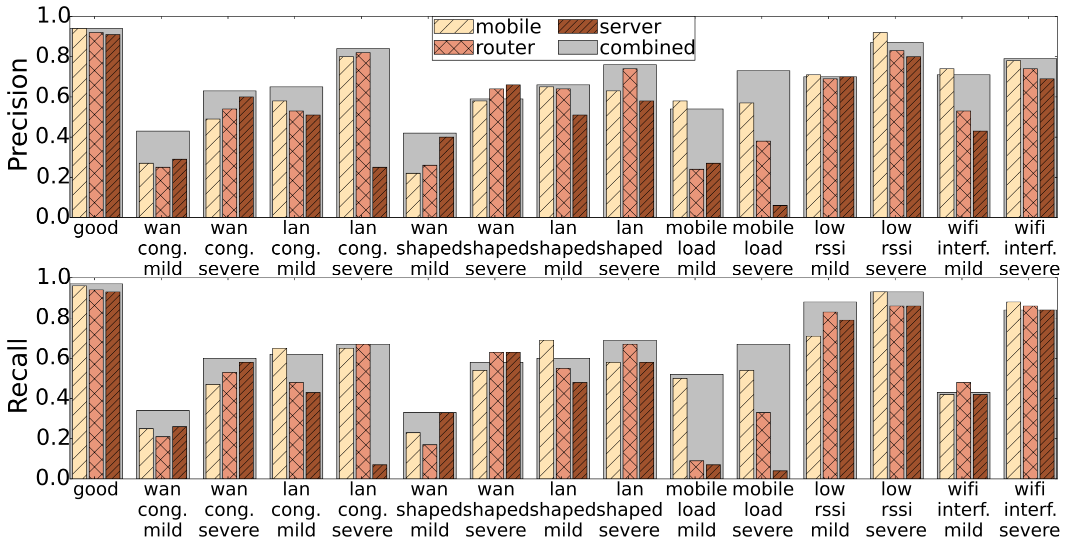


Figure 5.4.2: Precision and Recall for exact problem detection per VP.

Takeaway: Each of the entities can independently perform well when detecting a large variety of problems such as the ones related to wireless conditions, severe LAN and WAN congestion and shaping. However, mobile devices are much more accurate in identifying local problems related to high load or wireless interference. Furthermore, there cases such as WAN congestion and mobile load where the combination of all the VPs can significantly improve the performance. Finally, while we observe that all three parties are capable of identifying healthy video sessions with very high accuracy, these results indicate that in order to perform a full-scale accurate root cause analysis, some collaboration between the entities is desirable.

5.4.4 Which Features Help?

The objective of this section is to illustrate the improvements that can be achieved when using different features to train the model. We evaluate the system with the combination of the three VPs using seven different feature sets, RSSI, hardware metrics, interface utilization, network delay parameters, TCP metrics, all the available features and finally with the set of features after performing Feature Selection (FS) and Feature Construction (FC). For the network delay parameters, we consider all the RTT metrics we have available from the TCP flows.

Figure 5.4.3 shows the precision and recall for each of the inputs. When using only the RSSI or only mobile hardware metrics, the accuracy is lower than 35%. Using the interface utilization alone, yields precision and recall values near 55%, while the use of delay alone results in improved accuracies around 70%. The evaluation with the entire feature set further increases the obtained accuracy by 5% but even more improvement is reached when applying FS and FC, with precision and recall values above 80%.

From the information provided in Table 5.4.1, we observe that the utilization of the interfaces contributes significantly in the detection of the majority of problems. This result highlights the important role that feature construction plays in the problem detection capabilities of the system.

Moreover, one of the features that is used by almost all VPs for predicting congestion and shaping issues, is the first packet arrival time. This metric is an indicator of video sessions

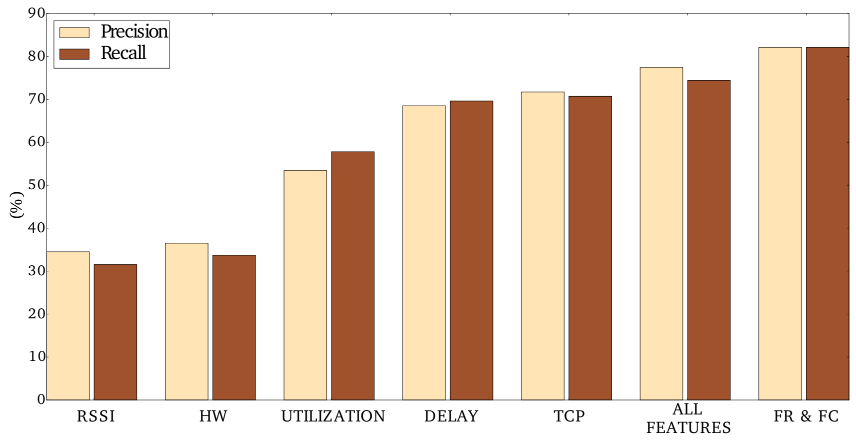


Figure 5.4.3: Problem detection accuracy for different feature sets.

with longer startup delays but it is also correlated with network issues such as delay and loss.

The metrics with higher predictive power for mobile load are the CPU and memory utilization. For the server and the router VPs where these metrics are not available, RTT is used instead but with very poor results as shown in Figure 4, as it does not hold information regarding the device's hardware state.

Takeaway: The results indicate that the RTT and link utilization, as measured by each vantage point are key metrics in performing RCA. Hardware and NIC metrics can further help us to separate individual local issues. Moreover, it is evident that feature construction and reduction plays a significant role in improving the system's accuracy as constructed features are highly ranked by the classifier.

5.5 Real World Experiments

In this section we describe and discuss the results of the system's evaluation in two real world settings. In the first environment, clients are in a corporate WiFi network where we can artificially introduce faults. In the second case, clients access videos over a wide range of wireless networks including both 3G and WiFi, where faults are not controlled and occur naturally. In both cases, clients retrieve videos from both a private server and YouTube.

5.5.1 Experiments With Induced Faults

The purpose of the the real world experiments with induced faults, is to get labeled data that will enable us to evaluate the robustness of the trained model on a real wireless network which is characterized by unpredictable topology, constant variations in traffic, signal strength and number of connected devices.

Setup

For the measurements, we distribute five Galaxy S II to equal number of users for a period of one week. The phones are again equipped with an application that automatically launches random videos from the top 100 list, while coordinating the network and hardware probes. The users were instructed to carry the phones with them while inside the wireless range in order to capture variations due to movement and received signal quality.

In these experiments, the videos are streamed from both our private video server and from YouTube with probabilities 0.25 and 0.75 respectively. We select these probabilities so that we end up with a dataset where the majority of measurements corresponds to YouTube sessions and a smaller part to streams from our server. Finally, the phones, the wireless AP and our server were instrumented with probes as described in 5.2.1.

Using the same methodology as the one in the controlled experiments described in Section 5.3, we introduce five different types of faults, lan congestion, wan congestion, mobile load, low rssi and wifi interference. Furthermore, we ensure that the conditions of the network allow to successfully load a video just before and after the induced fault. However, since this a semi-controlled environment, we cannot fully guarantee that during each video flow there are not additional (spontaneous) problems over the unmanaged Internet links or video services.

The collected dataset consists of 2619 instances from which 1962 are good, while 463 have mild and 194 have severe QoE issues.

Real-World Evaluation

Our goal is to evaluate the ability of the classifier to predict labels in the real world scenario based on the training that was performed using the controlled dataset.

In this part, we demonstrate the system's capability of detecting *the existence* of problematic instances using either one of the probes or the combination of all three. The detection is done with 88% accuracy when using the mobile probe, 84% when using the router and 81% when measurements from the server probe are only used. The combination of the three probes yields accuracy of 88.1%.

Figure 5.5.1 illustrates the Precision and Recall values for this phase of the evaluation. Overall, the results match the controlled experiments. In this case too, the mobile VP outperforms the other two VPs. However, one notable difference is the increase in both Precision and Recall for the mild problem detection. This can be attributed to the fact that the variations and background noise in the current environment is less than the variations we simulated in the controlled experiments.

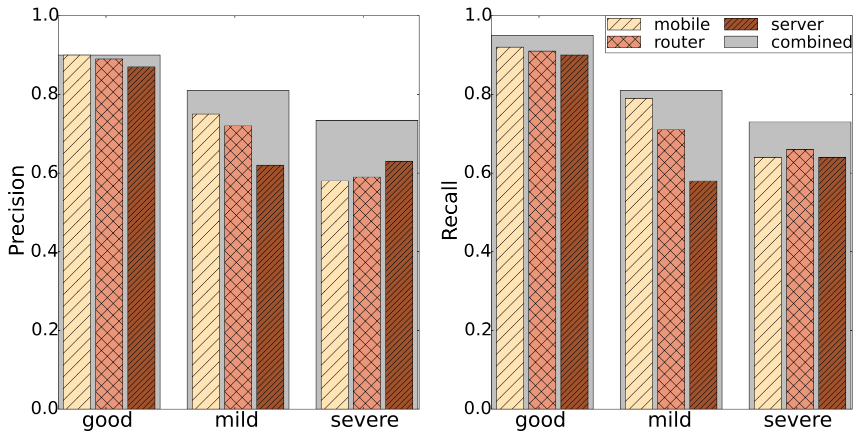


Figure 5.5.1: Precision and Recall for problem detection in the real world experiments per vantage point.

Furthermore, we also observe equally good robustness of the trained model in terms of detecting *the exact root cause* of a playback problem. In this case, the combined use of the three vantage points allows correct detection with accuracy of 82.9%. When using separately the mobile, the router and server VP we obtain accuracies equal to 81.1%, 80.5% and 79.3% respectively.

From Figure 5.5.2 we see better performance for device load and wireless medium issues which is to be expected given the strong correlation of these faults with specific hardware metrics. In the LAN congestion scenario we observe better results from the mobile and the router VP while for the case of WAN congestion the server is detecting problems with higher accuracy.

For each of the entities that participate in the video delivery this means that the VP on the client's device is necessary for detecting the root cause of the majority of problems. ISPs on the other hand, can effectively discover LAN faults but also wireless errors such as low RSSI and interference. Finally, content providers can perform WAN fault identification with good accuracy but fall short when it comes to finding faults that occur on the device or in the wireless medium.

Takeaway: Our findings here are in agreement with those in the previous experiments for problem detection and root cause identification. This is a strong indicator that *our system that was initially trained in a fully controlled environment can be successfully applied in the wild*. At the same time, smaller differences in the detection of some problems emphasize the importance of continuous training. While collecting large-scale ground-truth in the wild might not feasible, it is still possible to acquire some labels as specific problems can be recognized by experts within each entity (e.g., network engineers). Furthermore, ground-truth about the quality of experience can be given by means of crowd-sourcing (i.e., people complaining at call centers, or feedback provided by the users within the application).

5.5.2 Deployment Without Induced Faults

The final step in the evaluation is *detecting faults that were not induced by us* and, therefore, might be more complex. Furthermore, a particularly important aspect of this evaluation is to test the system in mobile networks, given that there is a constantly growing number of users who watch video over cellular broadband connections.

In this scenario too, we distribute five Samsung Galaxy S II devices to equal number of users for one month with the instruction to carry the phones with them at all times. The phones contain SIM cards with unlimited 3G data-plans, while the users were allowed to connect them to any WiFi access point. This approach allowed us to test the system on a multitude of networks that use either cellular or 802.11 technology.

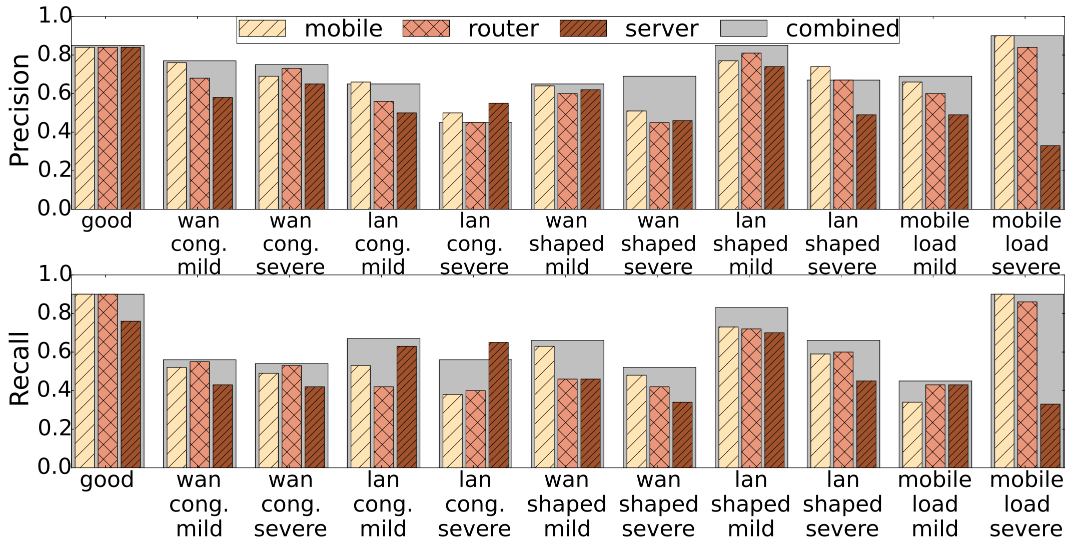


Figure 5.5.2: Precision and Recall for problem detection in the real world experiments

The videos are again streamed from both our private video server and YouTube with 1:3 ratio so that the final dataset is richer in measurements from the YouTube service. A probe collects network statistics on our video server for the sessions streamed from it. With this methodology we can have three different VP combinations, i) (mobile, router, server) when the user is streaming video from our server while using our WiFi, ii) (mobile, router) when YouTube videos are streamed on our WiFi, iii) (mobile, server) when videos are delivered from our server over other networks and iv) (mobile) when streaming from YouTube on other networks. Given that the majority of the videos were delivered over 3G and in order to make the results comparable between 3G and WiFi, we removed any features from the router (therefore only the mobile and server vantage points are used).

Similar to the previous scenario we use the trained model from the controlled experiments. For the real-world experiments, although all mobile-based measurements (e.g., hardware as well as the number of re-buffering events) are always available, the number of other metrics varies depending on the number of VPs that were used. The real-world dataset contains 3495 instances from which 2940 are good and 555 problematic.

Does it Work in the wild with real faults?

Since the experiments are done in the wild, we cannot obtain the ground truth for the root cause behind the stalls, only the ground truth for stalls and loading time. Therefore, we can only mark instances as good or problematic.

In terms of identifying the *existence* of a problem, the mobile probe, server and their combination still achieve a high accuracy and recall, as shown in Figure 5.5.3.

Similar to the controlled experiments, we find that the mobile VP is a better choice than the server for identifying both good and problematic instances, while the combined use improves the system's accuracy.

Takeaway: The results from the real world evaluation verify that the system is equally effective when detecting problems in the wild even when fewer VPs are available. This also reveals that the system can capture successfully cases of mobility although they were not covered in the training phase and it can cope with the diversity of mobile networks.

A closer look at the results shows that the detection of healthy video sessions is achieved with high accuracy, there is some loss regarding the identification of problematic videos. This loss occurs due to differences in the characteristics of the faults that we encounter in the real world as compared to the ones we induced manually in the previous sections. This effect can be minimized by introducing more VPs (e.g., on 3G RNCs) in order to get more fine grain information about how smaller variations affect the video QoE and by furthermore training the classifier with a wider range of problems. Finally, as discussed in the previous section, these figures are likely to be improved once more labeled faults are fed into the training set.

Identifying the Root-cause

We can use the trained model from the controlled experiments to predict the root cause of faults that occurred in the problematic sessions. The results of the predictor's output can be found in Table 5.5.1. As we observe, the most common type of problems occur within the users' local network (13% of all instances). Surprisingly only few (2%) of the instances are estimated to be caused by low RSSI or WiFi interference as typically the videos fail to

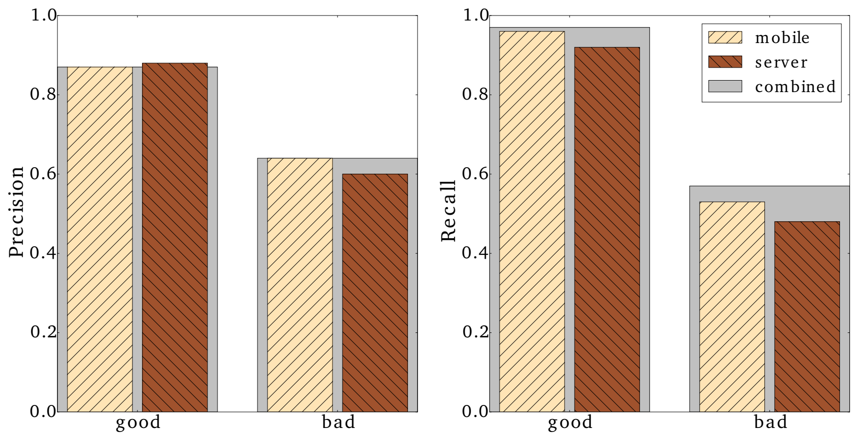


Figure 5.5.3: Precision and Recall for problem detection per VP pair in the real world.

even start a TCP flow when there is very low signal. Furthermore, a number of instances (4%) were problematic due to an estimated high mobile load.

As discussed in the previous section, we can directly calculate that the algorithm correctly identifies good instances with 85% accuracy. Furthermore, although it is not possible to verify all of these estimated root causes, we still have the ground truth for some of them: mobile load and low RSSI.

Figure 5.5.4(left) shows the distribution of CPU load on the mobile device for problematic videos sessions as predicted by the video server Vantage Point. Two different distributions of the CPU ground truth are given: video sessions that server VP labeled as high "mobile load" and the remaining video sessions. The results show that, although the server vantage point only has access to transport layer metrics (TCP statistics), the video flows that were estimated as high mobile load have indeed much higher CPU utilization.

Similarly, Figure 5.5.4(right) shows the distribution of RSSI for the instances that were considered as low RSSI from the point of view of the server's vantage point. As before, we observe that the server vantage point can successfully identify these instances despite the fact that the phones were connected to various WiFi and 3G networks.

Takeaway: These results further reinforce our hypothesis that a model that was trained in a controlled environment, is robust enough to be applied as a starting point on a real

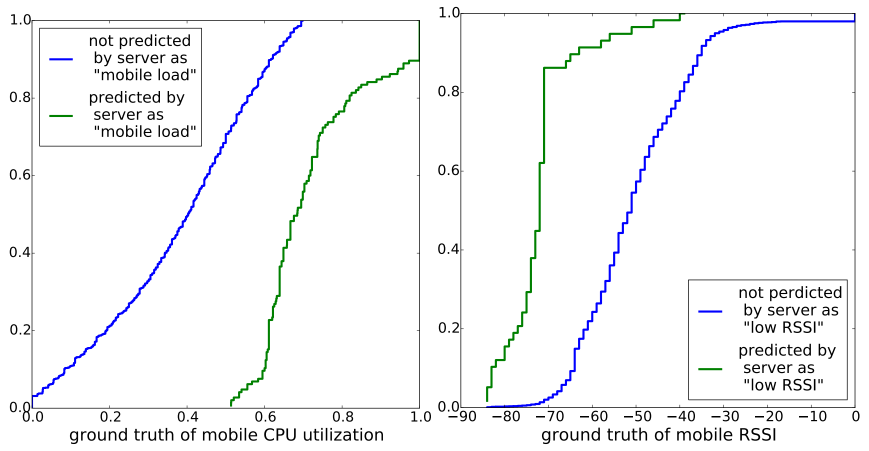


Figure 5.5.4: Comparing the server estimations about CPU (left) load and RSSI (right) to the ground truth

world environment where the network conditions and the faults can be highly dynamic and unpredictable. Furthermore, we observed that even the service provider VPs can identify problems that occurred within the users network or device (e.g., low RSSI or high CPU usage) without any external information. However, we also need to mention that the root-cause detection model is limited to detecting only the issues it was initially trained with. As a result, it will not be able to detect new types of anomalies that were not introduced in the training phase.

GOOD	WAN CONG.		LAN CONG.		MOBILE LOAD		LOW RSSI		WIFI INTER.	
	M	S	M	S	M	S	M	S	M	S
2499	163	166	18	446	2	132	26	0	43	0

Table 5.5.1: Real-world root cause predictions (M=mild, S=severe)

5.6 Practical Implications

For the end user, our results indicate that even an isolated mobile application that collects measurements from multiple layers can successfully identify a large number of problems without further instrumentation. Such a system can be a powerful tool towards diagnosing video playback QoE issues and where they have occurred. Therefore, when the user is made aware of the location of the problem and if it is originating from the local network or the device itself then troubleshoot it. Otherwise, the issue can be reported to the responsible entity to take the necessary action.

For the ISPs, the results demonstrate that they can also independently identify problematic sessions, even when traffic is encrypted. Furthermore, they can identify if the problems originate within their own network, in order to fix problematic segments and bottlenecks, but also guide users to solve problems in their home and/or their devices.

For content providers, there are deployment benefits such as detecting loaded servers and network segments in their CDN if the problem occurs on their side, or adapting the content for problematic connections without instrumenting the client when the problem is originating from either the user's or the ISP's side. This is a valuable tool when identifying SLAs and net neutrality violations.

Collaboration: As we showed in the majority of the controlled and real world evaluation scenarios, there are significant improvements (in terms of identifying all possible problems) when two or more entities collaborate to troubleshoot QoE issues. The greater benefits however, are obtained for the entities which collaborate with the end users, since the mobile device has access to valuable information of the local hardware and network performance. This calls for instrumented players or mobile devices.

At the same time, as collaborations might not be possible, an iterative root cause analysis might be employed where each of the entities independently perform analysis within their own infrastructure. Then they report to the other entities along the path whether or not the problem has occurred in their segment. In this way, no sensitive information is exchanged among users or providers, collaborations can be easier established and the deployment of the system can span over a wider range of networks and devices.

Continuous Training: Once the system is deployed in one or more entities, its fault detection and root cause analysis capabilities can be further improved by means of continuous training. With this approach, a feedback loop can constantly update the training dataset with new instances of already known issues and at the same time, introduce manually labeled instances corresponding to new anomaly types. Hence, the model can be re-trained with a more rich and diverse dataset and result in a higher accuracy system.

The continuous training methodology will help address one of the limitations of our system, which is the inability to detect faults that it has not been trained for yet in the lab. These would not only include new problems such as middleboxes and DNS or routing miss-configurations but also the co-occurrence of problems that jointly affect video QoE.

5.7 Chapter Summary

In this chapter we presented a multi-vantage point system for detecting video QoE issues and identifying their root cause. Our approach utilizes performance metrics from multiple layers which makes it agnostic to video characteristics and streaming mechanisms but also to encrypted traffic. With the aid of feature reduction and construction techniques, the detection and RCA of problems is done with a minimal set of performance metrics while we ensure that the methodology is generalizable and can be applied to different video services and clients. We further showed that each of the entities which contribute to the video delivery is capable of detecting poor QoE and identify underlying faults without having to share information with other parties.

The next step in this work, is to extend the list of problems that can be identified and train the system for multi-problem detection. To further improve the system's accuracy, we will examine dividing problematic sessions into more labels in order to obtain a more fine grain classification of the severity of the problem.

6
Combining Performance Metrics to Better Capture Per-Sector QoE in Mobile Networks

6.1 Introduction

In this chapter, we study how to bridge sector KPIs to reflect Quality of Experience (QoE) ground truth measurements, namely throughput, latency and video streaming stall events. We leverage one month of data collected in the operational network of mobile network operator serving more than 10 million subscribers. We extensively investigate up to which extent adopted methodologies efficiently capture QoE. Moreover, we challenge the current state of the art by presenting data-driven approaches based on Particle Swarm Optimization (PSO) metaheuristics and random forest regression algorithms, to better assess sector performance. Results show that the proposed methodologies outperforms state of the art solution improving the correlation with respect to the baseline by a factor of 3, and

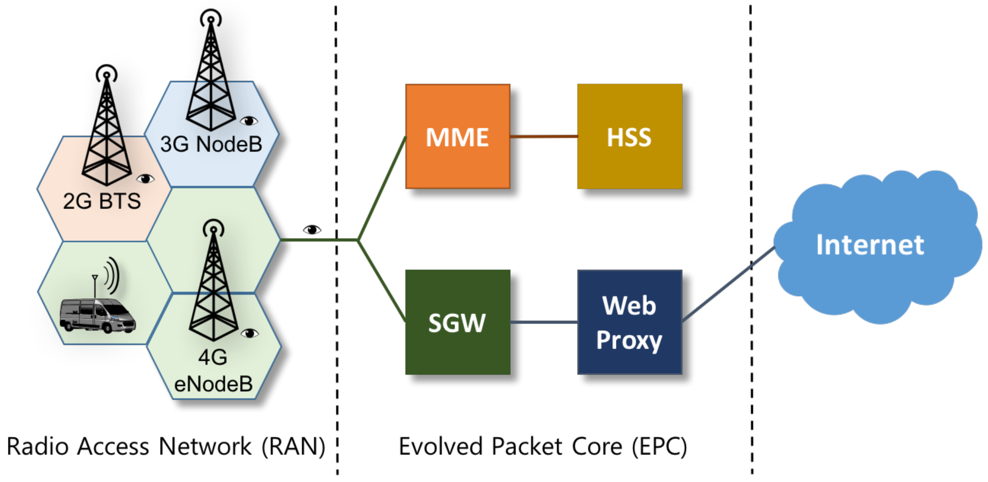


Figure 6.2.1: Sketch of a mobile network. Notice that each tower might house multiple sectors.

improving visibility on under-performing sectors. Our work opens new areas for research in monitoring solutions for enriching the quality and accuracy of the network performance indicators collected at the network edge.

It is important to note that the thesis author has made contributions in the video QoE parts of the work that is described in this chapter. More specifically, these contributions include the collection of video streaming traffic, the extraction of performance metrics and QoE indicators and the labeling of video instances in order to provide the ground truth for the training of the algorithm.

6.2 Monitoring network KPIs

Fig. 6.2.1 sketches the architecture of a 4G mobile network. The *radio access* consists of hundreds of thousands of components (e.g., sectors, towers, and controllers). Those compose the mobile network "last mile", and bridge the users' devices with the *core network*, which enables access to voice calls and the Internet. To support troubleshooting and optimization, the vendors provide monitoring platforms to (passively) collect KPIs from radio access network elements and backbone links. To simplify the monitoring complexity, KPIs normally are aggregated over time (with periodicity varying from minutes to hours) and users.

There is no definitive list of KPIs. However, through the joint effort of vendors and operators, it is easy to identify a set of KPIs whose importance is acknowledged by multiple parties [15–17, 27, 28]. Table 6.2.1 lists some examples. We can categorize KPIs into five different groups:

- Signaling: These KPIs are mostly related to faults such as failure to establish a Radio Access Bearer (RAB), or Radio Resource Control (RRC), or the fact that the sector cannot efficiently reach the radio controllers.
- Voice: These KPIs capture failure to establish or maintain a voice call.
- Data availability: These metrics reflect the availability of high-speed data channels (such as HSDPA/HSUPA) at any given time or the number of data-active connected devices.
- Data Congestion: These metrics capture the fact that the capacity was reached. For instance, the average number of users queuing to get an HSDPA/HSUPA channel, or the number of times a data connection had to be dropped to make room for a voice call. Furthermore, they indicate the percentage of time that the radio was active transmitting data (radio utilization).
- Radio: Radio KPIs have to do with interference, power statistics, measured wireless noise, signal conditions, etc.

6.2.1 Thresholding individual KPIs

The primary use of KPIs is to enable operators to monitor the health of the network and to quickly identify bottlenecks. Therefore, it is natural to use KPIs as a way to flag network conditions that deteriorate below an established performance limit [17, 26, 52]. As a result, for each KPI, a *threshold* has been set. Such thresholding mechanisms are widely used in the

KPI	Thres.	Category
Failed RRC/RAB requests ratio	$> 5\%$	Signaling
Signaling failure	$> 1\%$	
Call setup failure rate	$> 2\%$	Voice
Call drop rate	$> 5\%$	
HSDPA/HSUPA session setup success %	$< 90\%$	Data
Average Users Queuing	> 2	
Time Slots Transmitting	$> 80\%$	
Noise Rise	$> 10db$	Radio

Table 6.2.1: Example of some KPIs and thresholds that indicate poor QoE. Notice that these are vendor-recommended values and, thus, they might differ from the ones used by the operators.

industry, and allow network planners and radio resource operators to focus their attention where it is required. Table 6.2.1 shows examples of such thresholds.

Default threshold values are proposed by the equipment vendors, while operators further fine-tune them based on their experience, objectives, and domain knowledge. Therefore, significant investment is made through drive tests, controlled experiments, and A-B testing in order to identify performance bottlenecks, and how these relate to the KPIs and thresholds [14, 26–29].

6.2.2 Combining different KPIs

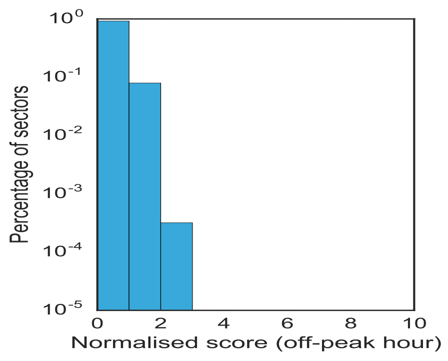
While establishing thresholds for each KPI enables a fine-grained vision of specific problems, each network element is associated with hundreds of metrics. It is then desirable to consolidate measurements into a single performance index so to i) quantify the "health" of each network element, ii) easily assess the whole network status and trends and iii) narrow down on sites that need attention.

An example of such indices is the *hotspot score* [17, 52, 53], which represents how "hot" or problematic a given sector is. It is a weighted combination of thresholded KPIs related to signaling, voice, data availability and data congestion:

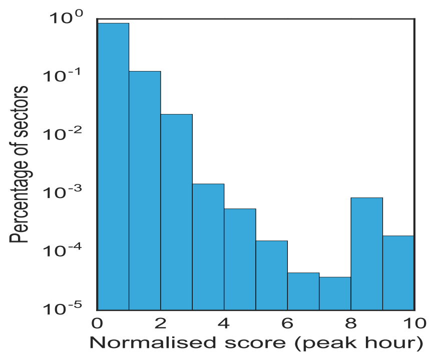
$$s_b = S(\mathbf{k}_b, \mathbf{w}, \mathbf{t}) = \sum_{i=1}^{n} w_i \cdot H(k_{b,i} - t_i), \tag{6.1}$$

where b is the sector under study generating n KPIs collected in the vector $\mathbf{k}_b$. The KPIs are associated to weight $\mathbf{w}$ and threshold $\mathbf{t}$ vectors, while $H(\cdot)$ is the Heaviside step function which outputs 1 when the KPI value $k_{b,i}$ reaches the corresponding threshold t_i and 0 otherwise. The higher the score s_b, the more "hot" the sector b is. In a nutshell, a hotspot score is a linear combination of the weights associated to KPIs that trigger.

Notice also that, since KPIs are gathered periodically, s_b is time dependent, but we avoid to express it in Eq. 6.1 for readability. This means that individual s_b values can be further combined (e.g., simply accumulated) to reflect performance over a longer time span. Sectors that exhibit high scores for a long period of time (e.g., a few days or even weeks) can then be flagged for intervention (such as changing the configuration, fixing possible faults, adding capacity, or even upgrading the whole site to a newer technology like 4G). However, when referring to a hotspot score in the following, we will consider the values computed using KPIs gathered at the baseline periodicity of the monitoring system, i.e., one hour for the network under study.



(a) Hourly score, off-peak



(b) Hourly score, peak

Figure 6.2.2: Ratio of sectors that achieve a certain hotspot score given the default formula. The majority of sectors have a score of zero.

6.2.3 Challenging the state of the art

To better understand the hotspot score dynamics, let us consider a few examples related to the whole set of 3G sectors of a real mobile network operator serving a country with over 10 million subscribers (see Sec. 6.4 for more details). Fig. 6.2.2a reports the fraction of sectors having a certain hotspot score for an off peak hour. We consider only the KPI values within this hour to create per-sector scores. As expected, when the network is lightly used, only a very small fraction of KPIs trigger with a maximum score of 2 (i.e., not critical for performance). Instead, the scenario significantly changes when considering the peak hour as reported in Fig. 6.2.2b. The distribution is bimodal, with a cluster of sectors having a score higher than 8.

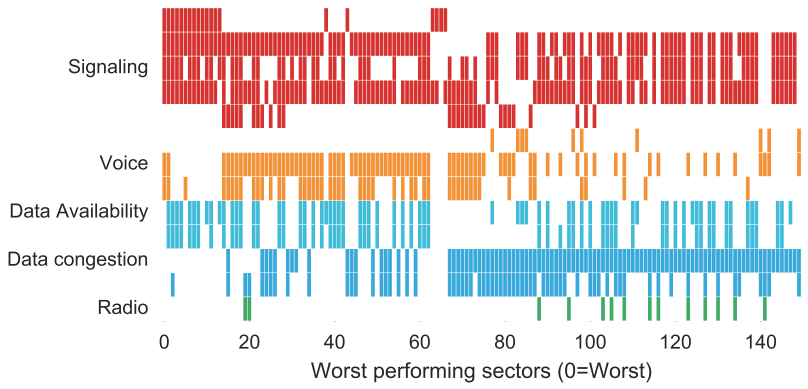


Figure 6.2.3: The 150 sectors with the worse hotspot score and the type of KPIs that triggered it.

Fig 6.2.3 shows 13 of the important KPIs (grouped in sub-classes) for the worse performing 150 sectors out of tens of thousands in the network.[1] Columns are associated to sectors, sorted by decreasing score values (worse performing sectors are on the left), while colors are used to associate KPIs into categories (colored cells highlight which KPI triggers for which sector).

As previously introduced, the hotspot score is a mixture of signaling, voice, and data KPIs. Among the three, typically signaling issues have the highest weights since they can prevent utilization of voice and data. We can observe how those KPIs (Fig. 6.2.3 top) trigger more than the remaining ones. From left to right, we find sectors with signalling issues, followed by the ones with poor voice quality, and finally the ones that are congested. Voice

[1]We cannot reveal the exact list of KPIs nor the exact number of sectors due to their sensitive nature.

problems are also highly valued due to the traditional service model of mobile operators, and due to the fact that users clearly perceive voice failures when they occur. Finally, data issues and congestion are given the lowest priority. Interestingly, the worse 60 sectors extensively trigger both voice and data KPIs, while for the rest there is a predominance of data KPIs.

The previous figures are clear examples that state of the art KPI analysis is useful to spot critical sites. However, this comes at the cost of a very rigid, static and empirical methodology for which we see a number of issues. First of all, it is unknown up to which extent the adopted approach reflects QoE as, to the best of our knowledge, no quantitative analysis has been performed. Secondly, even assuming the hotspot score function is properly configured, current techniques require a significant (domain knowledge and manual) effort to keep tools at pace with the constant evolution of Internet services and new technologies. Finally, while the hotspot score provides rankings of overall performance, it does not provide any indication about application specific-QoE.

One could argue that application-specific QoE is best captured through its own KPI. However, this has a clear overhead: monitoring application performance in a large-scale cellular network requires either the deployment of middle-boxes that are capable of interpreting application performance by analyzing the (mostly encrypted) traffic of millions of users, or it requires collaboration with individual application developers in order to share their KPIs. Therefore the question we try to answer is: *could we use network KPIs that are already collected to monitor the health of the network to get insights into app performance?*

We believe there is the need to focus on network performance monitoring methodologies that i) shed light on the quality that people experience when interacting with specific data applications (e.g., web or video) and ii) can adapt to the mobile network dynamics. Hence, in this work, we address the following two challenges:

- We *study hotspot scores* using data collected from a real mobile network serving over 10 million subscribers. Specifically, we investigate up to which extent the used threshold and weighting mechanisms are sufficient to identify under-performing sectors.

- We *provide new methodologies* to enable us to gain some visibility on under-performing sectors. Specifically, we want to leverage the vast amount of data collected to create an automated data-driven framework capable to go beyond current domain knowledge and empirically-driven approaches.

6.3 A data driven approach to KPIs analysis

We assume to have a set of KPI values per sector that we want to match with specific QoE metrics that are affected by the performance of the edge-network, such as web throughput and latency or video performance. In this section, we examine how we can use the KPIs to extract a score that better matches the ground truth. Based on Eq. 6.1, we identify two possible directions of exploration. On the one hand, we can keep the already defined hotspot formula, but create an optimization engine to better tune its parameters: the weights and the thresholds. On the other hand, we can create a new formula or an implicit model to combine the same input KPIs.

6.3.1 Optimizing the current score function

As mentioned, the most straightforward mechanism to better relate a given ground truth $\mathbf{g}$ and the scores $\mathbf{s}$ is by optimizing the thresholds $\mathbf{t}$ and weights $\mathbf{w}$ involved in the calculation. To relate $\mathbf{g}$ and $\mathbf{s}$, we can use standard correlation. However, in our application, we are not interested in the raw quantities in $\mathbf{g}$ and $\mathbf{s}$, but rather in the sector rankings they define. In fact, the two quantities could lie in different ranges or have a monotonic non-linear relation and still produce the same sector ranking. Therefore, a natural choice to measure the correlation between $\mathbf{g}$ and $\mathbf{s}$ is the Spearman's rank correlation coefficient [54], often called Spearman's ρ. The Spearman's ρ is only affected by changes in the rankings of the input variables, and is thus invariant to scale, location, and monotonous non-linear relations. Its output ranges from -1 (perfect reverse ranking) to 1 (exact same ranking).

Having defined the function relating $\mathbf{g}$ and $\mathbf{s}$, we can now formulate our problem in terms of a classical optimization problem [55]: we want to find the weights $\mathbf{w}$ and thresholds $\mathbf{t}$ that maximize

$$\rho(\mathbf{s}, \mathbf{g}), \tag{6.2}$$

where $\mathbf{s} = [s_1, \ldots s_m]$ and $\mathbf{g} = [g_1, \ldots g_m]$ are vectors collecting all m sector scores and ground truth measurements, respectively. Notice that, as the values in $\mathbf{w}$ and $\mathbf{t}$ are real numbers, we have a potentially infinite number of such combinations.

Given that the calculation of ρ is not directly differentiable (due to the ranking operation), and that it is computationally cheap (essentially involving two vector sorting, one vector subtraction, and one vector multiplication), we decide to solve the previous optimization problem with a *Metaheuristic algorithm* [56]. Meta-heuristics conform a general algorithmic framework for addressing optimization problems. Unlike classical optimization algorithms and iterative methods, meta-heuristics make few assumptions about the optimization problem being solved, and can thus be applied in a variety of problems, including problems with non-differentiable functions. Meta-heuristics allow us to define our own optimization function, while they provide a way of exploring the parameter space (thresholds $\mathbf{t}$ and weights $\mathbf{w}$) in a structured and efficient way. Despite being approximate and non-deterministic, metaheuristic algorithms can efficiently explore the search space and provide near-optimal solutions within a reasonable amount of time [55], specially if the optimization function is not computationally expensive (as it is in this case). These methods are often inspired by processes occurring in nature, such as Darwinian natural selection, annealing, and collective behaviour of ants [57].

Particle swarm optimization (PSO)

Particle swarm optimization (PSO) [58] is a population-based metaheuristic for solving continuous and discrete optimization problems [55]. PSO has recently gained increasing popularity among researchers and practitioners as a robust and efficient technique for solving difficult optimization problems. It makes few or no assumptions about the problem being optimized, can search very large spaces of candidate solutions, and can be applied to problems that are irregular, incomplete, noisy, dynamic, not necessarily differentiable, etc. (see [55, 58, 59] and references therein).

PSO operates by having *a population of candidate solutions*, which are metaphorically represented as *particles*. At each iteration, these particles explore the search-space according to their current positions and a velocity towards a goal. They iteratively calculate the fitness ρ corresponding to their current position, and update the latter according to the available knowledge of the search space. The movement of a particle is affected by the best position it has found but, in addition, it is also affected by the best known positions discovered by other particles [59]. Moreover, particle movements are not deterministic, but partly stochastic, thus facilitating the exploration of the search space.

Before delving into the details of our PSO algorithm, we first need to define what corresponds to a particle's position, which we will denote by the vector $\mathbf{x}_i$. Specifically, in our case, the position will correspond to the $2n$ weights and thresholds: $\mathbf{x}_i = [\mathbf{w}, \mathbf{t}]$. With that, we see that we can easily constrain the search space with some domain/intuitive knowledge, imposing upper and lower bounds for the particles' positions. For the weights, we define them to be $-1 \leq w_i \leq 1$ (a resulting weight of zero means that the triggered KPI does not influence the ground truth whereas values above or below zero contribute positively or negatively towards the score). For the thresholds, for each KPI, the minimum and maximum over all sectors is used such that $\min(k_i) \leq t_i \leq \max(k_i)$. When a particle exits the search space, its position is reset to a random location inside the established bounds.

Algorithm 1 details the PSO functioning. First of all, we initialize τ particles $\mathcal{P}_i$ with random positions $\mathbf{x}$, random velocities $\mathbf{v}$, and lowest possible personal fitness $\rho = -1$. Next, we start iterating, with a maximum number of iterations λ. When iterations are finished, we return the best found correlation $\rho^\star$ and position $\mathbf{x}^\star$ which, as mentioned, comprises the best found weights $\mathbf{w}^\star$ and thresholds $\mathbf{t}^\star$. The first inner loop of Algorithm 1 checks every particle's fitness by computing the Spearman's ρ. A particle i updates its best known location $\mathbf{x}_i^\star$ and found correlation $\rho_i^\star$ if the latter has improved over the previous value the particle had. The second inner loop performs the actual search towards a better solution.

The first step in the second inner loop is looking for promising positions found by the particle's neighbours. To facilitate exploration, it is common practice to consider different interactions between the particles, grouping them in so-called neighbourhood topologies [55]. In our implementation, particles are grouped following a *ring* topology (Fig. 6.3.1). Therefore, the neighborhood of a particle only consists of two other particles. Particles that are neighbors in the ring collaborate together and influence each other. This implicitly creates small groups of particles that explore different parts of the search space, while

Algorithm 1: Particle swarm optimization algorithm basics.

Input: The set of KPIs $\mathbf{k}$, a scoring function S, the ground truth $\mathbf{g}$, a fitness function ρ, and position bounds Γ.
Output: Best found correlation $\rho^\star$, weights $\mathbf{w}^\star$ and thresholds $\mathbf{t}^\star$.

Parameters: Maximum number of iterations λ, number of particles in the swarm τ, and mutation probability a.

```
// Initialize particles at random with correlations $\rho^\star = -1$
 $\mathcal{P} \leftarrow$ Init();
// Iterate
 for $l \in [1, \lambda]$ do
    // For each particle $\mathcal{P}_i = \{\mathbf{x}_i, \mathbf{v}_i, \mathbf{x}_i^\star, \rho_i^\star\}$
    for $i \in [1, \tau]$ do
        // Calculate $\rho$ in the current position
         $\mathbf{w}, \mathbf{t} \leftarrow \mathbf{x}_i$;
        for $b \in [1, m]$ do
            $s_b \leftarrow S(\mathbf{k}_b, \mathbf{w}, \mathbf{t})$;
        end
        $\rho_i = \rho(\mathbf{s}, \mathbf{g})$;
        // Update if better
         if $\rho_i \geq \rho_i^\star$ then
            $\rho_i^\star, \mathbf{x}_i^\star \leftarrow \rho_i, \mathbf{x}_i$;
        end
    end
    // For each particle $\mathcal{P}_i$, identify the $\mathcal{P}_j'$ particle in its
      // neighbourhood having the highest $\rho_j^\star$
      $\mathcal{P}' \leftarrow$ GetNeighbors($\mathcal{P}$);
    // For each particle $\mathcal{P}_i = \{\mathbf{x}_i, \mathbf{v}_i, \mathbf{x}_i^\star, \rho_i^\star\}$ and
    // best neighbor $\mathcal{P}_j' = \{\mathbf{x}_j, \mathbf{v}_j, \mathbf{x}_j^\star, \rho_j^\star\}$
    for $i \in [1, \tau]$ do
        // Compute new velocity using random vectors $\mathbf{u}$
         $\mathbf{v}_i = \chi \cdot \left(\mathbf{v}_i + \varphi_1 \mathbf{u}_1 \otimes (\mathbf{x}_i^\star - \mathbf{x}_i) + \varphi_2 \mathbf{u}_2 \otimes (\mathbf{x}_j^\star - \mathbf{x}_i)\right)$ ;
        // Mutate velocity components with probability $a$
         $\mathbf{v}_i \leftarrow$ Mutate($a, \mathbf{v}_i$);
        // Compute new position
         $\mathbf{x}_i = \mathbf{x}_i + \mathbf{v}_i$ ;
        // Constrain the particle within the desired bounds
         $\mathbf{x}_i \leftarrow$ Constrain($\Gamma, \mathbf{x}_i$);
    end
end
// Identify the particle $j$ with the best correlation $\rho_j^{\text{best}}$
 $\rho^\star, \mathbf{w}^\star, \mathbf{t}^\star \leftarrow \rho_j, \mathbf{x}_j$;
return $\rho^\star, \mathbf{w}^\star, \mathbf{t}^\star$
```

the ring itself is ultimately pulled towards the best known solution [59]. In the example of Fig. 6.3.1, particle A will be influenced to move towards particle G (as it has found a solution with better correlation), whereas particle B will move towards particle C. Particle G will keep moving towards the same direction.

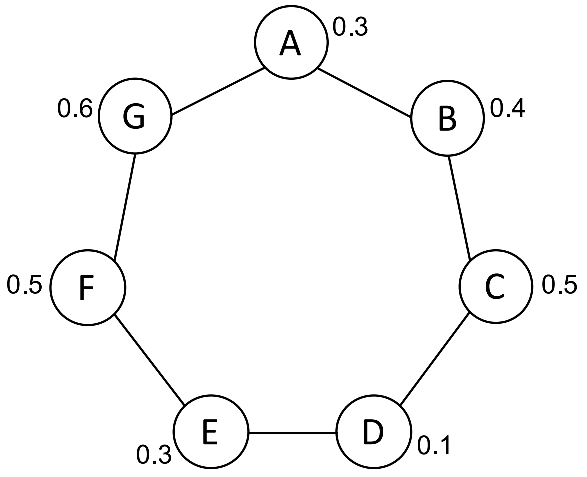


Figure 6.3.1: Swarm of 7 particles and the Spearman's ρ of their best discovered position. Each particle movement is influenced by its own best found position and, at the same time, is also guided toward the best known position of its two neighbours.

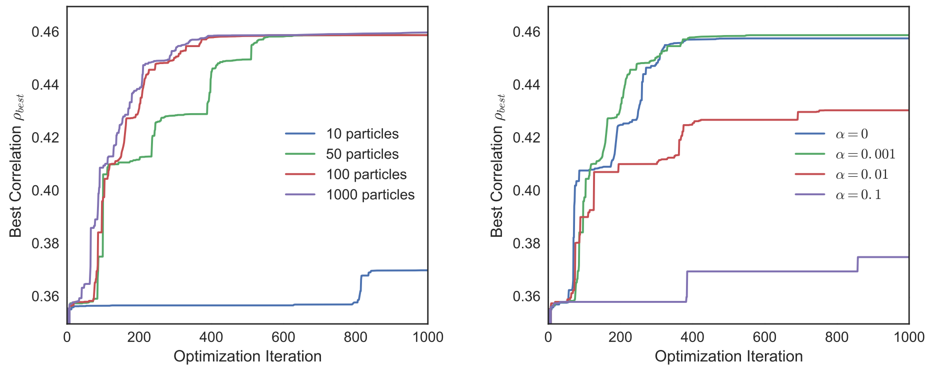
The next step in the second inner loop 1 updates the particles' velocity[2]. The new velocity of a particle $\mathcal{P}_i$ ($\mathbf{v}_i$) is influenced by the current position $\mathbf{x}_i$, the distance from the position with its best known fit $\mathbf{x}_i^\star$, and the distance from the best known fit of the closest particle $\mathbf{x}_j^\star$. To further enhance the exploration capabilities of particles, the velocity vectors defined by $\mathbf{x}_i^\star - \mathbf{x}_i$ and $\mathbf{x}_j^\star - \mathbf{x}_i$ are component-wise multiplied ($\otimes$) with random vectors $\mathbf{u}_1$ and $\mathbf{u}_2$, formed by uniformly-distributed random values between 0 and 1.

Constants χ, φ_1, and φ_2 are pre-set following the so-called Clerc's constriction method [59], which is common practice in the PSO literature. These constants control the behaviour of the particles, and allow an elegant and well-explained method for preventing explosion and ensuring convergence [58]. In our implementation, to prevent the stagnation of the swarm

[2]Notice that a velocity vector in more than one dimension has a modulus and an angle. Therefore, it also carries information of direction.

and to facilitate escaping from local maxima, a further random mutation of the velocity components is employed [60]. For that, we use a small probability α.

According to Algorithm 1, we need to define three parameters: the number of iterations λ, the number of particles τ, and the mutation probability α. From a theoretical stand point, the larger the value of λ and τ, the higher the chances to find globally optimal solutions [59]. However, an unreasonably high value of those parameters could harm the efficiency of the algorithm (in terms of computation time). In practice, we found that, with a ring of $\tau = 100$ particles, $\alpha = 0.001$ and $\lambda = 500$ iterations, we converge towards a solution that does not significantly improve if we wait longer or perform additional trials. Fig. 6.3.2 shows the sensitivity of the correlation ρ with respect to the number of iterations. The figure also shows that, for 500 interations, only unreasonable settings of τ and α produce non-optimal results. We empirically find these settings to correspond to $\tau < 50$ and $\alpha > 10^{-3}$. Therefore, we set $\alpha = 10^{-3}$.



(a) Swarm size **(b)** Mutation probability α

Figure 6.3.2: Sensitivity of PSO with respect to number of iterations considering different swarm sizes τ (a) and mutation probabilities α (b).

6.3.2 Improving over the current function

Until now, we have only discussed how to optimize the default weights and thresholds used in the scoring function of Eq. 6.1. However, such function could be limited by the application of the thresholds, which hide the *raw "analog"* KPIs value. Thus, we could potentially significantly improve the correlation between **g** and **s** by using the richer information in the raw KPI values. We envision two alternative hotspot score functions.

Linear combination of raw KPIs: this simply consists in removing the thresholding from Eq. 6.1, such that

$$\hat{S}_b(\mathbf{k}_b, \mathbf{w}) = \sum_{i=1}^{n} w_i \cdot k_{b,i}. \tag{6.3}$$

To avoid having heavily asymmetric weights, we however need to normalize the KPIs to be in the range between 0 and 1. This is the most simple model we can think of in order to combine KPIs into a single hotspot score.

Non linear models: in this case, we rely on machine learning algorithms to explore non-linear relationships between KPIs and the ground truth. More specifically, we will not create anymore a scoring function, but we will associate directly KPIs and ground truth **g**. A generic way to learn such association is through a machine learning regression algorithm [61].

A machine learning-based approach indeed offers a number of advantages. Firstly, the resulting model does not make any assumptions about the underlying function. Secondly, the model can (potentially) produce an actual value of the estimated performance rather than just a ranking. For instance, a regression tree [62] builds a model that exploits the KPIs to estimate the actual conditions for each sector at that time: $f(\mathbf{k}) \rightarrow \hat{\mathbf{g}}$. Depending on the availability of ground truth for the training set, the predicted value $\hat{\mathbf{g}}$ can be any condition that correlates with the KPIs.

At the same time, a machine learning-based approach presents some disadvantages. Firstly, most machine learning models optimize against the target variable (e.g., mean squared error), whereas in the optimization process we define our target: optimize the ranking. Thus, this does not exactly match current practice, as operators would prefer a prioritized ranking of sectors to address. Secondly, the resulting machine learning model is often not easily human-interpretable, and network engineers cannot modify it to include external restriction such as SLAs or other priorities. Furthermore, certain KPIs might have significant importance in terms of detecting a *failure* rather than a performance degradation. Thirdly, it breaks the current paradigm and the tools that are already used in operation.

In our experiments, we use the scikit-learn toolkit [63], a well-known machine learning framework offering various techniques (e.g., decision trees, random forests, SVMs, neural network regression, etc). In preliminary analysis, the best results were achieved with *random forests* [64]. We use the *mean square error* for our tree split criterion and 100 tree estimators. To avoid any over-fitting, we restrict the minimum number of samples that end up in leaf-nodes to 0.5% of the dataset, as this will not allow the tree to grow indefinitely. These parameters were studied with separate training set and they were later applied to our validation data set.

6.4 Data sets

As introduced in Section 6.3, we aim to study how to combine sector KPIs to obtain a synthetic score that better reflects users' QoE. For this purpose, we combine two data sources i) a set of KPIs collected per sector and per hour, ii) QoE metrics collected using weblogs at per HTTP request granularity and then associated to specific sectors. We can think of KPIs as the *features* that we aim to associate with the QoE *ground truth*.

We leverage one month of such measurements, collected between January and February 2016 in an operational mobile network serving over 10 million subscribers. To ease the analysis and reduce the impact of night/day fluctuations, we study the peak hour (same hour across multiple days). It is important to underline that the data sets capture the network at scale, i.e., all sectors and all customers observed during the investigation period.

6.4.1 Sector KPIs - Features

As described in Section 6.2, a number of KPIs related to i) signaling, ii) voice, iii) data availability, iv) data congestion and v) radio performance are collected. In total, we consider 21 KPIs [3] for n sectors, where n is in the order of hundreds of thousands. These KPIs are measured *hourly*.

[3]KPIs are selected based on both internal knowledge and vendors recommendation

6.4.2 Weblogs - ground truth

Web QoE metrics are provided by a web accelerator middlebox (*webproxy* in the following) that is deployed in the operator's network and is responsible to cache and compress HTTP objects (see Fig. 6.2.1). At the same time, for each HTTP request, the webproxy logs information such as timestamp, download duration, subscriber's id, bytes transferred, and URL[4]. More importantly, each transaction contains *TCP metrics* (e.g., min/max/avg round trip time, dropped or duplicated packets etc.) capturing the delivery performance between the middlebox and users' device.

Video QoE: To extract video QoE we exploit the fact that the YouTube player reports to Google servers summary statistics at the end of each video playback. These include if the video has successfully loaded, if the playback has started, paused or stopped, if there were stalls and how long these lasted [7]. Despite most of the content is now served via HTTPS [65], we still see a residual amount of YouTube videos served over HTTP and we use this to generate statistics per video transaction (e.g., number and duration of stalls, average resolution, etc).

Mapping requests to a sector: The webproxy logs do not contain any information related to the *sector* where the data were consumed. Therefore, we use *radio events* recorded by the Mobility Management Entity (MME) (Fig. 6.2.1) in order to enhance each web transaction with the set of sectors that were used. Indeed, MME servers logs the "control plane" messages (related to paging, radio channel requests, and handovers) each containing the sector from which users devices sent the message.

For each subscriber we can then create a time-line describing the sectors connected. With such information we can tag weblogs entries based on the sector where the requests have been generated. To avoid any ambiguities, if more than one sectors were used to serve a object (i.e., there was a handover), we discard the transaction (this occurs for less than 5% of the downloads).

Notice that this process is not trivial: radio events and weblogs corresponds to more than 3TB per day. In our case, we exploited the BigData cluster of the operator to implement this enrichment. However the described job requires processing that can take hours,

[4]Users' privacy is protected by proper anonymization techniques.

even in large data clusters.

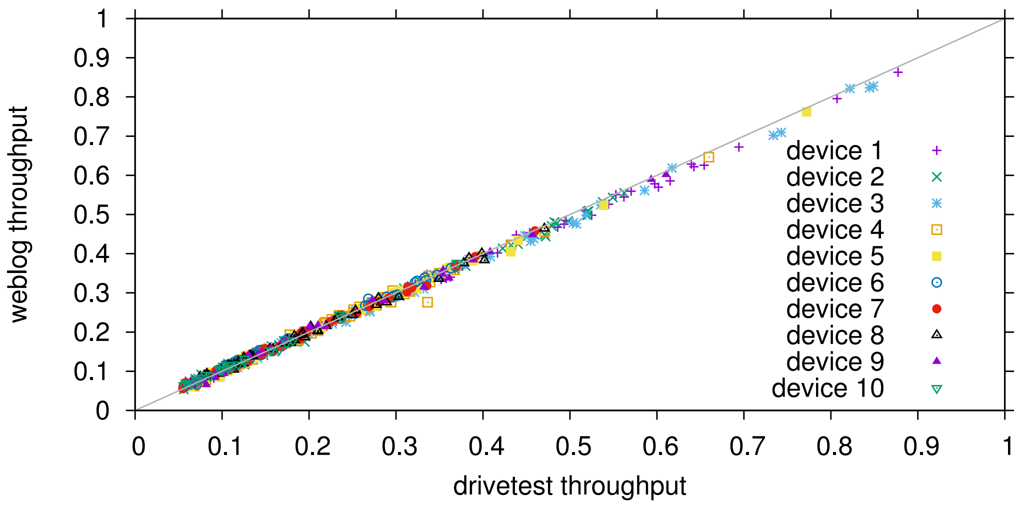


Figure 6.4.1: Weblog throughput validation using country-wide drive tests and various devices. The performance measured by the operator's middlebox matches the one measured experimentally.

Validation: We validated the fact that the weblog transactions can capture the resulting web QoE by performing large-scale drive tests. We discovered that the metrics correlate well with one exception: the weblog-based throughput estimation is only meaningful when considering larger downloaded objects ($> 700kB$). Therefore, in our analysis we filtered out smaller objects when measuring throughput. Fig. 6.4.1 demonstrates this correlation between hundreds of drive tests and the associated weblogs entries for different mobile devices. These results indicate that web transaction performance counters, as seen by the webproxy, are a good proxy for the conditions that are experienced by the end-users.
Per-sector metrics: The final step is to aggregate per requests metrics at a per sector and per-hour granularity (to match the KPI granularity). Therefore, we extract distribution related to the (min/max/avg and percentiles). In this work we provide correlations with respect to the median performance but we have internally considered other metrics too.

6.4.3 Combining the KPIs with the ground truth

At the end of this process we have hourly per sector information related to the ground truth **g** of i) *latency*, ii) *download throughput* and iii)*YouTube video streaming stalls*.

To make any statistically significant correlations we remove all samples where less than 20 web downloads were made by users in a given sector-hour. Overall, the data set contains more than *2 million of throughput and delay* samples and *7,000 video streaming* samples (there are fewer sectors that had more than 20 non-encrypted video downloads within one hour).

6.5 Evaluation

We start our evaluation with a high level comparison of the achieved correlation by all methodologies. Then, we drill down into the results of PSO and machine learning methodologies to better investigate their strengths and weaknesses. Finally, we investigate the intersection among the under performing sectors found by different methodologies, i.e., up to which extent they offer a different view on performance issues.

To reduce possible over-fitting, we randomly split each samples population in two halves: the first is used to identify the best solution (training), while with the second we assess the correlation with respect to ground truth (testing).

Notice that, for all results, weblogs QoE metrics are normalized with three distinct factors, one for throughput, one for latency, and one for video stalls, while hotspot scores are normalized in the range between 0 and 10. This allows to still have meaningful comparisons without revealing actual values which unfortunately cannot be publicly disclosed. Table 6.5.1 details the Spearman correlation coefficient ρ for all considered techniques. Results are averaged across 10 runs.

6.5.1 Overall results

Baseline (1): This corresponds to the achieved correlation with currently used weights and thresholds. Table 6.5.1 shows a small correlation. This means that the state of the

Approach	**Configuration**	**Delay**	**Thput**	**Stalls**
(1) Baseline	All KPIs	0.15	-0.14	0.07
(2) Baseline	Only data KPIs	0.19	-0.17	0.09
(3) PSO on S	Only weights	0.29	-0.26	0.17
(4) PSO on S	Only thresholds	0.30	-0.29	0.19
(5) PSO on S	Weights and thresholds	0.41	-0.36	0.22
(6) PSO on $\hat{S}$	weights	0.46	-0.42	0.23
(7) Non-linear	Random forest regression	0.40	-0.32	0.23

Table 6.5.1: Spearman's correlation for different optimization strategies between the resulting score and the ground truth (web delay, throughput, and video stalls). Positive correlation for values that increase as resulting hotspot metric increases (e.g., delay and video stalls) and negative when values decrease (e.g., throughput).

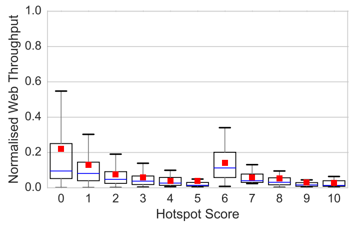
art solution indeed captures the performance of the sectors. In fact, recall the bimodal distribution between peak and non-peak hours seen in Fig. 6.2.2, and Fig. 6.2.3 detailing sectors triggering multiple KPIs. Both pictures were suggesting the presence of correlation which is now quantified in Table 6.5.1.

Interestingly, correlation is different for the considered QoE metrics, with video stalls presenting the smallest values. This is expected since video stalls depend on a number of other factors including users' device type, CDN providers, and is the metric for which we have the least amount of samples, i.e., the overall performance of sectors is expected to be associated to a varied set of services besides video streaming.

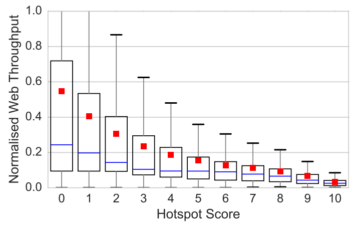
Baseline (only data KPIs) (2): As shown in Fig. 6.2.3, the default parameters of the baseline approach put emphasis on hotspots suffering from signal and voice issues. Given that we focus on understanding data-related QoE it is important to examine the correlation when *only data KPIs are considered*.

By only considering data KPIs in the baseline formula (2), we do notice a slight improvement with respect to (1), although the magnitude of the correlation is still small. This is an indication that a holistic approach which combines signaling, voice, and data might not be ideal to capture the performance of individual applications.

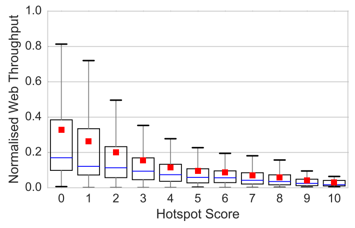
PSO on current hotspot formula ((3), (4), (5)) Applying PSO on the baseline formula improves correlation, but the entity of such improvement differs based on which parame-



(a) Baseline (all KPIs)



(b) PSO on $\hat{S}$: weights



(c) PSO on S: weights and thresholds

Figure 6.5.1: Detail PSO correlation results for throughput measurements.

ters are optimized. Specifically, optimizing only weights presents the smallest benefit (3), while the best solution is achieved optimizing both weights and thresholds (5) with a striking improvement factor of $\times 2.7$, $\times 2.5$, and $\times 3$ for delay, throughput, and video stalls, respectively. This is because PSO allows to identify an optimal set of parameters specific for each target metric, rather than enforcing a single configuration as for the baseline.

PSO on modified hotspot formula (6): As discussed in Sec. 6.3.2, the Heaviside function transform the raw KPI values into binary values (KPI triggered or not). Our intuition is that those values provide a wealth of information on performance, hence the raw KPI values should be directly exploited in the hotspot formula. We now see that when applying PSO on the modified hotspot formula $\hat{S}$ (Eq. 6.3), correlation further improves with respect to baseline by a factor of $\times 3.06$, $\times 3$, and $\times 3.2$ for delay, throughput, and video stalls, respectively.

Random forest regression (7): Finally, having assessed correlation for all discussed linear combinations, we explore implicit *non-linear* functions via random forest regression (Sec. 6.3.2). Results show that the obtained Spearman's correlation is not as good as the best PSO solution (6). The main reason is the nature of the objective function that is used in each algorithm: while in PSO we can set the optimization target to match our needs (in our case to provide a better ranking or Spearman's correlation), off-the-shelf regression trees depend on metrics such as mean squared errors that better correlate with ground truth values rather than the ranking. Still, it is interesting to notice that such algorithms already provide a significantly better solution than the baseline approach.

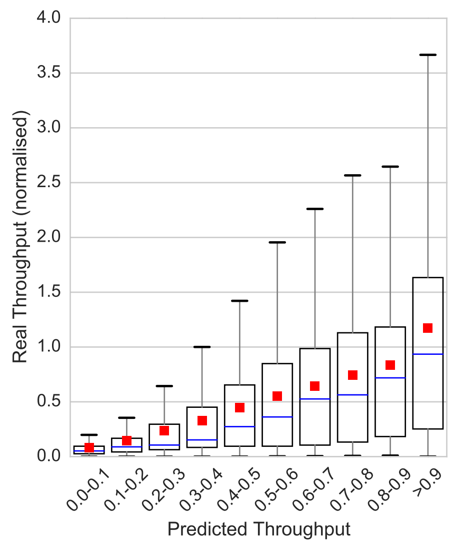
Takeaways*: Meta-heuristic and machine learning algorithms provide significant improvement over baseline approaches. It is also recommended to apply a simple linear combination of raw KPI values and to avoid the currently used thresholding mechanisms.*

6.5.2 Focusing on PSO

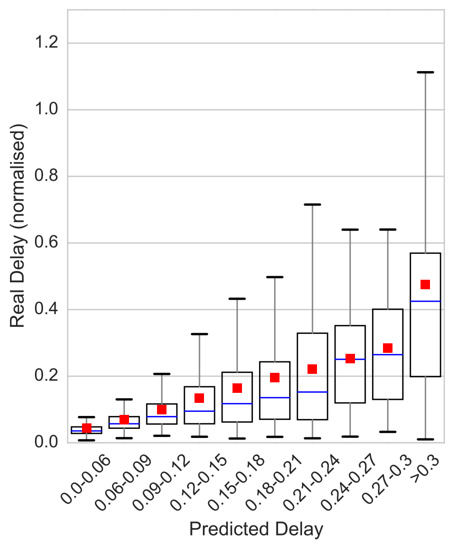
Fig. 6.5.1 shows the relationship between the hotspot score and the throughput for the baseline approach (Fig. 6.5.1a), the hotspot formula S optimized for both weights and thresholds (Fig. 6.5.1b), and the hotspot formula $\hat{S}$ without the Heaviside function (Fig. 6.5.1c). Specifically, we bin the performance score in bins from 0 to 10, and for each bin we plot the throughput distribution using box plots capturing 5^{th}, 25^{th}, 50^{th}, 75^{th}, 95^{th} percentiles. We further report the average of each bin with a dot.

Notice how the boxplots are relatively "flat" for the baseline approach, with a high concentration of sectors having a score of zero or six (Fig. 6.5.1a). This bimodal distribution, also noticed in Fig. 6.2.3, is the result of the (manual) tuning of weights and thresholds of the current monitoring system.

Conversely, PSO spreads the scores across all bins, better separating well-performing from poor-performing sectors. Notice also how median values for the two PSO optimizations (Figs. 6.5.1b,c) are very similar, but average values are better separated using the modified formula $\hat{S}$. More in details, while 37% of sectors have a score larger than 0 for Fig. 6.5.1b, this drops to 19% for Fig. 6.5.1c. Considering results for PSO on $\hat{S}$, sectors with score 10 have $\times$11.1 less throughput when compared with the sectors with score 0. In contrast, the baseline method (Figs. 6.5.1a) shows significantly smaller separation, namely $\times$5.7. Finally, sectors with score 0 have $\times$2.6 the throughput when compared to the baseline scoring, demonstrating that this method can better isolate such cases. Results for delay and video stalls are similar but we do not report them due to lack of space.



(a) Throughput



(b) Delay

Figure 6.5.2: Estimating web delay and throughput using the KPIs. We observe that the average delay matches the predicted value but there is a wide distribution within each bucket.

6.5.3 Focusing on Random Forests

Table 6.5.1 shows how PSO over-performs random forest regressions in terms of correlation due to the difference in the optimization function. However, the advantage of regression is that it provides an estimation of the actual conditions (e.g., estimates the expected delay, throughput an video stall *values*).

Fig. 6.5.2 shows these estimations of throughput (a) and delay (b) obtained with the regression. Specifically, we split regressed values in 10 bins, and for each we plot the distribution of ground truth values using box plots. In theory, a very good model would have very narrow distribution centered around each decile. While averages are indeed very close to the real value, each bin presents a wide distribution of values, i.e., the model is still affected by some noise. Nevertheless, we can clearly separate performing from under-performing sectors.

Such observation motivated us to perform an additional experiment: create a *classifier* based on ground truth buckets. In fact, despite the fact that regression allows to have an estimate of QoE, as a first approximation it can be sufficient to divide sectors in classes based on performance 'labels' (e.g., poor, medium, good). To achieve this, we need to partition regression values in classes (we obtained the raw thresholds for the split through conversations with the operator network teams).

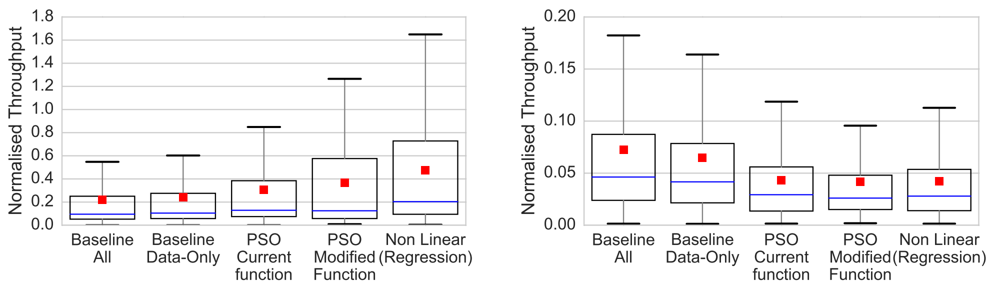
The selected thresholds however create highly imbalanced classes: we can rarely find samples that belong to the high delay or the low bandwidth case (most of the sectors exhibit medium or high performance even during peak-hours). To address this issue, we use a balanced tree to assign a higher weight to classes having smaller number of samples (low throughput or high latency). The reasoning is that providers mostly care about discovering under-performing sectors and some miss-classification between high and medium performance classes is tolerable. The confusion matrix in Table 6.5.2 reports the obtained classification accuracy for both throughput and latency. The best performance is obtained for good-vs-good and poor-vs-poor, i.e., the classes we aim to separate, while the major confusion comes from medium performance values being confused with poor performance values.

Real\Est.	**poor**	**medium**	**good**	**poor**	**medium**	**good**
poor	85%	14%	1%	71%	26%	3%
medium	32%	59%	9%	32%	42%	26%
good	5%	21%	74%	2%	16%	82%

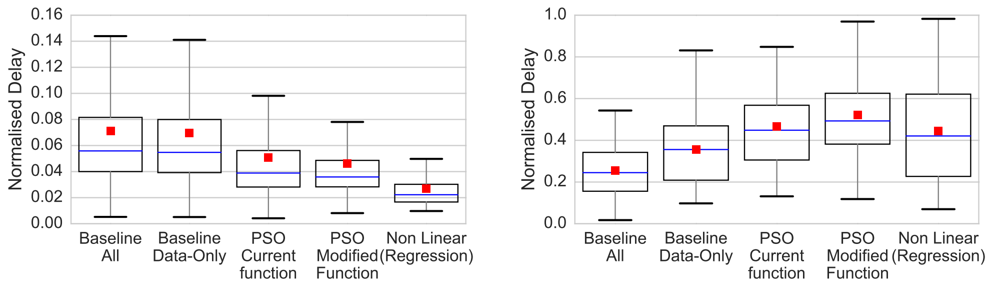
Table 6.5.2: Normalized confusion matrix for real v.s. predicted throughput (left) and delay (right). Ideally, all samples should be on the diagonal.

Takeaways: *Off-the-shelf machine learning regression techniques offer a slightly lower correlation with respect to PSO. However, they enable an estimation of the actual values of performance metrics (instead of a score) which is, as a first order approximation and without very fine-grained tuning, sufficiently accurate to separate correctly performing from under performing sectors.*

6.5.4 Comparing top/bottom performing sectors



(a) throughput for top (left) and bottom (right) sectors

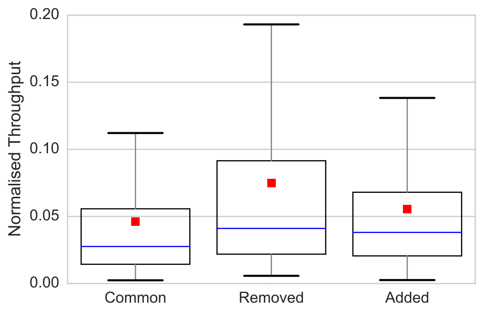


(b) latency for top (left) and bottom (right) sectors

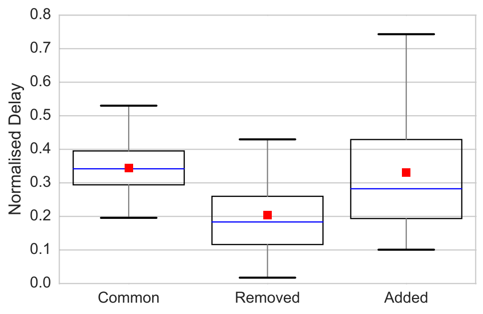
Figure 6.5.3: The actual throughput and latency of the sectors that ended up at the top or at the bottom 500 of each ranking. Results for video are similar.

The final objective of the discussed methodologies is to rank sectors based on performance. In our case, bottom performing sectors are the ones with the highest delay, lowest throughput and highest percentage of videos that experienced stalls. However, it is important to identify healthy sectors as well, since they can provide further information to understand the causes behind poor performance (e.g., comparing sites configurations, locations, etc.)

Comparing different approaches: We focus on the top and bottom 500 sectors, and for each group of sectors, Fig. 6.5.3 shows the distribution of the ground truth throughput



(a) Throughput



(b) Delay

Figure 6.5.4: Comparing the PSO optimized vs. baseline rankings for the top 500 samples. *Common* sectors appear in the top 500 of both rankings, *removed* sectors do not appear in the optimized ranking and *added* sectors did not appear in the default ranking.

(a) and latency (b) using box plots. As previously observed, the application performance of sectors that end up at the top and bottom of the rankings varies significantly with each methodology. We observe that PSO applied on the modified hotspot formula $\hat{S}$ (6) and random forest regression (7) achieve significantly better results in identifying the well-performing sectors compared to the baseline (left plots). The main reason is the nature of the original function: to identify faults, the baseline methodology would just assign a score equal to zero to most of the average and good performing sectors since no KPI thresholds are defined conservatively. Instead, PSO)6) is better at identifying poor-performing sectors in each of the applications (including for video stalls which results are not reported due to space constraints). These results further corroborate the overall comparison discussed in Sec. 6.5.1. Considering instead the bottom performing sectors (right plots), PSO (6) presents the most consistent performance across metrics, while random forest regression suffers of lower accuracy in assessing latency performance.

Comparing baseline and best solution found: In Fig. 6.5.4 we further quantify the intersection between the baseline and the best PSO approach (6) rankings. In particular, we focus on the worst 500 performing sectors and compare the QoE metrics distribution for sectors found in both, only PSO (added), and only baseline (removed) rankings. First of all,

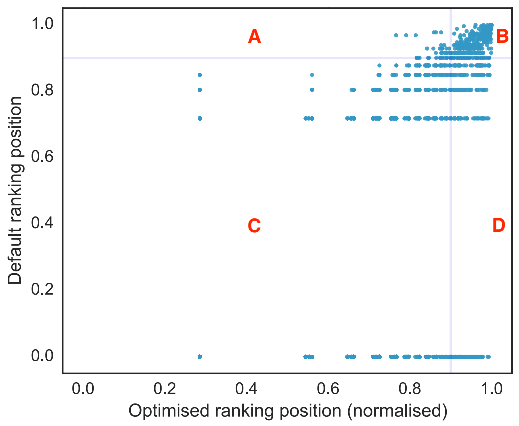


Figure 6.5.5: Comparing baseline vs. optimised rankings of the 10 percentile worse-performing sectors in terms of throughput ground truth. Ideally, both rankings should place them in their top 10 percentile too.

only 56% (49%) are found by both techniques for throughput (latency). On closer inspection, these are sectors that exhibit poor performance in all possible KPI classes (voice, data, signaling) and show the lowest throughput and delay. Notice how the remaining sectors captured only by the baseline approach (*removed*) present on average a higher throughput (lower latency) with respect to the common sectors, whereas the ones discovered (*added*) show performance that is as poor as the ones in the common set.

To conclude, Fig. 6.5.5 shows a scatter plot of the sectors' rank position between these two methods for the worse-performing 10% sectors in terms of throughput. An ideal ranking methodology would place these sectors in the top 90% (e.g., areas *A* and *B* for (1) and areas *B* and *D* for (6)). As expected, the majority of the sectors cluster at area *B*, indicating that both rankings are able to identify a good part of the under-performing sectors. More interestingly, areas *C* and *D* present sectors having very small scores according to the baseline approach. The reason is that the baseline methodology assigns a large number of sectors with score zero as none of the KPIs "triggered". This is compatible with the nature of the original function: to identify important faults in the network rather than sectors with poor throughput. We further notice that PSO puts most of these sectors at the top 60% of the ranking with quite a lot being correctly in the top 10% (lower and middle parts of area *D*). Results for delay and video stalls are similar (not shown due to space limitations), demonstrating that this methodology can better capture individual application performance. Finally, notice that points in area *B* do not perfectly lay on the bisect line, i.e., under performing sectors are spot as an aggregate, but severity is differently captured by the two methodologies.

Takeaways: *By applying a data-driven methodology like PSO, operators are empowered with a flexible tool that use the already collected KPIs to improve their view on under-performing sectors.*

Category	Throughput	Latency	Stalls
Signaling	2%	37%	3%
Voice	0%	1 %	2%
Availability	8%	30%	30%
Congestion	88%	26%	36%
Radio	2%	6%	29%

Table 6.5.3: Decomposing the most important features for each QoE component into KPI categories (information gain).

6.6 Discussion

Selecting QoE metrics: While we used KPIs to estimate web experience (delay, throughput, video), this methodology can be used to build a ranking for other QoE metrics that can potentially correlate with the KPIs. These can include dropped calls, VoIP performance, specific application performance (e.g., facebook, gaming, etc) or even customer satisfaction. The main requirement is to have enough samples of ground truth to build the model.

State of the art solutions try to provide a single performance scoring function to capture different types of problems. To quantify this effect, we focus on the worst 500 sectors ranked by the best PSO solution (6) and compute the fraction of the score associated to each KPI class. Table 6.5.3 compares the importance of the KPI classes for the three QoE metrics considered. As we observe, *KPIs' importance significantly varies depending on the QoE metric*. Throughput, depends almost exclusively on KPIs that have to do with congestion and availability of high-speed channels. Latency is a more complex phenomenon as it depends on signaling failures (e.g., failures to establish a dedicated channel), availability of resources and congestion. Finally, interestingly enough, even if video streaming is a throughput driven service, the importance of KPIs when capturing stalls significantly differs from generalized throughput: Video quality also depends on the radio conditions (interference, noise, etc) as these can create jitter and download stalls.

These results manifest the need of building an adaptive data-driven approach to exploit the, already collected, KPIs in order to better understand both historical and real-time performance of each application at each sector of the network.

Building a generic ranking:

We envision a system where multiple objective metrics (individually studied to optimize correlation with ground truth) are optimally combined into a single *Sector Priority Index (SPI)*. In this work, we focused on network measurements since, currently, sector rankings are based on a KPI scoring function. However, when defining the importance of a sector, one should consider also other information sources such as network sites profitability, CAPEX/OPEX investments, country regulations, etc. How to synthesize such SPI is an open question. Nevertheless, we believe that the optimization methodology discussed in this work can support the creation of such index.

Constant evolution: Network priorities and user QoE expectations are constantly changing as new usage paradigms emerge. Furthermore, the network is constantly evolving too, with 5G deployments being just a few years away. One of the implicit benefit of data-driven approaches is that they empower analysis automation. In our case, we envision a system that periodically adjust the hotspot score function parameters to capture long and short term trends. For instance, it can automatically incorporate seasonality effects. Similarly, automation can provide fine grained configuration across time (e.g., extract different optimization parameters to capture separately morning, afternoon, and night hotspots) or space (e.g., investigate separately residential areas from downtown districts). Notice that there is is no currently available solutions capable to achieve such a flexibility.

From KPIs to ground truth: In this work, we used weblogs to build an estimation about individual QoE components such as web throughput, delay, and video stalls. Building such a dataset requires some instrumentation (e.g., middle-boxes), and exhibits significant computation complexity as we have to bridge the gap of per user (or per flow) performance with per-sector metrics. Therefore, it is prohibitive to run such processes in a continuous manner. Our work allows to use relatively sparse samples of ground truth to associate them with KPIs that are already collected. Furthermore, other data sources can be used for ground truth: active tests (e.g., drive tests), crowd-sourced data such as OOKLA [66] and user surveys, or even quality metrics coming through collaboration with application developers and CDNs.

6.7 Summary

With the explosion of mobile internet traffic and the ever-evolving user expectations, it is of paramount importance for mobile operators to be able to quantify the performance of their network in terms of the delivered application services quality. We applied a novel data-driven methodology that builds upon the already collected sector KPIs and bridges them with different QoE metrics. By doing so, the system empowers operators with an automated methodology that provides better visibility on under-performing sectors. Moreover, our results indicate that the currently used solution that is based on thresholding is sub-optimal to identify critical sectors. This opens new areas or research for monitoring solutions enriching the quality and accuracy of the network performance indicators collected at the network edge.

7 Detecting Network Performance Issues with Contextual Anomaly Detection

7.1 Introduction

Network performance anomalies can be defined as abnormal and significant variations in a network's traffic levels. Being able to detect anomalies is critical for both network operators and end users. However, the accurate detection without raising false alarms can become a challenging task when there is high variance in the traffic.

To address this problem, we present in this chapter a novel methodology for detecting performance anomalies based on contextual information. The proposed method is compared with the state of the art and is evaluated with high accuracy on both synthetic and real network traffic.

More specifically, we first introduce a novel methodology for detecting contextual network performance anomalies and present the benefits of detecting anomalies using the proposed Contextual Anomaly Detection (CAD) algorithm. Next, we propose methods for improving the state-of-the-art algorithm in terms of accuracy but also scalability. Finally, we evaluate the context construction and anomaly detection stages of the methodology with both synthetic and real network data.

7.2 Contextual Anomaly Detection

This section presents the two distinct stages of the proposed methodology, i.e. the context construction and the anomaly detection phase.

7.2.1 Context Construction

The purpose of this phase, is to cluster together all instances that exhibit similar temporal characteristics across a longer period of time. This results in grouping together into a single context all timeseries that have similar temporal variances within a selected construction time window T_c and makes it easier to later understand deviations from the nominal context behavior.

In order to construct the context, it is necessary to calculate the pair-wise similarity between the instances in the dataset. To accomplish that, we need to employ an accurate timeseries distance measurement method.

However, when dealing with network performance measurements, probe failures, outages or irregular sampling rates may lead to sequences with missing samples, while differences in speed between sequences can occur due to the propagation delay of an anomaly. As a result, traditional methods such as the Euclidean distance cannot be relied upon for accurately calculating the distances between instances.

To address this issue, the pair-wise similarity between instances is calculated using Dynamic Time Warping (DTW) [67]. DTW is used to determine the distance between two sequences that may vary in speed, by warping them in the time dimension in order to properly align them and get a more accurate measure of their distance.

DTW is a very suitable solution for such cases since it can aid in reducing the number of False Positives (FP) and False Negatives (FN) and lead to higher accuracies as we will show in more detail in Section 7.4.1.

We specifically use the FastDTW [68] implementation of the algorithm which has an O(N) complexity as opposed to the O(N^2) complexity of the original implementation.

After the distances between all the sequences are calculated for a given T_c, the k-Nearest Neighbors (kNN) algorithm is used to classify them into different contexts. kNN has been proven to be a very effective solution when classifying time series sequences based on their distances. Moreover, the value of k is set equal to 1 in order to minimize the model's bias and increase the confidence of the predictions.

7.2.2 Anomaly Detection

After clustering a given instance into a context of similar instances based on its behaviour on a large time window, we want to examine whether it deviates from the average context behaviour for shorter periods of time.

Two generic examples of contextual anomalies are shown in Figures 2.4.1a and 2.4.1c, where athough the red sequences were previously identified as members of their context during the context construction phase for a larger time window, smaller parts of the sequences exibit a very different behavior from the respective context.

We therefore propose the following methodology for identifying any context members that show anomalous behavior within a given scoring time window T_s such that $T_s \leq T_c$.

Initially, we keep only the members of the context that were identified as TP in the context construction stage. Next, DTW is used to calculate the distance matrix of all the members of the context C_{T_s} for the new scoring window T_s. Finally, we define the Context Mean Distance (CMD) as the average value of the C_{T_s}.

Contextual Anomaly Definition: A context member O_{T_s} is identified as anomalous if its average distance from the rest of the context members is larger than the CMD plus one standard deviation.

$$\overline{dist(C_{T_s}, O_{T_s})} \geq \overline{dist(C_{T_s})} + \sigma(dist(C_{T_s})) \tag{7.1}$$

One of the most notable advantages of this approach is that it allows the detection of all the anomalies in a context for a specific scoring window in a single step. In other words, it is not required to evaluate each context member separately for anomalies, which can result in great performance benefits, specially when dealing with large contexts.

7.3 Datasets

7.3.1 FCC Data

The algorithm's performance is evaluated with the dataset obtained from the FCC Measuring Broadband America project [69]. The project aims to measure and report the performance of 13 major U.S. fixed and mobile broadband providers.

The measurements are collected with the aid of SamKnows [70] platform, where active measurement probes are installed in the home networks of broadband clients. The probes measure the performance of the last-mile by running tests against servers located in the providers' core network or that are part of the SamKnows infrastructure.

The FCC data contains 13 performance metrics which can be found in the related technical appendix [71]. For the evaluation we obtain a copy of the average RTT measurements which consists of measurements collected over 8 consecutive months in 2015. Apart from the average RTT, the data also includes the whitebox's ID, the server's FQDN, a location ID and the number of successful and failed tests.

Data Analysis

Before doing the evaluation with the FCC data in Section 7.5, we run a preliminary analysis of the dataset in order to verify that there is correlation between the instances based on the context they belong to.

To this end, we group all instances in the data based on the server domain that was used to perform the measurements. Hence, each group corresponds to all clients that are connected to the same server. Clients connecting to the same server share the same network provider and are located in the same geographical area. These peer groups are equivalent to contexts since the members in each group are expected to have similar performance characteristics.

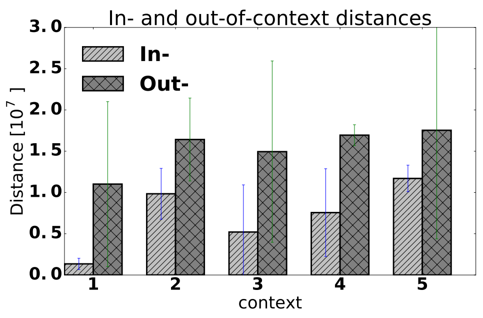


Figure 7.3.1: Average and std. deviation of the in- and out-of-context DTW distances.

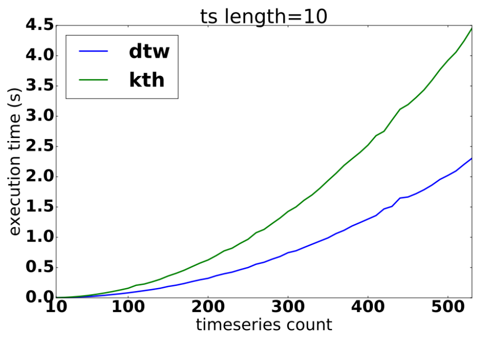


Figure 7.3.2: K_{th} vs. DTW execution t while increasing the sequence number.

Next, for 5 randomly selected servers we calculate the average and the standard deviation of the DTW distances among the members in the same context and the distances of the members with all the instances outside the context.

Figure 7.3.1 illustrates the average in- and out-of-context distances and their standard deviation. In all the 5 cases there is higher similarity among instances of the same context. In contrast, when comparing with out-of-context instances the distances are much higher in comparison. This shows that peer groups where the members have a clear correlation with each other do exist in the FCC dataset, which makes it very suitable for evaluating our methodology since our algorithm aims to automatically group together such behaviors based on the DTW distance.

7.3.2 Synthetic Data

In general, it is typical to not have ground truth in real data and the FCC data is no exception to this rule. A common approach to address this in order to properly evaluate the algorithm's performance, is to use an artificially generated dataset where the ground truth is known. Such a dataset, needs to be created in a way that it accurately represents real network measurements, to ensure that the evaluation results can be generalized to real data.

The synthetic data should consist of multiple contexts for the purpose of evaluating the algorithm's context detection capabilities. To find the most accurate model and the correct configuration parameters for creating each context, we take the average RTT measurements for the month of August and group by the server hostname. In this way, each group represents a different context with RTT measurements from multiple clients against the same server.

To identify the statistical model that best fits the data in each context, we apply the Goodness of Fit (GoF) methodology which allows us to compare the distribution of the data with other well-known distributions. The metric used to determine the GoF is the Sum of Squared Errors (SSE).

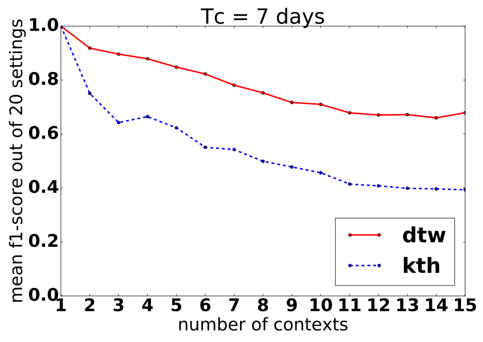
The GoF was performed for the contexts that correspond to the ten servers with the largest number of clients and measurements. The ranking of the best fitting distributions based on the SSE score, revealed that the model which best describes the data in each context is the *Johnson's* S_U [72].

Next, each set of parameters that were obtained from fitting each context is used for generating equal number of synthetic contexts. More specifically, every context consists of 100 time series that were created using Johnson's S_U and the corresponding settings. All the time series contain 1 sample per hour and a total length of 1 week.

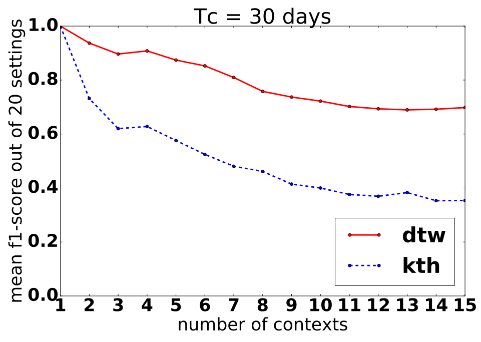
7.4 Evaluation with Synthetic Data

7.4.1 Context Construction Evaluation

For the purpose of comparing the performance and accuracy of DTW with the state of the art in CAD, the evaluation is performed with two different distance metrics, first using the



(a)



(b)

Figure 7.4.1: mean f1-score comparison for context construction with k-NN using K_{th} distance and DTW as distance metrics. $T_c = 1$ week (left) and 1 month (right)

K_{th} distance and then with DTW.

The K_{th} order statistic distance was presented in the work of Chen et al. [73], as a method to reduce the FP and FN when using the Minkowski distance to construct the context of a target time series.

Before doing the evaluations, each sequence is labeled according to the context it belongs to in order to provide the context construction ground truth. Then, 80% of the data is used for training the classifier and 20% for testing.

The two evaluations are repeated three times, each time modifying one of three variables, i.e. the number of context members, the context count and the construction window T_c. Each time a variable is gradually increased while the other 2 remain fixed.

Moreover, the data for each context is generated using 20 different settings in order to eliminate the possibility to get results that are specific to a particular configuration.

K_{th} Distance vs. Dynamic Time Warping

Figure 7.4.1 shows a comparison of the accuracy achieved using the two methods. Both graphs show the evolution of the f1-score when using the K_{th} distance and when using DTW, as the number of contexts increases for $T_c = 7$ days (left) and for 30 days (right).

In both cases the accuracy improvement when using DTW is evident. When the context count is equal or greater than 3, the accuracy gain is constantly above 20% and can reach up to 35% for $T_c = 7$ days.

The reason why the K_{th} distance is outperformed by DTW, is because the alignment of the two compared time series is done by rearranging their samples based on their point-wise distance. This can change the shape and the sample order of the sequences and return a less accurate distance value.

In contrast, DTW performs time warping to find the optimal alignment between the sequences without sample rearranging. Hence, when compared to K_{th}, DTW is a more reliable distance calculation method.

The next part of the comparison is done with regards to the execution time of the two algorithms. Both algorithms were implemented in Python and all the tests were executed on a Linux PC with an Intel Core i7 @ 3.4GHz.

Figure 7.3.2 shows the execution time comparison when increasing the number of instances in the dataset while keeping a fixed time series length. Here we observe that DTW constantly outperforms the K_{th} distance and when the time series count reaches 500, DTW is already twice as fast.

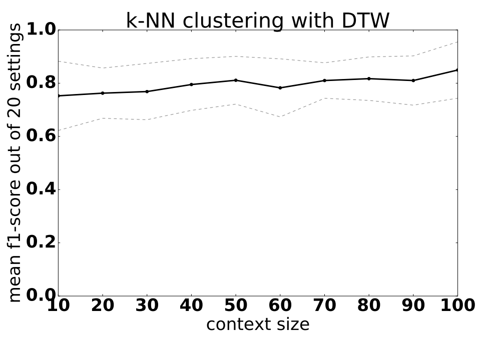
Context Construction with 1NN-DTW

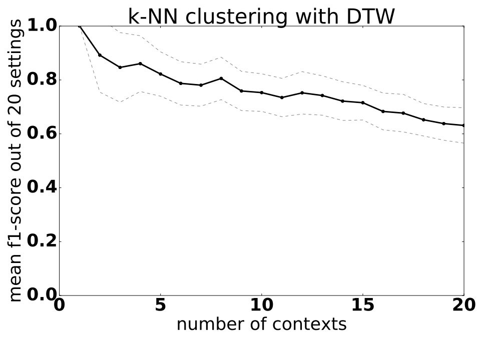
The first phase of the context construction evaluation is performed with an increasing context size, while the number of contexts are fixed to 5 and the T_c window is set to seven days. The number of members in each context is increased from 10 to 100 in steps of 10. In Figure 7.4.2 (left), the thick black line shows the mean f1-score obtained from the 20 different settings that were used in each of the steps. The grey dashed lines represent the mean $\pm$ one standard deviation.

These results show that the overall accuracy is improved as the context size is increasing. Specifically, we find that a 10% improvement is achieved when comparing the accuracy between the min and the max context size.

Contexts with larger number of members are identified more accurately, due to the addition of more observations to the training set without increasing the complexity of the data since each new observation has common characteristics with the other members.

Next, the evaluation is repeated with an increasing number of contexts, while the context



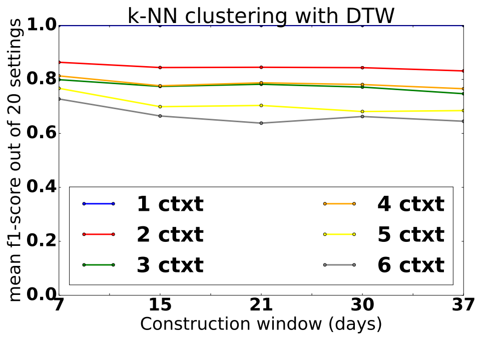


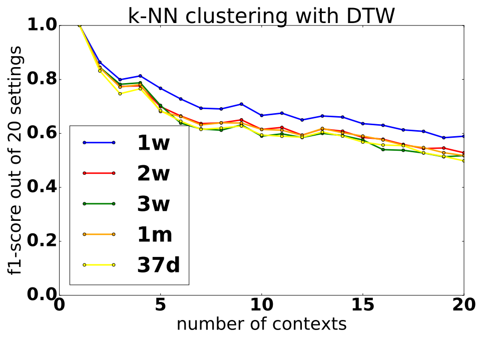
(a) **(b)**

Figure 7.4.2: Accuracy of the context construction with increasing context size (left) and increasing context count (right)

size is fixed to 100 members. The results from this evaluation that can be found in Figure 7.4.2 (right), show that there is a negative correlation between the accuracy and the number of contexts.

More specifically, we find that by increasing the context number the complexity of the dataset increases as well. This has a negative impact on the overall context detection accuracy since the algorithm has a more difficult task to classify larger number of observations with different characteristics.





(a) **(b)**

Figure 7.4.3: Accuracy of the context construction when increasing the T_c and the number of contexts.

The last phase of the evaluation is performed while modifying the length of T_c with values equal to 7, 15, 21, 30 and 37 days. The plots in Figure 7.4.3 show the accuracy for each T_c value while the number of contexts is increased. Both plots show that the increment of the construction window has a very small effect on the accuracy. This result is logical since a time series in a context is synthesized with the same settings and the window size only determines then number of samples that the series will have.

7.4.2 Anomaly Detection Evaluation

To evaluate the accuracy of the anomaly detection algorithm, it is required to have the ground truth that identifies the parts of the data which correspond to anomalous events. However, the FCC data does not provide such information and therefore it is not possible to verify if the detection algorithm is making correct predictions.

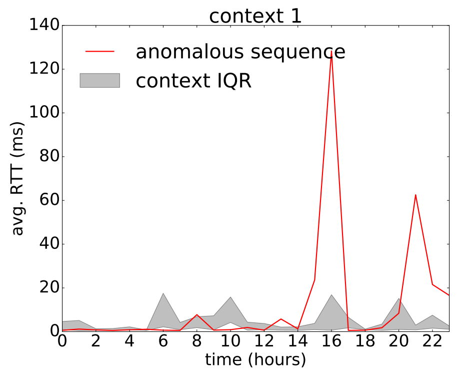
To overcome this problem, we follow an approach based on the statistical analysis of the contexts and anomalies, that allows us to evaluate the accuracy of the anomaly detection methodology. In more detail, we calculate the Inter-Quartile Range (IQR) for each context and each T_s. The IQR represents the data between the 25th and 75th percentiles and can be used to verify if the samples of a time series deviate from the main body of the distribution of the context data.

According to the definition of the IQR and given that a context member is following the behavior of the entire context, at least 50% of a member's samples must be within the IQR. In contrast, a much higher percentage is expected for outliers with anomalous behavior.

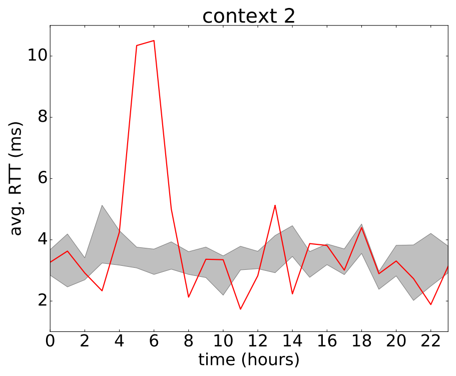
Finally, we identify the anomalies for each context and each scoring window using the approach described in 7.2.2. Then, we determine the percentage of sample points of each anomalous sequence that are outside its context's IQR.

The anomaly detection evaluation is done using 5 different contexts which are obtained from the context construction phase. For each context, only instances that were identified as TP are kept, to avoid adding outliers in the contexts.

The data of each context is split into smaller subsets by dividing the construction window in n equal scoring windows such that each $T_s = T_c/n$. For each context, the evaluation is repeated for all n consecutive scoring windows.



(a)



(b)

Figure 7.4.4: Examples of anomalies for 2 different contexts the respective context IQR.

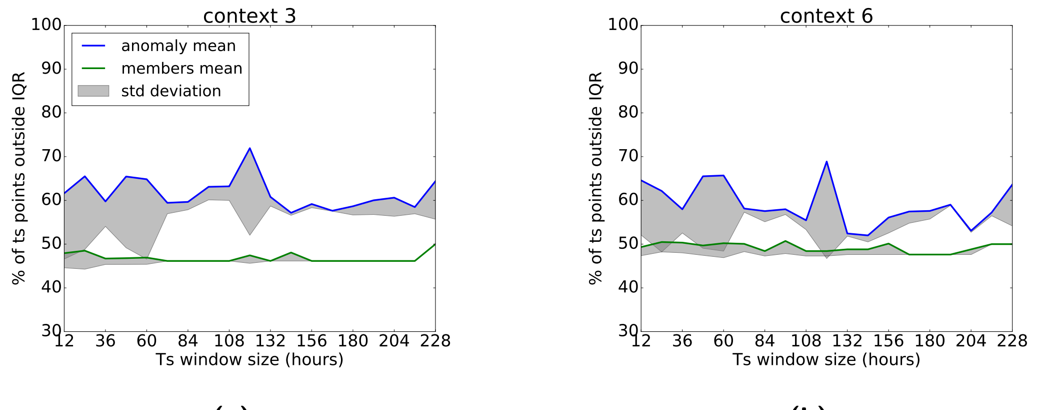
Additionally, the length of the scoring windows is gradually incremented in steps of 12 hours, to generate evaluation sets with fewer slices that contain longer sequences.

Figures 7.4.4a and 7.4.4b show anomalies that were detected in contexts 1 and 2 respectively. In 7.4.4a, the greatest part of the red sequence is within the IQR except from 2 spikes which are between 5 and 10 times higher than the corresponding IQR points. In 7.4.4b though, a much higher number of points of the anomalous sequence are outside the IQR, but the magnitude of the variations is significantly smaller.

These two examples show that the proposed methodology can work very well when detecting either short and bursty anomalies but also events that are characterized by small but constant variations.

The two plots in Figure 7.4.5 make a comparison between the characteristics of two different contexts and their detected outliers. The y axis in the plots represents the percentage of timeseries points outside the context IQR while the x axis shows the different T_s sizes that were used to perform the evaluation. Each point of the blue and green lines corresponds to the mean percentage of point outside the IQR for the outliers and context members respectively.

The mean is calculated across all outliers or members for the given T_s and for all the subwindows equal to T_s that fit in the entire context construction window. The grey areas



(a) **(b)**

Figure 7.4.5: Comparison of the characteristics of the detected anomalies and their respective context.

are equivalent to the standard deviation of the percentage of points outside the IQR for each time window and show how much the values can differentiate from the mean.

In both examples in Figure 7.4.5 it is clear that the the detected outliers have a much higher number of points outside the context IQR. At the same time, we observe that the context members have constantly more than 50% of their points inside the IQR.

This findings strongly support the validity of the anomaly detection methodology, since they show that there are significant differences in the characteristics of the sequences that were identified as anomalous and those that were not.

Finally, we run the evaluation with 20 different contexts to get the overall accuracy of the algorithm. A detected anomaly is a TP if 50% or more of its samples outside the respective context IQR, otherwise it is considered a FP. Moreover, a context member sequence with more than 50% points outside the IQR is marked as False Negative (FN).

From the total number of TP, FP and FN for each context, we calculate the algorithm's accuracy in terms of Precision, Recall and f1-score. More specifically, the average Precision, Recall and f1-score from the evaluation with all the contexts are 0.97, 0.81 and 0.88 respectively.

The high precision indicates a very small number of FP which in turn means that the algorithm is capable of accurately detecting the vast majority of the anomalies in each context. The smaller Recall value though, shows that there is an increased number of FN, which corresponds to a higher number of anomalies that were falsely identified as context members. The lower Recall values are attributed to sequences where the anomaly is very short in duration and the rest of the sequence is within the IQR limits.

The overall good accuracy indicated by the f1-score, can be further improved by fine-tuning equation (7.1). The threshold of 1 standard deviation can be increased to increase the Recall values and the overall accuracy.

7.5 Evaluation with the FCC Data

7.5.1 Context Construction Evaluation

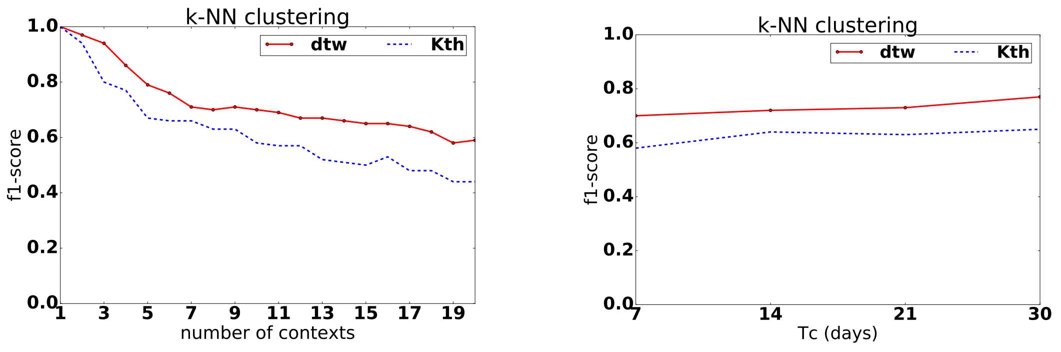
The evaluation with the FCC data is done using the average RTT measurements from August 2015 and following the same approach as with the synthetic data.

The evaluation is performed for different number of contexts, while the construction window is fixed to 1 week. Each context is defined as the collection of time series that correspond to measurements against the same server. In this way the context represents clients in the same geographical area that are using the network of the same provider.

In contrast to the evaluation with the synthetic data, we do not initially modify the number of members in each context. The number of members is adjusted when applying class balancing by means of undersapmling before the training phase. A balanced training set where all the classes have equal number of instances is necessary for creating a model that is not biased by under- or over-represented classes.

Next, we evaluate again using 1NN-DTW and we compare the benefits from using DTW over the K_{th} distance. The process is repeated while increasing the number of contexts in the dataset from 1 to 20 (Figure 7.5.1 (left)) while the construction window is fixed to 1 week.

From the figure we see that the use of DTW always results in a higher accuracy score as compared to the K_{th} distance. More specifically, the improvement from using DTW instead of K_{th} can reach up to 20% while the average accuracy gain throughout the evaluation is approximately 11%.



(a) **(b)**

Figure 7.5.1: f1-score comparison for the context construction evaluation with the FCC data. Using k-NN with k_{th} distance and DTW as distance metrics, while increasing the number of contexts (left) and the construction window (left).

The second phase of the evaluation is performed with a fixed number of 10 contexts and an increasing T_c from 7 to 30 days in 7 day steps. Figure 7.5.1 (right) shows the accuracy in each step for both DTW and K_{th} distances.

Again the results indicate that there is significant improvement in the context construction accuracy when using DTW. Similar to the findings in the synthetic data evaluation, we find that the length of the construction window has a small impact on the overall accuracy. Moreover, we see that the accuracy is improving when the T_c is increased, while a $T_c = 30$ days can lead to approximately 10% accuracy gain when compared to the respective result for $T_c = 7$ days.

The performance increase for larger time windows which was also observed in Section 7.4, is attributed to the information gain obtained by introducing longer sequences. This allows a more precise reconstruction of the context by the classifier, since the distances are calculated more accurately when considering a larger part of the sequences.

Overall, the results in this section show that the context construction can be performed successfully with real network measurements and maintain satisfactory accuracy even when the dataset consists of a large number of contexts.

Moreover, we verified with the FCC data as well that there are significant improvements in accuracy when using DTW instead of the K_{th} distance.

7.5.2 Anomaly Detection Evaluation

Following the same approach as in Section 7.4.2, five different contexts are obtained from the context construction phase and for each context the anomaly detection evaluation is repeated for n consecutive scoring windows. In turn, after each set of evaluations is complete, the value of n is incremented in order to get more but shorter scoring windows.

Figure 7.5.2 illustrates one detected anomaly for each of the first two contexts for $T_s = 24$. The anomaly in Figure 7.5.2a has smaller variance but almost the entirety of the points of the sequence are outside the IQR of its context. Figure 7.5.2b on the other hand, corresponds to an outlier that has more points inside the IQR but includes spikes 20 and 10 times higher than the IQR upper bound respectively.

This verifies our findings from the evaluation with the synthetic data and shows that the algorithm is capable of identifying anomalies that are longer in duration even if their deviation from the IQR is small, but also those that are short but large in magnitude.

Both these types of anomalies can be equally challenging to detect. Small variances that extend over longer time periods are difficult to distinguish from their context, especially when the context itself is ruled by high variance. Moreover, sparse and short-lived bursts often do not make a large enough impact on the overall variance of the time series in order to be identified as anomalies.

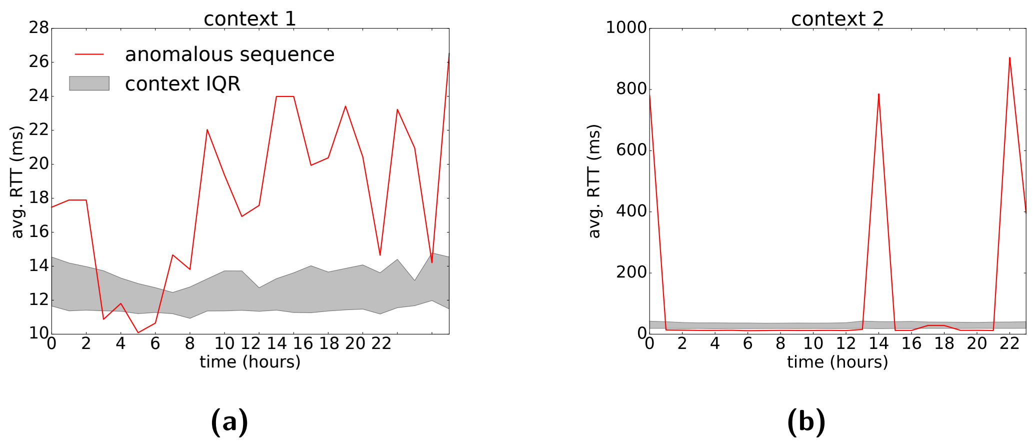
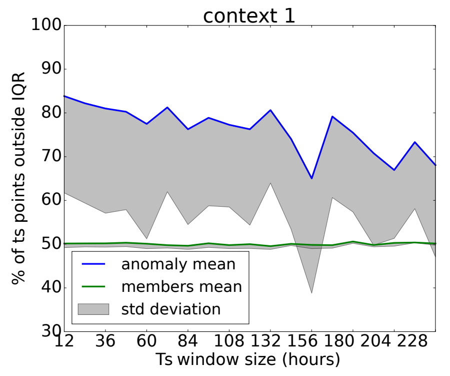
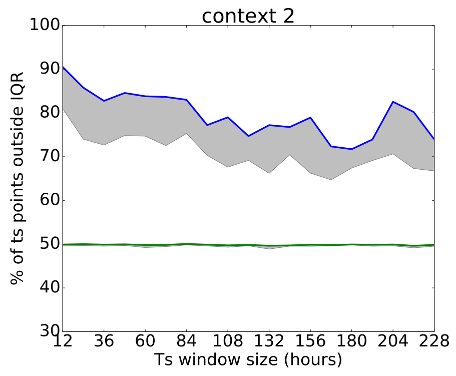


Figure 7.5.2: Examples of anomalies for 2 different contexts the respective context IQR.



(a)



(b)

Figure 7.5.3: Comparison of the characteristics of the detected anomalies and their respective context.

Additionally, in both Figures 7.5.3a and 7.5.3b we see that all context members have approximatelly half of their points inside the IQR with the variance being very small. The mean values close to 50% and the small deviations show that there are almost no outliers falsely detected as context members. This is an indicator that significantly increases the confidence for the algorithm's good performance.

Also, in both cases there is a clear distinction between the characteristics of the outliers and the context, where the outliers have mean values within 70% and 90%.

In all examples we observe a clear difference between the characteristics of the outliers and the characteristics of the context. This is an important observation that reveals that in contrast to the context members, a significant part of all the outliers are outside the IQR area and a strong indicator that the algorithm can accurately detect anomalies.

Moreover, the graphs in Figure 7.5.3 show that the T_s length has a small but noticeable impact on the anomaly detection, since the percentage of points outside the IQR decreases as the window size is increased.

Similar to the evaluation with the synthetic data, the anomaly detection is evaluated with 20 different contexts in order to get the overall accuracy of the algorithm. The resulting average values for the Precision, Recall and f1-score respectively are 0.96, 0.7 and 0.8.

Overall, the anomaly detection accuracy with the FCC data is at the same levels with the synthetic evaluation. However, there is a drop in the Recall value which indicates an increase in the number of FN. This is attributed to the higher number of small anomalies in the FCC data in terms of both duration and magnitude, that are more difficult to be detected by the algorithm. The detection thresholds can be adjusted accordingly, to more accurately detect this type of anomalies and improve the overall accuracy.

7.6 Summary

In this chapter we presented a novel approach for detecting network performance anomalies using contextual information. We have shown that not only this method can be successfully applied in both synthetic and real network traffic but it also offers improvements in terms of detection accuracy but also performance when compared to the state of the art algorithms.

8 Related Work

8.1 Video Characterization

Video Characterization refers to the process of identifying and categorizing videos according to their technical attributes, content or popularity, for the purpose of conveying information about them. The significance of this procedure, lies in understanding the viewers' preferences in terms of video quality and content, so as to evaluate current and future viewing patterns.

In [74] and [75], traffic from university campuses was captured and processed in order to characterize usage patterns and local and global video popularity respectively. Other researchers have "crawled" YouTube to collect meta information [76] or gather video and social statistics [77]. As a result, they found that video popularity and user preferences have a great impact on local and remote networks. Therefore, different caching polices were proposed to handle the increasing traffic generated by YouTube.

8.2 Video Delivery Infrastructure

The YouTube infrastructure and server selection mechanisms have been put under the microscope by researchers, as well as the physical location of YouTube servers [78], [79]. Additionally, in [80] there is an analytical comparison among YouTube and other video sharing services via crawling the websites and measuring delays. In [40], there is a comparison between PC and mobile users of YouTube and how their behavior can be related to system performance degradation. The conclusions derived in these papers agree on a load balancing mechanism that redirects YouTube users to preferred video servers in order to achieve a more uniform load distribution in the system. Additionally, in cases where load balancing resulted in redirection to non-preferred servers, there were factors such as DNS server variation, lack of video availability in some servers and high server load due to popular video content.

8.3 Video User Experience

With respect to the research concerning the YouTube user experience, Mok et al. [3] approached user QoE through investigating network QoS metrics. The procedure followed here, included a customized Flash video player able to detect buffering events which are in consequence related to user experience. A similar approach was followed in [4], where a custom browser-based plug-in was implemented to provide feedback about the videos' buffering status and predict possible disruptions in the playback due to buffer underflow events. In addition, the same authors [5], enhanced the aforementioned method for Wireless Mesh Network environments, with the addition of an application to perform resource management tasks. In [41] the Mean Opinion Score (MOS) scale was successfully related to the occurrence of increased packet loss that resulted in re-buffering events during video playback. The MOS represents the average of the scores when rating the quality of a service on a scale of 1 to 5, where lower numbers indicate poorer experience. Finally, in the work of Dobrian et al. [6], client-based tools in controlled lab settings where used to extract statistics for short and long Video on Demand (VoD) and for streaming video services. More specifically, user experience for different types of media content was evaluated in terms of quality metrics and content types. The work done in the publications related to User Experience is the most relevant to the one presented in this paper. Their results reveal that network QoS metrics such as Round Trip Time (RTT) and packet loss, may affect the buffering process of a YouTube video and therefore affect the user's experience. More specifically, either by using the re-buffering frequency or the buffering ratio as performance metrics, researchers were able to derive MOS marks and link network QoS to user QoE. Our work is mostly related to the work previously done on YouTube user experience. However, we differ from these papers for the reason that we do not rely on custom-made players or browser plug-ins to make measurements nor do we rely on controlled experiments performed in the lab. In contrast, we were able to extract all the important metrics for our study, under a real-life scenario, only from passively monitoring the related traffic in a network hosting thousands of users per day. Prometheus [20] uses passive measurements on a mobile network to estimate the QoE of two applications, Video on Demand and VoIP. For the video QoE only Buffering Ratio is considered as a QoE indicator, while the system is evaluated only on unencrypted traffic using binary classification to detect buffering issues with 84% accuracy.

Using similar approaches, OneClick [21] and HostView [22] develop predictive models to detect the QoE of multiple applications including video streaming, using network performance metrics. However, both approaches are limited by the requirement of instrumented devices to capture the feedback from the users.

In [12, 81, 82] YouTube buffer outages are detected by comparing the playback times of the video frames and the time stamps of the received packets. These methods rely on passive measurements and DPI to extract QoE information but require the inspection of each packet in order to calculate timing offsets. Therefore, this method can be difficult to scale when dealing with high number of videos and high frame-rates.

Hossfeld et al. [39] study the impact of the amplitude and frequency of representation switches on the user experience. The authors re-encoded a video in multiple qualities and introduced different levels and frequencies of switching and performed crowd-sourced experiments to detect correlations with the received MOS from the users. In this work only a single short video was used, which can be considered a very limited representation of the diverse content found in popular services.

In [38] the authors perform subjective tests in mobile networks to assess the impact that the video quality level and quality switching among other factors has on the users' experience. The experiments were conducted with a very limited sample of very short videos, while only the direction of quality switching, i.e. resolution upscaling or downscaling was taken into consideration but not the effects of the amplitude or the frequency.

Finally, the work of Liu et al. [83] investigates three factors that influence the user perceived quality, initial delay, stalling and quality level variation. The authors conducted experiments in the lab with different network conditions in order to derive functions for calculating each of the three impairment factors. The fact that the tests were performed in the lab however, minimizes the generalization of the results to real network conditions and to real streaming services where CDNs and different quality adaptation logics can create different effects in terms of initial delay and quality switches respectively.

Overall, although significant work has been done previously in detecting and quantifying the factors that affect the quality of video streaming, our work is the first that extensively studies these factors in a large scale network using encrypted traffic.

8.4 Path Diagnosis

The works presented here deal with common issues in wireless, broadband and WANs. This information provided useful insights for the problems that may affect the performance of video streaming services and the users QoE.

In [23] intra- and inter-ISP links were measured to identify issues affecting video streaming QoE. The findings show that most of the issues originate from fluctuations in intradomain links, however there is no clear correlation of these problems with QoE. Finally, [24], showed that voice streaming over backbone links is only affected in rare cases of packet loss.

[85] presents Pythia, a measurement framework for diagnosing performance problems in wide area networks. The proposed system relies on inter-domain monitoring and active measurement probes in order to detect and localize the root-cause of performance issues.

G-RCA [86] is a RCA platform that is capable of identifying the root cause of a performance issues in large IP networks and provides different root cause analysis tools for newly discovered issues.

8.5 Mobile Video Traffic Characterisation

In [87], the authors analyse YouTube traffic from a university campus network to conclude that caching improves the performance and the scalability of the service. Plissonneau et al. [88], study the impact of throughput and delay on YouTube abandonment for DSL users. In [89], distributed active measurements are used to measure YouTube and find the effect of redirections and load balancing on video performance. The authors of [90] propose a YouTube traffic generation model based on traces collected from real use cases.

Plissonneau et al. [91] analysed the performance of video streaming over 2G and 3G, while a more recent work [92] evaluated the impact of YouTube on mobile networks. [93] reported 10% packet loss due to redundant TCP connections when streaming on Android and iOS mobile devices. Hoque et al. [94] studied the energy consumption with five mobile video streaming services. [95] provided a comparative study between Android and iOS video streaming where larger number of duplicate data was found on iOS.

The information in the works mentioned above, allow us to obtain a more concrete understanding of the generated traffic patterns and important parameters that affect the performance of these services.

8.6 Video Streaming QoE and QoS Correlation

Krishnan et al. [25] used quasi-experimental designs correlate the abandonment rate with the startup delay or the total buffering time. In [3] the authors concluded that the main metric affecting the QoE is the rebuffering frequency. Dobrian et al. [6], show that abandonment is affected by the buffering ratio and startup time.

In [9] a predictive model for video QoE is used to improve user engagement by 20%. The same author in [10] employed machine learning to predict user engagement. In [11], user behaviour is correlated with startup delay, redirections and server response time. Schatz et al. [12] used passive network measurements at ISP-based VPs to infer the rebuffering frequency and duration.

In the related work of Casas et al. [96], the network measurement and analysis framework mPlane is used to detect YouTube QoE anomalies from a large-scale ISP network.

The analysis module raises alerts whenever the video download throughput and bit-rate ratio drops below a certain ratio, since this was found in a previous work to be correlated with buffer outages and stalls.

Contrary to these works, our system does not aim at improving user engagement nor at estimating the video QoE from QoS metrics. We focus on identifying video sessions with low QoE scores and accurately detecting the location and the root cause of the problem.

8.7 QoE from Network KPIs

Metaheuristics have been used in the past for network optimization and planning [97]. For instance, simulated annealing and tabu search were used to allocate radio channels or to discover the minimum connected dominating set for wireless networks [98]. Randomized greedy algorithms have been used to find the optimal location of base-stations in order to maximize the traffic covered and minimize installation costs [99]. In [100] cloud services are ranked based on diverse KPIs such as cost, performance, stability, usability, elasticity, etc., using multi-criteria decision-making [101]. With respect to these works, we attempt to build a methodology that is able to estimate individual QoE components using a set of already collected KPIs. To the best of our knowledge, this is the first time that someone addresses this challenging problem.

Over the years there have been many attempts to understand how KPIs can be used to spot cellular performance bottlenecks [53] and network planning [15]. In [26], an iterative process of network deployment and monitoring through KPIs and drive tests was used to identify the optimal network configuration. Nokia engineers demonstrated how controlled experiments such as drive tests and on-site inspections, together with A-B testing, can be used to establish the relation between ground truth and KPIs in order to optimize the network [16]. In [102], the authors show how drive tests can be used to identify the main KPIs that relate to QoS in a tetra network. Finally, field tests are used to build an empirical correlation between KPIs and throughput [103].

In addition to these works, there have been vendor recommendations on how to set performance thresholds and identify the worse performing sectors in the network. For instance, Huawei [52] also describes an iterative A-B process and provide recommendations for default values while Nokia [17] has extensively analyzed the meaning of each KPI. Our work breaks these long-term assumptions that QoE has to be build only with iterative field tests that are costly and inflexible. We propose a data-driven methodology that is able to automatically establish the relation between KPIs and the groundtruth in order to provide insights about underperforming sectors.

8.8 Anomaly Detection

This section covers two categories of related works, i.e. those that although not related to network performance anomalies, they deal with CAD and those that deal with anomaly detection specific to network performance.

To the extend of our knowledge, this is the first work that uses contextual information for network performance anomaly detection. Nevertheless, the concept of context-based detection algorithms is not new and has been presented in a few different fields in the past. The paper from Chen et al. [73], which is the most related to our work, presents a contextual change detection approach that uses the K_{th} distance for context construction and the TAD metric to detect changes. Our approach uses DTW distances and standard deviation for the anomaly detection respectively instead.

CAD has also been applied in big sensor data [104], where point anomalies are identified using a univariate Gaussian predictor, while the k-means-based contextual detection is used as a post-processing step. [105] used a prediction-based CAD for detecting stock market manipulation. Here, instead of using a time series' historical data to predict future values, predictions are made from contextual information. Although there is extensive work previously done to cover network security and network intrusion detection, significantly fewer articles have dealt with network performance anomalies. Here we present a few notable related publications in this field.

In the related literature, a wide array of different methods has been used to detect network anomalies. [19, 106] use PCA on backbone network traffic to capture the variance of anomalous time series. Other statistical methods such as Kalman filters in [107] and wavelets in [108] and [109] were also successfully used to perform anomaly detection. Other examples include, the use of Haar-wavelet analysis [110], a method based on entropy [111], change-point detection using the CUSUM algorithm [112–114], adaptive threshold analysis [115], Holt-Winters seasonal forecasting based methods [116], data reduction techniques with sketches [117, 118] and SNMP MIB Support Vector Machine (SVM) analysis [119]. However, in contrast to our work none of the these methods takes into consideration the contextual information when identifying anomalies.

9 Conclusions

This thesis has addressed some of the most important challenges regarding the detection, troubleshooting and prevention of video QoE issues. More specifically, this work provided the required tools and methods for identifying the nature of quality problems during a video session and the impact they have on the user's experience. The proposed solutions rely on passive measurements in order to minimize the interference with the traffic and the user and can detect problems with high accuracy in large-scale fixed or cellular networks, with clear-text and/or encrypted video streams.

The findings of this work showed that the redirections make the greatest contribution to the initial delays, while the stall duration has the greatest impact on the user's experience. This study was the first to the extend of our knowledge to look into the impact that pre-rolled video advertisements have on the QoE and find that non-skippable video ads lead to higher abandonment rates.

Also we showed that it is possible to achieve accuracies as high as 93% when detecting

QoE issues from encrypted video streams, based on a minimal set of features that are extracted from the transport layer. Among these features, we found that the video segment size and arrival time variations played the most significant part in identifying quality problems.

The thesis has presented a framework that can be used to perform root-cause analysis on video QoE faults. This framework can be deployed on multiple vantage points, e.g. the user's device, the home gateway, the ISP's core network, the video server and so on, in order to provide insights regarding the existence of QoE impairments, which was the offending part of the network path that gave rise to the impairments and what was the root cause behind them.

The framework was evaluated with over 80% when deployed in the wild and we showed that three vantage points are enough to successfully identify, locate and troubleshoot faults. Moreover, we find that each entity that can take advantage of the framework to detect and analyze QoE issues without having to share information with others, however the overall accuracy can be improved when collaboration between entities is possible.

We also proposed a novel data-driven methodology of combining the important KPIs in an ISP's mobile network to capture the under-performing parts of the network, where the delivered QoE of a service does not meet the provider's SLA. This approach, allows operators to pin-point "red" sectors that need to be prioritized for upgrades that will improve the quality received by the connected subscribers. At the same time, sectors that are not "red" but have increasing performance issues can be proactively dealt with in order to prevent the quality degradation of different services in the future.

By applying the proposed data-driven methodology, operators are empowered with a flexible tool that can utilize the already collected KPIs to improve the view on under-performing sectors in terms of the delivered QoE. Moreover, our results indicate that the currently used solution that is based on thresholding is sub-optimal to identify critical sectors. This opens new areas or research for monitoring solutions enriching the quality and accuracy of the network performance indicators collected at the network edge.

Finally, we have presented a novel approach for detecting network performance anomalies using contextual information. We have shown that not only this method can be successfully applied in both synthetic and real network traffic but it also offers improvements in terms of detection accuracy and performance when compared to the state of the art algorithms. This solution can have immediate benefits for network operators, since it allows them to detect network anomalies that affect individual clients as soon as they occur and troubleshoot them before they escalate to cause QoE issues.

Overall, this thesis has advanced the related state-of-the-art by contributing a series of novel methodologies, frameworks and algorithms that successfully tackle some very challenging problems in the related areas of research. These contributions can be adopted by researchers as tools that aid they study of video QoE and network performance anomalies. At the same time, they can be quite beneficial to fixed and mobile network operators who require updated and accurate solutions for monitoring the performance of their networks.

9.1 Future Work

This thesis has successfully addressed all the major challenges that were discussed in Section 1.1. Nevertheless, the always evolving ecosystem of technologies and formats of video streaming services maintains a constant research interest from both the academia and the industry for updating and extending the current state-of-the-art.

Towards this direction, one of the steps forward in this work is to extend the evaluation of the QoE issue detection methodologies to include a wider array of video streaming services. In this way, we can ensure that the proposed methods can be generalized to other video streaming systems, where different delivery mechanisms, traffic patterns and protocols could be encountered.

Moreover, the RCA framework that was described in Chapter 5 can be augmented with a set of active measurements to detect faults that passive measurements cannot capture such as DNS failures, the existence of middleboxes that alter the traffic and/or the video stream, routing and path changes and so on. Another interesting direction towards which that this work can be extended to, is to explore is the detection of multiple co-occurring faults that simultaneously cause the degradation of the video streaming quality. These features will greatly expand the framework's capabilities and allow a more comprehensive analysis of the causes that affect the user's experience.

Next, the framework can be deployed in the wild to capture measurements related to a wide range of faults. The resulting dataset can then be used to evaluate the performance of the Contextual Anomaly Detection algorithm with different types of anomalies. The successful evaluation with numerous network faults will establish the anomaly detection method as a highly generalizable and accurate solution that can be applied in many scenarios.

References

[1] Cisco. Cisco visual networking index: Global mobile data traffic forecast update. *White Paper*, February 2016.

[2] Caroline Gabriel. Managing the new mobile data network. *White Paper, Rethink Technology Research Ltd*, 2015.

[3] Ricky KP Mok, Edmond WW Chan, and Rocky KC Chang. Measuring the quality of experience of http video streaming. In *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*, pages 485–492. IEEE, 2011.

[4] Barbara Staehle, Matthias Hirth, Rastin Pries, Florian Wamser, and Dirk Staehle. Yomo: A youtube application comfort monitoring tool. *New Dimensions in the Assessment and Support of Quality of Experience for Multimedia Applications, Tampere, Finland*, pages 1–3, 2010.

[5] Barbara Staehle, Matthias Hirth, Rastin Pries, Florian Wamser, and Dirk Staehle. Aquarema in action: Improving the youtube qoe in wireless mesh networks. In *Internet Communications (BCFIC Riga), 2011 Baltic Congress on Future*, pages 33–40. IEEE, 2011.

[6] Florin Dobrian, Vyas Sekar, Asad Awan, Ion Stoica, Dilip Joseph, Aditya Ganjam, Jibin Zhan, and Hui Zhang. Understanding the impact of video quality on user engagement. *ACM SIGCOMM Computer Communication Review*, 41(4):362–373, 2011.

[7] Giorgos Dimopoulos, Pere Barlet-Ros, and Josep Sanjuas-Cuxart. Analysis of youtube user experience from passive measurements. In *Network and Service Management (CNSM), 2013 9th International Conference on*, pages 260–267. IEEE, 2013.

[8] Louis Plissonneau, Ernst Biersack, and Parikshit Juluri. Analyzing the impact of youtube delivery policies on user experience. In *Proceedings of the 24th international Teletraffic congress*, page 28. International Teletraffic Congress, 2012.

[9] Athula Balachandran, Vyas Sekar, Aditya Akella, Srinivasan Seshan, Ion Stoica, and Hui Zhang. Developing a predictive model of quality of experience for internet video. In *ACM SIGCOMM Computer Communication Review*, volume 43, pages 339–350. ACM, 2013.

[10] Athula Balachandran, Vyas Sekar, Aditya Akella, Srinivasan Seshan, Ion Stoica, and Hui Zhang. A quest for an internet video quality-of-experience metric. In *Proceedings of the 11th ACM workshop on hot topics in networks*, pages 97–102. ACM, 2012.

[11] Alessandro Finamore, Marco Mellia, Maurizio M Munafò, Ruben Torres, and Sanjay G Rao. Youtube everywhere: Impact of device and infrastructure synergies on user experience. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 345–360. ACM, 2011.

[12] Raimund Schatz, Tobias Hoßfeld, and Pedro Casas. Passive youtube qoe monitoring for isps. In *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on*, pages 358–364. IEEE, 2012.

[13] Jaana Laiho, Achim Wacker, and Tomáš Novosad. *Radio network planning and optimisation for UMTS*. John Wiley & Sons, 2006.

[14] Ajay R Mishra. *Fundamentals of cellular network planning and optimisation: 2G/2.5 G/3G... evolution to 4G*. John Wiley & Sons, 2004.

[15] Ralf Kreher and Karsten Gaenger. *Key Performance Indicators and Measurements for LTE Radio Network Optimization*. John Wiley & Sons, Ltd, 2010.

[16] Harri Holma and Antti Toskala. *WCDMA for UMTS: HSPA Evolution and LTE*. John Wiley & Sons, Inc., New York, NY, USA, 2007.

[17] Nitin Agarwal. Wcdma : Kpi analysis & optimization. *Nokia Technologies Co., Ltd.*, 2008.

[18] Haining Wang, Danlu Zhang, and Kang G Shin. Detecting syn flooding attacks. In *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1530–1539. IEEE, 2002.

[19] Anukool Lakhina, Mark Crovella, and Christophe Diot. Diagnosing network-wide traffic anomalies. In *ACM SIGCOMM Computer Communication Review*, volume 34, pages 219–230. ACM, 2004.

[20] Vaneet Aggarwal, Emir Halepovic, Jeffrey Pang, Shobha Venkataraman, and He Yan. Prometheus: Toward quality-of-experience estimation for mobile apps from passive network measurements. In *Proceedings of the 15th Workshop on Mobile Computing Systems and Applications*, page 18. ACM, 2014.

[21] K-T Chen, C-C Tu, and W-C Xiao. Oneclick: A framework for measuring network quality of experience. In *INFOCOM 2009, IEEE*, pages 702–710. IEEE, 2009.

[22] Diana Joumblatt, Jaideep Chandrashekar, Branislav Kveton, Nina Taft, and Renata Teixeira. Predicting user dissatisfaction with internet application performance at end-hosts. In *INFOCOM, 2013 Proceedings IEEE*, pages 235–239. IEEE, 2013.

[23] Mukundan Venkataraman and Mainak Chatterjee. Quantifying video-qoe degradations of internet links. *IEEE/ACM Transactions on Networking (TON)*, 20(2):396–407, 2012.

[24] Athina P Markopoulou, Fouad A Tobagi, and Mansour J Karam. Assessment of voip quality over internet backbones. In *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 1, pages 150–159. IEEE, 2002.

[25] S Shunmuga Krishnan and Ramesh K Sitaraman. Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs. *IEEE/ACM Transactions on Networking*, 21(6):2001–2014, 2013.

[26] Md. Ariful Alam. Mobile network planning and kpi improvement. In *Thesis, Linnaeus University, Department of Physics and Electrical Engineering, Sweden*, 2013.

[27] Ericsson. Measuring and improving network performance. *White paper*, 2014.

[28] Stephanie Yeo Ken Ting and Tiong Teck Chai. Wcdma network planning and optimisation. In *Telecommunication Technologies 2008 and 2008 2nd Malaysia Conference on Photonics. NCTT-MCP 2008. 6th National Conference on*, pages 317–322. IEEE, 2008.

[29] Ltd. Huawei Technologies Co. HUAWEI RAN KPI for performance management. *Instruction manual*, 2006.

[30] Giorgos Dimopoulos, Ilias Leontiadis, Pere Barlet-Ros, and Konstantina Papagiannaki. Measuring video qoe from encrypted traffic. In *Proceedings of the 2016 ACM on Internet Measurement Conference*, pages 513–526. ACM, 2016.

[31] Giorgos Dimopoulos, Ilias Leontiadis, Pere Barlet-Ros, Konstantina Papagiannaki, and Peter Steenkiste. Identifying the root cause of video streaming issues on mobile devices. In *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*, page 24. ACM, 2015.

[32] Ilias Leontiadis, Joan Serra, Alessandro Finamore, Dimopoulos Giorgos, and Papagiannaki Konstantina. The good, the bad, and the kpis: how to combine performance metrics to better capture underperforming sectors in mobile networks. In *Data Engineering (ICDE), 2017 IEEE 33rd International Conference on*. IEEE, 2017.

[33] Giorgos Dimopoulos, Pere Barlet-Ros, Constantine Dovrolis, and Ilias Leontiadis. Detecting network performance anomalies with contextual anomaly detection. *ACM SIGCOMM Computer Communication Review*, 48(2), 2017.

[34] Ashwin Rao, Arnaud Legout, Yeon-sup Lim, Don Towsley, Chadi Barakat, and Walid Dabbous. Network characteristics of video streaming traffic. *Proceedings of the Seventh COnference on emerging Networking EXperiments and Technologies*, page 25, 2011.

[35] Ricky KP Mok, Edmond WW Chan, Xiapu Luo, and Rocky KC Chang. Inferring the qoe of http video streaming from user-viewing activities. *Proceedings of the first ACM SIGCOMM workshop on Measurements up the stack*, pages 31–36, 2011.

[36] Guangtao Zhai, Jianfei Cai, Weisi Lin, Xiaokang Yang, Wenjun Zhang, and Minoru Etoh. Cross-dimensional perceptual quality assessment for low bit-rate videos. *IEEE Transactions on Multimedia*, 10(7):1316–1324, 2008.

[37] Tobias Hoßfeld, Michael Seufert, Matthias Hirth, Thomas Zinner, Phuoc Tran-Gia, and Raimund Schatz. Quantification of youtube qoe via crowdsourcing. In *Multimedia (ISM), 2011 IEEE International Symposium on*, pages 494–499. IEEE, 2011.

[38] Blazej Lewcio, Benjamin Belmudez, Amir Mehmood, Marcel Wältermann, and Sebastian Möller. Video quality in next generation mobile networks—perception of time-varying transmission. *Communications Quality and Reliability (CQR), 2011 IEEE International Workshop Technical Committee on*, pages 1–6, 2011.

[39] Tobias Hoßfeld, Michael Seufert, Christian Sieber, and Thomas Zinner. Assessing effect sizes of influence factors towards a qoe model for http adaptive streaming. In *Quality of Multimedia Experience (QoMEX), 2014 Sixth International Workshop on*, pages 111–116. IEEE, 2014.

[40] Alessandro Finamore, Marco Mellia, Maurizio M Munafò, Ruben Torres, and Sanjay G Rao. Youtube everywhere: Impact of device and infrastructure synergies on user experience. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 345–360. ACM, 2011.

[41] Jorgen Gustafsson, Gunnar Heikkila, and Martin Pettersson. Measuring multimedia quality in mobile networks with an objective parametric model. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pages 405–408. IEEE, 2008.

[42] Pere Barlet Ros. *Load shedding in network monitoring applications*. Universitat Politècnica de Catalunya, 2008.

[43] Josep Sanjuàs-Cuxart, Pere Barlet-Ros, and Josep Solé-Pareta. Measurement based analysis of one-click file hosting services. *Journal of Network and Systems Management*, 20(2):276–301, 2012.

[44] S Shunmuga Krishnan and Ramesh K Sitaraman. Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs. *IEEE/ACM Transactions on Networking*, 21(6):2001–2014, 2013.

[45] Xiaoqi Yin, Abhishek Jindal, Vyas Sekar, and Bruno Sinopoli. A control-theoretic approach for dynamic adaptive video streaming over http. In *ACM SIGCOMM Computer Communication Review*, volume 45, pages 325–338. ACM, 2015.

[46] Ewan S Page. Continuous inspection schemes. *Biometrika*, 41(1/2):100–115, 1954.

[47] “YouTube: Most Viewed Videos of All Time”. https://www.youtube.com/playlist?list=PLirAqAtl_h2r5g8xGajEwdXd3x1sZh8hC.

[48] Tstat, tcp statistic and analysis tool. http://tstat.polito.it/index.shtml.

[49] Tstat log files documentation. http://tstat.tlc.polito.it/measure.shtml#log_tcp_complete.

[50] “Most Viewed Non-Vevo / Non-Music Videos”. https://goo.gl/MHKpV4.

[51] Alessio Botta, Alberto Dainotti, and Antonio Pescapé. A tool for the generation of realistic network workload for emerging networking scenarios. *Computer Networks*, 56(15):3531–3547, 2012.

[52] Guo Hao X Kaiping. Gsm kpi monitoring and improvement guide. *Huawei Technologies Co., Ltd.*, 2008.

[53] Ana Nika, Asad Ismail, Ben Y Zhao, Sabrina Gaito, Gian Paolo Rossi, and Haitao Zheng. Understanding data hotspots in cellular networks. In *Heterogeneous Networking for Quality, Reliability, Security and Robustness (QShine), 2014 10th International Conference on*, pages 70–76. IEEE, 2014.

[54] C. E. Spearman. The proof and measurement of association between two things. *American Journal of Psychology*, 3-4:441–471, 1904.

[55] C. Blum and D. Merkle. *Swarm intelligence*. Springer, Berlin, Germany, 2008.

[56] Sean Luke. Essentials of metaheuristics. 2009, 2010.

[57] L. Bianchi, M. Dorigo, L. M. Gambardella, and W. J. Gutjahr. A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing*, 8(2):239–287, 2009.

[58] R. Poli, J. Kennedy, and T. M. Blackwell. Particle swarm optimization. *Swarm Intelligence*, 1(1):33–57, 2007.

[59] M. Clerc. *Particle swarm optimization*. ISTE, London, UK, 2006.

[60] P. C. Fourie and A. A. Groenwold. Particle swarms in size and shape optimization. In *Proc. of the Int. Workshop on Multidisciplinary Design Optimization*, pages 97–106, 2000.

[61] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. Springer, Berlin, Germany, 2nd edition, 2009.

[62] L. Breiman and J. Friedman. *Classification and regression trees*. Chapman and Hall/CRC, Monterey, USA, 1984.

[63] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[64] T. K. Ho. Random decision forests. In *Proc. of the Int. Conf. on Document Analysis and Recognition (ICDAR)*, pages 278–282, 1995.

[65] Alessandro Finamore, Marco Mellia, Zafar Gilani, Konstantina Papagiannaki, Vijay Erramilli, and Yan Grunenberger. Is there a case for mobile phone content pre-staging? In *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*, pages 321–326. ACM, 2013.

[66] Speedtest.net by ookla - the global broadband speed test. http://www.speedtest.net/.

[67] Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, 1994.

[68] Stan Salvador and Philip Chan. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, 11(5):561–580, 2007.

[69] Fcc measuring broadband america 2015. https://goo.gl/qWTfDD.

[70] Samknows: The global platform for internet measurement. https://www.samknows.com/.

[71] Fcc measuring broadband america 2015 technical appendix. https://goo.gl/Hch7jY.

[72] Norman L Johnson. Systems of frequency curves generated by methods of translation. *Biometrika*, 36(1/2):149–176, 1949.

[73] Xi C Chen, Karsten Steinhaeuser, Shyam Boriah, Snigdhansu Chatterjee, and Vipin Kumar. Contextual time series change detection. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, pages 503–511. SIAM, 2013.

[74] Phillipa Gill, Martin Arlitt, Zongpeng Li, and Anirban Mahanti. Youtube traffic characterization: a view from the edge. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 15–28. ACM, 2007.

[75] Michael Zink, Kyoungwon Suh, Yu Gu, and Jim Kurose. Characteristics of youtube network traffic at a campus network–measurements, models, and implications. *Computer networks*, 53(4):501–514, 2009.

[76] Meeyoung Cha, Haewoon Kwak, Pablo Rodriguez, Yong-Yeol Ahn, and Sue Moon. I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 1–14. ACM, 2007.

[77] Xu Cheng, Cameron Dale, and Jiangchuan Liu. Statistics and social network of youtube videos. In *Quality of Service, 2008. IWQoS 2008. 16th International Workshop on*, pages 229–238. IEEE, 2008.

[78] Vijay Kumar Adhikari, Sourabh Jain, and Zhi-Li Zhang. Youtube traffic dynamics and its interplay with a tier-1 isp: an isp perspective. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 431–443. ACM, 2010.

[79] Ruben Torres, Alessandro Finamore, Jin Ryong Kim, Marco Mellia, Maurizio M Munafo, and Sanjay Rao. Dissecting video server selection strategies in the youtube cdn. In *Distributed Computing Systems (ICDCS), 2011 31st International Conference on*, pages 248–257. IEEE, 2011.

[80] Mohit Saxena, Umang Sharan, and Sonia Fahmy. Analyzing video services in web 2.0: a global perspective. In *Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, pages 39–44. ACM, 2008.

[81] Pedro Casas, Michael Seufert, and Raimund Schatz. Youqmon: A system for online monitoring of youtube qoe in operational 3g networks. *ACM SIGMETRICS Performance Evaluation Review*, 41(2):44–46, 2013.

[82] Pedro Casas, Raimund Schatz, and Tobias Hoßfeld. Monitoring youtube qoe: Is your mobile network delivering the right experience to your customers? In *Wireless Communications and Networking Conference (WCNC), 2013 IEEE*, pages 1609–1614. IEEE, 2013.

[83] Yao Liu, Sujit Dey, Don Gillies, Faith Ulupinar, and Michael Luby. User experience modeling for dash video. In *Packet Video Workshop (PV), 2013 20th International*, pages 1–8. IEEE, 2013.

[84] Muhammad Zubair Shafiq, Jeffrey Erman, Lusheng Ji, Alex X Liu, Jeffrey Pang, and Jia Wang. Understanding the impact of network dynamics on mobile video user engagement. In *ACM SIGMETRICS Performance Evaluation Review*, volume 42, pages 367–379. ACM, 2014.

[85] Partha Kanuparthy and Constantine Dovrolis. Pythia: Diagnosing performance problems in wide area providers. In *USENIX Annual Technical Conference*, pages 371–382, 2014.

[86] He Yan, Lee Breslau, Zihui Ge, Dan Massey, Dan Pei, and Jennifer Yates. G-rca: a generic root cause analysis platform for service quality management in large ip networks. *IEEE/ACM Transactions on Networking*, 20(6):1734–1747, 2012.

[87] Phillipa Gill, Martin Arlitt, Zongpeng Li, and Anirban Mahanti. Youtube traffic characterization: a view from the edge. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 15–28. ACM, 2007.

[88] Louis Plissonneau, Taoufik En-Najjary, and Guillaume Urvoy-Keller. Revisiting web traffic from a dsl provider perspective: the case of youtube. In *Proc. of the 19th ITC specialist seminar*, 2008.

[89] Vijay Kumar Adhikari, Sourabh Jain, Yingying Chen, and Zhi-Li Zhang. Vivisecting youtube: An active measurement study. In *INFOCOM, 2012 Proceedings IEEE*, pages 2521–2525. IEEE, 2012.

[90] Pablo Ameigeiras, Juan J Ramos-Munoz, Jorge Navarro-Ortiz, and Juan M Lopez-Soler. Analysis and modelling of youtube traffic. *Transactions on Emerging Telecommunications Technologies*, 23(4):360–377, 2012.

[91] Louis Plissonneau and Guillaume Vu-Brugier. Mobile data traffic analysis: How do you prefer watching videos? In *Teletraffic Congress (ITC), 2010 22nd International*, pages 1–8. IEEE, 2010.

[92] Juan J Ramos-Muñoz, Jonathan Prados-Garzon, Pablo Ameigeiras, Jorge Navarro-Ortiz, and Juan M López-Soler. Characteristics of mobile youtube traffic. *IEEE Wireless Communications*, 21(1):18–25, 2014.

[93] Hyunwoo Nam, Bong Ho Kim, Doru Calin, and Henning Schulzrinne. A mobile video traffic analysis: Badly designed video clients can waste network bandwidth. In *Globecom Workshops (GC Wkshps), 2013 IEEE*, pages 506–511. IEEE, 2013.

[94] Mohammad Ashraful Hoque, Matti Siekkinen, Jukka K Nurminen, and Mika Aalto. Investigating streaming techniques and energy efficiency of mobile video services. *arXiv preprint arXiv:1209.2855*, 2012.

[95] Yao Liu, Fei Li, Lei Guo, Bo Shen, and Songqing Chen. A comparative study of android and ios for accessing internet streaming services. In *International Conference on Passive and Active Network Measurement*, pages 104–114. Springer, 2013.

[96] Pedro Casas, Pierdomenico Fiadino, Sarah Wassermann, Stefano Traverso, Alessandro D'Alconzo, Edion Tego, Francesco Matera, and Marco Mellia. Unveiling network and service performance degradation in the wild with mplane. *IEEE Communications Magazine*, 54(3):71–79, 2016.

[97] Simone L Martins and Celso C Ribeiro. Metaheuristics and applications to optimization problems in telecommunications. In *Handbook of optimization in telecommunications*, pages 103–128. Springer, 2006.

[98] M. Morgan and V. Grout. Metaheuristics for wireless network optimisation. In *Telecommunications, 2007. AICT 2007. The Third Advanced International Conference on*, pages 15–15, May 2007.

[99] Edoardo Amaldi, Antonio Capone, and Federico Malucelli. Planning umts base station location: optimization models with power control and algorithms. *IEEE Trans. Wireless Communications*, 2(5):939–952, 2003.

[100] Saurabh Kumar Garg, Steve Versteeg, and Rajkumar Buyya. A framework for ranking of cloud computing services. *Future Generation Computer Systems*, 29(4):1012 – 1023, 2013. Special Section: Utility and Cloud Computing.

[101] JosÉ. Figueira, Salvatore. Greco, and SpringerLink (Online service). *Multiple Criteria Decision Analysis: State of the Art Surveys*. International Series in Operations Research & Management Science,. Springer New York,, New York, NY :, 2005.

[102] Jose Dario Luis Delgado and Jesus Maximo Ramirez Santiago. Key performance indicators for QOS assessment in TETRA networks. *CoRR*, abs/1401.1918, 2014.

[103] Simo-Ville Hönö. Case studies of network planning for wireless broadband services – hsdpa and wimax. *Master Thesis, OmniTele*, 2007.

[104] Michael A Hayes and Miriam AM Capretz. Contextual anomaly detection in big sensor data. In *Big Data (BigData Congress), 2014 IEEE International Congress on*, pages 64–71. IEEE, 2014.

[105] Koosha Golmohammadi and Osmar R Zaiane. Time series contextual anomaly detection for detecting market manipulation in stock market. In *Data Science and Advanced Analytics (DSAA), 2015. 36678 2015. IEEE International Conference on*, pages 1–10. IEEE, 2015.

[106] Ling Huang, XuanLong Nguyen, Minos Garofalakis, Joseph M Hellerstein, Michael I Jordan, Anthony D Joseph, and Nina Taft. Communication-efficient online detection of network-wide anomalies. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 134–142. IEEE, 2007.

[107] Augustin Soule, Kavé Salamatian, and Nina Taft. Combining filtering and statistical methods for anomaly detection. In *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*, pages 31–31. USENIX Association, 2005.

[108] Paul Barford, Jeffery Kline, David Plonka, and Amos Ron. A signal analysis of network traffic anomalies. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pages 71–82. ACM, 2002.

[109] Polly Huang, Anja Feldmann, and Walter Willinger. A non-instrusive, wavelet-based approach to detecting network performance problems. In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pages 213–227. ACM, 2001.

[110] Wei Lu and Ali A Ghorbani. Network anomaly detection based on wavelet analysis. *EURASIP Journal on Advances in Signal Processing*, 2009:4, 2009.

[111] George Nychis, Vyas Sekar, David G Andersen, Hyong Kim, and Hui Zhang. An empirical evaluation of entropy-based traffic anomaly detection. In *Proceedings of the 8th ACM SIGCOMM conference on Internet measurement*, pages 151–156. ACM, 2008.

[112] Vasilios A Siris and Fotini Papagalou. Application of anomaly detection algorithms for detecting syn flooding attacks. In *Global Telecommunications Conference, 2004. GLOBECOM'04. IEEE*, volume 4, pages 2050–2054. IEEE, 2004.

[113] Haining Wang, Danlu Zhang, and Kang G Shin. Syn-dog: Sniffing syn flooding sources. In *Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*, pages 421–428. IEEE, 2002.

[114] Alexander G Tartakovsky, Boris L Rozovskii, Rudolf B Blažek, and Hongjoong Kim. Detection of intrusions in information systems by sequential change-point methods. *Statistical methodology*, 3(3):252–293, 2006.

[115] Shanyue Bu, Ruchuan Wang, and Hong Zhou. Anomaly network traffic detection based on auto-adapted parameters method. In *Wireless Communications, Networking and Mobile Computing, 2008. WiCOM'08. 4th International Conference on*, pages 1–4. IEEE, 2008.

[116] Jake D Brutlag. Aberrant behavior detection in time series for network monitoring. In *LISA*, volume 14, pages 139–146, 2000.

[117] Balachander Krishnamurthy, Subhabrata Sen, Yin Zhang, and Yan Chen. Sketch-based change detection: methods, evaluation, and applications. In *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, pages 234–247. ACM, 2003.

[118] Xin Li, Fang Bian, Mark Crovella, Christophe Diot, Ramesh Govindan, Gianluca Iannaccone, and Anukool Lakhina. Detection and identification of network anomalies using sketch subspaces. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 147–152. ACM, 2006.

[119] Jaehak Yu, Hansung Lee, Myung-Sup Kim, and Daihee Park. Traffic flooding attack detection with snmp mib using svm. *Computer Communications*, 31(17):4212–4219, 2008.

A
Publications

A.1 Journals

- (Under review) **G. Dimopoulos**, C. Dovrolis, P. Barlet-Ros, I. Leontiadis."Detecting Network Performance Anomalies With Contextual Change Detection." *ACM SIGCOMM Computer Communication Review (CCR)*, 47(2), 2017

A.2 Conferences

- **G. Dimopoulos**, P. Barlet-Ros, J. Sanjuas-Cuxart. "Analysis of YouTube User Experience from Passive Measurements". *9th International Conference on Network and Service Management (CNSM)*, pp. 260-267. 2013.

- **G. Dimopoulos**, I. Leontiadis, P. Barlet-Ros, K. Papagiannaki, P. Steenkiste. "Identifying the Root Cause of Video Streaming Issues on Mobile Devices. *11th ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT)*, p. 24. 2015. v

- **G. Dimopoulos**, I. Leontiadis, P. Barlet-Ros, K. Papagiannaki. "Measuring Video QoE from Encrypted Traffic". *2016 ACM on Internet Measurement Conference (IMC)*, pp. 513-526. 2016.

- I. Leontiadis, J. Serra, A. Finamore, **G. Dimopoulos**, K. Papagiannaki. "The Good, the Bad, and the KPIs: How to Combine Performance Metrics to Better Capture Underperforming Sectors in Mobile Networks." *IEEE International Conference in Data Engineering (ICDE)*. 2017.

A.3 Patents

- **G. Dimopoulos**, I. Leontiadis. "A Method and Computer Programs for Identifying Video Streaming QoE from Encrypted Traffic". E.U. Patent Application P-1600032 (Patent Pending).