

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE MATEMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

DANIEL A. OLIVEIRA

ENTENDENDO O EFEITO DAS CONDIÇÕES DA REDE NA QUALIDADE DE
EXPERIÊNCIA DO USUÁRIO

RIO DE JANEIRO
2021

DANIEL A. OLIVEIRA

ENTENDENDO O EFEITO DAS CONDIÇÕES DA REDE NA QUALIDADE DE
EXPERIÊNCIA DO USUÁRIO

Trabalho de conclusão de curso de graduação
apresentado ao Departamento de Ciência da
Computação da Universidade Federal do Rio
de Janeiro como parte dos requisitos para ob-
tenção do grau de Bacharel em Ciência da
Computação.

Orientador: Prof. Daniel Sadoc Menasche
Co-orientador: Prof. Edmundo Albuquerque de Souza Silva

RIO DE JANEIRO

2021

CIP - Catalogação na Publicação

048e Oliveira, Daniel Atkinson
 Entendendo o efeito das condições da rede na
 qualidade de experiência do usuário / Daniel
 Atkinson Oliveira. -- Rio de Janeiro, 2021.
 94 f.

 Orientador: Daniel Sadoc Menasche.
 Coorientador: Edmundo Albuquerque de Souza Silva.
 Trabalho de conclusão de curso (graduação) -
 Universidade Federal do Rio de Janeiro, Instituto
 de Matemática, Bacharel em Ciência da Computação,
 2021.

 1. QoS. 2. QoE. 3. Qualidade de vídeo. 4.
 Medidas. I. Menasche, Daniel Sadoc , orient. II.
 Silva, Edmundo Albuquerque de Souza, coorient. III.
 Título.

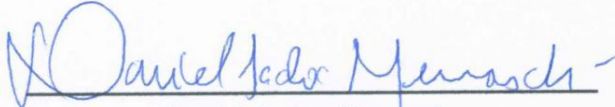
DANIEL A. OLIVEIRA

ENTENDENDO O EFEITO DAS CONDIÇÕES DA REDE NA QUALIDADE DE
EXPERIÊNCIA DO USUÁRIO

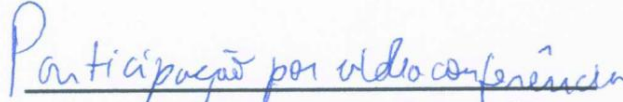
Trabalho de conclusão de curso de graduação
apresentado ao Departamento de Ciência da
Computação da Universidade Federal do Rio
de Janeiro como parte dos requisitos para ob-
tenção do grau de Bacharel em Ciência da
Computação.

Aprovado em 4 de março de 2021

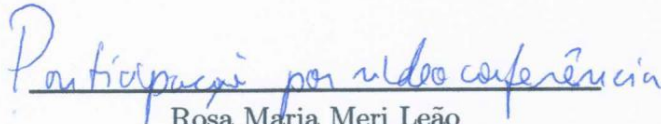
BANCA EXAMINADORA:



Daniel Sadoc Menasche
Ph.D. (UMass Amherst)



Edmundo Albuquerque de Souza e Silva
Ph.D. (UCLA)



Rosa Maria Meri Leão
Dr. (Paul Sabatier)

RESUMO

Foi previsto que, até 2022, aproximadamente 82% do tráfego na Internet será tráfego de vídeo (CISCO, 2019). A expectativa é de que as pessoas assistam os vídeos em diferentes equipamentos, como celulares, smart TVs, computadores e tablets. Ao mesmo tempo, os usuários têm se tornado cada vez mais exigentes quanto à qualidade dos vídeos. Nesse contexto, torna-se crucial que provedores de internet entendam como condições de rede afetam a qualidade dos vídeos, visto que isso impacta diretamente na qualidade de experiência (QoE) do usuário.

O objetivo principal deste trabalho é estudar a relação entre o tamanho do buffer do driver WiFi e a QoE percebida, fazendo uso de métodos interpretativos. A análise é baseada em experimentos que consistem na coleta de dados de uma aplicação de vídeo que é transmitida em uma rede monitorada. Coletamos métricas de vídeos do YouTube usando uma extensão do Google Chrome, implementada em javascript. Mais especificamente, foram coletados dados que permitem a obtenção de: latência inicial, taxa do vídeo, mudanças na taxa do vídeo e ocorrência e duração de rebufferizações. Essas métricas servem como *proxies* para a QoE percebida pelo usuário. Para entender como as métricas de QoE se comportam com mudanças no desempenho da rede, variamos as condições de rede, como, por exemplo, a taxa de perda de pacotes e, crucialmente, o tamanho do buffer do driver de WiFi do roteador de modo a analisar como as métricas de QoE se comportam sujeitas a essas variações. No futuro experimentos serão realizados com clientes voluntários de um provedor de internet para a criação de um modelo de inferência de métricas de QoE a partir de métricas de rede e o tamanho do buffer do driver WiFi.

Palavras-chave: QoS. QoE. qualidade de vídeo. medidas.

ABSTRACT

It has been predicted that, by 2022, approximately 82% of Internet traffic will be video traffic (CISCO, 2019). It is expected that people will watch the videos on many different equipments, like mobile phones, smart TVs, computers and tablets. Users are also becoming more demanding regarding the quality of videos. In this context it becomes crucial that Internet providers understand how network metrics affect the quality of videos, seeing as this impacts directly on the quality of experience (QoE) of the user.

The main objective of this work is to study the relation between the size of the WiFi driver buffer and the user's perceived QoE, using interpretative methods. The analysis is based on experiments comprised by the gathering of data from a video application which is transmitted in a monitored network. YouTube video metrics are collected using a Google Chrome extension, implemented in Javascript. More specifically, the data collected allows the calculation of startup delay, video bitrate, bitrate changes and occurrence and duration of rebuffering events. These metrics serve as a proxy to the user's perceived QoE. To understand how the QoE metrics behave according to different network levels of performance, network conditions are varied, such as packet loss rate and, crucially, the size of the router's WiFi buffer and QoE metrics are then collected, in order to understand how they change according to the variation of network performance. In the future, experiments will be conducted with volunteer clients of an internet service provider to create a model that infers QoE metrics from network metrics and the WiFi driver buffer size.

Keywords: QoE. QoS. video quality. measurements.

LISTA DE ILUSTRAÇÕES

Figura 1 – Funcionamento do DASH. Fonte: (VU; MASHAL; CHUNG, 2017).	17
Figura 2 – Buffers em Série, Cenário 1	19
Figura 3 – Buffers em Série, Cenário 2	19
Figura 4 – Buffers de transmissão de roteador com OpenWrt. Fonte: (LINUX, 2013).	19
Figura 5 – Problema do Nó Escondido. Fonte: (WEYULU; HANADA; KIM, 2017).	20
Figura 6 – Infraestrutura de Medição. Fonte: (ATKINSON et al., 2019)	23
Figura 7 – Métricas de QoE por Perda para o Trailer do Carro	34
Figura 8 – Métricas de QoE por Perda para o Trailer do Rei Leão	35
Figura 9 – Métricas de QoE por Perda para o Trailer Longo do Aquaman	36
Figura 10 – Métricas de QoE por Qualidade para o Trailer Longo do Aquaman	37
Figura 11 – Métricas de QoE por Tamanho do Buffer para o Trailer Curto do Aquaman (Experimento A)	39
Figura 12 – Métricas de QoE por Tamanho do Buffer para o Trailer Curto do Aquaman (Experimento B)	41
Figura 13 – Relação entre Métricas de QoE para Tamanho de Buffer 256 (Experimento A)	42
Figura 14 – Scree plot para Tamanho de Buffer 256 (Experimento A)	46
Figura 15 – Scree plot para Diferentes Tamanhos de Buffer	47
Figura 16 – Scores para Tamanho de Buffer 256 (Experimento A)	49
Figura 17 – Scores para Diferentes Tamanhos de Buffer	51
Figura 18 – Loadings para Tamanho de Buffer 256 (Experimento A)	52
Figura 19 – Loadings para Diferentes Tamanhos de Buffer	53
Figura 20 – Gráficos dos Dados Agregados	56
Figura 21 – SPE Com Diferentes Quantidades de PCs	61
Figura 22 – Scores com Destaque nos Outliers Detectados com SPE	62
Figura 23 – Gráficos Referentes à 1a Reprodução do Experimento B	63
Figura 24 – Gráficos Referentes à 10a Reprodução do Experimento B	63
Figura 25 – Gráficos Referentes à 37a Reprodução do Experimento A	64
Figura 26 – Scores e Loadings Referentes ao PC2 e PC3 do Experimento A	64
Figura 27 – Desvio Padrão dos Scores por Componente	65
Figura 28 – Loadings do Componente Principal 5	66
Figura 29 – SPE Médio por Tamanho de Buffer com Diferentes Quantidades de Componentes Principais	68
Figura 30 – Experimento B Após Filtragem de Outliers	69
Figura 31 – Experimento B Após Filtragem Adicional de Outliers	70

Figura 32 – R^2 por Quantidade de Componentes	72
Figura 33 – Análise dos Dados do Experimento B usando o Modelo A	74
Figura 34 – Análise da Capacidade do Modelo A Explicar os Dados de B	75
Figura 35 – Scores do Experimento B Usando Diferentes Componentes Principais do Modelo A	75
Figura 36 – Análise dos Dados do Experimento A usando o Modelo B	76
Figura 37 – Análise da Capacidade do Modelo B Explicar os Dados de A	77
Figura 38 – Dados dos Experimentos em Formato de Tensor	78
Figura 39 – Interação entre Métricas de Rede, Métricas de Aplicação e Métricas de QoE	79

LISTA DE TABELAS

Tabela 1 – Métricas de Rede Coletadas	24
Tabela 2 – Métricas de WiFi Coletadas	24
Tabela 3 – Métricas de QoE Coletadas	25
Tabela 4 – Propriedades de Objetos Vídeo Javascript	27
Tabela 5 – Eventos de Objetos Vídeo Javascript	28
Tabela 6 – Parâmetros HTTP do Youtube	29
Tabela 7 – Propriedades de Estatísticas para Nerds	31
Tabela 8 – Experimentos em Ordem Cronológica	32
Tabela 9 – Valor das Métricas de QoE com Buffer de Tamanho 256 (Experimento A)	43
Tabela 10 – Valor das Métricas de QoE com Buffer de Tamanho 256 após Centralização no Zero e Normalização (Experimento A)	44
Tabela 11 – Valor das Métricas de QoE Agregadas	55

LISTA DE ABREVIATURAS E SIGLAS

LI	Latência Inicial
BM	Bitrate Médio
RM	Resolução Média
DR	Duração de Rebufferizações
QMR	Quantidade de Mudanças de Resolução
RR	Razão de Rebufferizações
QR	Quantidade de Rebufferizações
QBU	Quantidade de Bytes de Upload
QBD	Quantidade de Bytes de Download
QPU	Quantidade de Pacotes de Upload
QPD	Quantidade de Pacotes de Download
QPP	Quantidade de Pacotes Perdidos
LM	Latência Média
RSSI	Received Signal Strength Indication
SNR	Signal to Noise Ratio
QBUW	Quantidade de Bytes de Upload WiFi
QBDW	Quantidade de Bytes de Download WiFi
QPUW	Quantidade de Pacotes de Upload WiFi
QPDW	Quantidade de Pacotes de Download WiFi
BMUW	Bitrate Médio de Upload WiFi
BMDW	Bitrate Médio de Download WiFi
QoE	Quality of Experience (Qualidade de Experiência)
QoS	Quality of Service (Qualidade de Serviço)
ISP	Internet Service Provider (Provedor de Internet)

RD	Roteador Doméstico
SO	Sistema Operacional
HD	High Definition (Alta Definição)
FPS	Frames per Second (Frames por Segundo)
UFRJ	Universidade Federal do Rio de Janeiro
INRIA	Institut National de Recherche en Informatique et en Automatique
HTTP	Hypertext Transfer Protocol
HAR	HTTP Archive
HAS	HTTP Adaptive Streaming
DASH	Dynamic Adaptive Streaming over HTTP
MPEG	Moving Pictures Expert Group (Grupo de Especialistas de Imagens em Movimento)
WLAN	Wireless Local Area Network
CSMA/CA	Carrier-Sense Multiple Access with Collision Avoidance
ACK	Acknowledgement
RTS	Request to Send
CTS	Clear to Send
AP	Access Point
MIMO	Multiple Input, Multiple Output
MPDU	Mac Protocol Data Unit
A-MPDU	Aggregate Mac Protocol Data Unit
SS	Single Stream
DS	Double Stream
MCS	Modulation Code Scheme
JSON	JavaScript Object Notation
MAC	Media Access Control

IP	Internet Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
CPU	Central Process Unit
RAM	Random Access Memory
FIFO	First In, First Out
SKB	Socket Kernel Buffer
BQL	Byte Queue Limit
CAKE	Common Applications Kept Enhanced
FQ-Codel	Flow Queue Codel
ISO	International Organization for Standardization (Organização Internacional para Padronização)
BUF	Propriedade "buffered" de objeto vídeo JavaScript
CUT	Propriedade "currentTime" de objeto vídeo JavaScript
DUR	Propriedade "duration" de objeto vídeo JavaScript
WFC	Propriedade "webkitDecodedFrameCount" de objeto vídeo JavaScript
WDR	Propriedade "webkitDroppedFrameCount" de objeto vídeo JavaScript
WVD	Propriedade "webkitDecodedByteCount" de objeto vídeo JavaScript
EVE	Instância de evento no log de JavaScript
PCA	Principal Component Analysis
MAD	Median Absolute Deviation
SVD	Singular Value Decomposition
SPE	Squared Prediction Error
PC	Principal Component (Componente Principal)
MOS	Mean Opinion Score

SUMÁRIO

1	INTRODUÇÃO	13
2	TECNOLOGIAS EM FOCO	16
2.1	COMO O YOUTUBE FUNCIONA	16
2.2	O BUFFER	17
2.3	COMO FUNCIONA O 802.11N	19
3	INFRAESTRUTURA E METODOLOGIA	23
4	OBTENÇÃO DAS MÉTRICAS DE QOE	27
5	EXPERIMENTOS E RESULTADOS	32
5.1	EXPERIMENTOS	32
5.2	ENTENDENDO E APLICANDO O PCA	42
6	TRABALHOS FUTUROS	79
7	CONCLUSÃO	82
	REFERÊNCIAS	83
	APÊNDICE A – CONFIGURAÇÃO DO ROTEADOR	88
A.1	CONFIGURANDO O OPENWRT E WIFI	88
A.2	MODIFICANDO O TAMANHO DO BUFFER	89
	APÊNDICE B – PROVA DE QUE OS AUTOVETORES DA MATRIZ DE COVARIÂNCIA EQUIVA- LEM À DIREÇÃO DE MAIOR VARIÂN- CIA	91
	APÊNDICE C – PROVA DE QUE OS AUTOVALORES DA MATRIZ DE COVARIÂNCIA CORRES- PONDEM À VARIÂNCIA NO EIXO DOS AUTOVETORES CORRESPONDENTES	93

APÊNDICE D – PROVA DE QUE MAXIMIZAR A VARI- ÂNCIA NO ESPAÇO DE PROJEÇÃO EQUI- VALE A MINIMIZAR O ERRO DE PRO- JEÇÃO	94
--	----

1 INTRODUÇÃO

Aplicações de vídeo têm se tornado cada vez mais comuns. Foi previsto que, até 2022, aproximadamente 82% do tráfego da internet virá de aplicações de vídeo (CISCO, 2019). Ao mesmo tempo, expectativas dos usuários quanto à qualidade do serviço que recebem também têm aumentado. Questões relacionadas à interação entre a rede residencial, a rede do ISP (Internet Service Provider ou Provedor de Internet) e os provedores de serviço na QoE (Quality of Experience ou Qualidade de Experiência) de aplicações de vídeo são de suma importância.

De acordo com FG IPTV da União Internacional de Telecomunicações (ITU, 2007), “Qualidade de Experiência (QoE) se refere à aceitabilidade geral de uma aplicação ou serviço, conforme as percepções subjetivas do usuário final.” Essa definição parece sugerir que, para determinar a QoE de uma aplicação ou serviço, é necessário obter feedback dos usuários, porém, esse processo pode ser muito complexo além de apresentar outras complicações e sutilezas relacionadas às condições dos experimentos (FIEDLER; MOLLER; REICHL, 2012). Existem, no entanto, outras formas de estimar a QoE de um determinado serviço. Por exemplo, para jogos online, sabe-se que alta latência, também conhecida na comunidade de jogos como “lag”, é prejudicial à QoE, não sendo necessário obter feedback de jogadores para determinar isso. Similarmente, não é necessário perguntar para uma pessoa se ela prefere uma resolução mais alta ao assistir um vídeo online, já que isso quase sempre é verdade. Visto isso, é possível escolher métricas relacionadas a aplicações de vídeo as quais podemos determinar se são positivamente ou negativamente correlacionadas à QoE. Neste trabalho, são coletadas métricas amplamente aceitas na comunidade científica como “Métricas de Qualidade” ou “Métricas de QoE”, ou seja, métricas que estão altamente correlacionadas com QoE, provenientes da camada da aplicação.

A maioria dos trabalhos que abordam qualidade de experiência envolvem coletas feitas a partir de roteadores que agregam um grande número de fluxos de rede de milhares de casas. Recentemente, coletas em roteadores têm sido feitas principalmente para entender a qualidade do serviço que o ISP provê (SUNDARESAN et al., 2012)(FCC, 2018), porém, poucos trabalhos abordam o ambiente residencial (HORA et al., 2018)(SANCHEZ et al., 2015). Uma grande vantagem de coletar dados no ambiente residencial, ou seja, na borda da rede, é que as respostas podem ser muito mais rápidas. Além disso, em geral, não é necessário nenhum hardware caro adicional, pois não são feitas análises em fluxos agregados. Menos trabalhos ainda abordam os problemas relacionados à interação entre o ambiente residencial e a rede do ISP na QoE. Mais especificamente, entender o impacto dos tamanhos dos buffers de WiFi dos roteadores domésticos (RD), na QoE de aplicações de vídeo, é muito relevante na área.

Os roteadores de gateway conectam a rede doméstica à internet, logo, a maioria das

análises interessantes, quanto às experiências dos usuários numa rede doméstica, podem ser conduzidas ao olharmos e modificarmos configurações nesses roteadores e observarmos os resultados. WiFi tem se tornado, cada vez mais, o método mais usado para acesso à internet em redes domésticas, mais especificamente, o protocolo 802.11n. De acordo com (CISCO, 2020), terão quase 628 milhões de hotspots públicos de WiFi em 2023, quatro vezes a quantidade que existia em 2018. O protocolo 802.11n deixa diversos aspectos cruciais dependentes de implementação. Um desses é o tamanho do buffer downstream, com cada driver de WiFi possuindo sua própria implementação dessa funcionalidade.

Neste trabalho descrevo um esforço em andamento em parceria com um grupo de pesquisa da UFRJ, um ISP de médio porte (Gigalink) e uma startup incubada na UFRJ (Anlix). O esforço é recente e consiste em entender o impacto que o tamanho do buffer downstream do driver WiFi de roteadores residenciais causam na QoE percebida do usuário em aplicações de vídeo. Mais especificamente, o modo WiFi testado é o do protocolo 802.11n e a plataforma de vídeo onde as reproduções foram feitas é o YouTube, o serviço de streaming de vídeo mais usado atualmente. De acordo com (INSIDER, 2018), em 2018, cinco bilhões de vídeo eram assistidos por dia na plataforma. Em experimentos futuros outras plataformas de vídeo serão usadas, como Netflix e Twitch, de forma a obter resultados mais generalizáveis.

Um dos objetivos deste trabalho é mostrar o potencial de conduzir estudos relacionados aos efeitos do tamanho do buffer WiFi em QoE de aplicações de vídeo, motivando trabalhos futuros nessa área. Um outro objetivo, é mostrar a relevância atual de modelos interpretativos e a necessidade de conhecimento de domínio para se fazer uma boa análise dos dados. Pretendo passar essa mensagem a partir de observações importantes sobre a relação entre as diferentes métricas de QoE e entre essas mesmas métricas e o tamanho do buffer do driver WiFi em aplicações de vídeo, levando em conta o contexto da rede, o que aumenta a interpretabilidade de qualquer modelo futuro que use essas métricas. Redes neurais, em sua maioria, são modelos cujos parâmetros são difíceis de interpretar, e nos tempos atuais, onde qualquer pessoa pode rapidamente treinar e fazer uso de uma rede neural é importante enfatizar o valor de conhecimento do domínio sob estudo e de modelos interpretativos. Juntos, esses fatores resultam num entendimento mais profundo do problema e dos resultados obtidos.

Além dos objetivos específicos para esse trabalho, também deve-se levar em conta que ele está inserido num contexto mais geral, sendo uma das etapas iniciais, que estão ocorrendo paralelamente, de um projeto muito maior. O objetivo final deste projeto é o desenvolvimento de uma ferramenta para ISPs, para monitoramento e inferência de regiões da rede onde a QoE dos usuários está baixa. Isto pode ser útil para planejamento de alocação recursos e no auxílio para tomadas de decisões, de forma preemptiva, o que pode trazer uma grande economia de custos para o ISP.

Os próximos capítulos serão estruturados da seguinte forma: No capítulo 2 o Youtube

e protocolo 802.11n serão apresentados e algumas particularidades serão discutidas. No capítulo 3 a infraestrutura de medição é apresentada, assim como parte da metodologia. No capítulo 4 são apresentada a forma que as métricas de interesse são obtidas a partir dos dados originais coletados. No capítulo 5 os experimentos são descritos, assim como os resultados obtidos a partir da análise dos dados. No capítulo 6 trabalhos futuros são propostos. O capítulo 7 apresenta a conclusão do trabalho. Para ver detalhes de como o roteador usado para os experimentos foi configurado ver a seção A do Apêndice.

2 TECNOLOGIAS EM FOCO

2.1 COMO O YOUTUBE FUNCIONA

Como mencionado anteriormente, a plataforma de streaming usada para os experimentos é o YouTube, logo, vale a pena detalhar a forma que o Youtube e a maioria das plataformas de vídeo funcionavam, e como progrediram ao longo do anos. Antigamente, a maioria dos serviços de streaming de vídeo eram fornecidos através do uso do protocolo UDP, porém, aos poucos, diferentes plataformas de streaming começaram a adotar streaming de vídeo usando HTTP através de TCP, incluindo o YouTube. Essa primeira forma de streaming usando HTTP consistia no download do vídeo inteiro antes da reprodução do mesmo. Rapidamente uma nova forma de streaming HTTP começou a ser adotada, que veio a ser conhecida como download progressivo. Nessa forma, o download do vídeo para o cliente é feito de forma progressiva, porém, com resolução fixa (KRISHNAPPA; BHAT; ZINK, 2013). Ou seja, segmentos de vídeos eram baixados progressivamente, enquanto o vídeo era reproduzido, porém, sempre correspondentes à mesma resolução. YouTube adotou esse formato de streaming, que permaneceu como o padrão no sítio por diversos anos. No final de 2011 um novo padrão ISO foi publicado, DASH (Dynamic Adaptive Streaming over HTTP), também conhecido como MPEG-DASH (SODAGAR, 2011). Em 2013 já existiam artigos que defendiam a adoção de DASH no YouTube (KRISHNAPPA; BHAT; ZINK, 2013) e em 2015 YouTube tornou essa forma de streaming padrão em todos os seus serviços (WIKIPEDIA, 2020). Hoje, todos os grandes serviços de streaming usam DASH.

Um dos problemas de download progressivo é que não possibilita que os segmentos de vídeo sejam adaptados para diferentes dispositivos e condições de rede, que é exatamente a proposta do DASH e, mais em geral, do HAS (HTTP Adaptive Streaming). HAS encode vídeos em diversos níveis de resolução e divide cada um desses níveis em segmentos de alguns segundos, disponibilizando-os através de HTTP. O cliente deve então requisitar o arquivo manifest, que no caso de DASH, é o Media Presentation Description (MPD), que descreve diversas propriedades do vídeo, como quantos níveis de resolução há, a localização e o codec usado para cada segmento. O cliente começa fazendo a requisição desse arquivo manifest e, subsequentemente, dos segmentos de vídeo. A escolha, no entanto, da resolução dos segmentos de vídeo, deve ser feita por um algoritmo de adaptação de bitrate, uma questão central à QoE em aplicações de vídeo (SODAGAR, 2011)(XIAOQI et al., 2015). A Figura 1 depicta esse processo, supondo que o algoritmo de adaptação de bitrate leva em conta somente as informações no arquivo manifest e estimação de throughput. DASH padronizou o formato desse arquivo manifest, porém o algoritmo de adaptação de resolução é completamente livre, dependente de implementação, por isso YouTube, Netflix

e outros serviços de streaming têm performances e comportamentos tão distintos. Nos experimentos, foi utilizada uma extensão para o Google Chrome desenvolvida por um grupo de pesquisa parceiro, INRIA/França, como descrita em (SCHMITT et al., 2019), para coletar os arquivos HAR, que são um log de todas as transações HTTP entre o browser e o sítio. Pelo fato do DASH ser usado pelo YouTube, a partir do arquivo HAR é possível obter dados numa granularidade bem fina, sendo os mais relevantes para esse esforço os descritos no capítulo 4.

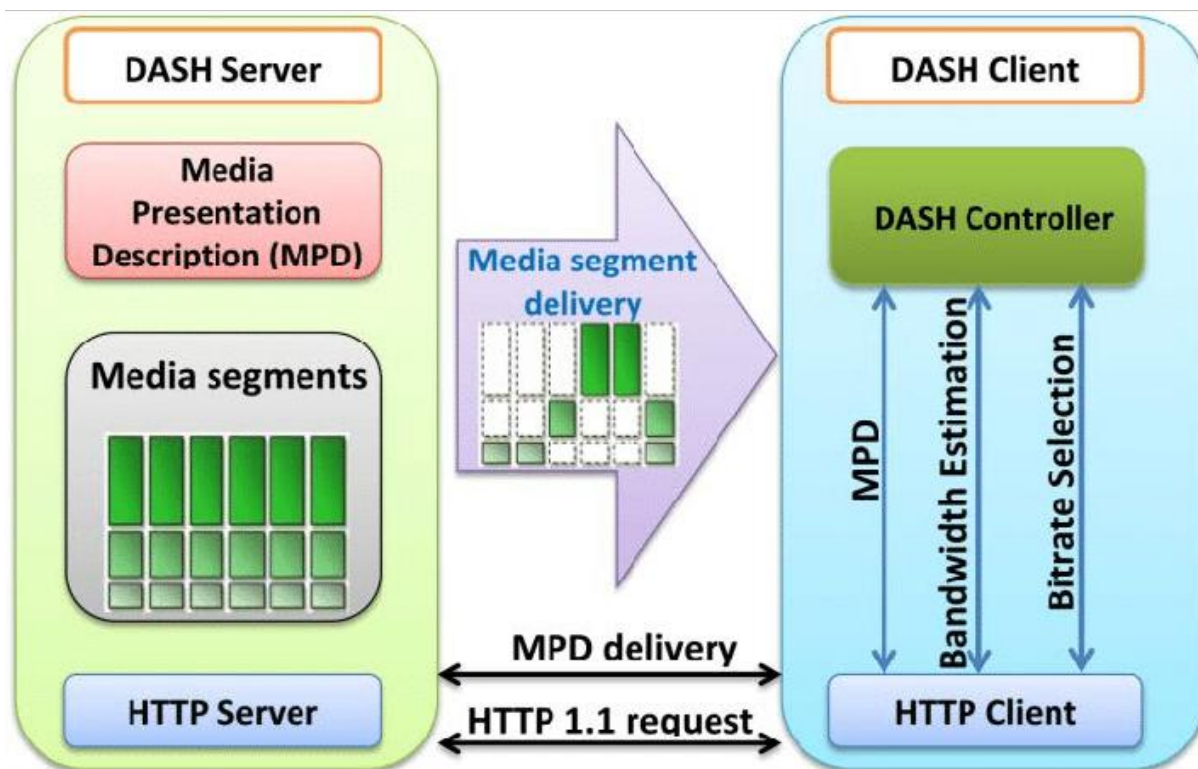


Figura 1 – Funcionamento do DASH. Fonte: (VU; MASHAL; CHUNG, 2017).

2.2 O BUFFER

Desde um pacote chegar no roteador até ser enviado para o computador onde os vídeos são reproduzidos ele passa por diversos buffers. É preciso entender onde ficam esses buffers e como se dá a dinâmica entre eles. O buffer que foi modificado nesse trabalho é o buffer de transmissão do driver WiFi do roteador, e, desse ponto em diante, quando for usado o termo "buffer WiFi" ou simplesmente "buffer", é esse buffer que estará sendo referido. Do ponto de vista de um pacote *downstream*, esse é o último buffer que ele atravessa no roteador. Para simplificar a análise, só será tratado o caso de saída de um pacote do roteador, pois esse trajeto percorre o buffer que foi modificado no trabalho, o último buffer que um pacote atravessa antes de chegar na placa de rede.

É importante entender que um buffer só é relevante se ele estiver no ponto de gargalo no trajeto dos pacotes. Suponha que temos uma conexão TCP entre Host A e Host

B somente com dois buffers em série (podemos abstrair que cada buffer representa um roteador ou switch), como representado na Figura 2, onde o Host A envia pacotes ao Host B. A capacidade dos links em cada ponto do trajeto é está indicada na figura. Como o link de saída do Buffer 2 tem capacidade menor que o link de entrada, quando a janela de congestionamento aumentar, o Buffer 2 não irá conseguir propagar os pacotes na mesma taxa em que os recebe, o que irá criar uma fila. Uma vez que essa fila encher o buffer pacotes serão descartados e, quando esses pacotes perdidos forem percebidos pelo Host A a janela irá diminuir e esse processo irá se repetir até que a conexão seja fechada. Logo, o ponto de gargalo nesse cenário é o Buffer 2. Analogamente, no cenário representado pela Figura 3, o ponto de gargalo é o Buffer 1. No primeiro cenário, o tamanho do Buffer 1 é irrelevante e no segundo cenário o tamanho do Buffer 2 é irrelevante. Para esse esforço é suposto que o buffer do RD é o ponto de gargalo, ou seja, que a capacidade do link entre o modem e o roteador é menor que a capacidade do link WiFi. Essa é uma suposição bem segura de se fazer, como poderá ser visto na próxima seção.

Em arquiteturas Unix/Linux, o buffer do driver fica entre a pilha IP e a placa de rede, como representado na Figura 4. Esse buffer, em geral, é circular, com política FIFO (First in, first out) como é o caso do buffer que foi testado neste trabalho. Cada entrada desse buffer possui um descritor de hardware que aponta para um SKB (Socket Kernel Buffer), onde o pacote é armazenado para ser manipulado pelo kernel. Quando a placa de rede está pronta para transmitir um pacote, é feita uma interrupção e então o kernel retira o pacote mais antigo do buffer e envia pelo barramento de dados até a placa de rede, para transmissão. Neste caso, a disponibilidade da placa de rede torna o buffer do driver o ponto de gargalo.

O buffer do driver possui diversas limitações. Ele não provê isolamento de fluxo, o que torna possível o fluxo proveniente de uma aplicação tomar grande parte da banda. Ele não provê isolamento por *hosts*, resultando num problema análogo, porém a nível de *hosts*. Além disso, ele não provê priorização de fluxos para serviços que necessitam de baixa latência, como é o caso de jogos online ou vídeo chamadas. No Linux, todas essas funcionalidades podem ser configuradas na camada de disciplinas de fila (QDiscs), que se encontra entre a pilha IP e o buffer do driver. O sistema operacional do roteador que foi usado nesse trabalho é o OpenWrt, que é uma distribuição Linux, logo, essa dinâmica também se aplica para o roteador de testes. Se o tamanho do buffer do driver for muito pequeno, filas começarão a ser formadas na camada de disciplinas de fila, essencialmente trazendo o controle de gerenciamento de filas e agendamento de pacotes para essa camada superior, que possui mais funcionalidades. Esse princípio levou à criação da ferramenta BQL (Byte Queue Limit), no linux (LWN, 2013), que drasticamente limita a quantidade de bytes que podem ser armazenados nos SKBs apontados pelo buffer do driver, de forma dinâmica, trazendo o gargalo à camada QDisc.

Nos experimentos, não foi usado nenhum algoritmo de gerenciamento de filas na ca-

mada QDisc, ou seja, o único buffer presente era o do driver WiFi, cuja quantidade de descritores de pacotes foi modificada de acordo com o experimento. No futuro, o buffer do driver será fixado e tanto BQL quanto mecanismos de gerenciamento de filas e agendamento de pacotes serão testados (HOEILAND-JOERGENSEN; TAHT; MORTON, 2018)(MACGREGOR; SHI, 2000).

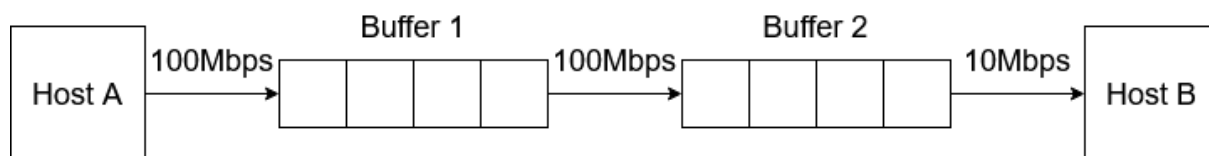


Figura 2 – Buffers em Série, Cenário 1

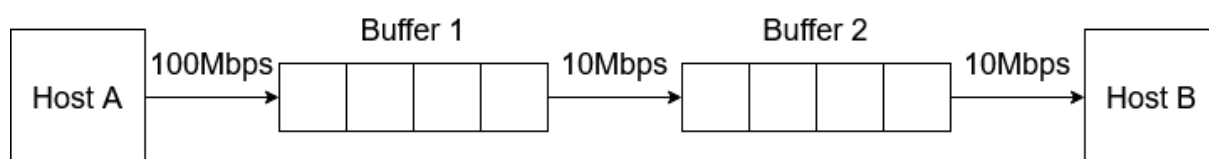


Figura 3 – Buffers em Série, Cenário 2

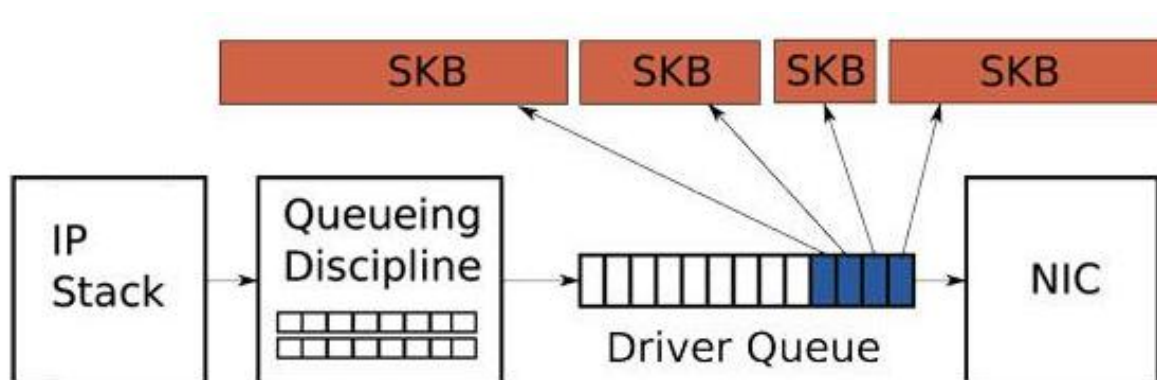


Figura 4 – Buffers de transmissão de roteador com OpenWrt. Fonte: (LINUX, 2013).

2.3 COMO FUNCIONA O 802.11N

Em 1997 o primeiro padrão IEEE 802.11 foi criado, especificando protocolos da camada MAC e da camada física possibilitando a implementação de WLANs (Wireless Local Area Networks) via comunicação WiFi. Desde a sua primeira iteração, diversas modificações foram feitas, com novas funcionalidades sendo adicionadas. Diferente de soluções com fio, como por exemplo o padrão IEEE 802.3 (Ethernet), no meio sem fio não é possível detectar transmissões de outros nós ao mesmo tempo em que se está transmitindo, o que impossibilita a detecção de colisões de quadros. Por isso, todos os padrões da família 802.11 utilizam o protocolo carrier-sense multiple access with collision avoidance

(CSMA/CA), da camada MAC. O protocolo estabelece que o nó que está enviando deve começar ouvindo o meio para detectar transmissões. Se detectar uma transmissão deve esperar um tempo aleatório, chamado de backoff. Se não detectar, transmite o seu quadro e aguarda um quadro ACK do receptor. Se não receber o quadro ACK depois de um tempo pré-estabelecido, espera um tempo aleatório chamado de binary exponential backoff até tentar transmitir novamente. Um problema que surge nesse protocolo é do nó escondido, representado na Figura 5.

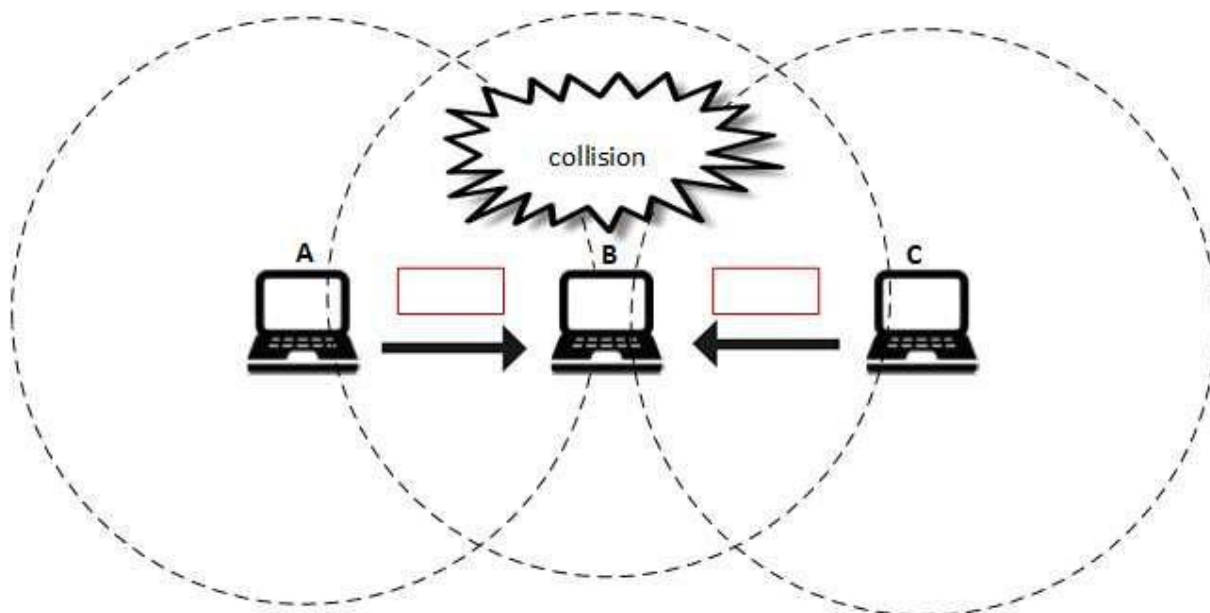


Figura 5 – Problema do Nó Escondido. Fonte: (WEYULU; HANADA; KIM, 2017).

O problema do nó escondido consiste no nó que está enviando não detectar um outro nó que também está enviando para o mesmo receptor, resultando em colisão. Uma medida para resolver isso é usar os quadros de controle RTS e CTS. Ao invés de enviar diretamente o quadro após perceber que o meio está livre, o nó envidor envia um quadro RTS (*Request to Send*) ao receptor e aguarda um quadro CTS (*Clear to Send*) do mesmo, que indica que o receptor reservou um espaço de tempo para que somente o primeiro nó envie. Isso diminui a probabilidade de colisão, porém, no caso de envio de quadros pequenos, pode resultar em maior latência.

O meio sem fio é muito vulnerável a interferências, tanto de fontes WiFi quanto fontes não WiFi (por exemplo, microondas e babás eletrônicas), que operam em frequências próximas de 2.4GHz, a frequência em que a maioria dos APs (Pontos de Acesso) operam, o que pode causar perda de quadros. Juntando a aleatoriedade do backoff com a vulnerabilidade a interferências, diversas questões são mais difíceis de resolver no WiFi do que no meio com fio, como por exemplo, a justiça de acesso ao meio em redes Ad-Hoc (MEHAOUED; BOURENANE; SEKHRI, 2020) ou, mais surpreendentemente, o cálculo da capacidade do meio (HORA et al., 2016). Mesmo décadas depois de sua primeira im-

plementação, 802.11 continua sendo um padrão muito estudado, com diversos problemas ainda ativamente discutidos.

O modo "n" do padrão, publicado em 2009, trouxe algumas funcionalidades que contribuíram para a maior onda de disseminação de uso de WiFi desde sua primeira implementação. Duas das principais adições do 802.11n foram agregação de quadros e MIMO (Multiple Input Multiple Output) com junção de canais e adaptação de taxa. Abaixo essas funcionalidades são descritas e a interação delas com o tamanho do buffer é discutida.

Agregação de Quadros

Uma medida que diminui o overhead da camada MAC, agregação de quadros, permite que diversos MPDUs (*Mac Protocol Data Units*) sejam agregados, formando um A-MPDU, e enviados em um único acesso ao meio. Para a resposta do envio desses quadros, é necessário o uso de Block ACKs, permitindo que diversos MPDUs sejam reconhecidos em um único acesso ao meio. O padrão define que o tamanho máximo de A-MPDUs é de 65,535 bytes, porém, o tamanho de fato depende da implementação, com cada driver de WiFi tendo o seu próprio algoritmo de agregação de quadros. Pode-se perceber que o tamanho do buffer está intrinsecamente ligado à agregação de quadros. Um tamanho de A-MPDU grande pode acarretar na necessidade de um buffer grande, se não, pode ocorrer perda de quadros. Já se os A-MPDUs não forem tão grandes, pode não ser necessário ter um buffer tão grande. De fato, buffers grandes demais podem ser prejudiciais em alguns cenários. Suponhamos que temos uma conexão TCP aberta, se os A-MPDUs forem pequenos, porém a taxa física for alta e o buffer do receptor for muito grande, pode ser que o buffer demore bastante para encher. Quando o buffer encher, o algoritmo de controle de congestão do TCP vai acionar, alertando ao enviador que houve congestão que então levará a um aumento da janela de contenção. Entretanto, pela demora para haver esse feedback ao enviador isso pode tornar o fluxo TCP mais instável (JOHARI; TAN, 2001) e pode haver um atraso bem grande até a taxa correta ser atingida, levando a subutilização do meio. Na literatura, esse fenômeno é chamado de atraso por fila.

MIMO com Junção de Canais e Adaptação de Taxa

802.11n introduziu MIMO, que possibilita o uso de múltiplas antenas para transmissão de dados. MIMO possui dois modos de operação, Single Stream (SS), que usa diversidade espacial, e Double Stream (DS), que usa multiplexação espacial. Esses modos especificam como os dados vão ser tratados e por quais antenas serão transmitidos. A junção de canais permite que dois canais de 20MHz sejam usados para transmitir numa frequência efetiva de 40MHz, um canal para controle e outro para extensão, o que, essencialmente permite dobrar a taxa. Usando diferentes modulações e esquemas de códigos (MCS) é possível obter taxas físicas de até 600Mbps. A escolha de qual MCS usar é feita pelo algoritmo de adaptação de taxa, e, assim como o algoritmo de agregação de quadros, depende da implementação. É fácil ver que, assim como o caso da agregação de quadros, a taxa física está intrinsecamente relacionada ao tamanho do buffer, com os mesmos problemas

podendo surgir.

Feitas essas observações, é seguro dizer que o buffer é uma das peças centrais na interação de funcionalidades do protocolo 802.11. De fato, será mostrado mais adiante que a QoE do usuário está muito relacionada ao tamanho do buffer, o que torna-o um objeto de estudo muito importante.

3 INFRAESTRUTURA E METODOLOGIA

O laboratório do qual eu faço parte, o LAND, tem uma parceria com um ISP de médio porte, Gigalink e uma startup incubada na UFRJ, Anlix. A partir dessa parceria nós conseguimos coletar medidas passivas e ativas dos roteadores domésticos de milhares de clientes do ISP. Nós não coletamos os headers dos pacotes ou qualquer dado que possa comprometer a privacidade do usuário. RDs são o lugar ideal para coletar medidas pois o tráfego de todos os dispositivos domésticos deve, necessariamente, passar por ele, exceto em poucos casos, como redes móveis. O fato da análise ser feita nos RDs também significa que não é necessário hardware sofisticado para fazer coleta e análise de fluxos agregados no backbone da rede. Os RDs na rede da Gigalink possuem um software livre como sistema operacional (SO), OpenWrt, e a cada minuto enviam informações para os nossos servidores localizados no LAND para análise de dados. A granularidade de um minuto é definida por restrições na infraestrutura do servidor de coleta. A Figura 6 ilustra a infraestrutura de medição em cada uma das residências participantes.

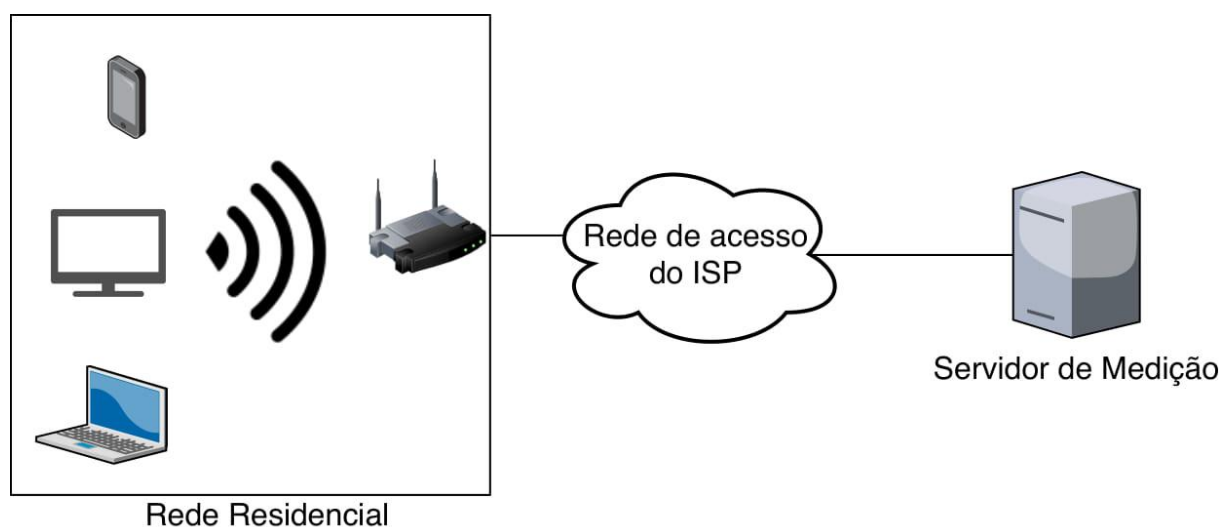


Figura 6 – Infraestrutura de Medição. Fonte: (ATKINSON et al., 2019)

A Tabela 1 contem as métricas gerais de rede que são coletadas das residências participantes. Todas essas métricas são coletadas dentro do intervalo de um minuto.

Também são coletadas métricas especificamente do WiFi, numa granularidade de um minuto. A Tabela 2 lista e descreve essas métricas.

Tabela 1 – Métricas de Rede Coletadas

MÉTRICA DE REDE	DESCRIÇÃO
Quantidade de Bytes de Upload (QBU)	Quantidade total de bytes que trafegam upstream do RD
Quantidade de Bytes de Download (QBD)	Quantidade total de bytes que trafegam downstream do RD
Quantidade de Pacotes de Upload (QPU)	Quantidade total de pacotes que trafegam upstream do RD
Quantidade de Pacotes de Download (QPD)	Quantidade total de pacotes que trafegam downstream do RD
Quantidade de Pacotes Perdidos (QPP)	Quantidade total de pacotes que são perdidos
Latência Média (LM)	Média do RTT de até 100 pacotes

Tabela 2 – Métricas de WiFi Coletadas

MÉTRICA WIFI	DESCRIÇÃO
Qualidade do Sinal (RSSI)	Qualidade relativa do sinal WiFi, indica a potência do sinal
Relação Sinal-Ruído (SNR)	Razão de sinal para ruído
Quantidade de Bytes de Upload WiFi (QBUW)	Quantidade total de bytes que trafegam upstream do RD via WiFi
Quantidade de Bytes de Download WiFi (QBDW)	Quantidade total de bytes que trafegam downstream do RD via WiFi
Quantidade de Pacotes de Upload WiFi (QPUW)	Quantidade total de pacotes que trafegam upstream do RD via WiFi
Quantidade de Pacotes de Download WiFi (QPDW)	Quantidade total de pacotes que trafegam downstream do RD via WiFi
Bitrate Médio de Upload WiFi (BMUW)	Bitrate médio que trafega upstream do RD via WiFi
Bitrate Médio de Download WiFi (BMDW)	Bitrate médio que trafega downstream do RD via WiFi

Nós estivemos coletando as métricas gerais de aproximadamente 4000 residências por diversos meses e planejamos coletar de milhares de residências a mais no futuro, o quanto a infraestrutura do LAND permitir. Atualmente, um número menor de residências participantes também tem as métricas WiFi coletadas, porém, está sendo feito um estudo atual para descobrir qual é o limite, em relação à carga de envio e escrita de dados, de residências que podem ter essas e as métricas gerais coletadas, de forma a maximizar os

dados coletados. Essas medidas podem ser interpretadas como séries temporais de milhares de usuários, que podem ser usadas para avaliar a performance da rede, a QoE do usuário e detectar ataques. Além disso, nós também possuímos a topologia da rede, de acordo com o MAC dos RDs. Isso permite que avaliações da rede, como um todo, ou por partes específicas, possam ser feitas, de acordo com as séries temporais de usuários ou grupos de usuários específicos, que podem ser mapeados nessa topologia.

Para realizar o esforço relacionado à QoE de vídeo percebida dos usuários, foi utilizada a extensão criada pelo grupo INRIA/França, para coletar dados do youtube que permitem a obtenção de métricas de QoE (essa extensão funciona com diversos serviços de streaming de vídeo, como youtube, netflix, hulu, twitch, entre outros, mas escolhi focar somente no youtube nesse trabalho, como prova de conceito). Essa extensão gera, a partir do vídeo sendo reproduzido, um arquivo JSON com os atributos de objeto de vídeo javascript, que variam ao longo da reprodução, e também coleta o arquivo HTTP Archive (HAR) que é um log das interações HTTP entre o browser e o sítio, formatado em JSON por padrão. Eu criei e subi um servidor em Node.js em uma das máquinas do LAND e redirecionei esses arquivos para serem enviados para esse servidor. Eu então escrevi um código em Python, usando a biblioteca Selenium, para automatizar as reproduções assim como um parser para extrair importantes métricas de QoE a partir dos arquivos HAR e dos arquivos que contém os atributos Javascript. A tabela 3 abaixo contém os nomes das métricas de QoE e suas descrições:

Tabela 3 – Métricas de QoE Coletadas

MÉTRICA QOE	DESCRIÇÃO
Latência Inicial (LI)	Tempo entre a página começar a carregar e o vídeo começar a reproduzir
Bitrate Médio (BM)	Taxa média de bits decodificados em Mbps
Resolução Média (RM)	Taxa média de resolução do vídeo (quantizada em níveis de 1 a 8 correspondentes as resoluções 144p a 2160p)
Quantidade de Mudanças de Resolução (QMR)	Quantidade de vezes que a resolução do vídeo mudou
Quantidade de Rebufferizações (QR)	Quantidade de vezes que ocorreu um evento de rebufferização
Razão de Rebufferizações (RR)	Razão do tempo total de rebufferização sobre o tempo total de reprodução do vídeo
Duração de Rebufferizações (DR)	Duração agregada de todos os eventos de rebufferização

Para todas essas métricas é fácil ter uma intuição de como elas correlacionam com a QoE. Por exemplo, quanto maior LI, mais tempo o usuário terá que esperar para assistir o vídeo, o que piora a QoE. Quanto maior QMR, mais vezes o vídeo irá ter sua resolução modificada, o que torna a experiência menos fluida para o usuário, diminuindo sua QoE. Quanto maior DR, RR e QR, mais tempo total terá de rebufferização, o que é pior para a QoE. Já com BM e RM, quanto maior essas métricas, melhor a QoE, pois quanto maior for a resolução, melhor a experiência do usuário. Apesar disso ser intuitivo, a forma que essas métricas se relacionam pode não ser, ainda mais quando adicionamos dados sobre o tamanho do buffer do driver de WiFi. Isso será explorado mais no capítulo 5.

Todas as reproduções foram feitas numa máquina com SO Linux Ubuntu 18.04, processador Intel Core i7-2600 de 3.4GHz com 8 núcleos e 16GB de memória RAM DDR3. Alguns dos experimentos tiveram perda de pacote simulada. Isso foi feito na mesma máquina onde os vídeos foram reproduzidos com o uso do `tc-netem`. No futuro, será usado um roteador a mais, para que a perda seja simulada antes no trajeto downstream, como foi feito com a emulação de tráfego em (PITTONI, 2016).

Todos os experimentos foram feitos a partir de uma conexão WiFi, usando o protocolo 802.11n. O roteador usado é um Archer C20 AC750 v4, da TP-Link, que possui uma CPU de 580 MHz, memória Flash de 8MB e memória RAM de 64MB. Em todos os experimentos o roteador foi colocado a cerca de 2 metros do desktop onde os vídeos estavam sendo reproduzidos, de forma a diminuir a interferência com o sinal de outros roteadores na vizinhança. A frequência usada foi 2.4GHz, porém, para experimentos futuros a frequência usada será 5GHz, para evitar ao máximo interferência com sinais de outros roteadores (PITTONI, 2016).

Todos os vídeos que foram usados nos experimentos são de ação e têm resoluções que variam entre 144p e 2160p (4K), ou entre 0.085Mbps e 15Mbps (bitrate médio). Escolhi esses vídeos pelo amplo alcance de resoluções e também para emular o “pior caso possível”, vídeos muito dinâmicos onde as possibilidades de compressão são limitadas. Em experimentos futuros outros tipos de vídeos também serão testados, porém, ainda com a mesma faixa de resoluções.

4 OBTENÇÃO DAS MÉTRICAS DE QOE

A extensão que foi desenvolvida pela equipe parceira, INRIA/França, permite a coleta de atributos de objeto de vídeo Javascript e arquivos HAR. O objeto vídeo Javascript possui diversas propriedades, algumas das quais são pertinentes para esse trabalho. Segue abaixo a Tabela 4 com essas propriedades e suas descrições.

Tabela 4 – Propriedades de Objetos Vídeo Javascript

PROPRIEDADE	CHAVE	DESCRIÇÃO
buffered	BUF	retorna um objeto TimeRanges representando as partes que foram bufferizadas de um vídeo
currentTime	CUT	define ou retorna a atual posição de playback de um vídeo (em segundos)
duration	DUR	retorna a duração do vídeo (em segundos)
webkitDecodedFrameCount	WFC	retorna a quantidade de frames que foram decodificados, cumulativamente
webkitDroppedFrameCoun	WDR	retorna a quantidade de frames que foram perdidos, cumulativamente
webkitVideoDecodedByteCount	WVD	retorna a quantidade de bytes que foram decodificados, cumulativamente

Assim como propriedades, também existem eventos associados aos objetos vídeo, alguns deles sendo bem importantes. A Tabela 5 mostra os eventos pertinentes, com seus códigos e descrições:

Um dos arquivos que a extensão envia como output para o nosso servidor local é um log, em formato JSON, dessas propriedades e eventos, ao longo da reprodução do vídeo. Escrevi um código em Python para extrair, a partir dessas propriedades, eventos e os timestamps associados a cada um, presentes nesse log, as seguintes métricas de QoE: LI, BM, QR, RR e DR. A seguir será explicado como cada uma dessas métricas foi obtida.

LI é o tempo entre a plataforma carregar e o vídeo começar a reproduzir. Seja EVE_i o número do evento correspondente à i -ésima atualização. Seja n a quantidade de atualizações de eventos Javascript no log e P o conjunto totalmente ordenado de todos os índices i , $i = 1, 2, \dots, n$, tal que $EVE_i = 13$, então

$$LI = ets_{\min P} - t_i, \quad (4.1)$$

Tabela 5 – Eventos de Objetos Vídeo Javascript

EVENTO	ID	DESCRIÇÃO
canplay	2	é ativado quando o browser pode começar a reproduzir o áudio/vídeo
canplaythrough	3	é ativado quando o browser pode começar a reproduzir o áudio/vídeo sem parar para bufferizar
pause	11	é ativado quando o áudio/vídeo é pausado
play	12	é ativado quando o áudio/vídeo começa a ser reproduzido ou não está mais pausado
playing	13	é ativado quando o áudio/vídeo está reproduzindo após ter sido pausado ou parado para bufferizar
waiting	22	é ativado quando o vídeo para porque precisa bufferizar o próximo frame

onde ts_i é o timestamp correspondente ao evento i e ti o tempo inicial, indicado no início do arquivo. Ou seja, quando o primeiro evento "playing" é disparado, esse é o momento em que terminou o período de LI, pois é o momento em que terminou a primeira bufferização do vídeo. Pegando a diferença entre esse timestamp e o timestamp correspondente ao início do log obtemos a LI.

BM é um número que equivale à taxa média de decodificação de bits para o vídeo inteiro. Além de BM, também defini uma sequência de números B indicando a taxa de decodificação de bits ao longo do vídeo. Encontrar BM é simples, basta pegar o último valor da propriedade WVD disponível no arquivo e dividir pelo tempo total do vídeo. Ou seja,

$$BM = \frac{WVD_m}{(tf - ti)}, \quad (4.2)$$

onde m é a quantidade de atualizações da propriedade WVD , WVD_i é o valor da propriedade WVD correspondente à i -ésima atualização e tf é o tempo final, indicado no início do arquivo. No caso da sequência de taxas temos

$$B_i = \frac{(WVD_{i+1} - WVD_i)}{(ts_{i+1} - ts_i)}, \quad i = 1, 2, \dots, m - 1, \quad (4.3)$$

onde ts_i é o timestamp correspondente à i -ésima atualização da propriedade WVD . É possível coletar os dados para a sequência B numa granularidade mais grossa, sendo o BM uma degeneração de B, mais especificamente, o caso em que somente pegamos WVD_1 e WVD_{m-1} .

Para calcular DR é necessário somar o tempo de todas as rebufferizações. Isso não conta a bufferização inicial, que é o tempo de LI. Nos testes feitos em laboratório o vídeo não foi pausado pelo cliente, a reprodução só parou por conta de eventos de bufferização. Isso significa que o evento número 13, "playing", só ocorreu após um evento de número 22, "waiting", também indicando que a quantidade de atualizações correspondentes a

ambos esses eventos é a mesma. O conjunto P já foi definido anteriormente. Agora, será definido W , o conjunto totalmente ordenado de todos os índices i , $i = 1, 2, \dots, n$, tal que $EVE_i = 22$. Logo,

$$DR = \sum_{i=2}^l (ets_{P(i)} - ets_{W(i)}), \quad (4.4)$$

onde l é a quantidade de elementos em P e W , $P(i)$ é o i -ésimo termo do conjunto P . Note que começamos em $i = 2$ pois $i = 1$ corresponde a LI.

Para calcular QR basta contar a quantidade de vezes que tivemos um evento de número 22, "waiting" e diminuir um, pois o primeiro corresponde à bufferização que resulta em LI. Logo,

$$QR = |W| - 1. \quad (4.5)$$

Além das métricas de QoE, os logs de objetos vídeo do Javascript também me permitiu obter dados referentes à ocupação do buffer, mais especificamente, que segmento do vídeo o buffer estava guardando. Feito isso para cada resolução, desde 144p até 2160p, juntando com a informação do playback do vídeo e a sequência de bitrates B , foi possível obter insight sobre o algoritmo de adaptação de resolução do youtube e o *data wastage* resultante. O tópico de *data wastage* no youtube já foi bastante investigado (KRISHNAPPA; BHAT; ZINK, 2013)(ANORGA et al., 2016), porém, para esse esforço não é relevante.

Os arquivos HAR de cada vídeo também foram úteis. Esse tipo de arquivo registra as interações HTTP entre o browser e a página. A partir desse arquivo é possível extrair informações importantes que estão contidas como valores para parâmetros no url de diversas requisições. Em (MONDAL, 2017) os autores analisam diversos desses parâmetros e detalham seus comportamentos e os significados que eles os atribuem. Na Tabela 6, abaixo, estão os parâmetros de interesse para esse trabalho e suas descrições:

Tabela 6 – Parâmetros HTTP do Youtube

PARÂMETRO	DESCRIÇÃO
itag	um identificador único para propriedades como container, resolução e codec do vídeo
rbuf	denota a quantidade de bits no buffer da aplicação

A partir do itag, junto com as respectivas timestamps, foi possível obter duas novas métricas de QoE: RM e QMR. A seguir será explicado como essas métricas foram obtidas.

De forma similar a BM, no caso de RM também faz sentido criar uma sequência R indicando a resolução ao longo do vídeo. A resolução, diferente do caso da propriedade WVD , não é cumulativa, então é melhor começar definindo a sequência R :

$$R_i = Q(D(itag_i)), \quad i = 1, 2, \dots, c, \quad (4.6)$$

onde c é a quantidade de segmentos de vídeo requisitados e D é uma função que mapeia itags para resoluções (um dicionário foi usado no código) e Q é uma função que mapeia resoluções para quantizações, definida da seguinte forma:

$$Q(res) = \begin{cases} 1, & \text{se } res = 144 \\ 2, & \text{se } res = 240 \\ 3, & \text{se } res = 360 \\ 4, & \text{se } res = 480 \\ 5, & \text{se } res = 720 \\ 6, & \text{se } res = 1080 \\ 7, & \text{se } res = 1440 \\ 8, & \text{se } res = 2160, \end{cases} \quad (4.7)$$

onde res está na unidade de pixels (p). Para RM, temos,

$$RM = \frac{1}{c} \sum_{i=1}^c R_i. \quad (4.8)$$

Note que uma forma mais precisa de calcular seria ponderar cada parcela R_i pela duração exata de cada segmento, porém, a diferença não é significativa. Podemos confirmar isso ao ver, no capítulo 5 que o *shape* das métricas RM e BM é bem similar. De fato, para a maioria dos experimentos, somente a métrica BM foi usada.

Para a métrica QMR, temos

$$QMR = \sum_{i=2}^c I(itag_i, itag_{i-1}), \quad (4.9)$$

onde I é uma função indicadora definida da seguinte forma,

$$I(a, b) = \begin{cases} 1, & \text{se } D(a) \neq D(b) \\ 0, & \text{se } D(a) = D(b). \end{cases} \quad (4.10)$$

O parâmetro *rbuf* traz uma outra forma de calcular a duração agregada dos eventos de rebufferização. Vale notar que existem outros parâmetros nas urls das requisições que podem ser úteis, assim como propriedades e eventos obtidos a partir dos objetos vídeo Javascript que certamente nos dão maior insight para o algoritmo de adaptação de bitrate e outras decisões da plataforma e que, de fato, foram investigados, porém, esses são os que foram pertinentes para esse esforço.

Apesar desses dois arquivos, obtidos com a extensão, já conterem muita informação, notou-se que havia uma oportunidade que não estava sendo explorada, uma facilidade que o youtube apresenta aos seus usuários: estatísticas para nerds. Em qualquer vídeo da plataforma pode-se clicar com o botão direito sob o *Viewport* e selecionar a opção *stats*

for nerds ou estatísticas para nerds. Isso abrirá uma janela que exibirá, dinamicamente, diversas propriedades do vídeo. Abaixo, a Tabela 7 mostra as mais importantes, seguido de seus significados:

Tabela 7 – Propriedades de Estatísticas para Nerds

PROPRIEDADE	DESCRIÇÃO
Viewport / Frames	indica as dimensões da tela de exibição do vídeo e, separado por uma barra, quantos frames foram perdidos (ex.: 0 dropped of 45)
Current / Optimal Res	indica a resolução e framerate atual e, separado por uma barra, a resolução e framerate ideal
Codecs	indica o codec atual
Connection Speed	estimativa de throughput feita pela plataforma
Buffer Health	indica quantos segundos o buffer da aplicação tem armazenado, em segundos frame

É possível ver que, a partir disso temos algumas informações redundantes, porém, agora, de mais fácil acesso, como a resolução e codec atual, a quantidade de dados armazenados no buffer da aplicação e a quantidade de frames perdidos. Adicionalmente, obtemos agora a estimativa de throughput e o tamanho do *viewport*, duas informações que o próprio Youtube já divulgou serem usadas no algoritmo de adaptação de bitrate, e que podem ser úteis para trabalhos futuros. Visto isso, um código foi criado em Python que, além de automatizar a reprodução dos vídeos, também coleta esses dados numa granularidade de meio segundo e os formata em JSON, para fácil acesso.

As reproduções que foram feitas e analisadas nesse esforço não contam com esses dados, porém, para trabalhos futuros esses dados serão levados em conta para análise.

5 EXPERIMENTOS E RESULTADOS

5.1 EXPERIMENTOS

A Tabela 8, abaixo, mostra todos os experimentos que foram feitos ao longo desse esforço, em ordem cronológica. Todos os vídeos usados para os experimentos são trailers que possuem resoluções que vão de 144p até 2160p (4K).

Tabela 8 – Experimentos em Ordem Cronológica

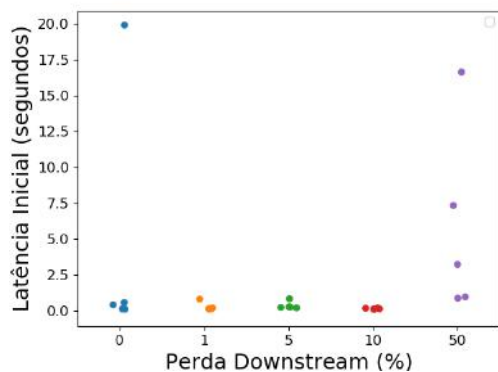
TRAILER	ID	DUR	PERDA	QUA	TAM_BUF	N_REP
Carro	-	1:45	0, 1, 5, 10, 50	auto	256	5
Rei Leão	g-Lqix0QaU4	1:46	0, 5, 10, 50	auto	256	10
Aquaman	9Yam5B _i asY	5:29	0, 5, 10, 15	auto	256	5
Aquaman	9Yam5B _i asY	5:29	0, 5, 10, 15	240p-2160p	256	1
Aquaman	bJA5nOzDf9o	2:35	0	auto	10, 20, 30, 256	10
Aquaman	bJA5nOzDf9o	2:35	0	auto	10, 20, 30, 40, 256	10

No primeiro experimento foi usado um trailer de um jogo de carro, com duração de 1m 45s. O vídeo foi excluído da plataforma desde a condução dos experimentos, logo, não há um ID válido para o mesmo. Cinco reproduções foram feitas para cada uma das seguintes taxas de perda downstream: 0%, 1%, 5%, 10% e 50%, mantendo o tamanho de buffer padrão de 256 pacotes e deixando a qualidade em *auto*, ou seja, deixando o algoritmo de adaptação de resolução do YouTube definir a qualidade. No total foram realizadas 25 reproduções, cinco para cada um dos *setups*. O objetivo desse primeiro experimento foi me familiar com a mudança da taxa de erro, usando a ferramenta *tenetm*, e, simultaneamente, ter algum entendimento da QoE sob a condição padrão do buffer sob diferentes taxas de perdas simuladas. Nessa época eu não sabia que era possível obter a resolução dos vídeos a partir do arquivo HAR então descartei-os, porém, a partir dos arquivos de javascript, consegui obter as métricas LI, DR, RR, QR e BM que são representadas nas Figuras 7a - 7e. Essas figuras são *striplots*, onde cada ponto representa uma reprodução. A variação no eixo horizontal é puramente para facilitar a diferenciação dos pontos e faz parte da implementação da biblioteca Seaborn, do Python, ou seja, as perdas simuladas foram exatamente 0%, 1%, 5%, 10% e 50%. Para experimentos futuros a métrica RR não será exibida visto que os gráficos de DR e RR sempre possuem o mesmo *shape*. Outra forma de entender isso é que a métrica RR sempre possui uma alta

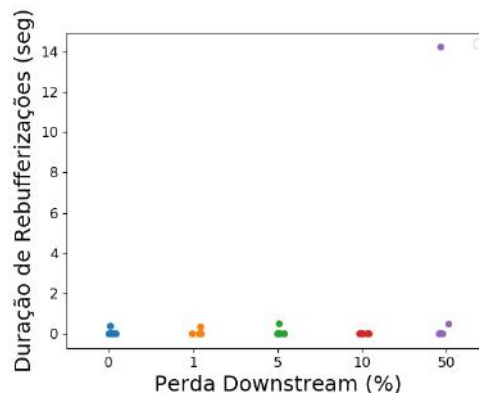
correlação positiva com DR, ou seja não adiciona muita informação referente à QoE. O mesmo é verdade para a métrica QR. Visto isso, a única métrica que será exibida de agora em diante, relacionada com o buffer da aplicação, é DR. Percebi que, com perda 0% houve mais LI do que com perda 10% e especulei que isso se dava pelo fato de que o algoritmo de adaptação de resolução estava sendo afetado pela sua estimativa de throughput (exibido como *network speed* nas estatísticas para nerds) o que resultaria na escolha de download de segmentos de resolução mais alta, no caso de perda de 0%, quando comparado ao caso de perda 10%, resultando em uma LI maior. Entretanto, ao olharmos BM é possível ver que a taxa média de bits decodificados ao longo dos vídeos reproduzidos com perda 0% é mais baixa do que em todos os casos de perda 1%, 5% e 10%, e, fazendo a média entre todas as reproduções, também menor do que no caso de perda 50%. As primeiras reproduções do vídeo nesse experimento foram feitas com perda 0%, pode ser que tenha ocorrido uma mudança na condição da rede entre o setup dos experimentos com perda 0% e os com perda 5%. Vale também ressaltar que cinco reproduções é um tamanho de amostra muito pequeno para tirar conclusões significativas. O motivo para essa anomalia ainda não foi determinado.

Para o experimento seguinte escolhi um outro vídeo, o trailer 2 do filme do Rei Leão (2019), de duração 1m46s. Para esse vídeo, realizei os mesmos experimentos, porém, dessa vez somente com perdas downstream de 0%, 5%, 10% e 50%, e 10 reproduções para cada configuração ao invés de 5, totalizando 40 reproduções. Novamente, as métricas QMR e RM não foram extraídas dos arquivos HAR. Como pode ser visto na Figura 8b, onde as reproduções de perda 50% foram omitidas, a LI é maior no caso em que não há nenhuma perda do que no caso em que há perda de 5% e 10%, exceto para um caso anômalo nos testes com perda 5% e um caso anômalo nos testes com perda 10%. Esse comportamento é consistente com o que foi observado no experimento anterior, com as reproduções com perda 50% tendo a maior variância de LI e, em média, maior LI do que as reproduções com as outras taxas, como pode ser visto na Figura 8a. Entretanto, diferente do experimento anterior, a métrica BM (Figura 8e) é significativamente mais alta nas reproduções de perda 0% do que nas reproduções com altas taxas de perda, o que é um comportamento muito diferente do que ocorreu no primeiro experimento. Além disso, a diferença da métrica BM entre as reproduções com taxa de perda 0% e as reproduções com outras taxas é maior do que o esperado, indicando que talvez também haja alguma anomalia nesse experimento. A métrica DR (Figura 8c) possui comportamento similar para as reproduções com perdas 0%, 5% e 10%, sendo muito mais alta nas reproduções com perda 50%. De fato, a diferença é tão pequena para as perdas 0%, 5% e 10% que é necessário tirar o caso de perda 50% para ver a diferença entre os valores da métrica, como está retratado na Figura 8d. O caso de perda 50%, indiscutivelmente apresenta QoE bem pior do que os outros. Já para os casos de perda 0%, 5% e 10%, se só olhássemos LI e DR seria tentador dizer que a QoE nos casos de perda 5% e 10% é superior aos casos de

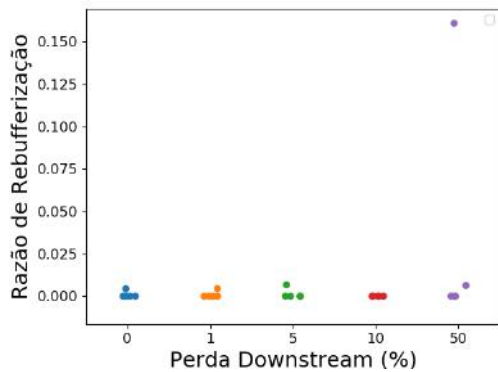
perda 0%, porém, como podemos ver na Figura 8e, o BM nos casos de perda 0% é muito maior, fazendo a média entre as reproduções, do que nos casos de perda 5% e 10%, o que, intuitivamente, é muito mais significativo para a QoE.



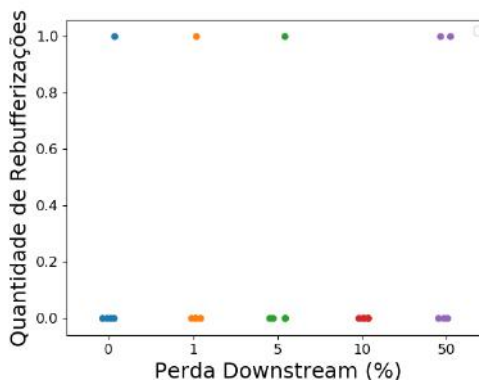
(a) Latência Inicial por Perda



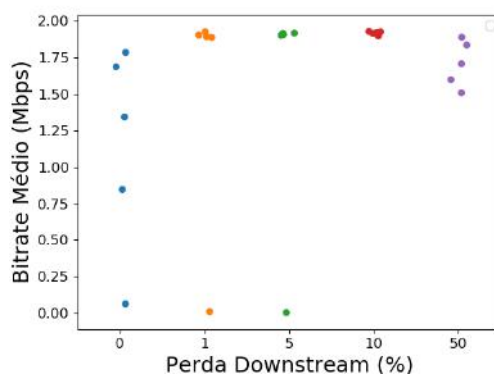
(b) Duração de Rebufferizações por Perda



(c) Razão de Rebufferizações por Perda



(d) Quantidade de Rebufferizações por Perda



(e) Bitrate Médio por Perda

Figura 7 – Métricas de QoE por Perda para o Trailer do Carro

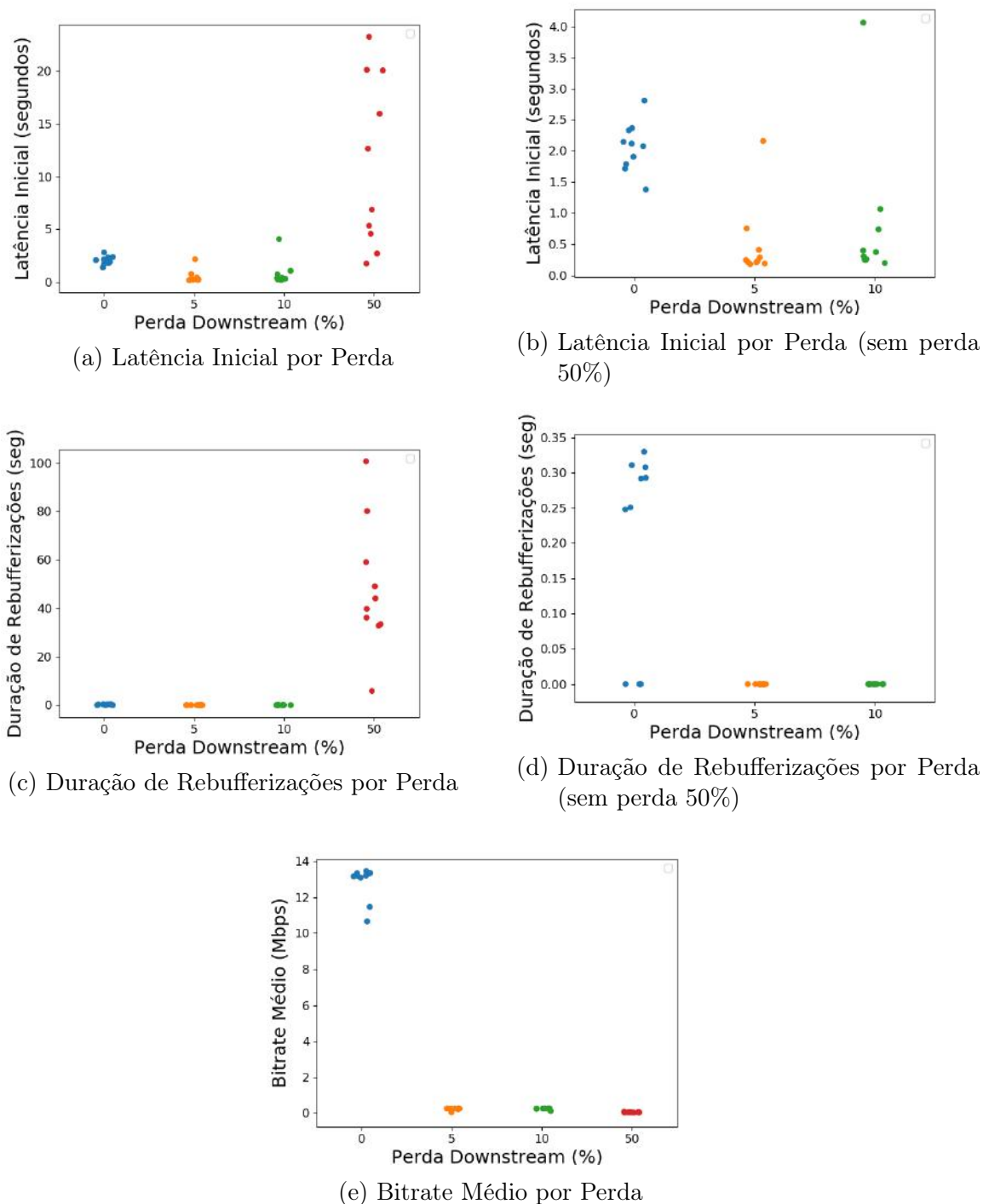


Figura 8 – Métricas de QoE por Perda para o Trailer do Rei Leão

No experimento seguinte, o vídeo usado novamente foi diferente, sendo agora um trailer do Aquaman, com duração de 5m29s, bem mais longo que os anteriores. Novamente as reproduções foram realizadas em *auto*, mas dessa vez as perdas downstream testadas foram 0%, 5%, 10% e 15%. O vídeo foi reproduzido cinco vezes para cada uma das configurações de perda, com o tamanho de buffer do driver WiFi downstream mantido em default, 256 pacotes, totalizando 20 reproduções. Novamente, as mesmas métricas de QoE foram

obtidas que nos experimentos anteriores, entretanto, ao contrário dos dois experimentos anteriores, a métrica LI (Figura 9a), no caso de perda 0%, é menor, fazendo a média entre as reproduções, do que no caso de perda 5% e 10%. Uma grande diferença entre esse experimento e o anterior é que, como pode ser visto na Figura 9c, não parece haver uma diferença significativa entre a métrica BM dos experimentos com perda 0%, 5% e 10%. Algumas coisas a se notar são as diferenças entre todos esses três primeiros experimentos: Os vídeos são diferentes, logo, os codecs também são, o tamanho desse vídeo também é bem maior do que os dos experimentos anteriores e todos os experimentos foram feitos em dias diferentes, ou seja, condições locais de rede sem fio provavelmente estavam bem diferentes. Este último fator foi remediado significativamente, com todos os experimentos sendo feitos no mesmo local e horário, com o mesmo computador para reprodução dos vídeos, sempre a dois metros de distância do roteador, porém, WiFi é muito sensível a sinais na mesma faixa de frequência, o que torna-o inconsistente (SEQUEIRA et al., 2012). Mais adiante no capítulo esse fenômeno se tornará saliente pela comparação de dois experimentos. Uma coisa que foi consistente entre todos os experimentos até agora é que as reproduções com maior perda apresentaram maior variância na métrica LI. O tópico de variância das métricas será aprofundado mais adiante.

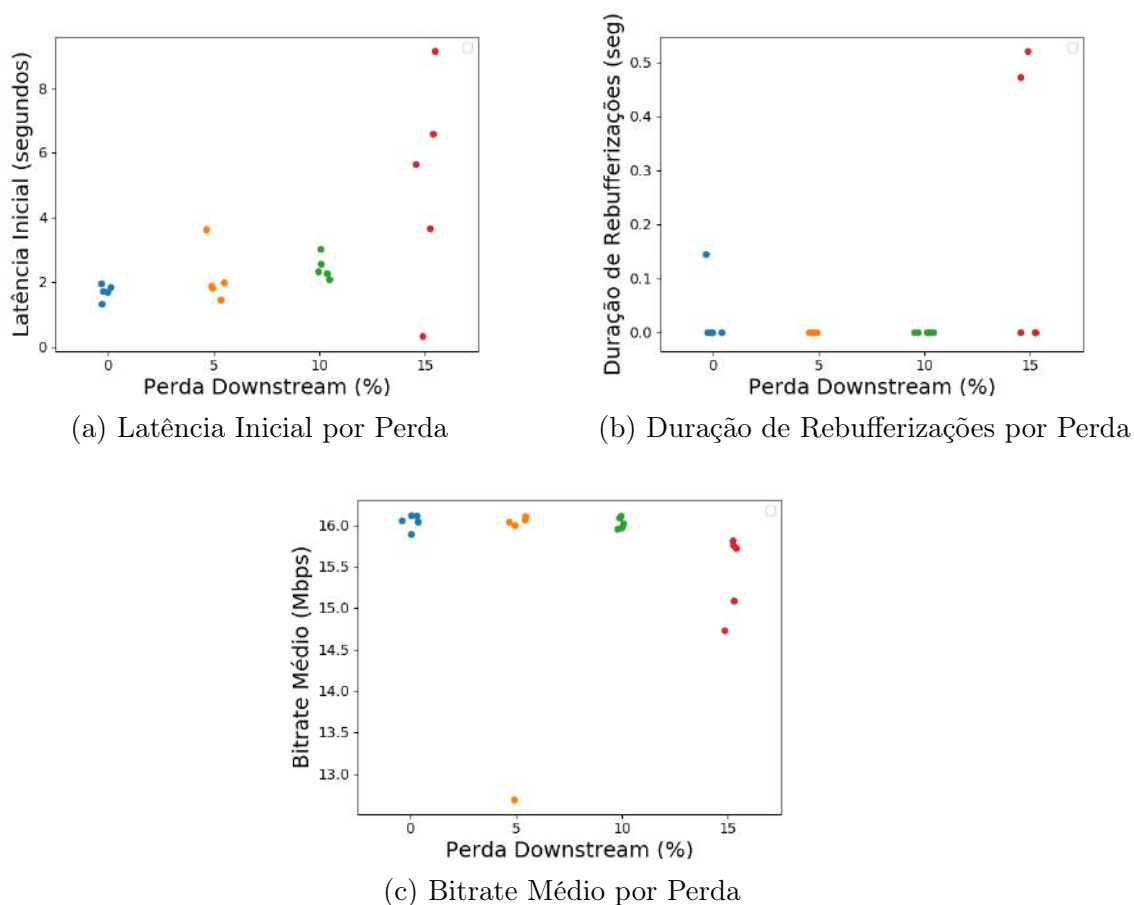


Figura 9 – Métricas de QoE por Perda para o Trailer Longo do Aquaman

O próximo experimento realizado foi o primeiro a fixar a resolução das reproduções. Isso foi feito usando uma extensão para o Google Chrome chamada YouTube HD + FPS (CHROME, 2020). O vídeo usado foi o mesmo que no experimento anterior. As taxas de perda simulada também foram as mesmas, e, para cada uma, o vídeo foi reproduzido uma vez para cada um dos seguintes níveis de resolução: 240p, 360p, 480p, 720p (HD), 1080p (Full HD), 1440p (2K) e 2160p (4K), totalizando 35 reproduções. Novamente, temos as métricas LI, DR e BM, representadas nas Figuras 10a, 10b e 10c, respectivamente. Para cada nível de resolução há bastante variância nos níveis de LI, o que é esperado, pois as reproduções foram feitas com taxas de perda média diferentes. Com a métrica BM, quanto maior a resolução, maior a variância, parecendo obedecer uma curva exponencial, porém, a quantidade de reproduções é muito pequena para essa afirmação ser conclusiva.

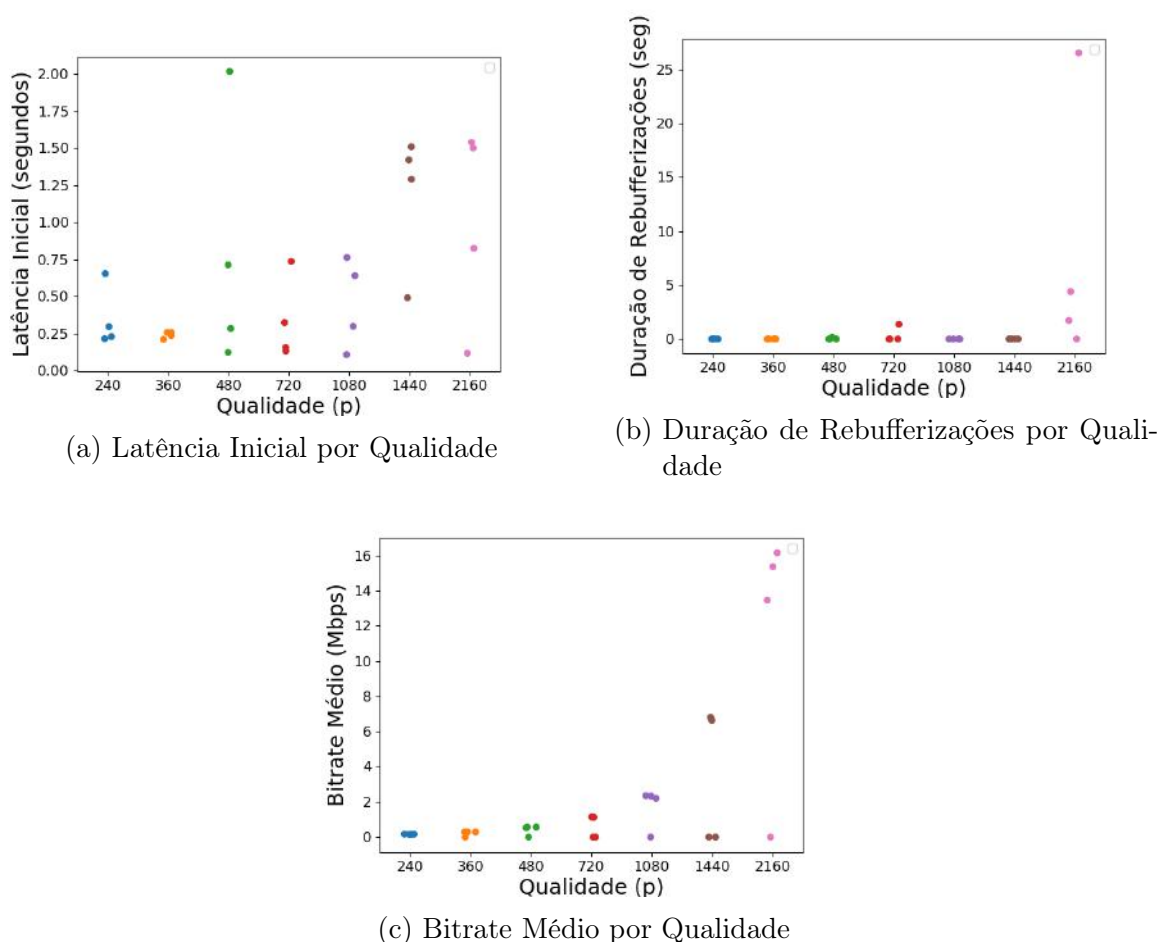


Figura 10 – Métricas de QoE por Qualidade para o Trailer Longo do Aquaman

O experimento seguinte marca a primeira vez que o tamanho do buffer downstream do WiFi foi modificado. Visto isso esse experimento será denominado de "Experimento A". Nesse experimento um outro vídeo foi escolhido, outro trailer do filme Aquaman, dessa vez com duração de 2m35s. Nenhuma perda foi simulada e todas as reproduções foram feitas em *auto*. A única variação que foi feita foi no tamanho do buffer WiFi, que

foi configurado para os tamanhos 10, 20, 30 e 256 (padrão do driver WiFi mt76) pacotes. Para cada uma dessas configurações houve 10 reproduções, totalizando 40 reproduções. Dessa vez as métricas QMR e RM também foram obtidas a partir dos arquivos HAR, além das mesmas que foram obtidas nos experimentos anteriores a partir dos objetos de vídeo Javascript. A métrica BM está representada na Figura 11a e RM na Figura 11b. A primeira coisa a perceber é que as duas mostram basicamente o mesmo comportamento, quanto maior o tamanho do buffer WiFi, maior a resolução do vídeo, e há uma diferença bem significativa entre os tamanhos de buffer 20 e 30. Não parece haver muita diferença entre os tamanhos de buffer 10 e 20 e, similarmente, entre os tamanhos de buffer 30 e 256. Isso pode ser significativo, pois é necessário muito mais memória para manter um buffer maior. Pelo outro lado, quanto menor o buffer maior será a carga na CPU e pode haver maior perda de pacotes. Encontrar o ponto de equilíbrio entre uso de memória e CPU, levando em conta a QoE é um problema de interesse para fabricantes de peças de roteadores e desenvolvedores de software para roteadores, além de provedores de conteúdos e ISPs, que usam esses dispositivos em suas infraestruturas. Continuando a análise das métricas BM e RM, as reproduções que possuem maior variância de BM (tamanho de buffer maior) são as que possuem menor variância de RM e vice versa. Com isso, é fácil concluir que os níveis de resolução mais altos possuem maior variância de bitrate que os níveis de resolução mais baixo. Pela alta correlação das métricas, somente a métrica BM será usada em análises futuras.

A Figura 11c retrata a métrica QMR para diferentes tamanhos de buffer. Somente olhando parece não haver correlação entre essa métrica e o tamanho do buffer WiFi. A Figura 11d retrata a métrica DR, e também não parece haver correlação entre essa métrica e o tamanho do buffer WiFi, somente por inspeção visual. A Figura 11e mostra a métrica LI e é fácil ver que, para esses casos, a média de LI entre as diferentes reproduções diminui conforme o tamanho do buffer aumenta, o que faz sentido intuitivo, pois para um tamanho muito pequeno de buffer, segmentos grandes ocasionarão perda de pacotes, resultando em retransmissões, o que levará a um atraso maior para o vídeo começar a reproduzir. Além disso, também percebe-se que a variância de LI também diminui quanto maior o tamanho do buffer. O motivo disso é mais difícil de interpretar, pois pode ter relação com o algoritmo de adaptação de taxa do youtube, que não é divulgado.

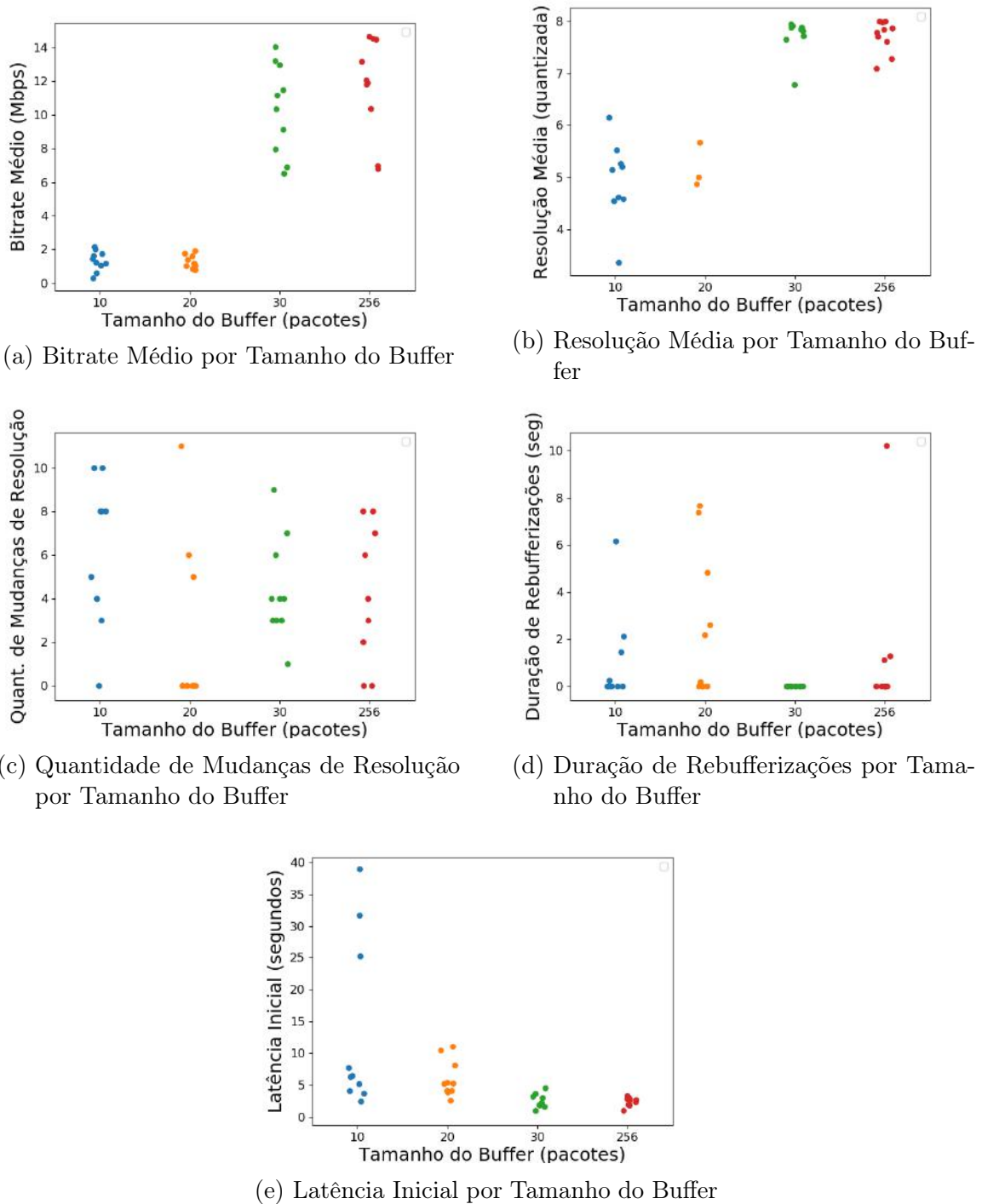


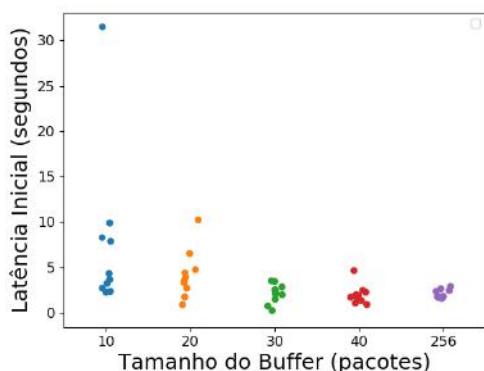
Figura 11 – Métricas de QoE por Tamanho do Buffer para o Trailer Curto do Aquaman (Experimento A)

Poucos dias depois do experimento A eu realizei mais um experimento final, com o mesmo vídeo, porém dessa vez com tamanhos de buffer WiFi de 10, 20, 30, 40, e 256 pacotes. Deixando a resolução em *auto*, foram feitas 10 reproduções para cada uma dessas configurações, totalizando 50 reproduções. Esse experimento será denominado "Experimento B". As métricas de QoE que foram coletadas no experimento A também

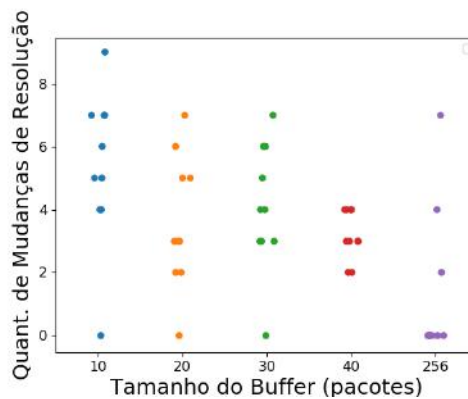
foram coletadas nesse experimento. A Figura 12a mostra a LI para esse experimento, que possui um padrão muito similar com a LI que foi observada no experimento A. Pela Figura 12b podemos ver que a QMR para esse experimento também tem um padrão muito similar com o que foi observado no experimento anterior, entretanto, dessa vez é possível ver mais facilmente que há uma correlação negativa entre QMR e o tamanho do buffer. Já a DR, que está retratada na Figura 12c, tem um padrão diferente. No experimento anterior, das 40 reproduções, 13 tiveram pelo menos um evento de rebufferização. Já nesse experimento, das 50 reproduções, somente duas tiveram pelo menos um evento de rebufferização, uma mudança bem drástica. A Figura 12d mostra o BM para esse experimento, que também está bem diferente do experimento anterior. Para tamanho de buffer 10, está bem parecido, porém, para tamanho de buffer 30, fazendo a média entre as reproduções, está menor do que do experimento A, e para os tamanhos de buffer 20 e 256, está, fazendo a média entre as reproduções, maior do que do experimento A. Para o tamanho de buffer 20 a média de BM entre as reproduções está por volta de 3 vezes o tamanho do experimento anterior, o que é significativamente diferente. A variância de BM para os tamanhos de buffer permanece similar, exceto para o tamanho de buffer 256, que diminuiu.

Com esses dois últimos experimentos é possível ver que, mesmo fixando o vídeo, horário, computador para reprodução, perda de pacotes e rede WiFi, os resultados ainda podem mudar de um dia para o outro, demonstrando empiricamente a inconsistência intrínseca do WiFi sob condições do mundo real.

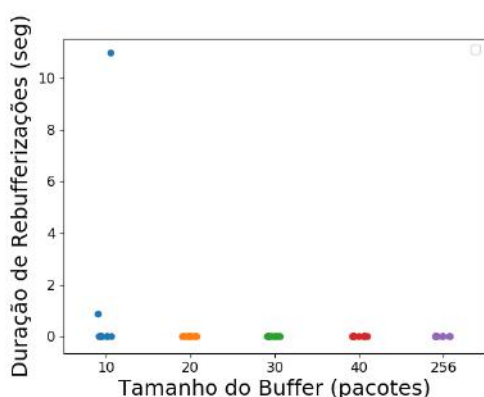
Agora que vimos como cada uma das métricas de QoE é afetada pelo tamanho de buffer do WiFi seria interessante entender como elas interagem entre si. As métricas de maior interesse são LI, BM, QMR e DR. Vamos começar visualizando somente uma delas, BM, em um eixo, como está na Figura 13a. Essas são medições de BM obtidas nas reproduções para tamanho de buffer WiFi de 256 pacotes no experimento A. Podemos também adicionar um eixo para ver como BM interage com LI, ambas as métricas providas das mesmas reproduções. Essa relação está representada na Figura 13b. Podemos ver que algumas reproduções que antes estavam próximas uma da outra, levando em conta somente a métrica BM, agora estão bem mais distantes, após termos adicionado uma métrica a mais. Podemos adicionar mais uma métrica, DR por exemplo, agora resultando na Figura 13c.



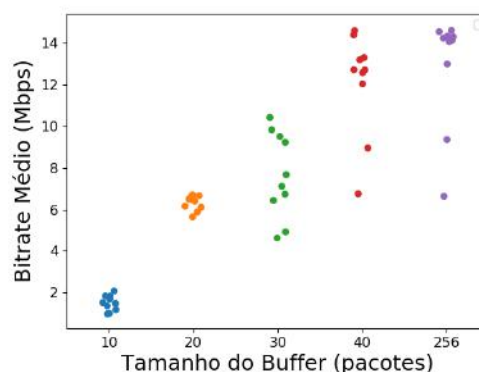
(a) Latência Inicial por Tamanho do Buffer



(b) Quantidade de Mudanças de Resolução por Tamanho do Buffer



(c) Duração de Rebufferizações por Tamanho do Buffer



(d) Bitrate Médio por Tamanho do Buffer

Figura 12 – Métricas de QoE por Tamanho do Buffer para o Trailer Curto do Aquaman (Experimento B)

Esse método de visualização, entretanto, chegou no seu limite. Se quisermos visualizar a interação entre as quatro métricas de QoE vamos precisar de uma ferramenta mais sofisticada. A ferramenta que eu usei para isso foi PCA (*Principal Component Analysis*). O PCA pode ser visto tanto como um método de extração de variáveis quanto como um método de redução de dimensionalidade. A ideia é projetar os dados numa nova base, tal que, ao se diminuir a dimensionalidade, a base preserva ao máximo as relações entre os pontos. Isso faz com que o PCA seja um ótimo método para visualizar correlação entre as variáveis.

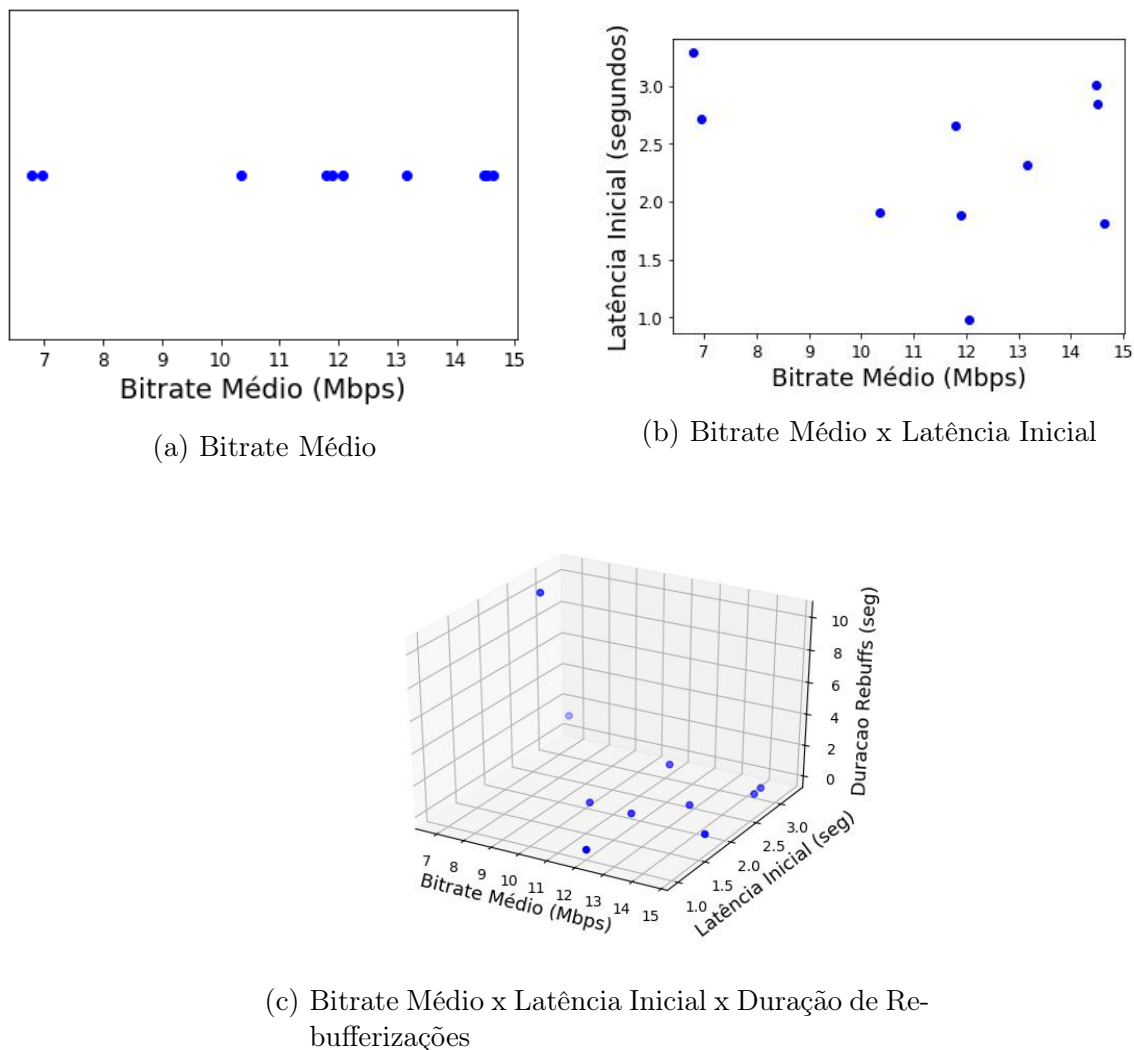


Figura 13 – Relação entre Métricas de QoE para Tamanho de Buffer 256 (Experimento A)

5.2 ENTENDENDO E APLICANDO O PCA

Para entender o PCA é informativo visualizar os dados em formato de tabela. A Tabela 9 mostra os valores das métricas de QoE das 10 reproduções que foram feitas no experimento A, com tamanho de buffer de 256 pacotes.

Antes de continuar, é necessário centralizar as variáveis no zero, o que significa tornar a média de cada variável igual a zero. Também é necessário se atentar ao fato de que todas as variáveis estão em escalas diferentes, então, além de centralizar no zero é também necessário fazer uma normalização. Para normalizar, cada termo de cada coluna, após ser centralizado no zero, será dividido pelo desvio padrão daquela coluna. Seja X a conversão dos valores da tabela para formato de matriz, o processo de centralização no

Tabela 9 – Valor das Métricas de QoE com Buffer de Tamanho 256 (Experimento A)

LI (seg)	BM (Mbps)	DR (seg)	QMR
3.292	6.783202639641763	1.114	8
2.318	13.159366338626	0	3
2.655	11.797391200790663	1.276	4
1.906	10.339545188030733	0	7
2.846	14.516733283114716	0	0
3.004	14.471148054140267	0	0
2.72	6.954072202080799	10.207	8
0.977	12.06681842101918	0.0	6
1.809	14.639035865627886	0	2
1.88	11.89801087491504	0	4

zero e normalização pode ser resumido pela seguinte equação:

$$x'_{ij} = \frac{(x_{ij} - \bar{x}_j)}{s_j}, \quad i = 1, 2, \dots, n, \quad j = 1, 2, \dots, d, \quad (5.1)$$

onde n é a quantidade de amostras, nesse caso igual a 10 e d é a quantidade de atributos, nesse caso igual a 4, x_{ij} é o i -ésimo termo da coluna j da matriz X , \bar{x}_j é a média amostral entre os termos da coluna j da matriz X , definida como

$$\bar{x}_j = \frac{1}{n} \sum_{i=1}^n x_{ij} \quad (5.2)$$

e s_j é o desvio padrão amostral entre os termos da coluna j da matriz X , definido como

$$s_j = \sqrt{\frac{\sum_{i=1}^n (x_{ij} - \bar{x}_j)^2}{n - 1}} \quad (5.3)$$

Feito isso, obtém-se a nova matriz X , representada na Tabela 10:

De acordo com (BRO; SMILDE, 2003), centralização é mais importante que a normalização. Um outro tipo de centralização que pode ser feita é pela mediana, ou seja, ajustar os dados para que cada coluna tenha mediana igual a zero o que pode tornar o modelo mais robusto. Em trabalhos futuros isso será testado. Um outro tipo de normalização que pode ser feita é dividir cada coluna pelo MAD (*Median Absolute Deviation*), porém, como dito em (BRO; SMILDE, 2003), a normalização não tem tanto efeito nos resultados, logo, a normalização pelo desvio padrão será mantida.

Agora é possível calcular a matriz de covariância amostral Σ , que indica a correlação entre as diferentes métricas (os termos "métricas", "atributos" e "variáveis" serão usados

Tabela 10 – Valor das Métricas de QoE com Buffer de Tamanho 256 após Centralização no Zero e Normalização (Experimento A)

LI (seg)	BM (Mbps)	DR (seg)	QMR
1.36255375	-1.68912702	-0.04577438	1.26200121
-0.03251337	0.51817417	-0.39575833	-0.3985267
0.45017412	0.04668544	0.00512095	-0.06642112
-0.6226239	-0.45799189	-0.39575833	0.92989563
0.72374478	0.98806764	-0.39575833	-1.39484344
0.9500493	0.97228694	-0.39575833	-1.39484344
0.54327408	-1.62997538	2.81096174	1.26200121
-1.9532372	0.13995579	-0.39575833	0.59779005
-0.76155769	1.03040636	-0.39575833	-0.73063228
-0.65986388	0.08151797	-0.39575833	-0.06642112

intercambiavelmente). Para calcular Σ é necessário calcular a covariância amostral entre cada uma das métricas, que é dada pela seguinte fórmula:

$$cov(a, b) = \frac{1}{n-1} \sum_{i=1}^n x_{ia}x_{ib}, \quad \forall a \neq b. \quad (5.4)$$

Também é necessário calcular a variância amostral de cada uma das métricas, que é dada pela seguinte fórmula:

$$var(a) = \frac{1}{n-1} \sum_{i=1}^n (x_{ia} - \bar{x}_a)^2, \quad (5.5)$$

porém, como os dados já foram centralizados em zero, $\bar{x}_a = 0$, $a = 1, 2, \dots, d$, logo,

$$var(a) = \frac{1}{n-1} \sum_{i=1}^n x_{ia}^2. \quad (5.6)$$

Em notação matricial isso é o equivalente de realizar a seguinte operação:

$$\Sigma = \frac{X^T X}{n-1} \quad (5.7)$$

Obtém-se, então, a matriz de covariância amostral abaixo:

$$\Sigma = \begin{bmatrix} 1 & -0.26341845 & 0.26660726 & -0.12142453 \\ -0.26341845 & 1 & -0.64436967 & -0.91479882 \\ 0.26660726 & -0.64436967 & 1 & 0.49577088 \\ -0.12142453 & -0.91479882 & 0.49577088 & 1 \end{bmatrix} \quad (5.8)$$

De fato, como a variância amostral de cada métrica é igual a um, o desvio padrão também é, logo, a matriz de covariância amostral Σ é exatamente a matriz de correlação. Uma forma de interpretar isso é que os valores da matriz de correlação indicam o quanto cada métrica está relacionada com as outras, com valores no intervalo $[-1, 1]$, sendo -1 indicativo de uma correlação negativa "perfeita" e 1 sendo uma correlação positiva "perfeita".

Logo, com a diagonal de 1s, todas as métricas estão perfeitamente correlacionadas com si mesmas. A partir de agora variância amostral e covariância amostral serão chamados de variância e covariância, por praticidade. Todos os resultados que usam a definição de variância e covariância, daqui em diante, podem ser trivialmente estendidos para variância amostral e covariância amostral. Como o processo de obtenção de Σ foi descrito aqui supõe-se que o leitor não ficará confuso com esse abuso de nomenclatura.

A matriz de correlação, apesar de já apresentar informações muito importantes, não pode ser aproveitada, diretamente, para visualizar a correlação entre as métricas. Ela é, entretanto, um passo intermediário para se aplicar o método PCA, que permitirá uma visualização, e, logo, um entendimento mais tácito dessas correlações. O próximo passo é encontrar os autovetores da matriz de correlação. Esses autovetores servirão como uma nova base para os pontos. Primeiro, serão encontrados os autovalores, analiticamente, resolvendo a seguinte equação:

$$\det(\Sigma - \lambda I) = 0, \quad (5.9)$$

onde λ é o vetor de autovalores de Σ e I é a matriz identidade com as mesmas dimensões que Σ . Após os autovalores serem encontrados é possível encontrar os autovetores correspondentes, resolvendo o seguinte sistema linear:

$$\Sigma v_i = \lambda_i v_i, \quad (5.10)$$

onde v_i é o autovetor que corresponde ao i -ésimo autovalor λ_i . Como se trata de uma matriz de correlação, o autovetor com maior autovalor correspondente é o eixo que, quando os dados são projetados em cima, mantém a maior variância dos dados; o autovetor com segundo maior autovalor correspondente é o eixo que é ortogonal ao primeiro autovetor e que, quando os pontos são projetados em cima, mantém a maior variância dos dados, o autovetor com terceiro maior autovalor correspondente é o eixo que é ortogonal tanto ao primeiro quanto o segundo autovetor e que, quando os pontos são projetados em cima, mantém a maior variância dos dados, e assim em diante. Além disso, os autovalores correspondem à variância explicada dos dados quando esses são projetados no eixo do autovetor correspondente. As provas de ambas essas afirmações podem ser encontradas nas seções B e C do Apêndice, respectivamente. Um termo mais intuitivo, que será usado desse ponto em diante, é que cada um desses autovalores "explica" uma proporção da variância dos dados. O ato de maximizar a variância explicada, intuitivamente, corresponde a representar bem os dados.

É possível, então, escolher os autovetores que serão utilizados para serem os eixos das novas *features* a partir da proporção da variância explicada pelos seus autovalores correspondentes. Seja m a nova dimensionalidade dos dados, com $m < d$. Seja $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_d$. Temos que

$$\frac{\sum_{i=1}^m \lambda_i}{\sum_{i=1}^d \lambda_i} \leq 1. \quad (5.11)$$

Pode-se então escolher um valor de m tal que, por exemplo, a proporção de variância explicada seja maior que 90%. Uma outra forma de escolher o valor de m é analisando os scree plots referentes aos autovalores. Scree plots mostram a proporção de variância explicada por cada autovalor, usado especificamente para aplicações do PCA. A Figura 14 mostra o scree plot correspondente às reproduções de tamanho de buffer de 256 pacotes do experimento A. Pela figura é possível ver que λ_4 explica muito pouco da variância dos dados. De fato, somente com duas dimensões já é possível explicar uma grande proporção da variância. O valor de condicionamento indicado na Figura 14 é a razão do maior autovalor sobre o menor. Os scree plots correspondentes às reproduções com outros tamanhos de buffer do experimento A e às reproduções com tamanhos diferentes de buffer do experimento B estão representados nas Figuras 15a - 15h.

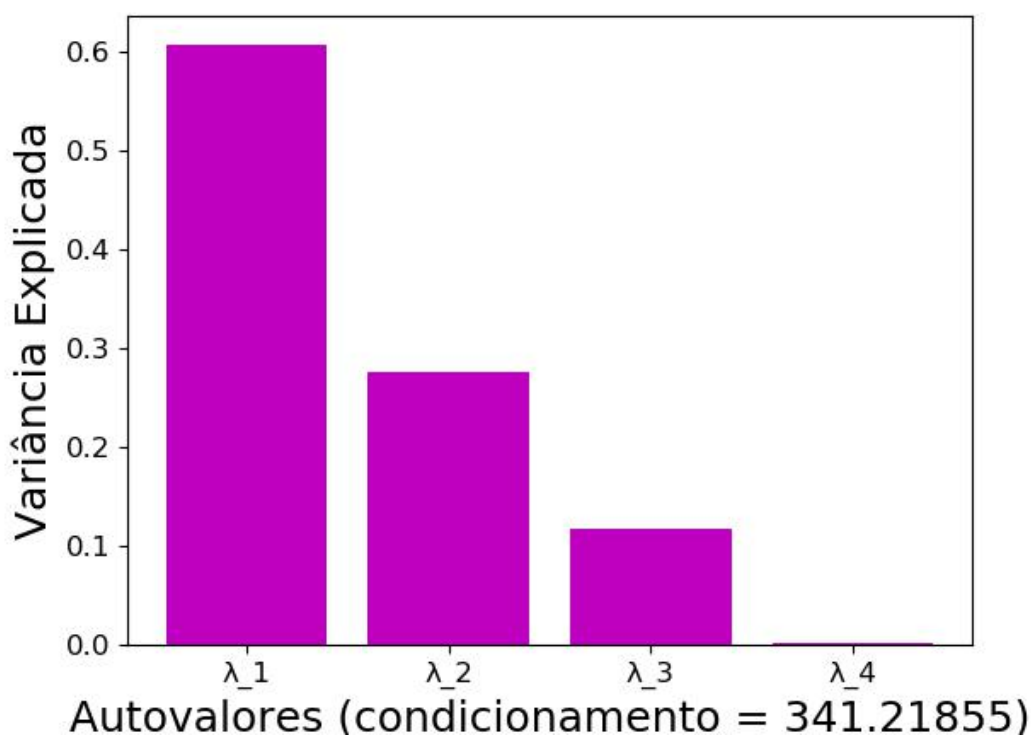
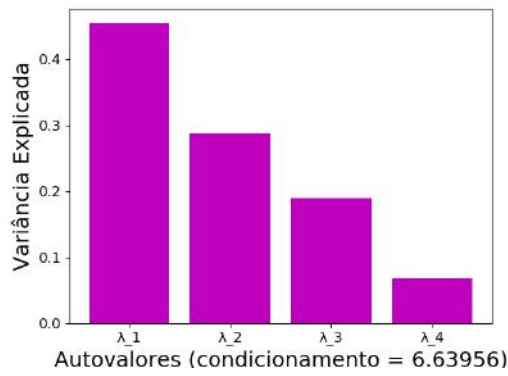
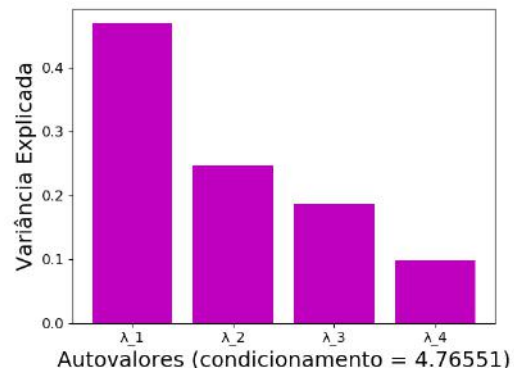


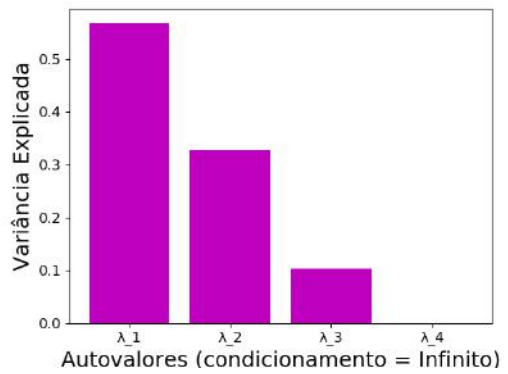
Figura 14 – Scree plot para Tamanho de Buffer 256 (Experimento A)



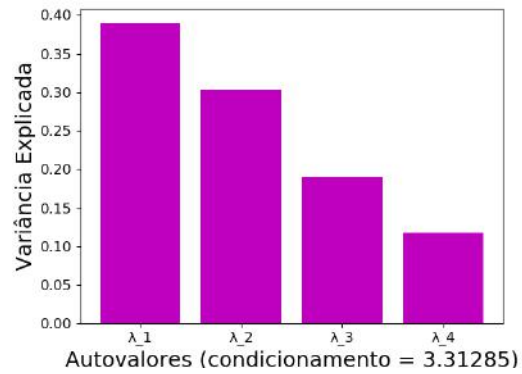
(a) Tamanho de Buffer 10 (Experimento A)



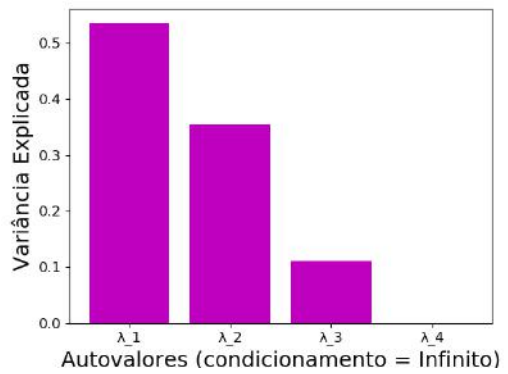
(b) Tamanho de Buffer 20 (Experimento A)



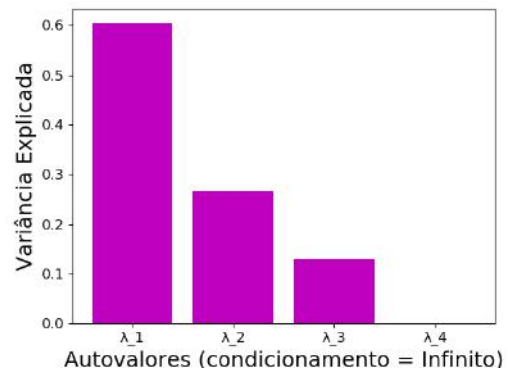
(c) Tamanho de Buffer 30 (Experimento A)



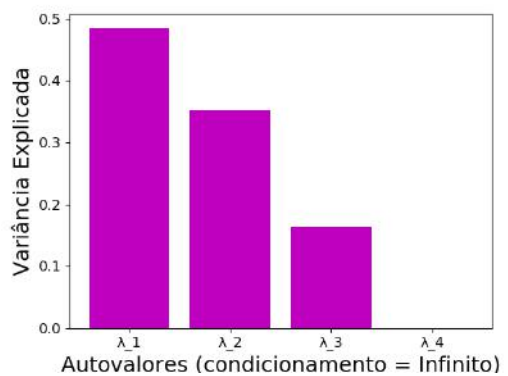
(d) Tamanho de Buffer 10 (Experimento B)



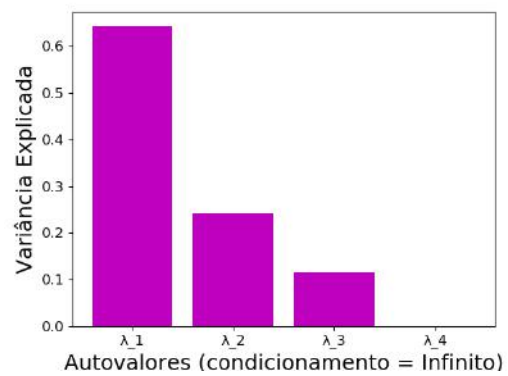
(e) Tamanho de Buffer 20 (Experimento B)



(f) Tamanho de Buffer 30 (Experimento B)



(g) Tamanho de Buffer 40 (Experimento B)



(h) Tamanho de Buffer 256 (Experimento B)

Figura 15 – Scree plot para Diferentes Tamanhos de Buffer

Os casos correspondentes às reproduções de tamanho de buffer 30 do experimento A e de tamanho de buffer 20, 30, 40 e 256 do experimento B, representados nas Figuras 15c, 15e, 15f, 15g e 15h, respectivamente, são interessantes, pois apresentam condicionamento infinito, ou seja, $\lambda_4 = 0$. Ao olhar as Figuras 11d e 12c, correspondentes à métrica DR em relação ao tamanho do buffer, para os experimentos A e B, respectivamente, percebe-se que os casos onde $\lambda_4 = 0$ correspondem aos casos onde $DR = 0$ para todos as reproduções. Em todos esses casos, a correlação entre DR e todas as outras métricas equivale a zero. Uma forma de interpretar esse fato é que conhecer o valor das outras métricas não nos permite inferir o valor de DR e vice-versa. Isso não acontece somente porque o valor de DR é zero. Contudo que todas as reproduções tenham o mesmo valor de DR, após a centralização no zero o resultado é o mesmo. Isso significa que é possível representar esses dados em três dimensões sem perder informação, ou seja, explicando toda a variância dos dados.

Após a escolha dos m autovetores que servirão como a nova base, também conhecidos como componentes principais (*Principal Components* ou PCs), é possível encontrar a projeção dos pontos de X nesses componentes da seguinte forma:

$$T = \begin{bmatrix} t_1 & t_2 & \dots & t_m \end{bmatrix} = \begin{bmatrix} Xv_1 & Xv_2 & \dots & Xv_m \end{bmatrix} \quad (5.12)$$

onde t_i , $i = 1, 2, \dots, m$ são os vetores coluna da chamada matriz de scores T , $n \times m$. A matriz de scores para $m = 2$, correspondente às reproduções de tamanho de buffer de 256 pacotes do experimento A, está representada na Figura 16. Seja V a matriz que contém os autovetores em suas colunas, a notação matricial para a obtenção da matriz de scores é

$$T = XV \quad (5.13)$$

Onde V é conhecida como a matriz de loadings.

No eixo horizontal da Figura 16 está o primeiro componente principal, ou seja, os pontos originais estão projetados no autovetor correspondente ao maior autovalor. Também é possível ver que, esse autovetor sozinho já explica 60% da variância do modelo. O eixo vertical explica 27%, então reduzindo a dimensão de quatro para dois é possível ainda explicar 87% da variância do modelo, além de permitir ver a correlação linear entre as reproduções do vídeo.

A variância total dos dados corresponde ao traço da matriz de correlação, que é a soma dos elementos na diagonal da matriz de correlação. Esse valor é igual à soma dos autovalores da matriz de correlação. Uma forma de entender intuitivamente porque a variância total é igual à soma desses autovalores é perceber que os autovetores podem ser interpretados como variáveis que não possuem correlação entre si. Para confirmar isso basta olhar a matriz de covariância amostral dos scores Σ_T , na equação 5.14, abaixo

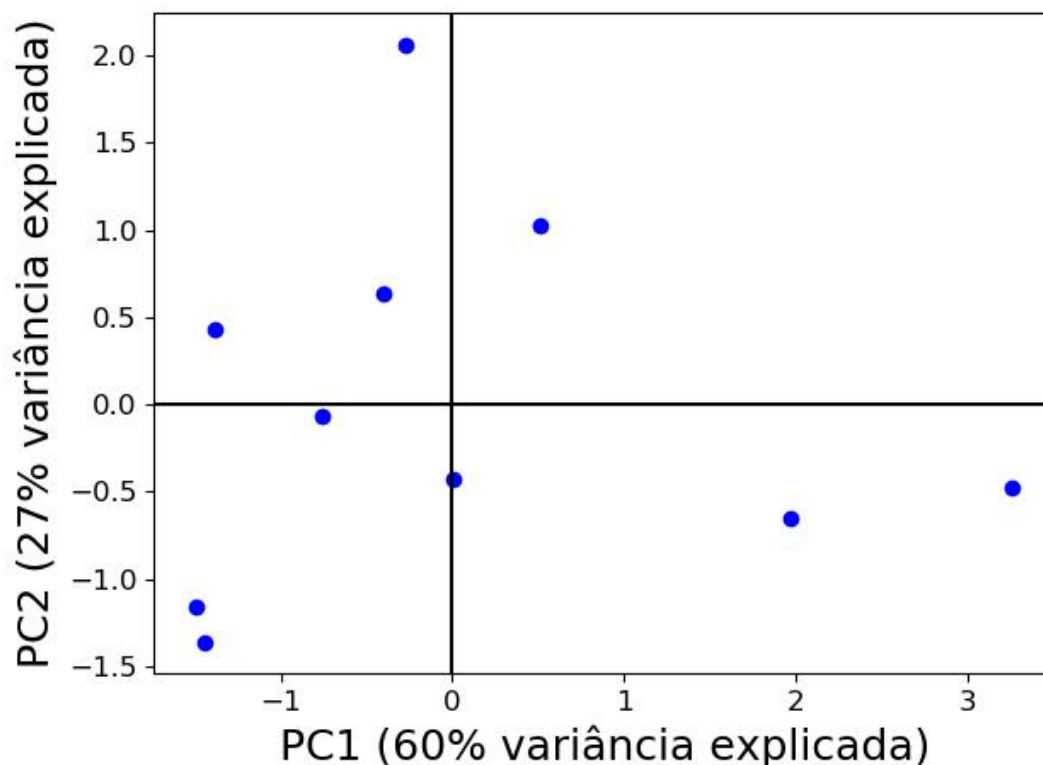


Figura 16 – Scores para Tamanho de Buffer 256 (Experimento A)

$$\Sigma_T = \begin{bmatrix} 2.42316 & 0 & 0 & 0 \\ 0 & 1.10294 & 0 & 0 \\ 0 & 0 & 0.00710 & 0 \\ 0 & 0 & 0 & 0.46680 \end{bmatrix} \quad (5.14)$$

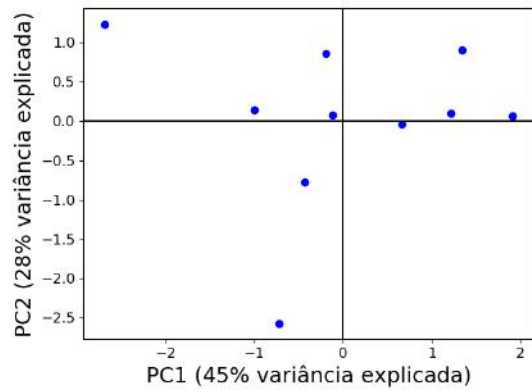
Como pode ser visto, os componentes principais só possuem correlação com si mesmos e os valores nas diagonais correspondem exatamente aos seus autovalores correspondentes. Também pode ser visto que a soma deles é igual ao traço da matriz Σ . Vale a pena notar que maximizar a variância no eixo de projeção equivale a minimizar a soma do quadrado das distâncias dos pontos até o eixo de projeção, também conhecido como erro de projeção, como está provado na seção D do Apêndice. Isso é interessante de se perceber pois reforça a ideia de que quanto mais variância o seu modelo explica, menor será o erro em relação aos dados originais.

Os scores, com $m = 2$, para todos os outros tamanhos de buffer do experimento A e para todos os tamanhos de buffer do experimento B se encontram representadas nas Figuras 17a - 17h.

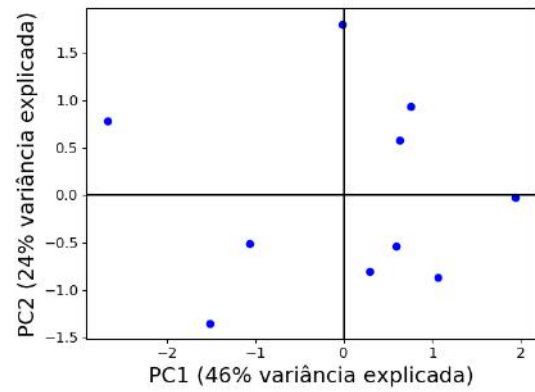
Entretanto, os scores não mostram a relação entre as métricas de QoE. A estrutura que mostra isso é a matriz de loadings, V . Uma forma de interpretar isso é entender que V está representando a contribuição de cada métrica de QoE para os valores de scores em cada reprodução. cada linha da matriz V corresponde ao loading para uma métrica,

com cada entrada dessa linha correspondendo à contribuição do respectivo componente principal no loading dessa métrica. Agora é possível visualizar como as métricas de QoE estão correlacionadas. A Figura 18 representa graficamente a matriz de loadings V .

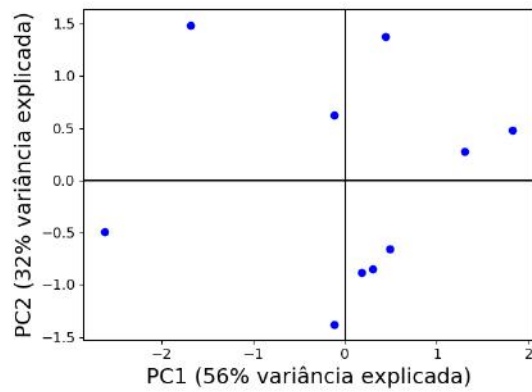
É possível ver que BM está negativamente correlacionado com QMR. BM também está muito longe de LI e DR, em relação ao primeiro componente principal, que explica a maior proporção da variância no modelo. Isso faz sentido, pois BM é a única das métricas de QoE que é positivamente correlacionada com a qualidade de experiência. Ao olhar a matriz de correlação que foi obtida anteriormente, pode-se perceber que os resultados obtidos com a matriz de loadings são coerentes com os resultados de Σ . A análise da correlação entre as métricas foi adiada para esse momento pois essas relações ficam muito mais tácitas quando visualizadas em gráficos do que quando são somente números numa matriz. A matriz de loadings, com $m = 2$, para todos os outros tamanhos de buffer do experimento A e para todos os tamanhos de buffer do experimento B se encontram representadas nas Figuras 19a - 19h.



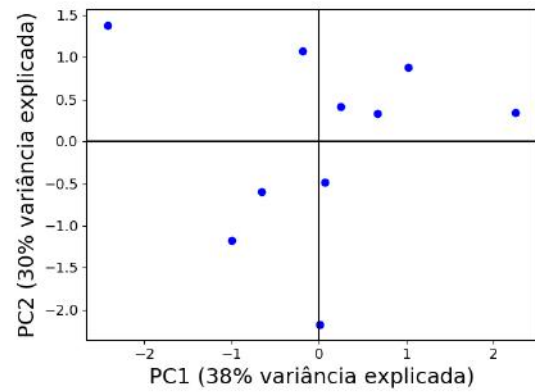
(a) Tamanho de Buffer 10 (Experimento A)



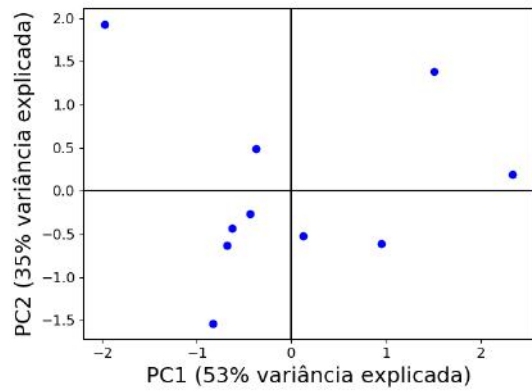
(b) Tamanho de Buffer 20 (Experimento A)



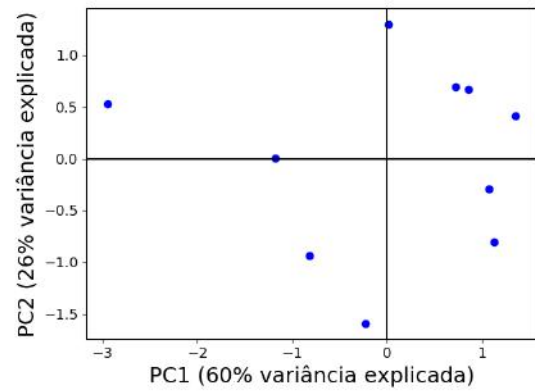
(c) Tamanho de Buffer 30 (Experimento A)



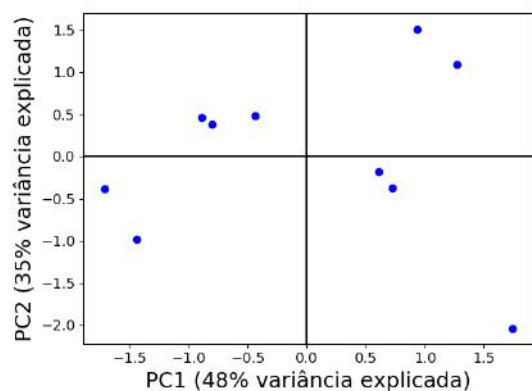
(d) Tamanho de Buffer 10 (Experimento B)



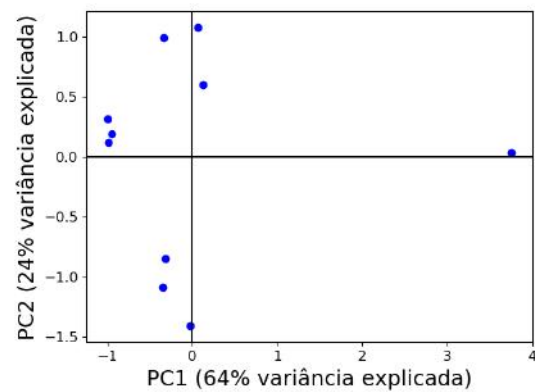
(e) Tamanho de Buffer 20 (Experimento B)



(f) Tamanho de Buffer 30 (Experimento B)



(g) Tamanho de Buffer 40 (Experimento B)



(h) Tamanho de Buffer 256 (Experimento B)

Figura 17 – Scores para Diferentes Tamanhos de Buffer

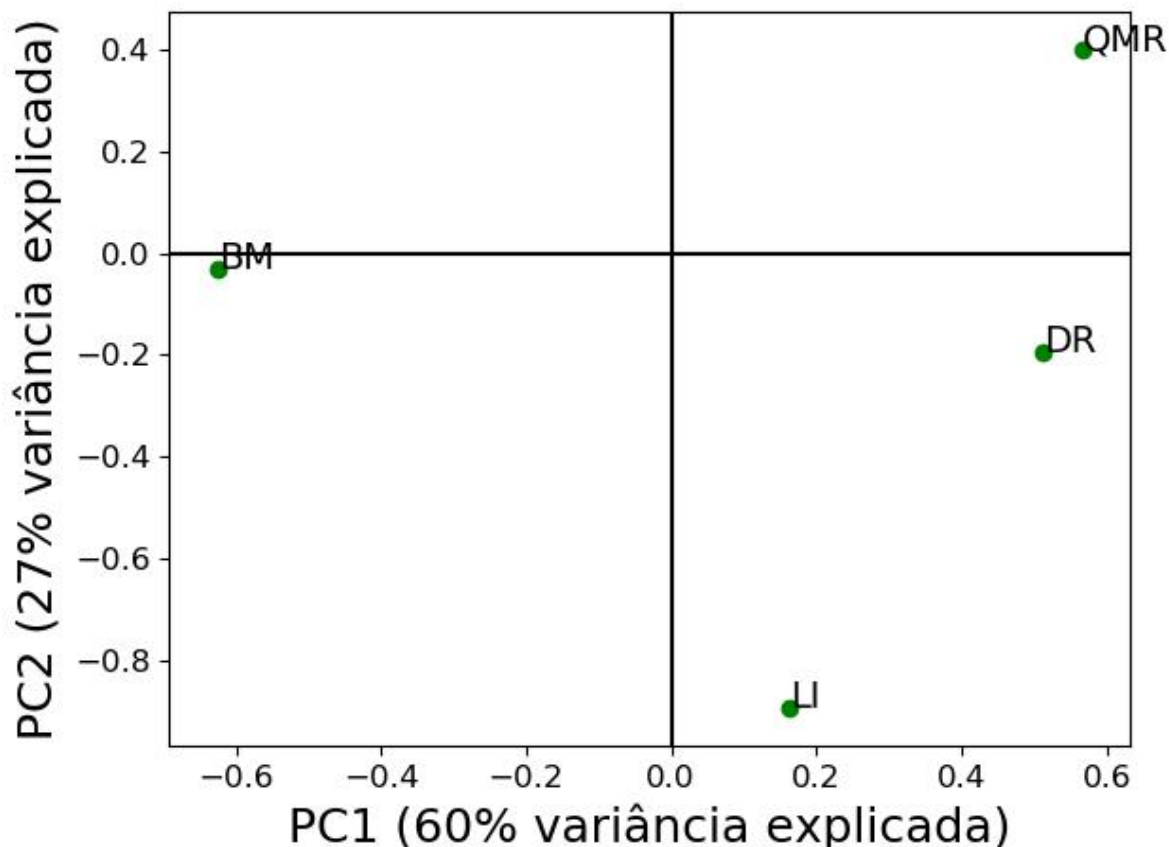
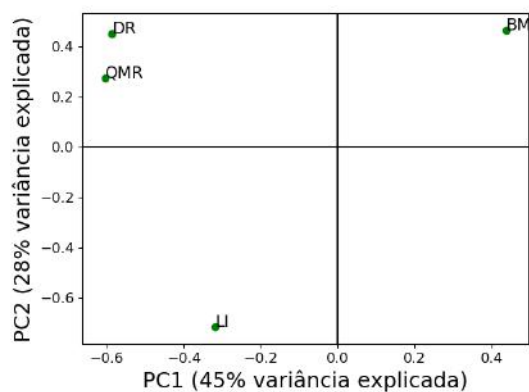
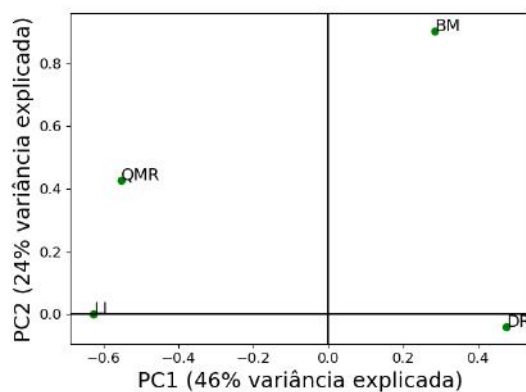


Figura 18 – Loadings para Tamanho de Buffer 256 (Experimento A)

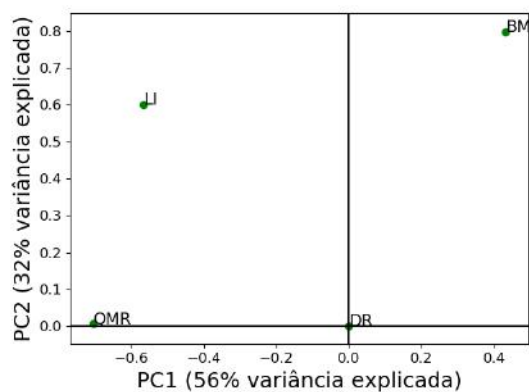
Em sete dos nove casos a métrica com a qual BM está mais negativamente correlacionada é QMR. Na Figura 19a, correspondente às reproduções com buffer de tamanho 10 do experimento A, DR tem alta correlação negativa com LI em relação ao segundo componente principal mais significativo, na Figura 19b, correspondente às reproduções com buffer de tamanho 20 do experimento A, DR tem alta correlação negativa com LI em relação ao componente principal mais significativo e na Figura 19d, correspondente às reproduções com buffer de tamanho 10 do experimento B, DR e LI estão negativamente correlacionadas em ambos os componentes principais mais significativos. Para esses tamanhos de buffer, existem reproduções onde o vídeo foi bufferizado bastante no início, o que equivale a alta LI e, por isso, não houve necessidade de rebufferização mais tarde, o que resulta em baixa DR. Também há reproduções onde houve pouca bufferização no início o que causou muita rebufferização mais tarde. Os mesmos casos em que o scree plot mostrou que $\lambda_4 = 0$, o score correspondente a DR é nulo para todos os componentes principais, o que serve como um bom *sanity check*.



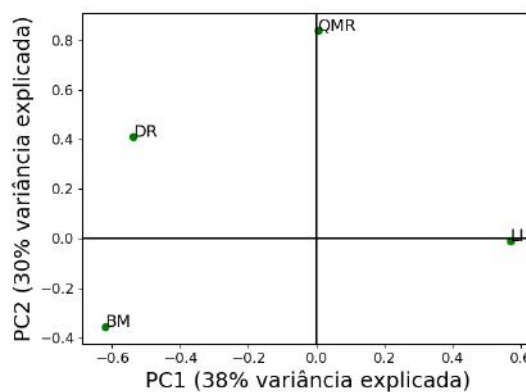
(a) Tamanho de Buffer 10 (Experimento A)



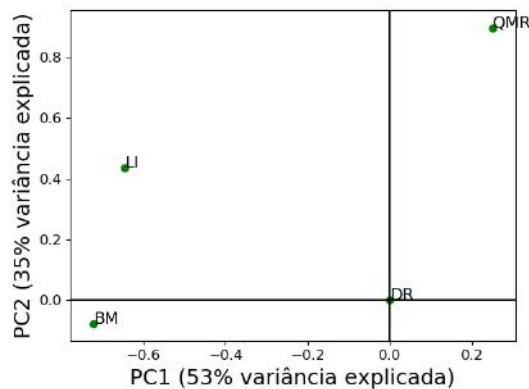
(b) Tamanho de Buffer 20 (Experimento A)



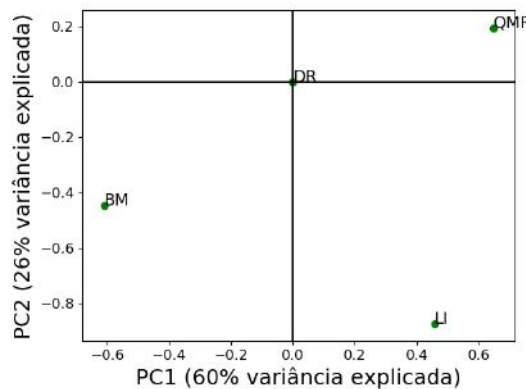
(c) Tamanho de Buffer 30 (Experimento A)



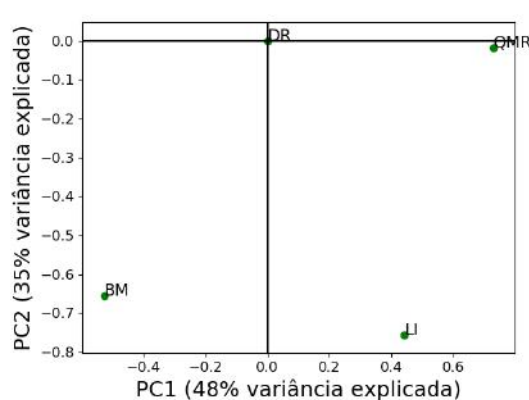
(d) Tamanho de Buffer 10 (Experimento B)



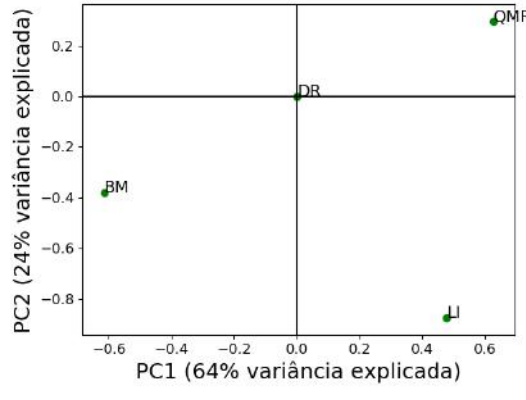
(e) Tamanho de Buffer 20 (Experimento B)



(f) Tamanho de Buffer 30 (Experimento B)



(g) Tamanho de Buffer 40 (Experimento B)



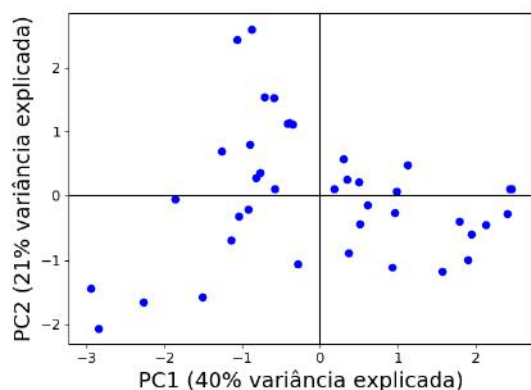
(h) Tamanho de Buffer 256 (Experimento B)

Figura 19 – Loadings para Diferentes Tamanhos de Buffer

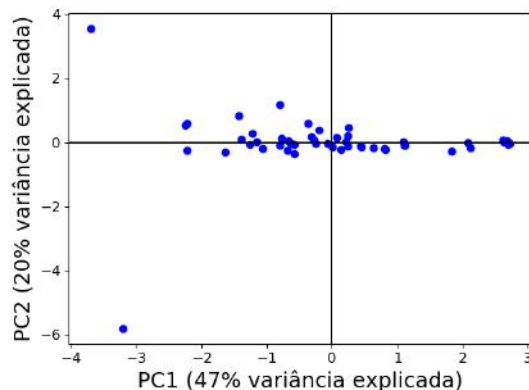
O PCA permitiu entender a correlação entre as métricas de QoE para cada um dos tamanhos de buffer porém essa ferramenta também pode ser usada para visualizar a correlação entre as métricas de QoE e o próprio tamanho do buffer WiFi (TBW). Para fazer isso primeiro é necessário montar a tabela que servirá como a matriz para o PCA. A Tabela 11 mostra o valor das métricas de QoE e TBW de cada reprodução do experimento A. Ao aplicar os mesmos passos feitos anteriormente é possível obter os scores e loadings, agora levando em conta cinco variáveis. Os scores e loadings referentes ao experimento A estão representados nas Figuras 20a e 20c, respectivamente. Olhando os loadings, percebe-se que TBW tem alta correlação positiva com BM, o que faz sentido intuitivamente, pois quanto maior o tamanho do buffer, maior esperamos que seja a resolução do vídeo, pois os segmentos de vídeo de resolução mais alta, que tem tamanho maior, podem ser salvos no buffer do roteador. O mesmo foi feito para o experimento B com scores e loadings representados nas Figuras 20b e 20d, respectivamente. Novamente, BM e TBW estão positivamente correlacionados. Inspeccionar os scores permite que reproduções outliers sejam detectadas. No caso do experimento B os outliers são bem salientes. Vale a pena ressaltar que essa detecção de outliers só está levando em conta os dois componentes principais mais significativos. Uma outra forma de detectar esses outliers será mostrada mais adiante, assim como uma forma simples de interpretar o motivo de serem outliers, uma das vantagens do PCA e um dos motivos que PCA é conhecido por ser um modelo interpretativo. As Figuras 20e e 20f mostram os scree plots do experimento A e B, respectivamente.

Tabela 11 – Valor das Métricas de QoE Agregadas

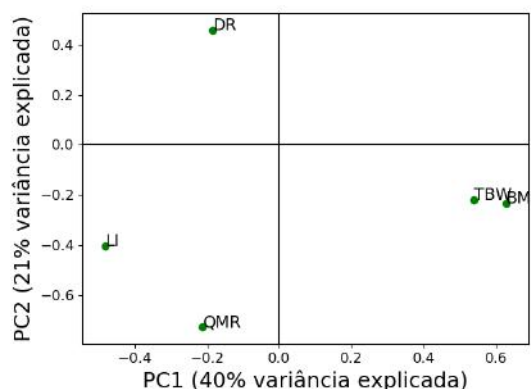
LI (seg)	BM (Mbps)	DR (seg)	QMR	TBW (pacotes)
31.6380000	1.99660946	2.11100000	10	10
3.66000000	1.42608857	1.44600000	8	10
5.15100000	1.15191145	0.24300000	4	10
2.42100000	2.13476487	0	5	10
25.2080000	1.21415351	0	8	10
7.67600000	0.579629539	6.14800000	10	10
6.27200000	1.03433027	0	8	10
38.8950000	0.277757952	0	4	10
6.43300000	1.60353040	0	3	10
4.06100000	1.72117785	0	0	10
4.09400000	1.58959218	0.18600000	0	20
3.86500000	1.73971002	0	0	20
2.57100000	1.37238203	7.65800000	0	20
11.0420000	0.760741149	0	0	20
4.08000000	1.12257558	2.16800000	0	20
5.18900000	1.00659408	7.36900000	0	20
5.36300000	1.00758225	2.59500000	0	20
10.4270000	1.15322947	0	11	20
5.24200000	0.837208971	0	6	20
8.07800000	1.89766885	4.82600000	5	20
0.972000000	10.3257558	0	6	30
1.86800000	6.87661469	0	4	30
1.59200000	12.9576063	0	3	30
3.17100000	11.1522227	0	4	30
2.18800000	7.93723633	0	3	30
2.98100000	14.0285511	0	4	30
4.51100000	11.4740865	0	7	30
1.97400000	9.11013346	0	3	30
1.83000000	13.1996818	0	1	30
3.62300000	6.49937150	0	9	30
3.29200000	6.78320264	1.11400000	8	256
2.31800000	13.1593663	0	3	256
2.65500000	11.7973912	1.27600000	4	256
1.90600000	10.3395452	0	7	256
2.84600000	14.5167333	0	0	256
3.00400000	14.4711481	0	0	256
2.72000000	6.95407220	10.2070000	8	256
0.977000000	12.0668184	0	6	256
1.80900000	14.6390359	0	2	256
1.88000000	11.8980109	0	4	256



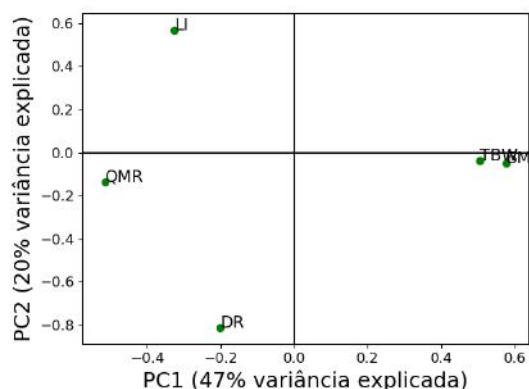
(a) Scores para o Experimento A



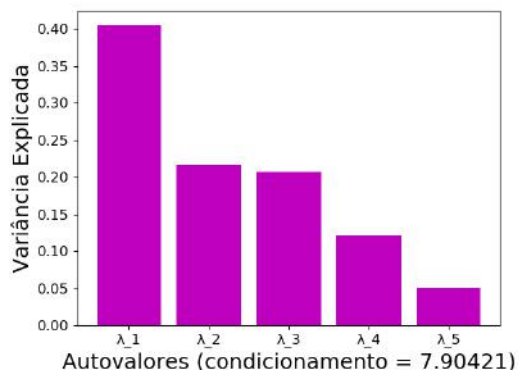
(b) Scores para o Experimento B



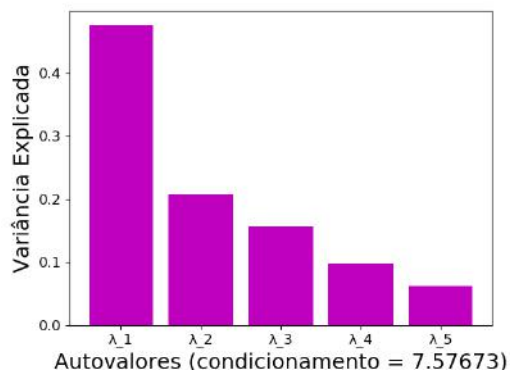
(c) Loadings para o Experimento A



(d) Loadings para o Experimento B



(e) Scree plot para o Experimento A



(f) Scree plot para o Experimento B

Figura 20 – Gráficos dos Dados Agregados

Algo a ser ressaltado é que o PCA não precisa ser aplicado em cima das métricas diretamente. Transformações podem ser aplicadas nas métricas e podem ser usadas no lugar das mesmas, ou em complemento com as métricas. De fato, algo que é recomendado no PCA é que, quando não se tem certeza quais variáveis são importantes deve-se usar todas, pois, naturalmente, as que forem menos importantes terão valor de loadings mais baixos, pelo menos para os componentes principais mais significativos. Existem algumas métricas de QoE que nem foram mencionadas, mas que podem ser derivadas a partir das que já foram obtidas ou diretamente dos arquivos HAR e dos arquivos de eventos e

propriedades de objeto vídeo javascript. Além disso, existem as que foram mencionadas e descritas, como QR, RR e RM que não foram analisadas muito a fundo. Essas variáveis adicionais e transformações das variáveis não foram usadas no PCA pois a complexidade da análise aumentaria significativamente. No futuro, experimentos serão feitos com clientes voluntários da Gigalink, onde será possível obter feedback dos mesmos. Esse tipo de feedback poderá ser usado para estabelecer um *ground truth* de QoE em aplicações de vídeo, indicado pelo MOS (*Mean Opinion Score*), como feito em (HORA et al., 2018), o que facilitará na avaliação da importância das métricas. Quando esses experimentos *on the wild* forem feitos essas métricas adicionais serão usadas e os efeitos das escolhas de diferentes transformações sobre métricas também será investigado.

Existe uma relação muito interessante entre o PCA e o SVD (*Singular Value Decomposition*) que vale a pena ser explorada. Seja X a matriz de dados após a centralização no zero e normalização. Ao aplicar o SVD à essa matriz, obtém-se a decomposição única $X = UDV^T$, onde U é uma matriz ortonormal $n \times d$, V é uma matriz ortonormal $d \times d$ e os valores singulares de X , $\sigma_1, \sigma_2, \dots, \sigma_d$ estão na matriz diagonal D , $d \times d$. Se então fizermos a operação $X^T X$, que equivale à matriz de correlação Σ , a menos de uma escala, temos que

$$X^T X = (UDV^T)^T UDV^T \quad (5.15)$$

$$= VDU^T UDV^T. \quad (5.16)$$

Como U é uma matriz ortonormal, $U^T U = I$, logo

$$X^T X = VDDV^T \quad (5.17)$$

$$= VD^2V^T. \quad (5.18)$$

Para obter exatamente a matriz de correlação, basta fazer

$$\frac{X^T X}{n-1} = V \frac{D^2}{n-1} V^T. \quad (5.19)$$

Os valores na matriz diagonal D^2 são os valores singulares ao quadrado, $\sigma_1^2, \sigma_2^2, \dots, \sigma_d^2$, que, pelo teorema espectral, equivalem aos autovalores da matriz $X^T X$, que, como visto anteriormente, equivalem à variância sobre os eixos dos autovetores correspondentes. Isso explica porque os valores singulares são simbolizados por σ , eles equivalem ao desvio padrão (comumente simbolizado na literatura como σ) sobre o eixo dos autovetores correspondentes. Pelo teorema espectral a matriz V contém os autovetores de $X^T X$ em suas colunas, ou seja, é a matriz de loadings. É possível então encontrar a matriz de scores

$$T = XV \quad (5.20)$$

$$= UDV^T V \quad (5.21)$$

$$= UD. \quad (5.22)$$

Definindo uma nova matriz auxiliar W como

$$W = X^T T \quad (5.23)$$

$$= X^T U D \quad (5.24)$$

$$= (U D V^T)^T U D \quad (5.25)$$

$$= V D U^T U D \quad (5.26)$$

$$= V D D \quad (5.27)$$

$$= V D^2. \quad (5.28)$$

Isso é o equivalente de multiplicar a matriz $X^T X$ pela matriz V , que contém os autovetores em suas colunas. Para isso se tornar saliente basta ver que

$$W = X^T T \quad (5.29)$$

$$= X^T X V. \quad (5.30)$$

Logo,

$$X^T X V = V D^2. \quad (5.31)$$

Ou seja, a matriz $X^T X$ multiplicada pela matriz que contém seus autovetores resulta na matriz que contém seus autovetores multiplicada pela matriz que contém os autovalores correspondentes, o que segue nossa intuição. O SVD permite que isso seja visualizado de forma muito clara.

No PCA, são escolhidos os m maiores autovalores e os demais são descartados. Seja V_m a matriz de loadings somente com os m autovetores correspondentes aos m maiores autovalores em suas colunas. Seja T_m a matriz de scores quando V_m é usada. Temos então que

$$T_m = X V_m \quad (5.32)$$

$$= U D V^T V_m. \quad (5.33)$$

Vamos analisar as operações matriciais presentes, começando pela direita. Sabemos que, por V ser uma matriz ortonormal, $V^T = V^{-1}$, logo, $V^T V = I$, como visto anteriormente. É fácil ver que quando a operação $V^T V_m$ é feita, o resultado é uma matriz identidade que possui somente as primeiras m colunas. Chamaremos essa matriz de I_m . Continuando, temos a operação $D I_m$. Essa operação resultará numa matriz que possui as primeiras m colunas de D . Chamaremos essa matriz de D_m . Logo, temos que

$$T_m = U D_m. \quad (5.34)$$

Entretanto, efetivamente, as únicas colunas de U que serão contabilizadas para a obtenção de T_m são as m primeiras. É possível então transformar D_m numa matriz $m \times m$, retirando

os vetores linhas que possuem somente valor zero. Seja U_m a matriz que contém as primeiras m colunas de U , temos que

$$T_m = U_m D_m. \quad (5.35)$$

É fácil ver que a matriz T_m tem posto m , que equivale à nova dimensionalidade dos dados. Seja M_m a aproximação de X usando somente os m autovetores correspondentes aos m maiores autovalores. Temos que

$$M_m = T_m V_m^T \quad (5.36)$$

$$= U_m D_m V_m^T, \quad (5.37)$$

que equivale exatamente à decomposição SVD truncada pelos primeiros m valores singulares. É fácil ver que a matriz M_m também possui posto m . Foi provado na seção C do Apêndice que os autovalores correspondem à variância explicada no espaço de projeção, ou seja, no espaço dos scores. Entretanto, somente uma transformação ortogonal que preserva o posto foi aplicada para levar ao espaço de projeção de volta para o espaço dos dados originais. Isso significa que as distâncias e ângulos entre os pontos foram preservados nessa transformação, o que significa que a variância explicada também foi preservada, pois a variância nada mais é que uma medida de divergência da média. Também foi provado na seção B que a média no espaço de projeção equivale a zero, igual à média no espaço original. A média também é preservada na transformação do espaço de projeção para o espaço do modelo, equivalendo a zero. Seja

$$e_{ij} = x_{ij} - m_{ij}, \quad i = 1, 2, \dots, n, \quad j = 1, 2, \dots, d. \quad (5.38)$$

A matriz E contém o que não pode ser explicado pelo PCA. Antes foi levantada a ideia de encontrar reproduções que são outliers fazendo inspeção visual do scatterplot dos scores. Uma outra forma de encontrar outliers é analisando a matriz residual E . Uma métrica muito usada para isso é o SPE (*Squared Prediction Error*), definida como

$$SPE_i = \sum_{j=1}^d e_{ij}^2. \quad (5.39)$$

Ou seja, o SPE é a medida de dispersão entre o modelo M e os dados reais X , por observação. Quando a matriz V não é truncada, ou seja, quando todos os autovetores são usados, o SPE é igual a zero para todas as observações, ou seja, o modelo consegue reproduzir totalmente os dados originais. Pelo SVD isso fica claro de ver, pois temos que,

$$M = TV^T \quad (5.40)$$

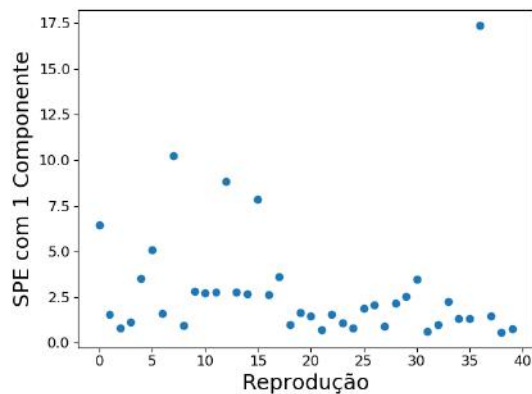
$$= XVV^T \quad (5.41)$$

$$= X, \quad (5.42)$$

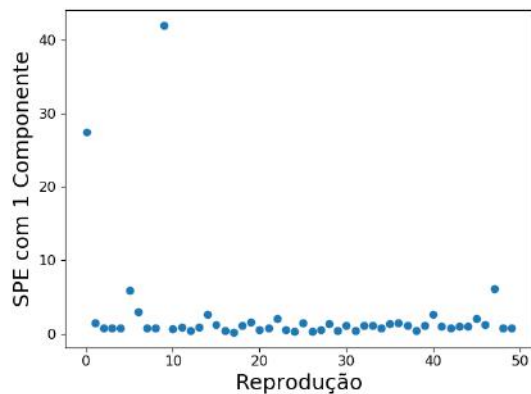
porém, uma outra forma de entender isso vem do fato da matriz V ser ortonormal. Ou seja, é uma matriz de rotação, que preserva todas as distâncias e ângulos. Quando a matriz V inteira é usada para gerar a matriz de scores T , isso não é nada mais que uma rotação ainda no mesmo espaço d -dimensional. Quando a transformação V^T é aplicada na matriz de scores, para obter a matriz M a rotação original só é invertida, voltando aos dados originais, sem perda.

Como no caso da detecção de outliers somente usando o scatterplot dos scores, a detecção de outliers usando o SPE também depende da quantidade de componentes principais que estão sendo usados para montar o modelo. Devido ao fato de SPE ser igual a zero para todas as observações quando a matriz V não é truncada, só é interessante olhar os valores de SPE quando menos componentes principais são usados. Os gráficos de SPE usando um até quatro componentes, para ambos os experimentos estão representados nas Figuras 21a - 21h. Para a análise de outliers tanto o SPE obtido a partir de um componente, quanto obtido a partir de dois componentes, serão usados, e resultam na detecção das mesmas reproduções como outliers.

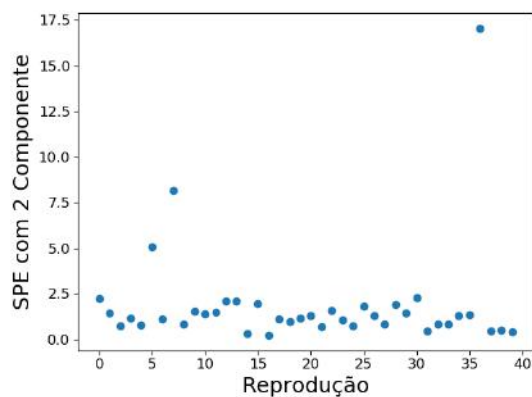
Como pode ser visto, para o experimento A foi encontrada uma reprodução outlier, a 37a. Para o gráfico de SPE com um componente, referente ao experimento B, a décima reprodução é a que tem maior valor de SPE enquanto a primeira reprodução é a que tem o segundo maior valor de SPE. Quando dois componentes são usados essa relação se inverte. Em ambos os casos, as duas reproduções continuam sendo as que têm maior valor de SPE, logo, ambas serão classificadas como outliers para o experimento B. As Figuras 22a e 22b mostram os scores, referentes aos dois componentes principais mais significativos, das reproduções agregadas dos experimentos A e B, respectivamente, como foi feito nas Figuras 20a e 20b, porém com as reproduções outliers destacadas em vermelho. No caso do experimento B os outliers eram facilmente detectados sem a necessidade da análise residual. Inclusive, as reproduções encontradas como outliers são as mesmas que foram identificadas anteriormente na análise dos scores agregados para o experimento B. Entretanto, no caso do experimento A não era possível detectar que aquela reprodução era outlier somente com a análise dos scores referentes aos dois componentes principais mais significativos.



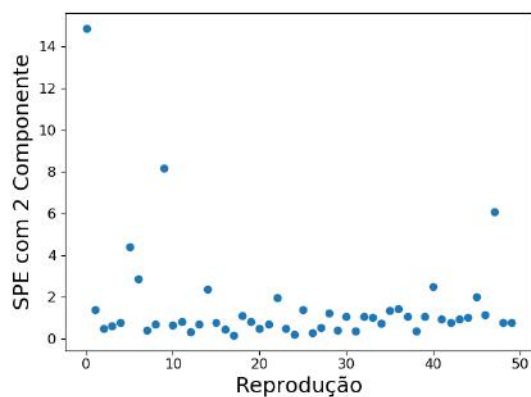
(a) SPE com Um PC (Experimento A)



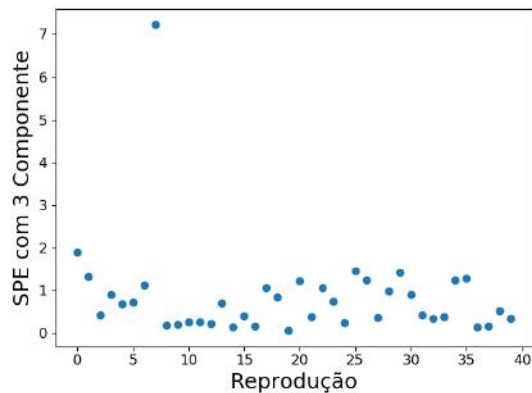
(b) SPE com Um PC (Experimento B)



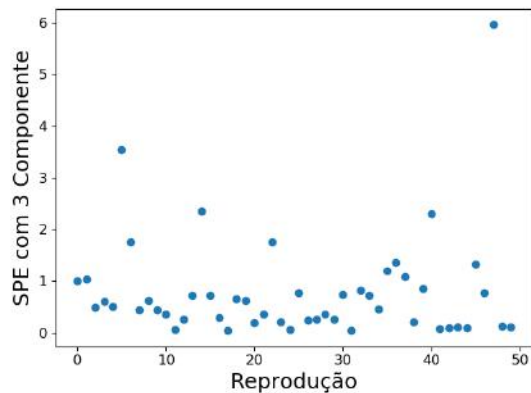
(c) SPE com Dois PCs (Experimento A)



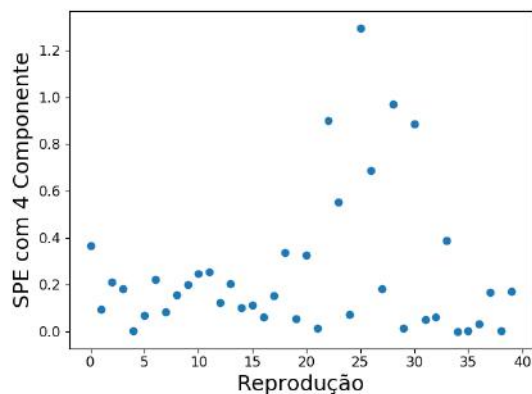
(d) SPE com Dois PCs (Experimento B)



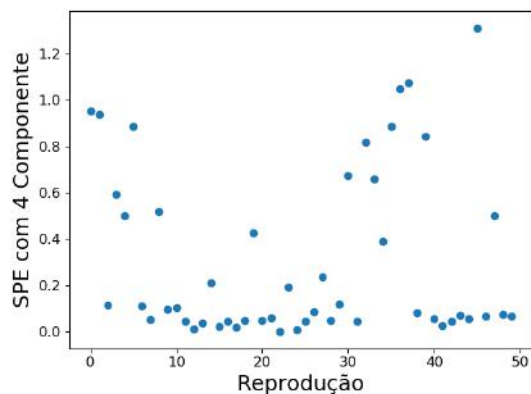
(e) SPE com Três PCs (Experimento A)



(f) SPE com Três PCs (Experimento B)



(g) SPE com Quatro PCs (Experimento A)



(h) SPE com Quatro PCs (Experimento B)

Figura 21 – SPE Com Diferentes Quantidades de PCs

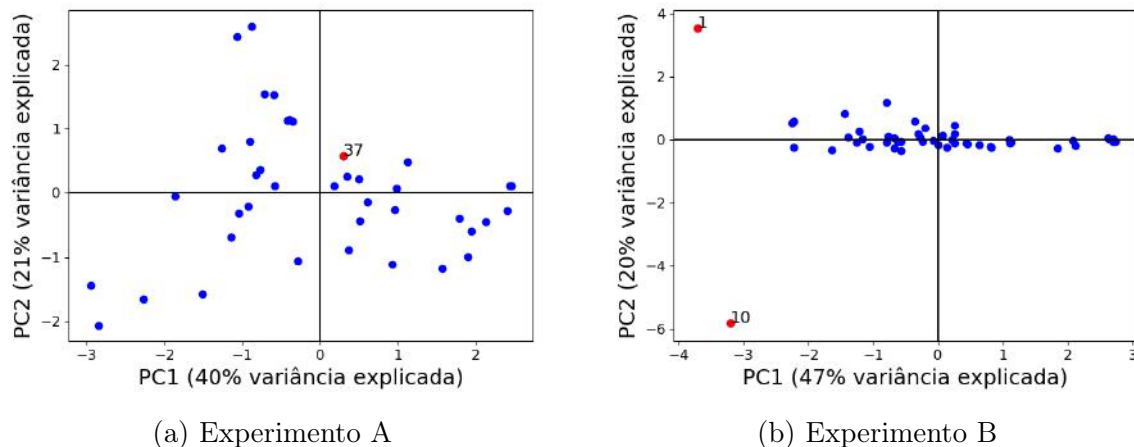


Figura 22 – Scores com Destaque nos Outliers Detectados com SPE

É possível identificar exatamente o que causou essas reproduções a serem outliers. Primeiro será analisado o outlier com número de reprodução 10, do experimento B. A Figura 24a mostra os valores das métricas de QoE para essa reprodução. É possível ver que o valor de DR é muito alto. Vale a pena lembrar que a média foi centrada em zero e o desvio padrão foi normalizado para igualar a um, logo, DR está a quase sete desvios padrões da média. Também é o caso que só duas das cinquenta reproduções no experimento B tiveram rebufferização, essa certamente é uma dessas. Ao olhar a Figura 20d, é possível ver que DR tem contribuição negativa de módulo bem alto para scores no componente principal 2 (o eixo do componente 2 está mais achatado que do componente 1), que é exatamente o componente principal em que a reprodução está mais distante das outras. Olhando para a Figura 24b, que mostra os valores de scores para a reprodução, pode-se perceber que o maior valor, em módulo, encontra-se no componente principal 2, como esperado. Além disso, a reprodução também teve valores de QMR quase dois desvios padrões acima da média e valores de BM um pouco mais de um desvio padrão abaixo da média. Levando em conta que QMR tem o menor valor de loading no componente um e BM tem o maior valor de loading no componente um, como pode ser visto na Figura 20d, isso explica porque a reprodução 10 é a que tem o segundo menor valor de score no componente um.

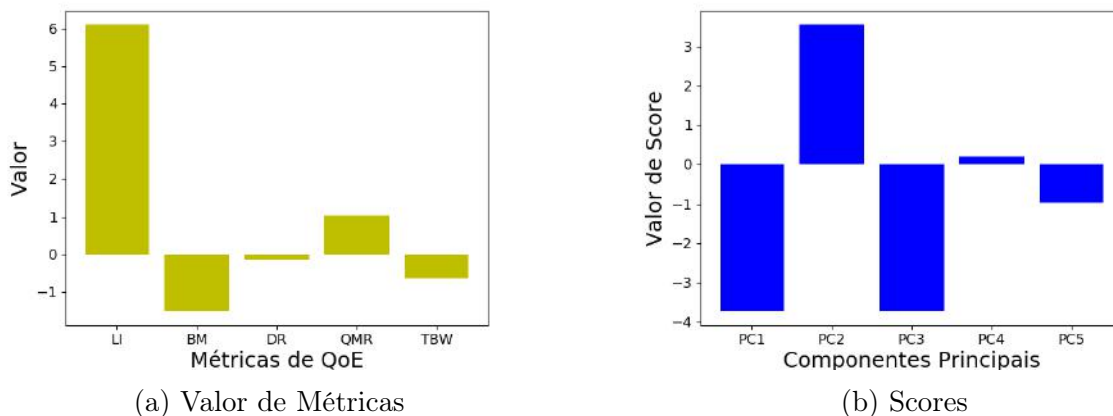


Figura 23 – Gráficos Referentes à 1a Reprodução do Experimento B

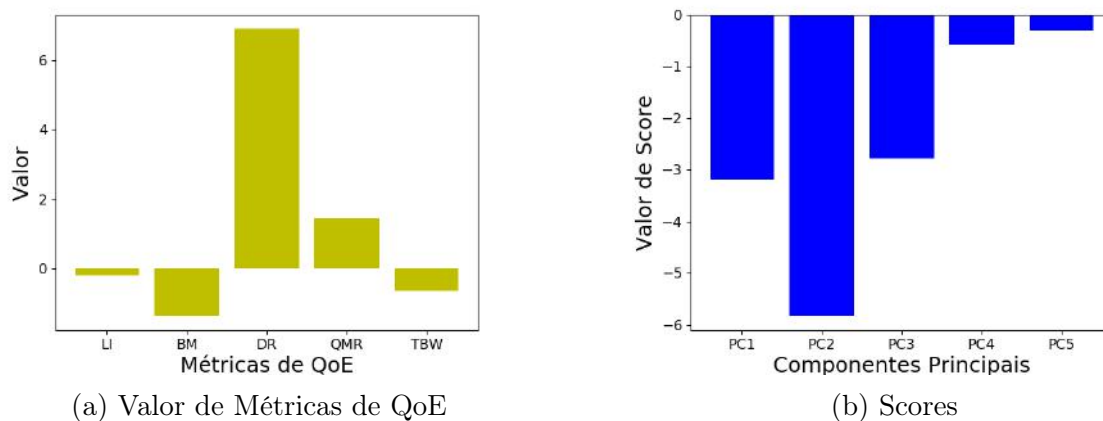


Figura 24 – Gráficos Referentes à 10a Reprodução do Experimento B

Agora será analisada a reprodução outlier do experimento A, com número de reprodução 37. Somente olhando a Figura 22a não seria óbvio classificá-la como um outlier. Entretanto, ao olhar a Figura 25b, que mostra os valores do score para essa reprodução, percebe-se que o valor referente ao componente principal 3 é significativamente mais alto que dos outros. Além disso, o valor do score referente ao componente principal 2 é o segundo maior, apesar de ser bem mais baixo que do componente 3. Esses fatores são o suficiente para motivar a inspeção dos scores e loadings nos componentes principais 2 e 3. A Figura 26a mostra os loadings para os componentes principais 2 e 3. Percebe-se que a métrica DR tem alta contribuição positiva para os scores referentes ao componente principal 3 e média contribuição positiva para os scores referentes ao componente principal 2. Como pode ser visto na Figura 25a a reprodução 37 possui valor de DR com mais de três desvios padrões acima da média, o que explica porque, quando olhamos os scores do experimento A referentes aos componentes principais 2 e 3 (Figura 26b), a reprodução 37 claramente corresponde ao caso de um outlier. Além disso, a magnitude dos valores, assim

como os sinais, nos componentes, corresponde com o valor de loadings de DR. Análise análoga pode ser feita com o outro outlier do experimento B.

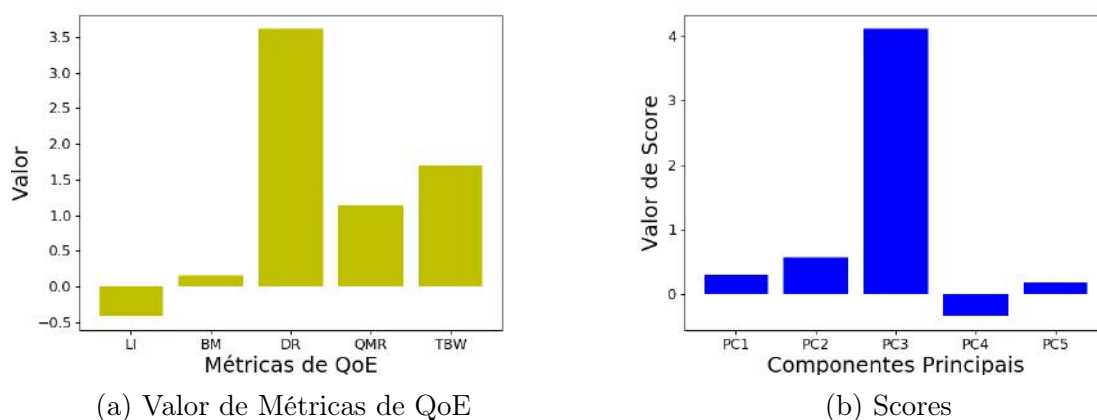


Figura 25 – Gráficos Referentes à 37a Reprodução do Experimento A

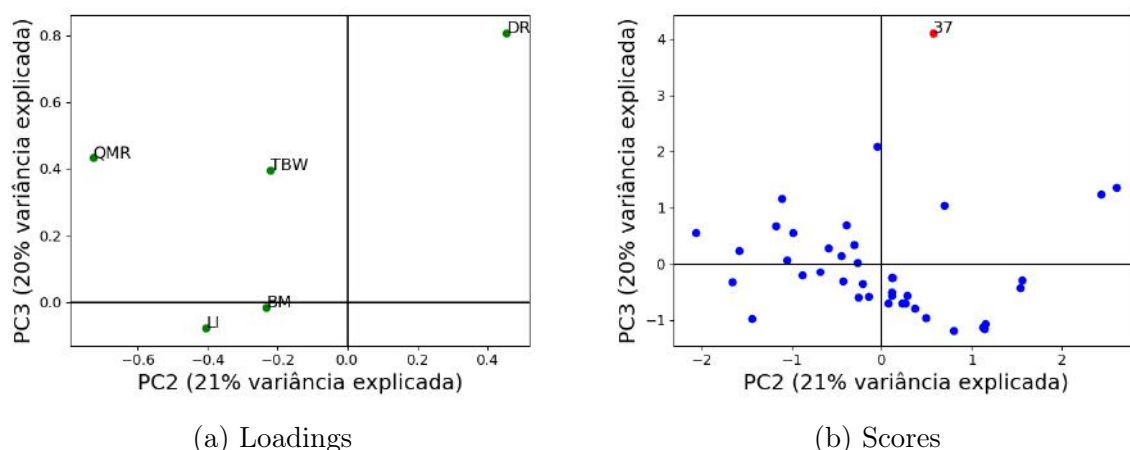


Figura 26 – Scores e Loadings Referentes ao PC2 e PC3 do Experimento A

Para as reproduções 1 e 10 do experimento B, identificadas como outliers, se as figuras referentes aos valores de LI e de DR por tamanho de buffer (Figuras 12a e 12c, respectivamente), forem analisadas, já é possível encontrar esses outliers, olhando as reproduções de tamanho de buffer de 10 pacotes. Para a reprodução 37 do experimento A, identificada como outlier, se a Figura 11d, referente aos valores de DR por tamanho de buffer for analisada, também já é possível encontrar esse outlier, olhando as reproduções de tamanho de buffer de 256 pacotes. Para esses casos, o que o PCA oferece é uma nova forma de detectar e interpretar esses outliers. Uma interpretação que o PCA dá para essas reproduções serem identificadas como outliers pela análise do SPE com um e dois componentes está relacionada com a reconstrução dos dados após a obtenção dos scores. Quando se obtém os scores só com um ou dois componentes, essencialmente fazendo uma transformação de rotação com redução de dimensionalidade, essas reproduções são mapeadas em valores

com erro muito alto referente ao original quando a transformação inversa é feita. Ou seja, o modelo com posto um e dois não consegue representar bem esses dados. Isso significa que essas reproduções possuem valores altos de scores para componentes acima de dois. As Figuras 27a e 27b mostram os desvios padrões dos scores por componentes para os experimentos A e B, que são a raiz dos autovalores correspondentes aos componentes. Já foi provado na seção B do Apêndice que a média nos scores é igual à média nos dados originais. Com essas informações pode ser visto que a reprodução 37 do experimento A possui score por volta de quatro desvios padrões acima da média, para o componente três. Similarmente, as reproduções classificadas como outliers do experimento B também possuem valores alguns desvios padrões abaixo ou acima da média para o componente três.

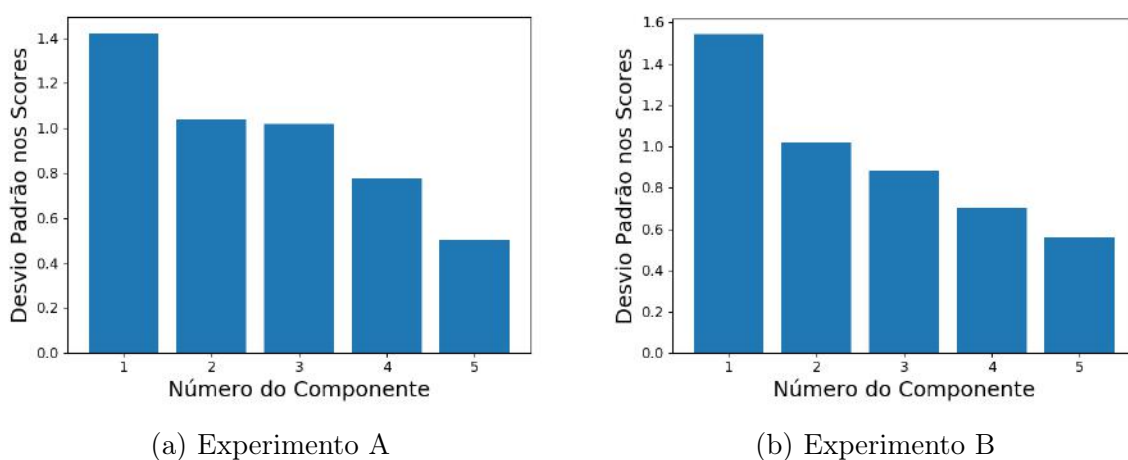


Figura 27 – Desvio Padrão dos Scores por Componente

Algo que vale a pena entender é o motivo dos gráficos de SPE obtidos a partir de mais de dois componentes não terem sido usados para detectar os outliers. A interpretação dos outliers que são detectados pelo SPE obtido a partir de um componente e os que são detectados pelo SPE obtido a partir de quatro componentes é diferente. O motivo mais imediato de se perceber é que as reproduções que são encontradas como outliers a partir do modelo com quatro componentes têm SPE absoluto menor do que as que são encontradas como outliers a partir do modelo com um componente. Isso já é um indicador de que o erro não é tão significativo. O que os outliers que são encontrados a partir de modelos com mais componentes apresentam são padrões mais diferentes, quando comparados com os que são encontrados a partir de modelos com poucos componentes. A intuição por trás disso está no fato de que os componentes principais mais altos são responsáveis por explicar as menores proporções de variância, ou seja, padrões incomuns de valores das métricas. As Figuras 28a e 28b mostram os valores de loadings referentes ao componente principal 5 dos experimentos A e B, respectivamente.

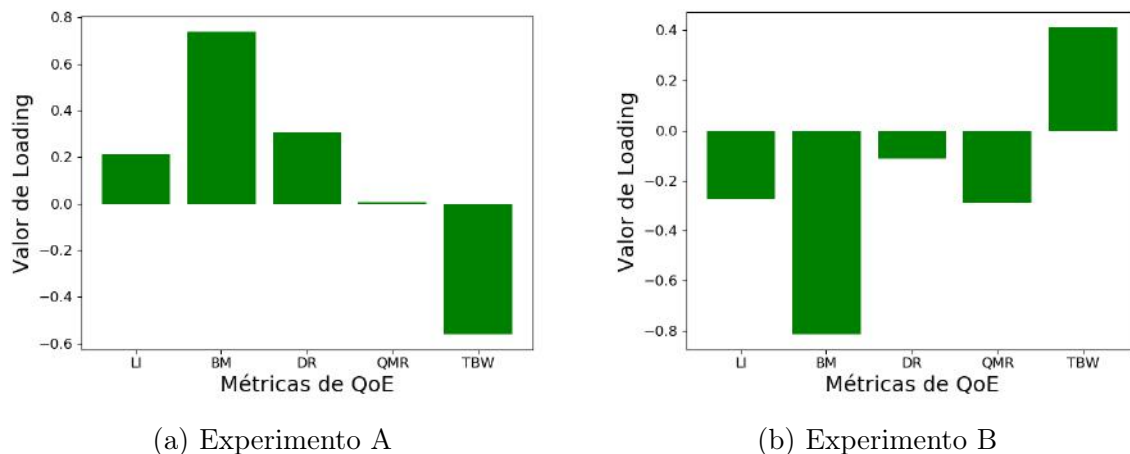


Figura 28 – Loadings do Componente Principal 5

Como pode ser visto, ambos apresentam alta correlação negativa entre BM e TBW, duas métricas que estão positivamente correlacionadas nos componentes principais mais significativos. Reproduções que forem encontradas como outliers de acordo com o modelo com quatro componentes terão valor alto nesse componente principal. Esses pontos podem ser interessantes de analisar, e essa detecção e interpretação de outlier é única para o PCA e, de forma mais geral, para métodos de análise fatorial, entretanto, para essa aplicação, não parece ser tão relevante. Um motivo adicional para ser feita a detecção de outliers a partir do SPE com um e dois componentes foi o fato das mesmas reproduções serem detectadas como outliers para ambos os modelos, em ambos os experimentos.

Os casos que foram detectados como outliers para o experimento B ocorreram em reproduções com tamanho de buffer WiFi de 10 pacotes. A reprodução 10 apresentou um valor diversos desvios padrões acima da média de DR enquanto a reprodução 1 apresentou um valor diversos desvios padrões acima da média de LI. Os valores das métricas de QoE e os scores da reprodução 1 do experimento B encontram-se representados nas Figuras 23a e 23b, respectivamente. De fato, olhando os gráficos de latência inicial e duração de rebufferizações, por tamanho de buffer, para o experimento B, representados nas Figuras 11e e 11d, respectivamente, é possível detectar essas reproduções outliers olhando o tamanho de buffer WiFi de 10 pacotes. Faz sentido que anomalias referentes a bufferizações tenham ocorrido com tamanho de buffer WiFi de 10 pacotes pois é esse tamanho que torna perda de pacotes mais comum, o que deve afetar fortemente o algoritmo de adaptação de taxa do YouTube, o que pode trazer muita variância nos resultados. Em experimentos futuros as estatísticas para nerds serão coletadas, o que inclui a estimativa de throughput do YouTube. Será interessante analisar a correlação da ocorrência dessas reproduções outliers com essa estimativa de throughput.

A reprodução que foi detectada como outlier no experimento A, entretanto, teve tamanho de buffer WiFi de 256 pacotes, o tamanho padrão e o mais alto que foi testado. A relação entre as métricas de QoE para essa reprodução são muito similares à relação entre

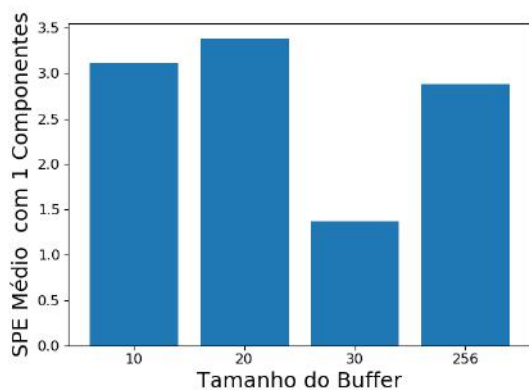
as métricas da reprodução 10 do experimento B. Assim como a reprodução 10 do experimento B, essa reprodução também poderia ter sido classificada como outlier somente olhando o gráfico de valores de DR por tamanho de buffer do experimento A, representado na Figura 11d. Isso não significa que o tamanho de buffer de 256 pacotes para o experimento B teve pior desempenho que o tamanho de buffer de 10 pacotes. Olhando o gráfico de valores de LI por tamanho de buffer do experimento A (Figura 11e) percebe-se que houve três reproduções com valores muito altos de LI, para o tamanho de buffer de 10 pacotes, 2.5 a 4 vezes o valor de DR para a reprodução que foi detectada como outlier. Nenhuma delas foi detectada como outlier pois três de dez reproduções com valores altos foi o suficiente para que, na hora de fazer a centralização na média, esses valores enviassem tanto a média que os valores de LI dessas reproduções não ficassem tão longe da média quanto, por exemplo, uma única reprodução que difere drasticamente das outras, como foi o caso com a reprodução 37, em relação ao valor de DR.

As Figuras 29a - 29h mostram o SPE médio para os diferentes tamanhos de buffer com modelos feitos de um componente até quatro componentes. O único caso onde o SPE médio para as reproduções com tamanho de buffer de 10 pacotes não é tão alto é para o modelo feito com quatro componentes, do experimento A, ainda assim, sendo o segundo valor de SPE mais alto, em média.

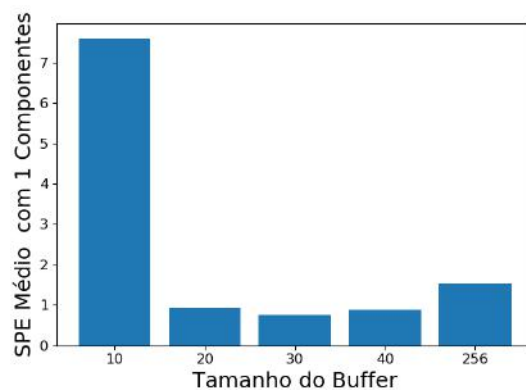
Um exercício interessante é ver o que acontece com o modelo quando se filtra os casos anômalos. O experimento B será o foco dessa análise. Após o modelo ser recalculado, sem os dois casos outliers, foram obtidos os novos loadings, representados na Figura 30a. Os vetores de loadings têm módulo igual a um quando todos os componentes principais são usados. Segue abaixo a prova.

A matriz ortonormal V contém os vetores de loadings em suas linhas. Sabemos que $V^T V = V V^T = I$. Logo, $v_a \cdot v_a = 1$, onde v_a é o vetor de loading referente à variável a e \cdot é o operador de produto interno. Logo, $|v_a| = 1, \forall a$, como queríamos demonstrar.

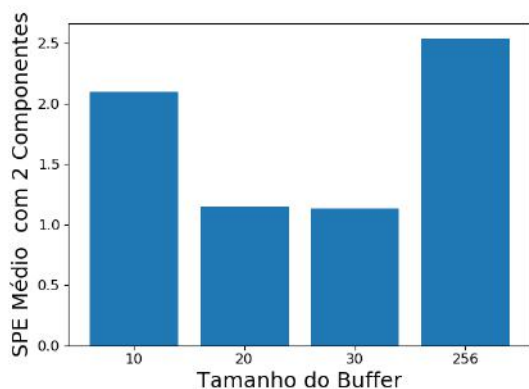
Ao comparar com os loadings originais obtidos, o valor de loading de DR está agora maior ainda no componente principal 2, com valor quase igual a um, logo, conclui-se que os loadings correspondentes a DR estão contribuindo praticamente inteiramente aos scores referentes ao componente principal 2. Percebe-se também que a proporção de variância explicada pelos dois componentes mais significativos diminui de 67% para 66%. A Figura 30b retrata o SPE desse novo modelo, com somente o componente principal mais significativo. Uma reprodução se destaca das outras, o que indica que pode ser feita uma nova filtragem, dessa vez da 4a reprodução (agora 3a pois a 1a já havia sido filtrada). Novamente a reprodução corresponde a um valor de tamanho de buffer de 10 pacotes. Fazendo a filtragem dessa reprodução adicional é obtido mais um modelo. Os loadings para esse novo modelo estão representados na Figura 31a.



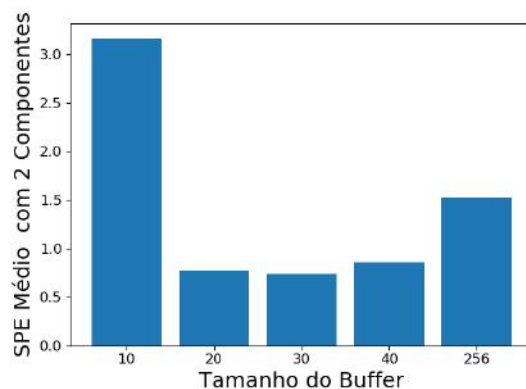
(a) Um PC (Experimento A)



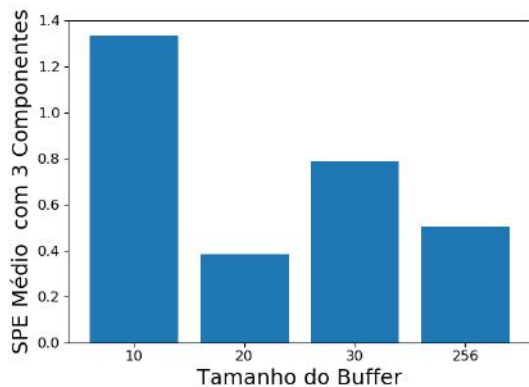
(b) Um PC (Experimento B)



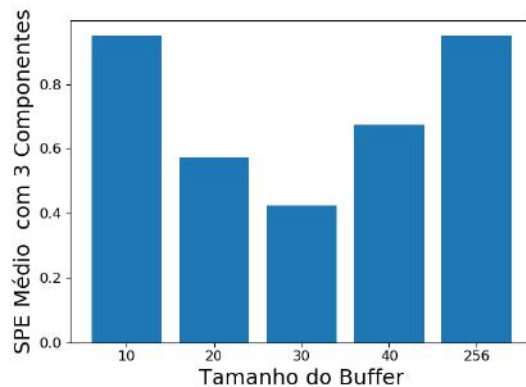
(c) Dois PCs (Experimento A)



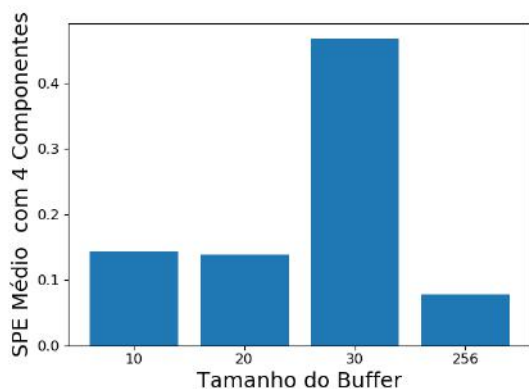
(d) Dois PCs (Experimento B)



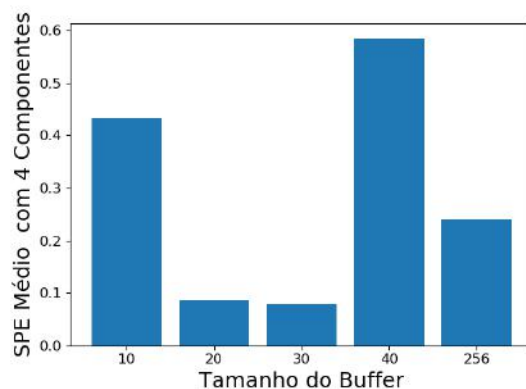
(e) Três PCs (Experimento A)



(f) Três PCs (Experimento B)



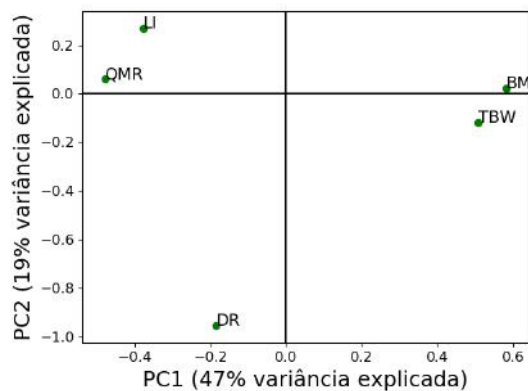
(g) Quatro PCs (Experimento A)



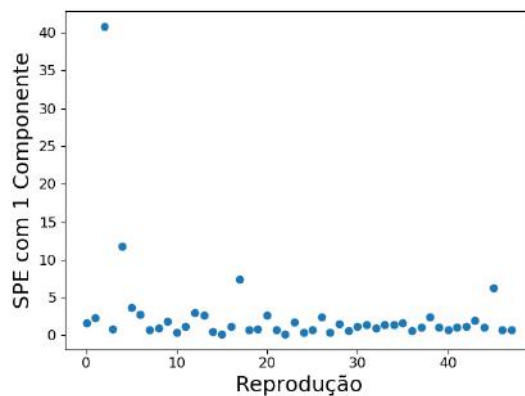
(h) Quatro PCs (Experimento B)

Figura 29 – SPE Médio por Tamanho de Buffer com Diferentes Quantidades de Componentes Principais

Essa figura trás duas observações importantes. Uma é que o valor de DR está como zero nos dois componentes principais mais significativos. Isso acontece porque a quarta e décima reprodução, ambas classificadas como outliers, correspondem aos únicos casos em que houve rebufferização no experimento B. Isso permite uma interpretação mais tácita do motivo pelo qual as reproduções foram identificadas como outliers. Isso também significa que agora o modelo pode ser representado completamente somente com quatro autovetores e autovalores, como pode ser conferido na Figura 31b, que mostra o scree plot para esse novo modelo. A outra observação é que nesse novo modelo a proporção de variância explicada pelos dois componentes principais mais significativos é 79%. Ou seja, a filtragem de somente uma observação num dataset com 48 observações mudou esse valor de 66% para 79%. Essa é uma mudança muito grande e é um sintoma da grande suscetividade do PCA a ruídos. Essa suscetividade se dá pois os dados referentes a uma variável têm seus valores subtraídos pela média dessa variável. Só é necessário que uma observação dessa variável tenha uma magnitude muito grande ou muito pequena em relação às outras para que essa centralização fique enviesada, como é o caso com DR no experimento A e DR no experimento B após os dois primeiros outliers serem filtrados. De fato, uma das conclusões em (BRO; SMILDE, 2003) é que a centralização no zero afeta muito mais o modelo do que a escala, podendo ser interpretado como a adição de um componente principal.

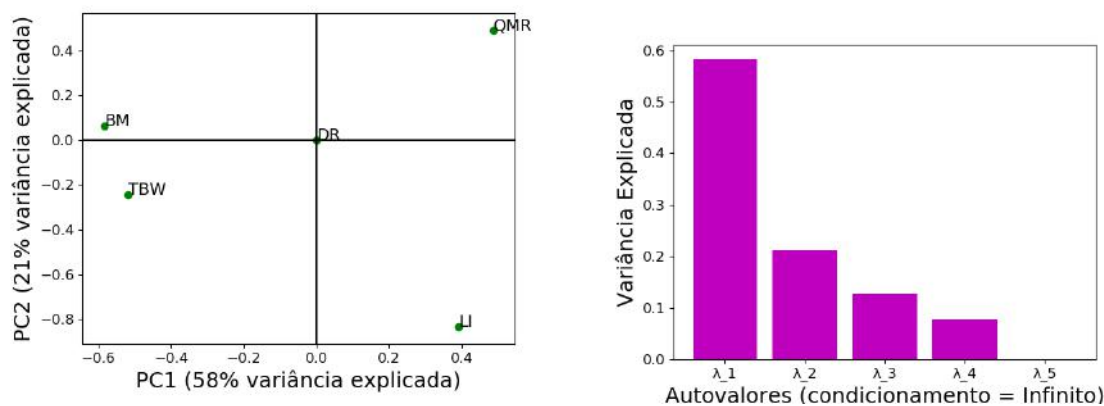


(a) Loadings do Experimento B após Filtragem de Outliers



(b) SPE do Experimento B com Um Componente após Filtragem de Outliers

Figura 30 – Experimento B Após Filtragem de Outliers



(a) Loadings do Experimento B após Filtragem Adicional de Outliers

(b) Scree plot do Experimento B após Filtragem Adicional de Outliers

Figura 31 – Experimento B Após Filtragem Adicional de Outliers

A importância e interpretação de outliers depende completamente da aplicação. Em alguns casos eles podem ser encontrados e filtrados para fazer um modelo que representa melhor os dados. Um exemplo onde isso pode ser útil é com os dados do experimento B. Se eu quiser fazer um modelo que representa o comportamento "padrão" de métricas de QoE sob condições de WiFi, casos anômalos não são desejáveis, pois somente enviesariam o modelo. Uma forma de interpretar isso é que o ruído estaria sendo modelado, ou *fittado*. A decisão em si do que qualifica um outlier geralmente é um processo mais "artesanal", porém alguns critérios e recomendações existem para diferentes métodos. Em outros casos encontrar o outlier é o objetivo em si. Um exemplo onde isso é o caso é na detecção de ataques DDOS numa rede onde se monitoram os dados de tráfego. Se o modelo foi treinado com dados onde não houve ataque, quando houver o ataque é provável que o modelo não conseguirá fazer uma boa aproximação dos dados e esse tráfego anômalo aparecerá nos residuais.

Uma outra análise que vale a pena ser feita usando o PCA é calcular o valor R^2 . R^2 pode ser calculada tanto por variável quanto para o modelo inteiro. No caso do cálculo por variável, temos que

$$R_a^2(k) = 1 - \frac{\text{var}(e_a)}{\text{var}(x_a)}, \quad a = 1, 2, \dots, d, \quad k = 1, 2, \dots, d, \quad (5.43)$$

onde e_a representa o a -ésimo vetor coluna da matriz de residuais E_k obtida a partir do uso dos k componentes principais correspondentes aos k maiores autovalores e x_a representa o a -ésimo vetor coluna da matriz de dados após centralização no zero e normalização pelo desvio padrão. Tanto R^2 total quanto por variável pode ser calculado referente a modelos formados a partir de diferentes quantidades de componentes principais, logo, a notação $R_a^2(k)$ é usada para indicar que está sendo calculada a medida R_a^2 referente ao modelo com k componentes principais. Como a variância para todas as métricas em X equivale

a um, essa equação pode ser reescrita como

$$R_a^2(k) = 1 - \text{var}(e_a), \quad a = 1, 2, \dots, d, \quad k = 1, 2, \dots, d. \quad (5.44)$$

Sabemos que $\text{var}(e_a)$ está contida no intervalo $[0, 1]$, pela construção da matriz E , logo, $R_a^2(k)$ é uma fração com valor no intervalo $[0, 1]$. Quanto maior a variância da coluna na matriz de residuais, correspondente à métrica a , menos o modelo M_k conseguiu explicar a variância, e menor será o valor de $R_a^2(k)$. Ou seja, $R_a^2(k)$ é uma medida do quanto a variância da métrica a foi explicada pelo modelo M_k , com $R_a^2(k) = 1$ indicando que o modelo explicou toda a variância dessa métrica e $R_a^2(k) = 0$ indicando que o modelo explicou nada da variância dessa métrica. De fato, $R_a^2(k)$ pode ser definida como

$$R_a^2(k) = \text{var}(m_a), \quad a = 1, 2, \dots, d, \quad k = 1, 2, \dots, d, \quad (5.45)$$

onde m_a equivale à a -ésima coluna do modelo M_k . Aplicando a definição de variância, temos que

$$R_a^2(k) = \sum_{i=1}^n (m_{ia} - \bar{m}_a)^2, \quad a = 1, 2, \dots, d, \quad k = 1, 2, \dots, d, \quad (5.46)$$

onde \bar{m}_a equivale à média da a -ésima coluna da matriz M_k . É fácil ver que, como M_k é o resultado de uma transformação ortogonal e então a inversa dessa transformação ortogonal, a média em cada coluna da matriz M_k equivale a 0, que é a média das colunas na matrix X após ser centrada em zero, para $k = 1, 2, \dots, d$. Logo,

$$R_a^2(k) = \sum_{i=1}^n m_{ia}^2, \quad a = 1, 2, \dots, d, \quad k = 1, 2, \dots, d. \quad (5.47)$$

Para o cálculo de R^2 para o modelo inteiro temos que

$$R^2(k) = 1 - \frac{\text{var}(E_k)}{\text{var}(X)}, \quad k = 1, 2, \dots, d, \quad (5.48)$$

onde

$$\text{var}(E_k) = \sum_{a=1}^d \sum_{i=1}^n \frac{(e_{ia} - \bar{e})^2}{l-1}, \quad k = 1, 2, \dots, d, \quad (5.49)$$

onde l é a quantidade de elementos na matriz E_k e \bar{e} é a média dos elementos no residual E . O processo para X é análogo. Entretanto, como E_k e X possuem a mesma quantidade de elementos e haverá a divisão de uma dessas parcelas pela outra, não será necessário fazer a divisão por $(l-1)$. Além disso, como os dados de X já estão centralizados em zero, $\bar{x}_a = 0, a = 1, 2, \dots, d$. Logo,

$$R^2(k) = 1 - \sum_{a=1}^d \sum_{i=1}^n \frac{(e_{ia} - \bar{e})^2}{x_{ia}^2}, \quad k = 1, 2, \dots, d. \quad (5.50)$$

Essa é uma medida de quanto da variância dos dados foi explicada pelo modelo M_k , sendo $R^2(k) = 1$ indicativo de que o modelo explicou os dados perfeitamente, e $R^2(k) = 0$

indicativo de que o modelo não conseguiu explicar minimamente os dados. Analogamente à medida R^2 por métrica, o valor de R^2 total pode ser definido como

$$R^2(k) = \sum_{a=1}^d \sum_{i=1}^n \frac{(m_{ia} - \bar{m})^2}{x_{ia}^2}, \quad (5.51)$$

onde \bar{m} é a média dos elementos no modelo M_k . Como a média de todas as colunas de M_k é igual a 0, temos que a média da matriz inteira é igual a 0, para todo $k = 1, 2, \dots, d$, logo,

$$R^2(k) = \sum_{a=1}^d \sum_{i=1}^n \frac{m_{ia}^2}{x_{ia}^2}, \quad k = 1, 2, \dots, d. \quad (5.52)$$

Uma forma adicional de representar a medida R^2 para o modelo inteiro é

$$R^2(k) = \frac{\text{var}(T_k)}{\text{var}(X)}, \quad (5.53)$$

Isso é verdade pois a variância no modelo M_k é igual à variância na matriz de scores T_k . Novamente, isso se dá pelo fato de M_k ser obtido a partir de uma transformação ortogonal aplicada em T_k , o que preserva ângulos e distâncias entre pontos, logo, preserva a variância. Como foi provador na seção B do Apêndice, a média dos elementos da matriz de scores T_k é igual a 0, para todo $k = 1, 2, \dots, d$, logo,

$$R^2(k) = \sum_{i=1}^n \frac{\sum_{a=1}^k t_{ia}^2}{\sum_{b=1}^d x_{ib}^2}, \quad k = 1, 2, \dots, d, \quad (5.54)$$

onde t_{ia} corresponde ao termo da i -ésima linha e a -ésima coluna da matriz T_k .

As Figuras 32a e 32b mostram os valores de $R^2(k)$ para os modelos dos experimentos A e B, respectivamente, de acordo com a quantidade k de componentes principais usados. Como pode ser visto, para $k = d$, toda a variância é explicada, ou seja, $R^2(d) = 1$. Isso faz sentido pois, como foi discutido anteriormente, tudo que foi feito em cima dos dados, nesse caso, foi uma rotação seguida da rotação inversa, na mesma dimensionalidade original, o que preserva toda a variância dos dados, ou seja, nenhuma informação é perdida.

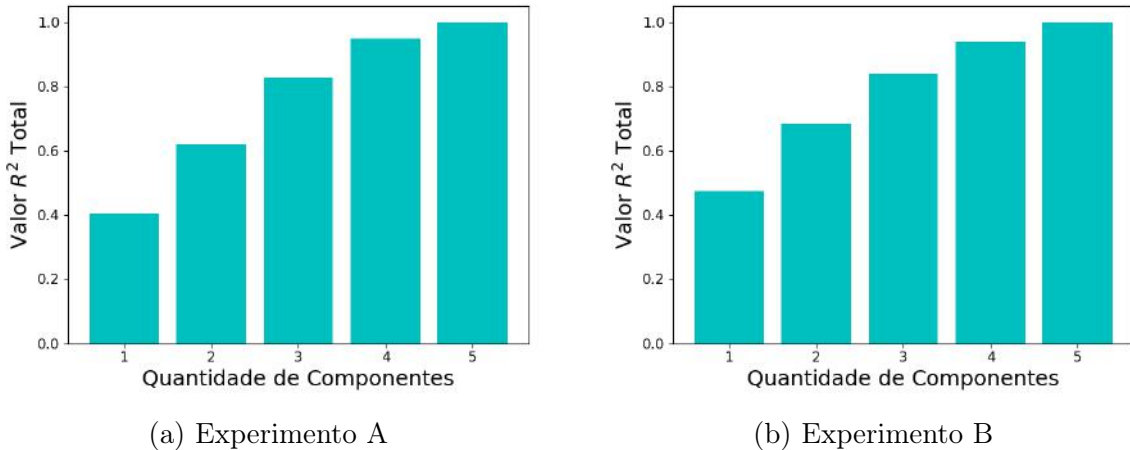


Figura 32 – R^2 por Quantidade de Componentes

Algo que vale a pena notar é que o valor de R^2 equivale à soma das proporções da variância explicada pelos autovalores correspondentes a cada componente principal. Isso ocorre pois as soluções do PCA são aninhadas, ou seja, o modelo com d componentes irá usar os $d - 1$ componentes do modelo com $d - 1$ componentes, que por sua vez irá usar os $d - 2$ componentes do modelo com $d - 2$ componentes, e assim em diante. Ou seja, ao somar cumulativamente, de forma crescente em relação ao número do componente principal, os valores representados nos scree plots para os experimentos A e B, obtém-se os valores de R^2 referentes aos modelos que usam essa quantidade de componentes principais. A medida R^2 é obtida de forma empírica, o que significa que comparar esses valores é uma boa forma de validar se o algoritmo do PCA foi implementado de forma correta ou não.

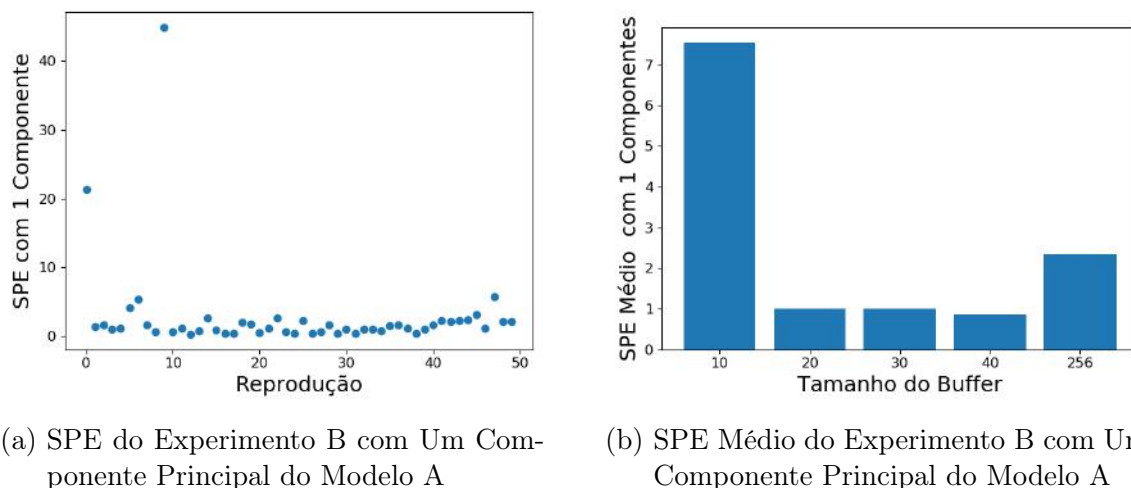
Uma análise adicional que pode ser feita é ver o quão bem o modelo obtido a partir do experimento A (a partir de agora chamado de modelo A e, analogamente para B, modelo B) explica os dados do experimento B e vice versa. Ambas as análises serão feitas, começando pela avaliação do uso do modelo A para *fit* os dados do experimento B. Seja V_A a matriz de loadings obtida a partir do experimento A. O primeiro passo é obter a projeção dos dados do experimento B, já centralizados e normalizados, contidos em X_B , nos autovetores contidos em V_A .

$$T_B^{(A)} = X_B V_A \quad (5.55)$$

Onde $T_B^{(A)}$ é a matriz de scores de B, obtida a partir do modelo A. Feito isso, pode-se obter o modelo $M_B^{(A)}$ (modelo B, obtido a partir do modelo A),

$$M_B^{(A)} = T_B^{(A)} V_A^T \quad (5.56)$$

Feito isso, é possível obter a matriz de residuais $E_B^{(A)}$, subtraindo-se os termos do modelo $M_B^{(A)}$ da matriz X_B . Uma vez que se tem a matriz de residuais é possível obter o SPE para esse novo modelo. A Figura 33a mostra o SPE obtido com o componente principal mais significativo do modelo A enquanto a Figura 33b mostra o valor médio de SPE por tamanho de buffer usando esse mesmo componente principal. Em média, o SPE está mais alto nesse caso do que usando o componente principal mais significativo do próprio modelo B, como esperado, mais especificamente, fica mais alto para os tamanhos de buffer de 20, 30 e 256 pacotes. Algo interessante a se notar é que os mesmos dois outliers podem ser identificados, indicando que eles divergem do comportamento geral de ambos os experimentos.



(a) SPE do Experimento B com Um Componente Principal do Modelo A

(b) SPE Médio do Experimento B com Um Componente Principal do Modelo A

Figura 33 – Análise dos Dados do Experimento B usando o Modelo A

A Figura 34a mostra os scores $T_B^{(A)}$ usando os dois componentes principais mais significativos do modelo A. Como pode ser visto, o componente principal mais significativo do modelo A explica 43% da variância dos dados do experimento B enquanto o segundo componente principal mais significativo do modelo A explica 12% da variância dos dados do experimento B, totalizando a explicação de 55% da variância do experimento B. Os dois componentes principais mais significativos do experimento B explicam 67% da variância dos dados do experimento B. A forma que a proporção da variância explicada dos dados do experimento B por cada um dos componentes principais do modelo A foi calculada foi pegando a diferença entre os valores de R^2 usando diferentes quantidades de componentes principais. Isso demonstra uma vantagem do cálculo empírico da variância no modelo final nos scores ao invés de diretamente pela proporção dos autovalores correspondentes aos componentes principais. As medidas R^2 dos dados do experimento B quando fittados pelo modelo A são representadas na Figura 34b. Como pode ser visto, diferente da medida R^2 quando se usa os componentes principais obtidos a partir da matriz de correlação dos próprios dados, quando se usa os componentes principais obtidos a partir de outro modelo, a diferença entre os valores não decresce a cada componente adicional. Isso significa que os dois componentes principais do modelo A que explicam a maior proporção de variância dos dados do experimento B não necessariamente são os dois primeiros. De fato, o componente principal 3 e 4 do modelo A explicam 18% da variância dos dados de B, que é mais do que o componente principal 2, que só explica 12%. As Figuras 35a e 35b mostram os scores $T_B^{(A)}$ levando em conta o primeiro e terceiro componente e o primeiro e quarto componente, respectivamente. Em ambos os casos a proporção de variância explicada pelos dois componentes é 61%.

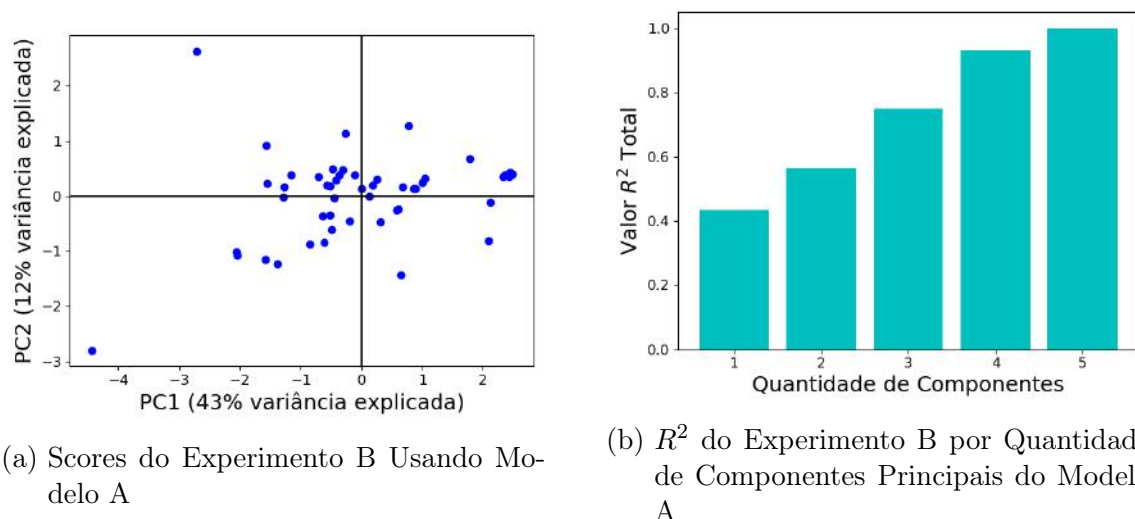


Figura 34 – Análise da Capacidade do Modelo A Explicar os Dados de B

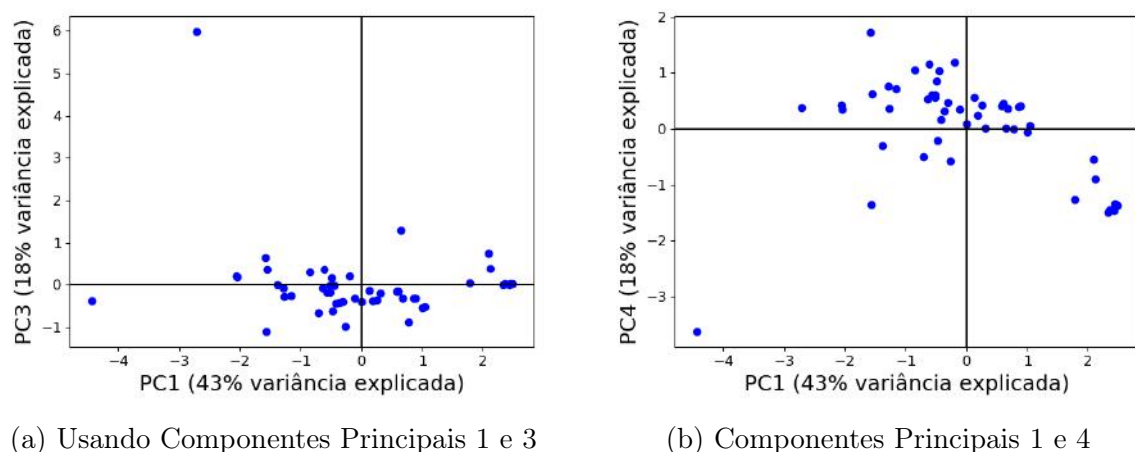


Figura 35 – Scores do Experimento B Usando Diferentes Componentes Principais do Modelo A

Análogo a como foi feito anteriormente, o SPE, R^2 e scores foram obtidos, dessa vez para os dados do experimento A, fittados pelo modelo B. O SPE por reprodução e o SPE médio por tamanho de buffer, usando o componente principal mais significativo do modelo B estão representados nas Figuras 36a e 36b, respectivamente. Como anteriormente, o valor médio do SPE está mais alto do que usando o modelo original de A. Além disso, como anteriormente, o mesmo outlier pode ser detectado.

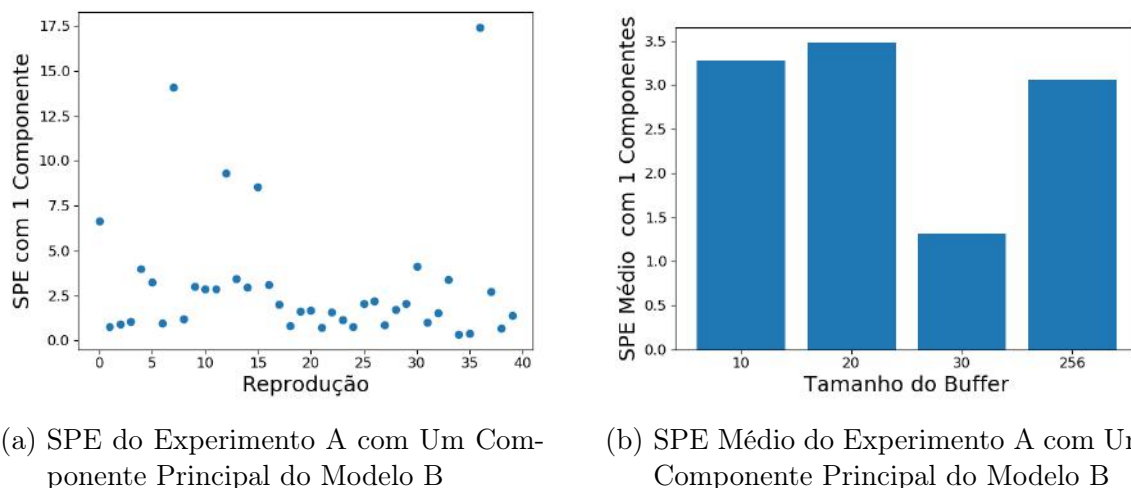
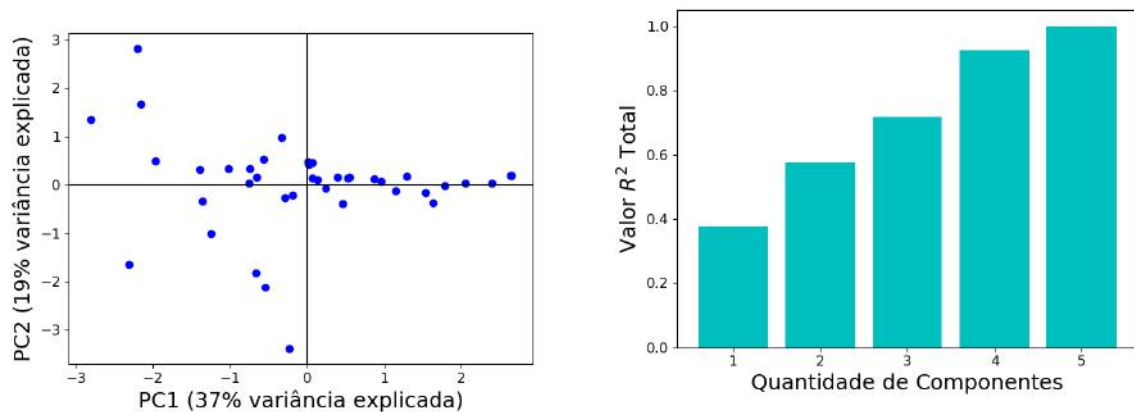


Figura 36 – Análise dos Dados do Experimento A usando o Modelo B

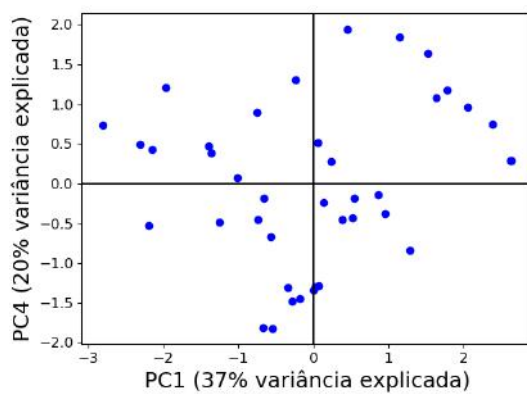
Os scores $T_A^{(B)}$ e a medida R^2 para os dados do experimento A usando os componentes principais do modelo B estão representados nas Figuras 37a e 37b, respectivamente. A proporção de variância explicada dos dados do experimento A pelos dois componentes principais mais significativos do modelo B é 56%, onde originalmente é 61%. Entretanto, usando os componentes principais 1 e 4, do modelo B, 57% da variância pode ser explicada, como pode ser visto na Figura 37c.

Num cenário onde há diversas variáveis somente os componentes principais mais significativos serão usados para avaliar o modelo. Levando isso em consideração, testar todas as combinações dos componentes principais não é uma metodologia sensata para se avaliar a capacidade de generalização de um modelo para fittar outros dados. Logo, levando em conta os componentes principais na ordem em que foram encontrados no modelo original, o modelo A descreve, com dois componentes principais, 55% da variância dos dados do experimento B, cujo modelo original consegue descrever 67%, enquanto o modelo B descreve, com dois componentes principais, 56% da variância dos dados do experimento A, cujo modelo original consegue descrever 61%. Logo, é possível concluir que o modelo B generaliza melhor para os dados. Isso já era esperado pois os dados do experimento B possuem 25% de amostras a mais do que os dados do experimento A, além de possuir amostras provenientes de experimentos feitos com um tamanho de buffer adicional. Uma outra análise que poderia ser feita é a criação de um modelo usando dados dos dois experimentos, porém isso será deixado como um trabalho futuro, quando houver mais dados coletados para diferentes dias, com diferentes tamanhos de buffer.



(a) Scores do Experimento A Usando Modelo B

(b) R^2 do Experimento A por Quantidade de Componentes Principais do Modelo B



(c) Scores do Experimento A Usando Componentes Principais 1 e 4 do Modelo B

Figura 37 – Análise da Capacidade do Modelo B Explicar os Dados de A

Antes do PCA ser aplicado, foi instrutivo visualizar os dados em formato de tabela, pois a dimensionalidade dos dados se tornou saliente, porém, principalmente pois isso induz o uso de uma matriz como objeto matemático para representá-los. Tabelas remetem imediatamente à ideia de matrizes, porém, ao inspecionar a Tabela 11 novamente, percebe-se que ela poderia ser dividida em diversas tabelas, uma para cada tamanho de buffer. De fato, quando o PCA foi aplicado para cada tamanho de buffer WiFi, individualmente, isso estava sendo feito implicitamente. Depois elas só foram "coladas" umas nas outras, adicionando-se uma dimensão, a de tamanho de buffer WiFi. Matrizes, no entanto, são um caso específico de um objeto matemático mais geral: o tensor.

Tensores podem ser de qualquer ordem d , $d \in \mathbb{Z}^+$. Vetores são tensores de ordem $d = 1$ e matrizes são tensores de ordem $d = 2$. Pode-se imaginar um tensor de terceira ordem como um cubo ou várias matrizes "empilhadas". Como foi sugerido anteriormente, a estrutura dos dados na Tabela 9 não encaixa muito bem com uma matriz; faz muito mais sentido imaginarmos os dados como várias matrizes empilhadas. A Figura 38 mostra como os dados podem ser imaginados em formato de tensores.

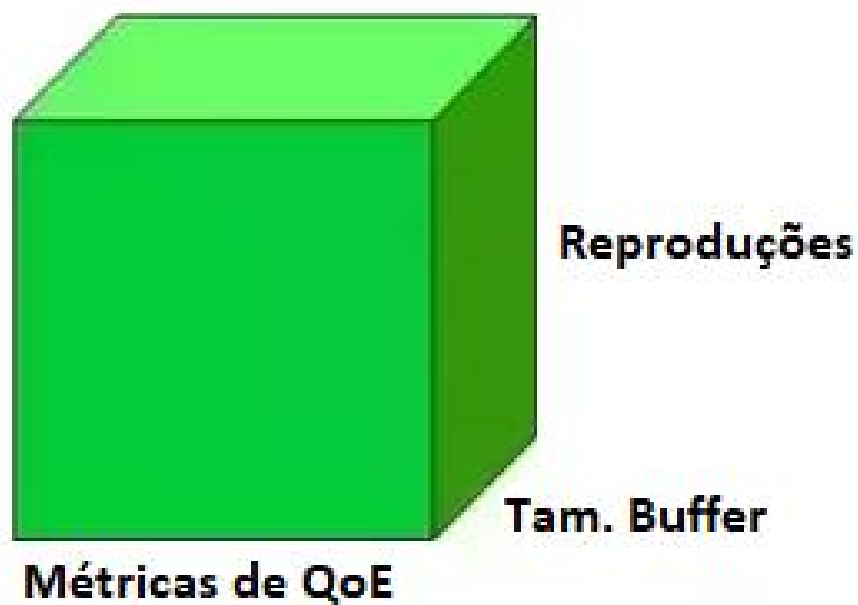


Figura 38 – Dados dos Experimentos em Formato de Tensor

Tensores possuem suas próprias propriedades, e até hoje, existem diversos problemas em aberto na área. Além disso, existem alguns métodos que possuem propriedades muito desejáveis na análise de tensores, são esses os métodos de decomposição tensorial, um deles sendo uma extensão natural do PCA. Por questões de tempo e espaço a descrição desses métodos e os resultados obtidos quando aplicados nos dados serão deixados para um trabalho futuro.

6 TRABALHOS FUTUROS

Como mencionado na introdução, o trabalho que foi descrito é uma das etapas iniciais de um projeto cujo objetivo final é o desenvolvimento de uma ferramenta para ISPs, baseado em QoE. Graças às parcerias com a startup Anlix e o ISP Gigalink, o LAND tem acesso a dados que são difíceis de obter, a partir dos milhares de RDs. Esses RDs permitem a coleta, atualmente, a cada minuto, de métricas agregadas da camada de rede. Entretanto, as métricas que foram usadas nas análises são provenientes da camada de aplicação, além do tamanho do buffer do driver. Pode-se interpretar o processo de obtenção das métricas de QoE como um mapeamento determinístico das métricas da aplicação para métricas de QoE, como representado na Figura 39a, pela seta preenchida. No futuro, experimentos parecidos serão feitos com usuários voluntários da Gigalink, onde tanto métricas de rede quanto métricas de aplicação serão coletadas, durante reproduções de vídeo em diferentes plataformas de streaming. As métricas de rede já são coletadas normalmente. As métricas de aplicação serão coletadas com o uso da extensão de Google Chrome desenvolvida pela equipe do INRIA. Novamente, a partir das métricas de aplicação, serão obtidas, de forma determinística, as métricas de QoE. Entretanto, é possível fazer um mapeamento das métricas de rede para as métricas de aplicação, como representado na Figura 39b. A seta, neste caso, não está preenchida, para indicar que pode haver erro nesse mapeamento. A ferramenta final deve usar somente métricas de redes que são coletadas a partir dos roteadores dos clientes, as mesmas que estão descritas no capítulo 4. Uma vez que o mapeamento das métricas de rede para as métricas de aplicação for feito, teremos um modelo que consegue inferir as métricas de QoE a partir das métricas de rede, como está indicado na Figura 39c. Além disso, o modelo pode ser estendido para também adicionar as métricas de WiFi, descritas no capítulo 4, ao conjunto de variáveis conhecidas.

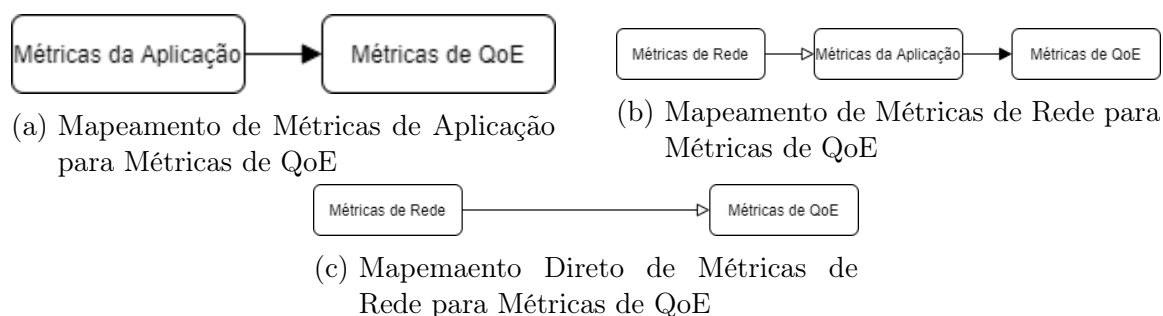


Figura 39 – Interação entre Métricas de Rede, Métricas de Aplicação e Métricas de QoE

Um problema, que já foi levantado no capítulo 2, é que o buffer do driver, o que está sendo analisado, não possui diversas capacidades que são desejáveis. A camada que possui essas funcionalidades é a QDisc, que fica numa camada acima do buffer do driver. É esse

o buffer que será analisado em experimentos futuros, pois, além de possuir as funcionalidades desejáveis, é altamente configurável. Modificar o buffer de driver de roteadores dos usuários na rede de um ISP não é prático, entretanto, usar a ferramenta BQL para trazer o controle do gerenciamento de filas e agendamento de pacotes para a camada QDisc é. Uma vez que o controle está nessa camada, o alto grau de configurabilidade pode ser explorado.

É possível que, além de coletar as métricas de rede e de aplicação dos usuários voluntários da Gigalink, também sejam feitas rápidas pesquisas onde usuários dão uma nota de um a cinco ao serviço de streaming de vídeo que consumiram, o que tornaria possível a análise de QoE a partir de técnicas baseadas em MOS. Dessa forma, a modelagem poderá se estender para a inferência da nota MOS a partir das métricas de rede. Entretanto, como mencionado na introdução, diversos cuidados devem ser tomados quando está se fazendo esse tipo de experimento.

Uma possível modelagem para inferir as métricas de QoE a partir de métricas de rede e outras métricas do roteador consiste no uso de estatística bayesiana. Estatística bayesiana possui algumas vantagens sobre métodos frequentistas. O uso de priors significa que não são necessárias tantas amostras para se obter resultados estatisticamente significativos. Além disso, o método não retorna um valor, e sim uma distribuição para as métricas de interesse. Por exemplo, dado um conjunto de valores de métricas de rede, o modelo retornaria uma distribuição de DR (Duração de Rebufferização). Mesmo que essas distribuições sejam complicadas é possível amostrar usando técnicas de Markov Chain Monte Carlo e então extrair estatísticas úteis.

Um dos trabalhos ocorrendo paralelamente a esse consiste em resolver o problema de amostragem da perda de pacotes. Esse problema surge a partir da granularidade da coleta dos dados. A métrica QPP nos dá a quantidade de pacotes que são perdidos a cada minuto, porém, perda de pacotes geralmente possui características de rajada, devido a diversos pacotes que são perdidos em janelas de tempo na ordem de milissegundos. Simular essas diferentes características de perda, a partir das amostras de QPP, é um passo importante para realizar experimentos mais precisos em laboratório. Não adianta realizar experimentos onde as condições de rede são completamente diferentes das condições que usuários comuns de um ISP experienciam. Os experimentos com os usuários da Gigalink podem ajudar nessa etapa, pois é provável que a granularidade na coleta dos dados desse subconjunto de usuários seja mais fina. O nível de granularidade depende de alguns fatores: do lado do servidor, depende tanto da capacidade de armazenamento quanto da velocidade de escrita desses dados. Do lado dos usuários, deve-se tomar cuidado para que a CPU dos roteadores não fique sobrecarregada, pois isso pode acarretar em perda indesejada de pacotes.

Outro projeto também em andamento consiste em correlacionar o padrão de perda de pacotes dos usuários com chamados que a Gigalink recebe. A extensão natural para

esse projeto é correlacionar a QoE dos usuários com os chamados. Isso pode ser de grande benefício para ISPs pois muitos recursos são gastos para atender e resolver os problemas reportados nos chamados. Além disso, diversos dos problemas que causam pessoas a ligar para seu ISP não são provenientes de problemas na rede e sim problemas com os equipamentos. Conseguir diferenciar esses tipos de chamado, além de inferir que usuários, ou, de forma mais geral, que partes da topologia, estão com uma QoE que os leva a realizarem esses chamados é de grande valor para ISPs.

Uma das maiores limitações desse trabalho foi o escopo dos experimentos. Não só a quantidade de reproduções foi pequena, por experimento, os tipos de vídeos foram parecidos e foram todos provenientes da mesma plataforma, o YouTube. Isso dificulta a obtenção de resultados estatisticamente significativos e generalizáveis. Em experimentos futuros haverá uma variedade maior de vídeos e plataformas testadas, além de mais reproduções. Para minimizar ao máximo o efeito de interferência no sinal do WiFi, que é intrinsecamente instável (BISWAS et al., 2015), será usada a frequência de 5GHz, que é menos comum. Além disso, dois roteadores serão usados, um com funcionalidade de AP e um para realizar a emulação de tráfego, como feito em (PITTONI, 2016). Antes dos experimentos serem feitos com os usuários da Gigalink, serão feitos experimentos adicionais, com as mudanças descritas neste capítulo, em laboratório. Além das métricas de QoE, também serão coletados nesses experimentos, as métricas de rede e WiFi que são coletadas dos roteadores dos usuários da Gigalink, entretanto, numa granularidade mais fina. Dessa forma, será possível comparar as condições do laboratório com os dos usuários, para que os ajustes necessários sejam feitos para experimentos futuros em laboratório.

7 CONCLUSÃO

Neste trabalho, uma interação complexa foi abordada: a qualidade de experiência de usuários assistindo vídeos através do WiFi. É bastante aceito na comunidade de QoE que as variáveis que foram usadas no trabalho representam adequadamente o espectro completo ou quase completo de qualidade de experiência que um usuário consumindo um vídeo pode experimentar, apesar da importância relativa de cada uma delas ainda ser debatida. Na área de redes, buffers são considerados partes fundamentais de qualquer arquitetura, pois todos os aspectos referentes a eles mudam o caráter estatístico da dinâmica de transferência de pacotes, existindo workshops dedicados somente a pesquisas relacionadas a buffers.

Juntando o tamanho do buffer de driver WiFi com as métricas de QoE, foi usado o PCA, um modelo interpretativo, para entender como essas variáveis se correlacionam. Com conhecimento adicional de redes, foi possível entender algumas das dinâmicas diferentes que ocorreram. O padrão mais consistente foi entre bitrate médio e o tamanho do buffer. Em todos os experimentos, essas duas métricas estavam positivamente correlacionadas. O segundo padrão mais consistente foi a relação entre o bitrate médio e a quantidade de mudanças de resolução. Houve alta correlação negativa entre essas duas variáveis, sendo que em sete dos nove diferentes tipos de testes (contando testes com buffers de mesmo tamanho porém em diferentes dias como diferentes tipos) a métrica de QoE com maior correlação negativa com bitrate médio foi a quantidade de mudanças de resolução. Ambos esses resultados correspondem com nossas intuições. Os padrões mais interessantes que surgiram envolveram a relação entre duração de rebufferizações e latência inicial. Em três dos nove diferentes tipos de teste, essas duas métricas estavam negativamente correlacionadas, o que indica que, uma quantidade não negligenciável de experimentos apresentou duração de rebufferizações alta e latência inicial baixa, e vice versa. É fácil entender o motivo disso, porém isso evidencia uma das grandes complicações com entender a QoE subjetiva dos usuários: pode ser que uma métrica que intuitivamente pareça ser negativamente correlacionada com QoE, empiricamente acabe sendo positivamente correlacionada. Somente com experimentos em outras plataformas e outros tipos de vídeo, assim como obtenção de feedback dos usuários, isso pode ser compreendido melhor.

Como já mencionado anteriormente este trabalho é um passo inicial num projeto maior, cujo objetivo é instrumentalizar QoE. Tanto os resultados das análises dos dados como o processo de superar os percalços encontrados ao longo da execução do trabalho contribuíram para indicar os caminhos mais propícios para o encaminhamento do projeto em que este trabalho está inserido.

REFERÊNCIAS

- AMMAR, D.; MOOR, K. d.; HEEGAARD, P. An experimental platform for qoe studies of webrtc-based multi-party video communication. **IJNCAA**, p. 89–94, 2018.
- ANORGA, J. et al. Youtube’s dash implementation analysis. **19th International Conference on Communications**, Zakynthos Island, p. 61–66, 2016.
- ATKINSON, D. et al. Buffering at the edge: Measuring from home-routers. **BS ’19: Proceedings of the 2019 Workshop on Buffer Sizing**, Stanford University, 2019.
- AYAD, I. et al. A practical evaluation of rate adaptation algorithms in http-based adaptive streaming. **Computer Networks**, p. 90–103, 2018.
- BISWAS, S. et al. Large-scale measurements of wireless network behavior. **ACM SIGCOMM Computer Communication Review**, v. 45, n. 4, 2015.
- BRO, R. **Multi-way Analysis**. Disponível em: https://www.youtube.com/watch?v=_gIb6PzBEc4&list=PL4L59zaizb3E-Pgp-f90iKHdQQi15JJJoL.
- BRO, R.; SMILDE, A. K. Centering and scaling in component analysis. **Journal of Chemometrics**, v. 17, p. 16–33, 2003.
- BUFFERBLOAT. **bufferbloat**. Disponível em: <https://www.bufferbloat.net/>.
- CHROME, G. **Youtube Auto HD + FPS**. 2020. Disponível em: <https://chrome.google.com/webstore/detail/youtube-auto-hd-%20fps/fcphghnknhkimeagdglkijnmpbagone?hl=en>.
- CISCO. **Cisco Visual Networking Index: Forecast and Trends**. 2019. Disponível em: <https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visualnetworking-index-vni/white-paper-c11-741490.html?dtid=ossdc000283>.
- CISCO. **Cisco Annual Internet Report (2018–2023) White Paper**. 2020. Disponível em: <https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html>.
- DEEK, L. et al. A practical framework for 802.11 mimo rate adaptation. **Computer Networks**, 2015.
- DOBRIAN, F. Understanding the impact of video quality on user engagement. **SIGCOMM**, 2011.
- FCC. **Measuring Broadband America**. 2018. Disponível em: <https://www.fcc.gov/general/measuring-broadband-america>.
- FIEDLER, M.; MOLLER, S.; REICHL, P. Quality of experience: From user perception to instrumental metrics. **Dagstuhl Reports**, v. 2, p. 1–25, 2012.
- GHASEMI, M. et al. Performance characterization of a commercial video streaming service. **IMC**, Santa Monica, 2016.

- HOEILAND-JOERGENSEN, T. et al. The flow queue codel packet scheduler and active queue management algorithm. **Internet Engineering Task Force (IETF)**, 2018.
- HOEILAND-JOERGENSEN, T.; TAHT, D.; MORTON, J. Piece of cake: A comprehensive queue management solution for home gateways. **2018 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)**, 2018.
- HORA, D. N. d. et al. Passive wi-fi link capacity estimation on commodity access points. **Traffic Monitoring and Analysis Workshop (TMA) 2016**, Louvain-la-Neuve, 2016.
- HORA, D. N. d. et al. Predicting the effect of home wi-fi quality on qoe. **INFOCOM 2018 - IEEE International Conference on Computer Communications**, Honolulu, 2018.
- HOSSFELD, T. et al. Quantification of youtube qoe via crowdsourcing. **IEEE ISM**, 2011.
- INSIDER, B. **How to get a billion views on YouTube**. 2018. Disponível em: <https://www.businessinsider.com/how-to-get-billion-views-viral-hit-youtube-2018-4>.
- JAMSHAD, K. et al. Buffer sizing in 802.11 wireless mesh networks. **2011 IEEE Eighth International Conference on Mobile Ad-Hoc and Sensor Systems**, 2011.
- JOHARI, R.; TAN, W. K. H. End-to-end congestion control for the internet: delays and stability. **IEEE/ACM Transactions on Networking**, v. 9, p. 818–832, 2001.
- JOUMBLATT, D. Z. et al. Predicting user dissatisfaction with internet application performance at end-hosts. **IEEE INFOCOM (mini-conference)**, Turin, p. 235–239, 2013.
- KOLDA, T. **Tensor Decomposition**. Disponível em: <https://www.youtube.com/watch?v=L8uT6hgMt00>.
- KRISHNAPPA, D. K.; BHAT, D.; ZINK, M. Dashing youtube: An analysis of using dash in youtube video service. **IEEE LCN**, 2013.
- KUROSE, J. F.; ROSS, K. W. **Computer Networking: A Top-Down Approach**. 6. ed. [S.l.]: Pearson, 2013.
- LAVRENKO, V. **Principal Component Analysis**. Disponível em: https://www.youtube.com/playlist?list=PLBv09BD7ez_5_yapAg86Od6JeeypkS4YM.
- LI, T.; LEITH, D.; MALONE, D. Buffer sizing for 802.11 based networks. **IEEE/ACM Transactions on Networking**, v. 19, p. 156–169, 2011.
- LINUX. **Queueing in the Linux Network Stack**. 2013. Disponível em: <https://www.linuxjournal.com/content/queueing-linux-network-stack>.
- LWN. **Byte Queue Limits**. 2013. Disponível em: <https://lwn.net/Articles/469652/>.
- MACGREGOR, M. H.; SHI, W. Deficits for bursty latency-critical flows: Drr++. **Proceedings IEEE International Conference on Networks 2000 (ICON 2000). Networking Trends and Challenges in the New Millennium**, Singapore, 2000.

- MAN7. **tc-netem**. Disponível em: <https://man7.org/linux/man-pages/man8/tc.8.html>.
- MEHAOUED, K.; BOURENANE, M.; SEKHRI, L. Outgoing-flows-number based service differentiation for fair and efficient medium access control in wireless ad hoc networks. **Wireless Personal Communications**, v. 113, 2020.
- MOITRA, A. **Tensor Decomposition and Their Applications**. Disponível em: https://www.youtube.com/watch?v=HcIN27_WqPU&t=8s.
- MONDAL, A. Candid with youtube: Adaptive streaming behavior and implications on data consumption. **NOSSDAV**, 2017.
- PEFKIANAKIS, I. et al. Window-based rate adaptation in 802.11n wireless networks. **Mobile Networks and Applications**, v. 18, p. 156–169, 2013.
- PEFKIANAKIS, I. et al. Mimo rate adaptation in 802.11n wireless networks. **Mobile Networks and Applications**, v. 18, p. 257–268, 2010.
- PITTONI, M. Online identification of last-mile throughput bottlenecks on home routers. **Networking and Internet Architecture**, 2016.
- SALINAS, M.-I. Home wi-fi impairments. **Networking and Internet Architecture**, 2018.
- SANCHEZ, M. A. et al. A measurement experimentation platform at the internet's edge. **IEEE/ACM Transactions on Networking**, v. 23, p. 1944–1958, 2015.
- SCHMITT, P. et al. Inferring streaming video quality from encrypted traffic: Practical models and deployment experience. **HAL-Inria**, 2019.
- SEQUEIRA, L. et al. The utility of characterizing the buffer of network devices in order to improve real-time interactive services. **LANC '12: Proceedings of the 7th Latin American Networking Conference**, p. 19–27, 2012.
- SEUFERT, M. et al. A survey on quality of experience of http adaptive streaming. **IEEE Communications Surveys Tutorials**, 2014.
- SEUFERT, M. et al. Youtube qoe on mobile devices: Subjective analysis of classical vs. adaptive video streaming. **IWCMC**, 2015.
- SHEN, W.-L. et al. Rate adaptation for 802.11 multiuser mimo networks. **IEEE Transactions on Mobile Computing**, v. 13, p. 35–47, 2014.
- SIEBER, C. The cost of aggressive http adaptive streaming: Quantifying youtube's redundant traffic. **IFIP/IEEE IM**, 2015.
- SIEBER, C. et al. Sacrificing efficiency for quality of experience: Youtube's redundant traffic behavior. **IFIP**, 2016.
- SILVA, E. d. S. e.; LEÃO, R.; CAMPOS, S. Performance issues of multimedia applications. **Lecture Notes in Computer Science**, p. 1031–1047, 2002.
- SILVEIRA, F.; SILVA, E. d. S. e. Predicting packet loss statistics with hidden markov models for fec control. **Computer Networks**, v. 56, p. 628–641, 2012.

- SKORDOULIS, D. et al. Ieee 802.11n mac frame aggregation mechanisms for next-generation high-throughput wlans. **IEEE Wireless Communications**, v. 15, p. 40–47, 2008.
- SODAGAR, I. The mpeg-dash standard for multimedia streaming over the internet. **IEEE MultiMedia**, v. 18, p. 60–67, 2011.
- SUNDARESAN, S. et al. Measuring home broadband performance. **Communications of the ACM**, v. 55, p. 100–109, 2012.
- SUNDARESAN, S.; FEAMSTER, N.; TEIXEIRA, R. Home network or access link? locating last-mile downstream throughput bottlenecks. **PAM 2016 - Passive and Active Measurement Conference**, Heraklion, p. 111–123, 2016.
- TSILIMANTOS, D.; KARAGKIOULES, T.; VALENTIN, S. Classifying flows and buffer state for youtube’s http adaptive streaming service in mobile networks. **ACM MMSys**, p. 138–149, 2018.
- VU, V.-H.; MASHAL, I.; CHUNG, T.-Y. A novel bandwidth estimation method based on macd for dash. **KSII Transactions on Internet and Information Systems**, v. 11, n. 3, p. 1441–1461, 2017.
- WEYULU, E.; HANADA, M.; KIM, M. W. Optimizing rts/cts to improve throughput in ad hoc wlans. **2017 Federated Conference on Computer Science and Information Systems (FedCSIS)**, Prague, 2017.
- WIKIPEDIA. **Youtube**. 2020. Disponível em: <https://en.wikipedia.org/wiki/YouTube>.
- XIAOQI, Y. et al. A control-theoretic approach for dynamic adaptive video streaming over http. **SIGCOMM '15: Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication**, p. 325–338, 2015.
- YIN, X. et al. A control-theoretic approach for dynamic adaptive video streaming over http. **SIGCOMM**, 2015.

APÊNDICES

APÊNDICE A – CONFIGURAÇÃO DO ROTEADOR

A.1 CONFIGURANDO O OPENWRT E WIFI

Para realizar os experimentos foi necessário se familiarizar com o roteador em mãos e o sistema operacional do mesmo. O roteador usado é um Archer C20 AC750 v4, da TP-Link. O sistema operacional instalado no mesmo é o OpenWrt, o mesmo que está presente nos roteadores dos clientes da Gigalink, o ISP de médio porte com o qual o LAND tem uma parceria. OpenWrt possui código aberto e uma comunidade muito ativa, com diversos tutoriais e páginas dedicadas às suas funcionalidades e possíveis configurações. O driver de WiFi aberto, mt76, é usado para essa configuração do roteador.

Para os primeiros experimentos a última versão do OpenWrt foi clonada do repositório git oficial. Isso pode ser feito com o seguinte comando:

```
git clone https://github.com/openwrt/openwrt.git. Deve-se então entrar no diretório, com o comando cd openwrt e então executar o seguinte comando:
```

```
./scripts/feeds update -a & ./scripts/feeds install -a. Após feito isso, deve-se configurar que versão de imagem iremos querer montar. Para isso, basta rodar o comando make menuconfig. Isso abrirá uma janela onde você deverá especificar a imagem. Para o roteador que foi usado nesse experimento deve-se escolher “Mediatek Ralink MIPS” na opção “Target System”, “MT76x8 based boards” na opção “Subtarget” e “TP-Link Archer C20 v4” na opção “Target Profile”. Uma vez feito isso deve-se selecionar a opção "Exit" e depois "Yes". Para montar a imagem basta rodar o comando make. Uma vez que a imagem foi montada ela se encontrará no diretório ./bin/targets/ramips/mt76x8/.
```

Para instalar a imagem no roteador usado nesse experimento é preciso usar o protocolo tftp. Primeiro, deve-se instalar o servidor tftp na máquina onde a imagem está armazenada. No caso dos experimentos essa máquina possui o sistema operacional Linux Fedora, logo, o comando a ser rodado é `dnf install tftp-server tftp -y`. Em seguida, deve-se recarregar o daemon systemd com o comando `systemctl daemon-reload` e então habilitar o servidor com o comando `systemctl enable --now tftp-server`. Por último deve-se configurar o firewall para permitir tráfego tftp com o comando `firewall-cmd --add-service=tftp --perm \&\& firewall-cmd --reload`.

Uma vez configurado o servidor tftp é possível enviar a imagem montada do computador para o roteador. Para fazer isso, deve-se copiar a imagem desejada para a área de transferência do tftp, renomeando-a, da seguinte forma:

```
cp /caminho/ate/imagem/openwrt-ramips-mt76x8-tplink_archer-c20-v4-squashfs-tftp-recovery-default.bin /var/lib/tftpboot/tp_recovery.bin. Em seguida, deve-se conectar o roteador, ainda desligado, ao computador através de um cabo ethernet. Deve-se mudar o ip da interface ethernet para 192.168.0.66. No computa-
```

dor que foi usado essa interface se chama `enp3s0`, logo o seguinte comando foi usado: `sudo ifconfig enp3s0 192.168.0.66`. Para finalizar basta segurar o botão de reset do roteador e apertar o botão de ligar, mantendo o botão de reset segurado entre 7 a 10 segundos. O bootloader do roteador então buscará a imagem no ip que foi estaticamente definido e então o firmware será baixado e escrito para a memória flash do roteador, que reiniciará e estará pronto para ser usado. Para confirmar que o firmware foi transferido o seguinte comando foi rodado: `tcpdump -vv -X port 69 -i enp3s0`, já que o protocolo age na porta 69.

Após o firmware ter sido instalado deve-se configurar um ip estático para a máquina que estiver usando com a seguinte máscara: `192.168.1.0/24`. Deve-se então acessar o roteador rodando o seguinte comando: `ssh root@192.168.1.1`. Uma senha pode então ser definida rodando o comando `passwd`. Agora, é necessário configurar o WiFi. Primeiro é necessário atualizar o gerenciador de pacotes rodando o seguinte comando: `opkg update`. Feito isso é necessário instalar o pacote `luci` com criptografia, para acessar a interface gráfica, rodando o seguinte comando: `opkg install luci-ssl-openssl`. Uma vez feito isso, deve-se acessar, pelo browser, o ip `192.168.1.1` e fazer o login com a senha para o root. Isso abrirá a página correspondente ao roteador, onde o WiFi poderá ser configurado. Para fazê-lo, deve-se acessar a seção “Network -> Wireless” e então selecionar o botão de “edit” do ssid “openwrt”. O modo WiFi que foi usado nos experimentos foi “n”, logo, essa opção foi escolhida no menu dropdown de “mode”. Na seção “Wireless Security” foi escolhido o protocolo WPA2-PSK. Uma chave também deve ser escolhida, por questões de segurança. Feito isso, deve-se clicar em “Save and Apply”, o que habilitará o WiFi.

A.2 MODIFICANDO O TAMANHO DO BUFFER

O trabalho consiste em entender como a QoE do usuário é afetada pelo tamanho dos buffers do driver de WiFi do roteador, porém, ainda não foi abordado como modificar o tamanho desses buffers. O tamanho desses buffers deve ser modificado antes da montagem da imagem, e, para os experimentos, somente o buffer downstream foi modificado, ou seja, o buffer responsável pelo tráfego que “entra na casa”. Cinco tamanhos foram testados para esse buffer: 10, 20, 30, 40 e 256 (tamanho padrão) pacotes.

O driver de WiFi se chama `mt76` e existe um diretório dedicado para ele na árvore de diretórios que foi clonado do git. Para encontrar esse diretório basta entrar no diretório `openwrt` que foi clonado e de lá rodar o comando `cd package/kernel/mt76`. É possível então ver que existe um Makefile que define diversas coisas, entre elas, a fonte do pacote, que é indicada pela variável `PKG_SOURCE_URL`. O valor dessa variável é o repositório do git do openwrt que possui o código fonte para o driver de WiFi: <https://github.com/openwrt/mt76>. Ao seguir esse link pode-se ver que existe um arquivo chamado `mt76.h`. Nesse arquivo existem duas variáveis que estão definidas logo

no início: `MT_TX_RING_SIZE` e `MT_RX_BUF_SIZE`. Essas variáveis guardam os tamanhos dos buffers downstream e upstream, respectivamente, com valores inteiros de 256 e 2048, respectivamente, com essas unidades correspondendo a quantidade de pacotes.

Para mudar o tamanho do buffer downstream o seguinte procedimento foi feito: o repositório `mt76` do `openwrt` foi clonado e o tamanho do buffer foi modificado no arquivo `mt76.h`. Após feito isso um repositório privado foi criado na minha conta pessoal do github, a árvore de diretórios `mt76` foi enviada para o repositório, a variável `PKG_SOURCE_URL` no Makefile do driver foi modificada para conter o meu repositório ao invés do oficial e então a imagem foi montada, seguindo os mesmos passos feitos anteriormente. Para cada tamanho de buffer o mesmo procedimento foi feito, porém somente o arquivo `mt76.h` foi modificado e atualizado no repositório privado após cada mudança. Feito isso para cada um dos tamanhos de buffer cinco imagens distintas foram geradas, uma para cada tamanho de buffer. Quando necessário, cada uma foi instalada no roteador para os experimentos, novamente seguindo os mesmos passos feitos anteriormente.

APÊNDICE B – PROVA DE QUE OS AUTOVETORES DA MATRIZ DE COVARIÂNCIA EQUIVALEM À DIREÇÃO DE MAIOR VARIÂNCIA

Seja X a matriz com os valores das métricas de QoE, onde x_{ij} é o valor da j -ésima métrica de QoE na i -ésima reprodução, $i = 1, 2, \dots, n$, $j = 1, 2, \dots, d$. Suponhamos também que a matriz X corresponda aos valores após as amostras já terem sido centradas no zero e normalizadas. Seja v um vetor qualquer de dimensão d . A projeção de x_i em v é dada pelo produto interno entre os dois, que equivale a $\sum_{j=1}^d x_{ij}v_j$. Logo, aplicando a definição de variância às projeções temos:

$$Var = \frac{1}{n} \sum_{i=1}^n \left(\sum_{j=1}^d x_{ij}v_j - \mu \right)^2, \quad (\text{B.1})$$

onde Var é a variância e μ é a média entre as projeções de x_i em v , para todo i . Logo,

$$\mu = \frac{1}{n} \sum_{i=1}^n \left(\sum_{j=1}^d x_{ij}v_j \right). \quad (\text{B.2})$$

Como os somatórios são finitos, podemos trocá-los de ordem, obtendo

$$\mu = \sum_{j=1}^d v_j \left(\frac{1}{n} \sum_{i=1}^n x_{ij} \right). \quad (\text{B.3})$$

Sabemos que $\frac{1}{n} \sum_{i=1}^n x_{ij}$ é a média amostral da variável j , que foi centrada no zero, logo,

$$\mu = 0. \quad (\text{B.4})$$

Uma forma de interpretar esse resultado é que a média dos valores projetados é equivalente à projeção da média. Temos então que

$$Var = \frac{1}{n} \sum_{i=1}^n \left(\sum_{j=1}^d x_{ij}v_j \right)^2. \quad (\text{B.5})$$

Agora, o objetivo é encontrar o vetor v que maximiza Var . Para isso é necessário adicionar uma restrição ao tamanho de v , se não a forma trivial de maximizar a equação é ficar aumentando irrestritamente o tamanho de v . A forma padronizada de fazer isso é definir que $\|v\| = 1$. Isso pode ser estabelecido na própria equação a ser otimizada, usando multiplicadores de Lagrange, da seguinte forma:

$$Var = \frac{1}{n} \sum_{i=1}^n \left(\sum_{j=1}^d x_{ij}v_j \right)^2 - \lambda \left[\left(\sum_{j=1}^d v_j^2 \right) - 1 \right]. \quad (\text{B.6})$$

Para maximizar Var , será aplicada a derivada parcial de Var em relação a componentes de v e isso será igualado a zero. Pela regra da cadeia obtemos:

$$\frac{\partial Var}{\partial v_a} = \frac{2}{n} \sum_{i=1}^n \left(\sum_{j=1}^d x_{ij}v_j \right) x_{ia} - 2\lambda v_a = 0. \quad (\text{B.7})$$

Como os dois somatórios são finitos, podemos trocar a ordem deles, obtendo a seguinte expressão:

$$2 \sum_{j=1}^d v_j \left(\frac{1}{n} \sum_{i=1}^n x_{ia} x_{ij} \right) = 2\lambda v_a. \quad (\text{B.8})$$

Entretanto,

$$\frac{1}{n} \sum_{i=1}^n x_{ia} x_{ij} = \text{cov}(a, j). \quad (\text{B.9})$$

Logo,

$$\sum_{j=1}^d \text{cov}(a, j) v_j = \lambda v_a. \quad (\text{B.10})$$

Para cada variável a terá uma equação dessas correspondente, $a = 1, 2, \dots, d$:

$$\begin{cases} \sum_{j=1}^d \text{cov}(1, j) v_j = \lambda v_1 \\ \sum_{j=1}^d \text{cov}(2, j) v_j = \lambda v_2 \\ \vdots \\ \sum_{j=1}^d \text{cov}(d, j) v_j = \lambda v_d. \end{cases} \quad (\text{B.11})$$

Cada uma dessas equações equivale ao produto interno entre uma linha da matriz de covariância Σ e v . Em notação matricial, isso é equivalente a

$$\Sigma v = \lambda v. \quad (\text{B.12})$$

Pela equação podemos ver que v é autovetor de Σ , ou seja, o vetor que dá a direção de maior variância da projeção de X nos autovetores da matriz de covariância é um dos próprios autovetores dessa matriz de covariância, como queríamos demonstrar. Argumento análogo pode ser usado para estender essa prova para os outros autovetores.

APÊNDICE C – PROVA DE QUE OS AUTOVALORES DA MATRIZ DE COVARIÂNCIA CORRESPONDEM À VARIÂNCIA NO EIXO DOS AUTOVETORES CORRESPONDENTES

Assim como feito na seção B do Apêndice, vamos começar aplicando a definição de variância das projeções. Temos então que

$$Var = \frac{1}{n} \sum_{i=1}^n \left(\sum_{j=1}^d x_{ij} v_j - \mu \right)^2. \quad (C.1)$$

Como foi provado na seção B do Apêndice, $\mu = 0$, logo,

$$Var = \frac{1}{n} \sum_{i=1}^n \left(\sum_{j=1}^d x_{ij} v_j \right)^2 \quad (C.2)$$

$$= \frac{1}{n} \sum_{i=1}^n \left(\sum_{j=1}^d x_{ij} v_j \right) \left(\sum_{a=1}^d x_{ia} v_a \right) \quad (C.3)$$

$$= \sum_{a=1}^d \sum_{j=1}^d \left(\frac{1}{n} \sum_{i=1}^n x_{ia} x_{ij} \right) v_j v_a \quad (C.4)$$

$$= \sum_{a=1}^d \left(\sum_{j=1}^d cov(a, j) v_j \right) v_a \quad (C.5)$$

Como v é um autovetor de Σ , nossa matriz de correlação, temos que $\sum_{j=1}^d cov(a, j) v_j$ é equivale ao produto interno da a -ésima linha de Σ com o vetor v , o que resulta em λv_a , logo

$$Var = \sum_{a=1}^d (\lambda v_a) v_a \quad (C.6)$$

$$= \lambda \|v\|^2. \quad (C.7)$$

Então

$$Var = \lambda, \quad (C.8)$$

como queríamos demonstrar. Argumento análogo pode ser usado para estender a prova para os outros autovalores.

Supõe-se que o leitor tenha lido anteriormente a prova na seção B do Apêndice, que possui a explicação para passos análogos aos que foram feitos nessa prova.

**APÊNDICE D – PROVA DE QUE MAXIMIZAR A VARIÂNCIA NO
ESPAÇO DE PROJEÇÃO EQUIVALE A MINIMIZAR O ERRO DE
PROJEÇÃO**

Assim como feito na seção B do Apêndice, começaremos aplicando a definição da variância no espaço das projeções definido por v . Temos então que

$$Var = \frac{1}{n} \sum_{i=1}^n (x_i \cdot v)^2, \quad (D.1)$$

onde x_i é o vetor linha da matriz de dados X , após centralização no zero e \cdot é o operador de produto interno.

O erro de projeção é definido como a soma do quadrado das distâncias dos pontos até o eixo de v . Usando a relação de pitágoras, é possível definir o erro como

$$Erro = \sum_{i=1}^n [||x_i||^2 - (x_i \cdot v)^2]. \quad (D.2)$$

Essa igualdade se dá pelo fato do eixo do erro de projeção ser perpendicular ao eixo de projeção.

Como $\frac{1}{n}$ é uma constante não precisa ser levada em conta na hora da maximização. Similarmente, $||x_i||^2$ também é uma constante, logo não precisa ser levada em conta na hora da minimização. Assim, temos que maximizar a variância equivale a maximizar a equação

$$\sum_{i=1}^n (x_i \cdot v)^2 \quad (D.3)$$

e minimizar o erro equivale a minimizar a equação

$$\sum_{i=1}^n -(x_i \cdot v)^2. \quad (D.4)$$

Porém, maximizar a equação D.3 equivale a minimizar a equação D.4, logo, maximizar a variância no eixo de projeção é equivalente a minimizar o erro de projeção, como queríamos demonstrar.