Adaptive Streaming: From Bitrate Maximization to Rate-Distortion Optimization

by

Zhengfang Duanmu

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2021

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Xiao-Ping Zhang Professor Dept. of Electrical, Computer & Biomedical Engineering Ryerson University
Supervisor(s):	Zhou Wang Professor, Dept. of Electrical & Computer Engineering University of Waterloo
Internal Member:	Otman Basir Professor, Dept. of Electrical & Computer Engineering University of Waterloo
Internal Member:	Oleg Michailovich Associate Professor, Dept. of Electrical & Computer Engineering University of Waterloo
Internal-External Member:	Edward R. Vrscay Professor, Dept. of Applied Mathematics University of Waterloo

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The fundamental conflict between the increasing consumer demand for better Quality-of-Experience (QoE) and the limited supply of network resources has become significant challenges to modern video delivery systems. State-of-the-art Adaptive Bitrate (ABR) streaming algorithms are dedicated to drain available bandwidth in hope to improve viewers' QoE, resulting in inefficient use of network resources. In this thesis, we develop an alternative design paradigm, namely Rate-Distortion Optimized Streaming (RDOS), to balance the contrast demands from video consumers and service providers. Distinct from the traditional bitrate maximization paradigm, RDOS must operate at any given point along the rate-distortion curve, as specified by a trade-off parameter. The new paradigm has found plausible explanations in information theory, economics, and visual perception.

To instantiate the new philosophy, we decompose adaptive streaming algorithms into three mutually independent components, including throughput predictor, reward function, and bitrate selector. We provide a unified framework to understand the connections among all existing ABR algorithms. The new perspective also illustrates the fundamental limitations of each algorithm by going behind its underlying assumptions. Based on the insights, we propose novel improvements to each of the three functional components.

To alleviate a series of unrealistic assumptions behind bitrate-based QoE models, we develop a theoretically-grounded objective QoE model. The new objective QoE model combines the information from subject-rated streaming videos and the prior knowledge about Human Visual System (HVS) in a principled way. By analyzing a corpus of psychophysical experiments, we show the QoE function estimation can be formulated as a projection onto convex sets problem. The proposed model presents strong generalization capability over a broad range of source contents, video encoders, and viewing conditions. Most importantly, the QoE model disentangles bitrate with quality, making it an ideal component in the RDOS framework.

In contrast to the existing throughput estimators that approximate the marginal probability distribution over all connections, we optimize the throughput predictor conditioned on each client. Although there are lack of training data for each Internet Protocol connection, we can leverage the latest advances in meta learning to incorporate the knowledge

embedded in similar tasks. With a deliberately designed objective function, the algorithm learns to identify similar structures among different network characteristics from millions of realistic throughput traces. During the test phase, the model can quickly adapt to connection-level network characteristics with only a small amount of training data from novel streaming video clients with a small number of gradient steps.

The enormous space of streaming videos, constantly progressing encoding schemes, and great diversity of throughput characteristics make it extremely challenging for modern data-driven bitrate selectors that are trained with limited samples to generalize well. To this end, we propose a Bayesian bitrate selection algorithm by adaptively fusing an online, robust, and short-term optimal controller with an offline, susceptible, and long-term optimal planner. Depending on the reliability of the two controllers in certain system states, the algorithm dynamically prioritizes the one of the two decision rules to obtain the optimal decision.

To faithfully evaluate the performance of RDOS, we construct a large-scale streaming video dataset – the Waterloo Streaming Video database. It contains a wide variety of high quality source contents, encoders, encoding profiles, realistic throughput traces, and viewing devices. Extensive objective evaluation demonstrates the proposed algorithm can deliver identical QoE to state-of-the-art ABR algorithms at a much lower cost. The improvement is also supported by so-far the largest subjective video quality assessment experiment.

Acknowledgements

Pursuing a Ph.D is an adventure, and this journey would not have been as fulfilling and rewarding without the guidance and the support of many people.

First and foremost, I would like to thank my supervisor Professor Zhou Wang. Prof. Wang is an amazing teacher and has had great influence on me. He not only taught me how to think big but also gave me very detailed suggestions on my research topics. More importantly, Prof. Wang has taught me to appreciate the beauty of simple designs, the intuition and philosophy behind the details. Despite the extraordinary achievement, he always stays humble and foolish. He is able to keep me motivated through the difficult times. I feel tremendously lucky and privileged to have had the opportunity to work with Prof. Wang.

I am honored to have Dr. Xiao-Ping Zhang, Dr. Oleg Michailovich, Dr. Otman Basir and Dr. Edward R. Vrscay on my thesis committee, and thank them for the time they took to review my thesis and for giving valuable advice and suggestions to further improve my work. In particular, Prof. Vrscay introduced me to applied functional analysis, which has become a central component in some of my research projects. I was also privileged to take the Inverse Problem course with Prof. Michailovich, where I was taught how to approach a problem with a Bayesian perspective.

I have had tremendous fun working with the Image and Vision Computing Group at University of Waterloo, and enjoyed many lively discussions in the office. I would like to thank Kede Ma, who in many ways, has been my mentor in my early research career; Wentao Liu, who motivated me to develop mathematical thinking; Zhuoran Li, Zhongling Wang, Jinghan Zhou, and Xinyu Guo who spent tons of time to my projects and helped me a lot as collaborators and friends; Diqi Chen, who introduced me to reinforcement learning; Jiheng Wang, who gave constructive suggestions on research; Abdul Rehman and Kai Zeng, who bootstrapped me at the early stage of my adaptive streaming research. I would also like to thank other past and present members of the Image and Vision Computing Lab: Shiqi Wang, Hojatollah Yehaneh, Shahrukh Athar, Jinghan Zhou, Xinyu Guo, Xiaoyu Tang, Honglei Su, Qi Liu, Wei Zhou, Delu Zeng, Yu Wang, Rasoul Mohammadi Nasiri, Kaiwen Ye, Xiongkuo Min, and Qingbo Wu. I am truly honored to be part of this family.

I am grateful to Natural Sciences and Engineering Research Council of Canada, who has supported my research over the last five years. The scholarship provides me an opportunity to fully focus on my research in visual communication. I would also like to express my gratitude to University of Waterloo for all the opportunities that they have given me over the years.

I would like to thank my friends outside the lab who had a positive impact on my life and made my Ph.D journey enjoyable: Yijie Liang, Lingyun Li, Ruiyi Pan, Chaojie Ou, Xin Wei, Weiya Ye, Jinglingzhang Liu, Mengge Chen, Qiang Zhao and many others.

Last but not least, I would like to thank my parents, Youmei Xu and Gang Duanmu, for their love and support. Although my dad left me during my Ph.D journey, he has always been my motivation to be a better man. My parents have made countless sacrifices for me, and have provided me with steady guidance and encouragement. This dissertation is dedicated to them.

Dedication

This is dedicated to Mom and Dad.

Table of Contents

List of Figures xiv

List of Tables xxii

List of Abbreviations xxvi

List of Symbols xxxi

1 Introduction 1

1.1 Background and Motivation 2

1.1.1 Dynamic Adaptive Streaming over HTTP 2

1.1.2 Functional Decomposition 4

1.1.3 Challenges in Adaptive Streaming 6

1.1.4 Trends in Multimedia Communication 8

1.2 Objectives 9

1.3 Thesis Overview 10

1.3.1 Organizational Themes 10

1.3.2 Dissertation Roadmap 12

1.3.3 Chapter Descriptions 13

2 Literature Review **15**

2.1 Reward Function 15

2.1.1 Subjective QoE Assessment 16

2.1.2 Objective QoE Assessment Models for Streaming Videos 23

2.2 Throughput Prediction 39

2.2.1 Bayesian View of Throughput Prediction 43

2.2.2 Existing Throughput Prediction Models 44

2.3 Bitrate Selector 54

2.3.1 Bayesian View of Bitrate Selector 57

2.3.2 Existing Bitrate Selectors 60

2.4 Overview of Adaptive Bitrate Algorithms 72

2.5 Validation of Adaptive Bitrate Algorithms 75

3 Adaptive Streaming: From Bitrate Maximization to Rate-Distortion Optimization **82**

3.1 Rate-Distortion Optimized Adaptive Streaming 85

3.2 Why RDOS? 87

3.3 Economic Interpretation 88

3.4 Connections to Bitrate Maximization Scheme 91

3.5 Summary 92

4 A Bayesian Quality-of-Experience Model for Adaptive Streaming Videos **94**

4.1 Prior QoE Model 96

4.1.1 Deterministic View 96

4.1.2 Probabilistic View 98

4.2 A Bayesian QoE Model . . . 99
4.2.1 Additional Assumptions . . . 99
4.2.2 Parameterization . . . 100
4.2.3 Model Training . . . 100
4.3 Experiments . . . 102
4.3.1 Experimental Setup . . . 102
4.3.2 Performance . . . 106
4.3.3 Best-case Validation . . . 106
4.3.4 Statistical Significance Test . . . 110
4.3.5 Ablation Experiment . . . 111
4.3.6 Impact of Step Sizes . . . 113
4.3.7 Impact of α . . . 114
4.4 Summary . . . 114

5 Delving into Connection-Level Throughput Prediction using Meta Learning 116

5.1 Motivation . . . 116
5.2 Meta Learning-based Throughput Prediction . . . 121
5.2.1 Problem Setup . . . 121
5.2.2 Learning Algorithm . . . 123
5.2.3 Bayesian Interpretation . . . 125
5.2.4 Implementation Details . . . 127
5.3 Evaluation . . . 128
5.3.1 MetaTP vs. Existing Throughput Predictors . . . 129
5.3.2 Ablation Experiment . . . 134

5.3.3 Continual Learning . . . 137
5.3.4 Discussion . . . 139
5.4 Summary . . . 140

6 EERO: Towards An Efficient, Robust, and Optimal Bitrate Selector 141
6.1 A Bayesian Framework for Bitrate Selection . . . 141
6.1.1 The Bayesian Interpretation of Existing Arts . . . 144
6.2 Towards Maximum A Posteriori Bitrate Selector . . . 150
6.2.1 Likelihood Function . . . 152
6.2.2 Prior Model . . . 153
6.2.3 Implementation Details . . . 155
6.2.4 Why EERO? . . . 159
6.3 Evaluation . . . 160
6.3.1 EERO vs. Existing Bitrate Selectors . . . 161
6.3.2 Ablation Experiment . . . 172
6.3.3 Discussion . . . 177
6.4 Summary . . . 179

7 System Validation 181
7.1 Constructing the Waterloo Streaming Video Database . . . 181
7.1.1 Transmitter . . . 182
7.1.2 Channel . . . 186
7.1.3 Receiver . . . 187
7.2 Objective Evaluation . . . 189
7.2.1 Experimental Setup . . . 189

7.2.2 Rate Distortion Optimization Paradigm vs. Bitrate Maximization Paradigm . . . 191
7.2.3 Full System Validation . . . 195
7.3 Subjective Evaluation . . . 200
7.3.1 Database Construction . . . 200
7.3.2 Subjective Testing . . . 204
7.3.3 Performance of ABR Algorithms . . . 207

8 Conclusion **212**

References **216**

A Detailed Experimental Setup of Meta Learning-Based Throughput Predictor **242**

A.1 Multi-Layer Perceptron Architecture . . . 242

B Construction of the Waterloo Streaming Video Database **244**

B.1 Encoding Configurations . . . 244
B.2 Packaging . . . 244
B.3 Viewing Device . . . 247

List of Figures

1.1 An overview of HTTP adaptive video streaming. 2

1.2 System diagram of adaptive bitrate streaming player. 3

1.3 The schematic diagram of the proposed framework. 5

1.4 Dissertation roadmap. 13

2.1 Samples of generalized rate-distortion surfaces for different video content. . 18

2.2 Knowledge map of objective QoE. 27

2.3 Graphical model representation of existing QoE models. The box is "plate" representing replicates. Each node represents a random variable (or group of random variables), and the links express probabilistic relationships between these variables. The observable variables are shaded in color. 30

2.4 There exists significant variation on the characteristics of streaming videos, evident by the distributions of (a) VMAF, (b) rebuffering duration, and (c) adaptation magnitude in six publicly available datasets. 38

2.5 Video streaming clients experience highly variable end-to-end throughput. . 40

2.6 The time-varying throughput is a multi-modal stochastic process. 41

2.7 The time-varying throughput is a multi-modal stochastic process. 42

2.8 Joint distributions of throughput values separated by three different temporal distances. 45

2.9 Graphical model representation of most existing throughput prediction models. The box is "plate" representing replicates. Each node represents a random variable (or group of random variables), and the links express probabilistic relationships between these variables. The observable variables are shaded in color. 46

2.10 Auto-correlation function of network throughput. 48

2.11 Graphical model representation of the hidden Markov model-based throughput generation process. The box is "plate" representing replicates. Each node represents a random variable (or group of random variables), and the links express probabilistic relationships between these variables. The observable variables are shaded in color. 52

2.12 The design tradeoff for the bitrate adaptation function. In general, ABR algorithms cannot achieve efficiency in computation time, robustness to unobserved states, and optimality in expected total reward simultaneously. . 56

2.13 The computation time of dynamic programming and exhaustive search in a typical adaptive streaming scenario. 59

2.14 Graphical model representation of the value-based bitrate selection process. The box is "plate" representing replicates. Each node represents a random variable (or group of random variables), and the links express probabilistic relationships between these variables. The observable variables are shaded in color. $\mathbf{s}$ encodes the current state observation (such as the past throughput histories and current buffer occupancy). $\mathbf{s}^+$ represents the states required to make optimal offline decisions with the cumulative reward q, including the future throughput prediction $\mathbf{c}$ and the future chunk statistics $\mathbf{v}$. The optimal action a^* is obtained by selecting the action that optimizes the cumulative reward. 61

2.15 Graphical model representation of the policy-based bitrate selection process. The box is "plate" representing replicates. Each node represents a random variable (or group of random variables), and the links express probabilistic relationships between these variables. The observable variables are shaded in color. $\mathbf{s}$ encodes the current state observation (such as the past throughput histories and current buffer occupancy). $\mathbf{s}^+$ represents the states required to make optimal offline decisions with the cumulative reward q, including the future throughput prediction $\mathbf{c}$ and the future chunk statistics $\mathbf{v}$. The optimal action a^* is obtained by selecting the action that optimizes the cumulative reward. The model-free reinforcement learning models do not explicitly predicts the future environment state $\mathbf{s}^+$ in the estimation of q. . 66

2.16 A reinforcement learning framework for bitrate adaptation. 67

2.17 The schematic diagram of the efficiency-robustness-optimality tradeoff of the existing bitrate selectors. 73

3.1 The relationship between bitrate and a perceptually motivated video quality model on (a) different video contents, (b) different encoding spatial resolutions, and (c) different video encoders and viewing devices. 83

3.2 Illustration of bandwidth sharing scenarios. (a) bitrate maximization paradigm. (b) rate-distortion optimization paradigm. 85

3.3 The illustration of the equilibrium "price" in the virtual market. 89

3.4 The illustration of the physical market between customers and video service providers, and the virtual market between the service providers and ABR agents. 91

3.5 The illustration of the the QoE market under the bitrate maximization scheme. 92

4.1 Pairwise comparison matrix $\mathbf{R}$. Each entry indicates the subjective preference of the row model against the column model. $\mathbf{R} - \mathbf{R}^T$ are drawn here for better visibility. 109

4.2 Global ranking results of the four QoE models. 110

4.3 Performance of Bayesian Streaming Quality Index (BSQI) with different number of bins. 113

4.4 Performance of BSQI with different α. 114

5.1 Approximating of two instance-level Bernoulli distributions (first column) using the standard supervised learning (second column), fine tuning with pretraining (third column), fine tuning with random initialization (fourth column), and meta learning (fifth column). Row 1 and row 2 illustrate the conditional distribution for the first class and the second class, respectively. Fine tuning from marginal distribution, random initialization, and meta learning-based prior are performed for five gradient steps. 117

5.2 Approximating two sub-population Markov processes (first column) using the standard supervised learning (second column), fine tuning with pretraining (third column), fine tuning with random initialization (fourth column), and meta learning (fifth column). Row 1 and row 2 illustrate the conditional distribution for the first class and the second class, respectively. Fine tuning from marginal distribution, random initialization, and meta learning-based prior are performed for five gradient steps. 119

5.3 Approximating two sub-population realistic throughput distributions (first column) using the standard supervised learning (second column), fine tuning with pretraining (third column), fine tuning with random initialization (fourth column), and meta learning (fifth column). Row 1 and row 2 illustrate the conditional distribution for the first class and the second class, respectively. Fine tuning from marginal distribution, random initialization, and meta learning-based prior are performed for five gradient steps. 120

5.4 The proposed meta learning-based throughput prediction framework. . . . 122

5.5 The computational graph of the meta learning algorithm. Straight arrows, crooked arrows, and plates denote deterministic computations, sampling operations, and repeated computations, respectively. 123

5.6 Illustration of the neural network configurations for MetaTP. We denote the parameterization of the LSTM and fully connected layer as "input channel × output channel". 127

5.7 The optimality and robustness scores in BSQI of the sub-components of EERO. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average. 133

5.8 The optimality and robustness scores in BSQI of the sub-components of EERO. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average. 134

5.9 The performance of MetaTP with other network architectures. 135

5.10 The performance of MetaTP with additional gradient steps. 137

5.11 The performance of MetaTP with additional training samples. 138

5.12 The performance of MetaTP with different portion of training samples. . . 139

6.1 The data-driven ABR algorithm Pensieve is susceptible to unobserved state-action pairs. The model predictive control-based algorithm Fugu achieves sub-optimal performance but is quite robust to environment shift. Case 1: Pensieve observed the encoding profile of video 1 in the training process. Case 2: Pensieve observed the encoding profile of video 2 in the training process. 149

6.2 The proposed EERO bitrate selection algorithm. (a) The ABR agent traverses the tree and collects the instantaneous reward along the path. At each terminal node of the look-ahead horizon, EERO may optionally estimate the future cumulative reward with a value function. (b) The algorithm estimates the optimal cumulative reward function $\tilde{q}$ based on the simulation. (c) In the end, EERO combines the observation at the test time $\tilde{q}$ and the prior estimate $\mu_{\boldsymbol{\theta}}$. 152

6.3 Illustration of the neural network configurations for EERO. We denote the parameterization of the fully connected "fc" layer as "input channel × output channel". 156

6.4 The schematic diagram of the efficiency-robustness-optimality tradeoff of bitrate selectors. EERO integrates the robustness of MPC and the optimality of reinforcement learning-based algorithms, thereby achieving a better efficiency-robustness-optimality tradeoff. Abbreviations: MPC, model predictive control; RL, reinforcement learning; IL, imitation learning. 159

6.5 Computation time of ABR algorithms. 168

6.6 The optimality performance of EERO and existing ABR algorithms in terms of three QoE measures. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average. 169

6.7 The optimality score of ABR algorithms on different throughput datasets. Results are given in terms of BSQI. 170

6.8 The optimality score of ABR algorithms with different encoding profiles. Results are given in terms of BSQI. 170

6.9 The robustness performance of EERO and existing ABR algorithms in terms of three QoE measures. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average. 171

6.10 The robustness score of ABR algorithms on different throughput datasets. Results are given in terms of BSQI. 172

6.11 The robustness score of ABR algorithms with different encoding profiles. Results are given in terms of BSQI. 172

6.12 The optimality and robustness scores in BSQI of the sub-components of EERO. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average. 173

6.13 The optimality and robustness scores in BSQI of EERO with different likelihood functions. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average. 174

6.14 The optimality and robustness scores in BSQI of EERO with different likelihood functions. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average. 175

6.15 The optimality and robustness scores in BSQI of EERO with different weighting schemes. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average. 176

6.16 The optimality and robustness scores in BSQI of EERO with only a portion of training data. Results are normalized against the performance of EERO. 179

7.1 Sample frames of source videos in the Waterloo Streaming Video database. All images are cropped for neat presentation. 182

7.2 Sample frames of H.264 encoded videos in the Waterloo Streaming Video database. All images are cropped for neat presentation. (a) Source reference video frame. (b)-(f) 1920×1080 at 8,000 kbps, 4,000 kbps 1,000 kbps, 500 kbps, and 100 kbps. (g)-(k) 1280$times$720 at 8,000 kbps, 4,000 kbps 1,000 kbps, 500 kbps, and 100 kbps. (l)-(p) 740$times$480 at 8,000 kbps, 4,000 kbps 1,000 kbps, 500 kbps, and 100 kbps. 185

7.3 Network characteristics of the three throughput trace databases. 188

7.4 The bitrate saving of rate-distortion optimized ABR algorithms over its default version. Error bars span ± one standard deviation from the average. 194

7.5 The cumulative density function of QoE change in terms of BSQI introduced by applying the RDOS reward function. 195

7.6 The bitrate saving of full RDOS over its competing algorithms. 198

7.7 The composition of rate-distortion reward obtained by the test ABR algorithms. 199

7.8 Snapshots of video sequences. 200

7.9 Network traces used in the subjective experiment. 202

7.10 Performance of ABR algorithms on each testing network trace. Results are normalized against the performance of RDOS. Error bars span ± one standard deviation from the average. 207

7.11 Performance of ABR algorithms on each content, video codec, and viewing device. Results are normalized against the performance of RDOS. Error bars span ± one standard deviation from the average. 208

7.12 Cumulative distribution function of Mean Opinion Score (MOS) generated from five competing ABR algorithms. 209

A.1 Illustration of the multi-layer perceptron configurations for MetaTP. We denote the parameterization of the fully connected layer as "input channel × output channel". 243

List of Tables

2.1 Summary of adaptive bitrate streaming evaluation studies. Abbreviations: NN, nearest neighbourhood; AM, arithmetic mean; HM, harmonic mean; EWMA, exponential weighted moving average; SVR, support vector regression; HMM, hidden Markov model; MLP, multi-layer perceptron. 54

2.2 Summary of adaptive bitrate streaming algorithms. Abbreviations: AM, arithmetic mean; HM, harmonic mean; EWMA, exponential weighted moving average; HMM, hidden markov model; MLP, multi-layer perceptron; RNN, recurrent neural network; QoE, quality-of-experience; RDO, rate distortion performance; LB, linear function of bitrate; NB, non-linear function of bitrate; LV, linear function of video quality assessment score; NV, nonlinear function of video quality assessment scores; TP, throughput prediction; BO, buffer occupancy; DH, download history; TH, throughput history; FC, information of future chunks; RL, reinforcement learning; DP, dynamic programming; SL, supervised learning; LUT, look-up table; CNN, convolutional neural network. 80

2.3 Summary of adaptive bitrate streaming evaluation studies. Abbreviations: BU, bandwidth utilization; RD, rebuffering duration; BV, bitrate variation; PSNR, peak signal-to-noise ratio; SSIM, structural similarity index; UE, user engagement; QoE, quality-of-experience; RDO, rate distortion performance. 81

4.1 Comparison of objective QoE models. Notations: r, bitrate; τ, rebuffering duration; Δr, bitrate variation; p, presentation quality measured by state-of-the-art video quality assessment methods; Δp, quality variation; s, spatial resolution. Abbreviations: QP, quantization parameter; ML, maximum likelihood; MAP, maximum a posteriori. 102

4.2 PLCC between the objective QoE model prediction and MOS on the benchmark datasets. 104

4.3 SRCC between the objective QoE model prediction and MOS on the benchmark datasets. 105

4.4 KRCC between the objective QoE model prediction and MOS on the benchmark datasets. 105

4.5 Statistical significance matrix based on F-statistics on the combination of Waterloo Streaming QoE Database-III (WaterlooSQoE-III), Waterloo Streaming QoE Database-IV (WaterlooSQoE-IV) LIVE-Netflix Video Quality of Experience Database-I (LIVE-NFLX-I), and LIVE-Netflix Video Quality of Experience Database-II (LIVE-NFLX-II) datasets. A symbol "1" means that the performance of the row model is statistically better than that of the column model, a symbol "0" means that the row model is statistically worse, and a symbol "-" means that the row and column models are statistically indistinguishable. 111

4.6 PLCC between the variants of BSQI prediction and MOS on the benchmark datasets. 112

4.7 PLCC between the variants of BSQI prediction and MOS on the benchmark datasets. 112

5.1 Quantitative results of MetaTP with different learning algorithms. 135

5.2 Quantitative results of MetaTP with different numbers of hidden states and layers. 136

6.1 Encoding Ladder of Video Sequences 162

6.2 Quantitative results of EERO with different numbers of neurons and layers in terms of BSQI. 177

6.3 Quantitative results of EERO with different quantization levels in terms of BSQI. 178

6.4 Percentage of change in BSQI of EERO at different RTT over the default model. 179

7.1 Implemented throughput predictors, reward functions, and bitrate selectors in the WaterlooSV dataset. Abbreviations: AM, arithmetic mean; HM, harmonic mean; EWMA, exponential weighted moving average; HMM, hidden markov model; MLP, multi-layer perceptron; MetaTP, meta learning-based throughput predictor; BO, buffer occupancy; LB, linear function of bitrate; NB, non-linear function of bitrate; LV, linear function of video quality assessment score; BSQI, Bayesian streaming quality index; RDOS, rate-distortion optimized streaming; Greedy/BO, greedy algorithm without taking buffer occupancy into consideration; DP, dynamic programming; SL, supervised learning; A2C, actor advantage critic; EERO, efficient, robust and optimal bitrate selector. 189

7.2 Spatial Information (SI), Temporal Information (TI), Frame Rate (FPS), and Description of Reference Videos 200

7.3 Statistical Significance Matrix Based on Wilcoxon-Statistics on the WaterlooSQoE-IV Dataset. A Symbol "1" Means That the Performance of the Row Algorithm Is Statistically Better Than That of the Column Algorithm, A Symbol "0" Means That the Row Algorithm Is Statistically Worse, and A Symbol "-" Means That the Row and Column Algorithms Are Statistically Indistinguishable 210

7.4 Percentage of bitrate saving of column model vs. row model with MOS as the QoE measure. 211

B.1 Encoder version and package. 244

B.2 Encoder configuration. 245

List of Abbreviations

A2C Advantage Actor Critic Algorithm 193

ABR Adaptive Bitrate iv, v, xv, xviii, xx, xxi, 3–9, 12–16, 19, 23, 29, 31, 36, 39–43, 48, 51, 53–58, 60, 70–79, 82–86, 88, 90–92, 104, 106–108, 110, 121, 133, 137, 139–146, 148, 150–152, 154, 156, 157, 159–161, 163–168, 170, 171, 174–176, 179–184, 186–193, 195, 197–199, 201–203, 207–215

ACR Absolute Category Scale 206

AIAD Additve Increase Additive Decrease 62

AIMD Additve Increase Multiplicative Decrease 62, 63, 196

AM Arithmetic Mean 131

BB Buffer-Based 73, 74, 164, 196, 202

BBS Bayesian Bitrate Selection 11, 12, 141, 155

BOLA Buffer-Occupancy-based Lyapunov Algorithm 70, 157, 164, 192, 193, 197

BSQI Bayesian Streaming Quality Index xvii, xxiii, 10, 11, 14, 102, 103, 106–114, 148, 155, 157, 169–171, 177, 191–193, 195, 199, 213

CDF Cumulative Density Function 132, 168

CDN Content Delivery Network 121

CNN Convolutional Neural Network 128, 165, 192

CSF Contrast Sensitivity Function 32

DASH Dynamic Adaptive Streaming over HTTP 2, 3, 18, 19, 21, 77, 78, 189, 190, 203

DQN Deep Q-Network 154, 155, 158

EERO EfficiEncy, Robustness, and Optimality 12, 141, 156, 159–161, 166–180, 195, 202, 213

EWMA Exponential Weighted Moving Average 131

FastMPC Fast Model Predictive Control 64, 65, 164, 192, 193, 197

FCC Federal Communications Commission 129, 130, 163, 186, 187

FHD Full High Resolution 165

HAS HTTP Adaptive Streaming 2, 41

HDS HTTP Dynamic Streaming 3

HDTV High Resolution Television 188, 203, 205, 247

HEVC High Efficiency Video Coding 201, 203

HLS HTTP Live Streaming 3

HM Harmonic Mean 131

HMM Hidden Markov Model 52, 53, 131, 132, 140

HSDPA High Speed Downlink Packet Access 107, 129, 186, 187

HTTP HyperText Transfer Protocol 3, 40

HVS Human Visual System iv, 8–10, 14, 18, 25, 26, 28, 31–34, 36, 39, 75, 86, 94, 95, 99, 106, 114, 158, 213

IP Internet Protocol 43, 118, 119, 129, 138, 214

ITU-R International Telecommunication Unit-Recommendation 108, 205, 206

KRCC Kendall Rank Correlation Coefficient 104, 106

LIVE-NFLX-I LIVE-Netflix Video Quality of Experience Database-I xxiii, 103, 104, 111

LIVE-NFLX-II LIVE-Netflix Video Quality of Experience Database-II xxiii, 38, 103, 111, 113

LR Linear Regression 131

LSTM Long Short-Term Memory 37, 127, 136

MAML Model Agnostic Meta Learning 123–126, 128, 134, 136

MetaTP Meta Learning-based Throughput Predictor 116, 120, 121, 125, 127–129, 132–140, 190, 195, 202, 213, 214

MLP Multi-Layer Perceptron 128, 131, 132, 136, 140, 156, 242

MOS Mean Opinion Score xxi, xxiii, xxiv, 26, 78, 104, 105, 112, 206–211

MPC Model Predictive Control 71, 72, 74, 144–146, 148, 150, 160, 165, 173, 174

MSS Microsoft Smooth Streaming 3

NELL Normalized Expected Log-Likelihood 131, 132

NN Nearest Neighbourhood 131

NSS Natural Scene Statistics 28, 33, 34

OSQP Operator Splitting Quadratic Program 103

PLCC Pearson Linear Correlation Coefficient 104, 106

PSNR Peak Signal-to-Noise Ratio 16, 186

QoE Quality-of-Experience iv, v, xiv, xvi, xxiii, xxiv, 1–3, 6–14, 16, 19–40, 54, 55, 58, 68, 71, 74–79, 83–88, 90–111, 113–115, 148, 157, 158, 165, 170, 180, 186, 188, 190–199, 201, 203–214

QoS Quality-of-Service 17, 23, 30, 31

QP Quantization Parameter 30, 104, 111, 186

RB Rate-Based 72, 74, 164, 191, 192, 195, 202

RDOS Rate-Distortion Optimized Streaming iv, v, xx, xxi, 11–13, 86–88, 90–92, 94, 181, 189, 191–195, 198, 199, 202, 207–214

RNN Recurrent Neural Network 11, 128, 137, 140, 198

RobustMPC Robust Model Predictive Control 197

RTT Round Trip Time 30, 39, 55, 168, 179, 190

SRCC Spearman Rank-order Correlation Coefficient 104, 106, 206, 207

SS Single Stimulus 204, 205

SSCQE Single Stimulus Continuous Quality Evaluation 204, 205

SSIM Structural Similarity Index 33, 186, 197

SSIMplus Structural Similarity Index Plus 18, 19, 96, 103, 188, 194

SVR Support Vector Regression 50, 131

TCP Transmission Control Protocol 40, 41, 43, 47, 49, 62, 85, 127, 215

TTP Transmission Time Predictor 136, 153

UCC University College Cork 129, 130, 163, 186

UHDTV Ultra High Resolution Television 108, 165, 188, 203, 205, 247

VMAF Video Multi-method Assessment Fusion 19, 32, 34, 37, 96, 103, 106, 107, 111, 112, 155, 162, 167, 184, 186, 188, 194, 197, 198, 203, 247

VQA Video Quality Assessment 16–19, 21, 34, 37, 103, 106, 157, 197

WaterlooSQoE-I Waterloo Streaming QoE Database-I 37, 103, 157, 158, 206, 207

WaterlooSQoE-II Waterloo Streaming QoE Database-II 103, 157, 158, 206, 207

WaterlooSQoE-III Waterloo Streaming QoE Database-III xxiii, 38, 103, 104, 106, 111, 113, 206, 207

WaterlooSQoE-IV Waterloo Streaming QoE Database-IV xxiii, xxiv, 38, 103, 104, 111, 200, 204–208, 210

WaterlooSV Waterloo Streaming Video 12, 14, 130, 181, 182, 187, 189, 190, 244, 247

List of Symbols

$\mathcal{A}_f$ The set of pre-defined feasible actions/ bitrate selections 63, 70

$\mathcal{A}$ The set of all possible actions/ bitrate selections 4, 7, 55, 59, 60, 65, 69–72, 82, 145, 148, 149, 152

$\mathcal{D}_{\mathbf{c}}$ Throughput dataset 43, 44, 122

$\mathcal{D}_{\mathbf{q}^+}$ Augmented bitrate selection dataset comprising of state, action, value tuples 68, 148

$\mathcal{D}_{\mathbf{q}}$ Bitrate selection dataset comprising of state, action, value tuples 58, 65, 142, 143, 146, 151, 154, 155

$\mathcal{D}_{\mathbf{x}}$ Subject-rate streaming video dataset 25, 100

$\mathcal{D}_{\mathbf{z}}$ Subject-rate streaming video dataset with extracted latent variables 34, 36–38, 95

$\mathcal{E}$ Streaming environment 4, 5, 82

G Granularity of throughput measurement 129

H Past throughput observation window size 47, 50, 155

K Look-ahead window size 43, 46, 51, 71, 72, 123, 125, 144, 145, 155, 160

L Loss function 122

$N_{\mathbf{c}}$ Number of samples in a throughput dataset 43

$N_{\mathbf{q}}$ Number of samples in a bitrate selection dataset 58, 142, 154

$N_{\mathbf{x}}$ Number of samples in a subject-rate streaming video dataset 25, 34

Q_t Quality-of-Experience of the t-th chunk 86

Q Quality-of-Experience 86, 88, 96, 98, 100, 101

R_t Bitrate of the t-th chunk 82, 88

R Bitrate 86

$\mathcal{S}$ The set of all possible state 4, 59, 60, 72, 82, 145, 149

$\mathcal{T}_i$ The i-th task in meta-learning 122

$\mathcal{T}$ A task in meta-learning 122, 124

T Total number of chunks 5, 7, 37, 43, 45, 55, 57, 59, 60, 71, 96, 129, 131, 142, 144, 145, 149, 150, 152, 153, 160

$\mathcal{U}$ All possible reward/utility function 4, 5

U_t Utility function of the t-th instance 141

U Reward/utility function 157

$\mathbf{X}$ A dataset of streaming video 25, 26

$\mathbf{y}$ A dataset of quality ratings 25, 26, 34

Z Number of chunk-level features 37, 55

a_{t+1} Next action/bitrate selection 4, 57, 83

a_t^* Optimal decision for the current chunk selection 141, 153

a_t Current action/bitrate selection 4, 57, 58, 82, 86, 88, 141, 142, 154

a^* Optimal action/bitrate selection 60, 143

a Action/bitrate selection 57–60, 62–65, 67–72, 141–152, 154, 158

β Variance of a Gaussian distribution 26, 58

b_k The buffer occupancy of the k-th chunk 39

$\mathbf{c}_1^T$ All past throughput observations 43, 45

$\mathbf{c}_1^t$ All past throughput observations up to the t-th instance 43, 44, 53

$\mathbf{c}_{t-H+1}^t$ Past H throughput observations 47, 50, 51

c_k Average throughput when downloading the k-th chunk 39

$\mathbf{c}_{t+1}^{t+K}$ The throughput observations from the $t+1$-th instance to the $t+K$-th instance 43, 44

c_{t+1} Average experienced throughput for the next chunk 46, 51, 53

c_t Instantaneous throughput at the t-th unit time 39, 46

$\mathbf{c}$ Throughput observations 122, 124

Δt_k The waiting time when downloading the k-th chunk 39

Δp_t Quality adaptation for the t-th chunk of streaming videos 96, 99, 157

Δp Quality adaptation 97, 100, 101

γ Discounting factor 57, 67, 142, 147, 158

$\mathbf{h}_{t+1}$ Hidden states of throughput condition at the $t+1$-th time instance 53

$\mathbf{h}_t$ Hidden states of throughput condition at the t-th time instance 53

$\mathbf{h}$ Hidden states for throughput dynamics 51

λ The rate-distortion tradeoff parameter 86, 90, 91, 193

μ Mean of a Gaussian distribution 150–153

$p_{\mathcal{E}}$ Distribution of streaming environment 4, 5

$\boldsymbol{\phi}_i$ Task-specific parameter of the i-th task 123–126, 129

$\boldsymbol{\phi}$ Parameter of distortion process 30, 31, 86

$\boldsymbol{\pi}$ Adaptive bitrate controller 4, 5, 57, 60, 82, 141, 144

p_{t-1} Presentation quality for the $t-1$-th chunk of streaming videos 96

p_t Presentation quality for the t-th chunk of streaming videos 96, 99, 157

p Presentation quality of streaming videos 100, 101

$q(\cdot)$ Chunk-level Quality-of-Experience function 99

q^* Optimal state-action value 58–60, 62–65, 68–72, 143, 144, 148, 149, 152, 154

$\tilde{q}$ A noisy sample of the optimal Q function 150, 151

q_t^* Optimal state-action value at the t-th instance 57, 58, 141, 142, 144

q_t State-action value at the t-th instance 57, 58

q State-action value 60, 71, 72

r_k The bit count of the k-th chunk 39

r_{t-1} Bitrate of the $t-1$-th chunk 71, 157

r_t Bitrate of the t-th chunk 71, 157

$\mathbf{s}_{1:t}$ All previous state 4

$\mathbf{s}_{t+1}$ Next state 4, 57, 83, 154

$\mathbf{s}_t$ Current state 4, 57, 58, 82, 141, 142, 153, 154

$\mathbf{s}$ State 57–60, 62–65, 67–72, 141–154, 158

τ_k Rebuffering duration when downloading the k-th chunk 39

τ_t Rebuffering duration when downloading the t-th chunk 71, 96, 99, 157

τ Rebuffering duration 100, 101

$\boldsymbol{\theta}_1$ Parameter of feature extractor 28, 29, 34, 51, 53

$\boldsymbol{\theta}_2^*$ Optimal model parameter 35, 36

$\boldsymbol{\theta}_2$ Parameter of regression model 28, 34–37, 51, 95, 98

$\boldsymbol{\theta}_i$ Model parameter at the i-th iteration 67, 147

$\boldsymbol{\theta}^*$ Optimal model parameter 25, 58, 60, 143

$\boldsymbol{\theta}_t^*$ Optimal parameter of the throughput prediction model for the t-th instance 44

$\boldsymbol{\theta}_t$ Parameter of the throughput prediction model for the t-th instance 44, 45

$\boldsymbol{\theta}$ Model parameter 25, 26, 28, 29, 36, 45–47, 50, 51, 58, 60, 62–65, 68–70, 72, 82, 122–126, 143, 144, 146–148, 150, 153, 154

$\tilde{u}_t$ Estimated instantaneous reward of the t-th instance 63

t_{k+1} The timestamp that the k-th chunk downloading finishes 39

t_k The timestamp that the k-th chunk downloading starts 39

t Segment index 4

u_t Instantaneous reward of the t-th instance 57, 59, 62, 63, 70, 149, 154

u Instantaneous reward 67, 147

$\mathbf{x}_0$ Reference video 33, 34

$\mathbf{x}_{t+1}$ The $t+1$-th chunk of streaming video 88

$\mathbf{x}_t$ The t-th chunk of streaming video 88, 94

$\mathbf{x}$ Streaming video 24–26, 28–31, 33, 34, 38, 39, 86, 94

y Quality-of-Experience score 24, 25, 28, 29, 31, 38, 39, 95

$\mathbf{z}_t$ Latent variables for the t-th chunk of streaming videos 94, 95

$\mathbf{z}$ Latent variables for streaming videos 28–31, 34, 37, 38, 94

Chapter 1

Introduction

This research addresses some long-standing problems of digital video delivery that are related to perceptually oriented network resource allocation. The root of these problems is the misunderstanding of viewers' QoE. In particular, traditional video delivery systems tangle bitrate with quality, aggressively draining all available bandwidth. Such "best-effort" approach misses out opportunities to save bandwidth usage and will eventually cause network congestion or sub-optimal user QoE. The phenomena is widely known as Tragedy of Commons in Economics [130].

In this thesis, we will explore a fundamentally different design philosophy for video delivery systems, where we consider video streaming as a rate-distortion optimization problem. Distinct from the bitrate maximization approach, we view bitrate as the cost of video delivery, which should be minimized while preserving a target QoE constraint. The new approach operates at any given point along the rate-distortion curve, as specified by a trade-off parameter.

Of course, the rate-distortion optimization perspective by itself does not provide a complete solution to the resource allocation problem. To instantiate the new paradigm, we develop three functional components in the video delivery system from a Bayesian perspective, including a perceptually grounded QoE model, a connection adaptive throughput predictor, and a deep reinforcement learning-based policy. The idea is to complement

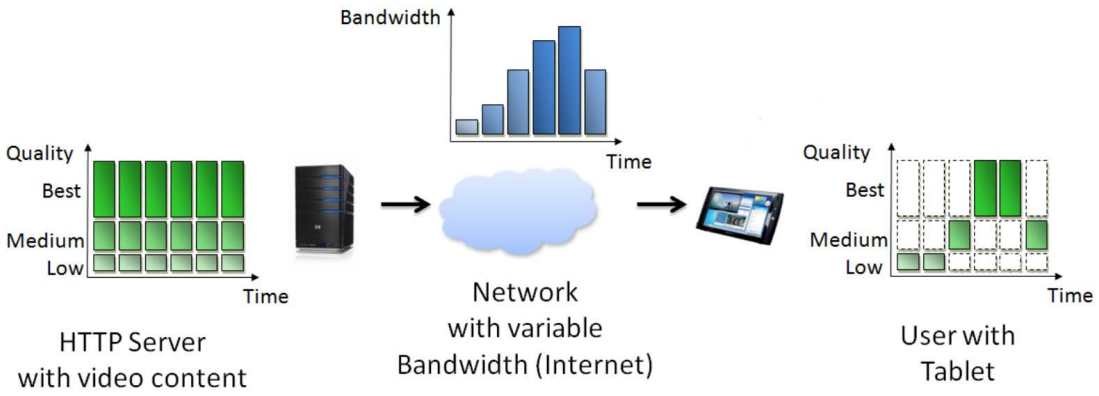


Figure 1.1: An overview of HTTP adaptive video streaming.

the information encoded in realistic training samples with suitable prior knowledge in the design of complex, high dimensional, and time sensitive systems.

1.1 Background and Motivation

1.1.1 Dynamic Adaptive Streaming over HTTP

Video traffic from content delivery networks is expected to occupy 71% of all consumed bandwidth by 2021 and exceed 82% by 2022 [215]. The explosion of data volume introduced by media streaming will quickly drain available network bandwidth in the next decade. Concurrent with the scarcity of network resources is the steady rise in user demands on video quality. With the emergence of new technologies such as 4K, high dynamic range, and high frame rate, viewers' expectation on video quality has been higher than ever. The trends in the shortage of network resources and the increasing demand in QoE has posed significant challenges to content providers supporting millions of users and devices.

Since the ratification of the Dynamic Adaptive Streaming over HTTP (DASH) standard in 2011 [196], video service providers have invested significant effort in the transition from the conventional connection-oriented video transport protocols towards HTTP Adaptive Streaming (HAS) due to its ability to traverse network address translations and firewall,

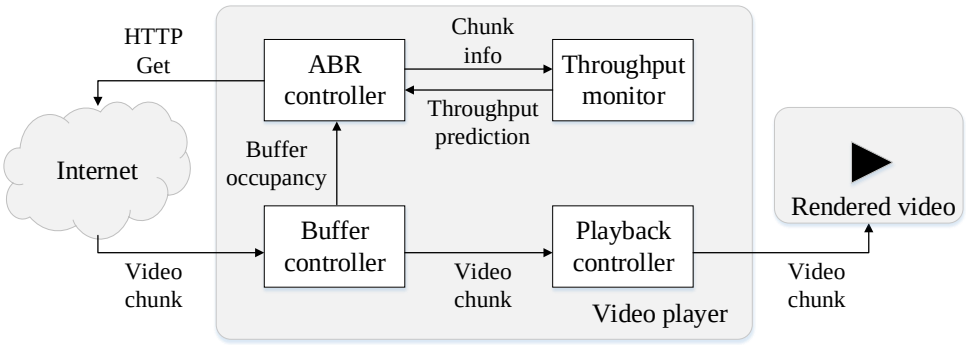


Figure 1.2: System diagram of adaptive bitrate streaming player.

reliability to deliver video packet, flexibility to react to volatile network conditions, and efficiency in reducing the server workload. Major streaming and media companies have instantiated a variety of competing adaptive streaming protocols such as HTTP Live Streaming (HLS), Microsoft Smooth Streaming (MSS), and HTTP Dynamic Streaming (HDS) to promote and catalyze the adoption of DASH into a real business. The DASH technology has empowered tens of world-class streaming applications including Netflix, YouTube, TikTok, Disney+, Youku, and *etc.*, each of which has received at least 10 million overall downloads [194].

Figure 1.1 illustrates the end-to-end process of streaming a video over DASH [63]. Specifically, a source video is encoded at a variety of bitrates and video attributes (such as spatial resolution, frame rate, and bit depth), and segmented into small HyperText Transfer Protocol (HTTP) file chunks of 2-10 seconds each at the video server. The bitrate-encoding attributes tuples are usually referred to as *bitrate ladder* or *encoding profiles*. Furthermore, the media information of each segment is stored in a *manifest* file, which is created at server and transmitted to clients to provide the specification and location of each segment. Throughout the streaming process, the video player at the client adaptively switches among the available streams by selecting video chunks at different quality levels. ABR algorithms, that determine the bitrate of the next segment to download, are not defined within the standard but deliberately left open for optimization. The key is to define an optimization criterion that aims at maximizing viewer QoE given limited bitrate resources.

Figure 1.2 shows a system diagram of ABR controller. At the start of each streaming session, a video player sends a token to a video service provider for authentication. Once the manifest file is received, the video player at the client then adaptively selects a video representation to download based on playback rates, buffer conditions and instantaneous throughput [196]. Concurrently, a buffer controller keeps depleting a *playback buffer* by sending the next second of video to a playback controller, and replenishes the buffer with the newly downloaded chunk. The remaining video playback time in the buffer is called *buffer occupancy*. If buffer occupancy runs out, a *rebuffering* event will occur. Besides, a *throughput* monitor estimates the network bandwidth according to the size and download time of the last chunk. Finally, the playback controller continues rendering the video received from the buffer unless rebuffering interrupts.

1.1.2 Functional Decomposition

To facilitate a better understanding of the ABR decision problem, we present a novel framework that decomposes ABR functionally into three sub-components. Specifically, the aforementioned streaming process can be recast as a reinforcement learning problem, where an agent ought to take actions in an environment in order to maximize the notion of cumulative reward. Before chunk t+1 is downloaded, the ABR controller $\boldsymbol{\pi}$, or the *agent*, performs an *action* $a_t \in \mathcal{A}$ based on all previous *states* $\mathbf{s}_{1:t}$, where $\mathbf{s}_t \in \mathcal{S}$ for all t, to determine which representation to download. The state $\mathbf{s}_t$ generally encodes information about throughput history, buffer occupancy, and previous downloaded representations before a_t is taken. Given a bitrate decision a_t and the previous *states* $\mathbf{s}_{1:t}$, the *environment* $\mathcal{E}$ consisting of the characteristics of streaming video and the future throughput will download a corresponding representation of the next chunk, updates the buffer occupancy, tracks the throughput, continues playing the video from the buffer, and returns all the updated *state* $\mathbf{s}_{t+1}$ to the agent. Then the agent takes another action a_{t+1} based on the new states. The cycle repeats until the whole video is streamed. Assuming that $p_{\mathcal{E}}$ depicts the real-world distribution of environment, the ultimate goal of the *agent* is to optimize the expectation of certain utility/reward function $\mathcal{U}$. Mathematically, the ABR problem can be expressed

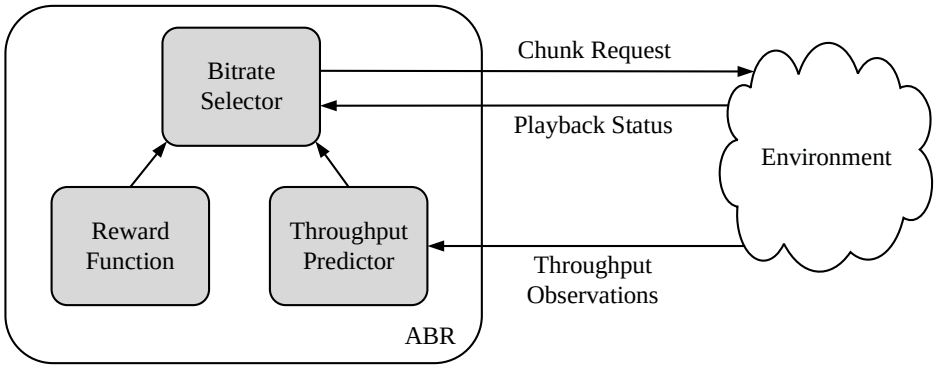


Figure 1.3: The schematic diagram of the proposed framework.

as

$$\begin{aligned} \underset{\boldsymbol{\pi}}{\text{maximize}} \quad & \mathbb{E}_{p_{\mathcal{E}}}[\mathcal{U}(\mathbf{s}_{1:T}, a_{1:T})] \\ \text{subject to} \quad & a_t = \boldsymbol{\pi}(\mathbf{s}_{1:t}) \\ & \mathbf{s}_{t+1} = \mathcal{E}(a_t, \mathbf{s}_{1:t}), \end{aligned} \tag{1.1}$$

where T represents the total number of chunks[1].

All ABR algorithms can be summarized by this framework, while they differ in how they model the environment $\mathcal{E}$, what reward function $\mathcal{U}$ to optimize, and using which strategy $\boldsymbol{\pi}$ to optimize the reward function. The transition probability distribution $p_{\mathcal{E}}$ is determined by three factors, including the ABR player buffer dynamics, the network dynamics, and the encoded video chunk characteristics. The buffer dynamic is a deterministic process that has been thoroughly investigated [93, 241]. In live streaming applications, the uncertainty in $p_{\mathcal{E}}$ is governed by the throughput evolution and the future video chunk properties. For example, one would need to explicitly model the time-varying chunk size and local quality distributions in order to make an optimal bitrate selection. For video on demand services, the chunk-level statistics can be obtained from the manifest file before the decision stage. The major uncertainty for the distribution of environment $p_{\mathcal{E}}$ arises from the stochastic nature of future network throughput [200]. In this study, we focus on the video on demand

[1]The proposed decomposition is a simplified version of the practical ABR system, without considering the player (decoding and rendering) speed, and potential transmission/decoding errors.

services, although live streaming is a straight-forward extension. As a result, we formalize the three modules as the throughput predictor, the reward function, and the bitrate selector respectively. The schematic diagram of the proposed framework is illustrated in Figure 1.3. At each decision step, the throughput predictor firstly estimates how much bitrate resource will become available for allocation, based on which and other state variables such as buffer occupancy the bitrate selector will select a representation that hopefully optimizes a meaningful reward function.

1.1.3 Challenges in Adaptive Streaming

The design of three aforementioned functional components heavily influence the performance of ABR algorithms. However, all of these algorithms face five primary practical challenges:

- ABR algorithms must balance the benefit of maximizing QoE and the cost of network congestion. However, these goals are inherently conflicting. For example, increasing bitrate from 5 Mbps to 10 Mbps may introduce imperceptible quality improvement, but may potentially cause server overload and network traffic. Conversely, consistently choosing the lowest possible bitrate resolves the network congestion at the cost of significant QoE degradation.

- While it has been widely accepted that accurate QoE measurement lies in the root of ABR systems, an easy-to-use, mathematically well-behaved, and perceptually grounded QoE model is still lacking. In practice, bitrate remains the major indicator of QoE. However, encoding two different videos with the same bitrate could result in a substantial difference in perceived picture quality, suggesting that bitrate and picture quality/QoE cannot be used interchangeably. This is in addition to the large differences in performance between different encoders/transcoders with different configurations. The actual user QoE also varies with respect to the device being used to display the video, another factor that cannot be taken into account by bitrate-driven video delivery strategies. To further complicate matters, existing subjective

studies suggest that QoE is further influenced by rebuffering, quality adaptation, and their interactions with video quality.

- ABR agent does not have perfect information about the network conditions, which can fluctuate over time and can vary significantly across environments. This complicates bitrate selection as different scenarios may alter the major input signals in ABR decision. For example, on time-varying cellular links, throughput prediction is often inaccurate and cannot account for sudden fluctuations in network bandwidth, resulting in underutilized network and low video quality or inflated download delays and rebuffering. To overcome this, ABR algorithms must prioritize more stable input signals like buffer occupancy in these scenarios.

- An ideal bitrate selection strategy should be efficient in running time, robust[2] to unobserved environmental states, and optimal in reward. However, these objectives are extremely difficult to meet in practice. Specifically, there are approximately $|\mathcal{A}|^T$ possible bitrate selection strategies in one streaming session, where $|\mathcal{A}|$ and T are the number of encoding profiles and the number of temporal chunks, respectively. In a typical streaming video ($|\mathcal{A}| \approx 10$ and $T \approx 200$), the search for global optimal solution is computationally intractable. Moreover, bitrate selection has very strict time requirements, where a second delay in bitrate selection leads to notable QoE losses. As a result, practical online bitrate selection algorithms have to compromise the optimality in order to obtain a reasonable efficiency. On the other hand, despite being computationally efficient and nearly optimal in certain scenarios, learning-based offline decision rules are often susceptible to unobserved environmental states, whose dimension is in the order of thousands.

- The validation of ABR algorithms corresponds to the evaluation of (1.1) for each ABR algorithm, which is a complex problem in its own right. First, the measurement of (1.1) requires the precise knowledge about the streaming environment. Quantitatively, this means specification of probability distributions over both network conditions and streaming videos, neither of which are available. In practice,

[2]In this thesis, we define robustness as the model generalizability to unobserved test samples.

many authors have to base their studies on empirical results computed from a limited set of example network traces and streaming videos. Second, since the HVS is the ultimate receiver of streaming videos, the only "correct" way to evaluate the reward function and the corresponding ABR system is by performing a subjective experiment. Unfortunately, subjective testing is inconvenient, time-consuming, and expensive. Restricted by these practical constraints, the quantity, representativeness, and reliability of evaluation data can be hard to satisfy simultaneously.

1.1.4 Trends in Multimedia Communication

If the bitrate allocation problem is our enemy in the thesis, multimedia communication technology is our arsenal. In particular, rate-distortion theory is the most important development in the history of digital signal communication, presenting the natural link between the cost of transmitting a signal and the distortion of the approximate signal at the receiver. For the purposes of this thesis, there are two crucial trends in multimedia communication technology: a deeper understanding of HVS and perceptually motivated signal processing systems, and the thrive in data-driven methods for communication systems.

There is an growing consensus in the video distribution industry that the design and operation of the full video delivery chain needs to be driven by the QoE appropriate to the end-users. The gigantic scale of video data transmission and previous success in perceptual image and video processing have attracted significant interests in understanding and modeling of subjective QoE responses for adaptive streaming videos. In particular, there have been more than 100 independent subjective experiments and numerous publicly available datasets dedicated to investigate and measure the QoE of streaming videos over the past decade [184]. A number of useful observations has been drawn such as 1) rebuffering significantly degrade viewers' QoE, 2) frequent quality adaptations irritate end users, and 3) the same video viewed at different environment produces drastically experience. All these studies, if exploited carefully, may lead to significant improvements over the existing ABR algorithms. However, this is not to say that predicting subjective QoE would be easy. We will still have to overcome significant challenges in developing a computationally efficient, mathematically well-behaved and HVS properties-conforming objective QoE model.

Another powerful trend is that multimedia communication and machine learning have begun to merge in two fundamental ways. First, while multimedia communications have developed largely as a model-driven field, the complexity of many emerging communication scenarios is giving rise to the need to introduce data-driven methods into the design and analysis of mobile networks. And, conversely, many machine learning problems are by their nature distributed due to either physical limitations or privacy concerns. This distributed nature gives rise to the need to consider mobile networks as part of learning mechanisms. The quintessential example of communication involving machine learning is the recommender systems, based on which companies like Amazon, Netflix, and Linkedin help users discover new and relevant items (products, videos, jobs, music), creating a delightful user experience while driving incremental revenue. Specifically, recommender systems learned purely from enormous training data were shown to significantly outperform traditional expert systems. Other examples of data-driven mechanisms in communication problems include proactive caching, resource allocation and security, and the consideration of communication issues arising in distributed learning problems such as federated learning and social learning. Machine learning is truly an integral component of modern multimedia communication.

In summary, present-day multimedia communication provides a very rich substrate for new ABR systems. The two key nutrients are in-depth knowledge of HVS and recent flourish of computation power, big data, artificial intelligence algorithms, and deep learning infrastructures.

1.2 Objectives

The objectives of this thesis are to overcome the fundamental limitations of traditional evaluation and design methodologies of video delivery systems by ways of perceptually motivated QoE modeling and rate-distortion optimized streaming. We aim to develop theories and algorithms for network system friendly, perceptually oriented, computationally efficient, environmental shift robust, and statistically optimal network resource allocation.

1.3 Thesis Overview

1.3.1 Organizational Themes

The central contribution of this thesis is the introduction of the rate-distortion optimized streaming system: a general solution to the five manifestations of the network resource allocation problem in this introduction. The following five themes unify presentation of the system in response to the five aforementioned challenges.

- Conflicting Objective Challenge: To balance the contrast demands from video consumers and service providers, we rethink the bitrate adaptation problem by asking the question: is there inefficient bandwidth usage in adaptive streaming? Throughout a set of simulation experiments, we show that current bitrate adaptation strategies result in significant bandwidth waste in a variety of scenarios. We demonstrate that the root cause of the problem can be attributed to the ignorance of content characteristics, encoder performances, and display devices. Given that there exists redundant bitrate usage in adaptive streaming, can we make more wise choices on the bitrate selection? Put another way, is it possible to maximize users' QoE while minimizing the network resource usage? Following this line of thought, we formulate bitrate adaptation as a generalized rate-distortion optimization problem. We show that it is possible to achieve an optimal balance between bitrate utilization and viewers' QoE with a broader design space of control algorithms and a deliberately designed objective function.

- QoE Complexity Challenge: We develop BSQI, a Bayesian framework for perceptually motivated QoE modelling. In contrast to the existing objective QoE models whose functional form and parameter configuration are selected on the basis of mathematical convenience, BSQI are built upon the combination of subject-rated streaming videos and meaningful prior knowledge about source videos, distortion process, and HVS. We show that all valid QoE functions must lie within a convex set resulted from the known properties of HVS, and thus the QoE function estimation can be formulated as a projection onto convex sets problem. Extensive experiments

on four benchmark QoE databases demonstrate that BSQI outperforms state-of-the-art objective QoE models. The robustness of BSQI is also significantly improved as confirmed by a novel analysis-by-synthesis experimental methodology. Most importantly, by explicitly disentangling the bitrate from QoE model, the proposed QoE measure fits perfectly into the RDOS system.

- Throughput Variability Challenge: Motivated by the shortcomings of the existing model-based throughput predictor and the abundant throughput data, we develop a data-driven throughput predictor by leveraging the latest advances in deep learning. Unlike the traditional methods used for training throughput predictor, our training process is performed in two stages. In the first stage, the proposed algorithm explicitly learns a generic prior model from a large corpus of throughput traces that can quickly adapts to a broad range of throughput characteristics. In the second stage, starting from the prior model and only a few connection-level throughput observations, we fine tune the pre-trained model for posterior inference with minimal number of gradient steps. Using the proposed algorithm, we end-to-end optimize a variant of Recurrent Neural Network (RNN) to predict the conditional probability distribution of the future throughput, effectively resolving the long-term dependencies between throughput observations. We empirically demonstrate that the proposed model outperforms the other throughput predictors in several experimental setup with only moderate model complexity.

- Efficient-Robust-Optimal Tradeoff Challenge: Motivated by the recent success of AlphaGo [187], we propose a bitrate selection framework, namely Bayesian Bitrate Selection (BBS), based on the combination of online traditional controllers and offline reinforcement learning-based policies. The online dynamic programming-based controller plays the role of likelihood function in the Bayesian theory. Despite the relatively demanding computational complexity, the online algorithm is guaranteed to provide a short-term optimal solution that is invariant to the probability distribution of streaming videos. The offline policy learns a prior bitrate selection model, *tabula rasa*, by interacting with the streaming environment characterized by the joint probability distribution of network conditions and streaming videos. In contrast to

the online controller, the data-driven policy can efficiently optimize long-term reward, at the cost of model robustness. BBS underlies a family of ABR algorithms that combine the online search policy and offline prior action-value belief in a principled way, from which we find a specific algorithm named EfficiEncy, Robustness, and Optimality (EERO). Depending on the reliability of the two policies in certain environmental status, EERO dynamically prioritizes one of the two decision rules to obtain the optimal bitrate decision. We compare EERO to state-of-the-art bitrate adaptation algorithms using trace-driven experiments spanning a wide variety of network conditions, streaming videos, and QoE measures. In all considered scenarios, EERO outperforms the existing techniques in terms of robustness and optimality with minimal computation overhead.

- Data Challenge: We believe that a large-scale database with great streaming video diversity and realistic throughput variability is critical to evaluate ABR algorithms. This motivates us to construct the Waterloo Streaming Video (WaterlooSV) database, which in current state consists of 250 4K pristine videos and 20,000 realistic throughput traces. To cover the diversity of video distribution network, we encode each source video into 180 representations for three commonly used video encoders. Using the WaterlooSV database as testbed and a state-of-the-art chunk-level simulator, we present the most comprehensive objective ABR evaluation without sacrificing the representativeness of data. To complement the objective evaluation with respect to the reliability of reward function, we also conduct so-far the largest subjective evaluation on a subset of ABR algorithms and environmental conditions. The full system implementation of RDOS outperforms the best existing scheme, with average bitrate saving ranging between 5%-54%.

1.3.2 Dissertation Roadmap

I have tried to write and illustrate the thesis in a way that will hopefully make it easily accessible and interesting to a broad range of readers, including computer network scientists, QoE experts, machine learning practitioners, and ABR engineers.

Figure 1.4 is a map of some paths that one may choose through the coming chapters, and the topic that one would cover. All readers may find it useful to firstly explore Chapter 2, which presents a comprehensive review of the existing ABR algorithms from a Bayesian perspective. It also introduces the notation used in the thesis. QoE experts will be most interested in the RDOS paradigm and objective QoE detailed in Chapter 3 and 4, and may wish to begin their exploration there. Chapter 5 and 6 assume knowledge of stochastic process and machine learning at the level of first year graduate course, but it is not essential to develop the intuition and digest the main ideas. Chapters 4, 5, and 6, each of which covers an individual ABR functional component, may be read in any order, while it is recommended to go through chapter 3 prior to chapter 4. Chapter 7 presents more sophisticated analysis and variations of the system, in which most ABR practitioners may be interested.

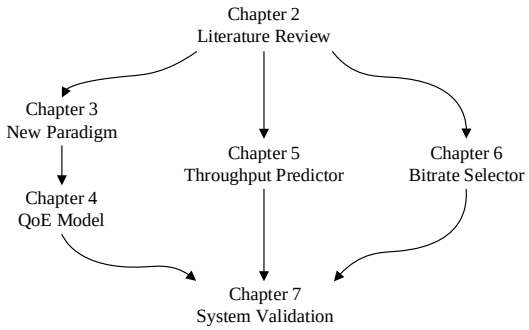


Figure 1.4: Dissertation roadmap.

1.3.3 Chapter Descriptions

The organization of the thesis is as follows.

Chapter 2 discusses the related work in the literature. It reviews the existing studies of throughput predictor, reward function, and bitrate selector from a Bayesian perspective. We also present an overview of prior ABR algorithms, which are essentially built upon different instantiations of the three functional components. In the end, we discuss a variety of validation procedures that have been widely used in the design of ABR systems.

Chapter 3 proposes the new philosophy for the design of ABR algorithms, which balances the conflicting objectives of video consumers and service providers. We compare the new approach to the traditional bitrate maximization paradigm, and illustrate the necessity of RDOS from three distinct perspectives.

Chapter 4 presents BSQI, a Bayesian framework for perceptually motivated QoE modelling. BSQI are built upon the combination of subject-rated streaming videos and the prior knowledge about HVS. By analyzing a corpus of psychophysical experiments, we show the QoE function estimation can be formulated as a projection onto convex sets problem.

Chapter 5 studies throughput prediction by leveraging the latest advances in deep learning. The proposed throughput predictor learns to quickly adapt to connection-level network characteristics, so as to achieve per-user optimization.

Chapter 6 develops a Bayesian bitrate selection framework that unifies all existing bitrate adaptation functions. We provide a specific implementation of the framework based on the combination of a variant of tree search and a state-of-the-art reinforcement learning-based policy. Depending on the reliability of the two controllers in certain states, the algorithm dynamically promotes one of the two decision rules to obtain the optimal decision.

Chapter 7 studies the overall performance of the three components in the adaptive streaming. It presents in detail the construction of the WaterlooSV dataset consisting of a wide variety of streaming videos and throughput traces. Based on the novel dataset, we present so-far the most comprehensive performance analysis of ABR algorithms with both objective evaluation and subjective evaluation.

Chapter 8 summarizes lessons learned and points to future directions.

Chapter 2

Literature Review

The proposed framework in Figure 1.3 not only provides a mathematical formulation to the ABR problem, but also decomposes the system into three components such that each component can be studied and reviewed independently. In this chapter, we will follow the functional decomposition to review the previous studies in reward function, throughput prediction, and bitrate selector in ABR streaming. We will then present an overview to ABR algorithms with an emphasize on the existing approaches for functional integration. In the end, we will review the existing methodologies to validate ABR algorithms.

2.1 Reward Function

The objective of ABR algorithm is for the player to obtain an optimal, or nearly-optimal, policy that maximizes the reward function or other user-provided reinforcement signal that accumulates from the immediate rewards. The reward function, which describes how the agent "ought" to behave, not only defines the objective function in an optimization framework, but also determines the evaluation criteria of competing mechanisms. Therefore, the design of reward function lies at the heart of adaptive streaming systems. Despite its importance, the research in the reward function has received very little interests. All the existing methods adopt a viewer-centric design paradigm, assuming the ultimate goal

of ABR algorithms is to optimize viewers' QoE. To this regard, we firstly introduce the existing subjective QoE assessment studies and summarize their key observations. We then present an overview of objective QoE methods from a Bayesian perspective, with the goals of unifying a wide spectrum of QoE approaches under a common framework and providing useful references to fundamental concepts accessible to vision scientists and video streaming practitioners.

2.1.1 Subjective QoE Assessment

Subjective testing is the first step towards understanding the perceptual QoE of streaming videos. Psychophysical experiments not only guide the development of objective QoE models, but also provides useful data to validate competing hypotheses and theories. In general, there are three types of distortion patterns in adaptive streaming videos including compression artifact, rebuffering, and quality adaptation [67, 184]. In this section, we will structure the review of subjective QoE studies with respect to the impairment category.

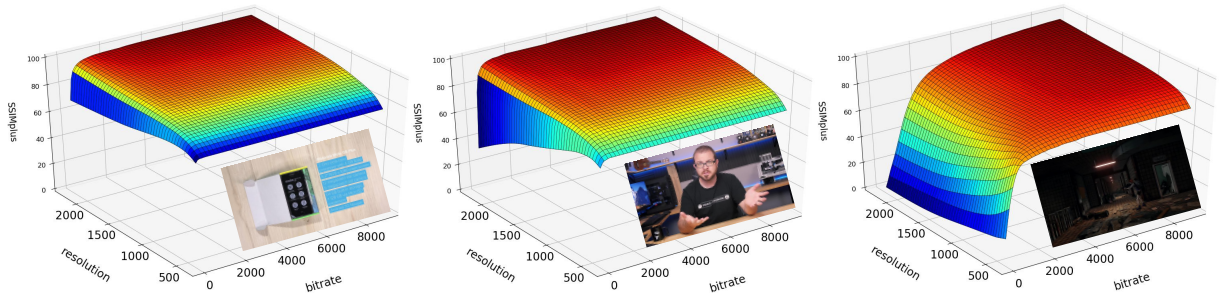
Video Quality Assessment Studies

Pioneering work on subjective VQA dated back to as early as 2000, when the video quality expert group investigated a class of visual fidelity measures in the context of MPEG-2 video compression [175]. The experiment illustrated the challenges in characterizing the perceptual quality with simple measures such as bitrate and Peak Signal-to-Noise Ratio (PSNR), encouraging further investigation in VQA. As a follow-up to this small scale experiment, the same group carried out a similar experiment covering a wider range of video contents and distortions [77]. It was further confirmed that notwithstanding the approximate monotonicity between the bitrate and quality, the bitrate required to compress videos for a specified visual quality varies dramatically with respect to the content. Since then, more and more researchers began to realize the difficulty in accurately quantifying the relationship between video content and visual quality, given the large diversity of digital videos. A number of experiments have been implemented to identify a set of features from source content that well correlate with the perceptual quality such as spatial frequency [226],

spatial information [158, 193, 234], temporal information [158], colorfulness [234], sample entropy [209], and motion strength [182, 183, 226, 234, 238]. For example, a video with stronger motion increases the encoding complexity, thus requiring a higher bitrate to achieve certain visual quality.

With the advance of networking and encoding technologies, multimedia streaming has gradually become the mainstream for video delivery. The broader space of distortion process motivates quality assessment techniques from novel perspectives. Assuming there is a causal relationship between the transmission channel congestion and the video quality degradation, the network Quality-of-Service (QoS) community tried to quantify visual quality with transmission errors, such as bit error rate, packet loss rate, and network jitter [106, 136, 152, 161, 220]. The network QoS approach has achieved limited success in predicting visual quality as the measurements and protocols used are oblivious to the actual content being transmitted over the network and have no direct relation to the video quality as perceived by the user [51]. A parallel line of research investigated the influence of video encoder on visual quality. Several independent subjective experiments reported that there has been a significant improvement in the compression efficiency of digital videos [40, 118, 142, 170, 183]. Specifically, the standard H.264 [232] encoder takes as much as 63% more bitrate than the state-of-the-art AV1 [28] to achieve the same level of perceptual quality. Even for the same video encoder, the rate-distortion performance strongly depends on encoding configurations such as the motion estimation method [129], the status of de-blocking filter [244], and the choice of distortion metric [218].

Thanks to the improvement in video acquisition and display devices, high spatial resolution, high frame rate, and high dynamic range videos are becoming increasingly more popular over the past decade. Although the extension in resolution and precision provides higher quality moving pictures to end users, these novel dimensions cast significant challenges to the VQA. A number of subjective tests have been dedicated to evaluate the impact of these new video format in the context of video streaming [4, 118, 146]. It turned out that the higher precision has a somewhat unexpected benefit to the video compression, apart from its intrinsically higher quality. To encode a video at a target bitrate and media attributes (*i.e.*, spatial resolution, frame rate, and dynamic range), one can either compress the video directly, or employ a resample-compression-resample encoding strat-



(a) (b) (c)

Figure 2.1: Samples of generalized rate-distortion surfaces for different video content.

egy [36, 120, 156]. It has been shown that a high-resolution, high-dynamic range encode may produce a quality lower than the one produced by encoding at the same bitrate but at a lower resolution and dynamic range [4, 36, 51]. This is because encoding more pixels/dynamic range with lower precision can produce a worse picture than encoding less pixels/dynamic range at higher precision combined with upsampling and interpolation. The resulting visual quality are further amplified or alleviated by the characteristics of the viewing device and viewing conditions, which interplay with HVS features such as the contrast sensitivity function [169, 173]. In our previous work, we formalize the complex relationship among source content, encoding bitrate, encoding attributes, and viewing display by the concept of generalized rate-distortion surface [47, 48]. Some samples of generalized rate-distortion surfaces of different video content are illustrated in Figure 2.1, where encoding bitrate and diagonal spatial resolution are plotted against Structural Similarity Index Plus (SSIMplus) [169] scores, an state-of-the-art objective VQA measure. One interesting observation is that visual quality is not necessarily a monotonic function of bitrate when other video attributes such as spatial resolution are taken into account, even for the same video content.

Despite the plethora of subjective experiments and observations, VQA is still a subjective of ongoing research [5, 50, 121]. Nevertheless, there is generally a shift in research focus from encoded video quality assessment to DASH specific impairments quality evaluation such as rebuffering and quality adaptation in the past decade. In the subsequent section, I will introduce the recent progress in the subjective evaluation of rebuffering and quality

adaptation.

Rebuffering Experience Studies

Initial exploratory studies [6, 62, 142, 159, 165, 192] suggest that rebuffering is the most severe impairment in DASH, which significantly degrades viewers' QoE. Based on this observation, the earliest ABR logic [63, 93] were dedicated to prevent rebuffering events. Unfortunately, it is extremely difficult to completely eliminate rebuffering due to the shortage in bandwidth resources and the highly variable throughput capacity. Furthermore, the overestimate on the impact of rebuffering is at risk of underrunning, resulting in overly conservative bitrate selection. To overcome these problems, it is important to establish a quantitative relationship between rebuffering and their perceptual quality by human observers. In particular, subjective rebuffering experience studies aim to answer the following two questions: 1) How much video quality is a human viewer willing to sacrifice to avoid one second rebuffering event? 2) If a rebuffering event cannot be avoided, what is the optimal position for it to take place?

To address the first question, early attempts investigate the perceptual quality-rebuffering tradeoff by applying linear regression on subject-rated videos with various rebuffering patterns and bitrate levels [6, 53, 87]. However, the conclusion varies significantly across datasets due to the vast diversity of video content, encoder, and viewing devices used in the subjective experiments. For example, one of the first subjective experiments in the perceptual tradeoff between video quality and rebuffering observed that the impact of a one-second rebuffering is equivalent to a bitrate reduction of 100 Kbps [6]. While in a larger-scale follow-up experiment, the influence of rebuffering is 8× stronger [53]. To combat the content and encoder dependency, subsequent analyses [54, 45] replaced bitrate with state-of-the-art VQA measures such as SSIMplus [169] and Video Multi-method Assessment Fusion (VMAF) [117] as the perceptual quality measure. Their experiments suggest that to maintain a fluid viewing experience, viewers were willing to reduce the intrinsic video quality by 33% on average. The conclusion was shown to be more accurate and general, evident by the superior prediction accuracy across multiple subject-rated streaming video datasets [11, 12, 53, 46]. Despite the demonstrated success, the perceptual quality-

rebuffering tradeoff problem is further perplexed by the duration neglect effects [79], which posits subjects tend to be insensitive to the duration of a long lasting video impairment. In the context of adaptive streaming, this effect suggests that not every second of rebuffering worth the same amount of bitrate resources. A number of subjective studies [68, 87, 89] have empirically observed a concave relationship between the rebuffering duration and perceptual QoE, where the penalty assigned to each second of rebuffering diminishes over time.

In the course of solving the tradeoff problem, researchers began to realize that the impact of rebuffering depends on other variables, which motivated them to look for the answer for the second question. The earliest research following this direction identified a fundamental difference between initial delays and rebuffering [85, 178]. Distinct from initial delay which is somewhat expected by today's consumers, rebuffering invokes a sudden unexpected interruption and distort the temporal video structure. Hence, rebuffering is processed differently by the human sensory system, *i.e.*, it is perceived much worse [56]. Ghadiyaram *et al.* [71] took a step further to systematically investigate the impact of rebuffering position on QoE. A rebuffering at the end tends to have a higher impact than the one at an earlier point, while the effect may not be statistically significant [46]. Another useful observation is that the overall QoE degrades with respect to the *frequency* of rebuffering events [71, 87, 142, 159]. Specifically, viewers prefer videos that have less number of freeze events (even if they are relative longer) to videos that have a sequence of short freezes through time. In addition to the position of rebuffering, motion strength has been recognized as another influencing factor of the rebuffering experience [125]. It was reported that a rebuffering occurring in a dynamic scenery significantly breaks the temporal structure of streaming video, and is thus perceived more annoying than in a stationary one. Unfortunately, the impact of motion strength has not been validated by an independent subjective experiment. Recently research discovered the presentation video quality as the third variable of rebuffering experience [12, 53, 54, 68]. An exploratory study [68] assumed that the impacts of video quality and rebuffering experience are independent and additive when investigating the combined effect of video compression, initial buffering, and rebuffering. However, the sample size in the studies is too small to make a statistically meaningful conclusion. With a more deliberately designed experiment, we

observed an interesting interaction between the presentation video quality and rebuffering in our previous study [54]. For a fixed rebuffering duration, human subjects tend to give a higher penalty to the video with a higher instantaneous video quality at the freezing frame. Further investigations confirm the interaction persists in more complex streaming scenarios [12, 53]. The phenomenon can be explained by the expectation confirmation theory [153], which suggests that subjects assess the QoE with respect to their original expectation formed by the presentation video quality and determine the extent to which their expectation is confirmed. To validate the competing hypotheses drawn from the existing studies, Ma *et al.* [127] generates visual stimuli with a novel "analysis-by-synthesis" approach. Specifically, the authors first synthesize a pair of stimuli that maximize/minimize one hypothesis while holding the other fixed. This procedure is then repeated, but with the roles of the two models reversed. Careful study of the stimuli indicates that given the same rebuffering duration, videos with higher presentation video quality consistently deliver higher overall QoE, despite the greater penalty for the rebuffering event.

Quality Adaptation Experience Studies

In contrast to the significant efforts in VQA and rebuffering experience studies, research in quality adaptation experience has drawn little attention. Although it has long been conjectured that quality adaptation has a negative impact on QoE, the hypothesis turns out to be surprisingly challenging to validate. Pioneering research on the quality adaptation experience dated back to 2003, when Zink *et al.* [245, 246] evaluated the QoE of scalable video transmission. To investigate the impact of layer change, they altered the temporal distribution of bitrate in streaming videos while fixing the average bitrate. Since then, the constant bitrate contour experimental protocol has been widely adopted in subjective QoE studies of DASH [72, 116, 125, 126, 139, 142, 150, 172, 174, 205, 214]. One common conclusion of these experiments included that QoE is negatively correlated with the magnitude of quality adaptation. Nevertheless, the proof is not technically sound as the perceptual quality is a concave function of the bitrate [221]. Specifically, a video sequence with a higher bitrate variance intrinsically possesses a lower average perceptual quality, regardless of quality adaptations. To overcome the limitations of the constant contour strategy, a few studies adopt a two-stage experiment procedure [168, 203], in which both the chunk-

level quality and overall QoE are evaluated by participants. These studies showed that viewers prefer positive over negative quality adaptations. Unfortunately, the conclusion is still questionable since the observation may be a consequence of the recency effect [79] rather than the quality adaptation direction. Somewhat surprisingly, the hypothesis was not fully justified until 2017. To address the confounding factors and better explore the space of quality adaptations, we carried out an path-analytical experiments on a large database of streaming videos [51, 52]. The experiment showed that the visual quality of a video chunk following a negative quality adaptation is generally perceived to be lower than its intrinsic quality (*i.e.*, when it is displayed independently), and the amount of penalty is correlated with the intensity of negative quality adaptation. On the other hand, positive adaptation (switching to a higher bitrate) generally receives an additional reward. This phenomenon can also be well explained by the expectation confirmation theory [153]. Aside from the basic proof, some experiments have been demonstrated that the quality adaptation experience is further affected by the adaptation type, the intrinsic quality, the content variation, and the interactions between them [51, 150].

Summary

To sum up, the existing subjective QoE studies have drawn the following key observations:

- In general, QoE is a function of video quality, rebuffering experience, and quality adaptation experience.
- Video quality is generally considered to be a monotonic function of the encoding bitrate, but the relationship strongly depends on the video content, encoder configuration, and viewing devices.
- Even when the video content, encoder configuration, and viewing devices are fixed, video quality is not strictly monotonic with respect to the encoding bitrate due to the influence of encoding spatial resolution, frame rate, and dynamic range.
- The rebuffering experience is a monotonic function of rebuffering duration and frequency.

- The rebuffering experience is further influenced by the position of rebuffering, video motion characteristics, and the presentation video quality where the freezing occurs.
- Quality adaptations influence the QoE by modifying the perceived quality of subsequent video segments.
- The quality adaptation experience is also affected by the adaptation type, the intrinsic quality, and the content variation.

2.1.2 Objective QoE Assessment Models for Streaming Videos

Although subjective QoE studies provide reliable evaluations, they are inconvenient, time-consuming and expensive. Most importantly, they are not applicable in the real-time ABR decision making. Therefore, highly accurate and low complexity objective QoE models are desirable to enable efficient design of quality-control and resource allocation protocols for media delivery systems. Thanks to the joint effort from multiple research communities such as networking, vision science, signal processing, and machine learning, there has been an accelerated development in objective QoE research in the past decade. Several design principles have emerged and have been shown to be effective at creating QoE models, many of which are well correlated with perceptual quality when tested using the current public streaming video databases [10, 11, 53].

Despite showing great promise, several outstanding challenges remain in the fundamentals of QoE research. First, a well-structured problem formulation is missing that not only provides a unified framework to understand the connections between QoE models, but also identifies potential ways for future development. Second, the multi-discipline nature of QoE research gives rise to misconceptions and ambiguities concerning some basic QoE terminologies. In particular, QoE is frequently confused with bitrate, perceptual metric, and network QoS, resulting in vague optimization goals, inconsistent psychophysical experimental protocols, and inadequate evaluation criteria. Third, many algorithms are derived in ad-hoc manner where assumptions are implicit, making it extremely challenging to fairly evaluate competing hypotheses and recognize their limitations. Fourth, while it seems obvious that a successful QoE model has to relate to the visual processing system

in some way, many methods fail to draw a connection to vision science. As a result, it is often difficult to make an intuitive sense of how and why an QoE model works. With a growing number of new QoE models emerging each year, we have seen more "symptoms" arising from the aforementioned fundamental issues.

The Bayesian theory has found profound applications in vision science by offering a principled yet simple computational framework for perception that accounts for a large number of perceptual effects and visual behaviors [108]. Meanwhile, Bayesian inference and estimation theories have been employed extensively in a wide variety of computer vision, signal processing, computer graphics, and machine learning methods [164]. In this thesis, we attempt to bridge the gap between the two, by laying out a generic conceptual framework for quantifying QoE from a Bayesian perspective. We provide a general formulation of the objective QoE problem, highlighting a branch of statistical models that underpin the existing QoE methods. We discuss two types of Bayesian networks for QoE with distinct definitions on visual quality. We also identify common source of prior information for developing artificial vision systems, and discuss a series of examples in which researchers have used a specific type of prior knowledge. Finally, we elaborate why the most frequently used bitrate-based model fails to accurately predict the perceptual QoE.

Bayesian View of Objective QoE Assessment

The goal of QoE model is to determine the subjective quality rating y given a streaming video $\mathbf{x}$. The problem can be formulated as a Bayesian inference problem, where the objective is to determine the probability distribution $p(y|\mathbf{x})$, which may be followed by a decision making process that generates a deterministic estimate of y. There are generally two distinct approaches to solving the inference problem.

The first approach firstly solves the inference problem by determining the quality level-conditional densities $p(\mathbf{x}|y)$ for each quality level y and the prior label probabilities $p(y)$. Then one can use Bayes' theorem in the form

$$p(y|\mathbf{x}) = \frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})}, \tag{2.1}$$

to find the posterior quality distribution $p(y|\mathbf{x})$. The denominator in Bayes' theorem can be found in terms of the quantities appearing in the numerator, because

$$p(\mathbf{x}) = \int p(\mathbf{x}|y)p(y)\mathrm{d}y. \tag{2.2}$$

The models generated from this approach is known as *generative models*, because by sampling from them it is possible to generate synthetic data points in the input space. However, due to the lack of training data and effective learning methods, generative models have not drawn much attention from QoE researchers. As a result, we focus on the second approach in this review.

Alternatively, the second approach aims to determine the posterior quality probabilities $p(y|\mathbf{x})$ directly. This approach is simpler in the sense that we do not need to model the streaming video space, of which we only have limited understanding. However, building an accurate model of $p(y|\mathbf{x})$ still requires sampling and performing subjective tests on all possible streaming videos, neither of which is feasible in practice. Therefore, most existing QoE models are focused on the following problem: Given a set of training data $\mathcal{D}_\mathbf{x}$ comprising $N_\mathbf{x}$ input videos (and optionally some side-information such as network characteristics) $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_{N_\mathbf{x}})$ and their corresponding target quality scores $\mathbf{y} = (y_1, ..., y_{N_\mathbf{x}})$, find a posterior quality distribution $p(y|\mathbf{x}, \mathcal{D}_\mathbf{x})$ that best approximates $p(y|\mathbf{x})$ in the HVS. It should be noted that $p(y|\mathbf{x}, \mathcal{D}_\mathbf{x})$ can be regarded as a point estimate of $p(y|\mathbf{x})$ as the latter would be fully recovered by $\int p(y|\mathbf{x}, \mathcal{D}_\mathbf{x})p(\mathcal{D}_\mathbf{x})\mathrm{d}\mathcal{D}_\mathbf{x}$ if we sample all possible data $\mathcal{D}_\mathbf{x}$. The problem is further simplified by assuming the training data are independent and identically distributed, so that the predictive distribution can be parametrized [37] as

$$p(y|\mathbf{x}, \mathcal{D}_\mathbf{x}) = \int p(y|\mathbf{x}, \boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D}_\mathbf{x})\mathrm{d}\boldsymbol{\theta}, \tag{2.3}$$

where $\boldsymbol{\theta}$, $p(y|\mathbf{x}, \boldsymbol{\theta})$ and $p(\boldsymbol{\theta}|\mathcal{D}_\mathbf{x})$ represent the parameters of the HVS model, the quality rating generation process and the posterior distribution over parameters, respectively. Given the enormous space of $\boldsymbol{\theta}$, the computation of the integral in Equation 2.3 is prohibitively expensive. As a result, a common practice is to approximate the predictive distribution $p(y|\mathbf{x}, \mathcal{D}_\mathbf{x})$ by a point estimate $p(y|\mathbf{x}, \boldsymbol{\theta}^*)$, where

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathcal{D}_\mathbf{x}) = \arg\max_{\boldsymbol{\theta}} p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})p(\boldsymbol{\theta}). \tag{2.4}$$

The specific form of the likelihood function $p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})$ is not known in practice. To fully specify the problem, it is usually assumed that the likelihood function follows a Gaussian distribution

$$p(y|\mathbf{x}, \boldsymbol{\theta}, \beta) = \mathcal{N}(y|f(\mathbf{x}; \boldsymbol{\theta}), \beta), \tag{2.5}$$

where $f(\mathbf{x}; \boldsymbol{\theta})$ and β represent the mean and variance of the Gaussian distribution, respectively. It is easy to show that the maximum likelihood solution of $\boldsymbol{\theta}$ is equivalent to the best least-square solution with respect to the MOS under this assumption.

Direct estimation of $\boldsymbol{\theta}$ [46] from a set of training data is problematic, because of the fundamental conflict between the enormous size of the streaming video space and the limited scale of affordable subjective testing. Specifically, a typical "large-scale" subjective test allows for a maximum of several hundreds or a few thousands of test videos to be rated. Given the combination of source videos, distortion types and distortion levels, realistically only a few hundreds of test videos (if not fewer) can be included, which is the case in all known subject-rated databases. By contrast, digital videos live in an extremely high dimensional space, where the dimension equals the number of pixels, which is typically in the order of billions. Therefore, a few hundreds of samples that can be evaluated in a typical subjective test are deemed to be extremely sparsely distributed in the space. Furthermore, it is difficult to justify how a few hundreds of streaming videos can provide a sufficient representation of the variations of real-world video. As a result, the fundamental problem in the objective QoE is to develop a meaningful prior parameter distribution $p(\boldsymbol{\theta})$, which encodes the configuration of the HVS.

Over the past decades, various QoE models have been developed where the key difference lies in the assumptions about the prior distribution $p(\boldsymbol{\theta})$. In general, three types of knowledge may be used for the design of QoE measures, as shown in Figure 2.2. Most systems attempt to incorporate knowledge about the HVS, which can be further divided into bottom-up knowledge and top-down assumptions. The former includes the computational models that have been developed to account for a large variety of physiological and psychophysical visual experiments [52, 81, 154]. The latter refers to those general hypotheses about the overall functionalities of the HVS [224].

Knowledge about the possible distortion processes is another important information

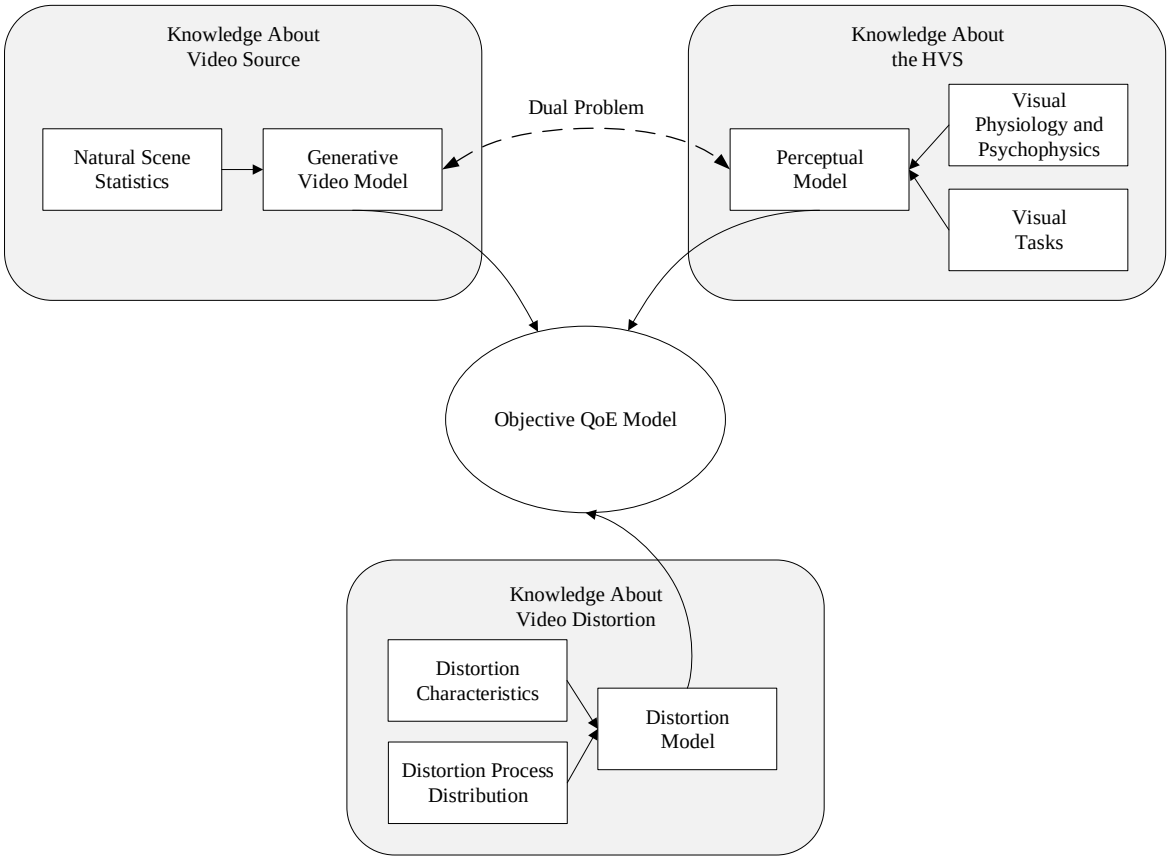


Figure 2.2: Knowledge map of objective QoE.

source in the design of objective QoE models. This type of information generally includes the appearance of certain distortion pattern and the distribution of distortion processes in practice. For example, one can explicitly construct features that are aware of particular artifacts, such as compression [54], rebuffering [87], and quality adaptation [52], and then assign penalties to these distortions. Also, it is much easier to create streaming video examples that can be used to train these models, so that more accurate QoE prediction can be achieved. This type of knowledge is typically deployed in QoE models that are designed to handle a specific artifact type.

The third type is knowledge about the visual world to which we are exposed. It es-

sentially summarizes what natural videos should, or should not, look like. It is known that there exist strong statistical regularities of the natural videos [188]. If an observed streaming video significantly violates such statistical regularities, then the video is considered unnatural and is presumably of low quality. The statistical properties of natural videos, which are often referred to as NSS, have profound impact on the research in the general-purpose visual quality assessment [223] and are still making significant impacts in the deep learning era. In computational neuroscience, it has long been conjectured that the HVS is highly adapted to the natural visual environment [13], and therefore, the modeling of natural scenes and the HVS are dual problems [185].

A Two-Layer Hierarchical Bayesian QoE Framework

The QoE prediction problem is very challenging due to the fundamental conflict between the enormous size of the streaming video space and the limited number of videos available for observation. To overcome the curse of dimensionality problem, all existing QoE measures share a common probabilistic graphic model. Specifically, the conditional quality distribution can be decomposed as

$$p(y|\mathbf{x};\boldsymbol{\theta}) = \int p(\mathbf{z}|\mathbf{x};\boldsymbol{\theta}_1)p(y|\mathbf{z};\boldsymbol{\theta}_2)\mathrm{d}\mathbf{z}, \tag{2.6}$$

where $\mathbf{z}$, $p(\mathbf{z}|\mathbf{x};\boldsymbol{\theta}_1)$, and $p(y|\mathbf{z};\boldsymbol{\theta}_2)$ are a low dimensional latent variable, a feature extractor, and a regression model, respectively. In this Bayesian network, the objective QoE model parameters $\boldsymbol{\theta}$ consist of both the parameters of the feature extractor and the regression model (*i.e.*, $\boldsymbol{\theta} = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\}$). In general, the marginalization over the latent variable $\mathbf{z}$ in (2.6) is not tractable to compute exactly. To avoid this issue, most methods consider an approximation that makes use of a point estimate instead of performing the integration over $\mathbf{z}$ in (2.6). The basic idea of the two-layer Bayesian network is to map the entire streaming video space $\mathbf{x}$ onto a space of much lower dimension $\mathbf{z}$, where, it is hoped, the QoE problem will be easier to solve. The dimensionality reduction is possible because natural videos exhibit strong statistical regularities [188] and that there are a limited number of distortion processes involved in adaptive video streaming [184]. On top of the feature extractor, the regression model $p(y|\mathbf{z};\boldsymbol{\theta}_2)$ is responsible to estimate the overall QoE score from only incomplete information.

Thanks to its conceptual simplicity and practical effectiveness, the combination of feature extractor and regression model has found ubiquitous applications in the field of video processing, computer vision, and adaptive streaming. In addition to the performance consideration, the choice of the hierarchical structure is also a natural consequence of the sequential planning problem. To be specific, some ABR algorithms explicitly look ahead for information from future chunks and select the video segment with the optimal expected QoE. At the instance of the bitrate selection, however, the objective QoE model has to estimate the perceptual quality of future chunks with only a small number of features embedded in the manifest file, when the complete information about future chunks are not available to the ABR player yet. The past decade has witnessed various instantiations of the two-layer QoE framework. Motivated by the special architecture, we will separately review feature extractors and regression models that are commonly used in the existing objective QoE models, with an emphasis on their underlying assumptions.

The enormous space of $\mathbf{x}$ suggests that a parametric model $p(y|\mathbf{x}; \boldsymbol{\theta})$ would intrinsically exhibit a very high dimension in terms of $\boldsymbol{\theta}$. For example, parameters $\boldsymbol{\theta}$ of the simplest linear model lie in a space whose dimension is equal to the dimension of input variables $\mathbf{x}$. To reduce the complexity of the problem, it is desirable to work with a quality-aware representation lying in a much lower dimensional space. The feature extractor, that qualitatively determines which piece of information in a streaming video is relevant to the QoE, plays a central role in the objective QoE model. However, the extraction of the optimal feature set is non-trivial. Care must be taken during transformation because often information is discarded, and if this information is important to the solution of the problem then the overall accuracy of the system can suffer. On the other hand, if too much information is preserved, it would require excessive training data or prior knowledge to design the subsequent regression module in order to avoid overfitting. Formally, the objective of feature extractors is to minimize the dimension of the hidden state $\mathbf{z}$ such that the conditional mutual information $\mathbb{E}_{\mathbf{z}}[I(\mathbf{x}, y|\mathbf{z})] = 0$. To reduce the dimensionality of streaming videos, all existing approaches make a priori assumptions about the transformation $p(\mathbf{z}|\mathbf{x}; \boldsymbol{\theta}_1)$, although they differ in type of source knowledge embedded in the prior distribution of the model parameters. We summarize these techniques as follows:

- Distortion Process-Based Approach: Assuming there exists a causal relationship be-

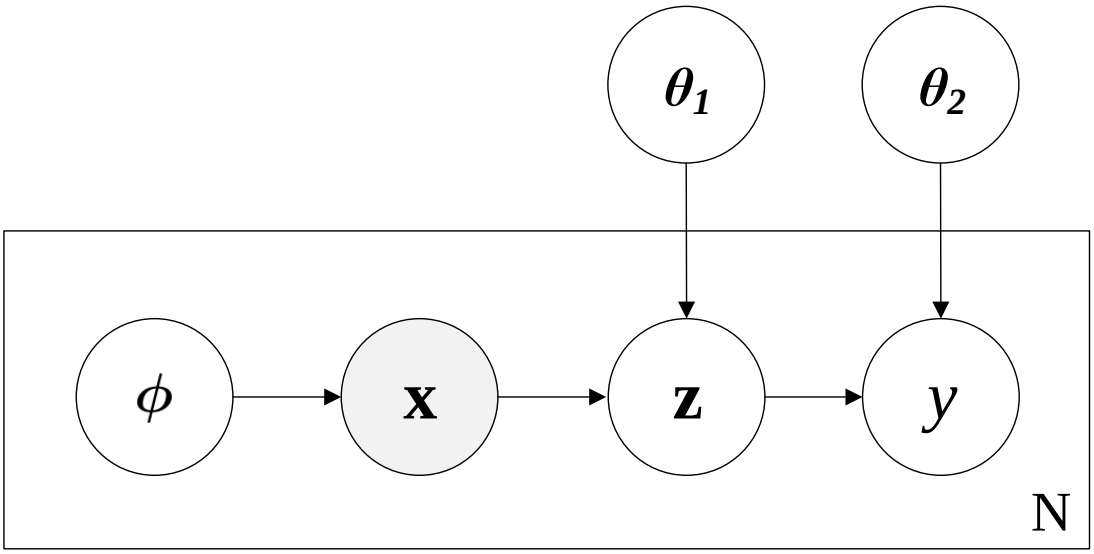


Figure 2.3: Graphical model representation of existing QoE models. The box is "plate" representing replicates. Each node represents a random variable (or group of random variables), and the links express probabilistic relationships between these variables. The observable variables are shaded in color.

tween impairments in the communication pipeline and the QoE, the QoS approach tried to identify a set of objective performance measures that correlate well with the subjective quality evaluation. A unique property of this approach is that the extracted features $\mathbf{z}$ do not depend on the visual signal $\mathbf{x}$ or its pristine counterpart, but is a function of the distortion process parametrized by $\boldsymbol{\phi}$ instead. As a result of the decoupling between the visual content and viewers' QoE, the distortion process-based feature extractors are often referred to as the QoS-based approach. Over the past decades, there have been a wide variety of QoS-based methods ranging from generic network-level features such as bit error rate, packet loss rate, network jitter, Round Trip Time (RTT), and average bandwidth [106, 136, 152, 161, 220] to application specific features such as Quantization Parameter (QP), encoding bitrate,

rebuffering duration, and bitrate variation [124, 236, 241]. Although this approach has achieved promising results in individual reports, it often struggles to deliver competitive performance in a more comprehensive benchmark including more diverse source contents, video encoders, and ABR algorithms [53]. One plausible explanation of the phenomenon is the fundamental gap between the QoS and subjective QoE. The inconsistency between these terms is made apparent by the probabilistic graphic model of the two-layer Bayesian model in Figure 2.3, where $\boldsymbol{\phi}$ encodes the set of parameters that influences the likelihood of a particular streaming video $\mathbf{x}$. Due to the presumably causal relationship, these network QoS measures $\mathbf{z}$ constitute a subset of $\boldsymbol{\phi}$, which partially determines the distribution of $\mathbf{x}$. Even with the complete set of $\boldsymbol{\phi}$ and a perfect regression model, the network QoS models correspond to the probability distribution

$$p(y|\boldsymbol{\phi}) = \int p(\mathbf{x}|\boldsymbol{\phi})p(y|\mathbf{x})\mathrm{d}\mathbf{x}. \tag{2.7}$$

This marginal distribution is generally different from the true QoE distribution $p(y|\mathbf{x})$, as the structure of $p(\mathbf{x}|\boldsymbol{\phi})$ may be extremely complicated. Perhaps a deeper reason to the failure of QoS approach resides in the conditional independence between the service performance $\boldsymbol{\phi}$ and the QoE score y. In particular, a naïve subject can consistently assess the quality of a streaming video without access to the underlying transmission channel. The fundamental gap between the QoE-based models and perceptual QoE suggests that the use of such knowledge in the feature extraction process may not be preferable.

- HVS-Based Approach: Natural video signals are inherently redundant [188]. Vision scientists have recognized for centuries that HVS understands the visual scene by building a low-dimensional internal representation of the world, which discards part of the input information. One evidence of the visual hypothesis is metamer, which refers to the phenomenon that two distinct physical objects are perceived as identical (*e.g.*, color and texture). Motivated by the functionality of HVS, HVS-based feature extractors are developed to simulate the perceptual visual encoding process. In practice, most feature extractors in this direction project each video chunk to

a single scalar, representing the presentation video quality. The general methodology, however, can be applied to generate other QoE related quantities such as visual saliency, interestingness, and perceptual motion strength [125, 243]. Depending on the underlying presentation video quality measure, these QoE models can be further divided into bottom-up approach and top-down approach [50, 224].

Bottom-up methods extract video quality representations based on the visibility of error signals. Specifically, it has been found that the HVS is relatively insensitive to certain types of visual patterns. First of all, the HVS is known to have different sensitivity to the spatial frequency content in visual stimuli. The relationship between the sensitivity of the HVS and the spatial frequency content in visual stimuli can be modeled by the CSF [14], which peaks at a spatial frequency around four cycles per degree of visual angel and drops significantly with both increasing and decreasing frequencies. Second, the presence of one signal can sometimes reduce the visibility of another video component, a phenomenon known as the contrast masking effect. In general, a masking effect is strongest when the signal and the masker have similar spatial location, frequency content, and orientations. Third, the perception of luminance obeys Weber's law, which can be expressed mathematically as $\frac{\Delta L}{L} = C$, where L is the background luminance, ΔL is the just noticeable incremental luminance over the background by the HVS, and C is a constant called the Weber fraction. Motivated by the different sensitivity of the HVS to visual stimuli, a variety of QoE feature extractors in the literature share a similar error visibility paradigm. All these artificial visual models can be decomposed into five stages, including pre-processing, CSF filtering, channel decomposition, error normalization, and error pooling [50, 224]. The final output of the computational models encodes certain quantities that are hopefully measured by part of HVS. Typical examples of the approach includes [10, 45], which are built upon VMAF [117] as the chunk-level feature extractor.

Unfortunately, HVS is too complex to be modeled precisely. Many bottom-up methods are based on simplified assumptions and limited psychophysical experiments, that may not generalized well in practice. To overcome the challenges, a different top-down approach was taken by making use of the knowledge about the overall functionality

of the HVS. The widely accepted structural similarity paradigm assumes that the HVS is highly adapted to extract structural information from the viewing field. It follows that a measurement of structural similarity (or distortion) should provide a good approximation to perceptual video quality. Pioneering the structural similarity approach, Wang *et al.* proposed to define the nonstructural distortions as those distortions that do not modify the structure of objects in the visual scene, and all other distortions to be structural distortions [224]. For example, a spatial domain implementation of the structural similarity paradigm called the Structural Similarity Index (SSIM) separates the task of similarity measurement into three independent comparisons: luminance, contrast and structure. Various QoE models [16, 54] that employ SSIM or its close variants as the chunk-level feature extraction models are essentially top-down HVS models.

- NSS Approach: It turns out that there exists a distinct way to look at the quality representation problem, *i.e.* from the video formation point of view. The information theoretic paradigm assumes that each reference video $\mathbf{x}_0$ is a sample from a very special probability distribution $p(\mathbf{x}_0)$, *i.e.*, the class of natural scenes. Most real-world distortion processes disturb these statistics and make the video signal unnatural, suggesting that each distorted video $\mathbf{x}$ comes from a distinct probability distribution $q(\mathbf{x})$. As a result, the similarity between $\mathbf{x}$ and $\mathbf{x}_0$ can be measured by some information theoretic distance/divergence between these two probability distributions. Although the use of information theoretic distances as perceptual quality seems somewhat arbitrary, there exists a non-trivial connection between the two concepts. Specifically, it has long been hypothesized that the HVS is adapted to optimally encode the visual signals [13]. Because not all signals are equally likely, it is natural to assume that the perceptual systems are geared to best process those signals that occur most frequently. Thus, the statistical properties of natural scene have a direct impact to the characteristics of the HVS. Indeed, the statistical video modeling is shown to be the dual problem of the error visibility-based perceptual models [185]. To implement this idea, one has to specify the mathematical forms of natural video distribution $p(\mathbf{x}_0; \boldsymbol{\theta}_{1_1})$, distorted video distribution $q(\mathbf{x}; \boldsymbol{\theta}_{1_2})$, and the information theoretic distance measure $d_{\text{INFO}}\big(p(\mathbf{x}_0; \boldsymbol{\theta}_{1_1}), q(\mathbf{x}; \boldsymbol{\theta}_{1_2}); \boldsymbol{\theta}_{1_3}\big)$, where we

have represented our prior knowledge about the source video and the distortion process by $\boldsymbol{\theta}_1 = \{\boldsymbol{\theta}_{1_1}, \boldsymbol{\theta}_{1_2}, \boldsymbol{\theta}_{1_3}\}$. To simplify the problem, it is often assumed that video statistics are locally homogeneous and the patches within a video are independent and identically sampled from the corresponding distribution. The probability distributions are then estimated from a stack of sub-videos within the pair of distorted and reference videos. All information theoretic VQA methods can be explained by the framework. As an initial attempt in this paradigm, the Information Fidelity Criterion [185] models the natural video distribution $p(\mathbf{x}_0; \boldsymbol{\theta}_{1_1})$ as a Gaussian Scale Mixture [217]. To derive the model for the distorted video distribution $q(\mathbf{x}; \boldsymbol{\theta}_{1_2})$, the method assumes the distortion process to consist a simple signal attenuation and additive Gaussian noise. Finally, the perceptual quality is measured by the mutual information between $p(\mathbf{x}_0; \boldsymbol{\theta}_{1_1})$ and $q(\mathbf{x}; \boldsymbol{\theta}_{1_2})$. As a close variant of the Information Fidelity Criterion, Visual Information Fidelity and its descendant VMAF approach the HVS as a "distortion channel", which introduces stationary, zero mean, additive white Gaussian noise to the videos in the wavelet domain [117, 186]. Inherited from these base quality models, the QoE models [10, 8, 45, 92] utilize prior knowledge about NSS in the feature extraction process.

Despite the demonstrated effectiveness, all the aforementioned feature extraction schemes are solely derived from prior knowledge with a deterministic form. They do not incorporate the likelihood function to adapt these features to the domain of QoE assessment. The full Bayesian feature extraction scheme is an open question to be explored in the future.

Even with the reduced dimensionality, the design of objective QoE models is still a challenging task, partly because the sequential nature of the streaming video. In particular, the dimensionality grows linearly with the number of segments, suggesting that the latent representation of each video $\mathbf{z}$ still lies in a very high dimensional space. There has been two distinct approaches to tackle the problem, both of which can be derived from the Bayesian perspective. Given a dataset of observations $\mathcal{D}_\mathbf{z}$ comprising $N_\mathbf{x}$ latent variables $\mathbf{Z} = (\mathbf{z}_1, ..., \mathbf{z}_{N_\mathbf{x}})$ and their corresponding target quality scores $\mathbf{y} = (y_1, ..., y_{N_\mathbf{x}})$, the objective of the regression model is to obtain a set of parameters $\boldsymbol{\theta}_2$ that optimizes the the posterior parameter distribution $p(\boldsymbol{\theta}_2|\mathcal{D}_\mathbf{z})$. According to the Bayes' theorem, the posterior distribution can be decomposed as $p(\boldsymbol{\theta}_2|\mathcal{D}_\mathbf{z}) \propto p(\mathcal{D}_\mathbf{z}|\boldsymbol{\theta}_2)p(\boldsymbol{\theta}_2)$. Existing QoE models can

be categorized based on their assumptions about the prior parameter distribution $p(\boldsymbol{\theta}_2)$ as follows:

- Strong Prior: Given the limited training data in the latent space, the first approach mainly relies on strong prior assumptions about $p(\boldsymbol{\theta}_2)$ to estimate the posterior distribution. To simplify the problem, four basic assumptions are commonly made. The first is that the notion of QoE can be defined locally, and that the overall QoE can be obtained by a linear combination of the chunk-level QoE scores. Typically, one makes a Markov assumption that the chunk-level quality distribution, when conditioned on its previous segment, is independent of the segments beyond the neighborhood. The second is an assumption of temporal homogeneity: the chunk-level QoE distribution is the same across all temporal positions. The two assumptions jointly suggest that

$$p(y|\mathbf{z};\boldsymbol{\theta}_2) = \mathcal{N}(y|\frac{1}{T}\sum_{t=1}^{T} y^t, \beta), \tag{2.8}$$

  where $y^t = g(\mathbf{z}^t;\boldsymbol{\theta}_2)$[1] denotes the chunk-level QoE. It should be noted that the mapping between the local latent variables $\mathbf{z}^t$ and the local QoE y^t shares a common functional form across all temporal indices. The third is an additive assumption that the impact of each dimension in $\mathbf{z}^t$ is independent from other dimensions in predicting the local QoE scores. This assumption can be mathematically expressed as $g(\{\mathbf{z}^t\}; \boldsymbol{\theta}_2) = \sum_{j=1}^{J} g_j(z^{t,j};\boldsymbol{\theta}_2^j)$, where $z^{t,j}$ and $g_j(\cdot;\boldsymbol{\theta}_2^j)$ denote the j-th dimension in $\mathbf{z}^t$ and the dimensional specific activation function, respectively. In addition to the three assumptions, most objective QoE models in this category also make assumptions about the specific form of $g(\mathbf{z}^t;\boldsymbol{\theta}_2)$ along each dimension. Initial attempts incorporated certain functions with pre-defined parameters as the latent space quality predictor. Popular choices of the activation operator include linear function [16, 124, 231, 241], exponential function [54, 86, 156, 174], and logarithmic function [191, 236]. In the case of linear function, the chunk-level QoE can be computed by $y^t = \boldsymbol{\theta}^\top \mathbf{z}^t$, where $\top$ denotes the transpose operator. Since the parameters are fixed, we have $p(\boldsymbol{\theta}_2{=}\boldsymbol{\theta}_2^*) = 1$

[1] According to the Markov assumption, the chunk-level quality y^t is a function of $\mathbf{z}^{t-1}$ and $\mathbf{z}^t$. However, by the technique of feature enrich, we can denote the feature set $\mathbf{z}^t$ at each time instance t as the aggregation of the previous chunk-level feature and the present chunk-level feature without loss of generality.

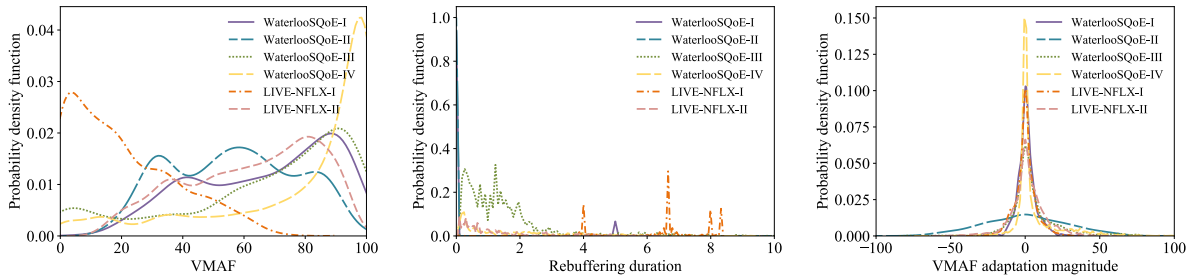
and $p(\boldsymbol{\theta}_2{=}\boldsymbol{\theta}_2{}') = 0$ for any function $\boldsymbol{\theta}_2{}' \neq \boldsymbol{\theta}_2^*$. Consequently, the posterior distribution $p(\boldsymbol{\theta}|\mathcal{D}_\mathbf{z})$ converges to the prior distribution $p(\boldsymbol{\theta})$ for any likelihood function and dataset as long as $p(\mathcal{D}_\mathbf{z}|\boldsymbol{\theta}_2^*) > 0$. Recent studies have indicated that these overly simplistic models with manually tuned parameters have achieved limited success in representing the relationship between the latent variables and the subjective QoE [10, 53]. Several efforts have put forth to improve the prediction accuracy by optimizing the adjustable model parameters $\boldsymbol{\theta}_2$ on publicly available datasets [53, 46]. The combination of data fitting and a priori assumptions have achieved highly competitive performance on existing benchmarks. Although the branch of QoE models have received broad acceptance in real-world ABR systems [3, 131, 191, 237, 241], it is important to recognize their limitations. One common drawback of the approach is that the prior distribution is often selected on the basis of mathematical convenience rather than as a reflection of any prior beliefs. The resulting strong inductive bias may manifest itself in many ways. For example, the subjective QoE response with respect to each feature can vary significantly from exponential and logarithmic functions. Generally speaking, the problem applies to all model-based QoE measures that rely on a predefined functional form. Furthermore, the additive assumption is also problematic for QoE modeling, where the impact of one latent variable is hardly independent to the other. In particular, recent experiments have illustrated that the joint impact of conventional feature pairs on the QoE is statistically significant [10, 51, 54]. The assumption becomes increasingly deficient as the dimensionality of the latent space expands. Last, while the recent investigations in the subjective local quality integration mechanism partially validated the first two assumptions [51], there are still some obstacles to truly relying on the hypotheses. Specifically, the psychophysical experiments were conducted using relatively simple patterns, such as the streaming videos with two segments. Can the models for the interactions between two chunks generalize to evaluate interactions between tens or hundreds of chunks in practice? Is this limited number of simple-stimulus experiments sufficient to build a model that can predict the visual quality of complex-structured streaming videos? The answers to these questions are currently not known, and are subject to future research.

- Non-informative Prior: Supposing HVS is too complex to understand, the second

approach aims to approximate the posterior distribution from the likelihood function $p(\mathcal{D}_{\mathbf{z}}|\boldsymbol{\theta}_2)$. With the emergence of subject-rated QoE databases [10, 11, 51, 53, 54, 46, 58, 71], the data-driven approach has dominated the objective QoE research. A broad range of statistical models such as non-linear auto-regressive model [10], neural network [189], support vector machine [8], random forest [52, 157], and Long Short-Term Memory (LSTM) [58] have been utilized to map streaming video features to subjective opinion scores. These models employ a maximum likelihood estimator

$$\boldsymbol{\theta}_2^* = \arg\max_{\boldsymbol{\theta}_2} p(\mathcal{D}_z|\boldsymbol{\theta}_2) \tag{2.9}$$

to obtain the optimal model parameters, effectively assuming a non-informative prior in the Bayesian inference problem [19]. Although these QoE models can fit arbitrary complex continuous functions [84], they often suffer from the generalization problem. Specifically, it has been observed that the performance of QoE models trained on one database reduces significantly on other benchmark datasets, largely due to the distribution mismatch in the visual content and the distortion process across datasets [10, 11, 53, 46, 71]. There are at least four sources for the generalization problem. First, in spite of the reduced dimensionality, the latent variable $\mathbf{z}$ still lives in a high dimensional space. Each streaming video is represented by a $Z \times T$-dimensional vector when chunk-level feature extractors are employed, where Z and T represent the number of chunk-level features and the total number of chunks, respectively. On the other hand, a typical "large-scale" subjective test allows for a maximum of several hundreds or a few thousands of test videos to be rated. Given the enormous space of latent variables, a few thousands of subject-rated samples are deemed to be extremely sparsely distributed in the space. Second, the learning-based models assume that the training samples and testing samples come from the same distribution. However, the assumption has never been justified in the existing studies and may hardly hold in practice. A motivating example is shown in Figure 2.4, where the probability density functions of video presentation quality measured by a state-of-the-art VQA model VMAF [117], rebuffering duration, and quality adaptation magnitude in six publicly available streaming QoE datasets are presented. Clearly, there is significant variability on the characteristics of streaming videos across different datasets, suggesting that an objective QoE model optimized on a simple dataset such as Waterloo Streaming



(a) Distribution of VMAF (b) Distribution of rebuffering duration (c) Distribution of adaptation magnitude

Figure 2.4: There exists significant variation on the characteristics of streaming videos, evident by the distributions of (a) VMAF, (b) rebuffering duration, and (c) adaptation magnitude in six publicly available datasets.

QoE Database-I (WaterlooSQoE-I) [54] may yield very poor predictions on complex datasets as WaterlooSQoE-III [53], WaterlooSQoE-IV [43], and LIVE-NFLX-II [11], and vice versa. The streaming video probability density estimation is further complicated by the concept drift problem [64], where the characteristics of streaming video changes over time. For example, the drift in streaming video distribution may arise from the advancement of video acquisition [99, 105, 149], compression [36, 147, 208], transmission [16, 93, 102, 131, 191, 241], and reproduction systems [118, 146, 227], and the steady rise in viewers' expectation on video quality [52, 153]. Third, the maximum likelihood estimator generally assumes that each ($\mathbf{z}$, y) pair in the training set $\mathcal{D}_{\mathbf{z}}$ is independent and identically distributed. In practice, however, the existing QoE datasets typically generate multiple streaming videos for each reference video to cover the diversity of distortion processes, suggesting that the training data are not independent and identically distributed. Fourth, the consistency of subjective QoE ratings among streaming video databases is only moderate due to drastically different experimental conditions. Strictly speaking, the quality ratings of a streaming video $\mathbf{x}$ collected from a subjective experiment are essentially samples from a context conditional quality distribution $p(y|\mathbf{x}, \mathbf{t})$, where $\mathbf{t}$ encodes the information about experiment environment, instruction, training process, presentation order, and

experiment protocol. As a result, the subjective quality ratings obtained from different experiments cannot be simply aggregated into a larger QoE dataset $p(y|\mathbf{x})$. These data challenges constantly arise in QoE research and will remain a challenging issue in the future.

One common drawback of both approaches is the lack of perceptually meaningful prior distributions. In particular, none of these models make use of the knowledge about natural videos, distortion processes, and the HVS, despite the plethora of dedicated subjective experiments over the past decade. It remains to be seen how much improvement can be achieved with these informative priors in the Bayesian framework.

2.2 Throughput Prediction

The network throughput directly determines the download time of each video chunk and the buffer occupancy, which further influences the viewers' QoE. Mathematically, the relationship between the rebuffering duration τ_k when downloading the k-th chunk and the instantaneous throughput c_t can be described by

$$\begin{aligned} c_k &= \frac{1}{t_{k+1} - t_k - \Delta t_k} \int_{t_k}^{t_{k+1}-\Delta t_k} c_t \mathrm{d}t, \\ \tau_k &= \max(\frac{r_k}{c_k} - b_k - \Delta t_k, 0), \end{aligned} \tag{2.10}$$

where t_k, t_{k+1}, Δt_k, c_k, r_k, and b_k represent the timestamp that the k-th chunk downloading starts, the timestamp the download finishes, the waiting time such as RTT, the average throughput during the download period, the bit count of the k-th chunk, and the buffer occupancy before the download starts, respectively. If c_t can be known precisely, then the rebuffering duration and the overall QoE can be computed deterministically. Solving the ABR problem in (1.1) corresponds to finding the bitrate trajectory with the maximum reward in a deterministic environment, which can be easily solved by traditional dynamic programming algorithm at the beginning of each streaming session. On the other hand, in case that one does not have any knowledge about c_t, the download time $\frac{r_k}{c_k}$ would follow a uniform distribution. To reduce the probability of rebuffering, an ABR algorithm has

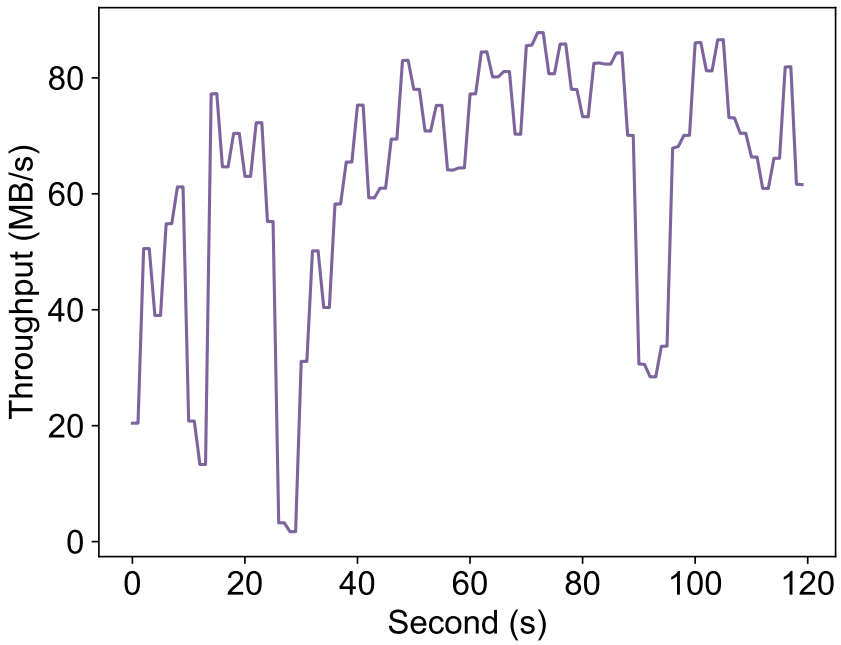


Figure 2.5: Video streaming clients experience highly variable end-to-end throughput.

to make conservative bitrate decisions, resulting in sub-optimal QoE. Clearly, an accurate throughput predictor may improve the performance of ABR algorithms.

Unfortunately, accuracy throughput prediction can be extremely challenging. The difficulty may arise from five aspects. First of all, regardless of the underlying technology or the transport protocol used for any content transmission, the available bandwidth is highly variable in time. Figure 2.5 is a sample trace reported by an ABR video player, showing how the measured throughput varies wildly from 82 MB/s to 500 kB/s in a very short period. Each point in the figure represents the average throughput when downloading a video chunk. Apparently, one can hardly extract simple patterns such as symmetry, periodicity, and smoothness in the design of throughput prediction models. Indeed, ABR researchers often find it difficult to identify which piece of information is relevant and determine an optimal parametrization for the throughput predictor. The substantial variation can be caused by many factors, such as WiFi interference, congestion in the network, congestion in the client (*e.g.* anti-virus software scanning incoming HTTP traffic), Transmission Control

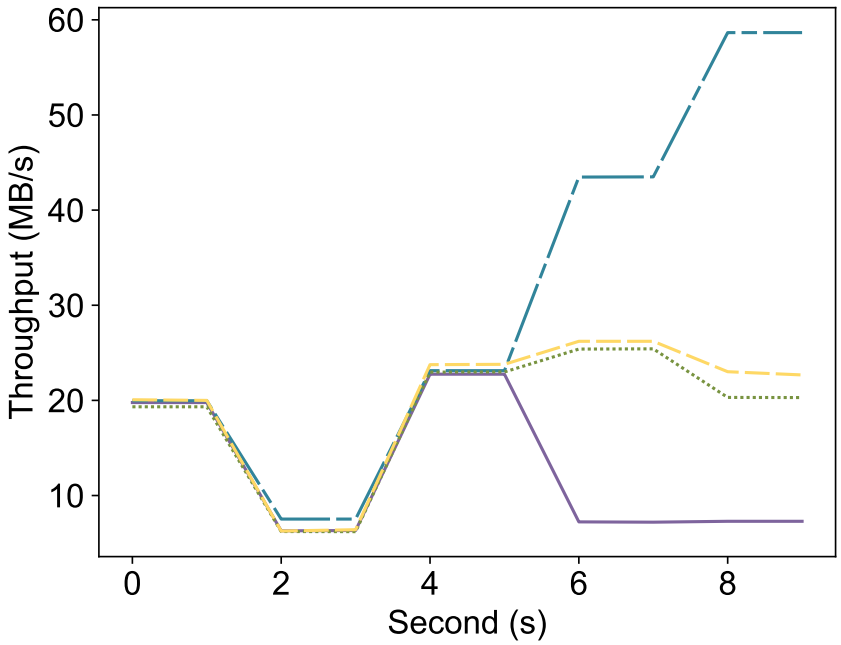


Figure 2.6: The time-varying throughput is a multi-modal stochastic process.

Protocol (TCP) slow start, TCP congestion control, or congestion at an overloaded video server. The wide variety of influencing factors underlie the second challenge in the throughput prediction. In particular, many influencing factors such as the server congestion, the number of competing players in a bottleneck link, the WiFi interference, and the status of anti-virus software are not observable to the ABR player. To accurately predict future throughput, the model also has to infer the hidden states, whose dimension and value are not available in practice, with a high precision. Third, the time-varying throughput is a stochastic process by nature. Figure 2.6 shows a few real-world throughput traces with similar characteristics at the first five seconds. The throughput traces vary significantly starting at the sixth second, suggesting that the past observation conditional probability distribution may be multi-modal. The approximation of multi-modal probability distributions usually involves sophisticated optimization techniques [19]. Another cause of the inability of a ABR player to estimate its fair share of available bandwidth is the discrete nature of HAS traffic pattern. Specifically, an ABR player can only reserve a fixed amount

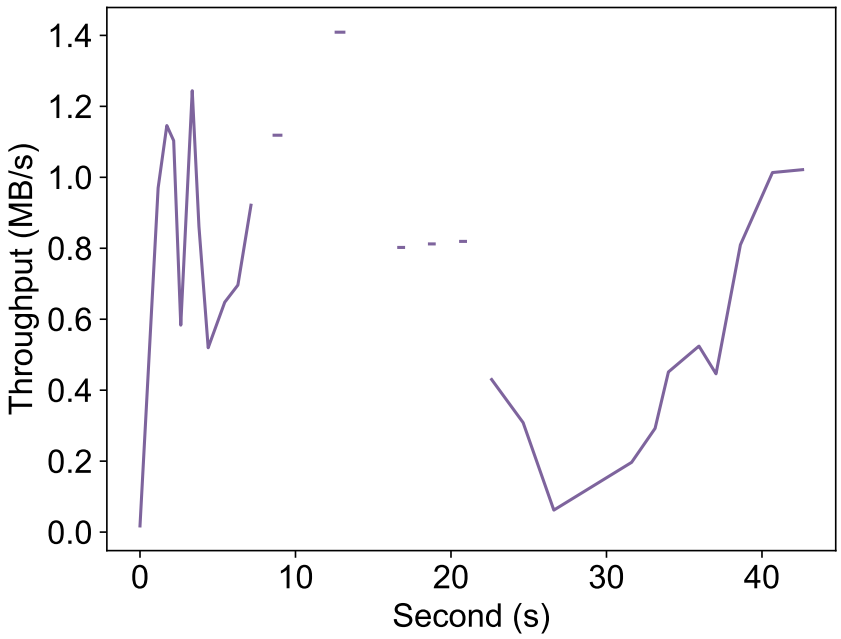


Figure 2.7: The time-varying throughput is a multi-modal stochastic process.

of unplayed videos in the buffer due to the memory constraints. In scenarios where the playback buffer cannot accommodate video from an additional chunk download, the ABR player pauses requests for a certain period of time before retrying. Figure 2.7 illustrates an example of such ON-OFF traffic pattern, from which we can observe that the average observed throughput is not evenly spaced. The special traffic pattern makes bandwidth estimation techniques that are dedicated to network traffic with fluid flow characteristics inadequate. Fifth, the four challenges are further exacerbated by the demand in long-term bandwidth estimation. In particular, state-of-the-art ABR algorithms rely on multi-step throughput prediction in order to perform a global reward optimization. The uncertainty in the throughput at each timestamp has a cascading effect to the subsequent state prediction accuracy.

2.2.1 Bayesian View of Throughput Prediction

Without loss of generality, the objective of throughput predictor at each instance t is to estimate K-step future throughput $\{c_j\}_{j=t+1}^{t+K}$ based on the past throughput observations $\{c_i\}_{i=1}^{t}$. For simplicity, we denote $\mathbf{c}_{t+1}^{t+K} = \{c_j\}_{j=t+1}^{t+K}$. Motivated by the stochastic nature of throughput and our discussion in reward function, we can also formulate the throughput prediction problem as a Bayesian inference problem, where the goal is to determine the probability distribution $p(\mathbf{c}_{t+1}^{t+K}|\mathbf{c}_1^t)$, $\forall t \in [1, T]$. Some ABR algorithms that are built upon deterministic throughput prediction append a decision making process to generate a point estimate c_i for each $i \in [t+1, t+K]$. In general, there are two approaches to solving the Bayesian inference problem.

The first approach starts with approximating the joint probability distribution $p(\mathbf{c}_1^T)$. At each state t, one can utilize Bayes' theorem

$$p(\mathbf{c}_{t+1}^{t+K}|\mathbf{c}_1^t) = \frac{p(\mathbf{c}_1^{t+K})}{p(\mathbf{c}_1^t)} \tag{2.11}$$

to estimate the past state conditional distribution. The throughput distribution in the denominator can be determined by marginalizing over the throughput of future states

$$p(\mathbf{c}_1^t) = \int_{\mathbf{c}_{t+1}^T} p(\mathbf{c}_1^T) \mathrm{d}\mathbf{c}_{t+1}^T. \tag{2.12}$$

The numerator can be obtained similarly. However, there have been limited studies in this direction, possibly because of the computationally inefficient marginalization.

The second approach directly estimates the state conditional probability distributions $p(\mathbf{c}_{t+1}^{t+K}|\mathbf{c}_1^t)$, $\forall t \in [1, T]$. This approach alleviates the burdensome computation of marginalization at the cost of providing a prediction model for each t. However, building accurate models of $p(\mathbf{c}_{t+1}^{t+K}|\mathbf{c}_1^t)$ still requires repetitively sampling every single state, which grows exponentially with t. Therefore, most existing throughput predictors solve the following problem: Given a set of training data $\mathcal{D}_\mathbf{c}$ comprising $N_\mathbf{c}$ throughput traces (and optionally some side-information such as Internet Protocol (IP) address and TCP state) $\mathbf{C} = (\mathbf{c}_1, ..., \mathbf{c}_{N_\mathbf{c}})$, find a posterior probability distribution $p(\mathbf{c}_{t+1}^{t+K}|\mathbf{c}_1^t, \mathcal{D}_\mathbf{c})$ for each $t \in [1, T]$ that best approximates $p(\mathbf{c}_{t+1}^{t+K}|\mathbf{c}_1^t)$ in the real-world network. Again, $p(\mathbf{c}_{t+1}^{t+K}|\mathbf{c}_1^t, \mathcal{D}_\mathbf{c})$ is essentially a point

estimate of $p(\mathbf{c}_{t+1}^{t+K}|\mathbf{c}_1^t)$ since the ground-truth distribution can be obtained by integrating the former distribution over all realistic dataset $\mathcal{D}_\mathbf{c}$. If we further assume the training data are independent and identically distributed, then the predictive distribution for instance t can be parametrized [37] as

$$p(\mathbf{c}_{t+1}^{t+K}|\mathbf{c}_1^t, \mathcal{D}_\mathbf{x}) = \int p(\mathbf{c}_{t+1}^{t+K}|\mathbf{c}_1^t, \boldsymbol{\theta}_t) p(\boldsymbol{\theta}_t|\mathcal{D}_\mathbf{c}) \mathrm{d}\boldsymbol{\theta}_t, \tag{2.13}$$

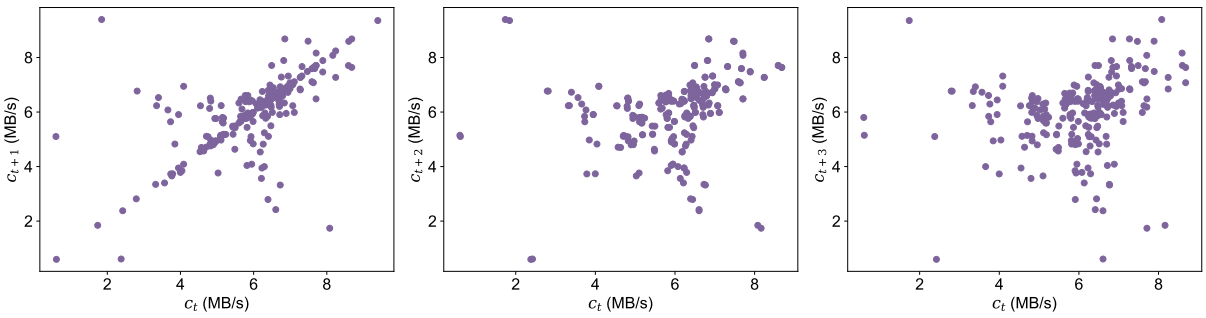
where $\boldsymbol{\theta}_t$, $p(\mathbf{c}_{t+1}^{t+K}|\mathbf{c}_1^t, \boldsymbol{\theta}_t)$, and $p(\boldsymbol{\theta}_t|\mathcal{D}_\mathbf{c})$ represent the parameters of t-th order throughput prediction model, the future throughput generation process, and the posterior distribution over parameters, respectively. Generally, one needs to estimate the time dependent model parameter $\boldsymbol{\theta}_t$ for each t. The sequential nature of throughput model combined with the enormous space of $\boldsymbol{\theta}_t$ suggests that the computation of the integral in (2.13) is prohibitively expensive. In practice, algorithm develops usually make use of a point estimate $p(\mathbf{c}_{t+1}^{t+K}|\mathbf{c}_1^t, \boldsymbol{\theta}_t^*)$ instead of performing in integral, where

$$\boldsymbol{\theta}_t^* = \arg\max_{\boldsymbol{\theta}_t} p(\boldsymbol{\theta}_t|\mathcal{D}_\mathbf{c}) = \arg\max_{\boldsymbol{\theta}_t} p(\mathbf{c}_{t+1}^{t+K}|\mathbf{c}_1^t, \boldsymbol{\theta}_t) p(\boldsymbol{\theta}_t). \tag{2.14}$$

Although realistic throughput data can be collected at low cost and at scale, direct estimation of $\boldsymbol{\theta}_t$ from a set of training data can still be challenging. In particular, a streaming session may last for hours, suggesting that the dimension of throughput data is in the order of thousands. To prevent overfitting and reduce the complexity, all existing throughput predictors make a priori assumptions about the form of $\boldsymbol{\theta}_t$. In the subsequent section, we introduce a wide range of throughput predictors from the naïve nearest neighbour predictor to state-of-the-art neural network-based models and identify their underlying assumptions.

2.2.2 Existing Throughput Prediction Models

Despite the apparent benefits from accurate resource estimation, throughput prediction has only become an active research topic since the past few years. Various network capacity models have been developed, which can be roughly categorized into model-based and data-driven methods depending on the usage of likelihood function $p(\mathbf{c}_{t+1}^{t+K}|\mathbf{c}_1^t, \boldsymbol{\theta}_t)$. We summarize these techniques in the subsequent sections.



(a) (b) (c)

Figure 2.8: Joint distributions of throughput values separated by three different temporal distances.

Model-based Methods

Due to the lack of training data, pioneering works in throughput prediction have to make a series of simplifications and assumptions about the network characteristics. Specifically, all model-based methods employ a fixed parameterization to model network dynamics without relying on the observed data, although they differ in the detailed assumptions. Here we present the most typical throughput predictors in this category.

- Stateless Predictor: The simplest bandwidth predictor assumes that the future throughput are independent of the observed data in the past [93]. It further posits that the network characteristics is constant across all timestamps. Specifically, the stateless throughput predictor can be described by $\boldsymbol{\theta}_t = \boldsymbol{\theta}$ for all time instance t, where $\boldsymbol{\theta}$ is the prior belief of the most probable capacity value. The history independent model corresponds to a particular joint distribution of throughput $p(\mathbf{c}_1^T|\boldsymbol{\theta}) = \mathcal{N}(\mathbf{c}|\boldsymbol{\theta}, I_T)$, where $\boldsymbol{\theta}$ is a T dimensional vector with each entry being $\boldsymbol{\theta}$ and I_T represents a $T \times T$ identity matrix.

- Nearest Neighbourhood Predictor: Even from a casual inspection of throughput traces, one can see that neighboring temporal locations are highly correlated (*e.g.*, the traces in Figure 2.6 and Figure 2.7). This is reflected in the scatter plot of pairs of throughput values in Figure 2.8 (a). Motivated by the observation, algorithm

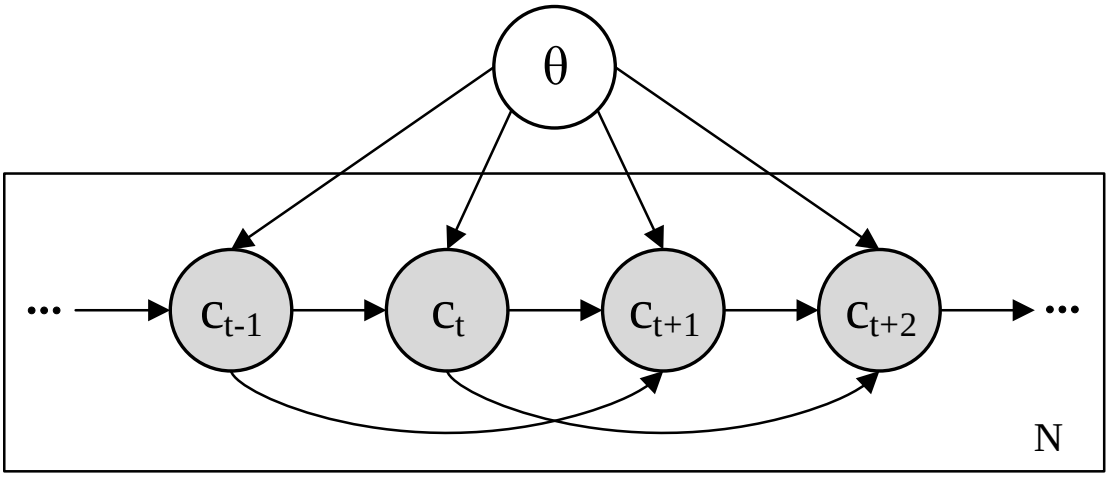


Figure 2.9: Graphical model representation of most existing throughput prediction models. The box is "plate" representing replicates. Each node represents a random variable (or group of random variables), and the links express probabilistic relationships between these variables. The observable variables are shaded in color.

developers replace the independence assumption with a Markov assumption. In particular, it is often assumed that the future throughput can be solely determined by the immediate previous throughput observation. Mathematically, the deterministic estimator can be derived from a Gauss Markov throughput distribution

$$\begin{aligned} p(c_{t+1}|\mathbf{c}_1^t, \boldsymbol{\theta}_t) &= p(c_{t+1}|c_t, \boldsymbol{\theta}_t) \\ &= \mathcal{N}(c_{t+1}|c_t, \beta_t), \end{aligned} \tag{2.15}$$

where β_t is the standard deviation of the Gaussian distribution. To further reduce the complexity, one usually makes a stationary assumption such that $p(c_{t+1}|c_t, \boldsymbol{\theta}) = p(c_{t+2}|c_{t+1}, \boldsymbol{\theta})$. Note that we have dropped the dependency of model parameters $\boldsymbol{\theta}$ on the temporal index t. The throughput in the interval $[t+1, t+K]$ can be predicted in a auto-regressive fashion.

- Arithmetic Mean: We have several observations from Figure 2.8. First, although

the throughput at $t+1$ is highly correlated with its nearest neighbour in time, the network capacity occasionally experiences bursty traffic as a consequence of the time-varying nature of the available bandwidth, or the dynamics of TCP. The instant throughput derived from a single chunk is prone to such short-term fluctuations. Second, the throughput at instance t can influence the epoch beyond its nearest neighbour $t+1$. Based on the two observations, many studies proposed to estimate the future bandwidth by averaging of the throughput observations over a horizon H [63, 122]. Following this assumption, the throughput at time $t+1$ is generally an averaging function of the previous H observations $f(\mathbf{c}_{t-H+1}^{t};\boldsymbol{\theta})$. The H-step Markov assumption combined with Gaussian assumption gives the conditional throughput distribution of form

$$\begin{aligned} p(c_{t+1}|\mathbf{c}_1^t,\boldsymbol{\theta}) &= p(c_{t+1}|\mathbf{c}_{t-H+1}^t,\boldsymbol{\theta}) \\ &= \mathcal{N}(c_{t+1}|f(\mathbf{c}_{t-H+1}^t;\boldsymbol{\theta}),\beta), \end{aligned} \tag{2.16}$$

where $f(\mathbf{c}_{t-H+1}^{t};\boldsymbol{\theta})$ is an averaging function of the previous H observations. The arithmetic mean throughput predictor further assumes the averaging operator is a linear function

$$f(\mathbf{c}_{t-H+1}^t;\boldsymbol{\theta}) = \frac{1}{H}\sum_{i=t-H+1}^{t} c_i. \tag{2.17}$$

Figure 2.9 shows the probabilistic graphic model of a 2-step Markov throughput predictor.

- Harmonic Mean: Owing to the discrete nature of video segment transmission in adaptive streaming, the throughput observations are not evenly spaced in time. To this end, Jiang *et al.* [102] proposed a harmonic mean-based throughput prediction model as an alternative to the arithmetic mean. The harmonic mean corresponds to a variant of the arithmetic mean-based throughput predictor from a Bayesian perspective, where the averaging function in (2.17) are replaced by

$$f(\mathbf{c}_{t-H+1}^t;\boldsymbol{\theta}) = \frac{H}{\sum_{i=t-H+1}^{t}\frac{1}{c_i}}. \tag{2.18}$$

The reason for using this approach is twofold. First, the harmonic mean is more appropriate when we want to compute the average of rates which is the case with

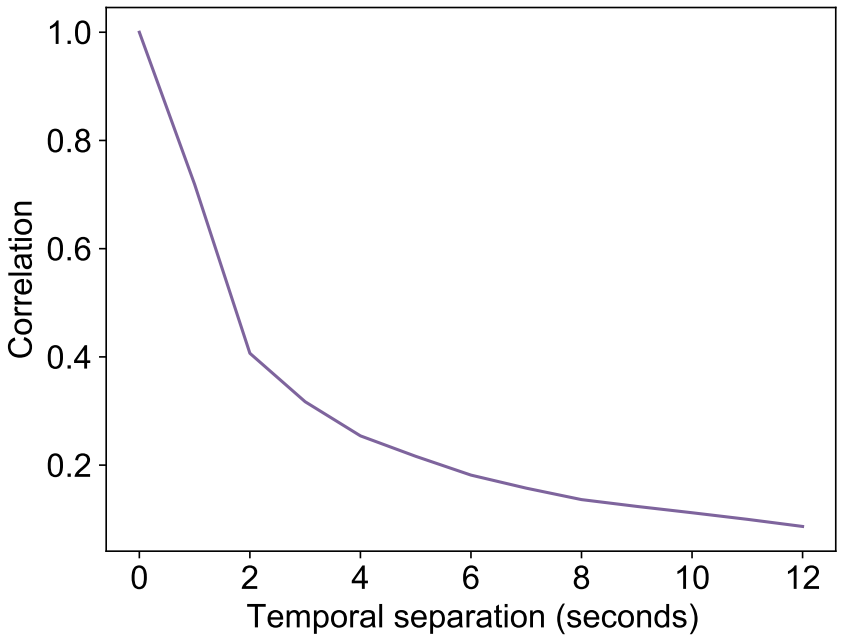


Figure 2.10: Auto-correlation function of network throughput.

throughput estimation. Second, it is also more robust to larger outliers [102]. Thanks to its simplicity, the harmonic mean-based model has become the default throughput predictor in many ABR algorithms.

- Exponential Weighted Moving Average: The exponential weighted moving average model is another variant of the arithmetic mean throughput predictor [16, 119]. This model assigns different weight to the previous throughput observations in (2.17), based on the observation that the correlation of the throughput at two instances is a function of the relative position. From the examples in Figure 2.10, one can see that the strength of the correlation falls exponentially with respect to distance.

Data-driven Methods

Although model-based throughput predictors receive ubiquitous acceptance in real-world ABR systems, it is crucial to realize their shortcomings. A summary of some of the

potential problems is as follows:

- Most model-based network capacity models are based on linear operators that are derived from casual observations in a limited number of throughput traces. This is problematic for two reasons. First, the Internet consists of many non-linear units such as TCP congestion control and WiFi interference. Second, the network traces used to derive the prediction models are usually collected from a single streaming environment, whereas realistic throughput variability is much more significant across different connection types, network protocols, and spatial locations. As a result, the generalization capability of these models remains limited.

- Even if the throughput characteristic is linear, the manually selected parameters may deviate significantly from the realistic setting. Furthermore, the adaptation of these models to new environment (*e.g.*, from 3G to 4G network) involves cumbersome parameter tuning that is expensive and time-consuming.

- To reduce the complexity, model-based approach makes Markov assumption about the throughput series. According to a recent study [200], however, the evolution of throughput exhibits long-term dependency, and may even be affected by initial throughput conditions and throughput evolution patterns. If the relevant piece of information to be remembered falls outside the history window, the models with fixed history memory cannot use it.

- Most models from the first approach can only produce a uni-modal probabilistic prediction or simply a deterministic estimate of future throughput. However, we can see from Figure 2.6 that the conditional distribution may be multi-modal in certain situations. For example, there exists uncertainty about how many players will join the bottleneck to compete for bandwidth at the next time instance.

To overcome these challenges, a different approach was taken by exploiting the knowledge from realistic network traces. Specifically, the prediction engines in the second category make use of the maximum a posteriori estimator in (2.14) to obtain the most plausible model parameters. Over the past decade, the maximum a posteriori approach has

dominated the field of network characterization by delivering unparallel prediction accuracy [237]. Theoretically, the new generation of throughput prediction methods achieved the progress by gradually lifting overly simplistic assumptions made by model-based approaches. We briefly present the evolution of the data-driven throughput prediction models.

- Linear Regression: The linear regression model is a close variant of the arithmetic mean-based predictor in (2.17), where the averaging function has a linear form

$$f(\mathbf{c}_{t-H+1}^{t}; \boldsymbol{\theta}) = \boldsymbol{\theta}^{\top} \mathbf{c}_{t-H+1}^{t}. \tag{2.19}$$

  Assuming a uniform prior distribution, the optimal parameter $\boldsymbol{\theta}$ is determined by maximizing 2.14 across a corpus of realistic throughput traces. The learning-based linear model is shown to achieve a better prediction accuracy in recent studies [102, 200]. However, as the simplest implementation of the data-driven approach, the linear regression model still shares many common assumptions with the traditional approach such as linearity, Markovian, Gaussian, and stationarity.

- Support Vector Regression: A linear function $f(\mathbf{c}_{t-H+1}^{t}; \boldsymbol{\theta})$ is fairly restrictive and may not be able to describe the true function. To alleviate the linear assumption, a few studies proposed to use sophisticated machine learning methods for throughput prediction, with Support Vector Regression (SVR) [135] being the most representative model. SVR enriches the model capability by augmenting the feature vector $\mathbf{c}_{t-H+1}^{t}$ with non-linear bases derived from $\mathbf{c}_{t-H+1}^{t}$. For example, if $H = 2$ (*i.e.*, $\mathbf{c}_{t-1}^{t} = (c_{t-1}, c_t)$), one can augment it with $\phi(\mathbf{c}_{t-1}^{t}) = (c_{t-1}, c_{t-1}^2, c_{t-1}c_t, c_t, c_t^2)$. The linear regression in the augmented feature space $f(\mathbf{c}_{t-H+1}^{t}; \boldsymbol{\theta}) = \boldsymbol{\theta}^{\top}\phi(\mathbf{c}_{t-H+1}^{t})$ then produces a non-linear fit in the original feature space. Note $\boldsymbol{\theta}$ has more dimensions than before. The more dimensions $\phi(\mathbf{c}_{t-H+1}^{t})$ has, the more expressive f becomes. To avoid overfitting, SVR also imposes a zero-mean Gaussian prior over the parameter $\boldsymbol{\theta}$, as opposed to the uniform distribution in linear regression.

- Multi-Layer Perceptron: There are at least two limitations with the SVR throughput prediction model. First, the feature extraction function ϕ is selected somewhat arbitrarily in SVR, which may be sub-optimal in the regression task. Second, SVR

still produces a point estimate to the future throughput from a uni-modal probability distribution. To overcome these problems, a neural network-based throughput predictor was proposed recently [237]. Using a variant of stochastic gradient descent, the model end-to-end optimizes parameters for both the feature extractor and the regression model. By uniformly quantizing the output space and minimizing a divergence measure, the neural network-based prediction engine can model a wide range of throughput probability distributions $p(c_{t+1}|\mathbf{c}_{t-H+1}^{t},\ \boldsymbol{\theta})$[2]. This particular parametrization corresponds to a Gaussian mixture model for throughput density function

$$p(c_{t+1}|\mathbf{c}_{t-H+1}^{t},\boldsymbol{\theta}) = \int p(\mathbf{h}|\mathbf{c}_{t-H+1}^{t};\boldsymbol{\theta}_1)p(c_{t+1}|\mathbf{h},\mathbf{c}_{t-H+1}^{t},\boldsymbol{\theta}_2)\mathrm{d}\mathbf{h}, \tag{2.20}$$

where $\mathbf{h}$ and $p(c_{t+1}|\mathbf{h},\ \mathbf{c}_{t-H+1}^{t},\ \boldsymbol{\theta}_2)$ represent the mixture variable and a Gaussian distribution, respectively. In particular, the Gaussian distribution $p(c_{t+1}|\mathbf{h},\ \mathbf{c}_{t-H+1}^{t},\ \boldsymbol{\theta}_2)$ is pre-defined by the quantization levels such that the mean of each individual Gaussian component lies at the center of each throughput bin. On the other hand, the mixture distribution $p(\mathbf{h}|\mathbf{c}_{t-H+1}^{t};\ \boldsymbol{\theta}_1)$ is optimized by the maximum likelihood estimator. To produce the throughput prediction for the next K chunks, K models are learnt separately in practice. However, the combination of chunk-level throughput prediction and the decoupling of temporal throughput characteristics is fundamentally flawed since the average throughput level during the download of the $t+2$-th chunk is a function of the chunk size of the $t+1$-th video segment.

- Hidden Markov Model: Despite the demonstrated success, all aforementioned throughput density models are Markovian. In general, however, the network throughput evolution may express long-term dependency. Furthermore, many variables that directly influence the throughput level are not observable to the ABR player. If these variables are available to the model, or at least can be estimated, we expect the prediction model would enjoy a substantial improvement. Motivated by the two observations,

[2]In the original paper, the multi-layer perceptron is optimized to predict the download time given the current state observations and the target bitrate level. Since for a fixed target chunk size, there exists a one-to-one mapping between the download time and throughput value. Without loss of generality, the download time prediction model can also be considered as a throughput predictor.

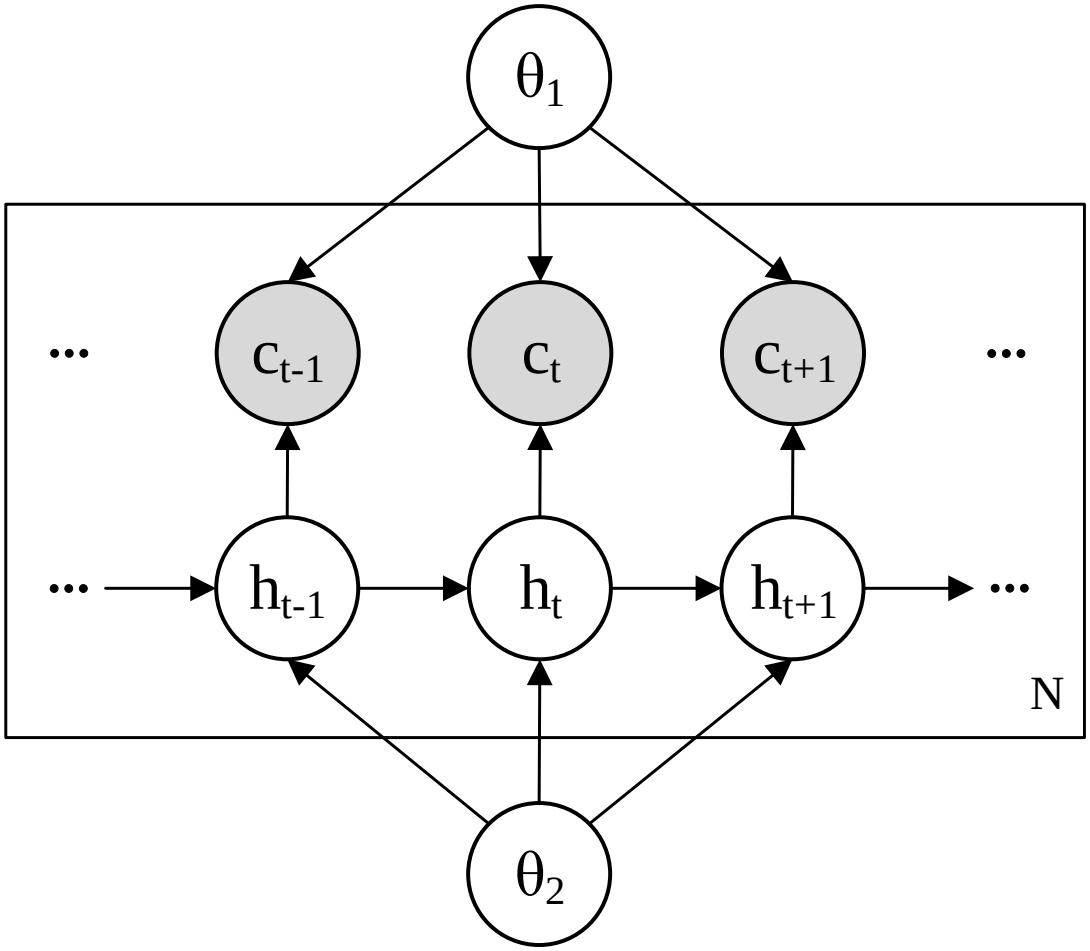


Figure 2.11: Graphical model representation of the hidden Markov model-based throughput generation process. The box is "plate" representing replicates. Each node represents a random variable (or group of random variables), and the links express probabilistic relationships between these variables. The observable variables are shaded in color.

Sun *et al.* proposed to characterize the throughput density with a Hidden Markov Model (HMM) [200]. The conditional throughput distribution can be mathematically

expressed by

$$p(c_{t+1}|\mathbf{c}_1^t, \boldsymbol{\theta}) = p(c_{t+1}|\mathbf{h}_{t+1}, \boldsymbol{\theta}_1)$$
$$p(\mathbf{h}_{t+1}|\mathbf{h}_1^t; \boldsymbol{\theta}_2) = p(\mathbf{h}_{t+1}|\mathbf{h}_t; \boldsymbol{\theta}_2), \tag{2.21}$$

where $p(c_{t+1}|\mathbf{h}_{t+1}, \boldsymbol{\theta}_1)$ and $p(\mathbf{h}_{t+1}|\mathbf{h}_t; \boldsymbol{\theta}_1)$ encode the emission probability distribution and the Markov state transition probability distribution. Note that we have implicitly assumed the distribution to be stationary, as the model parameters do not depend on the index t. Figure 2.11 presents the probabilistic graphic model of the HMM. For simplicity, the emission probability distribution is assumed to be Gaussian. The emission density function, the transition probability distribution, and the initial state probability distribution can be optimized by the expectation maximization algorithm [19] in a data-driven fashion. The model achieves state-of-the-art accuracy, reducing the median prediction error by ∼50% comparing to other baseline solutions [200]. One plausible explanation of the superior performance is that HMM can effectively lift the observation space Markov assumption and the Gaussian assumption. In particular, the model is capable of capturing the long-term dependency of throughput evolution since the hidden state $\mathbf{h}_{t+1}$ is a function of all previous observations $\mathbf{c}_1^t$. In addition, HMM can well characterize multi-modal probability distributions. However, it should be noted that the HMM-based prediction engine is still Markovian. For example, considering a scenario where two ABR players are sharing a bottleneck link. One of the players has reached the upper limit of the buffer occupancy such that it experiences the ON-OFF download pattern. The chance of the flow transitioning from asleep to awake increases the longer the player has been in the asleep state. In other words, the time-varying hidden state in another player is not Markovian.

A summary of existing throughput predictors is given in Table 2.1. In essence, we have witnessed a transition from linear, short memory, deterministic, and model-based throughput predictor to non-linear, long term, probabilistic, and data-driven throughput density functions. Nevertheless, it can be observed that all prediction engines retain the Markov and stationary assumptions, which may not hold in practice. The two assumptions, if addressed properly, may lead to further improvement in prediction accuracy.

Table 2.1: Summary of adaptive bitrate streaming evaluation studies. Abbreviations: NN, nearest neighbourhood; AM, arithmetic mean; HM, harmonic mean; EWMA, exponential weighted moving average; SVR, support vector regression; HMM, hidden Markov model; MLP, multi-layer perceptron.

Model	Year	Linear	Markov	Gaussian	Stationarity	Trainable
NN	2012	✓	✓	✓	✓	✗
AM	2012	✓	✓	✓	✓	✗
HM	2014	✓	✓	✓	✓	✗
EWMA	2007	✓	✓	✓	✓	✗
Linear	2014	✓	✓	✓	✓	✓
SVR	2007	✗	✓	✓	✓	✓
HMM	2016	✗	✓	✗	✓	✓
MLP	2020	✗	✓	✗	✓	✓

2.3 Bitrate Selector

The bitrate selector, often referred to as adaptation logic and switching logic, is the module within ABR schemes that determines the profile of the next chunk to be requested. Taking the output from resource estimation module, the reward function, and the future chunk presentation as inputs, bitrate selectors aim to return a representation which optimizes the expected total reward.

Although the target seems straight-forward, the design of adaptation functions faces three primary challenges.

- The bitrate selection for the next chunk has a cascading effect on the state of video player. For instance, selecting a high bitrate may deplete the player buffer and restrict the feasible bitrate range of subsequent chunks to avoid rebuffering. On the other hand, if an adaptation function excessively prioritizes the long-term reward, video consumers may quit the streaming session early due to the unacceptable initial QoE. The cascading effect of bitrate selection makes it difficult to obtain a global optimal adaptation strategy, which involves the long-term system dynamics at each

decision stage. In general, given a streaming video with $|\mathcal{A}|$ encoding profiles and T segments, there are a total of $|\mathcal{A}|^T$ adaptation trajectories, each with a different expected total reward. In a typical streaming session ($|\mathcal{A}| \approx 10$ and $T \approx 200$, the search for global optimal solution is computationally intractable. To reduce the uncertainty in streaming environment, however, adaptation functions have to solve the complex optimization problem at each bitrate selection step.

- The optimality challenge is further perplexed by the strict requirement in computation time. In particular, a second delay in bitrate decision may lead to notable QoE losses, partially because the delay reduces the effective bandwidth in all streaming sessions (the relationship between the effective bandwidth and the computation time in bitrate selection can be understood by considering the delay as a part of RTT). In addition, the postponed decision suggests that ABR agents have to rely on outdated throughput estimation, buffer occupancy, and QoE status of video consumers, inevitably resulting in sub-optimal bitrate selection.

- Besides the two obstacles, adaptation functions also have to combat the diversity of environment states. To optimize the long-term reward, ABR agents should make efficient use of all available information to the player, including current buffer occupancy, future throughput dynamics, available chunk profiles, and bitrate selection history. As a result, the dimension of input space is in the order of $Z \times T$, where Z and T represent the dimension of chunk-level features and the number of chunks in the planning trajectory, respectively. Although each streaming video is encoded into a limited number of representations, the resulting bitrate is time-varying [93, 233]. Furthermore, the number and specification of encoding profiles often adapt to the characteristics of streaming videos, encoder types, and the capability of service providers [35, 95, 147, 208, 242]. A generic input space is then necessary to accommodate the diversity of streaming environment, suggesting a high dimensionality of chunk-level features Z. Given the enormous input space, it is extremely challenging to explore all possible states with a limited training data. In practice, adaptation schemes are often calibrated on a subset of observation states which are sparsely distributed in the input space. It remains questionable whether

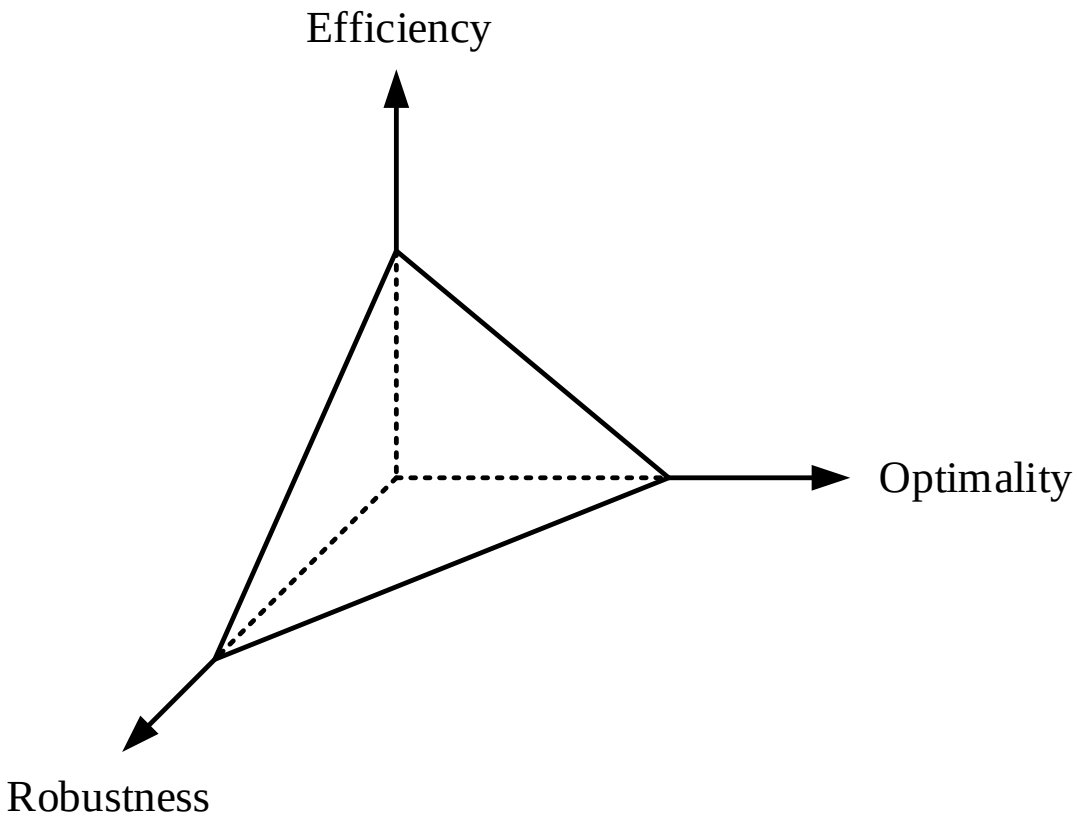


Figure 2.12: The design tradeoff for the bitrate adaptation function. In general, ABR algorithms cannot achieve efficiency in computation time, robustness to unobserved states, and optimality in expected total reward simultaneously.

> these algorithms can generalize well to unobserved environmental states (*e.g.*, novel encoding profiles).

A all-round bitrate selector should tackle the three challenges at once. However, it turns out that existing ABR logic struggles to optimize the three objectives simultaneously. Instead, they often compromise one or two dimensions to achieve a better performance in the other aspects. We formalize the phenomena the efficiency-robustness-optimality

tradeoff in Figure 2.12.

2.3.1 Bayesian View of Bitrate Selector

Given a reward function and a throughput predictor, there exists an optimal action-value q_t^*, which is the maximum expected return achievable by following any strategy, after seeing some sequence $\mathbf{s}$ and then taking some action a, $q_t^* = Q^*(\mathbf{s}_t = \mathbf{s}, a_t = a) = \max_\pi \mathbb{E}[U_t|\mathbf{s}_t = \mathbf{s}, a_t = a, \boldsymbol{\pi}]$. Here, $\boldsymbol{\pi}$ is a policy mapping sequences to actions (or distributions over actions). $U_t = \sum_{l=0}^{T-1} \gamma^l u_{t+l}$ denotes the expected cumulative reward starting at the time instance t, where u_t, $0 < \gamma \leq 1$, and T represent the instantaneous reward by receiving the t-th chunk, a constant discounting future rewards, and the number of chunks in the streaming session, respectively. Once the optimal action-value q_t^* for each pair of state $\mathbf{s}_t$ and action a_t is derived, we can select the action a that maximizes the action-value q_t^*.

In the context of ABR streaming, there exists an algorithm that produces the ground-truth optimal label of q_t^*. Given a state $\mathbf{s}_t$ consisting of the current playback buffer level, the future chunk profiles, and the throughput distribution (or a sample from the distribution), the optimal action-value function can be solved by value iteration (also known as dynamic programming) [201]. Specifically, the optimal action-value q_t^* obeys an important identity known as the Bellman equation. If q_{t+1}^* of the playback state $\mathbf{s}_{t+1}$ at the next time-step was known for all possible actions a_{t+1}, then the optimal strategy is to select the bitrate a_{t+1} maximizing the expected value $q_t^* = \mathbb{E}_{s_{t+1}\sim\mathcal{E}}[u_{t+1} + \gamma \max_{a_{t+1}} q_{t+1}^*]$. By using the Bellman equation as an iterative update, one can gradually improve the action-value function q_t starting from random initialization. Such value iteration algorithms eventually converge to the optimal action-value function q_t^* [201]. Although theoretically it is possible to integrate the value iteration algorithm into ABR players to produce online bitrate selections, the iterative computation is inconvenient, time-consuming, and expensive. Instead, it is common to use a function to approximate the optimal overall reward q_t^* for each state-action pair. Since ($\mathbf{s}_t$, a_t, q_t^*) are accessible, we can consider the formulation of the optimal bitrate selector as a supervised learning problem.

From a Bayesian perspective, one feasible approach to solve the bitrate selection problem is to estimate the optimal action-value function $p(q_t^*|\mathbf{s}_t, a_t)$. Note that the action-value

q_t^* expresses a probabilistic structure because some reward functions (*e.g.*, QoE function) are inherently stochastic. Given a set of training data $\mathcal{D}_\mathbf{q}$ comprising $N_\mathbf{q}$ state action pairs $(\mathbf{S}, \mathbf{a}) = \Big((\mathbf{s}_1, a_1), ..., (\mathbf{s}_{N_\mathbf{q}}, a_{N_\mathbf{q}})\Big)$ and their corresponding optimal overall rewards q_t^* $= (q_1^*, ..., q_{N_\mathbf{q}}^*)$, we aim to find a posterior action-value distribution $p(q_t|\mathbf{s}_t, a_t, \mathcal{D}_\mathbf{q})$ that best approximates $p(q_t^*|\mathbf{s}_t, a_t)$ for all t. The problem can be simplified by assuming an infinity long decision process such that the dependency on t can be dropped. Alternatively, we can enrich the state variable $\mathbf{s}$ with the time instance t. $p(q^*|\mathbf{s}, a, \mathcal{D}_\mathbf{q})$ can be considered as a point estimate of $p(q^*|\mathbf{s}_t, a_t)$ according to the Bayes' Rule $p(q^*|\mathbf{s}, a) = \int p(q^*|\mathbf{s}, a, \mathcal{D}_\mathbf{q})p(\mathcal{D}_\mathbf{q})\mathrm{d}\mathcal{D}_\mathbf{q}$. If the training data in $\mathcal{D}_\mathbf{q}$ are independent and identically distributed, one can parametrize the action-value function [37] as

$$p(q^*|\mathbf{s}, a, \mathcal{D}_\mathbf{q}) = \int p(q^*|\mathbf{s}, a, \boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D}_\mathbf{q})\mathrm{d}\boldsymbol{\theta}, \tag{2.22}$$

where $p(q^*|\mathbf{s}, a, \boldsymbol{\theta})$ and $p(\boldsymbol{\theta}|\mathcal{D}_\mathbf{q})$ represent the parametric action-value probability density function, and the posterior distribution over parameters, respectively. Given the enormous space of $\boldsymbol{\theta}$, the integration in Equation (2.22) is computationally intractable. As a result, a common practice is to approximate the predictive distribution $p(q^*|\mathbf{s}, a, \mathcal{D}_\mathbf{q})$ by a point estimate $p(q^*|\mathbf{s}, a, \boldsymbol{\theta}^*)$, where

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathcal{D}_\mathbf{q}) = \arg\max_{\boldsymbol{\theta}} p(\mathbf{q}|\mathbf{S}, \mathbf{a}, \boldsymbol{\theta})p(\boldsymbol{\theta}). \tag{2.23}$$

The specific form of parametric action-value function $p(\mathbf{q}^*|\mathbf{S}, \mathbf{a}, \boldsymbol{\theta})$ is generally not known. In practice, existing ABR algorithms usually minimize the mean squared error between the predicted action-value and the ground-truth q^*, which implicitly assumes a Gaussian parametric likelihood function

$$p(q^*|\mathbf{s}, a, \boldsymbol{\theta}) = \mathcal{N}(q^*|\mu(\mathbf{s}, a; \boldsymbol{\theta}), \beta), \tag{2.24}$$

where $\mu(\mathbf{s}, a; \boldsymbol{\theta})$ and β represent the mean and variance of the Gaussian distribution, respectively.

Direct estimation of $\boldsymbol{\theta}$ using the maximum likelihood method is problematic, because of the fundamental conflict between the enormous state-action space and the limited capacity to collect ground-truth q^* samples. Specifically, in the general case that the instantaneous

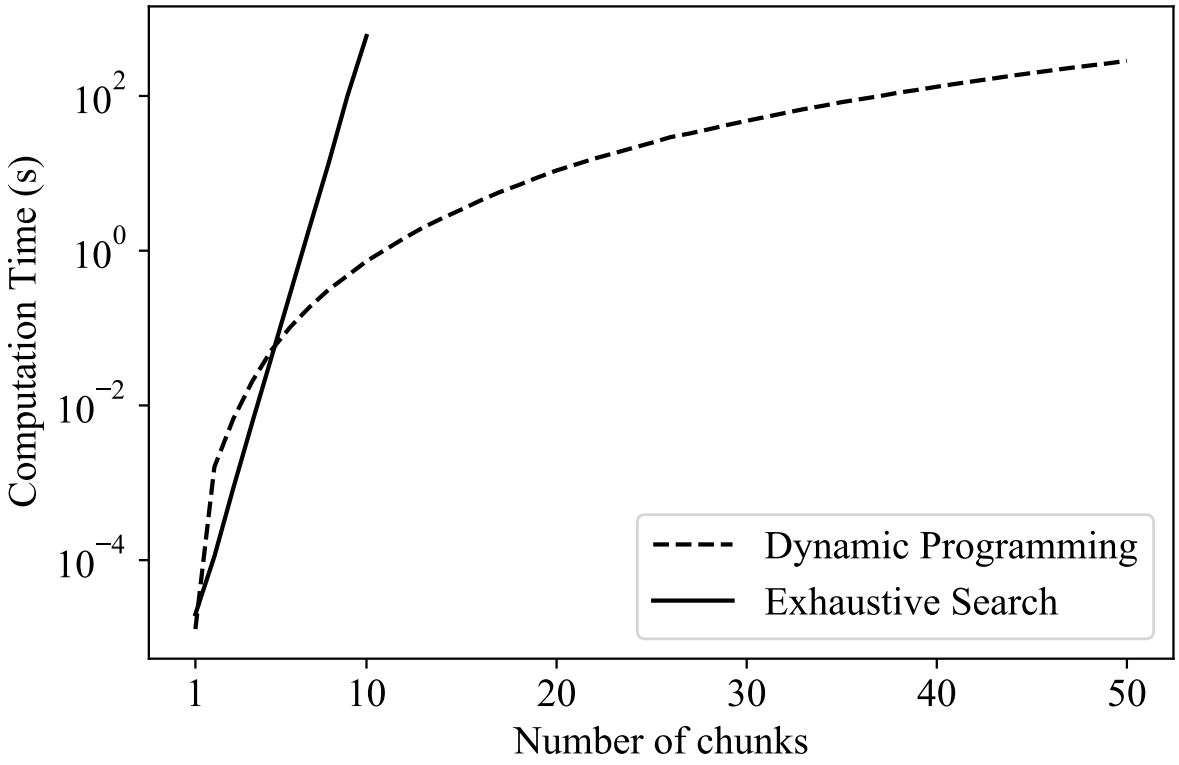


Figure 2.13: The computation time of dynamic programming and exhaustive search in a typical adaptive streaming scenario.

reward u_t is a function of all previous state and actions, the only feasible approach to determine q^* is by exhaustively searching all the combinations of bitrate selection. The computational complexity of exhaustive search is in the order of $O(|\mathcal{A}|^T)$, where $|\mathcal{A}|$ denotes the cardinality of the action space. Even if we simplify the problem by assuming that u_t enjoys a Markov property, obtaining each ($\mathbf{s}$, a, q^*) sample still involves solving a dynamic programming problem, whose time complexity is $O(T \times |\mathcal{S}| \times |\mathcal{A}|)$. $|\mathcal{S}|$ represents the cardinality of the state space. In the case of adaptive streaming, $\mathbf{s}$ may encode the statistics of future chunks (*e.g.*, chunk size and quality) and throughput prediction, suggesting that $|\mathcal{S}| \propto T$. Either way, the computation of q^* quickly becomes intractable as the planning window T becomes longer. Figure 2.13 shows the computational time of both approaches

with respect to T in a simplified ABR problem, where $|\mathcal{A}|) = 6$ and $|\mathcal{S}|) \approx 500,000$. We can observe that the computation of q^* on a sequence of $T = 45$ takes around five minutes, exceeding the time budget of a subjective experiment. A typical "large-scale" dataset allows for a maximum of a few thousands of action-value to be sampled. By contrast, the action-value function live in a high dimensional space, which is typically in the order of hundreds of thousands. Therefore, a few thousands of action-value samples are deemed to be sparsely distributed in the space. As a result, the maximum likelihood approach may hardly generalize to unobserved state-action pairs. To overcome the limitation, various of prior distributions $p(\boldsymbol{\theta})$ have been developed, which will be detailed in the subsequent section.

Once the predictive distribution $p(q^*|\mathbf{s}, a, \boldsymbol{\theta}^*)$ is determined, the optimal next chunk a^* can be obtained by selecting the bitrate with the maximum overall reward

$$\pi(a|\mathbf{s}) = \prod_{a' \neq a} p(Q(\mathbf{s}, a) \geq Q(\mathbf{s}, a')). \tag{2.25}$$

The probabilistic graphic model of the value-based approach is illustrated in Figure 2.14. One disadvantage of the value-based method is the computation overhead introduced by (2.25), especially in the case that $p(q^*|\mathbf{s}, a, \boldsymbol{\theta}^*)$ has a complicated structure.

Alternatively, one can directly model the optimal policy $\boldsymbol{\pi}(a|\mathbf{s})$ in (2.25) with a parametric approximation $\boldsymbol{\pi}(a|\mathbf{s}, \boldsymbol{\theta})$. Despite the distinction, these two approaches share a very similar training data generation procedure. Most of the fundamental concepts in the derivation of prior distribution $p(\boldsymbol{\theta})$ are also common to the two methods. The policy-driven approach can efficiently shift the cumbersome computation in the online decision process to the training procedure. However, these models usually require more training data to approximate the nonlinear mapping between value q and action a. For consistency, we adopt the value-based perspective to review the existing adaptation functions.

2.3.2 Existing Bitrate Selectors

Over the past decades, a wide variety of bitrate selectors have been developed where the major difference lies in the assumptions about the prior distribution $p(\boldsymbol{\theta})$. While all these

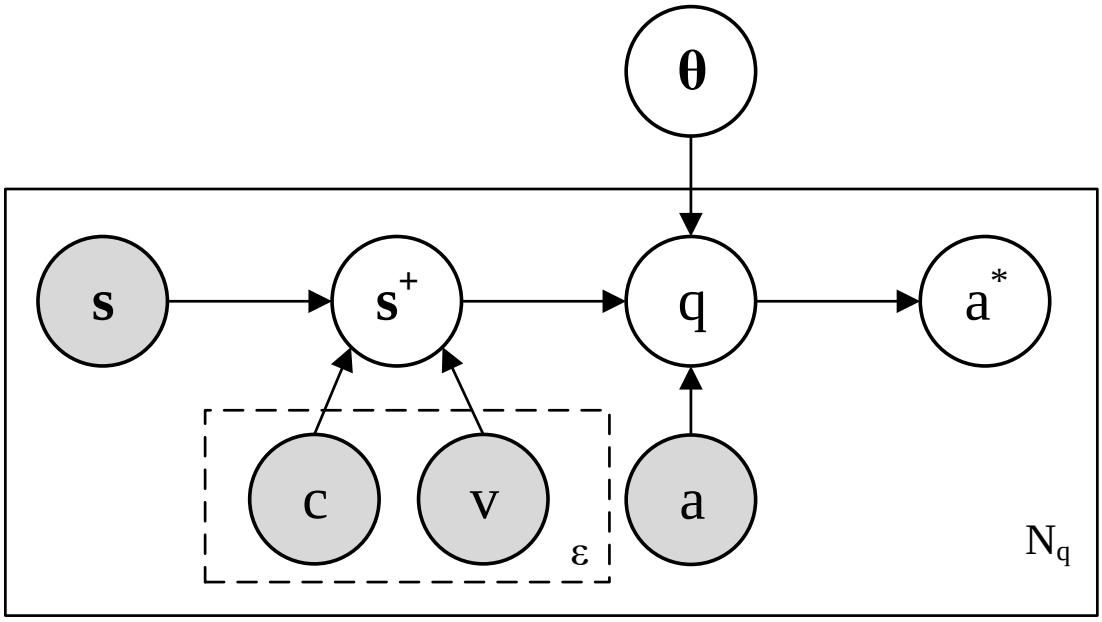


Figure 2.14: Graphical model representation of the value-based bitrate selection process. The box is "plate" representing replicates. Each node represents a random variable (or group of random variables), and the links express probabilistic relationships between these variables. The observable variables are shaded in color. $\mathbf{s}$ encodes the current state observation (such as the past throughput histories and current buffer occupancy). $\mathbf{s}^+$ represents the states required to make optimal offline decisions with the cumulative reward q, including the future throughput prediction $\mathbf{c}$ and the future chunk statistics $\mathbf{v}$. The optimal action a^* is obtained by selecting the action that optimizes the cumulative reward.

methods attempt to optimize the reward function, they often present a different tradeoff in efficiency, robustness, and optimality. Existing adaptation rules can be categorized based on when the actual logic is generated. The offline decision rules are developed before it is applied to a particular streaming video, whereas the online algorithms derive the adaptation function on the fly. In the subsequent discussion, we will present a Bayesian

interpretation of both approaches. We will also analyze the position of these algorithms in the efficiency-robustness-and optimality space.

Offline Bitrate Selectors

The offline adaptation rules approximate a parametric function $p(q^*|\mathbf{s}, a, \boldsymbol{\theta})$ before interacting with a streaming session. The fixed adaptation function can either be derived from expert knowledge, or learnt by sophisticated machine learning methods. Here we introduce existing offline bitrate selectors in detail.

- Heuristic Methods: The earliest bitrate selectors employ hand-crafted rules to determine which chunk to request, although they all assume higher bitrate leads to higher quality. One of the most representative methods is the Additve Increase Multiplicative Decrease (AIMD) strategy [122]. Inspired by the congestion control in TCP, the algorithm switches up to the next higher representation level when the estimated download time is less than a pre-defined threshold. On the other hand, if the expected download time exceeds a upper threshold, AIMD decreases the target bitrate by half. When the two thresholds for switching are not met, the algorithm keeps the selected bitrate. Since the algorithm only considers three bitrate levels as feasible actions, the resulting reward is sub-optimal in terms of the instantaneous reward u_t.

  To reduce the quality variation, some Additve Increase Additive Decrease (AIAD) schemes [2, 134] are proposed, which restrict the adaptation space to the neighbouring bitrate level of the current representation. Similar to its close variant AIMD, the AIAD strategies quantize the state space based on some pre-defined thresholds. Then, a bitrate selection in the filtered action space is taken in each state region according to the educated guess of the instantaneous reward function. One of the major improvement of the instantiation in [134] is the expanded state space. By incorporating both the buffer occupancy information and the throughput prediction, the algorithm can theoretically attain a higher reward.

  Another heuristic adaptation function, called FESTIVE [102], can also be considered as a variant of AIMD. FESTIVE not only limit the magnitude of bitrate increasing,

but also the rate of adaptation. In particular, the strategy only switches to the next higher level and uses a lower rate of upward switches at higher bitrates. In the case of bandwidth drop, FESTIVE does not restrict the target adaptation level to its immediate lower representation. After the AIMD-based pre-processing, FESTIVE reduces the cardinality of action space by to two at each decision stage. The major difference between FESTIVE and AIMD is that FESTIVE explicitly optimizes the instantaneous reward function on a subset of available bitrate levels. Other heuristic adaptation functions [103, 141] share a similar quantize-filter-maximize strategy, which differ in their choice of quantization level and filtering schemes.

These heuristic approaches correspond to a family of prior action-value distributions. First, the filtered action space suggests that $p(q^* = 0|\mathbf{s}, a = a', \boldsymbol{\theta}) = 1$ for all $a' \notin \mathcal{A}_f$, where $\mathcal{A}_f$ represents the feasible set of bitrate selections. For example, $\mathcal{A}_f = \{a_{t-1}^-, a_{t-1}, a_{t-1}^+\}$ for the AIMD method, where a_{t-1}, a_{t-1}^-, and a_{t-1}^+ are the previous representation level, its immediate lower and higher encoding profiles, respectively. Second, the hand-crafted bitrate selection rule are dedicated to optimize the reward function, suggesting that $p(q^* = \tilde{u}_t|\mathbf{s}, a = a', \boldsymbol{\theta}) = 1$ for all $a' \in \mathcal{A}_f$. In most heuristic methods, $\tilde{u}_t$ can be regarded as an educated guess of the instantaneous reward, whereas the approaches involving an explicit optimization make use of the precise instantaneous reward function $\tilde{u}_t = u_t$. Since the functional form of these policies is deterministic, we have $p(\boldsymbol{\theta} = \boldsymbol{\theta}_{\text{heuristic}}) = 1$ and $p(\boldsymbol{\theta} = \boldsymbol{\theta}') = 0$ for any function $\boldsymbol{\theta}' \neq \boldsymbol{\theta}_{\text{heuristic}}$. There have also been efforts to improve the biased subjective prior by integrating the information in the likelihood function $p(\mathbf{q}^*|\mathbf{S}, \mathbf{a}, \boldsymbol{\theta})$ [3]. It has been shown that the Bayesian treatment significantly outperforms its heuristic counterpart.

Thanks to the reduced action space, these heuristic approaches are computationally efficient. Specifically, the complexity of each bitrate selection is $O(|\mathcal{A}_f|)$. However, the heuristic approaches are by no means close to optimal. Furthermore, these hand-crafted adaptation rules, which are usually derived from observations in a limited streaming session and environmental conditions, are often susceptible the unobserved states.

- Imitation Learning: Another drawback of the empirical priors is that the prior policy distribution makes strong assumption about the reward function and throughput dynamics, which by themselves are subjects of ongoing research. If a novel reward function or a better throughput predictor become available, the expert knowledge of $p(q^*|\mathbf{s}, a, \boldsymbol{\theta})$ can be difficult to obtain. Motivated by the limitations, state-of-the-art adaptation functions determine the prior action-value distribution via demonstrations from an optimal policy.

  Fast Model Predictive Control (FastMPC) is one of the first instantiations of the imitation learning framework [241]. The algorithm learns a lookup table from noisy samples from the optimal action-value function $p(q^*|\mathbf{s}, a)$, where each entry stores the expected cumulative reward $\mathbb{E}[q^*]$ for a particular state-action pair. At each state $\mathbf{s}'$, FastMPC sweeps through all entries with $\mathbf{s} = \mathbf{s}'$ in the lookup table, and selects the option with the maximum total reward. In general, the state variable lies in a continuous space, whose dimensionality is well above 100. Furthermore, the action-value q^* is also a function of the available actions a, which may be content-adaptive. To simplify the problem, the authors make a series of assumptions. First, FastMPC reduces the state space to three dimensions consisting of the current buffer occupancy, the average future throughput, and the current bitrate level. The pre-processing procedure can be regarded as a mapping from the raw observation space to a latent variable space. Second, each state variable is made discrete by quantizing the state space with a fixed step size. Third, the training process is performed on a pre-defined bitrate ladder. Unfortunately, the tabular method do not scale to large state-action space [131].

  With the recent exciting development of deep learning methodologies, a end-to-end bitrate adaptation solution becomes possible. Comyco [92] takes advantage of powerful neural networks to model the optimal q^*. The training of neural network and lookup table differ in several ways. First, in opposition to the tabular approach which estimates the long-term reward separately for each state without any generalization, Comyco uses a generic function approximator to estimate the action-value function. In essence, each $(\mathbf{s}, a, q^*)$ sample would perturb the action-value function for all states and actions. Second, the neural network model takes raw state observations as

input, significantly expanding the state space. Thanks to the huge capacity of neural network, the model can digest massive data in the training process to combat the curse of dimensionality. The model is sent to the client player to perform real-time bitrate adaptation decisions once it is fully calibrated.

Since these adaptation rules do not integrate information from novel observations, one can regard them as prior action-value distributions $p(q^*|\mathbf{s}, a, \boldsymbol{\theta})$. In contrast to heuristic methods, for which the prior distribution is fixed before any data are observed, these algorithms choose a prior distribution that best explains a dataset $\mathcal{D}_{\mathbf{q}}$. Specifically, the marginal likelihood of the observed data is given by $p(\mathcal{D}_{\mathbf{q}}|\boldsymbol{\theta})$. Maximizing the quantity as a function of $\boldsymbol{\theta}$ gives a point estimate of $\boldsymbol{\theta}$, an instance of a method known as empirical Bayes [19]. One of the biggest advantage of the empirical Bayes' method over the subjective prior is its flexibility to adapt to new environment and reward signals. In particular, empirical Bayes' method can automatically determine appropriate prior distributions at any tasks by repeating the training procedure on task-specific data.

Imitation learning-based bitrate selectors can be executed extremely efficiently because the optimal decision is given by either querying a lookup table, or performing a simple feed-forward operation. The computation complexity of these operations is merely $O(|\mathcal{A}|)$, which can be further reduced by parallel computation. In practice, these computations take only a few milliseconds even on lightweight computation devices such as smartphones. Furthermore, an increasing number of devices have equipped with dedicated hardware to accelerate standard neural network computations [96], suggesting that more powerful statistical models may be deployed in the future. Thanks to the pre-training, imitation learning methods demonstrate state-of-the-art performance on states/environment that have been experienced in the training set. Nevertheless, given the fundamental conflict between the enormous state-action space and the limited capacity to collect ground-truth q^* samples, FastMPC and Comyco may not generalize well on unobserved states/environment.

- Reinforcement Learning: In fact, imitation learning-based methods may even fail on training sequences due to the distribution mismatch problem [176]. Specifically, the

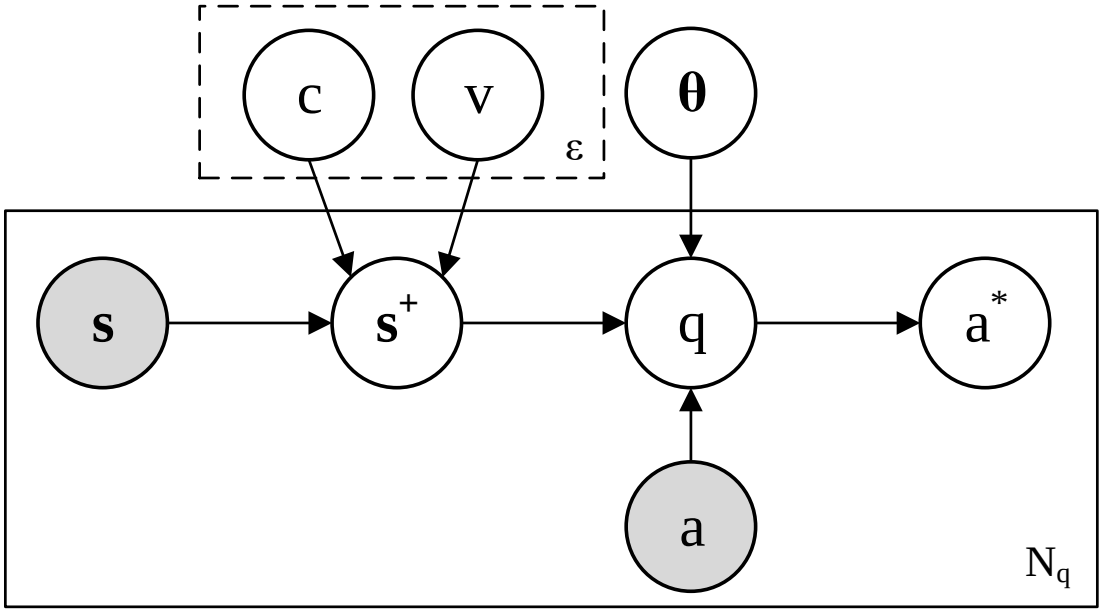


Figure 2.15: Graphical model representation of the policy-based bitrate selection process. The box is "plate" representing replicates. Each node represents a random variable (or group of random variables), and the links express probabilistic relationships between these variables. The observable variables are shaded in color. $\mathbf{s}$ encodes the current state observation (such as the past throughput histories and current buffer occupancy). $\mathbf{s}^+$ represents the states required to make optimal offline decisions with the cumulative reward q, including the future throughput prediction $\mathbf{c}$ and the future chunk statistics $\mathbf{v}$. The optimal action a^* is obtained by selecting the action that optimizes the cumulative reward. The model-free reinforcement learning models do not explicitly predicts the future environment state $\mathbf{s}^+$ in the estimation of q.

learnt agent may make some tiny errors at each state, such that the updated state may gradually deviate from the state distribution in the training set. Recall that these supervise learning approaches achieve less competitive performance on unobserved states/environment. The distribution mismatch problem has been thoroughly

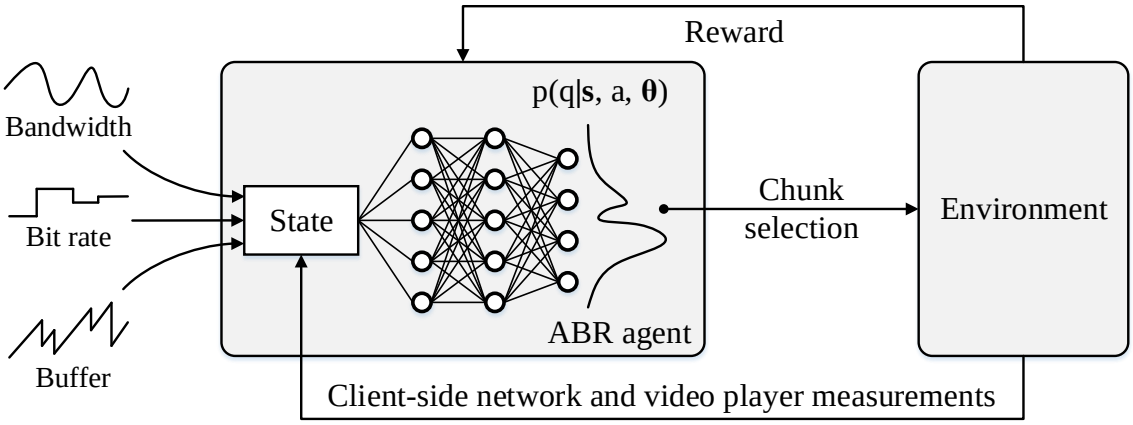


Figure 2.16: A reinforcement learning framework for bitrate adaptation.

investigated in other sequential decision making problems, such as autonomous driving [179] and computer games [138, 187]. A common solution is reinforcement learning [201]. As opposed to the imitation learning that learns the policy from expert demonstration, reinforcement learning learns to improve an agent's decision making ability during the interaction with the environment. Figure 2.16 summarizes how reinforcement learning can be applied to bitrate adaptation. The basic idea behind many reinforcement learning algorithms is to estimate the action-value function, by using the Bellman equation as an iterative update. Specifically, a parametric action-value function approximator can be trained by minimizing a sequence of loss functions $L_i(\boldsymbol{\theta}_i)$ that changes at each iteration i

$$L_i(\boldsymbol{\theta}_i) = \mathbb{E}_{\mathbf{s},a\sim\rho(\cdot)}\Big[(q_i - Q(\mathbf{s}, a; \boldsymbol{\theta}))^2\Big], \tag{2.26}$$

where $q_i = \mathbb{E}_{\mathbf{s}'\sim\mathcal{E}}[u + \gamma \max_{a'} Q(\mathbf{s}', a'; \boldsymbol{\theta}_{i-1})|\mathbf{s}, a]$ is the target for iteration i and $\rho(\mathbf{s}, a)$ is a probability distribution over states $\mathbf{s}$ and actions a that are commonly referred to as the behaviour distribution [138]. The parameters from the previous iteration $\boldsymbol{\theta}_{i-1}$ are held fixed when optimising the loss function $L_i(\boldsymbol{\theta}_i)$. Note that the targets depend on the network weights; this is in contrast with the targets used for imitation

learning, which are fixed before learning begins [138]. Under some mild conditions, it can be proven that the parametric action-value function approximator converges to the optimal solution [201]. Despite a slower convergence rate, the learning agent experiences many states falling out of the distribution $\rho^*(\mathbf{s}, a)$ in the training stage, where $\rho^*(\mathbf{s}, a)$ represents the behaviour distribution of the optimal policy. As a result, reinforcement learning approaches are generally more robust in unobserved states/environment.

Pioneering work on reinforcement learning-based bitrate selector dates back to 2013, when Claeys *et al.* [32] investigated the feasibility of basic Q-learning [201] in the context of adaptive streaming. The method uses Bellman equation as the updating rule to learn a tabular action-value function. Various follow-up works extend the study by expanding the state space [133], refining the definition of QoE [212], and improving the efficiency of the naïve Q-learning [33]. These tabular reinforcement learning methods share similar limitations with their imitation learning counterpart. Recently, many studies proposed to apply deep reinforcement learning in adaptive video streaming [91, 131], with Pensieve [131] being the most representative method. Pensieve trains a neural network model to perform bitrate selection using a variant of deep Q-learning [137, 138], while the basic idea still comes from (2.26). Thanks to the capacity of neural network, Pensieve and its variants can take raw environment observations as input without making unrealistic assumptions for dimensionality reduction. It has been empirically demonstrated that these deep learning models outperform the traditional reinforcement learning by a sizable margin [131].

From a Bayesian view, the reinforcement learning approach also produces a prior action-value distribution $p(q^*|\mathbf{s}, a, \boldsymbol{\theta})$. Conceptually, the agent-environment interaction process can be regarded as a data augmentation method to the imitation learning such that the agent not only observes state-action pairs from the optimal behaviour distribution $\rho^*(\mathbf{s}, a)$, but also experiences out of distribution samples. Some reinforcement learning methods even explicitly encourage exploration of the state space with a deliberately designed loss function [131]. Applying the updating rule in (2.26) on the augmented dataset $\mathcal{D}_{\mathbf{q}^+}$ corresponds to maximizing the likelihood function $p(\mathcal{D}_{\mathbf{q}^+}|\boldsymbol{\theta})$ under the Gaussian assumption, which is also an instance of the empirical

Bayes method [19].

The reinforcement learning agent only takes a feed-forward operation and optionally a maximization step to produce a bitrate selection, whose computation complexity is $O(|\mathcal{A}|)$. Furthermore, the learnt action-value function can theoretically converges to the ground-truth $p(q^*|\mathbf{s}, a)$, suggesting that the approach is also effective in optimizing the reward function. The augmented data generated from the agent-environment interaction process acts as a regularizer and helps reduce overfitting when training a statistical model. As a result, reinforcement learning usually exhibits a relatively stronger generalization capability than imitation learning on unfamiliar environment/states. Nevertheless, reinforcement learning-based adaptation functions may still fail to generalize, especially when the available actions in the test set deviate significantly from the training set.

To sum up, offline adaptation functions can be interpreted as certain prior distributions of the action-value function $p(q^*|\mathbf{s}, a, \boldsymbol{\theta})$. These fixed adaptation rules can achieve very high reward within a very short decision period. However, the prior distributions encode strong assumptions about the streaming environment (including the characteristics of streaming videos, the network variability, and the reward function), making the offline algorithms difficult to generalize in practice.

Online Bitrate Selectors

In contrast to the offline counterpart, online bitrate selectors adaptively generate the action-value distribution $p(q^*|\mathbf{s}, a, \boldsymbol{\theta})$ based on the observation of state action pairs $(\mathbf{s}, a)$ in a streaming session. Without large amount of training data, state-of-the-art online decision making strategies can achieve equally competitive performance. We summarize the recent progress of the techniques as follows.

- Greedy Algorithm: The heuristic bitrate selectors compromise reward to obtain a high computational efficiency. In practice, however, the number of commonly used representations is only in the order of hundreds, most of which may not appear simultaneously in a particular bitrate ladder. The reduction in computation time

from $O(|\mathcal{A}|)$ by taking all actions into considerations to $O(|\mathcal{A}_f|)$ is negligible. It is, therefore, reasonable to promote the optimality over efficiency. Following this direction, a majority of ABR algorithms utilize the greedy algorithm in hopes of obtaining a higher reward. The greedy algorithm sweeps through all actions, and select the one with the highest instantaneous reward u_t. These adaption logic can be generally categorized into three classes according to their state space. Algorithms in the first category selects the next chunk only based on throughput prediction, which are commonly referred to as rate-based strategies [63, 119]. For example, if the goal was to maximize the bitrate usage, rate-based greedy algorithm would choose the maximum possible bitrate below the predicted throughput. The second class advocate bitrate adaptation solely based on buffer occupancy while making strong assumptions about the network characteristics. As a result, they are often dubbed buffer-based strategies [93, 191]. Interestingly, these algorithms often employ buffer occupancy-related reward functions. For example, a state-of-the-art buffer-based strategy named Buffer-Occupancy-based Lyapunov Algorithm (BOLA) are dedicated to minimize the unplayed video in the playback buffer. Unfortunately, both classes of algorithms are discarding possibly useful information, resulting in sub-optimal performance. The third group makes use of both buffer occupancy and throughput predictions in the decision making [16, 35]. For example, if there are abundant video segments in the playback buffer, one can afford to select a chunk with bitrate higher than the available throughput.

From a Bayesian perspective, applying greedy algorithm is equivalent to approximating the posterior distribution with a noisy sample and a non-informative prior. Specifically, the look-ahead and reward evaluation procedure at a particular state $\mathbf{s}$ can be regarded as sampling from the optimal action-value function $p(q^*|\mathbf{s}, a)$ for each a. Since the value estimation only involves the instantaneous reward, we can consider the obtained reward as a noisy sample of $p(q^*|\mathbf{s}, a)$ such that $q^* \approx u_t$. Alternatively, one can interpret the instantaneous reward u_t as a lower bound of the action-value $q^* = \sum_{l=0}^{T} u_{t+l}$. Non-informative prior is imposed because the method does not make a priori assumption about the form of the action value function $p(q^*|\mathbf{s}, a, \boldsymbol{\theta})$.

Since the greedy bitrate selector traverses through the action space at each decision stage, the computation complexity of the algorithm is $O(|\mathcal{A}|)$. With only a slight increment in computation time, the algorithm can significantly boost the performance of heuristic approaches [46]. Furthermore, the algorithm is very robust to unobserved environment states, thanks to the non-informative prior.

- Model Predictive Control (MPC): In general, the greedy algorithm fails to produce the optimal solution, and may even produce the worst possible solution. For example, let us consider a widely used reward function [3, 131, 200, 241]

$$u_t = r_t - \tau_t - |r_t - r_{t-1}|, \tag{2.27}$$

where r_t and τ_t represent the bitrate and the rebuffering duration during the download of the t-th chunk. The motivation behind the reward function is that subjective QoE is degraded by both rebuffering and quality adaptation. The greedy algorithm will constantly pick the initial bitrate r_0, which is usually the lowest representation level, as the bitrate-related $r_t - |r_t - r_{t-1}|$ term is identical for all representations. By choosing the lowest representation, the algorithm reduces the likelihood of rebuffering τ_t, thereby optimizing the instantaneous reward. Clearly, the local optimal strategy converges to a very poor solution.

Realizing limitations of the traditional approach, Yin *et al.* proposed to solve the optimization problem by MPC [241]. MPC generalizes the greedy algorithm by extending the planning horizon to $1 \leq K \leq T$. At the t-th bitrate decision, the algorithm estimates the expected total reward over the interval $[t, t + K]$ for each action a, and select the action that optimizes the value. At the next iteration, the algorithm takes the updated state information as the input, re-plans the bitrate trajectory, and produces the optimal bitrate selection. The control-theoretic ABR framework has a very similar Bayesian interpretation to the greedy algorithm, where the only difference is that the noisy sample of $p(q^*|\mathbf{s}, a)$ now takes the form $q \approx \sum_{l=0}^{K-1} \gamma^l u_{t+l}$.

The MPC framework provides a flexible control knob to adjust the tradeoff between efficiency and optimality. When $K = 1$, MPC degrades gracefully to the greedy algorithm, prioritizing efficiency over optimality. As K approaches T, the algorithm

gradually converges to the global optimal solution at the cost of extremely high computational complexity. Recall that the reward optimization problem over an interval K can be solved by exhaustive search or dynamic programming, whose time complexity are $O(|\mathcal{A}^K|)$ and $O(K \times |\mathcal{S}| \times |\mathcal{A}|)$, respectively. To obtain a reasonable approximation of action-value q within a limited time budget, the look-ahead horizon K typically ranges from 5 to 8 [3, 200, 237, 241]. MPC also inherits the robustness from the greedy algorithm, since they do not make any assumption about the throughput dynamics and reward function. In particular, the throughput prediction can be embedded into the state variable without re-calibrating the control policy. Reward functions can also be applied in the framework in a plug-and-play fashion.

In summary, online adaptation logic generally follows a sample-estimate-optimize procedure. The methods approximate the posterior action-value function $p(q^*|\mathbf{s}, a, \boldsymbol{\theta})$ using the maximum likelihood estimator solely based on a noisy sample from the theoretically optimal policy. By adjusting the number of planning steps K, online ABR programs may compromise computation time in order to reduce the noise level. Most importantly, these bitrate selectors are very robust to the perturbation of streaming environment. Figure 2.17 summarizes the performance of existing adaptation functions in terms of efficiency, robustness, and optimality.

2.4 Overview of Adaptive Bitrate Algorithms

Over the past decade, significant efforts have been denoted in the development of ABR algorithms [119, 93, 141, 200, 191, 102, 241, 131, 30, 32, 212, 3, 91, 92, 237], all of which can be explained by the presented framework. A complete list of summary is given in Table 2.2. Here we only provide a brief description to the most representative approaches.

Rate-Based (RB) algorithm [63] is the default ABR controller in the DASH standard. The name rate-based comes from the state space of the bitrate selector, which does not depend on the buffer occupancy. RB algorithm is composed of an arithmetic mean throughput predictor, a bitrate-centric reward function, and a greedy adaptation logic. Specifically,

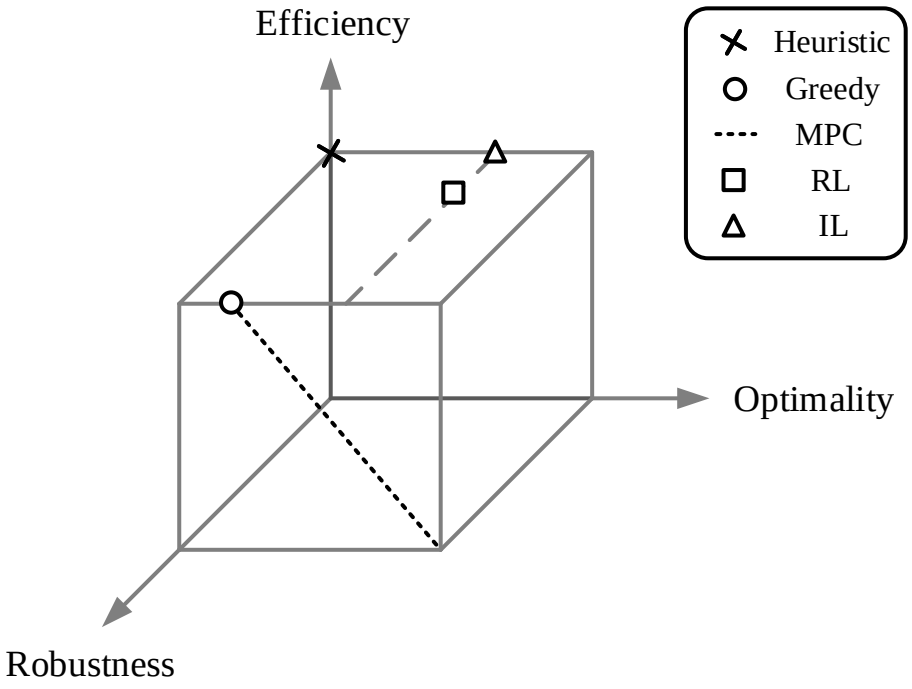


Figure 2.17: The schematic diagram of the efficiency-robustness-optimality tradeoff of the existing bitrate selectors.

the function picks the maximum available bitrate below the throughput prediction over the past five chunks. Assuming the throughput prediction is accurate, one can show that the algorithm are dedicated to maximize the following reward function

$$U = \sum_{t=1}^{T} r_t - \alpha \tau_t, \tag{2.28}$$

where $\alpha \to \infty$. Despite its simplicity, some recent subjective experiments demonstrated that rate-based is on par with the state-of-the-art ABR algorithms [53, 46].

Thanks to its simplicity, Buffer-Based (BB) algorithm [93] has become another baseline ABR method in all subsequent studies. In contrast to the rate-based algorithm, BB

advocates decision making based only on buffer occupancy, while regarding throughput variations as un-modeled disturbances. Specifically, BB employs a monotonically increasing function $f(\cdot)$ between the buffer occupancy and the bitrate selection. Although the derivation of the adaptation rule seems somewhat arbitrary, BB is later well investigated by another buffer-oriented method [191]. It has been shown that BB implicitly utilizes a first-order stationary throughput distribution whose mean is encoded by the model parameters, a buffer occupancy weighted QoE function, and a greedy adaptation logic. In particular, the reward function of BB can be expressed as

$$U = \frac{\sum_{t=1}^{T} g(r_t) - \alpha\tau_t - \beta b_t}{T_{\text{end}}}, \tag{2.29}$$

where $g(\cdot)$, b_t, T_{end}, α, and β denote a non-linear function, the buffer occupancy of time t, the overall duration of the streaming session, and two weighting parameters, respectively. The function $g(\cdot)$ uniquely determines the functional form of the adaptation logic $f(\cdot)$.

MPC makes the first attempt to explicitly define the three functional components in our framework, which unifies RB methods and BB algorithms in a principled way. By default, MPC uses the harmonic mean throughput predictor and applies dynamic programming to optimize the following reward function

$$U = \sum_{t=1}^{T} r_t - \sum_{t=1}^{T} \tau_t - \sum_{t=2}^{T} |r_t - r_{t-1}|, \tag{2.30}$$

where the third term penalizes frequent quality adaptations. Nevertheless, the MPC framework is general enough to incorporate a wide range of throughput predictors and reward functions [237].

Recently, reinforcement learning-based ABR algorithms have gained an increasing popularity, with Pensieve being the most representative algorithm [131]. Following these methods, a neural network agent gradually learn to optimize the expected future reward via their interactions with simulated streaming environments. Inherited from the model-free reinforcement learning framework, Pensieve does not need to explicitly model the throughput dynamics. Instead, the learnt policy merges the throughput predictor and bitrate selector into an unified model. The probabilistic graphic model of model-free reinforcement

learning methods and their variants is shown in Figure 2.15. Nevertheless, it is possible to distill the underlying data-driven throughput model from Pensieve [132, 201]. In [131], the authors also showed that Pensieve can adapt to different instantiations of reward functions.

More recently, a few studies also formulate the adaptive bitrate selection as an supervised learning problem [92, 241]. For example, Comyco [92] trains a statistical model to imitate actions produced by an offline optimal agent. Akin to Pensieve, Comyco also models the throughput predictor (implicitly) and the bitrate selector using a single neural network, which optimizes a close variant of reward function in 2.30.

2.5 Validation of Adaptive Bitrate Algorithms

With many ABR algorithms at hand, it becomes pivotal to compare their performance, so as to find directions for further advancement. Mathematically, the performance validation corresponds to the evaluation of (1.1) for each ABR algorithm, which is a complex problem in its own right. To be specific, the difficulty in the computation of the expected reward function arises from two aspects. First, the measurement of (1.1) requires the precise knowledge about the streaming environment. Quantitatively, this means specification of probability distributions over both network conditions and streaming videos, neither of which are available. In practice, many authors base their studies on empirical results computed from a set of example network traces and streaming videos that are representative of the relevant environment. Generally speaking, the quantity and representativeness of these samples are the keys to the success of the sampling-based evaluation procedure. Second, since the HVS is the ultimate receiver of streaming videos, the only "correct" way to evaluate the reward function and the corresponding ABR system is by performing a subjective experiment. Unfortunately, subjective testing is inconvenient, time-consuming, and expensive. As a result, some compromises to the quantity and representativeness of data have to be made in a practical subjective evaluation. Alternatively, one can take objective QoE models as an approximation of the subjective QoE in the ABR algorithm evaluation, which inevitably restricts the reliability of the experiment. In general, all the ABR evaluation approaches aim to approximate (1.1), while they put different emphasis

on the tradeoff among quantity, representativeness, and reliability of data. We summarize the existing evaluation methodologies as follows:

- Offline Simulation + Objective Evaluation: To obtain bitrate selection trajectories, this approach simulate the streaming process with a network emulator, which can either sample a bandwidth data from a certain distribution or faithfully shape the bandwidth according to a target throughput trace. Once the bitrate decisions are generated for each ABR algorithm, the algorithm developers usually compute certain statistics of streaming events as the performance measure. Given the limited realistic throughput traces and streaming videos, early researches relied on small-scale synthetic bandwidth data with only one source content as a representation of the practical streaming environment [2, 35, 102, 103, 119, 144, 239]. Equally simplistic component was the evaluation criteria, where bitrate utilization, rebuffering duration, and bitrate variation were commonly used to assess individual aspects of ABR algorithms. However, due to the tradeoff between video bitrate and rebuffering duration, it is theoretically impossible for an ABR algorithm to achieve the best performance in all aspects. In fact, researchers often found it difficult to conclude which ABR algorithm is better based on these separate performance measures. With the aim to evaluate ABR algorithms in a more realistic streaming environment, Yin *et al.* [241] presented one of the largest objective evaluations at that time, covering a total of 2,000 real-world throughput traces. The study also explicitly defined the QoE as both the optimization goal and the evaluation criterion. Following their seminal work, the evaluation criterion of ABR algorithms have gradually converged to a single measure of QoE [3, 16, 92, 131, 191, 202], although each study may adopt different definition of QoE. One of the biggest advantages of this approach is its high efficiency, as it is not necessary to perform the authentic streaming process. State-of-the-art chunk-level simulator allows an ABR algorithm to "experience" 100 hours in only a few minutes, making the setup an preferable choice for a very large scale validation experiment. Nevertheless, this evaluation methodology maximizes the quantity of data at the cost of the representativeness of input data and the reliability of reward function. Thanks to its simplicity and efficiency, the combination of objective evaluation and offline streaming environment simulation remains the most prevalent evaluation method for

ABR algorithms since the ratification of DASH.

- Online Deployment + Objective Evaluation: The offline simulation may suffer from the concept drift problem [64], where the characteristics of streaming video and network condition change over time. For example, it is dangerous to assume a hypothesis that works well on 3G network would generalize to 4G without testing it in the realistic 4G environment. The problem has also been known as the "staleness problem" in the network community for decades [65], where mature solutions exist. The most straightforward solution is to deploy the system under test in the real environment, such that the system can experience up-to-date input data. Following this line of thought, several efforts advocated the evaluation of ABR algorithms in a real-world streaming environment [200, 237]. These studies typically deploy a couple of competing ABR algorithms in practical video delivery platforms such as YouTube, Netflix, and Hulu for a short period. In each streaming session, some quality-related features are recorded in the backend, based on which objective evaluation is conducted. The online deployment of ABR algorithms enjoys the most realistic streaming environment for performance evaluation. Yet numerous obstacles remain in the path of extensive application of this approach. First, most adaptive streaming platforms are proprietary, suggesting that this method is not applicable for an average ABR developer. Even if video service providers are willing to share their infrastructures to ABR researchers, the resulting data are not allowed to be released due to license and privacy issues, making the evaluation study difficult to reproduce. Second, the pilot studies can only be hold within a short time period to avoid the potential loss of QoE, reputation of the service, and eventually the revenue of the service provider. Consequently, the quantity of data is bounded by the capacity of the experiment. Third, the ABR algorithms are not evaluated under identical throughput traces and streaming videos due to the uncontrolled experimental protocol, suggesting that the reliability of such experiment is also compromised.

- Offline Simulation + Subjective Evaluation: Despite the improvement of objective QoE models over the past decade, subjective evaluation still remains the most reliable way to evaluate the QoE of a streaming video. A typical controlled subjective evalu-

ation study takes the following steps. At the beginning, a handful of source contents, encoders, network traces, viewing devices, and ABR algorithms are selected to cover the diversity of streaming environment. A set of offline simulations are conducted, during which the relevant streaming activities such as bitrate decision and rebuffering duration are recorded. Based on the streaming logs, researchers either reconstruct each streaming session using video processing tools and store the resulting videos to the hard drive, or customize DASH players to enable the reproduction of a pre-defined bitrate decision trajectory. A training session is then performed to get participants familiar with the experimental procedure and calibrate their quality scales prior to the main subjective experiment. During the subjective experiment, participants are instructed to provide a QoE score after the playback of each streaming video. The ground truth QoE label of each test stimuli, often referred to as MOS, is obtained by applying a series of post-processing techniques on the raw quality ratings to reduce the potential sampling noise. Thanks to the controlled experiment and noise reduction technique, this approach is generally considered as the most accurate way to measure the performance of ABR algorithms. Nevertheless, only have a limited number of studies taken this method [10, 53], partly because its low cost efficiency. More importantly, the scale of subjective experiments is restricted by the fatigue effect, which suggests that the reliability of subjective ratings gradually degrades with respect to the number of test samples. Given the limited capacity of subjective testing against the large variety of streaming environment, it is virtually impossible to obtain sufficient subject-rated streaming videos.

- Online Deployment + Subjective Evaluation: At first glance, the combination of online deployment and subjective evaluation may seem ideal for the evaluation of ABR algorithms, in which the quantity, representativeness, and reliability of data can be optimized simultaneously. Somewhat surprisingly, so far none of the evaluation schemes have employed the traditional subjective evaluation in the wild. The primary obstacle to such a strategy is the lack of motivation for the experimental participation. In contrast to the laboratory experiments that usually attract participants by financial compensation, a random viewer on the Internet do not have strong incentives to provide a quality rating when consuming online videos. Even without

the stumbling block, there is still a lack of proven outlier removal scheme for such data analysis. To overcome these problems, a few studies proposed to replace the cumbersome subjective QoE ratings by other subject-related quantities. A common choice of such surrogate QoE is the user engagement [93, 237], where it is assumed that the early exit of a streaming session is a direct consequence of quality degradation. Since the viewing duration is automatically recorded by the ABR player once a viewer closes a streaming session, it can be obtained with very little effort. However, this approach neglects the impact of viewers' tolerance, random exit, and loss of interests. Other proposals aim to infer the QoE from user-viewing activities such as pause, refresh, and play with full screen [140]. These methods suffer from similar limitations to the user engagement approach. Nevertheless, once the solution to these problems is found, the subjective evaluation method of ABR algorithms in the realistic environment has the potential to change the landscape of the fields of ABR and QoE.

Each of the aforementioned methods exhibits its own benefits and drawbacks, and they complement each other in terms of quantity, representativeness, and reliability. As a result, it is ideal to employ multiple schemes in the evaluation of ABR algorithms.

A summary of existing ABR evaluation studies is given in Table 2.3. Most of the existing studies suffer from the following limitations: (1) the number of source videos too small to represent the real-world scene; (2) advanced video encoders are not included; (3) the quality assessment studies are conducted on few out-dated devices; (4) the realistic throughput corpus are of limited size; (5) the studies do not cover a comprehensive list of ABR algorithms; (6) only one experiment protocol is utilized as the evaluation methodology, which inevitably results in a biased conclusion that favors a particular tradeoff among quantity, representativeness, and reliability; (7) the experiment data are not publicly available.

Table 2.2: Summary of adaptive bitrate streaming algorithms. Abbreviations: AM, arithmetic mean; HM, harmonic mean; EWMA, exponential weighted moving average; HMM, hidden markov model; MLP, multi-layer perceptron; RNN, recurrent neural network; QoE, quality-of-experience; RDO, rate distortion performance; LB, linear function of bitrate; NB, non-linear function of bitrate; LV, linear function of video quality assessment score; NV, nonlinear function of video quality assessment scores; TP, throughput prediction; BO, buffer occupancy; DH, download history; TH, throughput history; FC, information of future chunks; RL, reinforcement learning; DP, dynamic programming; SL, supervised learning; LUT, look-up table; CNN, convolutional neural network.

Study	Year	Throughput Predictor			Reward Function			Trajectory Planner		
		Objective	Model	Data-driven	Objective	Model	Data-driven	Input	Algorithm	Model
RB [63]	2010	$p(c_{t+1}\|c_t)$	AM	✗	QoE	LB	✗	TP	Greedy	–
AIMD [122]	2011	$p(c_{t+1}\|c_t)$	AM	✗	QoE	LB	✗	TP, BO	Heuristic	–
QDASH [141]	2012	$p(c_{t+1}\|c_t)$	Linear	✗	QoE	NB	✗	TP, BO, DH	Heuristic	–
Elastic [35]	2013	$p(c_{t+1}\|c_t)$	HM	✗	BO	–	✗	TP, BO	Greedy	–
Q-Learning [32]	2013	–	–	–	QoE	LB	✗	TH, BO, DH	RL	LUT
FAQ [33]	2014	–	–	–	QoE	LB	✗	TH, BO, DH	RL	LUT
PANDA [119]	2014	$p(c_{t+1}\|c_t)$	EWMA	✗	QoE	LB	✗	TP	Greedy	–
FESTIVE [102]	2014	$p(c_{t+1}\|c_t)$	HM	✗	QoE	NB	✗	TP, DH	Heuristic	–
SARA [103]	2015	$p(c_{t+1}\|c_t)$	HM	✗	QoE	LB	✗	TP, BO, DH, FC	Heuristic	–
BB [93]	2015	–	–	–	QoE + BO	LB	✗	BO	Greedy	–
RobustMPC [241]	2015	$p(c_{t+1}\|c_t)$	HM	✗	QoE	LB	✗	TP, BO, DH	DP	–
FastMPC [241]	2015	$p(c_{t+1}\|c_t)$	HM	✗	QoE	LB	✗	TP, BO, DH	SL	LUT
CS2P [200]	2016	$p(c_{t+1}\|c_t)$	HMM	✓	QoE	LB	✗	TP, BO, DH	DP	–
SDNDASH [16]	2016	$p(c_{t+1}\|c_t)$	EWMA	✗	QoE	LV	✗	TP, BO, DH	Greedy	–
BOLA [191]	2016	–	–	–	QoE + BO	NB	✗	BO	Greedy	–
ValueIter [30]	2016	–	–	–	QoE	NB	✗	TH, BO, DH	RL	LUT
Pensieve [131]	2017	–	–	–	QoE	LB	✗	TH, BO, DH, FC	RL	CNN
Oboe [3]	2018	$p(c_{t+1}\|c_t)$	baseline dependent		QoE	LB	✗	baseline dependent	RL + baseline	LUT
Comyco [92]	2019	–	–	–	QoE	LV	✓	TH, BO, DH, FC	SL	RNN
Fugu [237]	2020	$p(c_{t+1}, ..., c_{t+K}\|c_t)$	MLP	✓	QoE	LV	✗	TP, BO, DH	DP	–
RDOS	2020	$p(c_{t+1}, ..., c_{t+K}\|c_t, c)$	RNN	✓	RDO	NV - LB	✓	TP, BO, DH, FC	RL + DP	CNN

Table 2.3: Summary of adaptive bitrate streaming evaluation studies. Abbreviations: BU, bandwidth utilization; RD, rebuffering duration; BV, bitrate variation; PSNR, peak signal-to-noise ratio; SSIM, structural similarity index; UE, user engagement; QoE, quality-of-experience; RDO, rate distortion performance.

Study	Year	ABR	Environment				Evaluation		
		Algorithm	Video	Encoder	Throughput	Display	Methodology	Controlled	Criterion
[239]	2010	1	1	1	3	1	Objective	–	PSNR, RD
[139]	2011	1	1	1	5	1	Subjective	–	QoE
[2]	2011	3	1	1	∼10	1	Objective	✓	BU
[144]	2012	5	1	1	3	1	Objective	✓	BU, RD, BV
[35]	2013	4	1	1	2	1	Objective	✓	BU, RD, BV
[119]	2013	4	1	1	N/A	1	Objective	✗	BU, BV
[102]	2014	4	1	1	N/A	1	Objective	✗	BU, BV
[93]	2015	2	4 days of real-world streaming				A/B test	✗	UE
[103]	2015	2	1	1	4	1	Objective	✓	BU, BV
[241]	2015	6	1	1	2000	1	Objective	✓	QoE
[200]	2016	3	8 days of real-world streaming				Objective	✗	BU, RD, BV, QoE
[202]	2016	10	1	1	1	1	Objective	✓	BU, RD, BV
[16]	2016	5	1	1	6	1	Objective	✓	BU, RD, BV
[191]	2016	6	1	1	98	1	Objective	✓	QoE
[131]	2017	6	1	1	400	1	Objective	✓	QoE
[3]	2018	5	1	1	571	1	Objective	✓	QoE
[53]	2018	6	20	1	13	1	Subjective	✓	QoE
[11]	2018	4	15	1	7	1	Subjective	✓	QoE
[92]	2019	5	5	1	600	1	Objective	✓	QoE
[237]	2020	5	∼7 months of real-world streaming				Objective	✗	SSIM, RD, UE
Ours	2020	**15**	**250**	**3**	**∼20,000**	**3**	Objective	✓	QoE, RDO
Ours	2020	5	5	2	9	3	Subjective	✓	QoE, RDO

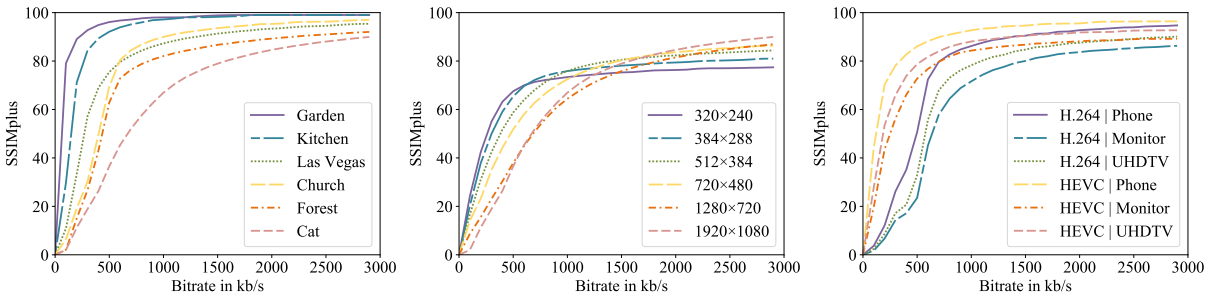
Chapter 3

Adaptive Streaming: From Bitrate Maximization to Rate-Distortion Optimization

Despite the diversity of instantiations, almost all ABR algorithms instantiate a bitrate maximization paradigm. These methods share a common formulation

$$\begin{aligned} \underset{\theta}{\text{maximize}} \quad & \mathbb{E}_{p_{\mathcal{E}}} \sum_{t=1}^{T} R_t(a_t) \\ \text{subject to} \quad & a_t = \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}_{1:t}) \\ & \mathbf{s}_{t+1} = \mathcal{E}(a_t, \mathbf{s}_{1:t}), \end{aligned} \tag{3.1}$$

where R_t, $\boldsymbol{\pi}$, and $\mathcal{E}$ represent the average bitrate of chunk t, the control policy, and the streaming environment, respectively. The formulation encodes the physical transmission and decision making process as follows. Before chunk $t+1$ is downloaded, the ABR controller $\boldsymbol{\pi}$ parametrized by $\boldsymbol{\theta}$, performs an *action* $a_t \in \mathcal{A}$ based on all previous *states* $\mathbf{s}_{1:t}$, where $\mathbf{s}_t \in \mathcal{S}$ for all t, to determine which representation to download. The state $\mathbf{s}_t$ generally represents the information about throughput history, buffer occupancy, and previous downloaded representations before a_t is taken. Given a bitrate decision a_t and the previous *states* $\mathbf{s}_{1:t}$, the *environment* $\mathcal{E}$ consisting of the characteristics of streaming video and the future throughput will download a corresponding representation of the next chunk,



(a) Impact of content (b) Impact of spatial resolution (c) Impact of encoder and display device

Figure 3.1: The relationship between bitrate and a perceptually motivated video quality model on (a) different video contents, (b) different encoding spatial resolutions, and (c) different video encoders and viewing devices.

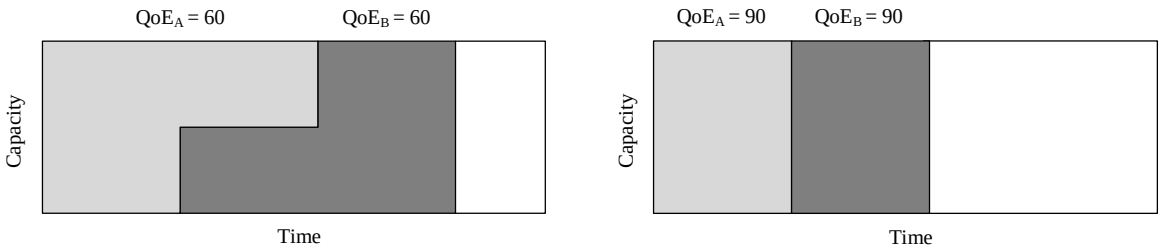
updates the buffer occupancy, tracks the throughput, continues playing the video from the buffer, and returns all the updated *state* $\mathbf{s}_{t+1}$ to the agent. Then the agent takes another action a_{t+1} based on the new states. By utilizing the bitrate maximization paradigm, one is implicitly making the following two assumptions. First, bitrate is a good estimate of video QoE. Second, service provider incurs no cost for the storage and transmission of each bit. Unfortunately, none of the assumptions hold true in practice. As a result of these unrealistic assumptions, bitrate maximization-based ABR systems often suffers from the following limitations.

- The Quality Definition Problem: It is often tempting to assume bitrate is a good measure of quality in an adaptive streaming system because 1) for a given video content, a given spatial/temporal resolution, and a given video encoder, typically higher bitrate leads to better QoE; 2) higher spatial/temporal resolution of the same video content typically leads to higher quality, but requires higher bitrate; 3) it is easy to operate and optimize as bitrate is accessible without fully decoding the compressed video stream. However, the assumption contradicts the rate-distortion theory [17], and may deteriorate in different compression, transmission and reproduction systems [36, 227, 229, 230]. A motivating example is shown in Figure 3.1, where the relationships between bitrates and quality scores predicted from a perceptually mo-

tivated, cross-resolution, and cross-device video quality model SSIMplus [169] at a variety of operating conditions are presented. We summarize the key observations as follows. First, there is a significant amount of rate-distortion variability across different video contents. Specifically, it takes very little resource (around 500 kb/s) to nearly losslessly encode the content "Garden", while encoding the content "Cat" at the same quality level takes more than 6 times bitrate. As a result, an ABR algorithm optimized on "Cat" may introduce significant bitrate waste when streaming "Garden". Second, rate-distortion curves at different encoding resolutions exhibit distinct characteristics, even for the same content. Each resolution may have a bitrate region in which it outperforms other resolutions. Third, there is significant diversity in the rate-distortion characteristics for a video encoded by different video encoders and viewed at different display ports.

- The Conflicting Demand Problem: The most fundamental problem with the traditional approach is the neglect/misjudgement of service providers' demand, which arises as a natural consequence of the second assumption. In particular, it is assumed that both video consumers and service providers are benefited from draining the available bandwidth. However, encoding, storage, and transmission at high bitrate run at the risk of server overload [38]. Currently, the best way to overcome the problem is to incorporate more servers and content delivery network [101], which are very expansive to purchase and maintain [26]. As a result, video service providers have a strong incentive to reduce bitrate consumption.

- The Tragedy of Commons Problem: The two unrealistic assumptions jointly introduce the Tragedy of Commons problem to the bitrate maximization-based ABR systems, where bitrate resources are distributed across different video consumers in a sub-optimal fashion. Specifically, the perceptual quality of streaming video usually exhibits a concave relationship with respect to bitrate, suggesting that marginal increase in QoE gradually decreases with the increment of bitrate. Such "best-effort" approach miss out opportunities to save bandwidth usage for competing video players sharing the same bottleneck and will eventually cause network congestion or sub-optimal viewer QoE. To illustrate this point, consider the scenario in Figure 3.2



(a) (b)

Figure 3.2: Illustration of bandwidth sharing scenarios. (a) bitrate maximization paradigm. (b) rate-distortion optimization paradigm.

(a), where two video players sharing bandwidth join the network sequentially. In the case of bitrate maximization strategy, both players select the highest bitrate when it does not observe a completing player. Consequently, Player A does not finish the download in time when player B joins the network. According to the default characteristics of TCP, both players have to equally shares the bandwidth and download videos at a significantly lower bitrate, which inevitably lead to in sub-optimal QoE. The phenomena is widely known as Tragedy of Commons in Economics [130]. The problem can be efficiently solved if both players download videos with a sufficiently high quality but at a lower bitrate such that they do not need to compete for resources, as illustrated in Figure 3.2 (b).

Some additional problems that may be caused by the second assumption, *e.g.*, unnecessarily increasing video bitrate even when there is no room for QoE gain; excessive bitrate is prone to create rebuffering events, *etc.*

3.1 Rate-Distortion Optimized Adaptive Streaming

Our goal is to provide a new methodology to guide the design and evaluation of ABR algorithms. In this section, we provide solutions to the aforementioned problems one by

one. We show that the new paradigm RDOS arises as a natural consequence of these solutions.

To address the quality definition problem, we first disentangle bitrate R from the notion of QoE Q. We argue that bitrate and QoE are intrinsically two different quantities that should not be considered equivalent. Since HVS is the ultimate receiver of streaming videos, the most reliable way to evaluate QoE is by performing a subjective experiment. It is worth noting that once a viewer observes a streaming video $\mathbf{x}$, the QoE is independent of bitrate R which is an innate component of the distortion process $\boldsymbol{\phi}$. Put another way, human viewers can reliably evaluate QoE without the access to the encoding bitrate. In practice, however, subjective evaluation is inconvenient, time-consuming, and expensive. Most importantly, it is not applicable in the real-time ABR decision making. Alternative, one can construct an artificial vision system to replace the HVS, which has been investigated for decades [224]. Being independent of the bitrate, such perceptual QoE measurement can generalize well to a wide range of source content, encoding specifications, viewing devices, and even other distortion processes.

Equipped with a better understanding of QoE, we are now ready to solve the conflicting demand problem. To balance the adversarial objectives between video consumers and service providers, we propose to formulate the adaptive streaming problem as a rate-distortion optimization problem

$$\begin{aligned} \underset{\theta}{\text{maximize}} \quad & \mathbb{E}_{p_{\mathcal{E}}}\left[Q(\mathbf{s}_{1:T}) - \lambda \textstyle\sum_{t=1}^{T} R_t(a_t)\right] \\ \text{subject to} \quad & a_t = \mathcal{G}_\theta(\mathbf{s}_{1:t}) \\ & \mathbf{s}_{t+1} = \mathcal{E}(a_t, \mathbf{s}_{1:t}), \end{aligned} \tag{3.2}$$

where $\lambda > 0$ denotes a weighting parameter. In general, the overall QoE in each streaming session is a function of all the visited states $\mathbf{s}_{1:T}$, which encode the quality adaptation trajectories and the information about each rebuffering event. This new RDOS paradigm respects both the demand of viewers, who would like to optimize their QoE $Q(\mathbf{s}_{1:T})$, and the requirement of service providers, who are inclined toward minimizing bitrate consumption $\sum_{t=1}^{T} Q_t(a_t)$.

The RDOS paradigm also alleviates the inefficiency problem to some extent, since it does not blindly maximizes bitrate usage. Instead, the rate-distortion optimized streaming

agent seeks a solution with the marginal QoE improvement higher than the worth of its bitrate increment. Such conservative strategy usually selects chunks of sufficiently high QoE without spending too much bitrate, reduces the probability of network sharing, and thus results in a better overall QoE. The phenomena is illustrated in Figure 3.2 (b).

3.2 Why RDOS?

We argue that the ultimate goal of bitrate adaptation is to balance the limited supply of network resource and the increasing demand of users' QoE. Rate-distortion theory appears to be a natural fit to the resource allocation problem. To further motivate the use of RDOS, we provide three interpretations to understand the framework.

We can view RDOS as a source coding problem with a fidelity criterion, closely related to vector quantization. Specifically, for the number of bits required to transmit the video under a time-varying channel, RDOS provides a version of the signal with a certain fidelity. The bitrate adaptation engine can be interpreted as an online video encoder that adaptively picks chunk-level encoding configurations from a determinant codebook according to the environment status. The criterion for the encoding strategy is the minimization of a Lagrangian cost function wherein the perceptual distortion is weighted against the number of bits associated with each video chunk using a Lagrange multiplier.

An alternative interpretation is to view the bitrate adaptation problem as a QoE maximization problem. Although video quality usually exhibits a monotonic relationship with respect to encoding bitrate, the function connecting them is usually nonlinear, time-varying, and signal-dependent. Further, the problem of QoE maximization becomes increasingly ill-conditioned as we increase the number of dimensions in streaming videos such as spatial resolution, frame rate, bit depth, and viewing devices. In particular, many video representations in the attribute-quality space may possess the same perceptual quality. The ill-conditioning leads to a lack of consistency in the representation selection, resulting in inefficient resource allocations. The Lagrangian formulation can regularize the representation selection, leaning towards a solution with the minimum bitrate usage on an equal-distortion contour.

Another useful approach to learn RDOS is to view the bitrate adaptation problem as a supply decision problem in an abstract QoE market. In this market, the streaming service company, who acts as the buyer, purchases the goods, *i.e.* QoE, from the ABR controller, who acts as the seller. To better explain the economic view, we rewrite the objective function in (3.2) as

$$\underset{\theta}{\text{maximize}} \quad \frac{1}{\lambda}\tilde{Q} - \tilde{R}, \tag{3.3}$$

where $\tilde{Q} = \mathbb{E}_{p_\mathcal{E}} \sum_{t=1}^{T} Q(a_t, \mathbf{x}_t, \mathbf{x}_{t+1})$ denotes the expected overall QoE, and $\tilde{R} = \mathbb{E}_{p_\mathcal{E}} \sum_{t=1}^{T} R_t(a_t)$ denotes the expected total bitrates. The first term in (3.3) can be interpreted as the total "revenue" made by the seller, where $\frac{1}{\lambda}$ is the "price" in bitrates per unit QoE, and the expected total QoE is the quantity of goods the ABR controller is willing to supply. The second term can be interpreted as the total cost for delivering this amount of QoE. Therefore, (3.3) indicates the total "profit" the ABR controller could make at the "price" of $\frac{1}{\lambda}$. To maximize the total "profit", the seller should deliver a QoE at which the marginal cost equals the price [130], *i.e.* $\frac{1}{\lambda} = \frac{\partial \tilde{R}}{\partial \tilde{Q}}$.

Taking the framework as a starting point, we delve into the design of individual modules including a QoE model Q, a reinforcement learning-based bitrate adaptation function that explicitly optimizes the rate-distortion performance, and a realistic and content-aware state space $\mathcal{X}$.

3.3 Economic Interpretation

In the economic interpretation, we analogize the ABR agent as the supplier of QoE in a virtual market, so we may use the supply curve to characterize the ABR agent. In the context of economy, the supply curve describes the relationship between the price and the supplied quantity [130]. In order to maximize the profit, the supplier often produces as many goods as possible until the marginal cost of an extra unit of goods equals the price in the market. This means that the supply curve can also be determined by the mapping between the supplied quantity and the marginal cost at this amount of output. In the scenario of adaptive streaming, the cost is network bandwidth measured in bitrate, and the output is the QoE. It turns out that the rate-distortion function exactly describes the

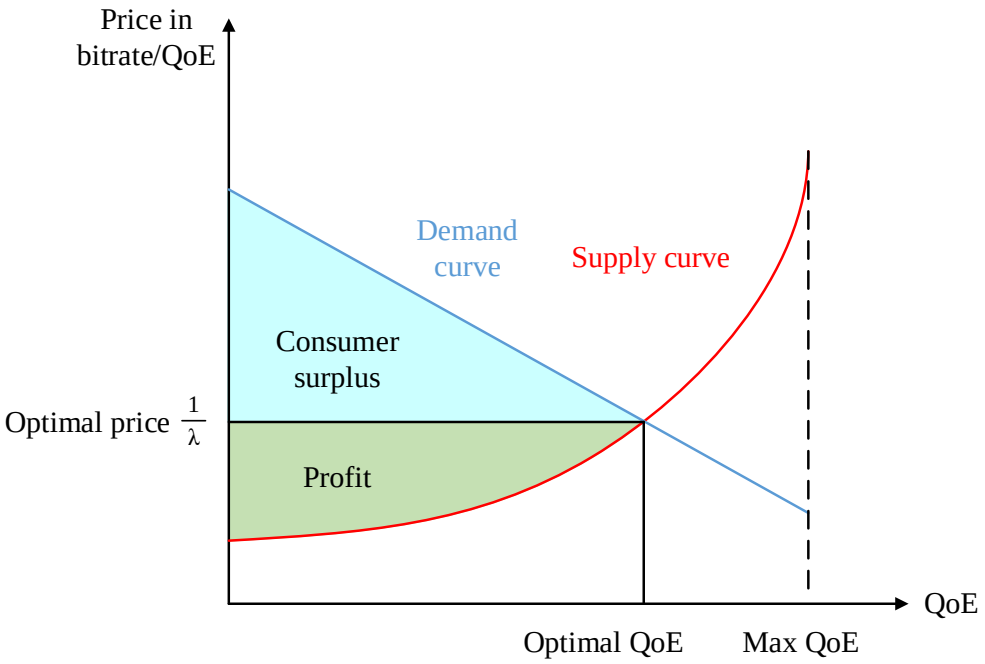


Figure 3.3: The illustration of the equilibrium "price" in the virtual market.

relationship between the cost and the supplied quantity. As a result, the supply curve can be obtained by taking the derivative of the inverse rate-distortion function as shown in Figure 3.3. Under some mild conditions, the rate-distortion function can be shown to be concave [21], suggesting that the supply curve is monotonically increasing. It is also worth noting that the supply curve varies according to the video content and the network condition.

Regarding the demand side of the QoE market, we may also define a demand curve to describe at each price, how much QoE people would like to consume. In fact, the demand curve in the virtual market depends on its counterpart in a real-world market of video services. We conceptually show the interaction between the two markets in Figure 3.4. In the physical market, an end user pays money to video service companies for streaming videos at certain quality levels. This is similar to what we do when subscribing a video service, such as Netflix. Then the companies invest the money in bitrate resources, which are then used to "buy" the QoE from the ABR agent in the virtual market as we discussed in the previous paragraph. Therefore, the demand curve depends on the customers' willingness to pay for an extra unit of QoE in the physical market. A recent study has demonstrated that video service providers exhibit a diminishing marginal utility of QoE with both mathematically models and empirical evidence [82], suggesting that the demand curve is monotonically decreasing. A conceptual demand curve is also drawn in Figure 3.3.

Now we can answer the question: what is the optimal value for λ? The answer is that it depends on the characteristics of the QoE market. In an ideal case, the QoE market can achieve an equilibrium price at the intersection of the supply and demand curves thanks to the interplay of many consumers and service providers. It has been proved that the equilibrium price yields the maximum market efficiency [130], which can be measured as sum of the consumer surplus and the profit (producer surplus) as shown in Figure 3.4. The equilibrium price is thus optimal in this sense. In other scenarios, we may also apply appropriate economic tools to analyze the optimal λ. Therefore, the proposed RDOS framework not only redefines the objective of the adaptive streaming task, but also provides a systematic means to deal with various application scenarios.

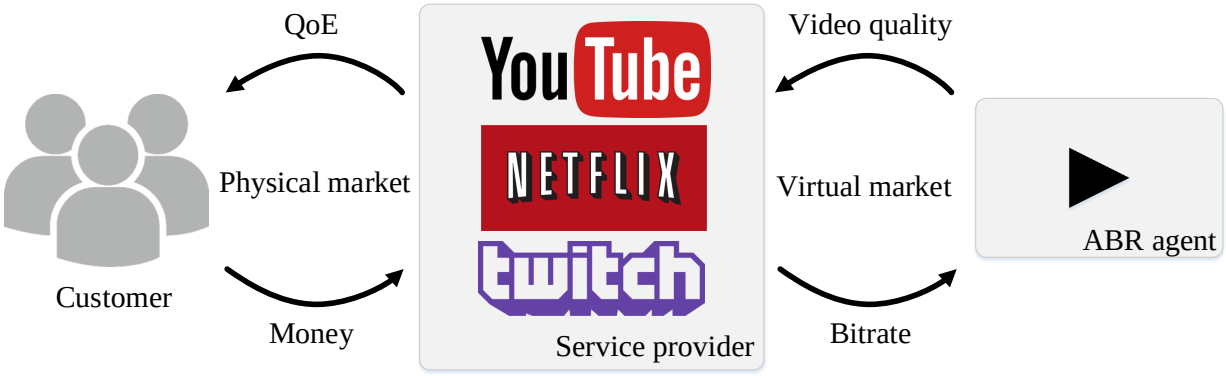


Figure 3.4: The illustration of the physical market between customers and video service providers, and the virtual market between the service providers and ABR agents.

3.4 Connections to Bitrate Maximization Scheme

In contrast to the rate-distortion performance, most existing ABR algorithms [3, 30, 32, 93, 102, 119, 131, 141, 191, 200, 212, 241] are focused on maximizing the bitrates of streaming videos under given network conditions. By comparing (3.1) to (3.2), we find that the bitrate maximization scheme treats bitrate usage as delivered QoE, and set the value of λ to zero, *i.e.* the price of each unit QoE to infinity. Such settings cause at least three drawbacks. First, equaling quality to bitrate implies that the marginal cost of QoE keeps constant, leading to a completely flat supply curve as shown in Figure 3.5. Second, for different video contents, the bit maximization scheme generates an identical supply curve. The extremely biased estimate of the supply curve will surely result in a sub-optimal market efficiency and an impaired QoE. Third, the infinity price always drives the ABR agent to produce the maximum QoE, where the marginal return to the consumer might be very low. In other words, we may miss out the opportunity to save bitrates while still delivering satisfactory QoE.

Although we have assumed the supply and demand curves of QoE to be monotonic for illustration purpose, the RDOS framework does not rely on these assumptions. In fact, the economic analysis is still valuable in finding the optimal operating point to balance between QoE and bitrate based pricing. Specifically, the RDOS framework defines the

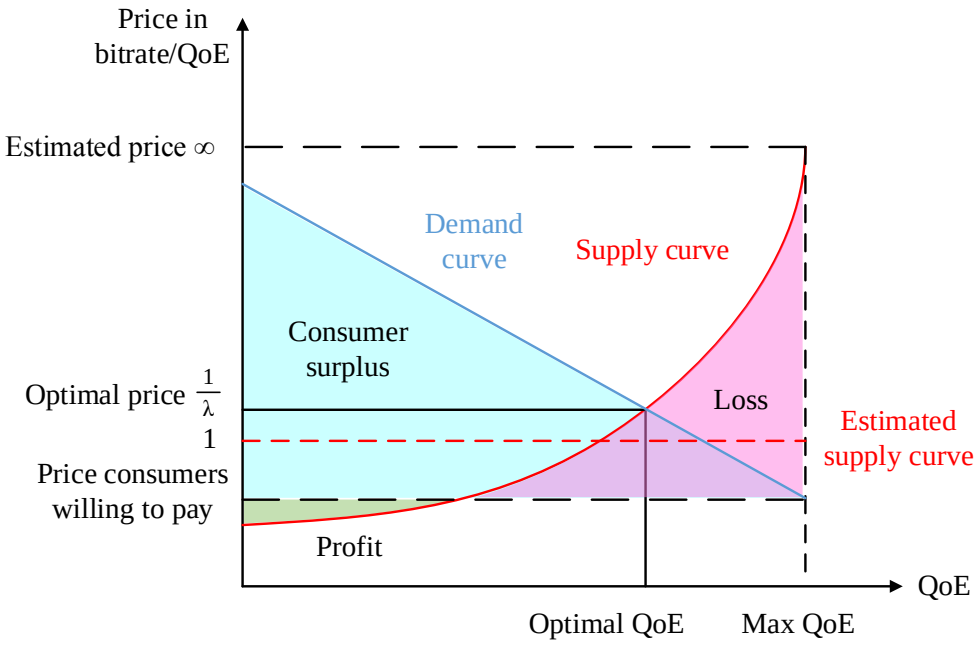


Figure 3.5: The illustration of the the QoE market under the bitrate maximization scheme.

adaptive streaming problem as a trade in a virtual market, where the behaviors of buyer and seller are influenced by the demand and supply of QoE. The most economically efficient approach requires a precise model of demand curve and supply curve, such that an optimal price could be derived. The benefits of being able to define a price $\frac{1}{\lambda}$ are twofold. The parameter not only reflects the true cost in video distribution, and may also be used as a way to define priority of end users.

3.5 Summary

In this section, we propose a novel paradigm for the development of ABR algorithms. In contrast to the traditional bitrate maximization paradigm, the new RDOS must operate at any given point along the rate-distortion curve, effectively balancing the conflicting requirement from video consumers and service providers. We motivate the new design philosophy from three distinct perspectives. To instantiate the RDOS paradigm, one has to

develop an objective QoE model that can accurately predicts subjective quality evaluation, which is the topic of the subsequent chapter.

Chapter 4

A Bayesian Quality-of-Experience Model for Adaptive Streaming Videos

In Chapter 2, we have learnt that existing objective QoE models share a common two-layer Bayesian network. The first stage maps the raw streaming video signal $\mathbf{x}$ to a latent space $\mathbf{z}$ with a much lower dimensionality, where the QoE prediction problem should hopefully be easier to solve.

Traditional QoE models employ chunk-level bitrate and rebuffering duration as the feature set, which achieved limited success in predicting subjective QoE ratings. In our previous work, we developed a new feature extractor which maps each video chunk $\mathbf{x}_t$ into a three dimension latent space $\mathbf{z}_t$, consisting of perceptual video quality, rebuffering duration, and the magnitude of quality change. Thanks to the informative prior about HVS, the feature set has demonstrated outstanding performance in various independent studies [8, 10, 11, 16, 51, 54]. Most importantly, the feature set does not comprise bitrate, making the resulting QoE model independent of bitrate. As illustrated in Chapter 3, the bitrate independent QoE model is an important component in the RDOS paradigm. Therefore, we inherit the feature extractor from our previous study, and focus on the regression model in this chapter.

There have been two distinct approaches to model the regression function $p(y|\{\mathbf{z}_t\}_{t=1}^T; \boldsymbol{\theta}_2)$. The first approach makes strong prior assumptions about the regression model parameter $p(\boldsymbol{\theta}_2)$. In addition to the Markov assumption, temporal homogeneous assumption, and additive assumption, objective QoE models in this category also make assumptions about the specific form of activation function along each dimension. A common drawback of the approach is that the prior distribution is often selected on the basis of mathematical convenience rather than as a reflection of any prior beliefs. The second approach aims to approximate the posterior parameter distribution solely from the likelihood function $p(\mathcal{D}_\mathbf{z}|\boldsymbol{\theta}_2)$. Unfortunately, these models suffer from strong generalization problem due to the lack of training data.

Motivated by the limitations of the existing methods, we aim to develop an objective QoE model that can fuse the prior knowledge about QoE and subject-rated streaming video data in a principled way. Bayesian method appears to be a natural fit to the information fusion problem. To be specific, one can employ the maximum a posteriori estimator to obtain the optimal model configuration

$$\begin{aligned} \boldsymbol{\theta}_2^* &= \arg\max_{\boldsymbol{\theta}_2} p(\boldsymbol{\theta}_2|\mathcal{D}_z) \\ &= \arg\max_{\boldsymbol{\theta}_2} p(\mathcal{D}_z|\boldsymbol{\theta}_2)p(\boldsymbol{\theta}_2). \end{aligned} \tag{4.1}$$

But even given such a unified framework, the QoE prediction problem is still non-trivial. In particular, traditional prior distributions rely on a number of strong assumptions and generalizations, strictly restricting the space of feasible solution. For example, the QoE function can vary significantly from exponential and logarithmic functions, even with the optimal model parameters. As will be demonstrated in subsequent sections, existing prior models cannot make efficient use of the training data. On the other hand, simply removing these assumptions would degenerate the maximum a posteriori approach to the maximum likelihood estimator, resulting in the overfitting problem. Therefore, a meaningful prior probability model for the HVS configuration is of central importance for this application. While many recent works acknowledge the importance of prior knowledge in the objective QoE models [10, 54, 127], a careful analysis, modeling, and evaluation of the models has yet to be done. We wish to address this void.

4.1 Prior QoE Model

In this section, we derive a prior QoE model by analyzing a corpus of subjective QoE experiments. To simplify the discussion, we start with a deterministic formulation of the prior QoE model. In the end of the derivation, we will also present a probabilistic interpretation of the resulting prior model.

4.1.1 Deterministic View

Formally, the overall QoE can be denoted as $Q(\{p_t,\ \tau_t,\ \Delta p_t\}_{t=1}^{T})$, where p_t, τ_t, and $\Delta p_t = p_t - p_{t-1}$ represent the presentation quality (which may be measured by any modern video quality assessment models such as VMAF [117] and SSIMplus [169]), the rebuffering duration, the magnitude of quality adaptation of chunk t, respectively. T denote the number of chunks in the streaming video. Defining the space of QoE functions helps us build a model of these functions. It not only guides us as to the form such a model should take, but also determines the constraints these functions must satisfy. We begin by summarizing observations from a collection of existing subjective QoE studies, and then formulate the domain knowledge to define the space of these functions. For the brevity of math formulation, we will use simplified notations for the rest of this section unless otherwise stated. Specifically, we will omit all the identical variables of the QoE function Q in the same equation, and only emphasize the factors that are different. First, various subjective tests [42, 87] have attested that rebuffering duration is negatively correlated with the overall QoE of streaming videos. Formally, we may summarize this observation by

$$Q(\tau_t = \tau^a) \geq Q(\tau_t = \tau^b), \forall \tau^a \leq \tau^b, t. \tag{4.2}$$

Note that we have used the simplified notation in (4.2) to show that the two compared video streams are only different in the rebuffering duration of chunk t.

The second assumption is that, given the same rebuffering length, the QoE drop tends to be greater when the presentation quality of the previous chunk is higher, *i.e.*

$$\begin{aligned} &Q(p_{t-1} = p^a, \tau_t = 0) - Q(p_{t-1} = p^b, \tau_t = \tau) \leq \\ &Q(p_{t-1} = p^a, \tau_t = 0) - Q(p_{t-1} = p^b, \tau_t = \tau), \forall \tau, p^a \leq p^b, t. \end{aligned} \tag{4.3}$$

Such a trend has been observed in recent subjective tests [12, 54], and may be explained by the expectation confirmation theory [153].

The third assumption is elicited from the fact that, given a constant presentation quality and a fixed total duration of rebuffering, the overall QoE degrades as the number of rebuffering occurrences increases [86, 142, 159]. Mathematically, this may be expressed as

$$Q(\tau_{t-1} = \tau^a, \tau_t = \tau^b) \leq Q(\tau_{t-1} = 0, \tau_t = \tau^a + \tau^b), \forall \tau^a, \tau^b, t. \tag{4.4}$$

The fourth remark is that, given the same rebuffering duration, videos with higher presentation quality consistently deliver higher overall QoE, despite the greater penalty for the rebuffering event [127]. This statement can be formulated as

$$Q(p_t = p^a) \leq Q(p_t = p^b), \forall p^a \leq p^b, t. \tag{4.5}$$

We then analyze the functional properties with respect to the quality adaptation. The fifth assumption suggests that people always assign a penalty to presentation quality degradation, reward to quality elevation, and neither penalty nor reward when no quality adaptation occurs [51, 72, 142, 168]. Mathematically, the assumption can be expressed as

$$\begin{cases} Q(\Delta p_t = \delta p^a) \leq Q(\Delta p_t = 0), & \forall \delta p^a \leq 0, t \\ Q(\Delta p_t = \delta p^b) \geq Q(\Delta p_t = 0), & \forall \delta p^b \geq 0, t \end{cases}. \tag{4.6}$$

Further analysis [51, 142, 150, 168] on the relationship between the QoE adjustment and the intensity of quality adaptation Δp indicates that subjects tend to give greater QoE penalty or reward when quality drops or improves by a greater amount. This finding, together with the fifth assumption, prompts our sixth assumption: QoE is monotonically increasing with regards to Δp:

$$Q(\Delta p_t = \delta p^a) \leq Q(\Delta p_t = \delta p^b), \forall \delta p^a \leq \delta p^b, t. \tag{4.7}$$

Experiments in [51] find that quality degradation occurring in the high quality range leads to greater amount of penalty than that occurring in the low quality range, while

quality elevation in the high quality range results in smaller rewards. Such an observation leads to the seventh assumption that

$$\begin{aligned} &Q(p_t = p^a, \Delta p_t = \delta p) - Q(p_t = p^a, \Delta p_t = 0) \geq \\ &Q(p_t = p^b, \Delta p_t = \delta p) - Q(p_t = p^b, \Delta p_t = 0), \forall \delta p, p^a \leq p^b, t. \end{aligned} \tag{4.8}$$

Another commonly observed trend in QoE is that the reward for a positive quality adaptation is relatively smaller than the penalty for a negative one given the same intensity of quality adaptation and the same average presentation quality [51, 150, 168]. Formally, this can be summarized by

$$\begin{aligned} &Q(p_t = p^a, \Delta p_t = 0) - Q(p_t = p^a, \Delta p_t = -\delta p) \geq \\ &Q(p_t = p^a - \delta p, \Delta p_t = \delta p) - Q(p_t = p^a - \delta p, \Delta p_t = 0), \forall p^a, \delta p \geq 0, t. \end{aligned} \tag{4.9}$$

In summary, we define the space of QoE functions Q as

$$\mathcal{W}_Q \coloneqq \{Q : \mathbb{R}^{3T} \to \mathbb{R} | Q \text{ satisfying constraints (4.2) to (4.9)}\}. \tag{4.10}$$

The inequality constraints in (4.10) represent a cone [19], which is convex by its definition.

4.1.2 Probabilistic View

The conversion from the inequality constraints in (4.10) to its probability representation is straight-forward. Let $\boldsymbol{\theta}_2$ denote the parameters of the regression function Q, then the constraint in (4.2) corresponds to the following prior distribution

$$p_1(\boldsymbol{\theta}_2) = \begin{cases} \epsilon, & \forall Q(\{p_t, \tau_t, \Delta p_t\}_{t=1}^T; \boldsymbol{\theta}_2) \text{ satisfying (4.2)} \\ 0, & \text{otherwise} \end{cases}, \tag{4.11}$$

where ϵ represents certain probability density for each feasible parameter configuration such that $p_1(\boldsymbol{\theta}_2)$ sum to 1. The constraints in (4.3)–(4.9) can be transformed into prior probability distributions of $\boldsymbol{\theta}_2$ in a similar fashion, which can be denoted as $p_2(\boldsymbol{\theta}_2)$–$p_8(\boldsymbol{\theta}_2)$, respectively. The simple aggregation of constraints in (4.10) implicitly assumes the independence of individual assumptions. Therefore, the joint prior probability distribution of QoE models may be obtained by

$$p(\boldsymbol{\theta}_2) = \frac{\prod_{i=1}^8 p_i(\boldsymbol{\theta}_2)}{\int_{\boldsymbol{\theta}} \prod_{i=1}^8 p_i(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta}}. \tag{4.12}$$

4.2 A Bayesian QoE Model

Our discussion on the prior QoE models has been encouraging. However, the general form of the QoE function still exhibits a very high dimensionality. To obtain a meaningful approximation, some further assumptions have to be made. In this section, we present the roadmap to design a perceptually grounded objective QoE model.

4.2.1 Additional Assumptions

The observations from existing psychophysical experiments not only illustrate the feasible functional form of QoE models, but also point out the joint impact among the three dimensional features in QoE. As a result, we can effectively replace the specific form assumption and the additive assumption in the traditional prior model by the HVS imposed constraints in (4.10). However, existing subjective QoE studies do not provide enough information in the temporal aspects. For example, how an impairment that appears early in a streaming session affects the QoE in the subsequent QoE in a long run is still a subject of ongoing research. There have also been limited studies [174] investigating the validity of the temporal homogeneous assumption. In this study, we adopt a conservative approach by inheriting the Markov assumption and the temporal homogeneous assumption. Nevertheless, the proposed Bayesian framework is general enough to incorporate more prior knowledge once they become available.

Mathematically, the Markov assumption and the temporal homogeneous assumption can be jointly expressed by

$$Q(\{p_t, \tau_t, \Delta p_t\}_{t=1}^{T}) = \frac{1}{T}\sum_{t=1}^{T} q(p_t, \tau_t, \Delta p_t),$$

where $q(\cdot)$ is the chunk-level QoE function, which is invariant to t. For simplicity, we will drop the subscript t in the rest of this section unless otherwise specified. By incorporating these assumptions, we reduce the original problem to the estimation of a three dimensional function $q(p_t, \tau_t, \Delta p_t)$.

4.2.2 Parameterization

Strictly speaking, $\mathcal{W}$ is a space of continuous functions, but we may approximate it in terms of a vector space by densely sampling the supporting domain of Q. To avoid potential bias introduced by models with specific form, we propose to use a non-parametric model. Let the supporting domain of Q be $\{(p,\ \tau,\ \Delta p)|p \in [0, P_{\max}],\ \tau \in [0, \tau_{\max}],\ \Delta p \in [-p, P_{\max} - p]\}$, where $P_{\max}$ and $\tau_{\max}$ indicate the best quality and maximum rebuffering duration, respectively. By uniformly sampling p, τ, and Δp, we can represent the function Q with a finite-size tensor $\mathbf{Q} \in \mathbb{R}^{(I+1)\times(J+1)\times(K+1)}$. The element $q_{i,j,k}$ denotes the QoE at $(p,\ \tau,\ \Delta p) = \left(\frac{i-1}{I}P_{\max}, \frac{j-1}{J}\tau_{\max}, \frac{k-1}{K}P_{\max}\right)$. We then vectorize $\mathbf{Q}$ as $\mathbf{q} \in \mathbb{R}^{(I+1)\times(J+1)\times(K+1)}$ for the convenience of further formulation. We employ the uniform vectorization for two reasons. First, the exact form of QoE functions (*e.g.* exponential, logarithmic) cannot be known a priori. To this regard, the uniform sampling implicitly serves as a non-informative prior on the form of QoE functions. Our second motivation is closely related to the flat assumption, which will be detailed in subsequent discussion. In particular, when the QoE functions are band-limited, they can be fully recovered from these samples when the sampling density is larger than the Nyquist rate. Finally, we are able to approximate the functional space $\mathcal{W}$ with a vector space

$$\mathcal{W}_{\mathbf{q}} \coloneqq \{\mathbf{q} \in \mathbb{R}^{(I+1)\times(J+1)\times(K+1)} | \mathbf{G}\mathbf{q} \leq \mathbf{h}, \mathbf{B}\mathbf{q} = \mathbf{c}\},$$

where $\mathbf{G}, \mathbf{h}, \mathbf{B}$ and $\mathbf{c}$ are constructed so that all the entries in $\mathbf{q}$ should satisfy the constraints in (4.10).

4.2.3 Model Training

Even though the theoretical space of the rebuffering QoE function is restricted to a cone, it is still infinite-dimensional. Ideally, the optimal rebuffering QoE function should be the one that best explains the subjective data and lives in the theoretical space. Specifically, given a training set of $\mathcal{D}_{\mathbf{x}}$ video sequences, each of which has a QoE rating Q, we want to obtain a vector $\mathbf{q}^* \in \mathcal{W}_{\mathbf{q}}$ that minimizes the mean squared error between the model

prediction and subject-rated data

$$\epsilon_{\mathrm{F}} \coloneqq \frac{1}{\mathcal{D}_{\mathbf{x}}} \sum_{m=1}^{\mathcal{D}_{\mathbf{x}}} \left[Q - \frac{1}{T} \sum_{t=1}^{T} q_{i_{m_t}, j_{m_t}, k_{m_t}} \right]^2,$$

where i_{m_t}, j_{m_t}, and k_{m_t} encode the corresponding indices of presentation quality, rebuffering duration, and quality adaptation magnitude for the t-th chunk of m-th video in the vector $\mathbf{q}$, respectively. However, existing subject-rated streaming video datasets contain very limited samples, which are sparsely distributed in the feature space. In particular, some (p, τ, Δp) combinations never appear in the training set, suggesting the optimization problem is ill-conditioned. To obtain a meaningful solution, we impose flat prior on the function Q. Mathematically, flatness regularization can be represented as the second-order differences along i, j, and k axes

$$\epsilon_{\mathrm{S}} \coloneqq \frac{1}{(I+1)(J+1)(K+1)} \sum_{i=1}^{I+1} \sum_{j=1}^{J+1} \sum_{k=1}^{K+1} \left[\left(\frac{\partial^2 q_{i,j,k}}{\partial i^2} \right)^2 + \left(\frac{\partial^2 q_{i,j,k}}{\partial j^2} \right)^2 + \left(\frac{\partial^2 q_{i,j,k}}{\partial k^2} \right)^2 \right]. \tag{4.13}$$

It is not hard to see that both ϵ_{F} and ϵ_{S} take quadratic forms of $\mathbf{q}$. As a result, we are able to estimate the rebuffering QoE matrix $\mathbf{Q}$ by solving the following quadratic programming problem

$$\begin{aligned} \underset{\mathbf{q}}{\text{minimize}} \quad & L = \epsilon_{\mathrm{F}} + \alpha \epsilon_{\mathrm{S}} \\ \text{subject to} \quad & \mathbf{q} \in \mathcal{W}_{\mathbf{q}}, \end{aligned} \tag{4.14}$$

where $\alpha > 0$ is a weighting factor. Once the optimization problem is solved, $\mathbf{q}^*$ is saved as a look-up table to query the QoE score of each video segment. The convexity of $\mathcal{W}_{\mathbf{q}}$ and the objective function implies that there exists a unique solution for the optimization problem. The problem can be efficiently solved with projected gradient descent-based algorithms such as alternating direction method of multipliers [22]. Minimizing the loss function in (4.14) is equivalent to solving the maximum a posteriori problem (4.1), with a Gaussian likelihood function and a prior probability distribution given by the product between a Gaussian distribution over (4.13) and a uniform distribution in (4.12).

Table 4.1: Comparison of objective QoE models. Notations: r, bitrate; τ, rebuffering duration; Δr, bitrate variation; p, presentation quality measured by state-of-the-art video quality assessment methods; Δp, quality variation; s, spatial resolution. Abbreviations: QP, quantization parameter; ML, maximum likelihood; MAP, maximum a posteriori.

QoE model	Features	Markov	Temporal homogeneity	Additive	Functional form	Training method
Mok2011 [141]	τ	✓	✓	✓	linear	—
FTW [86]	τ	✓	✓	✓	exponential	—
Liu2012 [124]	r, τ	✓	✓	✓	linear	—
Xue2014 [236]	QP, τ	✓	✓	✓	logarithmic	ML
Yin2015 [241]	r, τ, ΔR	✓	✓	✓	linear	—
Spiteri2016 [191]	r, τ	✓	✓	✓	logarithmic	—
Bentaleb2016 [16]	p, τ	✓	✓	✓	linear	—
SQI [54]	p, τ, Δp	✓	✓	✗	exponential	—
P.1203 [157]	r, s, τ, Δr, QP	✗	✗	✗	random forest	ML
VideoATLAS [8]	p, τ, Δp	✗	✗	✗	SVR	ML
BSQI	p, τ, Δp	✓	✓	✗	non-parametric	MAP

4.3 Experiments

In this section, we first describe the experimental setups including considered objective QoE models, benchmark databases, and evaluation criteria. We then compare BSQI with classic and state-of-the-art objective QoE models. Furthermore, we also developed a efficient methodology for examining the best-case performance of objective QoE models. Finally, we conduct a series of ablation experiments to identify the contributions of the core factors in BSQI.

4.3.1 Experimental Setup

Objective QoE Models

We evaluate the performance of 11 objective QoE models for adaptive streaming videos. The competing algorithms are chosen to cover a diversity of design philosophies, including 8 classic parametric QoE models: FTW [86], Mok2011 [141], Liu2012 [124], Xue2014 [236], Yin2015 [241], Spiteri2016 [191], Bentaleb2016 [16], and SQI [54], 2 state-of-the-art learning-

based QoE models: VideoATLAS [8] and P.1203 [157], and the proposed BSQI. A description of the existing QoE models is shown in Table 4.1. The implementation for VideoATLAS are obtained from the original authors and we implement the other nine QoE models. We have made the implementation of the models publicly available at https://github.com/zduanmu/pysqoe. For the purpose of fairness, the parameters of all models are optimized on the WaterlooSQoE-I [54] and the Waterloo Streaming QoE Database-II (WaterlooSQoE-II) [51] datasets, except for P.1203 [157] whose training methodology is not specified in the original paper. The WaterlooSQoE-I dataset contains 60 compressed videos, 60 compressed videos with initial buffering, and 60 compressed videos with rebuffering. The WaterlooSQoE-II dataset involves 588 video clips with variations in compression level, spatial resolution, and frame-rate. For the models with hyper-parameters, we randomly split the datasets into 80% training and 20% validation set, and the hyper-parameters with the lowest validation loss are chosen. For BSQI, we set the maximum rebuffering duration τ_{max} to 10, while the penalty of a rebuffering event longer than 10 can be easily obtained by extrapolating the tensor $\mathbf{Q}$. We set the step size $I = J = K$ to 10, roughly characterizing the standard deviation of subjective presentation quality evaluation. The maximum presentation quality value $p = 100$ is inherited from state-of-the-art VQA measures SSIMplus and VMAF. Although we can learn a initial buffering experience tensor independent from $\mathbf{Q}$, it introduces unnecessary model complexity. Instead, we discount the impact of initial buffering with $\frac{1}{9}$ and set the expectation to the initial quality p_{-1} to 80 following the recommendation by [54]. We apply Operator Splitting Quadratic Program (OSQP) [195] to solve the quadratic programming problem in (4.14). The fidelity-flatness tradeoff parameter $\alpha = 1$ is optimized on the validation set. In the subsequent section, we will also show that BSQI performs consistently over a broad range of α.

Benchmark Databases

We compare BSQI with state-of-the-art objective QoE models on four subject-rated adaptive streaming video datasets, including LIVE-NFLX-I [12], LIVE-NFLX-II [11], WaterlooSQoE-III [53], and WaterlooSQoE-IV [43]. The LIVE-NFLX-I dataset consists of 112 streaming videos derived from 14 source content with 8 handcrafted playout patterns. The LIVE-NFLX-II dataset consists of 420 streaming videos generated from content-adaptive encod-

Table 4.2: PLCC between the objective QoE model prediction and MOS on the benchmark datasets.

QoE model	LIVE-NFLX-I	LIVE-NFLX-II	WaterlooSQoE-III	WaterlooSQoE-IV	Average	Weighted Average
Mok2011 [141]	0.292	0.512	0.173	0.046	0.256	0.166
FTW [86]	0.286	0.568	0.323	0.147	0.331	0.263
Xue2014 [236]	—	0.788	0.387	0.166	0.447	0.328
Liu2012 [124]	0.524	0.732	0.609	0.282	0.537	0.438
Yin2015 [241]	0.376	0.673	0.722	0.323	0.524	0.466
VideoATLAS [8]	0.100	0.644	0.385	0.675	0.451	0.586
P.1203 [157]	0.325	0.817	0.769	0.636	0.637	0.679
Bentaleb2016 [16]	0.741	0.898	0.625	0.682	0.737	0.713
Spiteri2016 [191]	0.612	0.731	**0.809**	0.685	0.709	0.714
SQI [54]	**0.756**	**0.910**	0.673	0.717	0.764	0.745
BSQI	**0.753**	**0.905**	0.794	**0.720**	**0.793**	**0.769**

ing profile, bitrate adaptation algorithms and network conditions. The WaterlooSQoE-III dataset contains 450 streaming videos of 20 source content recorded from a set of streaming experiment. The WaterlooSQoE-IV dataset contains $1,350$ highly-realistic streaming videos constructed from 5 video contents, 2 video encoders, 9 real-world network traces, 5 ABR algorithms, and 3 viewing devices. The streaming videos in different datasets are of diverse characteristics since they are generated from different source videos, encoding profiles, adaptive streaming algorithms, and network conditions. We do not evaluate Xue2014 on the LIVE-NFLX-I dataset because QP of the streaming videos are not publicly available.

Evaluation Criteria

Three criteria are employed for performance evaluation by comparing MOS and objective QoE scores according to the recommendation by the video quality experts group [216]. We adopt Pearson Linear Correlation Coefficient (PLCC) to evaluate the prediction accuracy, Spearman Rank-order Correlation Coefficient (SRCC) and Kendall Rank Correlation Coefficient (KRCC) to assess prediction monotonicity. A better objective QoE model should have higher PLCC, SRCC, and KRCC.

Table 4.3: SRCC between the objective QoE model prediction and MOS on the benchmark datasets.

QoE model	LIVE-NFLX-I	LIVE-NFLX-II	WaterlooSQoE-III	WaterlooSQoE-IV	Average	Weighted Average
Mok2011 [141]	0.335	0.516	0.152	0.056	0.265	0.171
FTW [86]	0.325	0.549	0.184	0.082	0.285	0.197
Xue2014 [236]	—	0.778	0.388	0.219	0.462	0.360
Liu2012 [124]	0.438	0.732	0.598	0.468	0.559	0.539
Yin2015 [241]	0.441	0.686	0.741	0.541	0.602	0.601
VideoATLAS [8]	0.076	0.673	0.469	0.670	0.472	0.603
Spiteri2016 [191]	0.493	0.711	**0.798**	0.662	0.662	0.680
P.1203 [157]	0.415	0.821	0.797	0.668	0.675	0.708
Bentaleb2016 [16]	0.650	0.883	0.718	0.692	0.735	0.730
SQI [54]	0.644	**0.906**	0.690	0.690	0.735	0.732
BSQI	**0.655**	**0.893**	0.776	0.699	**0.756**	**0.747**

Table 4.4: KRCC between the objective QoE model prediction and MOS on the benchmark datasets.

QoE model	LIVE-NFLX-I	LIVE-NFLX-II	WaterlooSQoE-III	WaterlooSQoE-IV	Average	Weighted Average
Mok2011 [141]	0.275	0.425	0.112	0.044	0.214	0.137
FTW [86]	0.251	0.425	0.135	0.072	0.221	0.156
Xue2014 [236]	—	0.582	0.262	0.148	0.148	0.253
Liu2012 [124]	0.324	0.524	0.434	0.319	0.319	0.378
Yin2015 [241]	0.327	0.482	0.543	0.379	0.379	0.427
VideoATLAS [8]	0.050	0.491	0.330	0.480	0.338	0.432
Spiteri2016 [191]	0.376	0.501	0.597	0.461	0.484	0.490
P.1203 [157]	0.300	0.619	**0.604**	0.479	0.501	0.520
Bentaleb2016 [16]	0.479	0.712	0.521	0.495	0.552	0.538
SQI [54]	0.475	**0.735**	0.496	**0.504**	0.553	0.543
BSQI	**0.488**	**0.722**	0.584	**0.575**	**0.572**	**0.558**

4.3.2 Performance

Tables 4.2, 4.3, and 4.4 show the PLCC, SRCC, and KRCC on the benchmark datasets, respectively, from which we have several observations. First, the objective QoE models which employ advanced VQA models as the presentation quality measure generally performs favorably against the conventional bitrate-based QoE models. In particular, Bentaleb2016 significantly outperforms Yin2015, where the only difference between them is the video quality measure. The results provide strong evidence for our use of VMAF as the presentation quality measure. Second, although the learning-based QoE models perform competitively on certain test sets, they fail miserably on the other benchmark datasets. Specifically, the performance degradation of P.1203 and VideoATLAS from one dataset to another can be as large as 0.406 and 0.575, suggesting that the learning-based models exhibit low generalizability to diverse streaming environments. By contrast, BSQI achieves state-of-the-art performance on all three datasets, thanks to the effectiveness of the domain knowledge. Third, the classic QoE models with a fixed parametric form cannot faithfully capture the subjective QoE response on streaming videos with complex distortion pattern, evident by the low prediction accuracy on the WaterlooSQoE-III. In spite of the authors' effort in designing functional forms to conform known HVS properties [54, 86, 236], the QoE functions can vary significantly from exponential and logarithmic functions. On the other hand, BSQI does not assume a particular form of QoE functions and instead maximizes the mathematically well-behaveness. In summary, we believe the performance improvement arises because 1) BSQI is equipped with a HVS inspired VQA measure that generalize well to a variety of video contents, encoders, and viewing devices; 2) the training procedure optimizes the quality prediction accuracy regularized by the prior knowledge of HVS; and 3) the proposed model does not make inaccurate a priori assumptions on the form of QoE functions.

4.3.3 Best-case Validation

Objective QoE model is not only used to evaluate, but also to optimize a variety of ABR algorithms and systems. A good rule of thumb is that an optimized system is only as good as the optimization criterion used to design it [222]. Conversely, the performance

of an objective QoE model can be assessed via synthesizing optimal streaming videos with respect to an objective QoE model followed by visual inspection of the generated stimulus [224, 228]. Specifically, given a set of encoded and segmented videos and a realistic network trace, we can generate an optimal streaming video in terms of each objective QoE model. Subjective evaluation of the synthesized stimuli provides a best-case validation of the underlining objective QoE models. A good objective QoE model should produce perceptually better streaming videos comparing to the other schemes.

We select 12 high-quality videos of diverse complexity to constitute the test sample set. All videos have the length of 30 seconds. Using the source sequences, each video is encoded with two types of encoding strategy including the traditional fixed bitrate encoding [147] and the state-of-the-art per-title encoding suggested by Netflix [36]. In the fixed bitrate encoding, each video is encoded into 10 pre-defined representations. While in the per-title encoding, the number of compressed versions and the choice of encoding configuration depend on the characteristics of source videos. Specifically, we select the bitrate-resolution pair such that i) At a given bitrate, the produced encode should have as high quality as possible, and ii) The perceptual difference between two adjacent bitrates should fall just below one just-noticeable different (the difference in VMAF $\approx$ 10). We segment the test sequences the encoded videos with GPAC's MP4Box [112] with a segment length of 2 seconds for the following reasons. First, 2-second segments are widely used in the development of ABR algorithms [131, 241] and deployment of real-world streaming applications [57, 113], primarily due to its flexibility for stream adaption to bandwidth changes and for its strong impact on reducing the latency of video delivery. Second, it allows us to derive test videos in an efficient way such that they cover a diverse adaptation patterns in a limited time. We randomly selected 12 network traces from both the 3G High Speed Downlink Packet Access (HSDPA) dataset [171] and the 4G Belgium dataset [211] to cover a diversity of network conditions. The HSDPA dataset contains network traces that have significant variability and low average bandwidth, making it a strong test for the QoE models in the complicated scenarios. Traces in the Belgium dataset exhibit higher average throughput and lower standard deviation, which closely represents the realistic streaming environment. We compare BSQI with three objective QoE models that have guided the development of ABR algorithms, including Yin2015, Spiteri2016, and Bentaleb2016. We

present results for the offline optimal scheme [131, 191], which is computed using dynamic programming with complete future throughput information. The dynamic programming-based method generates globally optimal streaming videos for the considered QoE models, completely eliminating the influence of inaccurate throughput estimation. For each source video, we randomly select a network trace and optimize the streaming videos with respect to the four objective QoE models. In the end, we obtain a total of 192 streaming videos generated from 24 (source videos, network traces) pairs $\times 2$ encoding strategies $\times$ 4 ABR algorithms. An online demonstration of the experiment is available at [44].

The subjective user study adopts the pairwise comparison methodology in which a pair of streaming videos generated from the same video contents and network traces are presented to human viewers. The subjective experiment is setup as a normal indoor home settings with an ordinary illumination level, with no reflecting ceiling walls and floors. A customized interface is created to render a pair of 1920×1080 videos side-by-side on a 27 inch Ultra High Resolution Television (UHDTV). The display is calibrated in accordance with the recommendations of International Telecommunication Unit-Recommendation (ITU-R) BT. 500 [100]. For each video pair, the subjects are forced to choose which one has a better perceptual quality. A total of 15 naïve subjects, including 7 males and 8 aged between 18 and 55, participate the subjective experiment. Visual acuity and color vision are confirmed from each subject before the subjective test. A training session is performed, during which, 3 video pairs that are different from the videos in the testing set are presented to the subjects. We used the same methods to generate the videos used in the training and testing sessions. Therefore, subjects knew what distortion types would be expected before the test session, and thus learning effects are kept minimal in the subjective experiment. For each subject, the whole study takes one hour, which is divided into two sessions with a 5-minute break in-between.

The results of the subjective experiment can be summarized as a 4×4 matrix $\mathbf{R}$, where $r_{i,j}$ represents the probability of QoE model i better than QoE model j. Figure 4.1 shows the result matrix $\mathbf{R}$, where the higher value of an entry (warmer color), the stronger the row model against the column model. It is obvious that BSQI performs favorably to the competing models. We further aggregate the pairwise comparison results into a global ranking via the maximum likelihood method for multiple options [127, 163, 210]. Let

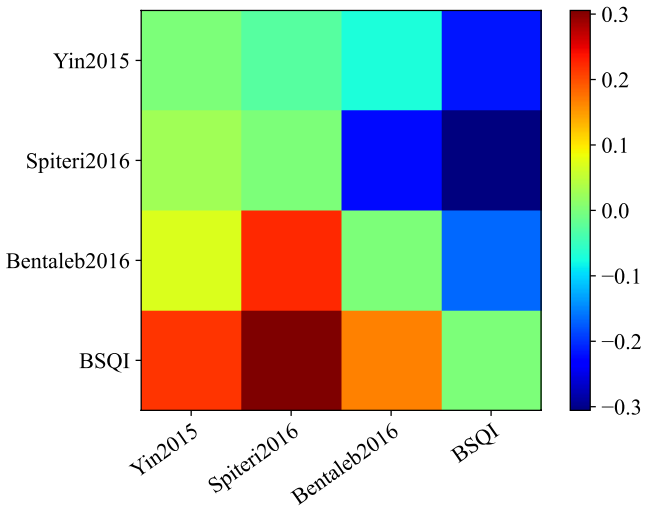


Figure 4.1: Pairwise comparison matrix $\mathbf{R}$. Each entry indicates the subjective preference of the row model against the column model. $\mathbf{R} - \mathbf{R}^T$ are drawn here for better visibility.

$\boldsymbol{\mu} = [\mu_1, \mu_2, \mu_3, \mu_4] \in \mathbb{R}^4$ be the global ranking score vector, we maximize the log-likelihood of $\boldsymbol{\mu}$

$$\begin{aligned} \arg\max_{\boldsymbol{\mu}} \quad & \sum_{i,j} r_{i,j} \log(\Phi(\mu_i - \mu_j)) \\ \text{subject to} \quad & \sum_i \mu_i = 0, \end{aligned}$$

where $\Phi(\cdot)$ is the standard normal cumulative distribution function. The constraint $\sum_i \mu_i = 0$ is introduced to resolve the translation ambiguity. The optimization problem is convex and enjoys efficient solvers. A larger μ_i means the optimal streaming video in terms of the i-th model is perceptually better than the optimal samples generated by other QoE models in general. Figure 4.2 shows the experimental results. It can be seen that BSQI significantly outperforms the standard QoE models. By taking a closer look at the trace-specific experiment results, we find that BSQI consistently delivers the best performance across different experiment setup, although the improvement is less significant on the 4G dataset. We notice that the small performance gain in the 4G experiment arises from the abundant bandwidth resource, especially when the highest resolution of streaming videos is restricted by the pairwise comparison experiment. Note that the maximum width/height

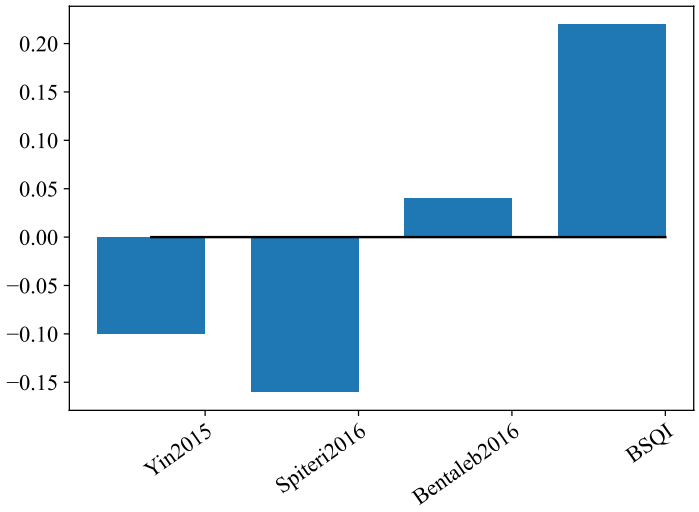


Figure 4.2: Global ranking results of the four QoE models.

of one test stimulus can be at most half of the width/height of the display. We expect a more significant improvement in the realistic setting where 4K, high dynamic range, and high framerate video contents are involved. The results have significant implications on the development of ABR algorithms. Specifically, state-of-the-art ABR algorithms have achieved a performance plateau levels and significant improvement has become difficult to attain. However, the enormous difference in perceptual relevance between the bitrate-based QoE model and BSQI suggests that further improvement is attainable simply by adopting perceptually motivated optimization criterion.

4.3.4 Statistical Significance Test

To ascertain that the improvement of the proposed model is statistically significant, we carry out a statistical significance analysis by following the approach introduced in [186]. First, a nonlinear regression function is applied to map the objective quality scores to predict the subjective scores. We observe that the prediction residuals all have zero-mean, and thus the model with lower variance is generally considered better than the

Table 4.5: Statistical significance matrix based on F-statistics on the combination of WaterlooSQoE-III, WaterlooSQoE-IV LIVE-NFLX-I, and LIVE-NFLX-II datasets. A symbol "1" means that the performance of the row model is statistically better than that of the column model, a symbol "0" means that the row model is statistically worse, and a symbol "-" means that the row and column models are statistically indistinguishable.

	FTW	Mok2011	Liu2012	Yin2015	VideoATLAS	Spiteri2016	P.1203	Bentaleb2016	SQI	BSQI
FTW	-	-	0	0	0	0	0	0	0	0
Mok2011	-	-	0	0	0	0	0	0	0	0
Liu2012	1	1	-	-	0	0	0	0	0	0
Yin2015	1	1	-	-	0	0	0	0	0	0
VideoATLAS	1	1	1	1	-	0	0	0	0	0
Spiteri2016	1	1	1	1	1	-	-	0	0	0
P.1203	1	1	1	1	1	-	-	0	0	0
Bentaleb2016	1	1	1	1	1	1	1	-	0	0
SQI	1	1	1	1	1	1	1	1	-	0
BSQI	1	1	1	1	1	1	1	1	1	-

one with higher variance. We conduct a hypothesis testing using F-statistics. Since the number of samples exceeds 50, the Gaussian assumption of the residuals approximately hold based on the central limit theorem [19]. The test statistic is the ratio of variances. The null hypothesis is that the prediction residuals from one quality model come from the same distribution and are statistically indistinguishable (with 95% confidence) from the residuals from another model. After comparing every possible pairs of objective models, the results are summarized in Table 4.5, where a symbol '1' means the row model performs significantly better than the column model, a symbol '0' means the opposite, and a symbol '-' indicates that the row and column models are statistically indistinguishable. It can be observed that the proposed model is statistically better than all other methods on the streaming video QoE database.

4.3.5 Ablation Experiment

We conduct a series of ablation experiments to single out the core contributors of BSQI. We first take bitrate [124, 241], logarithmic bitrate [191], and QP [236] as the presentation video quality measure as opposed to VMAF and then train the QoE model with the proposed

Table 4.6: PLCC between the variants of BSQI prediction and MOS on the benchmark datasets.

QoE model	LIVE-NFLX-I	LIVE-NFLX-II	WaterlooSQoE-III	WaterlooSQoE-IV	Average	Weighted Average
BSQI with bitrate	0.622	0.722	0.670	0.618	0.658	0.647
BSQI with log bitrate	0.686	0.715	0.787	**0.738**	0.732	0.741
BSQI with QP	—	0.776	0.416	0.184	0.459	0.343
BSQI with VMAF	**0.753**	**0.905**	**0.794**	0.720	**0.793**	**0.769**

Table 4.7: PLCC between the variants of BSQI prediction and MOS on the benchmark datasets.

Constraint #	LIVE-NFLX-I	LIVE-NFLX-II	WaterlooSQoE-III	WaterlooSQoE-IV	Average	Weighted Average
None	0.731	0.903	0.663	0.681	0.745	0.720
(4.2)	0.743	0.902	0.788	0.718	0.788	0.766
(4.2)(4.3)	0.748	0.904	0.780	0.719	0.788	0.765
(4.2)(4.3)(4.4)	0.748	0.896	**0.800**	0.713	0.788	0.764
(4.2)(4.3)(4.4)(4.5)	**0.753**	**0.905**	0.794	**0.720**	**0.793**	**0.769**
(4.2)(4.3)(4.4)(4.5)(4.6)	**0.753**	**0.905**	0.794	**0.720**	**0.793**	**0.769**
(4.2)(4.3)(4.4)(4.5)(4.6)(4.7)	**0.753**	**0.905**	0.793	**0.720**	**0.793**	**0.769**
(4.2)(4.3)(4.4)(4.5)(4.6)(4.7)(4.8)	**0.753**	**0.905**	0.794	**0.720**	**0.793**	**0.769**
(4.2)	0.744	0.902	0.788	0.718	0.788	0.766
(4.3)	0.743	0.906	0.758	0.717	0.781	0.760
(4.4)	0.743	0.895	0.798	0.713	0.788	0.764
(4.5)	0.753	0.902	0.787	0.717	0.790	0.766
(4.6)	0.745	0.884	0.770	0.691	0.773	0.744
(4.7)	0.745	0.884	0.770	0.692	0.773	0.744
(4.8)	0.745	0.884	0.770	0.691	0.773	0.744
(4.9)	0.746	0.884	0.770	0.692	0.773	0.744
BSQI	**0.753**	**0.905**	0.794	**0.720**	**0.793**	**0.769**

optimization framework. In order to map the range of presentation video quality measure into the same perceptual scale [0, 100], we apply a linear transform to the alternative measures before the training stage. From Table 4.6, we observe that BSQI achieves the best performance with the state-of-the-art video quality measure VMAF.

Next, we analyze the impact of the knowledge-imposed constraints on the quality prediction performance. We start from a baseline model by solving the problem in (4.14) with no constraints and gradually increase the number of constraints. We then investigate the validity of each observation by imposing only one constraint in a variant model. The results are listed in Table 4.7, from which the key observations are as follows. First, the performance of BSQI generally improves with respect to the number of imposed con-

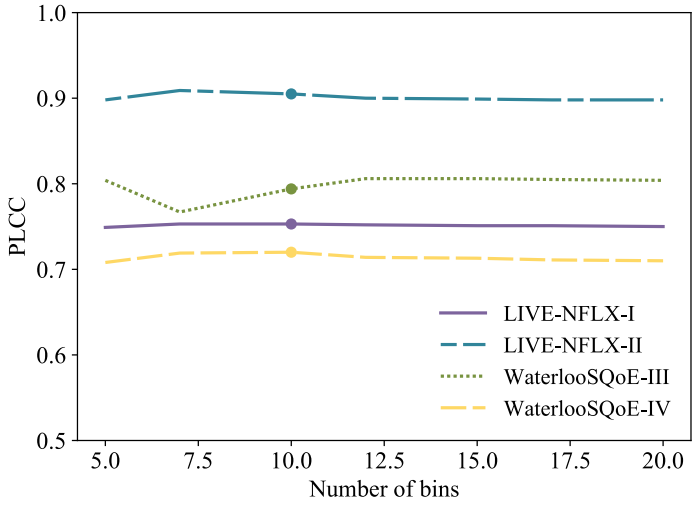


Figure 4.3: Performance of BSQI with different number of bins.

straints, advocating the effectiveness of prior knowledge in regularizing the objective QoE functions. Second, while some of the constraints do not improve the performance of BSQI by themselves, the joint model achieves state-of-the-art performance. This suggests that the constraints may be complement to each other. Third, the constraint (4.4) has drastically different impacts on the LIVE-NFLX-II dataset and the WaterlooSQoE-III dataset, suggesting that the validity of the constraint may be influenced by other factors. A careful investigation may further improve the performance of the proposed QoE model.

4.3.6 Impact of Step Sizes

In previous experiments, we set the bin sizes of presentation video quality and rebuffering duration to 10 and 1, respectively. To investigate the impact of step sizes, we train several variants of BSQI, where the number of bins ranges from 5 to 20. We show the experimental results in Figure 4.3. Theoretically speaking, the performance of BSQI should increase monotonically with respect to the precision of feature representations. However, the observation does not echo our expectation, which may be a consequence of insufficient

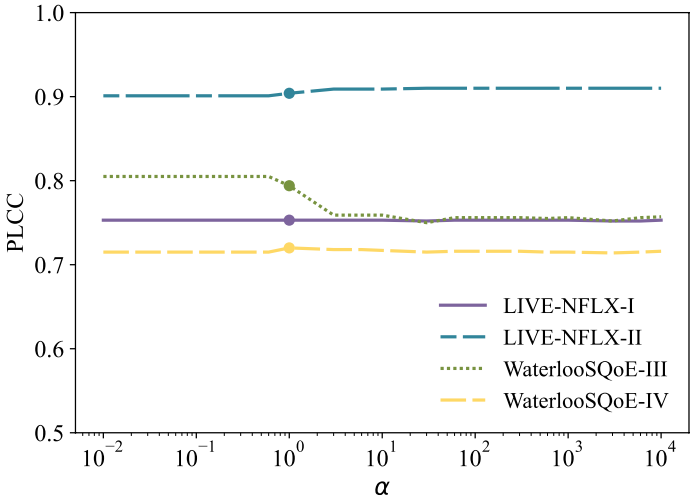


Figure 4.4: Performance of BSQI with different α.

training data and intrinsic noise in the subjective opinion scores. Nevertheless, BSQI is generally very robust to a broad range of bin sizes.

4.3.7 Impact of α

The parameter α in BSQI determines the tradeoff between fidelity and flatness of the QoE functions. Although the optimal parameter is obtained from cross-validation in previous experiments, we also perform an experiment to investigate the impact of α. Specifically, we train several versions of BSQI, where α ranges from 0.01 to $10,000$. The results are shown in Figure 4.4, from which we can observe that the performance of BSQI is generally insensitive to α.

4.4 Summary

In this chapter, we propose a novel objective QoE model for adaptive streaming videos, namely BSQI, by regularizing a non-parametric model with known HVS properties. BSQI

outperforms the existing objective QoE models by a sizable margin over a wide range of video contents, encoding configurations, network conditions, and viewing devices, which we believe arises from a perceptually motivated video quality representation, a knowledge constrained optimization framework, and a non-parametric model of QoE functions.

Chapter 5

Delving into Connection-Level Throughput Prediction using Meta Learning

In this chapter, we first discuss the fundamental limitations of existing throughput predictors, which motivate us to develop a meta learning-based throughput distribution model. In contrast to the traditional approach that maximizes the marginal likelihood function, the proposed Meta Learning-based Throughput Predictor (MetaTP) are dedicated to approximate connection-level network dynamics. By making effective use of the abundant network traces, the data-driven approach do not need to unrealistic assumptions, thereby outperforming the state-of-the-art scheme with a sizable margin. By combining a generic prior model and a connection-level likelihood function, we show that MetaTP can quickly adapt to a broad range of network environment at very little cost.

5.1 Motivation

As discussed in Section 2.2, existing approaches build a generic throughput prediction model for all network flows. The "one-size-fits-all" scheme achieves, for most cases, promis-

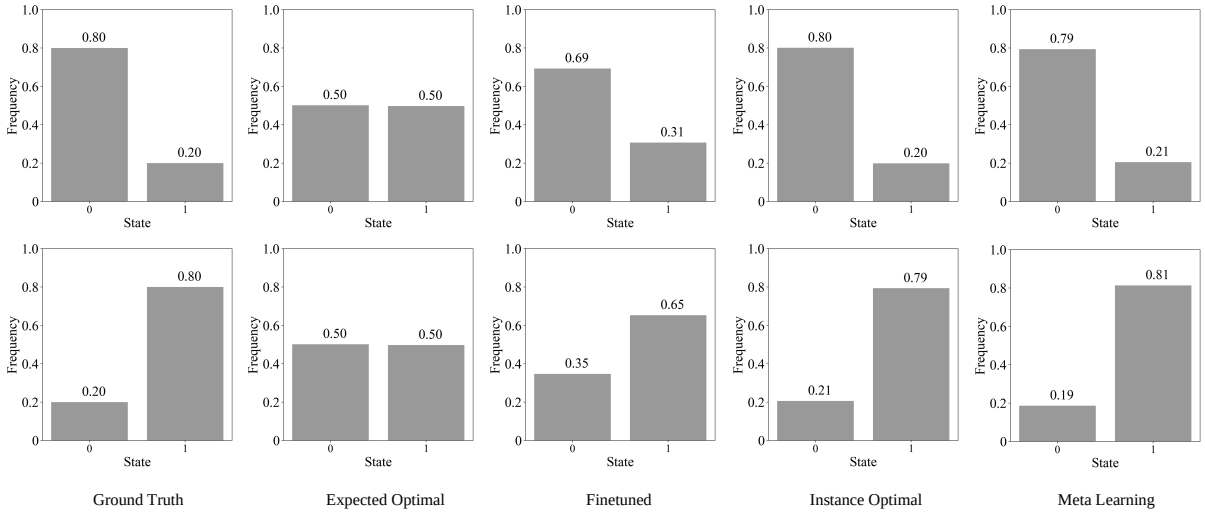


Figure 5.1: Approximating of two instance-level Bernoulli distributions (first column) using the standard supervised learning (second column), fine tuning with pretraining (third column), fine tuning with random initialization (fourth column), and meta learning (fifth column). Row 1 and row 2 illustrate the conditional distribution for the first class and the second class, respectively. Fine tuning from marginal distribution, random initialization, and meta learning-based prior are performed for five gradient steps.

ing prediction accuracy on a test set with very similar characteristics to the training set. However, these models often struggle to deliver equally competitive performance in real-world network environment that is inevitably more complex, especially when the connection-level network dynamics deviate significantly from the marginal throughput distribution.

There are generally three approaches to solve the problem. A straight-forward solution to the distribution mismatch problem is to directly learn a throughput prediction model for each client. However, there are often insufficient connection-level throughput observations to obtain reasonable throughput prediction models [171, 237]. The second approach is to adapt the generic throughput model to viewer conditional throughput distributions with limited throughput observations from each source component. Though, the marginal throughput distribution may not provide meaningful information to its sub-population.

The third approach takes connection-level features such as IP address as the input of throughput prediction models. Unfortunately, there is generally no regularities between these features and the network characteristics. Even though one may cluster sessions with similar features, the identification of useful handcrafted features and clustering relies heavily on manual parameter tuning, which can backfire when their design assumptions are violated.

Figure 5.1 provides a motivating example, where the goal is to approximate two source Bernoulli distributions from their samples. The class prior model, which encodes the likelihood a sample comes from one of these sub-populations, follows a uniform distribution. The traditional approach learns a marginal distribution of the two classes with aggregated training samples, neglecting the heterogeneity of instance conditional distributions. It can be observed that the standard supervised learning model that digests all training data at once converges to a very poor solution (in fact not better than random guess). Furthermore, using the marginal state distribution as the model initialization may introduce inductive bias in the fine-tuning of the class conditional distributions [19]. As a result, it takes even more training data/computational resources to accurately approximate the conditional distributions than simply starting with non-informative prior.

Figure 5.2 shows a similar example in the context of sequential data prediction. In this problem, we are given limited training data uniformly sampled from two distinct Markov processes, whose transition probabilities are given in the first column of Figure 5.2. Concretely, we can assume the two Markov processes as two connection-level throughput distributions. The objective of the problem is to produce accurate throughput prediction for both of the two clients. Traditional learning scheme generates a generic throughput predictor by applying the maximum likelihood estimator on all available training data, resulting in a Markov process as shown in the second column of Figure 5.2. Unfortunately, the expected optimal throughput predictor has almost no correlation with the two client-specific network dynamics. Even if one adapts the pretrained model to fit client conditional distributions with additional connection-level observations, the convergence rate is significantly reduced by the strongly biased initialization. On the other hand, learning the client conditional distributions from scratch may produce models with excessive variance due to the limited training data from each of the two sub-populations. This dilemma is a

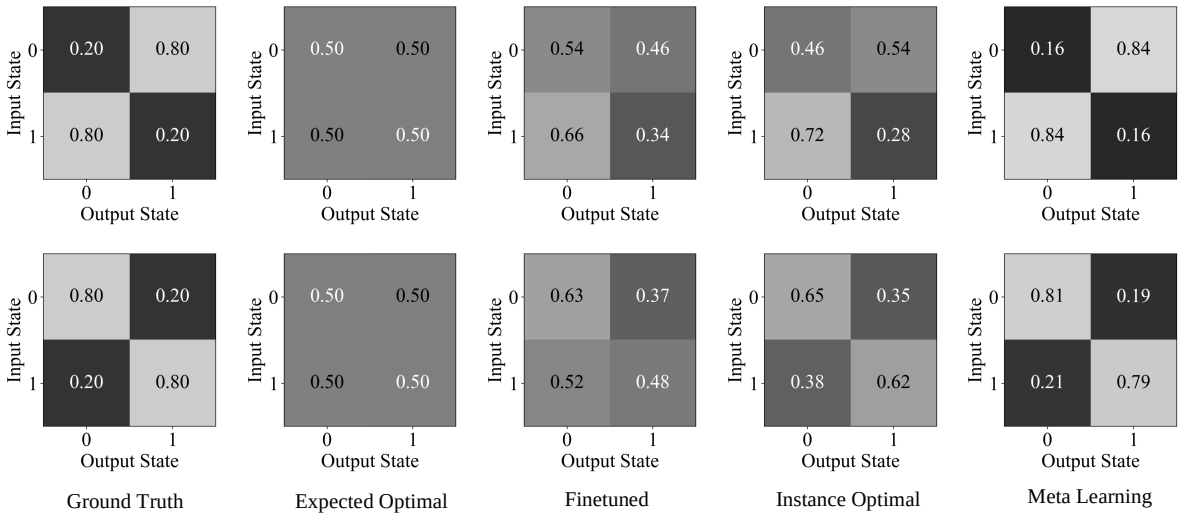


Figure 5.2: Approximating two sub-population Markov processes (first column) using the standard supervised learning (second column), fine tuning with pretraining (third column), fine tuning with random initialization (fourth column), and meta learning (fifth column). Row 1 and row 2 illustrate the conditional distribution for the first class and the second class, respectively. Fine tuning from marginal distribution, random initialization, and meta learning-based prior are performed for five gradient steps.

manifestation of the bias-variance tradeoff [19].

It turns out that the source component shift problem [197] is not limited to the illustrative examples. Rather, the "symptom" commonly arises in practical throughput prediction systems. To demonstrate the universality of the problem, we present a similar analysis on realistic throughput distributions. We consider throughput distributions from two IP addresses where there exist abundant throughput data. To simplify the analysis, we assume the two throughput distributions to be Markov chains, each with three states[1]. Instead of knowing the transition matrices a priori, we approximate the two ground-truth distributions using maximum likelihood method on all available class-specific training data. In the rest of the experiment, we extract a subset of training data to simulate the limited sample

[1]Note that all existing throughput prediction models are built upon the Markov assumption.

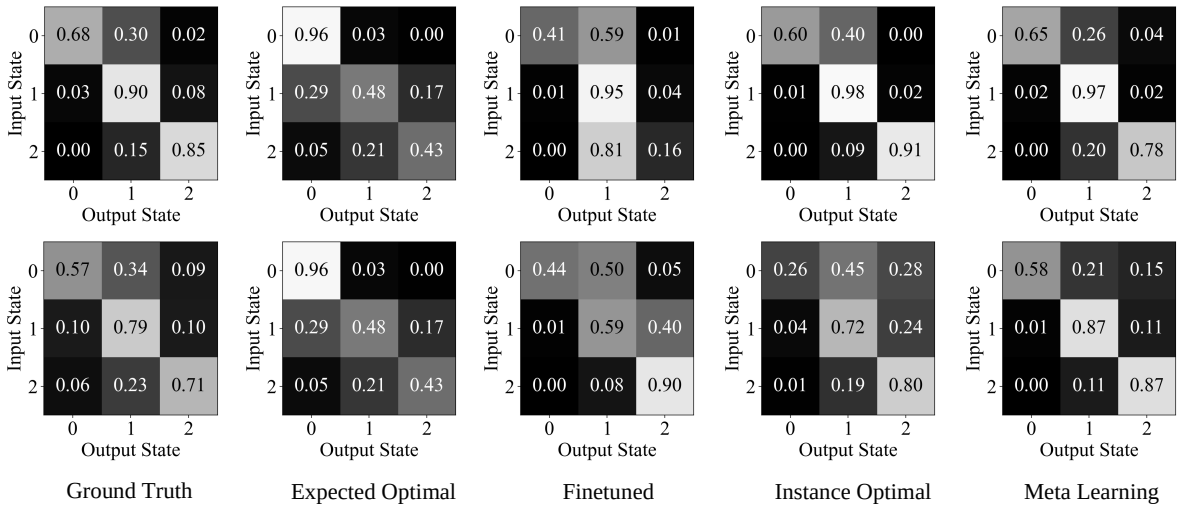


Figure 5.3: Approximating two sub-population realistic throughput distributions (first column) using the standard supervised learning (second column), fine tuning with pretraining (third column), fine tuning with random initialization (fourth column), and meta learning (fifth column). Row 1 and row 2 illustrate the conditional distribution for the first class and the second class, respectively. Fine tuning from marginal distribution, random initialization, and meta learning-based prior are performed for five gradient steps.

scenario in practice. We then apply traditional supervised learning, fine tune the learnt model with class conditional training data, and train two throughput prediction models from random initialization. The experimental results are given in Figure 5.3, where we can observe that practical throughput predictors also suffer from the distribution mismatch problem.

Following the Bayesian theory, we formulate the throughput distribution modeling as a few-shot learning problem. In contrast to the traditional approaches that either employ hand-crafted rules to cluster individual sessions or blindly ingest the entire training set, we propose a learning algorithm, namely MetaTP that can automatically identify the common structure shared across different network environment. During the pretraining stage, the parameters of the prior throughput distribution are explicitly optimized to be

easy to fine-tune. From a dynamical systems standpoint, our learning process can be viewed as maximizing the sensitivity of the loss functions of new clients with respect to the parameters [61]. Then, the posterior throughput distribution can be obtained by combining the data-driven prior model and the likelihood function embedded in a small amount of session-level throughput observations. Thanks to the effectiveness of the proposed learning scheme, MetaTP can produce accurate prediction in the three motivating examples shown in Figure 5.1, Figure 5.2, and Figure 5.3. In this study, we will present a roadmap to develop a practical connection-aware throughput predictor.

5.2 Meta Learning-based Throughput Prediction

We aim to develop client adaptive throughput prediction models with a limited number of client-specific throughput observations. The proposed learning framework is illustrated in Figure 5.4. First, a dedicated edge server trains a prior throughput model using meta learning on a throughput dataset. The dataset consists of throughput traces from a great number of clients, whereas there are only limited training samples (usually up to 100 samples) for each client. The server sends the pretrained model to all ABR players that request videos from Content Delivery Network (CDN). Each client fine tunes the generic prior model using limited number of local network flow observations, resulting in a posterior throughput prediction model. Finally, the clients may periodically send their local throughput dataset to the meta learner to prevent the data staleness problem. Apparently, the key is to obtain a meaningful prior throughput model. In this section, we will define the problem setup and present the meta learning algorithm.

5.2.1 Problem Setup

Our objective is to train a model that can quickly adapt to new environment using only a few datapoints and training iterations. Following the terminology in meta learning [61], we refer to the approximation of each client-level throughput prediction model as a task. To accomplish this, we can exploit the common structures shared across a set of relevant tasks. In effect, the meta-learning problem treats entire tasks as training examples.

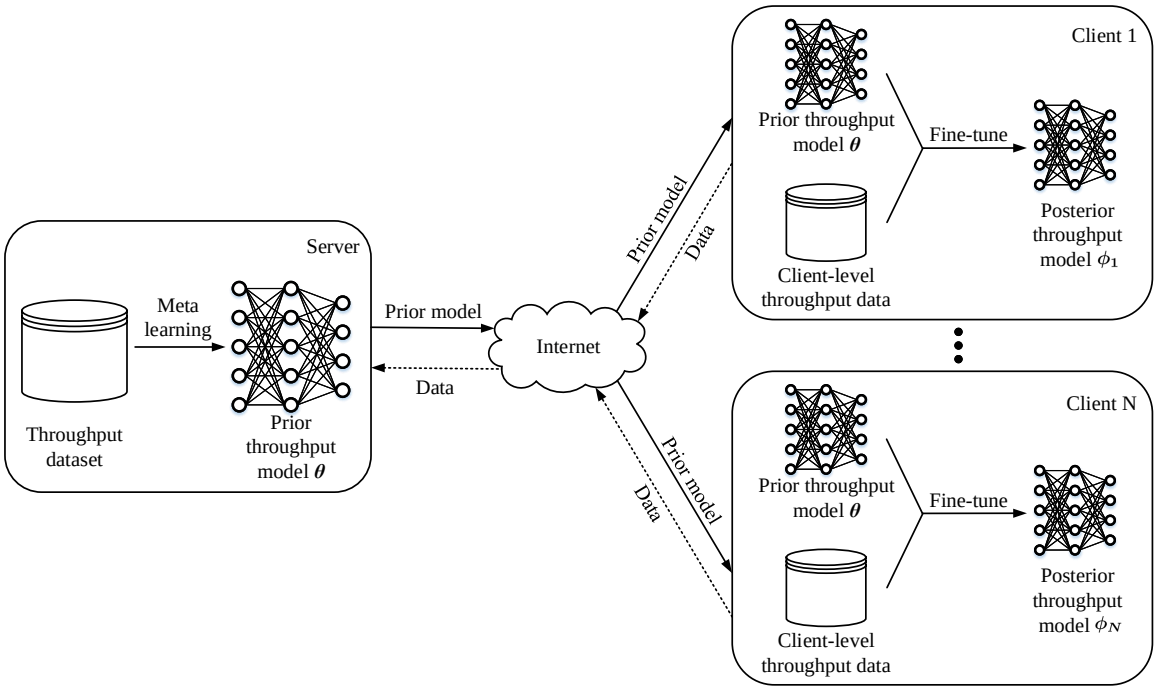


Figure 5.4: The proposed meta learning-based throughput prediction framework.

Formally, given a dataset $\mathcal{D}_{\mathbf{c}}$ that defines a distribution over a family of tasks $p(\mathcal{T})$, where each task $\mathcal{T} = \{L,\ p(\mathbf{c})\}$ consists of a loss function L and a connection-specific throughput distribution $p(\mathbf{c})$, the goal of meta learning-based throughput prediction model is to learn a generic throughput model $p(\mathbf{c};\ \boldsymbol{\theta})$ that can quickly adapt to a new task in the distribution $p(\mathcal{T})$. The loss function L provides task-specific feedback to guide the learning of throughput prediction models. Specifically, the meta learning stage can be decomposed into the following steps. First, a task $\mathcal{T}_i$ is drawn from $p(\mathcal{T})$. Second, a base model is then learnt from N samples from $p_{\mathcal{T}_i}(\mathbf{c})$ and feedback from the corresponding loss $L_{\mathcal{T}_i}$. Third, the learning agent re-draws M samples from $p_{\mathcal{T}_i}(\mathbf{c})$. Last, the base model $p(\mathbf{c};\ \boldsymbol{\theta})$ is improved by considering how the test error on new datapoints in $p_{\mathcal{T}_i}(\mathbf{c})$ changes with respect to the parameters. During the meta testing (also referred to as fast adaptation) phase, new tasks are sampled from $p(\mathcal{T})$, and the prior model $p(\cdot;\ \boldsymbol{\theta})$ is fine tuned with N samples from the throughput distribution $p_{\mathcal{T}_i}(\mathbf{c})$ to obtain the connection-level model $p(\mathbf{c};$

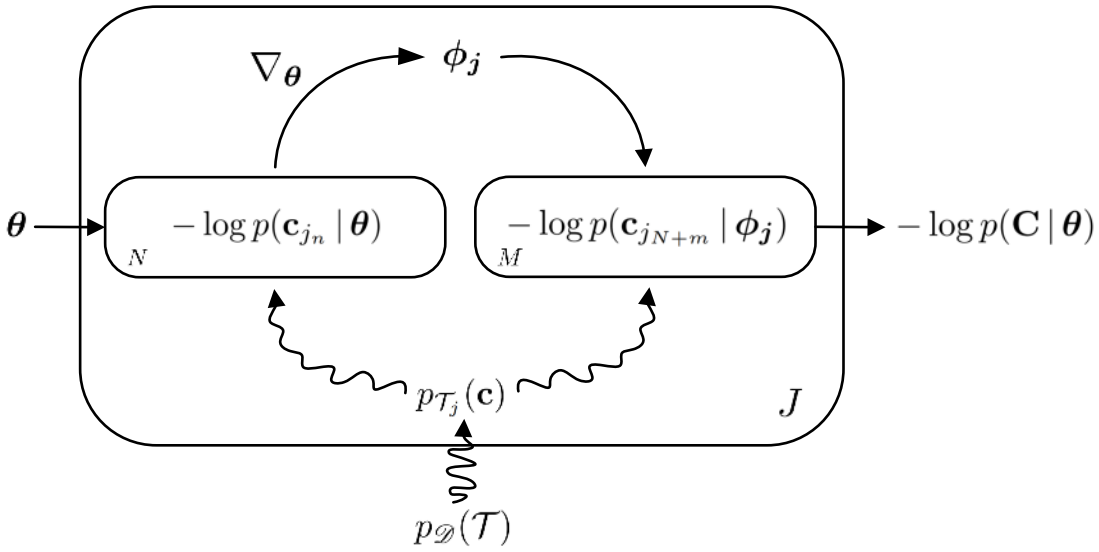


Figure 5.5: The computational graph of the meta learning algorithm. Straight arrows, crooked arrows, and plates denote deterministic computations, sampling operations, and repeated computations, respectively.

$\boldsymbol{\phi}_i$). The computational graphic is illustrated in Figure 5.5.

5.2.2 Learning Algorithm

To instantiate the proposed throughput prediction framework, we adopt a state-of-the-art meta learning algorithm namely Model Agnostic Meta Learning (MAML) [61]. Thanks to the scalable gradient descent procedure, MAML is naturally applicable to complex function approximators. The detailed learning algorithm is explained below.

The learning algorithm starts with a randomly initialized throughput prediction model $p(c^{t+1}|\mathbf{c}^{1:t}; \boldsymbol{\theta})$ for all $t \leq K - 1$, where c^{t+1}, $\mathbf{c}^{1:t}$, and K denote the throughput value of the immediately next instance, the throughput observations from all previous instances, and the total number of throughput observations in a trace, respectively. As discussed in

Section 2.2, building a conditional sequence model for all t is equivalent to estimating a generative throughput model $p(\mathbf{c};\boldsymbol{\theta})$. For simplicity of notation, we consider a generative throughput distribution in the subsequent analysis. To obtain a task-specific throughput prediction model $p(\mathbf{c};\boldsymbol{\phi}_i)$, we update the base model using one gradient descent step

$$\boldsymbol{\phi}_i = \boldsymbol{\theta} - \alpha_s \nabla_{\boldsymbol{\theta}} L_{\mathcal{T}_i}\Big(p(\mathbf{c}_{i_1},...,\mathbf{c}_{i_N};\boldsymbol{\theta})\Big), \tag{5.1}$$

where $\mathbf{c}_{i_1},...,\mathbf{c}_{i_N}$ are samples from the task-specific throughput distribution and α_s is the step size in meta learning. The base model parameters are trained by optimizing for the performance of $p(\mathbf{c};\boldsymbol{\phi}_i)$ with respect to $\boldsymbol{\theta}$ across all tasks sampled from $p(\mathcal{T})$. Specifically, the objective of meta learning is

$$\begin{aligned}&\min_{\boldsymbol{\theta}} \sum_{\mathcal{T}_i \sim p(\mathcal{T})} L_{\mathcal{T}_i}\Big(p(\mathbf{c}_{i_{N+1}},...,\mathbf{c}_{i_{N+M}};\boldsymbol{\phi}_i)\Big) = \\ &\min_{\boldsymbol{\theta}} \sum_{\mathcal{T}_i \sim p(\mathcal{T})} L_{\mathcal{T}_i}\Bigg(p\Big(\mathbf{c}_{i_{N+1}},...,\mathbf{c}_{i_{N+M}};\boldsymbol{\theta} - \alpha_s \nabla_{\boldsymbol{\theta}} L_{\mathcal{T}_i}\Big(p(\mathbf{c}_{i_1},...,\mathbf{c}_{i_N};\boldsymbol{\theta})\Big)\Big)\Bigg).\end{aligned} \tag{5.2}$$

In contrast to the traditional supervised learning where the loss is computed using the up-be-optimized parameters $\boldsymbol{\theta}$, the loss in meta learning is computed using the updated parameters $\boldsymbol{\phi}_i$. MAML effectively optimizes the model parameters such that one or a small number of gradient steps on a new task will produce maximally effective behavior on that task [61]. The optimization problem in (5.2) can also be solved by stochastic gradient descent algorithm, such that the model parameters $\boldsymbol{\theta}$ are updated as

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \beta_s \sum_{\mathcal{T}_i \sim p(\mathcal{T})} L_{\mathcal{T}_i}\Big(\mathbf{c}_{i_{N+1}},...,\mathbf{c}_{i_{N+M}};\boldsymbol{\phi}_i)\Big), \tag{5.3}$$

where β_s denotes the meta-optimization step size.

The computation of (5.3) involves a gradient through a gradient, which requires an additional back propagation step through $p(\mathbf{c};\boldsymbol{\phi}_i)$ to compute Hessian-vector products. This operation is supported by off-the-shelf deep learning libraries such as Tensorflow [1], PyTorch [160], and JAX [23]. It has been found that one can approximate the update rule in (5.3) by dropping this backward pass, and still achieve reasonable performance [61, 151].

Two common loss functions used for throughput predictors are mean absolute error and cross-entropy, which we will detail below. Though the learning framework can be

readily extended to other loss functions. By using the mean absolute error as the loss function [135], one implicit assumes the observation conditional connection-level throughput distribution $p(c^{t+1}|\mathbf{c}^{1:t})$ follows an constant variance Laplacian distribution, whose mean can be approximated by a function $f(\mathbf{c}^{1:t}; \boldsymbol{\phi}_i)$. With this simplification, the original density estimation problem is transformed into a regression problem. The loss function takes the form

$$L_{\mathcal{T}_i}(\boldsymbol{\phi}_i) = \sum_{\mathbf{c}_j^{1:t}, c_j^{t+1} \sim \mathcal{T}_i} ||f(\mathbf{c}_j^{1:t}; \boldsymbol{\phi}_i) - c_j^{t+1}||_1, \tag{5.4}$$

for all $t \in [2, K-1]$. The sequential throughput observations $\mathbf{c}_j^{1:t}$ and c_j^{t+1} can be extracted from the j-th throughput trace. Alternatively, some throughput prediction models consider a more general task-specific distribution by imposing the cross-entropy loss [237], which takes the form

$$L_{\mathcal{T}_i}(\boldsymbol{\phi}_i) = -\sum_{\mathbf{c}_j^{1:t}, c_j^{t+1} \sim \mathcal{T}_i} c_j^{t+1} \log(p(\mathbf{c}_j^{1:t}; \boldsymbol{\phi}_i)), \tag{5.5}$$

where the throughput observation c_j^t at each time instance is quantized into several bins. In this study, we utilize the mean absolute error loss in the training of MetaTP for simplicity, because the calibration of bin size in the cross-entropy approach involves cumbersome manual parameter tuning. However, the general methodology is still applicable to optimize any differentiable loss function.

5.2.3 Bayesian Interpretation

Following the discussion in Section 2.2 and a theoretic analysis in [73], we present a Bayesian interpretation of the meta learning-based throughput prediction model. In MAML, each task-specific parameter $\boldsymbol{\phi}_i$ is distinct from but should influence the estimation of the parameters $\{\boldsymbol{\phi}_i' | i' \neq i\}$ from other tasks. This intuition can be governed by a meta-level parameter $\boldsymbol{\theta}$ on which each task-specific parameter is statistically dependent. In effect, the task-specific parameters are mutually independent conditioned on the meta-level parameter $\boldsymbol{\theta}$, which encodes the prior knowledge of the common structures across all network environments. Given some data in each task, we can apply the maximum likelihood method to estimate the meta-level parameter $\boldsymbol{\theta}$ by integrating out the task-specific parameters $\boldsymbol{\phi}_i$.

Mathematically, let $\mathbf{C}$ collectively denote all training data, the marginal likelihood of the observed data is given by

$$p(\mathbf{C}|\boldsymbol{\theta}) = \prod_i \Big(\int p(\mathbf{c}_{i_{N+1}}, ..., \mathbf{c}_{i_{N+M}}|\boldsymbol{\phi}_i) p(\boldsymbol{\phi}_i|\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\phi}_i \Big). \tag{5.6}$$

Maximizing (5.6) as a function of $\boldsymbol{\theta}$ gives a point estimate for $\boldsymbol{\theta}$, an instance of a method known as empirical Bayes [18, 73]. In practice, the marginalization over task-specific parameters $\boldsymbol{\phi}_i$ is computationally intractable. To overcome this issue, MAML makes use of a point estimation $\boldsymbol{\phi}_i$ for each task instead of performing the integration over $\boldsymbol{\phi}_i$. This approximation can be mathematically expressed by

$$-\log p(\mathbf{C}|\boldsymbol{\theta}) \approx -\sum_i \Big(p(\mathbf{c}_{i_{N+1}}, ..., \mathbf{c}_{i_{N+M}}|\hat{\boldsymbol{\phi}}_i) \Big), \tag{5.7}$$

where we have recovered the objective function in (5.2). Therefore, the meta learning process essentially approximates the meta-level parameters $\boldsymbol{\theta}$ using the empirical Bayes method.

During the fast adaptation stage, our objective is to find the optimal task-specific parameter $\boldsymbol{\phi}_i$. The maximum a posteriori estimate of $\boldsymbol{\phi}_i$ corresponds to the global mode of the posterior distribution $p(\boldsymbol{\phi}_i|\mathbf{c}_{i_{N+1}}, ..., \mathbf{c}_{i_{N+M}}, \boldsymbol{\theta}) \propto p(\mathbf{c}_{i_{N+1}}, ..., \mathbf{c}_{i_{N+M}}|\boldsymbol{\phi}_i) p(\boldsymbol{\phi}_i|\boldsymbol{\theta})$. The likelihood function $p(\mathbf{c}_{i_{N+1}}, ..., \mathbf{c}_{i_{N+M}}|\boldsymbol{\phi}_i)$ measures how well a model fits the new connection-specific throughput observations. The prior model $p(\boldsymbol{\phi}_i|\boldsymbol{\theta})$ specifies how much common characteristics shared across all network conditions is explained by the current model parameter $\boldsymbol{\phi}_i$. Note that when $p(\boldsymbol{\phi}_i|\boldsymbol{\theta})$ encodes a non-informative prior, the learning algorithm reduces to the random initialization solution. In the case of linear model, the early stopping of a gradient descent procedure corresponds to a Gaussian prior distribution [73, 180]

$$p(\boldsymbol{\phi}_i|\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\phi}_i; \boldsymbol{\theta}, \mathbf{Q}), \tag{5.8}$$

where $\mathbf{Q}$ is a symmetric positive definite matrix that depends on the step size α_s and the co-variance structure of $\mathbf{C}$. In the non-linear case, the point estimate is not necessarily the global mode of a posteriori. Nevertheless, one can still interpret the point estimate given by truncated gradient descent as the value of the mode of an implicit posterior over $\boldsymbol{\phi}_i$ resulting from a negative log-likelihood, and regularization penalties and the early stopping procedure jointly acting as priors [19, 73].

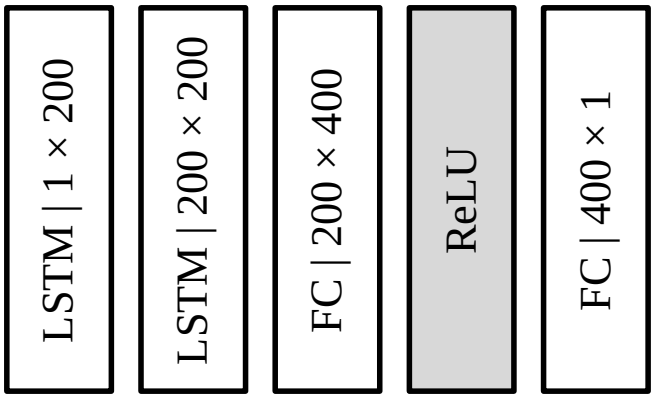


Figure 5.6: Illustration of the neural network configurations for MetaTP. We denote the parameterization of the LSTM and fully connected layer as "input channel × output channel".

5.2.4 Implementation Details

Input: To predict the throughput at $t+1$-th instance, MetaTP takes all previous throughput observations $\mathbf{c}^{1:t} = [c^1, ..., c^t]$ as input. Nevertheless, the proposed model can be easily adapted to ingest additional input features such as Internet service provider, server identifier, and TCP states [200] by adjusting the input feature channels. A recent study has demonstrated that neural network-based throughput prediction may enjoy significant improvement in prediction accuracy with these low-level features [237]. In this study, however, we neglect the auxiliary inputs for two reasons. First, the focus of the study is to demonstrate the effectiveness of the meta learning scheme in throughput prediction, rather than investigating the optimal feature selection. Second, most existing throughput datasets do not provide these information for training and validation.

Network Architecture: We instantiate the throughput predictor using a neural network, which consists of two LSTM [83], a hidden neuron with non-linear activation function, and a linear output layer. The parametrizations of recurrent, fully connected, and connectivity from layer to layer are detailed in Figure 5.6.

We choose to use RNN as the basic building block of throughput predictor instead of Multi-Layer Perceptron (MLP) and Convolutional Neural Network (CNN) for two reasons. First, the evolution of throughput exhibits long-term dependency, and may even be affected by initial throughput conditions and throughput evolution patterns [200]. Such a case presents obvious problems for fixed size history window approaches [131], which attempt to resolve the hidden state by making the chosen action depend on a fixed number of the most recent observations and actions. If the relevant piece of information to be remembered falls outside the history window, the model cannot use it. On the other hand, our RNN structure can efficiently encode high-order statistics of throughput conditions with a hidden state variable [83]. Second, unlike history window approaches, RNN does not have to represent entire histories, but can in principle extract and represent just the relevant information for an arbitrary amount of time. However, MLP and CNN usually fail to discover the correlation between a piece of information and the moment at which this information becomes relevant, given the distracting observations between them.

Learning Algorithm Instantiation: Following the recommendation in MAML [61], we adopt the Adam optimization algorithm [107] as the meta-optimize with a mini-batch of 10 tasks. We sample $N = 5$ traces from each task in each training iteration. For pretraining, we start with the learning rate $\alpha_s = 10^{-2}$ and subsequently lower it by a factor of 10 when the loss plateaus, until $\alpha_s = 10^{-4}$. Other parameters in Adam are set by default. During fine tuning, α_s and M are fixed to 10^{-4} and 5, respectively. To evaluate performance, we fine tune a single meta-learned model with $\alpha_s = 10^{-4}$ and for 5 gradient steps. We implement MetaTP using Pytorch [160], which allows for automatic differentiation through the gradient updates during meta learning.

5.3 Evaluation

In this section, we first compare MetaTP with classic and state-of-the-art throughput prediction models. We then conduct a series of ablation experiments to identify the core contributors of MetaTP. Furthermore, we investigate the capability of MetaTP to incorporate more sequentially incoming training data. Finally, we shed light on some practical

concerns with using meta learning generated throughput prediction models.

5.3.1 MetaTP vs. Existing Throughput Predictors

We conduct a trace-driven experiment to validate the performance of the proposed MetaTP model. To begin with, we describe the experimental setups including the evaluation framework, throughput datasets, competing throughput prediction models, and evaluation criteria. The detailed experimental results are given in the end of the subsection.

Experimental Setup

Evaluation Framework: We employ a trace-driven experimental methodology to evaluate the performance of throughput prediction models. In particular, we collect throughput traces from a great variety of real-world network environments. Each trace contains a sequence of throughput measurements from a client device. The t-th entry encodes the average throughput value observed in the interval $[(t-1)G, tG]$, where G represent the granularity of measurements. We refer to such a period as an "epoch". To facilitate the connection-level throughput characterization, each client (identified by its IP address) involved in the experiment has to contribute a sufficient number of throughput traces. Given a throughput trace, each throughput prediction model evaluates $p(\hat{c}^{t+1}|\mathbf{c}^{1:t}, \boldsymbol{\phi}_i)$ for all $t \in [1, T-1]$, and the prediction results are compared to the ground truth value c^{t+1}.

Datasets: To train and evaluate MetaTP and state-of-the-art throughput prediction models on realistic network conditions, we created a corpus of network traces by combining several public datasets: two broadband datasets namely Federal Communications Commission (FCC) [60] and Puffer [237], a 3G dataset called HSDPA [171], two 4G datasets from University College Cork (UCC) [167] and Belgium [213], and a 5G dataset from UCC [166]. The FCC dataset contains more than 1 million throughput traces, each of which records the average throughput over $2,100$ seconds at a granularity of 5 seconds. Each trace is associated with a unique connection identifier. We select $10,000$ sessions from 500 clients by randomly cutting from the raw connection-level throughput traces, each with a duration of 120 seconds. The HSDPA dataset comprises 3G throughput measurements at a

granularity of 1 second, collected from mobile devices that were streaming video while in transit. The experiments were performed in the period Sep. 13, 2010 to Apr. 21, 2011 in Norway. 6 out 11 scenes contain more than five traces, resulting in a total of 78 valid traces. We apply a sliding window to generate $1,000$ throughput traces to form the 3G network dataset. The Belgium dataset consists of 40 4G bandwidth traces recorded along several routes in and around the city of Ghent at a 1-second granularity. The UCC dataset is composed of client-side cellular key performance indicators collected across different mobility patterns (static, pedestrian, car, tram and train). The 4G trace dataset contains 135 traces, with an average duration of fifteen minutes per trace at a granularity of one sample per second. The dataset also contains synthetic throughput traces from 100 mobile users. We consider each route corresponds to a network environment, based on which the connection-level throughput traces are extracted. To match the duration of our selected FCC traces, we generate $10,000$ traces using a sliding window across the two 4G datasets, each with a duration of 120 seconds. The 5G dataset comprises 45 traces collected from 15 environmental conditions. We follow a similar way to pre-process the data, obtaining $1,000$ traces. Started since Jan. 26, 2019, the Stanford Puffer dataset is an ongoing research project that collects connection-level data in a realistic streaming video environment. To date, the dataset includes more than 5M individual throughput traces collected on the Amazon Mechanical Turk platform. We select all traces from June 2019 to May 2021 in the construction of the WaterlooSV database. During the experiment, a number of environmental statistics are logged from both video servers and players per video chunk. As a result, the Puffer dataset does not provide the throughput at a fixed granularity. To make the data format consistent, we apply linear interpolation to the source data such that the a throughput measurement is recorded every two seconds. Each trace in the Puffer dataset is assigned a SessionID, and throughput traces with the identical SessionID are generated from the same connection. We cut every raw trace into 120-second sequences and randomly select 20 traces for every task. With this pro-processing above, we randomly select $10,000$ traces from 500 users. In total, we obtain $32,000$ throughput traces from $1,600$ clients, each with duration of 120 seconds.

We randomly split the dataset into 60% training set, 20% validation set, and 20% meta-test set based on the UserID. As a result, throughput prediction models do not observe the

test environment during the pretraining process. We optimize the throughput prediction models on the training set with various hyper-parameters, and select the best model on the validation set. To simulate the few-shot learning scenario, we randomly split traces from each UserID in the meta-test set into 20% fine tuning (also called fast adaptation) set and 80% test set. The pretrained models are adapted to each connection condition represented by UserID using the fine tuning set, and are then evaluated on the test set. We over-sample the 4G and broadband throughput traces to balance the task distribution during pretraining.

Throughput Predictors: We consider several representative throughput prediction models listed in Table 2.1, including Nearest Neighbourhood (NN), Arithmetic Mean (AM) [63], Harmonic Mean (HM) [102], Exponential Weighted Moving Average (EWMA) [102], Linear Regression (LR) [80], HMM [200], and MLP [237]. We optimize the trainable parameters of these models on the training set, and manually tune the hyper-parameters of machine learning-based throughput predictors on the validation set, including (γ, C) for SVR, the number of hidden state in HMM, and (number of layers, bin size) in MLP.

Evaluation Criteria: Given the stochastic nature of network dynamics, ideally we should measure the performance of throughput predictors using some divergence measures between the ground-truth task-specific joint throughput distribution and the probabilistic model prediction. In practice, however, the ground-truth connection-level throughput distribution is unavailable. Instead, we only have limited samples from the realistic throughput distributions. To this end, we use the following two proxy evaluation criteria to compare the performance of throughput prediction models.

- Normalized Expected Log-Likelihood (NELL) is computed as

$$\begin{aligned} \text{NELL} &= \frac{1}{JT}\sum_{j=1}^{J}\log p(\mathbf{c}_{i_j};\boldsymbol{\phi}_i) \\ &= \frac{1}{JT}\sum_{j=1}^{J}\sum_{t=1}^{T-1}\log p(c_{i_j}^{t+1}|\mathbf{c}_{i_j}^{1:t}) + \log p(c_{i_j}^{1}) \end{aligned} \tag{5.9}$$

  where J and T represent the number of test samples per task, and the number of datapoints in each trace, respectively. NELL measures the likelihood that the observed

data is generated by a particular model. It has become the de facto performance measure for deep generative models [155]. Given the stochastic nature of network dynamics, we believe NELL is also an appropriate evaluation criterion for throughput prediction models. Higher NELL suggests better performance.

For deterministic throughput prediction models, we assume the prediction results come from the maximum likelihood solution of a Gaussian distribution, which is centered at the predicted value. We estimate the variance of the Gaussian distribution to maximize the log-likelihood on the training set. The evaluation of probabilistic throughput predictors is straight-forward.

- The L_1 metric is defined as

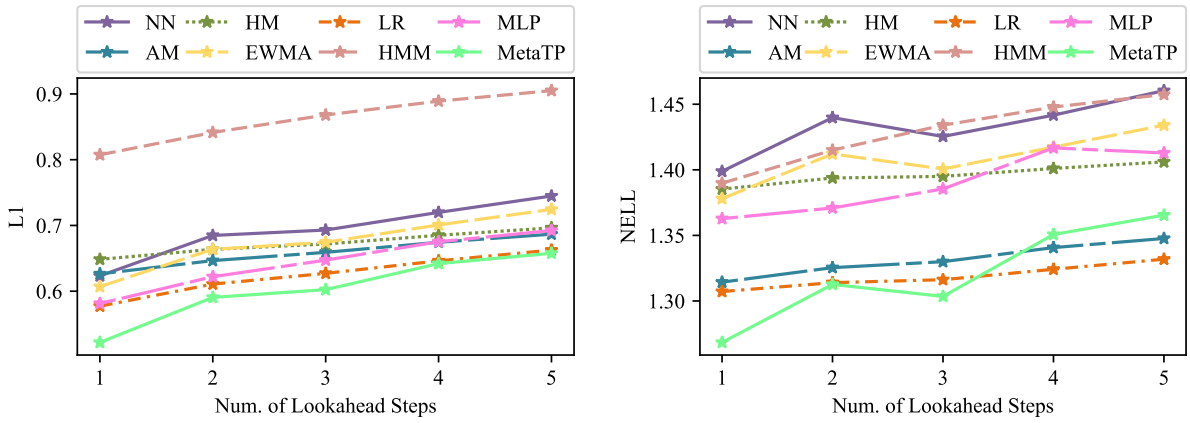
$$L_1 = |\hat{c}^{t+1} - c^{t+1}|, \tag{5.10}$$

where $\hat{c}^{t+1}$ and c^{t+1} are the predicted throughput and actual throughput, respectively. The motivation for using L_1 as the evaluation criteria is threefold. A lower L_1 corresponds to a better prediction accuracy.

We directly compute L_1 on each datapoint in the test set for deterministic throughput prediction models, whereas for probabilistic models, we firstly obtain a point estimate of throughput value $\hat{c}^{t+1}$ with the maximum likelihood estimator and then apply (5.10).

Experimental Results

Figure 5.8 depicts the NELL and L_1 Cumulative Density Function (CDF) of the throughput predictors across all sessions, from which we have two observations. First, although most throughput prediction models achieve similar performance at stable network connections, MetaTP consistently produces the best prediction accuracy over a wide range of scenarios. In particular, MetaTP can effectively reduce the moderate ($L_1 > 2$) and significant prediction errors ($L_1 > 4$) over its best competitors by 2% and 6%, respectively. Second, somewhat surprisingly, the linear regression model rivals or outperforms state-of-the-art data-driven models MLP and HMM. This suggests that there exists significant heterogeneity in the connection-level network dynamics. The good prediction accuracy on

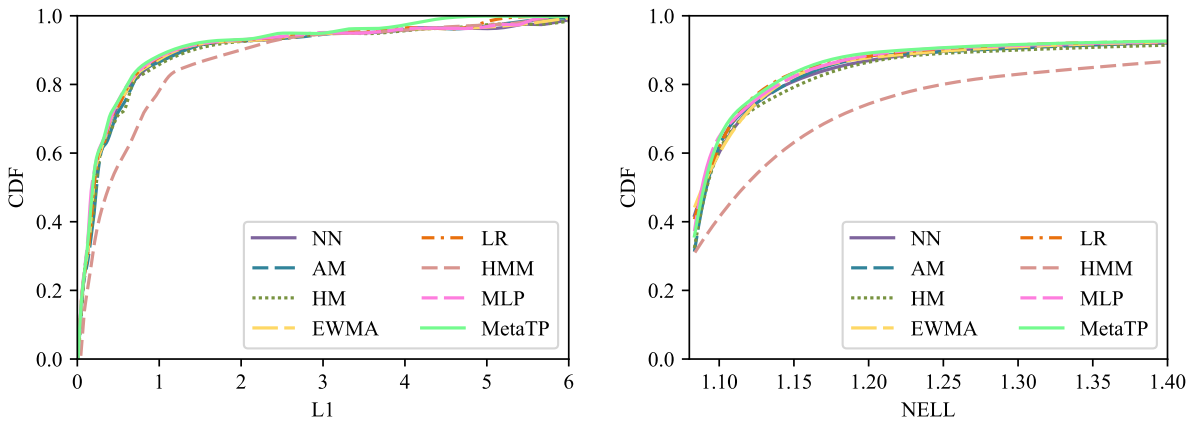


(a) L_1 (b) NELL

Figure 5.7: The optimality and robustness scores in BSQI of the sub-components of EERO. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average.

the observed environment does not generally transfer to novel viewers, whose throughput dynamics may deviate significantly from the training set.

Look-Ahead Horizon: We also study the performance of throughput prediction models for longer prediction horizon up to 5. The long-term prediction is crucial for adaptive streaming algorithms because theoretically an ABR player can only obtain the global optimal solution with the perfect knowledge about the throughput characteristics over the entire session [131]. In practice, some ABR algorithms [200, 237, 241] explicitly require throughput prediction into a fixed look-ahead horizon. To adapt the competing models for longer prediction horizon, we modify the output node of network architecture accordingly. Figure 5.7 shows the experimental results over a range of prediction horizon. We see that MetaTP outperforms the existing algorithms for all look ahead horizon with a sizable margin, achieving 2%-10% improvements over the best competitor on average.



(a) L_1 (b) NELL

Figure 5.8: The optimality and robustness scores in BSQI of the sub-components of EERO. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average.

5.3.2 Ablation Experiment

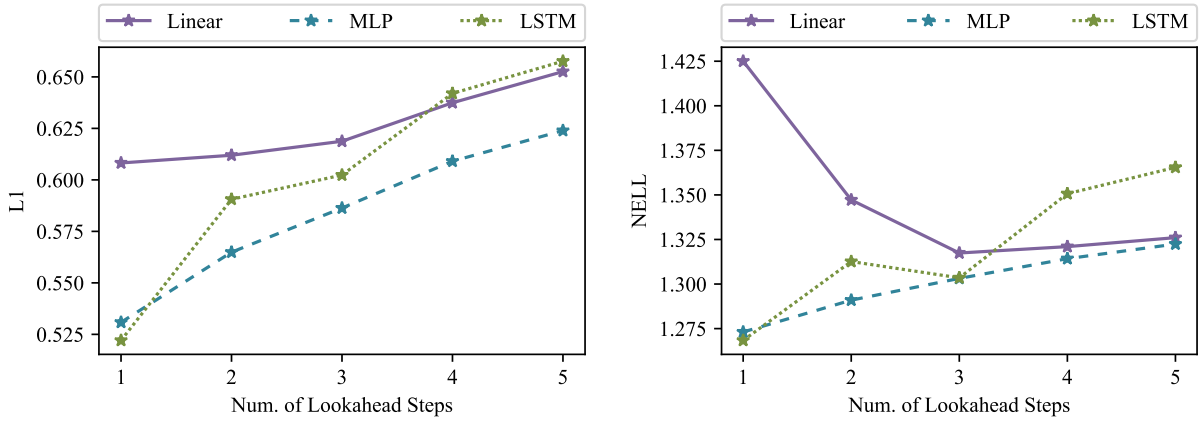
We conduct a series of ablation experiments to single out the core contributors of MetaTP. Specifically, we investigate the impacts of training method and network architecture.

MetaTP vs. Other Baselines

We start with analyzing the impact of meta learning in throughput prediction. There are two features that distinguishes MAML from traditional supervised learning approaches. First, the objective function of MAML is evaluated using the updated model parameters $\boldsymbol{\phi}_i$. Second, the maximum a posteriori is optimized with an early stopping procedure. To this end, we compare MetaTP with four baseline models: (1) pretraining on all tasks using the traditional maximum likelihood method, and fine tuning on the fast adaptation set till convergence (namely pretrain), (2) pretrain with early stopping procedure, (3) training on the fast adaptation set from random initialization till convergence (namely random initialization), and (4) random initialization with early stopping training method. All competing

Table 5.1: Quantitative results of MetaTP with different learning algorithms.

Model	L_1	NELL
pretrain	0.554	1.275
pretrain + early stopping	0.553	1.274
random initialization	0.763	1.482
random initialization + early stopping	1.059	1.504
MetaTP	0.522	1.263



(a) L_1 (b) NELL

Figure 5.9: The performance of MetaTP with other network architectures.

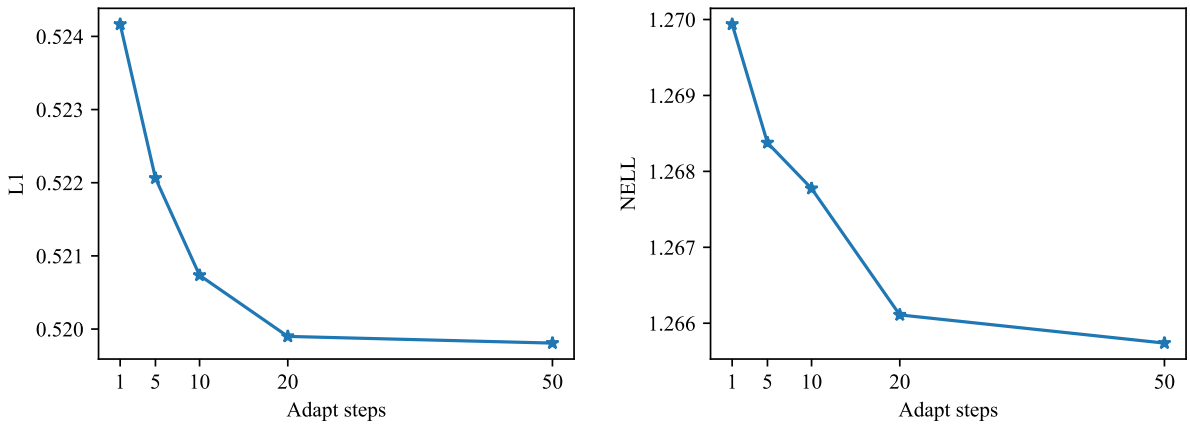
models share a common network architecture, and are optimized by a variant of stochastic gradient descent algorithm Adam [107]. We perform the experiments for ten times and report the median performance to mitigate the bias introduced from a particular random initialization and the stochastic optimization process. From Table 5.1, we observe that fine tuning from the two alternative prior models do not lead to competitive performance. In particular, Training the neural network model with only few throughput traces results in significant overfitting. Furthermore, early stopping only negatively influences the performance of fine tuning. We conclude that the gradient through a gradient computation is the key to the success of MetaTP.

Table 5.2: Quantitative results of MetaTP with different numbers of hidden states and layers.

# of Hidden States	# of Layers	L_1	NELL
50	1	0.577	1.405
50	2	0.565	1.399
50	3	0.553	1.380
100	1	0.560	1.397
100	2	0.549	1.379
100	3	0.534	1.269
150	1	0.546	1.271
150	2	0.538	1.267
150	3	0.529	1.263
200	1	0.535	1.274
200	2	0.522	1.263
200	3	0.519	1.260

Network Architecture

Next, we examine the impact of neural network architecture on the prediction accuracy. We start from a baseline by replacing the LSTM by a naïve linear regression model that takes a fixed window of throughput history as input. To extend the model capacity, we also train a MLP model that has been used in a state-of-the-art throughput predictor Transmission Time Predictor (TTP). The parametrizations of the MLP is detailed in Appendix A.1. Both model are adapted to take the past 8 throughput observations as input and to produce throughput predictions for the next 5 time instances as output. We apply MAML to train the baselines. All competing models employ Adam [107] as the meta-optimizer. The results in Figure 5.9 illustrate that MetaTP is generally insensitive to the network architecture. On the other hand, by comparing the performance of the meta learned MLP with the experimental results from the previous subsection, we can observe that MAML can also improve throughput prediction models with sufficient model capacity. This observation reinforces our conclusion that the meta learning scheme is the core contributor of MetaTP.



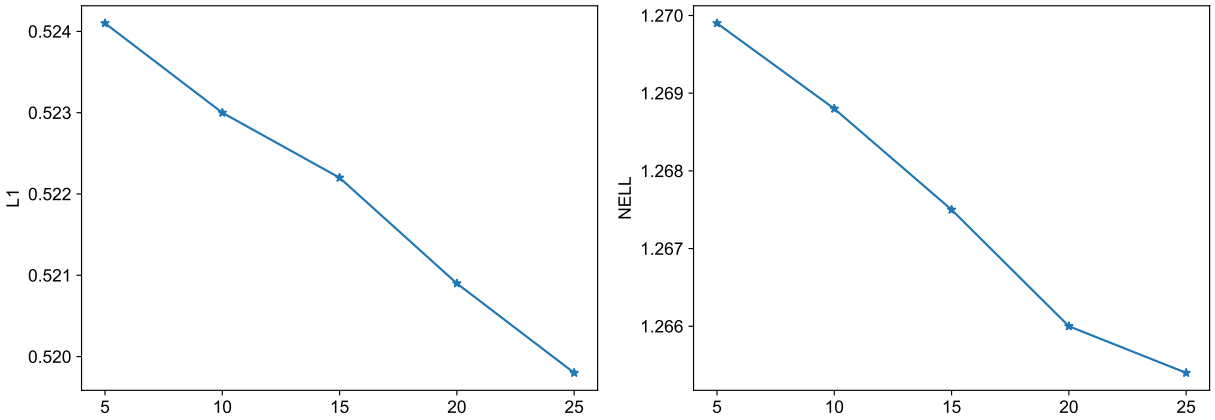
(a) L_1 (b) NELL

Figure 5.10: The performance of MetaTP with additional gradient steps.

We also investigate the impact of hyper-parameters in the RNN model. Starting from the default network architecture in Figure 5.6, we sweep through a range of hidden state numbers and recurrent layer numbers. Results from this sweep are presented in Table 5.2. As shown, performance begins to plateau once the number of hidden states exceeds 150. We also find that the network of 3 cascaded RNN layers yields the best performance. Nevertheless, the performance varies relatively insignificantly with respect to the RNN architecture.

5.3.3 Continual Learning

Given the greedy optimization scheme, it is natural to ask the following question: Can MetaTP continue to improve with additional gradient updates and/or throughput observations? If the throughput predictor converges to a local optimum after one gradient step, the ABR players equipped with MetaTP cannot benefit from more throughput measurements. To answer the question, we extend the experiment in Section 5.3.1 by fine tuning the meta-learned model on varying numbers M of throughput traces and gradient steps. Note that the dataset in Section 5.3.1 is inadequate for the evaluation of continual learn-

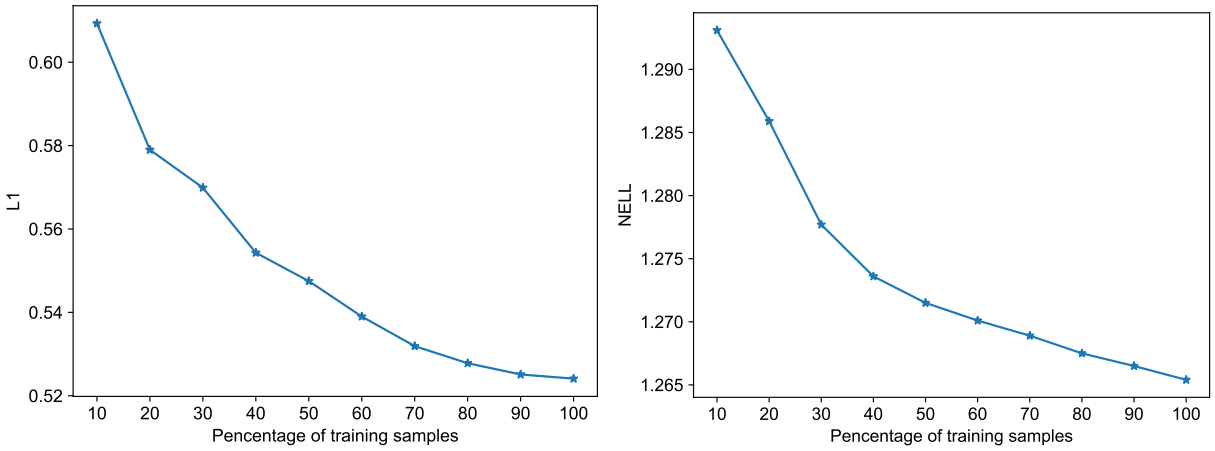


(a) L_1 (b) NELL

Figure 5.11: The performance of MetaTP with additional training samples.

ing, because the number of connection-level throughput traces is limited. We augment the meta-test dataset as follows. First, we go back to the source throughput datasets and extract all IP addresses with more than 50 measurement sessions. Second, we filter out traces to make sure that there is no overlap in connection identifiers between the original training set and the augmented meta-test set. The meta-test set is randomly split into 60% fast adaptation set and 40% test set. To investigate the impact of training samples, we randomly draw M samples from the fast adaptation set to fine tune the MetaTP.

Figure 5.10 and Figure 5.11 demonstrate the experimental results, from which we have two observations. First, MetaTP continues to improve with additional gradient steps and training samples, despite being trained for maximal performance with only one gradient step. This improvement suggests that the meta learning algorithm optimizes the throughput predictor in a generalizable way such that the prior model parameters lie in a region that is amenable to fast adaptation, rather than overfitting to parameters that only improves after one training iteration. As a result, it is possible to continuously optimize MetaTP as new connection-level throughput observations become available.



(a) L_1 (b) NELL

Figure 5.12: The performance of MetaTP with different portion of training samples.

5.3.4 Discussion

Training Time: To measure the overhead of generating meta learning-based throughput prediction models, we profile the training process of MetaTP. Training a prior model requires approximately 0.72M iterations, where each iteration took 80 ms on a single NVIDIA Pascal Titan X GPU. Thus, the meta learning process took about 16 hours. Albeit somewhat expensive, the training cost is incurred offline. The meta learning can be applied infrequently depending on the environment stability (*e.g.* whenever the next-generation of network becomes available). To simulate the fast adaptation process in practical ABR applications, where the computation power of edge device may be limited, we fine tune the prior model on a single Intel Core i7-4790 processor at 3.60 GHz. The fast adaptation by default requires one backward pass, which took 10 ms. Given the minimal cost, it is feasible to apply the continual learning as discussed in Section 5.3.3.

Training Data: To understand the impact of training data in the meta learnt prior model, we compare our default model with other models trained with only 10%-90% training data. We observe significant performance gain with increase in the number of training traces, as shown in Figure 5.12.

Computation Time: The throughput prediction in a practical ABR system should be computationally efficient at test time. A second delay in bitrate selection could results in a significant degradation in the overall performance. To this end, we compare the computation complexity of the proposed RNN model with the existing throughput predictors. The experiment is performed on a single Intel Core i7-4790 processor at 3.60 GHz. Each forward pass of MetaTP took 2 ± 1 ms, which is on par with the state-of-the-art HMM and MLP. The ~ 2 ms delay in ABR decision would have negligible impact to the system as suggested by a recent study [131].

5.4 Summary

In this chapter, we propose a connection conditional throughput prediction model, namely MetaTP, using meta learning. MetaTP can effectively solve the distribution mismatch problem incurred by the traditional "one-size-fits-all" models, and alleviate the overfitting problem experienced by the maximum likelihood methods with connection-level throughput traces. Extensive experimental results demonstrate the effectiveness of the proposed MetaTP with higher accuracy and improved robustness in environmental variations. Thanks to the Bayesian framework, the proposed method can be naturally extended with continual learning scheme to obtain further improvement.

Chapter 6

EERO: Towards An Efficient, Robust, and Optimal Bitrate Selector

In this chapter, we recast the bitrate adaptation as a Bayesian inference problem, with the goal of unifying a wide spectrum of ABR algorithms under a common framework. This new BBS framework allows us to identify the successes and limitations of the existing approaches for bitrate selection, based on which we develop a new ABR algorithm. The proposed algorithm, namely EERO, is a natural extension of two prevailing and complementary design paradigms, effectively integrating their strengths in a principled way.

6.1 A Bayesian Framework for Bitrate Selection

Given a reward function and a throughput distribution/sample, the goal of bitrate adaptation function is to select the optimal bitrate a_t^* given a state observation $\mathbf{s}_t$. The goodness of the action a_t^* is measured by the expected cumulative reward function $q_t^* = Q^*(\mathbf{s}_t = \mathbf{s}, a_t = a) = \max_{\boldsymbol{\pi}} \mathbb{E}[U_t|\mathbf{s}_t = \mathbf{s}, a_t = a, \boldsymbol{\pi}]$, where $\boldsymbol{\pi}$ and U_t represent a policy function that maps a state sequence to an action and the discounted cumulative reward starting at the time instance t. The discounted cumulative reward q_t^* and the function mapping the state-action to the value are often termed as the Q-value and Q-function in the reinforcement

learning literature. Mathematically, the discounted cumulative reward can be expressed as

$$U_t = \sum_{l=0}^{T-1} \gamma^l u_{t+l}, \tag{6.1}$$

where $0 < \gamma \leq 1$ and T denote a constant discounting future reward and the number of chunks in a streaming session, respectively. As a result, the key of ABR problem is to approximate the optimal action-value function $q_t^* = Q^*(\mathbf{s}_t = \mathbf{s}, a_t = a)$.

The Q-function approximation problem can be recast as a Bayesian inference problem, where the objective is to estimate the action-value conditional distribution $p(q_t^*|\mathbf{s}_t, a_t)$. The general form of the distribution is not known. However, it is possible to sample the distribution using sophisticated optimization techniques. Given a state $\mathbf{s}$ comprising the current buffer occupancy, the previous bitrate trajectories, the future network dynamics (either from a throughput trace or a throughput predictor), and the future chunk characteristics (either from a manifest file or a streaming video model), one can obtain a sample from the distribution $p(q_t^*|\mathbf{s}_t, a_t)$ with value iteration (an instance of dynamic programming) [201]. Specifically, the optimal cumulative reward can be acquired by iteratively applying the following updating rule till convergence

$$q_t^* = \mathbb{E}_{s_{t+1} \sim \mathcal{E}}[u_{t+1} + \gamma \max_{a_{t+1}} q_{t+1}^*]. \tag{6.2}$$

Although it seems appealing to integrate the sampling scheme in the bitrate decision process, the high-dimensional iterative computation is inconvenient, time-consuming, and expensive. Instead, it is common to utilize a function approximator to estimate the action-value function based on a set of offline generated samples. Formally, given a set of training data $\mathcal{D}_\mathbf{q}$ comprising $N_\mathbf{q}$ state action pairs $(\mathbf{S}, \mathbf{a}) = \Big((\mathbf{s}_1, a_1), ..., (\mathbf{s}_{N_\mathbf{q}}, a_{N_\mathbf{q}})\Big)$ and their corresponding optimal cumulative rewards $\mathbf{q}^* = (q_1^*, ..., q_{N_\mathbf{q}}^*)$, we aim to find a posterior action-value distribution $p(q_t^*|\mathbf{s}_t, a_t, \mathcal{D}_\mathbf{q})$ that best approximates $p(q_t^*|\mathbf{s}_t, a_t)$ for all t. We can drop the dependency on t by assuming an infinity long decision process or enriching the state variable $\mathbf{s}$ with the time instance t. $p(q_t^*|\mathbf{s}_t, a_t, \mathcal{D}_\mathbf{q})$ can be considered as a point estimate of $p(q_t^*|\mathbf{s}_t, a_t)$ according to the Bayes' Rule $p(q_t^*|\mathbf{s}_t, a_t) = \int p(q_t^*|\mathbf{s}_t, a_t, \mathcal{D}_\mathbf{q}) p(\mathcal{D}_\mathbf{q}) \mathrm{d}\mathcal{D}_\mathbf{q}$. If the training data in $\mathcal{D}_\mathbf{q}$ are independent and identically distributed, one can parametrize

the action-value function [37] as

$$p(q^*|\mathbf{s}, a, \mathcal{D}_\mathbf{q}) = \int p(q^*|\mathbf{s}, a, \boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{D}_\mathbf{q}) \mathrm{d}\boldsymbol{\theta}, \tag{6.3}$$

where $p(q^*|\mathbf{s}, a, \boldsymbol{\theta})$ and $p(\boldsymbol{\theta}|\mathcal{D}_\mathbf{q})$ represent the parametric action-value probability density function, and the posterior distribution over parameters, respectively. Given the enormous space of $\boldsymbol{\theta}$, the integration in Equation (6.3) is computationally intractable. As a result, a common practice is to approximate the predictive distribution $p(q^*|\mathbf{s}, a, \mathcal{D}_\mathbf{q})$ by a point estimate $p(q^*|\mathbf{s}, a, \boldsymbol{\theta}^*)$, where

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathcal{D}_\mathbf{q}) = \arg\max_{\boldsymbol{\theta}} p(\mathbf{q}^*|\mathbf{S}, \mathbf{a}, \boldsymbol{\theta}) p(\boldsymbol{\theta}). \tag{6.4}$$

The specific form of parametric action-value function $p(\mathbf{q}^*|\mathbf{S}, \mathbf{a}, \boldsymbol{\theta})$ is generally unknown. In practice, existing ABR algorithms usually minimize the mean squared error between the predicted action-value and the ground-truth q^*, which implicitly assumes a Gaussian likelihood function

$$p(q^*|\mathbf{s}, a, \boldsymbol{\theta}) = \mathcal{N}(q^*|\mu(\mathbf{s}, a; \boldsymbol{\theta}), \sigma^2), \tag{6.5}$$

where $\mu(\mathbf{s}, a; \boldsymbol{\theta})$ and σ^2 denote the mean and variance of the Gaussian distribution, respectively. Once the predictive distribution $p(q^*|\mathbf{s}, a, \boldsymbol{\theta}^*)$ is determined, the optimal action a^* can be obtained by selecting the bitrate with the maximum cumulative reward

$$\pi(a|\mathbf{s}) = \prod_{a' \neq a} p(Q(\mathbf{s}, a) \geq Q(\mathbf{s}, a')). \tag{6.6}$$

It should be noted that the variance of the Gaussian distribution in (6.5) generally varies with respect to different state action pairs. Existing ABR methods only approximate the mean of the action-value distribution, essentially assuming a fixed variance σ^2 across the state action space. It can be shown that the optimal policy in (6.6) under this condition is only a function of $\mu(\mathbf{s}, a; \boldsymbol{\theta})$ in (6.5) [207]. During the decision making stage, most bitrate adaptation functions [32, 33, 91, 92, 131, 212] adopt a winner-take-all strategy by always selecting the maximum likelihood bitrate

$$a^* = \max_a \boldsymbol{\pi}(a|\mathbf{s}), \tag{6.7}$$

which leads to the highest mean cumulative reward. Some efforts [92, 131] have also directly modeled the optimal policy $\boldsymbol{\pi}(a|\mathbf{s})$ with a parametric approximation $\boldsymbol{\pi}(a|\mathbf{s}, \boldsymbol{\theta})$, but the equivalence between the two approaches is made clear in (6.6). For better consistency, we perform the subsequent analysis from the action-value perspective.

6.1.1 The Bayesian Interpretation of Existing Arts

In this section, we will focus on the Bayesian interpretation of two prevailing bitrate adaptation logic, which are closely related to the proposed method. The Bayesian view of a more comprehensive list of ABR algorithms is provided in Section 2.3.2.

Model Predictive Control

MPC [241] is a control theoretic ABR algorithm that approximates the action-value probability distribution $p(q^*|\mathbf{s}, a)$ on the fly. At the t-th bitrate decision, the algorithm estimates the expected cumulative reward over the interval $[t, t + K]$ for each action a, and select the action that optimizes the Q-value, where $K \leq T$ is the planning window. At the next iteration, the algorithm takes the updated state information as the input, re-plans the bitrate trajectory, and produces the optimal bitrate selection.

From a Bayesian perspective, MPC corresponds to the maximum likelihood estimation of the posterior distribution with a noisy sample. Specifically, the look-ahead and reward evaluation procedure at a particular state $\mathbf{s}$ can be regarded as sampling from the optimal action-value function $p(q^*|\mathbf{s}, a)$ for each a. Since the value estimation only involves the cumulative reward in a truncated decision window, we can consider the obtained reward as a noisy sample $\tilde{q} = \tilde{Q}(\mathbf{s}, a)$ of $p(q^*|\mathbf{s}, a)$ such that $\tilde{q} = \sum_{l=0}^{K-1} \gamma^l u_l$. Alternatively, one can interpret the truncated cumulative reward $\tilde{q}$ as a lower bound of the action-value q_t^*

$= \sum_{l=0}^{T} u_{t+l}$. Formally, MPC solves the following optimization problem

$$\begin{aligned} \mu(\mathbf{s}, a)^* &= \arg\max_{\mu(\mathbf{s},a)} p\Big(q^* = \tilde{q} | \mu(\mathbf{s}, a)\Big) \\ &= \arg\min_{\mu(\mathbf{s},a)} -\log \mathcal{N}\Big(q^* = \tilde{q} | \mu(\mathbf{s}, a), \sigma^2\Big) \\ &= \arg\min_{\mu(\mathbf{s},a)} \frac{\Big(\mu(\mathbf{s}, a) - \tilde{q}\Big)^2}{2\sigma^2}. \end{aligned} \tag{6.8}$$

It is apparent that the maximum likelihood solution is $\mu(\mathbf{s}, a) = \tilde{Q}(\mathbf{s}, a)$ for all feasible bitrate selection $a \in \mathcal{A}$. MPC takes a non-informative prior in the Bayesian framework because it does not have any knowledge about the mean action-value $\mu(\mathbf{s}, a)$ before the noisy sample is observed.

The MPC framework provides a flexible control knob to adjust the tradeoff between efficiency and optimality. When $K = 1$, MPC degrades gracefully to the greedy algorithm, prioritizing efficiency over optimality. As K approaches T, the algorithm gradually converges to the global optimal solution at the cost of extremely high computational complexity. Recall that the reward optimization problem over an interval K can be solved by exhaustive search or dynamic programming, whose time complexity are $O(|\mathcal{A}^K|)$ and $O(K \times |\mathcal{S}| \times |\mathcal{A}|)$, respectively. $|\cdot|$ denotes the cardinality operator. To obtain a reasonable approximation of action-value q within a limited time budget, the look-ahead horizon K typically ranges from 5 to 8 [3, 200, 237, 241]. Some recent studies have shown that the truncated dynamic programming may deviate from the global optimal solution [92, 131]. Nevertheless, MPC is quite robust to the unobserved environment, partially because it learns the posterior distribution at test time. Furthermore, it does not make any assumption about the throughput dynamics and reward function. In particular, novel throughput prediction models and reward functions can be applied in the framework in a plug-and-play fashion.

Learning-Based Approach

A significant effort has been devoted to improving the optimality of ABR algorithms based on sophisticated machine learning techniques. These data-driven methods learn to make

ABR decisions either from expert demonstration or through observations of the resulting performance of past decisions. We review representative methods in each of these two sub-categories and discuss what role they play in the Bayesian framework.

- Imitation Learning: Comyco [92] is an offline bitrate adaptation rule that approximates the action-value distribution in a simulated environment. In contrast to MPC that approximates the Q-value at test time with noisy samples, Comyco [92] employs a parametric model to imitate the behaviour of an expert policy. Given a throughput trace, a streaming video, a player status, and a reward function, the imitation learning algorithm firstly computes a global optimal cumulative reward by solving the optimization problem in (6.2) with dynamic programming. By repeatedly maximizing the cumulative reward across a wide range of state action pairs, one can generate a training set $\mathcal{D}_{\mathbf{q}} = \Big((\mathbf{s}_1, a_1, q_1^*), ..., (\mathbf{s}_{N_{\mathbf{q}}}, a_{N_{\mathbf{q}}}, q_{N_{\mathbf{q}}}^*)\Big)$. Despite the considerably high complexity, the cost of expert demonstration is incurred offline. Comyco [92] then takes advantage of powerful neural networks to predict the optimal action at a given state using supervised learning. Formally, the method minimizes the behavioral cloning loss

$$L(\boldsymbol{\theta}) = \frac{1}{N_{\mathbf{q}}} \sum_{i=1}^{N_{\mathbf{q}}} \Big(\mu(\mathbf{s}_i, a_i; \boldsymbol{\theta}) - q_i^*\Big)^2, \tag{6.9}$$

  to produce the optimal action-value function approximator $\mu(\mathbf{s}, a; \boldsymbol{\theta})$.

  Since the adaptation rule does not integrate information from test time observations, one can regard it as a prior distribution $p(\boldsymbol{\theta})$ for the action-value function. The behavioral cloning method in (6.9) can be derived from a maximum likelihood

approach

$$\begin{aligned}\boldsymbol{\theta}^* &= \arg\max_{\boldsymbol{\theta}} p(\mathcal{D}_{\mathbf{q}}|\boldsymbol{\theta}) \\ &= \arg\max_{\boldsymbol{\theta}} \prod_{i}^{N_{\mathbf{q}}} p(q_i^*|\mathbf{s}_i, a_i; \boldsymbol{\theta}) \\ &= \arg\min_{\boldsymbol{\theta}} -\sum_{i=1}^{N_{\mathbf{q}}} \log p(q_i^*|\mathbf{s}_i, a_i; \boldsymbol{\theta}) \\ &= \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N_{\mathbf{q}}} \Big(\mu(\mathbf{s}_i, a_i; \boldsymbol{\theta}) - q_i^*\Big)^2,\end{aligned} \tag{6.10}$$

where we have assumed a Gaussian likelihood function in (6.5). The optimization problem in (6.10) gives a point estimate for $\boldsymbol{\theta}$, an instance of a method known as empirical Bayes [19] due to its use of the data to estimate the parameters of the prior distribution.

- Reinforcement Learning: As opposed to the imitation learning that learns the policy from expert demonstration, reinforcement learning learns to improve an agent's decision making ability from the interaction with a simulated environment. The basic idea behind many reinforcement learning algorithms is to estimate the action-value function, by using the Bellman equation [201] as an iterative update. Specifically, a parametric action-value function approximator can be trained by minimizing a sequence of loss functions $L_i(\boldsymbol{\theta}_i)$ that changes at each iteration i

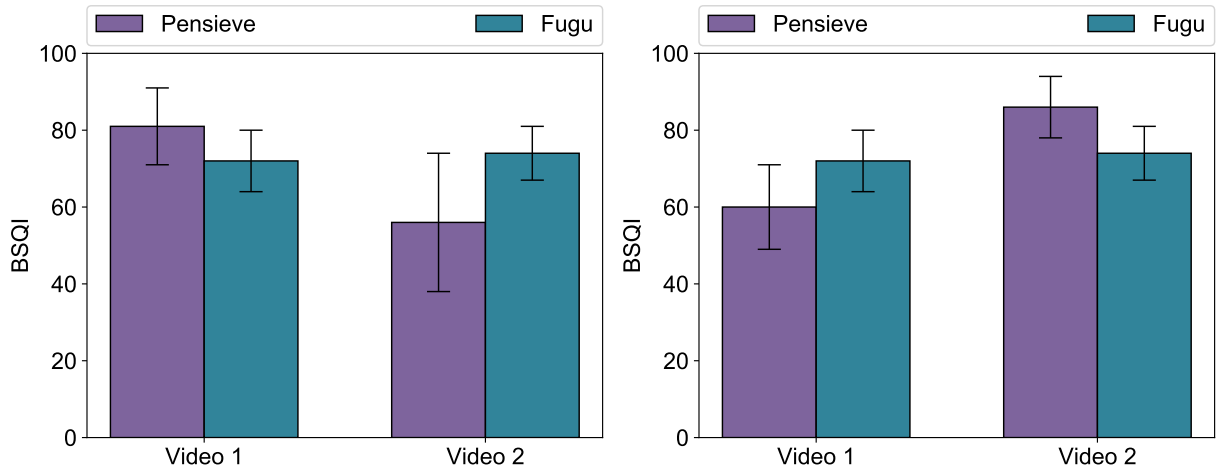
$$L_i(\boldsymbol{\theta}_i) = \mathbb{E}_{\mathbf{s},a\sim\rho(\cdot)}\Big[(q^i - \mu(\mathbf{s}, a; \boldsymbol{\theta}))^2\Big], \tag{6.11}$$

where $q^i = \mathbb{E}_{\mathbf{s}'\sim\mathcal{E}}[u + \gamma \max_{a'} \mu(\mathbf{s}', a'; \boldsymbol{\theta}_{i-1})|\mathbf{s}, a]$ is the target for iteration i and $\rho(\mathbf{s}, a)$ is a probability distribution over states $\mathbf{s}$ and actions a that are commonly referred to as the behaviour distribution [138]. The parameters from the previous iteration $\boldsymbol{\theta}_{i-1}$ are held fixed when optimising the loss function $L_i(\boldsymbol{\theta}_i)$. Note that the targets depend on the network weights; this is in contrast with the targets used for imitation learning, which are fixed before learning begins [138]. Under some mild conditions, it can be proven that the parametric action-value function approximator converges to the optimal solution [201]. Despite a slower convergence rate, the learning agent

experiences many states falling out of the distribution $\rho^*(\mathbf{s}, a)$ in the training stage, where $\rho^*(\mathbf{s}, a)$ represents the behaviour distribution of the optimal policy. As a result, reinforcement learning approaches are generally more robust in unobserved states/environment.

From a Bayesian view, the reinforcement learning approach also produces a prior action-value distribution $p(q^*|\mathbf{s}, a, \boldsymbol{\theta})$. Conceptually, the agent-environment interaction process can be regarded as a data augmentation method to the imitation learning such that the agent not only observes state-action pairs from the optimal behaviour distribution $\rho^*(\mathbf{s}, a)$, but also experiences out of distribution samples. Some reinforcement learning methods even explicitly encourage exploration of the state space with a deliberately designed loss function [131]. Applying the updating rule in (6.11) on the augmented dataset $\mathcal{D}_{\mathbf{q}^+}$ corresponds to maximizing the likelihood function $p(\mathcal{D}_{\mathbf{q}^+}|\boldsymbol{\theta})$ under the Gaussian assumption, which is also an instance of the empirical Bayes method [19].

It has been shown that learning-based ABR algorithms achieve state-of-the-art performance in the environment that the agents experienced in the training process [32, 92, 131, 212]. The learnt action-value function can theoretically converge to the ground-truth $p(q^*|\mathbf{s}, a)$ given sufficient training data, suggesting that the approach is also effective in optimizing the reward function. Furthermore, these agents only take a feed-forward operation and optionally a maximization step to produce a bitrate selection, whose computation complexity is $O(|\mathcal{A}|)$. However, these methods have demonstrated limited generalization capability on state action pairs falling out of the behaviour distribution. Figure 6.1 shows a motivating example, where a representative reinforcement learning-based ABR algorithm Pensieve [131] and MPC are both evaluated on two streaming videos. Both of the streaming videos in the experiment receive a per-title encoding recipe [208], such that the encoding profiles of the first and second videos are $[300, 700, 1800, 3000, 5700, 8000]$ kbps, and $[200, 600, 1000, 1600, 2800, 4300, 6000, 9000]$ kbps, respectively. We firstly train Pensieve with hundreds of streaming videos including the first test video to optimize a QoE model BSQI [45] (but with non-overlapping training/testing throughput traces). We then repeat the same training procedure with the role of the two videos reversed, and evaluate the performance on both test videos. The experimental results confirm our claim that



(a) Case 1

(b) Case 2

Figure 6.1: The data-driven ABR algorithm Pensieve is susceptible to unobserved state-action pairs. The model predictive control-based algorithm Fugu achieves sub-optimal performance but is quite robust to environment shift. Case 1: Pensieve observed the encoding profile of video 1 in the training process. Case 2: Pensieve observed the encoding profile of video 2 in the training process.

learning-based approaches do not generalize well on unobserved state action samples, a problem known as overfitting. The root cause of the problem is the fundamental conflict between the enormous state-action space and the limited computation power for ground-truth label generation. Specifically, in the general case that the instantaneous reward u_t is a function of all previous state and actions, the only feasible approach to determine q^* is by exhaustively searching all the combinations of bitrate selection. The computational complexity of exhaustive search is in the order of $O(|\mathcal{A}|^T)$, where $|\mathcal{A}|$ denotes the cardinality of the action space. Even if we simplify the problem by assuming that u_t enjoys a Markov property, obtaining each ($\mathbf{s}$, a, q^*) sample still involves solving a dynamic programming problem, whose time complexity is $O(T \times |\mathcal{S}| \times |\mathcal{A}|)$. $|\mathcal{S}|$ represents the cardinality of the state space. In the case of adaptive streaming, $\mathbf{s}$ may encode the statistics of future chunks (*e.g.*, time-varying chunk size and quality) and throughput prediction, suggesting that $|\mathcal{S}| \propto T$. Either way, the computation of q^* quickly becomes intractable as the plan-

ning window T becomes longer. A typical "large-scale" dataset allows for a maximum of a few thousands of streaming sessions to be labeled. By contrast, the action-value function lives in a high dimensional space, which is typically in the order of hundreds of thousands. Therefore, a few thousands of action-value samples are deemed to be sparsely distributed in the space.

6.2 Towards Maximum A Posteriori Bitrate Selector

In the previous section, we have reviewed two prevailing ABR bitrate adaptation logic from a Bayesian perspective. Interestingly, these algorithms are complementary to each other in two important ways. First, MPC is quite robust to the change in behaviour distribution, although it can only achieve sub-optimal performance in real-time. On the other hand, data-driven algorithms are close to optimal in observed state-action regions, but may not perform adequately in real-world environment that is inevitably more complex, especially with constantly evolving video delivery modules (*e.g.*, per-title encoding schemes [36, 49, 208], better video encoders [28, 198], and a broader range of viewing devices [169]). Second, MPC and learning-based methods serve as the likelihood function and the prior distribution in the Bayesian inference problem, respectively. These observations motivate us to develop a new ABR algorithm by combining the advantages of these approaches.

The objective of bitrate selection function in the aforementioned sequential learning setting is to approximate the posterior parameter distribution $p(\boldsymbol{\theta}|\tilde{q})$, where $\tilde{q}$ is a noisy sample generated by online dynamic programming. Inherited from the existing approaches [32, 33, 92, 131, 212, 241], the proposed algorithm also assumes a Gaussian likelihood function whose variance across the state-action space is uncorrelated. It follows that the winner-take-all strategy in (6.7) only depend on the mean value of the posterior action-value distribution [207]. As a result, we reduce the general problem in (6.4) to the estimation of $p\Big(\mu(\mathbf{s},\, a;\, \boldsymbol{\theta})|\tilde{q}\Big)$. Formally, we would like to find a set of mean values that maximizes the posterior distribution

$$\mu^* = \arg\max_{\mu} p(\mu|\tilde{q}) = \arg\max_{\mu} p(\tilde{q}|\mu)p(\mu), \tag{6.12}$$

where the likelihood function $p(\tilde{q}|\mu)$ is given by (6.8). Existing data-driven methods approximate the prior distribution $p(\mu)$ with a point estimate in (6.10) or (6.11), neglecting the reliability of the estimation. However, the point estimate may deviate significantly from the real distribution. To this end, we take a second-order approximation around the empirical Bayes estimated mean value $\mu_{\boldsymbol{\theta}}$ by assuming a Gaussian prior distribution

$$p(\mu|\mathcal{D}_{\mathbf{q}}) = \mathcal{N}(\mu|\mu_{\boldsymbol{\theta}}, \sigma_{\mu}^2), \tag{6.13}$$

where the variance of the distribution σ_{μ}^2 is a function of ($\mathbf{s}$, a, $\mathcal{D}_{\mathbf{q}}$). The application of the Gaussian distribution over a point estimate has been a very common technique in the machine learning literature, evident by the prevalence of Laplace's method [111] and kernel method [19]. According to the Taylor's theorem [75, 73], the second-order approximation generally provides a tighter error bound than a point estimate, which is essentially a first-order expansion of the prior distribution. Furthermore, we can make use of this approximation to incorporate the varying uncertainty in Q-values across the input space. Substituting (6.13) into (6.12) and taking the negative logarithm, the objective function becomes

$$\begin{aligned} \mu^* &= \arg\min_{\mu} -\log p(\tilde{q}|\mu) - \log p(\mu|\mu_{\boldsymbol{\theta}}) \\ &= \arg\min_{\mu} \frac{(\mu - \tilde{q})^2}{2\sigma^2} + \frac{(\mu - \mu_{\boldsymbol{\theta}})^2}{2\sigma_{\mu}^2}. \end{aligned} \tag{6.14}$$

It is then straight-forward to show that the maximum a posteriori mean action-value takes the form

$$\mu^* = \frac{\sigma_{\mu}^2}{\sigma_{\mu}^2 + \sigma^2}\tilde{q} + \frac{\sigma^2}{\sigma_{\mu}^2 + \sigma^2}\mu_{\boldsymbol{\theta}}, \tag{6.15}$$

where $\tilde{q}$, $\mu_{\boldsymbol{\theta}}$, and σ_{μ} are functions of state and action. Equation (6.15) underlies a novel ABR algorithm by adaptively combining the cumulative reward $\tilde{q}$ from an online bitrate adaptation function and a learning-based algorithm parametrized by $\mu_{\boldsymbol{\theta}}$. The weighting scheme has an intuitive interpretation as follows. The ABR algorithm would converge to the learned policy if the uncertainty of the cumulative reward at the current state and action σ_{μ} is low, because

$$\lim_{\sigma_{\boldsymbol{\mu}} \to 0} \mu^* = \mu_{\boldsymbol{\theta}}. \tag{6.16}$$

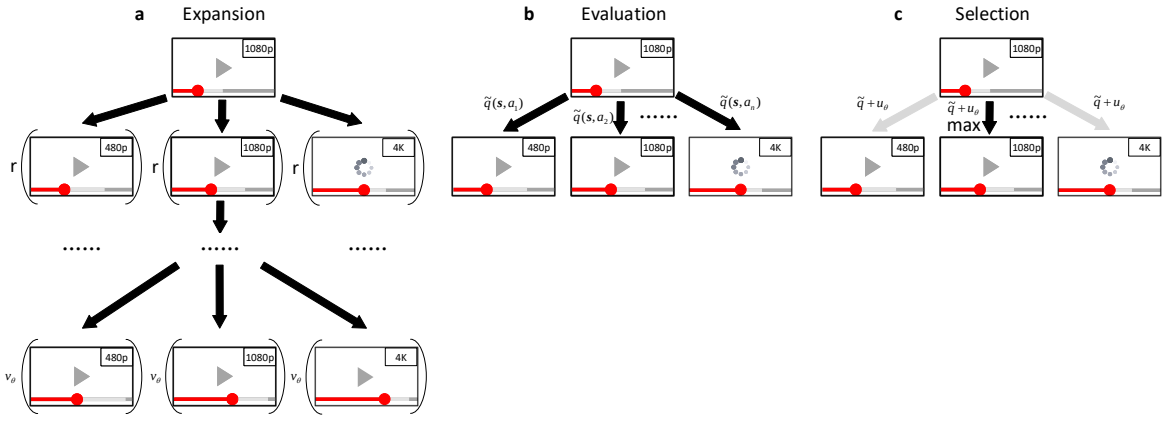


Figure 6.2: The proposed EERO bitrate selection algorithm. (a) The ABR agent traverses the tree and collects the instantaneous reward along the path. At each terminal node of the look-ahead horizon, EERO may optionally estimate the future cumulative reward with a value function. (b) The algorithm estimates the optimal cumulative reward function $\tilde{q}$ based on the simulation. (c) In the end, EERO combines the observation at the test time $\tilde{q}$ and the prior estimate $\mu_{\boldsymbol{\theta}}$.

On the other hand, in case that the player encounters an unobserved state-action pair, the proposed method would prioritize the bitrate decision provided by the likelihood function. We will introduce the specific choice of likelihood function and prior distribution in the subsequent section.

6.2.1 Likelihood Function

The likelihood function in the framework is responsible to produce an estimate of the mean action-value μ on the fly, generally by sampling the action-value distribution $p(q^*|\mathbf{s}, a)$ in situ. An accurate sample from the distribution may be obtained by recursively computing the optimal value function in a search tree containing approximately $|\mathcal{A}|^T$ bitrate selection trajectories, where $|\mathcal{A}|$ and T represent the number of bitrate levels and the number of chunks in a streaming video, respectively. In a typical streaming session, $|\mathcal{A}| \approx 10$ and

$T \approx 150$, thus exhaustive search is infeasible. How to approximately sample from the distribution in a limited time budget is crucial to the design of the likelihood function.

The most prevalent solution is to apply dynamic programming in a short decision window, essentially discarding the cumulative reward falling out of the horizon. However, the truncated optimization strategy often lead to short-sighted bitrate selections that are far from global optimal [131]. The deficiency exists because the unbalanced state value (*i.e.*, the expected cumulative reward) at the leaf nodes of the decision window. To address this problem, we propose to account for the cumulative reward below the leaf node $\mathbf{s}$ by an parametric value function $v(\mathbf{s}; \boldsymbol{\theta})$ that predicts the optimal outcome from the state. Formally, the overall cumulative reward is approximated by

$$\sum_{t=0}^{L} \gamma^t r(\mathbf{s}_t, a_t^*) \approx \sum_{t=0}^{K} \gamma^t r(\mathbf{s}_t, a_t^*) + \gamma^{K+1} v(\mathbf{s}_{K+1}; \boldsymbol{\theta}), \tag{6.17}$$

where a_t^* is the optimal action at the state $\mathbf{s}_t$ and

$$v(\mathbf{s}_{K+1}; \boldsymbol{\theta}) = \max_a \mu(\mathbf{s}_{K+1}, a; \boldsymbol{\theta}). \tag{6.18}$$

The mean action-value function can be obtained by any appropriate prior models, which will be introduced shortly. The schematic diagram of the algorithm is illustrated in Figure 6.2. We employ a state-of-the-art throughput predictor TTP [237] to model the state transition probability distribution required by the dynamic programming, although it is also possible to apply other throughput predictors.

6.2.2 Prior Model

The prior model $p(\mu)$ encodes our prior belief of the cumulative reward function for each state and action. Although it is possible to engineer subjective prior [63, 93, 102, 141] about the overall goodness of an action, the handcrafted adaptation rules may deviate significantly from the long-term optimal objective [131, 241]. Another drawback of these priors is that they make strong assumption about the reward function and throughput dynamics, which by themselves are subjects of ongoing research. If a novel reward function or a better throughput predictor become available in the future, the expert knowledge of

$p(q^*|\mathbf{s}, a, \boldsymbol{\theta})$ can be difficult to acquire. In this study, we explore a data-driven method to learn an objective prior model, tabula rasa. However, the proposed framework is general enough to incorporate other prior models. We will discuss the issue in the subsequent analysis.

We incorporate deep reinforcement learning to train a prior action-value model. Instead of utilizing off-the-shelf reinforcement learning-based algorithms that either lack the compatibility with the proposed framework (*e.g.*, Pensieve discards the action-value and directly produces the optimal action) or learn a tabular Q function for each sequence without any generalization [32, 33, 212, 241], we apply the deep Q-learning algorithm [138] to train a neural network-based prior model, namely Deep Q-Network (DQN). The basic idea of deep Q-learning is to integrate value iteration of the expert strategy into the training process. The neural network iteratively improves the cumulative reward estimation based on its previous iteration by solving the optimization problem in (6.11). Differentiating the loss function with respect to the weights we arrive at the following gradient

$$\nabla_{\boldsymbol{\theta}_i} L_i(\boldsymbol{\theta}_i) = \mathbb{E}_{\mathbf{s},a\sim\rho(\cdot);\mathbf{s}'\sim\mathcal{E}}\Big[\Big(u + \gamma \max_{a'} \mu(\mathbf{s}', a'; \boldsymbol{\theta}_{i-1}) - \mu(\mathbf{s}, a; \boldsymbol{\theta}_i)\Big)\nabla_{\boldsymbol{\theta}_i}\mu(\mathbf{s}, a; \boldsymbol{\theta}_i)\Big]. \quad (6.19)$$

Rather than computing the full expectations in the above gradient, it is computationally expedient to approximate the gradient using stochastic gradient estimation computed over a batch of throughput traces and streaming videos. To reduce the variance of the gradient, a common strategy is to utilize the simulation results from previous iterations, which is known as experience replay.

Combing these machine learning techniques, we summarize the training process of deep Q-learning-based prior adaptation logic as follows. The algorithm starts by random initializing the DQN and clearing the dataset $\mathcal{D}_{\mathbf{q}}$ of capacity $N_{\mathbf{q}}$. At each iteration, the trainer randomly samples a batch of throughput traces and streaming videos, which gives us the state variable $\mathbf{s}_t$. For each pair of throughput trace and streaming video, we select a random action a_t with a fixed probability of ϵ. Otherwise, the DQN takes the current state of the player $\mathbf{s}$ and the feasible actions a as input and produces an estimate of the cumulative reward, based on which an optimal action a_t is generated using Equation (6.7). The ABR player emulator then executes the action a_t and returns the instantaneous reward u_t and the next state $\mathbf{s}_t$. The observed data tuple ($\mathbf{s}_t$, a_t, u_t, $\mathbf{s}_{t+1}$) is appended into the

dataset $\mathcal{D}_{\mathbf{q}}$. Depending on the capacity of $\mathcal{D}_{\mathbf{q}}$, the learning algorithm may remove the least recent observation from the experience buffer to discourage less favorable actions. Based on the instantaneous reward function, we update the target Q-value using value iteration in (6.2). At the end of each iteration, we perform a gradient descent step on the DQN with (6.19). The training cycle is repeated till model convergence.

6.2.3 Implementation Details

Input: The likelihood function and prior model in the BBS framework are executed in parallel. Both modules take three pieces of information as the state variable to inference Q-values, including

- Past Throughput Observations: The algorithm represents the network condition with the previous throughput observations $\mathbf{c}_{t-H:t-1}$ for the past H time instances. Inherited from Pensieve, the algorithm has a default history window of $H = 8$.
- Video Content Characteristic: To accommodate the highly variable bitrate ladders, we propose a generic representation of the encoded video streams. Specifically, we quantize the log bitrate space into $N = 30$ bins. The available bitrate levels for a specific streaming video is 0-1 encoded by a feature vector, such that a compressed version lying in the corresponding bitrate interval is labeled as 1 and 0 otherwise. By using this feature embedding, we have implicitly made use of the prior knowledge that a standard encoding profile would not include two similar representations lying in one bitrate bin. In case that the assumption is violated, one can extend the dimension of the feature vector by incorporating other encoding parameters such as spatial resolution, frame rate, and bit depth. When using BSQI as the reward function, we also include the VMAF score of each chunk to account for the heterogeneity of the video complexity. The video quality scores are encoded in a similar fashion. We consider the rate-distortion features of the next $K = 5$ chunks to represent the video content. In the end, we extract 300 features to represent the video content characteristics.

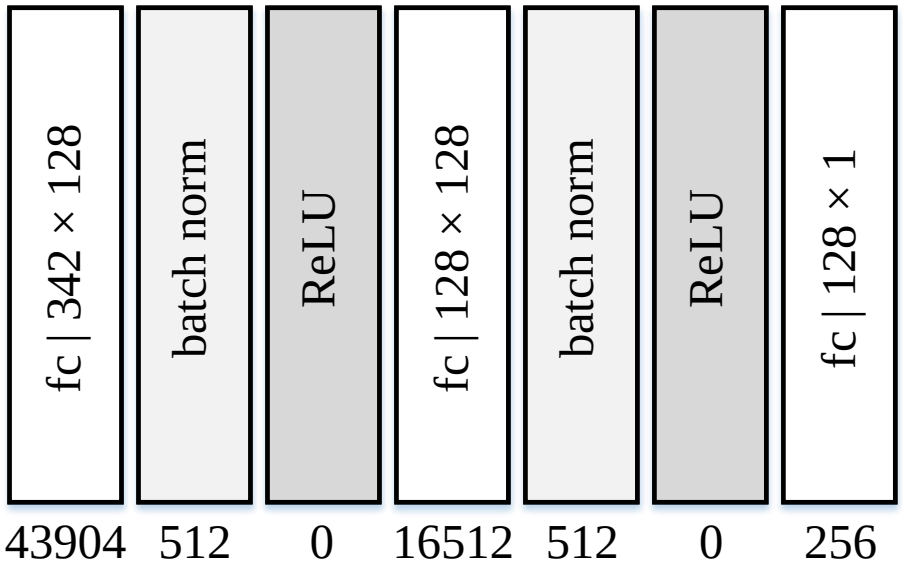


Figure 6.3: Illustration of the neural network configurations for EERO. We denote the parameterization of the fully connected "fc" layer as "input channel × output channel".

- Video Player Status: The last relevant feature for describing the ABR players' state is the current video playback status. The status is represented by the previous chunk selection (including both the bitrate and quality), the current buffer occupancy, and the percentage of downloaded videos. This information source provides a 4-dimensional vector.

Each action is represented by a one-hot vector adapted from the log bitrate representation described in the video content characteristic feature extraction. In total, our instantiation of EERO represents each state-action pair with a 342-dimensional vector.

Network Architecture: We now describe the the exact architecture used for prior cumulative reward function model. Since we do not have any prior knowledge about the form of Q-function (such as translation in-variance), we leverage a MLP as the function approximator. The regression model consists of 2 hidden layers of size 128, followed by

batch normalization [98], and a rectified linear unit non-linearity [145]. The output layer is a fully-connected linear layer with a single output variable. The network architecture and the number of parameters in each layer is depicted in Figure 6.3.

Reward Functions: It has been widely accepted that the ultimate goal of ABR algorithm is to optimize end viewers' QoE. The definition of QoE is a subject of ongoing research. Thus, we consider a wide range of objective QoE measures from heuristic linear bitrate model to state-of-the-art Bayesian VQA-based model. All these models decompose the overall QoE into chunk-level instantaneous reward u_t by taking the temporal homogeneous assumption. Mathematically, the general form of these QoE models can be expressed by

$$u = \frac{1}{T}\sum_{t=1}^{T} U(p_t, \tau_t, \Delta p_t), \tag{6.20}$$

where p_t, τ_t, and Δp_t represent the bitrate, the rebuffering duration, and the bitrate variation of the t-th chunk, respectively. We consider three choices of U as follows.

1. The linear bitrate model takes the form $U(r_t, \tau_t, \Delta p_t) = r_t - \alpha\ \tau_t - \beta\ |r_t - r_{t-1}|$, where α and β are model parameters. Albeit its limited correlation with subjective QoE ratings, the model proposed in [241] has received nearly ubiquitous acceptance in the field of ABR [131, 200]. We optimize the two free parameters on WaterlooSQoE-I and WaterlooSQoE-II such that the model can maximally explain the subjective QoE data.

2. Another commonly used QoE measure is the logarithmic bitrate model that was used by BOLA [191]. This model captures the phenomenon that the marginal gain in QoE decreases with respect to the bitrate. Some recent studies have demonstrated that this simple change leads to a significant improvement in the prediction accuracy [46, 53]. The log bitrate model can be expressed by $U(r_t, \tau_t, \Delta p_t) = \log r_t - \alpha\ \tau_t - \beta\ \log |r_t - r_{t-1}|$. Similarly, we optimize the free parameters on WaterlooSQoE-I and WaterlooSQoE-II using linear regression.

3. Last, we employ BSQI as the objective QoE measure. The form of instantaneous QoE function $U(r_t, \tau_t, \Delta p_t)$ is learnt from data. In contrast to the two QoE models above,

the presentation video quality measure is adapted to the streaming video complexity and viewing condition. Meanwhile, the method imposes several constraints that are derived from subjective QoE assessment studies to guarantee that the model is consistent with certain HVS properties. For fairness, the model is also optimized on WaterlooSQoE-I and WaterlooSQoE-II. Interested readers may refer to Chapter 4 for more details.

Learning Algorithm Instantiation: The learning of the DQN adopts the Adam optimization algorithm [107] with a batch size of $3,200$. We start with a learning rate of 10^{-2} and subsequently lower it by a factor of 10 when the loss plateaus, until 10^{-4}. The discount factor γ is set to 0.99, essentially covering the rewards of 100 decision steps. We adopt 32 agents to asynchronously collect training data, while the back propagation is performed synchronously [41]. To accelerate the training process, we utilize the chunk-level streaming simulator in [131] for its demonstrated efficacy. We train our model for $100,000$ epoches till convergence. Our reinforcement learning code is implemented in PyTorch [160].

The winner-take-all decision process implies that the resulting policy is invariant to the translation and scaling of the Q-function. As a result, we can define the standard deviation of the action-value distribution $\sigma = 1$ in (6.15) without loss of generality. We follow the approach in [187] to model the uncertainty of a Q-value σ_μ with its visit frequency. Specifically, we set $\sigma_\mu = \frac{1}{N(\mathbf{s},a)^\alpha}$, where $N(\mathbf{s}, a)$ and α represent the visit count of the state action pair in the training process and a temperature parameter. We implement $N(\mathbf{s}, a)$ as a lookup table, motivated by the sparsity of observed training samples in the input space. In accordance to the recommendation in [241], we quantize the state-action space using 100 bins for buffer occupancy, 100 bins for throughput prediction, and 30 bins for bitrate level. Additionally, we include 10 bins to represent video quality, whose range is $[0, 100]$. We empirically find that these parameters work reasonably well across a wide range of settings. Thanks to the sparsity of training sample, we can efficiently encode the visit count with a sparse tensor such that the never-encountered entries do not have to be recorded. The sparse tensor only takes around 60 kilobytes, resulting in insignificant storage overhead in the modern streaming video receivers.

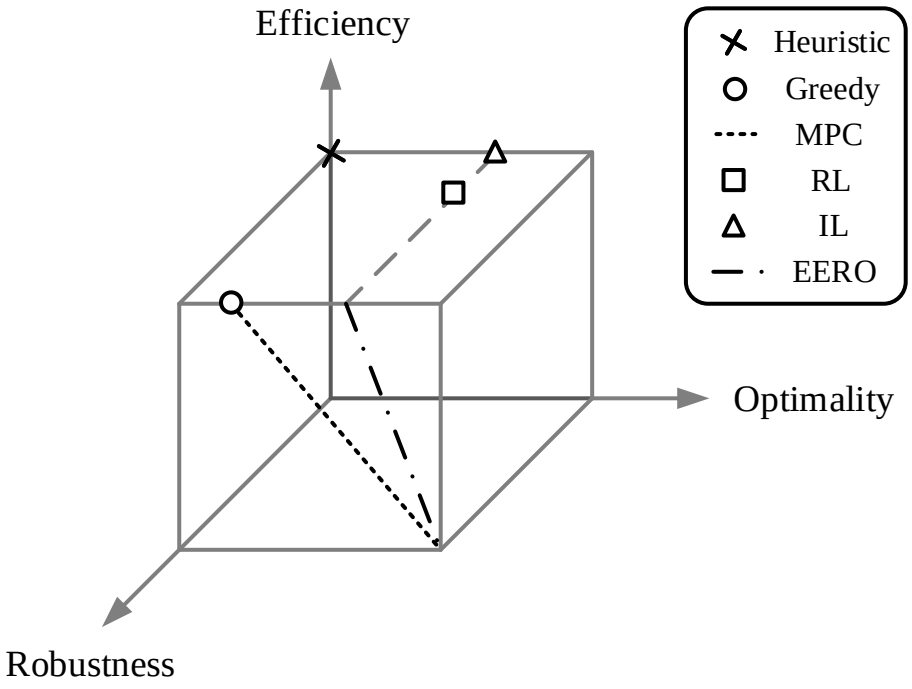


Figure 6.4: The schematic diagram of the efficiency-robustness-optimality tradeoff of bitrate selectors. EERO integrates the robustness of MPC and the optimality of reinforcement learning-based algorithms, thereby achieving a better efficiency-robustness-optimality tradeoff. Abbreviations: MPC, model predictive control; RL, reinforcement learning; IL, imitation learning.

6.2.4 Why EERO?

To further motivate the proposed algorithm EERO, we revisit the design space of ABR algorithms. Conceptually speaking, all bitrate adaptation logic lie in a three-dimensional space as shown in Figure 6.4. An ideal adaptation algorithm should be efficient in computation time, robust to unobserved state-action space, and close to global optimal solution in standard streaming environment. However, existing ABR algorithms often neglect one

or two of these dimensions. For example, heuristic approaches are computationally efficient due to the reduced action space. However, they often rely on unrealistic assumptions and overly simplified optimization algorithms, suggesting that the algorithms exhibit low robustness and optimality. MPC is quite robust to different bitrate ladders and network dynamics [237], partly because it does not make any prior assumption about the action-value distribution. The MPC framework trades off efficiency with optimality by adjusting the look-ahead horizon K. When $K = 1$, MPC degrades gracefully to the greedy algorithm. Learning-based approaches generally employ a function approximator to estimate the optimal decision produced by MPC with K approaching T. The resulting agent is responsive in computation and nearly optimal in an experienced environment. The augmented data generated from the agent-environment interaction process in reinforcement learning acts as a regularizer and helps reduce overfitting comparing to the imitation learning algorithms. Unfortunately, these empirical Bayes-based algorithms do not always deliver equally competitive performance, especially on unobserved bitrate ladders and never/hardly observed throughput conditions. More generally, learning-based algorithms by themselves are insufficient to overcome the generalization problem in a complex environment such as adaptive streaming [187]. To hit a sweet spot in the efficiency-robustness-optimality space, the proposed algorithm combines the optimality of learning-based models and the robustness of MPC in a unified framework. Furthermore, these two policies can be executed in parallel on dedicated hardware, making the resulting algorithm computationally efficient.

6.3 Evaluation

In this section, we evaluate the performance of EERO with trace-driven experiments. The goal of our experimental evaluation is to answer the following questions:

1. How does EERO compare with existing ABR algorithms? We find that given a fixed time budget, EERO performs at least on par with the best existing technique in a broad range of scenario, with average improvements of 7%-22%.

2. Where does the improvement come from? We conduct a series of ablation experiments to identify the contributions of the core factors in EERO. We show that although

a better likelihood function can moderately enhance the performance, the major improvement of EERO comes from the Bayesian Q-function approximation.

3. How do hyper-parameters such as neural network architecture and quantization level affect EERO? Our experiment suggests that the performance of EERO in terms of reward function is rather insensitive to these parameters.

4. What should be considered in the practical deployment of EERO? We analyze various versions of EERO calibrated with different training time, training data, and environmental conditions, based on which we shed light on the practical concerns with using EERO in practical adaptive streaming systems.

6.3.1 EERO vs. Existing Bitrate Selectors

We first describe the simulation setups including a streaming video database, network traces, viewing conditions, and evaluation criteria. We then compare EERO with classic and state-of-the-art ABR algorithms.

Experimental Setup

Video Dataset: We construct a new video database which contains 15 high quality 4K videos that span a great diversity of video contents. To make sure that the videos are of pristine quality, we carefully inspect each of the videos multiple times and remove those videos with visible distortions. The duration of the videos ranges from 3 minutes to 40 minutes, with an average duration of 7 minutes. Using the aforementioned sequences as the source, each video is distorted by the following process sequentially to accommodate constantly progressing encoding specifications.

- Spatial down-sampling: We down-sample the source video using the bi-cubic filter to six spatial resolutions (3840×2160, 2560×1440, 1920×1080, 1280×720, 720×480, 640×360) according to YouTube recommendation [242].

Table 6.1: Encoding Ladder of Video Sequences

Index	Resolution	Bitrate	Index	Resolution	Bitrate
1	640×360	300 Kb/s	8	1280×720	3000 Kb/s
2	640×360	375 Kb/s	9	1920×1080	4300 Kb/s
3	640×360	560 Kb/s	10	1920×1080	5800 Kb/s
4	640×360	750 Kb/s	11	2560×1440	8100 Kb/s
5	640×360	1050 Kb/s	12	3840×2160	11600 Kb/s
6	720×480	1750 Kb/s	13	3840×2160	16800 Kb/s
7	1280×720	2350 Kb/s			

- Compression: We encode the down-sampled sequences using three commonly used video encoders, *i.e.*, H.264, HEVC, and AV1, with two-pass encoding [36, 76, 110]. We uniformly sample 30 target bitrate on the interval 300 Kbps to 40 Mbps in the log bitrate space. The full encoding specification is detailed in Appendix B.1.

In total, we obtain 180 representations for each streaming video. In our experiment, we include four bitrate ladders that are widely used in practical video delivery systems. The first bitrate ladder in Table 6.1 is a combination of the Netflix's recommendation [147] and Apple's recommendation [95]. This encoding profile is fixed across all streaming videos. We also include three per-title encoding schemes that are described in [36, 48, 208]. The bitrate-centric encoding strategy [35] selects bitrate-resolution pairs such that i) At a given bitrate, the produced encode should have as high quality as possible, and ii) The perceptual difference between two adjacent bitrates should fall just below one just-noticeable different (the difference in VMAF $\approx$ 10). The quality-centric encoding strategy [48] pre-defines 10 target quality levels, and exhaustively searches the constant target quality contour to obtain the representation with the minimal bitrate. The per-title optimization algorithm in [208] determines the encoding strategy by optimizing the overall quality of the encoded representations subject to some constraints on resources such as server storage and throughput capacity. We segment the test sequences with GPAC's MP4Box [112] with a segment length of 4 seconds. We split the streaming video corpus into three datasets. We first exclude one of the encoding profiles from the compilation, and denote the dataset as

$\mathcal{D}_R$. The removed dataset $\mathcal{D}_R$ is used in the robustness analysis, due to its unique characteristics. We randomly split the remaining videos into 80% training set $\mathcal{D}_T$ and 20% testing set $\mathcal{D}_O$. This leave-one-out data segmentation scheme is repeated for all dataset, such that each encoding strategy would appear in $\mathcal{D}_R$ once.

Network Traces: To train and evaluate the ABR algorithms in realistic network conditions, we created a corpus of network traces by combining several public datasets: a broadband dataset collected by FCC [60], two 4G datasets from UCC [167] and Belgium [213], a 5G dataset from UCC [166], and a Live Television streaming dataset named Puffer [237]. The FCC dataset contains more than 1 million throughput traces, each of which records the average throughput over $2,100$ seconds at a granularity of 5 seconds. Each trace is associated with a unique connection identifier. We select $10,000$ sessions from 500 clients by randomly cutting from the raw connection-level throughput traces, each with a duration of 120 seconds. The Belgium dataset consists of 40 4G bandwidth traces recorded along several routes in and around the city of Ghent at a 1-second granularity. The UCC dataset is composed of client-side cellular key performance indicators collected across different mobility patterns (static, pedestrian, car, tram and train). The 4G trace dataset contains 135 traces, with an average duration of fifteen minutes per trace at a granularity of one sample per second. The dataset also contains synthetic throughput traces from 100 mobile users. We consider each route corresponds to a network environment, based on which the connection-level throughput traces are extracted. To match the duration of our selected FCC traces, we generate $10,000$ traces using a sliding window across the two 4G datasets, each with a duration of 120 seconds. The 5G dataset comprises 45 traces collected from 15 environmental conditions. We follow a similar way to pre-process the data. The Puffer dataset contains the data collected from January 2019 and we use the data from June 2019 to May 2021 in our experiment. The Puffer dataset does not provide the throughput at a fixed granularity. To make the data format consistent, we apply linear interpolation to the source data such that the a throughput measurement is recorded every two seconds. Each trace in the Puffer dataset is assigned a SessionID, and throughput traces with the identical SessionID are generated from the same connection. We cut every raw trace into 120-second sequences and randomly select 20 traces for every user. In the end, we randomly select $10,000$ traces from 500 users. We again segment the throughput corpus into three

datasets based on the following procedure. First, we uniformly sample 80% users from each throughput dataset. For each user in the set, we randomly split the connection-level traces into 80% training set $\mathcal{D}_T$ and 20% optimality test set $\mathcal{D}_O$. The traces from the remaining 20% users form the robustness test set $\mathcal{D}_R$. We apply five-fold cross validation and report the median performance.

Bitrate Selectors: Exhaustive evaluation of all ABR algorithms is difficult as it involves optimizing over an infinite-dimensional functional space. To this end, we evaluate the following ABR algorithms, ranging from the naïve de facto rate-based algorithm to the state-of-the-art reinforcement learning algorithms, and optimize the free parameters by empirical simulations on the training dataset:

- RB algorithm [63] employs a variant of greedy algorithm to optimize a linear bitrate-based reward function. The available future throughput is predicted by the arithmetic mean of observed throughput over the past five chunks. The algorithm makes a conservative bitrate decision without using the buffer occupancy status.

- BB algorithm [93] is another representative greedy bitrate selection algorithm which determines the optimal action based on the current buffer occupancy. We employed the function suggested in [93], where bitrate is chosen as a piece-wise linear function of buffer occupancy. We set a lower reservoir and cushion to be 5 and 10 seconds, respectively.

- BOLA [191] makes greedy optimization to maximize a reward function that encourages higher bitrate, shorter rebuffering duration, and lower buffer occupancy. Under some mild assumptions, the problem can be solved by Lyapunov optimization, which select optimal bitrates solely based on buffer occupancy observations.

- FastMPC [241] uses both buffer occupancy observations and throughput predictions using harmonic mean of the past 5 chunks to select bitrate. In each step, the algorithm maximizes a reward function over a horizon of five future chunks, and re-plans the trajectory as new state variables become available. The optimization problem is solved offline and its solution is stored as a lookup table. We use 100 bins for

throughput prediction, 100 bins for buffer level, and 30 bins for the past bitrate level. We train a lookup table for each reward function.

- Pensieve [131] is a model-free reinforcement learning-based ABR algorithm that learns to select optimal bitrate from scratch. The algorithm learns a CNN-based policy that takes throughput observations and buffer occupancy of previous eight chunks as input and produces the optimal bitrate decision. In our experiment, we optimized three versions of Pensieve to adapt to different notions of QoE. One of the biggest advantages of the algorithm is that it can efficiently optimize the long-term reward.

- Fugu [237] is a close variant of MPC [241]. This algorithm employs a data-driven download time prediction model to estimate the cumulative reward in the next 5 steps on the fly. Based on the noisy sample from the action-value distribution, the algorithm selects the action with the highest cumulative reward.

- Comyco [92] is an imitation learning-based offline ABR algorithm. This method first applies dynamic programming over a look-ahead horizon of 8 to obtain a dataset of training samples. Ingesting the state-action pairs as the target input-output variables, a neural network policy model is then optimized to match the offline optimal solution.

- For comparison, we also present results for the oracle scheme, which is computed using dynamic programming with complete future throughput information. This offline optimal serves as an (unattainable) upper bound on the reward that an omniscient policy with complete and perfect knowledge of the future network throughput could achieve.

Viewing Devices: The ultimate receiver of streaming videos are human beings, who consume multimedia on a large variety of viewing devices. In this study, we consider three mostly used viewing devices according to [104], including Full High Resolution (FHD) monitor, smartphone, and UHDTV. Note that the presentation quality is a function of viewing device, which we take into account with device adaptive presentation QoE scores.

Evaluation Criteria: In this chapter, we are interested not only in how well does EERO perform in a streaming environment that the agent has experienced during the training process, but also in its performance on unobserved state action pairs resulted from a distribution shift. We consider three evaluation criteria, which are

- Efficiency: We define efficiency as the computation time required for each bitrate adaptation, including resource estimation, action-value distribution sampling, prior Q-value inference, and action selection. This measure is computed as

$$E = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{L_i}\sum_{j=1}^{L_i}(t_{ij}^{t} - t_{ij}^{s}), \tag{6.21}$$

  where N, L_i, t_{ij}^{t} and t_{ij}^{s} denote the number of streaming sessions in a test set, the number of chunks in the i-th streaming session, the time instance that the j-th chunk request in the i-th streaming session is sent out, and the time instance when this specific bitrate selection process starts. A more efficient ABR algorithm would exhibit a less computation time.

- Optimality: The optimality is defined as the average cumulative reward that an ABR algorithm receives on the test set $\mathcal{D}_O$, where the test set $\mathcal{D}_O$ and the training set come from the same distribution (*i.e.*, these two datasets share the same encoding strategy and the source of throughput traces). Mathematically, the optimality measure can be summarized by

$$O = \frac{1}{|\mathcal{D}_O|}\sum_{i=1}^{|\mathcal{D}_O|}\frac{1}{L_i}\sum_{j=1}^{L_i}u_j(r_l, \tau_l, \Delta r_l), \tag{6.22}$$

  where $|\mathcal{D}_O|$ represent the number of streaming sessions in the dataset. Note that we normalize the cumulative reward by the number of chunks to eliminate the impact of video duration.

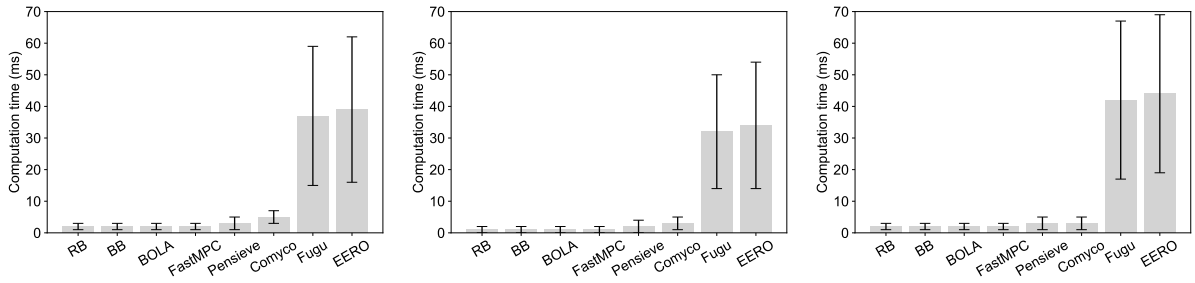
- Robustness: The robustness measure is defined in a similar fashion to the optimality, but is evaluated on a test set $\mathcal{D}_R$ with different characteristics. Specifically, the robustness can be mathematically expressed as

$$R = \frac{1}{|\mathcal{D}_R|}\sum_{i=1}^{|\mathcal{D}_R|}\frac{1}{L_i}\sum_{j=1}^{L_i}u_j(r_l, \tau_l, \Delta r_l), \tag{6.23}$$

where the test set $\mathcal{D}_R$ comprises different streaming videos, encoding profiles, and network traces to the training set.

Evaluation Framework: To evaluate the algorithm efficiency in a realistic setting, we perform a real-player emulation. Specifically, we randomly sample $N = 1,000$ streaming video and throughput trace pairs from $|\mathcal{D}_O|$ to form the test set. These test videos are stored on a Apache web server. Meanwhile, the VMAF scores of each video are embedded in the corresponding manifest file such that the player has the access to the chunk-level video quality. We implement test ABR algorithms in dash.js (version 3.2.2) [63], which are deployed on three client devices, including iPhone 12, iPad Pro, and a desktop with an Intel i7-6900K 3.2GHz CPU. For FastMPC, we compress the javascript code directly instead of performing run-length coding on the lookup table. We find that the simplification introduces minimum overhead and the code size is close to the original implementation [241]. To perform feed-forward prediction in the browser, we convert the neural network models of Pensieve, Fugu, Comyco, and EERO to Tensorflow.js [97] and save the models in the client local storage via IndexedDB [143]. The client video player is a customized Chromium browser (version 91) supporting H.264, HEVC, and AV1 playback. The server runs on the a computer with an Intel i7-6900K 3.2GHz CPU. To minimize the variance introduced by manual operation, we develop a customized program in Python to automate the video streaming process. After each video streaming session, a log file is generated on the client device, including the start and finish time of each bitrate decision process.

We employ a PyTorch [160] implementation of ABR algorithms in the optimality and robustness experiments. Given the immense state space, it is prohibitively expensive to perform an exhaustive evaluation of ABR algorithms in a realistic setting. Specifically, it would take more than $1,000$ years to download all possible combinations of streaming videos in a real-player emulation setup. Nevertheless, we can leverage the chunk-level simulator in [131] to evaluate the optimality and robustness, which has been previously demonstrated to faithfully model the application layer network. For each chunk download, the PyTorch ABR agent takes the current state and available bitrate levels as input and send an action to the environment. The simulated environment assigns a download time that is solely based on the chunk's bitrate and the input network throughput trace. The chunk-level simulator carefully keeps track of the player states such as buffer occupancy



(a) Desktop (b) Tablet (c) Phone

Figure 6.5: Computation time of ABR algorithms.

and download history. After each chunk download, the simulator passes several state observations to the ABR agent and records the instantaneous reward. In our objective evaluation, the streaming simulator is configured to emulate the network conditions from our corpus of network traces, along with an 80 ms RTT, between the client and server.

Experimental Results

Efficiency: Figure 6.5 shows the average computation time of the 8 test ABR algorithms. We summarize the key observations as follows. First, EERO does not introduce notable computation cost to its base modules. Empowered by parallel computation, the algorithm is as good as the least efficient computation between dynamic programming and feed-forward inference in all scenario considered. Second, thanks to the dedicated neural engine [96], the average computation time of EERO can be kept below 50 ms even on mobile devices. Our client-side implementation is more efficient to the server-end deployment in Pensieve, which incurs extra time for data transmission. Recent studies have illustrated that such little latency has negligible impact on the cumulative reward [131, 237].

Optimality: Figure 6.6 shows the average reward that each scheme achieves on our test set $\mathcal{D}_O$. We provide more detailed results in the form of full CDFs on different encoding profiles and network environment in Figure 6.7 and Figure 6.8, respectively. There are two key takeaways from these results. First, we find that EERO exceeds the performance of the best existing ABR algorithm with a sizable margin on all scenarios considered. Two close

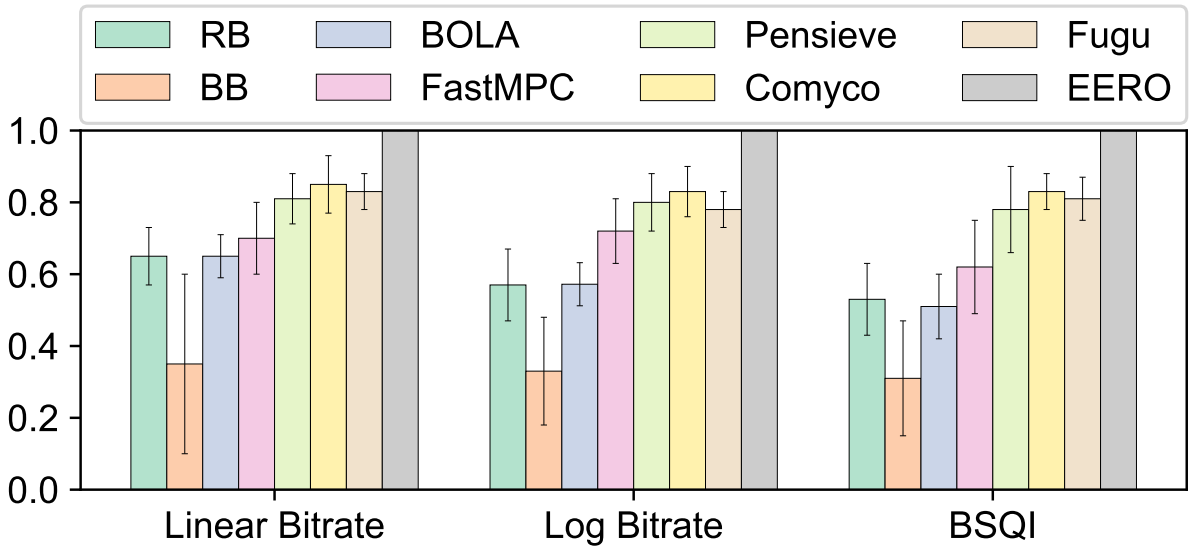
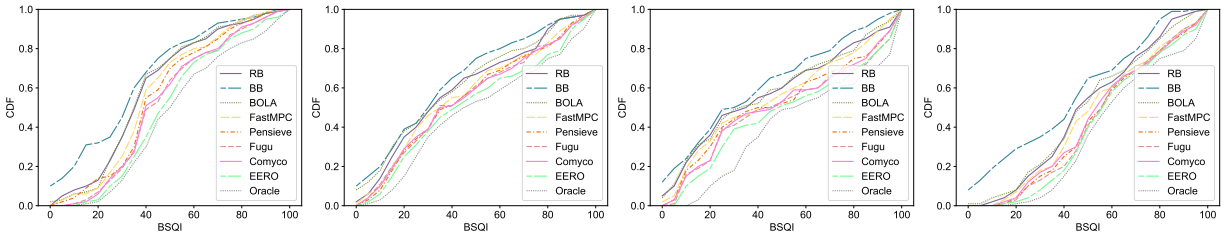


Figure 6.6: The optimality performance of EERO and existing ABR algorithms in terms of three QoE measures. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average.

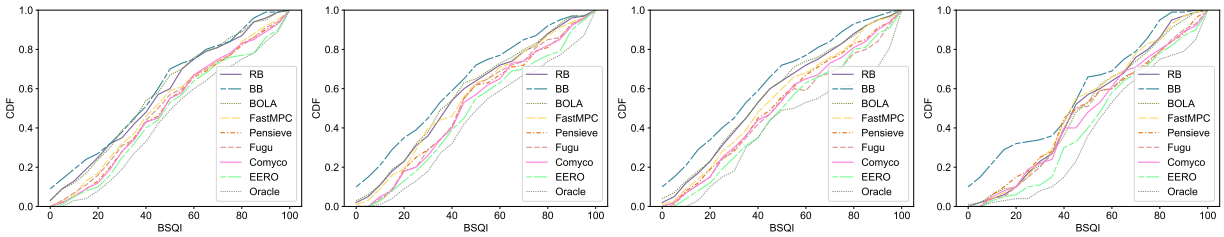
competing algorithms are Fugu and Comyco. Each of these schemes has a state-action region in which it outperforms its competitor. For instance, Comyco achieves a higher reward on the "one-size-fits-all" Netflix bitrate ladder, while Fugu has an edge on streaming video adaptive encoding profiles. Thanks to the Bayesian framework, EERO consistently delivers the highest reward by combining the best of two approaches. On average, EERO outperforms Comyco by 17% in terms of linear bitrate reward. The performance gap expands to 18% and 22% for the log bitrate and BSQI, respectively. We observe a similar results for Fugu.

Second, EERO's performance is within 2.5%-9.2% of the offline optimal solution across all streaming videos in $\mathcal{D}_O$. The offline optimal is unattainable because it requires the complete knowledge about future throughput, while throughput dynamics in practice is a stochastic process in nature. This suggests that there is little room for improvement in an environment the agent has experienced.



(a) FCC (b) 4G (c) 5G (d) Puffer

Figure 6.7: The optimality score of ABR algorithms on different throughput datasets. Results are given in terms of BSQI.



(a) Netflix (b) Bitrate-oriented (c) Quality-oriented (d) Resource-constrained

Figure 6.8: The optimality score of ABR algorithms with different encoding profiles. Results are given in terms of BSQI.

Robustness: Figure 6.9 demonstrates the average reward that each scheme exhibits on the test set $\mathcal{D}_R$. Figure 6.10 and Figure 6.11 illustrate more detailed results on different encoding profiles and network environment, from which we draw three observations. First, EERO again rivals or outperforms the existing top performer with across a wide range of QoE models, throughput traces, and streaming videos. The best existing ABR algorithm is Fugu, which adapts reasonably well to different bitrate ladders. Despite working in a relatively unfamiliar environment (more than 80% state-action pairs has not been observed in $\mathcal{D}_R$), EERO still improves the noisy reward observation by using the learnt prior. As a result, EERO outperforms Fugu by 7%, 10%, and 11% in linear bitrate, log bitrate, and BSQI, respectively. However, the improvement is less significant comparing to the performance gain on $\mathcal{D}_O$.

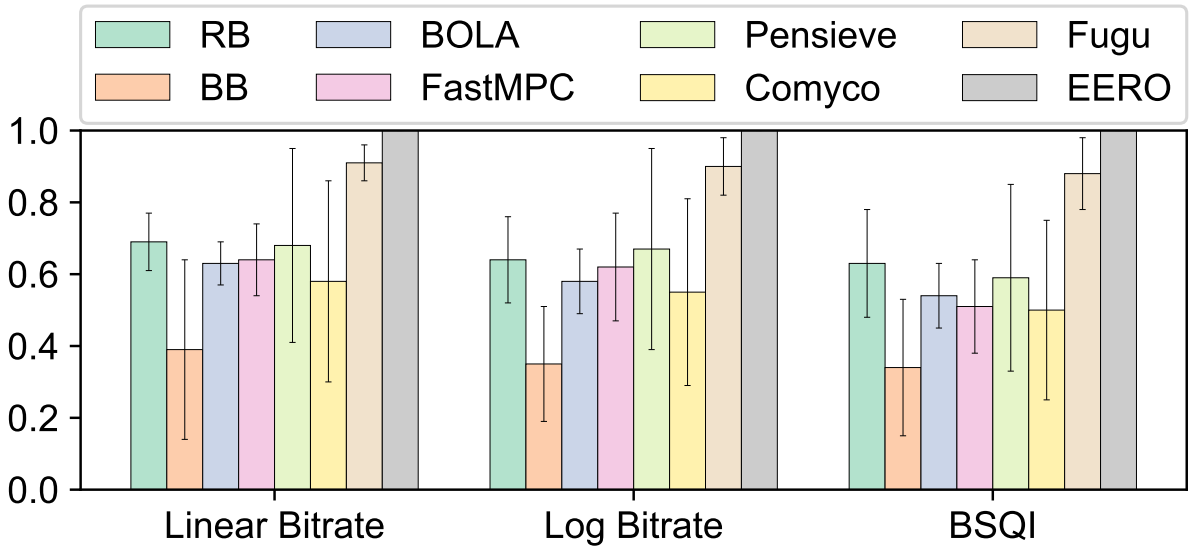
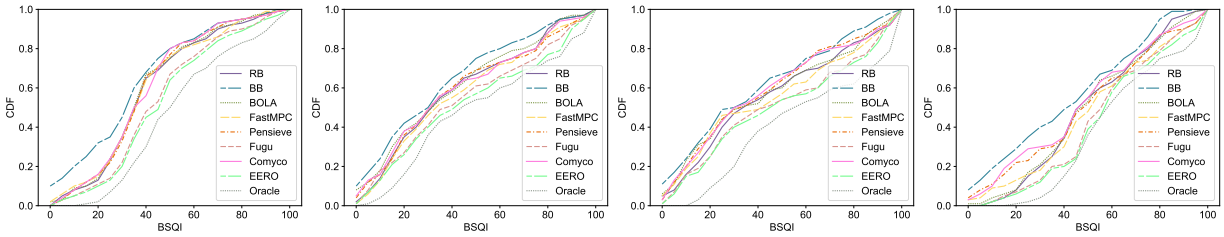


Figure 6.9: The robustness performance of EERO and existing ABR algorithms in terms of three QoE measures. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average.

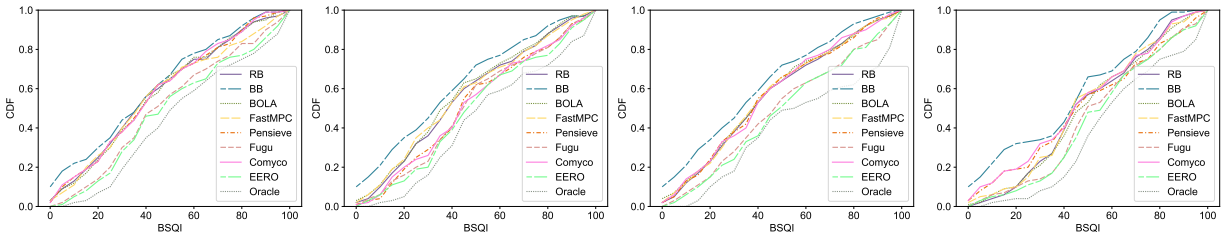
Second, prior-based ABR algorithms such as Pensieve and Comyco incur a huge performance degradation. Somewhat surprisingly, these state-of-the-art algorithms are inferior to the de facto rate-based algorithm in terms of robustness. In particular, Comyco experiences a ∼30% loss in the cumulative reward when tested on the unobserved state-action space, suggesting that the offline data-driven model suffers from severe overfitting problem. This issue is less prominent in ABR algorithms that sample the action-value distribution on the fly. Nevertheless, Fugu suffers 5%, 7%, and 7% drop from the optimality test results in terms of linear bitrate, log bitrate, and BSQI, respectively. The phenomenon motivates the development of more robust throughput prediction models.

Third, although EERO achieves the best robustness score in the experiment, the performance gap between EERO and the oracle widens to 13% in terms of BSQI. The result suggests that there is still room for improvement in the robustness of ABR algorithms. We believe that future ABR algorithms may improve the robustness from two aspects.



(a) FCC (b) 4G (c) 5G (d) Puffer

Figure 6.10: The robustness score of ABR algorithms on different throughput datasets. Results are given in terms of BSQI.



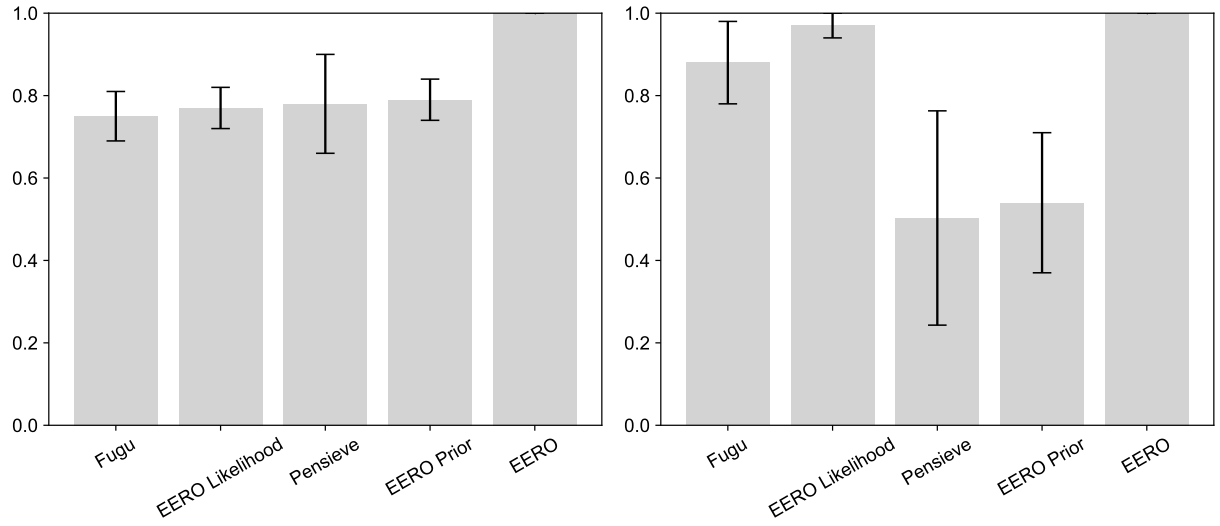
(a) Netflix (b) Bitrate-oriented (c) Quality-oriented (d) Resource-constrained

Figure 6.11: The robustness score of ABR algorithms with different encoding profiles. Results are given in terms of BSQI.

First, one may improve the accuracy of throughput prediction model by adapting it to connection-level throughput characteristics. Second, more efficient sampling scheme may be derived to extend the look-ahead horizon.

6.3.2 Ablation Experiment

In this section, we conduct a series of ablation experiments to single out the core contributors of EERO. We begin by comparing EERO to deliberately designed variant algorithms to provide a deeper understanding of the scheme. We then analyze how robust EERO is to varying system parameters (e.g. network architecture and quantization level of σ_μ).



(a) Optimality

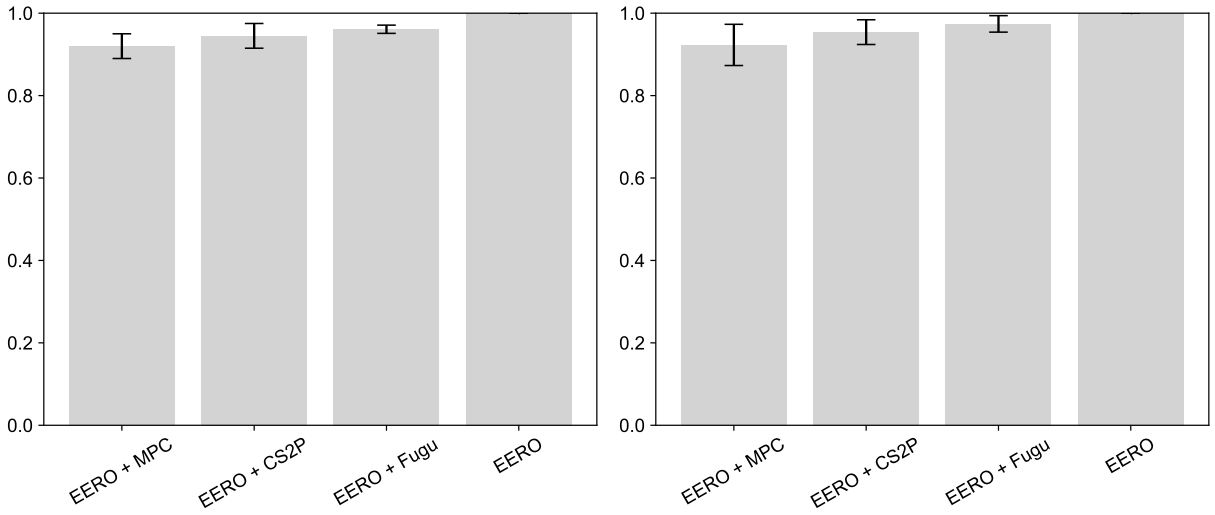
(b) Robustness

Figure 6.12: The optimality and robustness scores in BSQI of the sub-components of EERO. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average.

EERO vs. Other Baselines

EERO without MAP: The fundamental difference between EERO and other adaptation functions is the Bayesian value estimation. It is therefore natural to ask the question: which module is the core contributor to the superior performance of EERO? To this end, we decompose EERO into a value enhanced MPC-based likelihood function and a reinforcement learning-based prior model, and evaluate their performance on the test set. We compare these subroutines with their close variants Fugu [241] and Pensieve [131]. The experimental result is given in Figure 6.12. We observe that these subroutines by themselves do not achieve state-of-the-art results. The value enhanced dynamic programming outperforms its baseline Fugu. Moreover, the weighted fusion brings the performance to the next level. We conclude that the Bayesian action-value function approximation is the key to the success of EERO.

Value Enhanced Tree Search vs. Other Likelihood Functions: In the previous

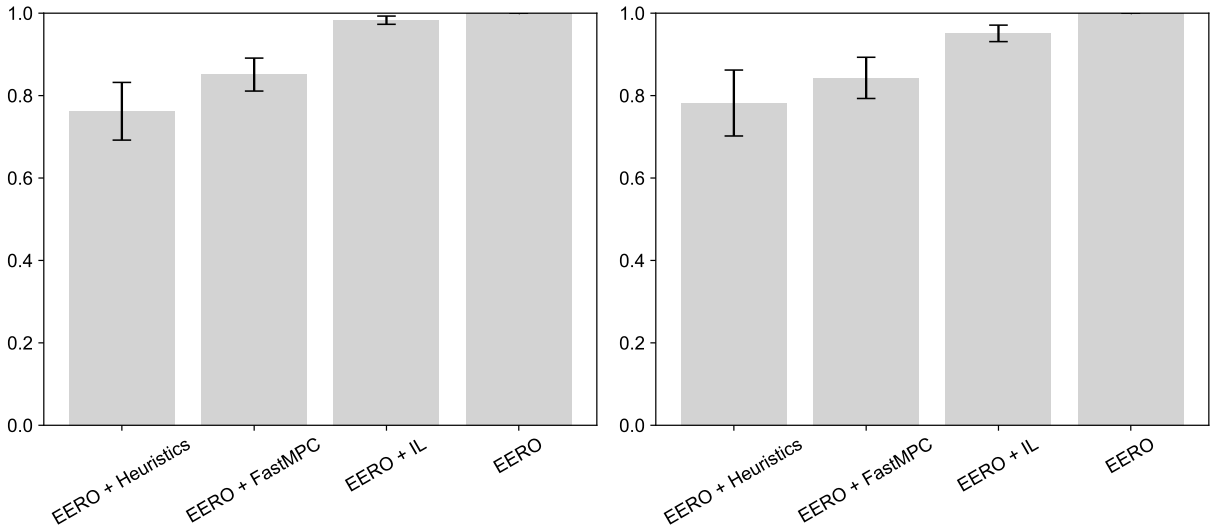


(a) Optimality (b) Robustness

Figure 6.13: The optimality and robustness scores in BSQI of EERO with different likelihood functions. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average.

experiment, we have found that the value boosted dynamic programming outperforms the default MPC. Motivated by this observation, we would like to find whether a better likelihood function would results in a higher cumulative reward in EERO. To answer this question, we replace the enhanced MPC in EERO by the truncated dynamic programming in MPC, CS2P + MPC, and Fugu (neural network-based throughput predictor + MPC) and evaluate their performance according to the same experimental procedure. The experimental results are shown in Figure 6.13. We find that EERO generally achieves a higher reward when equipped with a better likelihood function, suggesting that EERO may benefit from further improvement of online bitrate adaptation algorithms in the future.

Reinforcement Learning vs. Other Priors: Following the analysis of likelihood function, we are also interested in the impact of the prior model on EERO. We experiment with three alternative prior action-value distributions, including a heuristic ABR rule, FastMPC [241], and an imitation learned prior model. Specifically, the heuristic prior model assigns the minimum reward to most available actions to reduce the search space in



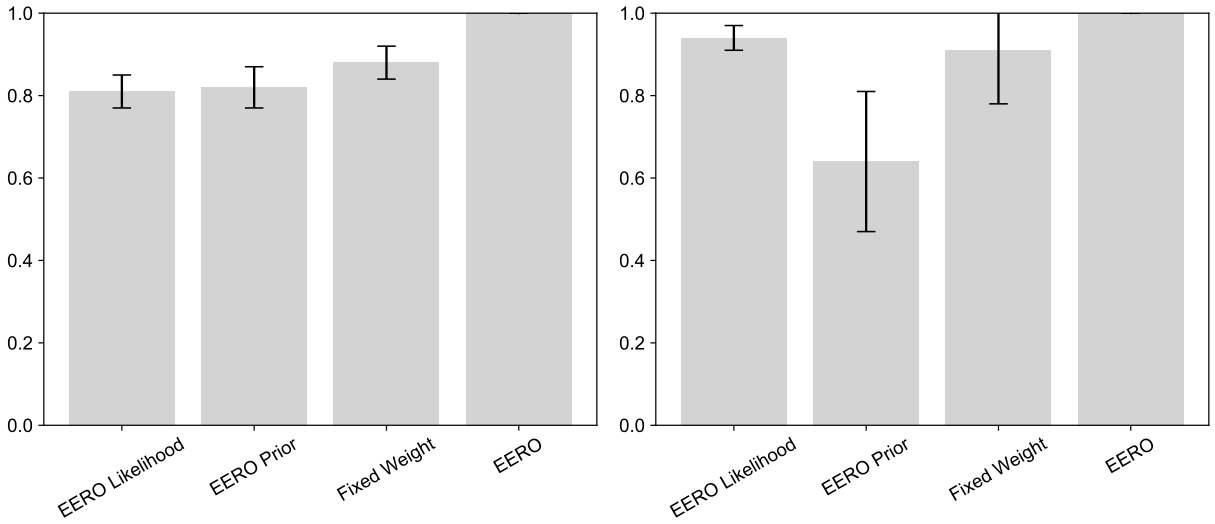
(a) Optimality

(b) Robustness

Figure 6.14: The optimality and robustness scores in BSQI of EERO with different likelihood functions. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average.

accordance with the traditional ABR rules [2, 102, 103, 134, 141]. We uniformly distribute the prior probability mass to four actions, including the lowest bitrate, the bitrate immediately below/above the current representation, and the current representation. Since there is no well-defined uncertainty measure in the heuristic method, we use a fixed σ_μ, which is optimized on a small validation set. For the imitation learning-based prior model, we follow the approach in [92] to generate a neural network-based ABR scheme. To eliminate potential inductive bias, we reuse the network architecture in deep Q-learning. The tensor σ_μ governing the uncertainty of prior estimate is trained according to the default procedure. The performance of these variants of EERO is presented in Figure 6.14. As shown, EERO can be improved by incorporating a more accurate prior model. Additionally, the imitation learning-based EERO achieves a equally promising optimality, while it is not as robust as the reinforcement learning model.

Learned σ_μ vs. Fixed σ_μ: The last key component in the proposed framework is the



(a) Optimality

(b) Robustness

Figure 6.15: The optimality and robustness scores in BSQI of EERO with different weighting schemes. Results are normalized against the performance of EERO. Error bars span ± one standard deviation from the average.

uncertainty of the action-value estimation. Our theoretical derivation establishes a natural connection between the behavior of the policy and the confidence in the action-value estimation. EERO adaptively prioritizes one of the subroutines, according to their confidence in a state-action region. In this experiment, we examine whether a naïve linear combination could lead to a similar improvement. To this regard, we experiment with a Bayesian ABR model with a fixed uncertainty value σ_μ over the entire state-action space. We manually tune the parameter by optimizing the model performance on a validation set. The resulting optimality and robustness scores are given in Figure 6.15. Interestingly, we find that a fixed combination rule can still achieve certain improvements upon the baseline prior model. However, EERO enjoys a significant gain by modeling the heterogeneity in uncertainty across the input space.

Table 6.2: Quantitative results of EERO with different numbers of neurons and layers in terms of BSQI.

# of Hidden States	# of Layers	Optimality	Robustness
32	1	58.8 ± 20.1	58.0 ± 21.2
32	3	61.9 ± 19.4	58.2 ± 20.6
32	5	63.3 ± 16.8	58.3 ± 20.4
64	1	61.4 ± 19.0	57.9 ± 21.0
64	3	67.8 ± 14.3	58.8 ± 20.5
64	5	69.1 ± 13.9	59.1 ± 20.3
128	1	65.9 ± 18.7	60.6 ± 21.2
128	3	72.5 ± 13.3	63.6 ± 19.2
128	5	72.9 ± 12.8	63.9 ± 18.7

Sensitivity Analysis

Network Architecture: We experiment with different numbers of neurons and layers. The experimental results are given in Table 6.2, where the number of neurons in each fully connected layers is given in the second column. We find that the gain induced by larger network capacity pleatues at around 128 neurons per layer.

Quantization Level: Starting from the default lookup table of σ_μ in EERO, we sweep through a range of quantization levels to understand the impact that each has on BSQI. Results from this sweep are presented in Table 6.3. As shown, performance begins to plateau once the number of bins for buffer occupancy and throughput prediction each exceed 100. Additionally, quantization levels of quality beyond 10 introduce marginal improvements in the optimality and robustness.

6.3.3 Discussion

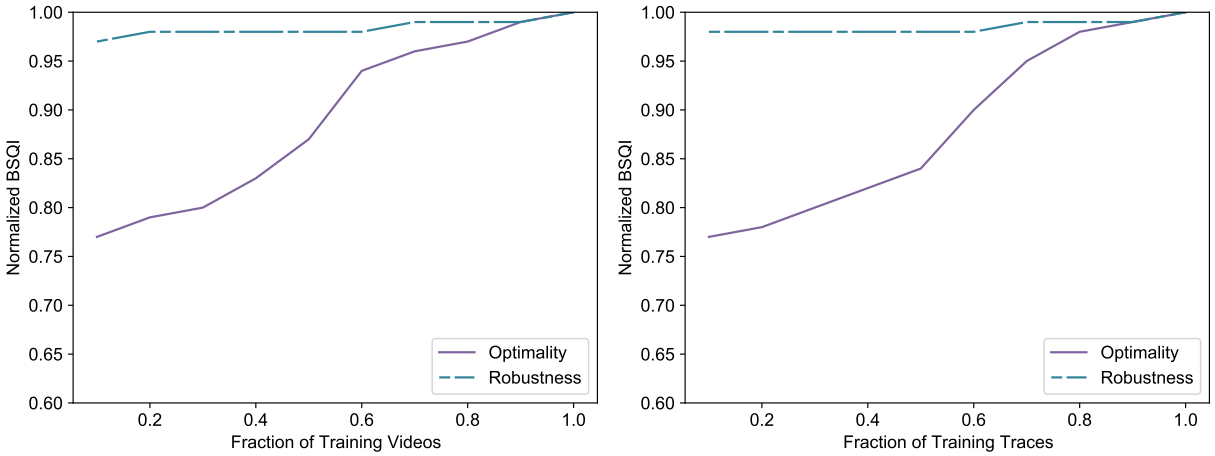
In this section, we discuss some practical concerns with deploying EERO in a video delivery system. The major difficulty in the distribution of EERO comes from the training of reinforcement learning-based prior model, for which we present a quantitative analysis.

Table 6.3: Quantitative results of EERO with different quantization levels in terms of BSQI.

Variable	# of Bins	Optimality	Robustness
Buffer Occupancy	33	67.8 ± 20.6	60.4 ± 20.9
Buffer Occupancy	66	71.4 ± 19.4	62.2 ± 20.6
Buffer Occupancy	100	72.5 ± 13.3	63.6 ± 19.2
Throughput Prediction	33	66.4 ± 18.5	58.3 ± 20.0
Throughput Prediction	66	71.8 ± 14.2	61.8 ± 20.5
Throughput Prediction	100	72.5 ± 13.3	63.6 ± 19.2
Bitrate	10	56.1 ± 21.1	55.0 ± 22.2
Bitrate	20	63.4 ± 17.4	60.2 ± 19.8
Bitrate	30	72.5 ± 13.3	63.6 ± 19.2
Quality	5	72.9 ± 18.7	60.6 ± 21.2
Quality	10	72.5 ± 13.3	63.6 ± 19.2
Quality	20	71.1 ± 12.8	63.9 ± 18.7

Training Time: We profile the training process of EERO to measure the flexibility of system deployment. Training a reinforcement learning-based prior model requires approximately 1M iterations. Each iteration involves $1,000$ Monte Carlo rollout and a backward pass for model update. We parallel the simulation process with 32 CPUs to rollout the streaming data and compute the back-propagation using a NVIDIA Pascal Titan X GPU. The prior learning process took about 10 hours. Albeit the significant overhead, the training cost is incurred offline.

Training Data: To understand the impact of the amount of training data on the performance of EERO, we train the uncertainty aware prior policy of EERO with only a portion of training set. We use the same test set and evaluation criteria described in Section 6.3.1. Figure 6.16 (a) illustrates the optimality and robustness scores with respect to the number of training videos. We find that EERO generally benefits from observing more streaming videos a priori. The result for network traces is shown in Figure 6.16 (b), from which we have a similar observation.



(a) Impact of Training Videos

(b) Impact of Training Traces

Figure 6.16: The optimality and robustness scores in BSQI of EERO with only a portion of training data. Results are normalized against the performance of EERO.

RTT (ms)	0	20	40	60	80	100	120	140	160
EERO	0.08	0.15	0.17	0.12	0	-0.16	-0.33	-0.51	-0.72

Table 6.4: Percentage of change in BSQI of EERO at different RTT over the default model.

Round-Trip Time: The RTT of ABR streaming is set to 80 ms in our simulator (note that the feed-forward prediction time can also be considered as part of RTT). We test the performance at different latency settings. The results are demonstrated in Table 6.4. We observe that EERO is fairly insensitive to latency conditions.

6.4 Summary

In this chapter, we propose a unifying framework for ABR algorithms from a Bayesian perspective. Existing algorithms approximate either the likelihood function or the action-value prior, resulting in a sub-optimal solution. Motivated by the novel insights, we provide a specific implementation in the expanded design space that is efficient, robust, and optimal.

Over a broad set of network conditions, streaming videos, and QoE measures, we found that EERO outperforms existing ABR algorithms by 7%–22%.

Chapter 7

System Validation

With many ABR algorithms at hand, it becomes pivotal to compare their performance. The existing validation studies fall short in scale, representativeness, and reliability. In this chapter, we construct the WaterlooSV database by carefully walking through the selections of each of the key components in the ABR streaming process, from source contents, encoding profiles, network traces, viewing devices, testing environment setups, to experimental methodologies. The WaterlooSV database is the largest among all streaming video databases in the literature. Building upon the dataset, we demonstrate how the RDOS paradigm improves the existing ABR algorithms. We then perform a detailed objective analysis on different combinations of throughput predictor, reward function, and bitrate selector in terms of efficiency, robustness, and optimality. To complement the limited reliability of reward functions in the objective evaluation, we carry out a large-scale subjective evaluation on ABR algorithms using a subset of the WaterlooSV dataset.

7.1 Constructing the Waterloo Streaming Video Database

Adaptive streaming is a standard communication problem composed by a transmitter, a channel, and a receiver. To comprehensively evaluate the performance of ABR algorithms, we need to faithfully reproduce each of the modules in the practical communication prob-

Figure 7.1: Sample frames of source videos in the Waterloo Streaming Video database. All images are cropped for neat presentation.

lem. In this section, we walk through the design choice of the WaterlooSV database for each module and discuss potential alternatives along the path.

7.1.1 Transmitter

Source Content: Although it is possible to make use of the video representations in the real-world streaming platforms, the approach suffers from two drawbacks. First, the copyright protected videos are restricted from copy, distribution, edit, and built upon. The proprietary content significantly hinders the reproducibility of ABR evaluation studies. Second, some analyses such as full-reference video quality assessment explicitly require the availability of pristine videos. However, most online videos underwent a series of manipulations such as resampling and compression that degrade the visual quality, prohibiting further investigations. To this end, we construct a new video database which contains 250

high quality 4K videos that span a great diversity of video contents. We resort to the Internet and elaborately select 200 keywords to search for creative commons licensed videos. The keywords can be broadly classified into 8 categories: human, animal, plant, landscape, cityscape, still-life, transportation, and computer synthesized content. We initially obtain more than 50,000 4K videos. Many of them contain significant distortions or inappropriate content, and thus a sophisticated manual process is applied to refine the selection. To make sure that the selected videos are of pristine quality, we perform two rounds of screening to remove those videos with visible distortions. In the first stage, we filter out videos with deficient attributes such that they are unlikely to retain pristine quality. Specifically, we remove videos with bitrate, frame rate, and color channel less than 7,000 kbps, 24 fps, and 3, respectively. After this step, about 10,000 videos remain. To make sure that the remaining videos are of pristine quality, we further carefully inspect each video multiple times by zooming in and remove videos with visible compression distortions. Eventually, we end up with 250 high-quality 4K videos. Sample frames are shown in Figure 7.1, where we can see the richness of video content. The duration of the videos ranges from 30 seconds to 1 hour, with an average duration of 6 minutes. As a reference point, the average duration of YouTube videos is 11 minutes [204].

Encoding: In practical video delivery, each video is encoded into multiple representations to cover a wide range of network capacity and video quality. Existing streaming video datasets employ a fixed encoding recipe, referred to as a bitrate ladder, in the evaluation of ABR algorithms. There are two limitations associated with the approach. First, the bitrate ladder used in the experiments varies significantly across different studies, inevitably resulting in sampling bias. Second, these fixed encoding profiles may deviate wildly from the encoding strategy in a practical content delivery network. In particular, different video service providers adopt different video encoders to facilitate content storage and distribution. For example, Netflix, Apple, and Google are promoting H.264 [233], HEVC [198], and AV1 [28], respectively. Furthermore, video service providers are migrating from fixed bitrate ladder to per-title encoding profile generation [36, 49, 208], since each video content exhibits a unique generalized rate-distortion function [47, 48]. The optimal encoding strategy for streaming videos is still a subject of ongoing research. To cover the wide variety of encoding strategy, we densely sample the generalized rate-distortion function to make

the dataset readily extendable to novel content preparation processes. Each video in the database is distorted by the following process sequentially:

- Spatial downsampling: We downsample the source video using the bicubic filter to six spatial resolutions (3840×2160, 2560×1440, 1920×1080, 1280×720, 854×480, 640×360) according to the recommended resolution by YouTube [242].
- Compression: We encode the downsampled sequences using three commonly used video encoders, *i.e.*, H.264 [233], HEVC [198], and AV1 [28] with two-pass encoding. For each spatial resolution, we uniformly sample 30 target bitrates in the log bitrate space on the interval 100 kbps to 45 Mbps, in accordance with the YouTube's recommendation [242]. Note that the target bitrate interval also roughly covers the encoding recommendations of other service providers such as Apple [95] and Netflix [147]. The full encoding specification is detailed in Appendix B.1.

In total, we obtain 180 (hypothetical reference circuit [100]) $\times$ 250 (content) $\times$ 3 (encoder) $= 135,000$ video representations (currently the largest in the ABR literature). Sample distorted video frames are shown in Figure 7.2.

In our experiment, we include four bitrate ladders that are widely used in practical video delivery systems. The first bitrate ladder in Table 6.1 is a combination of the Netflix's recommendation [147] and Apple's recommendation [95]. This encoding profile is fixed across all streaming videos. We also include three per-title encoding schemes that are described in [36, 48, 208]. The bitrate-centric encoding strategy [35] selects bitrate-resolution pairs such that i) At a given bitrate, the produced encode should have as high quality as possible, and ii) The perceptual difference between two adjacent bitrates should fall just below one just-noticeable different (the difference in VMAF ≈ 10). The quality-centric encoding strategy [48] pre-defines 10 target quality levels, and exhaustively searches each constant target quality contour to obtain the representation with the minimal bitrate. The per-title optimization algorithm in [208] determines the encoding strategy by optimizing the overall quality of the encoded representations subject to some constraints on resources such as server storage and throughput capacity.

Figure 7.2: Sample frames of H.264 encoded videos in the Waterloo Streaming Video database. All images are cropped for neat presentation. (a) Source reference video frame. (b)-(f) 1920×1080 at 8,000 kbps, 4,000 kbps 1,000 kbps, 500 kbps, and 100 kbps. (g)-(k) 1280$times$720 at 8,000 kbps, 4,000 kbps 1,000 kbps, 500 kbps, and 100 kbps. (l)-(p) 740$times$480 at 8,000 kbps, 4,000 kbps 1,000 kbps, 500 kbps, and 100 kbps.

Packaging: We implement a customized Python module, namely PyDASH, on top of GPAC's MP4Box [112] for video packaging. PyDASH is capable of performing feature

extraction at high efficiency and insert these information into the manifest file as the metadata. A sample manifest file generated by PyDASH is illustrated in Listing B.2. Many features are useful later in the bitrate adaptation process. To make the dataset readily extendable, we extract a comprehensive list of features that are commonly used in the QoE monitoring and ABR algorithms with PyDASH, including bitrate, resolution, frame rate, chunk duration, VMAF, PSNR, SSIM, QP, and motion vector magnitude. We segment the test sequences with a segment length of 4 seconds following the recommendation in [114].

7.1.2 Channel

Network Traces: To faithfully represent realistic network conditions, we employ the combination of several existing datasets: a broadband dataset collected by FCC [60], a 3G dataset named HSDPA [171], two 4G datasets from UCC [167] and Belgium [213], a 5G dataset from UCC [166], and a Live Television streaming dataset named Puffer [237].

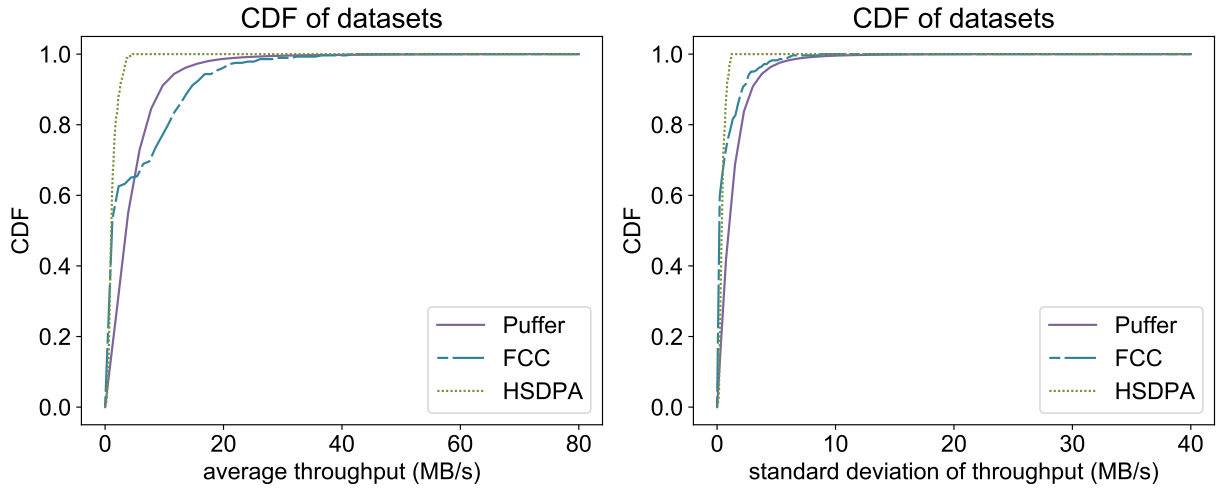
The FCC dataset contains more than 1 million throughput traces, each of which records the average throughput over $2,100$ seconds at a granularity of 5 seconds. Each trace is associated with a unique connection identifier. We select $10,000$ sessions from 500 clients by randomly cutting from the raw connection-level throughput traces, each with a duration of 120 seconds. The HSDPA dataset comprises 3G throughput measurements at a granularity of 1 second, collected from mobile devices that were streaming video while in transit. The experiments were performed in the period Sep. 13, 2010 to Apr. 21, 2011 in Norway. 6 out 11 scenes contain more than five traces, resulting in a total of 78 valid traces. We apply a sliding window to generate $1,000$ throughput traces to form the 3G network dataset. The Belgium dataset consists of 40 4G bandwidth traces recorded along several routes in and around the city of Ghent at a 1-second granularity. The UCC dataset is composed of client-side cellular key performance indicators collected across different mobility patterns (static, pedestrian, car, tram and train). The 4G trace dataset contains 135 traces, with an average duration of fifteen minutes per trace at a granularity of one sample per second. The dataset also contains synthetic throughput traces from 100 mobile users. We consider each route corresponds to a network environment, based on which the connection-level throughput traces are extracted. To match the duration of our selected

FCC traces, we generate 10,000 traces using a sliding window across the two 4G datasets, each with a duration of 120 seconds. The 5G dataset comprises 45 traces collected from 15 environmental conditions. We follow a similar way to pre-process the data, obtaining 1,000 traces. Started since Jan. 26, 2019, the Stanford Puffer dataset is an ongoing research project that collects connection-level data in a realistic streaming video environment. To date, the dataset includes more than 5M individual throughput traces collected on the Amazon Mechanical Turk platform. We select all traces from June 2019 to May 2021 in the construction of the WaterlooSV database. During the experiment, a number of environmental statistics are logged from both video servers and players per video chunk. As a result, the Puffer dataset does not provide the throughput at a fixed granularity. To make the data format consistent, we apply linear interpolation to the source data such that the a throughput measurement is recorded every two seconds. Each trace in the Puffer dataset is assigned a SessionID, and throughput traces with the identical SessionID are generated from the same connection. We cut every raw trace into 120-second sequences and randomly select 20 traces for every task. With this pro-processing above, we randomly select 10,000 traces from 500 users. In total, we obtain 32,000 throughput traces from 1,600 clients, each with duration of 120 seconds.

The characteristic of each dataset is shown in Figure 7.3. Among five datasets, throughput is the most stable in broadband network and the most variable in mobile network. Furthermore, the 3G HSDPA dataset exhibits the lowest average throughput. Therefore, the HSDPA dataset provides a stress test to ABR algorithms, while broadband datasets can serve as a test-bed for inefficient bitrate usage.

7.1.3 Receiver

ABR Algorithms: We provide an open-source implementation of ABR algorithms in both Python and Javascript. We implement the library in a way that 1) it is easy to perform ablation experiment, and 2) it is easily extendable. In particular, each ABR algorithm consists of three functional components, namely throughput predictor, reward function, and bitrate selector. In the current stage, there are seven throughput predictors, six reward functions, and eight bitrate selectors in the WaterlooSV database. The implemented



(a) (b)

Figure 7.3: Network characteristics of the three throughput trace databases.

algorithms in each module are summarized in Table 7.1. Customized ABR rules can be easily obtained by plugging in different implementations of the three modules.

Viewing Conditions: The ultimate receivers of streaming videos are human beings, who consume multimedia on a large variety of viewing conditions. According to the capability of viewing device, the adaptive streaming player may re-sample the video to different resolution, frame rate, and dynamic range, each of which may change QoE in some way. The re-sampled video signal goes through the viewing environment and arrives in the retina. The transmission process may further amplify or alleviate visual distortions by the background luminance level and viewing distances, which interplays with HVS features such as the contrast sensitivity function [173]. To make the problem tractable, we simplify the viewing environment by only taking the viewing distance into consideration. In this study, we consider three mostly used viewing devices including High Resolution Television (HDTV), smartphone, and UHDTV and their typical viewing distance according to [104]. Note that the presentation quality is a function of viewing environment, which we take into account with device adaptive VMAF scores (see Appendix B.3 for more details). An alternative presentation quality measure is SSIMplus [169], which offers richer and more

Table 7.1: Implemented throughput predictors, reward functions, and bitrate selectors in the WaterlooSV dataset. Abbreviations: AM, arithmetic mean; HM, harmonic mean; EWMA, exponential weighted moving average; HMM, hidden markov model; MLP, multi-layer perceptron; MetaTP, meta learning-based throughput predictor; BO, buffer occupancy; LB, linear function of bitrate; NB, non-linear function of bitrate; LV, linear function of video quality assessment score; BSQI, Bayesian streaming quality index; RDOS, rate-distortion optimized streaming; Greedy/BO, greedy algorithm without taking buffer occupancy into consideration; DP, dynamic programming; SL, supervised learning; A2C, actor advantage critic; EERO, efficient, robust and optimal bitrate selector.

Modules	Algorithms
Throughput predictor	AM, HM, EWMA, Linear, HMM, MLP, MetaTP
Reward function	BO, LB, NB, LV, BSQI, RDOS
Bitrate selector	Heuristic, Greedy/BO, Greedy, DP, SL, Q-Learning, A2C, EERO

precise device-adaptive scoring, though its implementation is not publicly available.

7.2 Objective Evaluation

Building upon the WaterlooSV database, we perform a comparative study to illustrate the advantages of the RDOS paradigm over the traditional bitrate maximization paradigm. We then evaluate the full system implementation of RDOS.

7.2.1 Experimental Setup

Evaluation Framework: Existing studies evaluate the performance of ABR algorithms by performing video streaming between a video server and a DASH client. The typical evaluation architecture consists of four modules: two computers with a direct network connection emulating a video client and server. Streaming videos are pre-encoded and hosted on an Apache Web server. The main components of this architecture are the bandwidth shaping and the network emulation nodes which are both based on Ubuntu utilities.

The available bandwidth for the client is adjusted every second according to bandwidth traces. The video client, where ABR algorithms are deployed, rendered videos at full screen while the video server is a simple HTTP server. However, given the diverse streaming environment in the WaterlooSV database, it is prohibitively expensive to perform an exhaustive evaluation of ABR algorithms using the real-player emulation setup. In particular, it would take around 4,000 years to download and render all possible combinations of streaming videos for a single player. Nevertheless, we can leverage the chunk-level simulator in [131], which has been previously demonstrated to faithfully model the application layer network. The streaming simulator maintains an internal representation of the client's playback buffer. For each chunk download, the simulator computes a download time based on the chunk bitrate and the input network throughput traces. The environment model then drains the playback buffer by the current chunk download time to simulate video playback during the download, and replenishes the buffer with the duration of the downloaded chunk. The simulator carefully keeps track of rebuffering events that arise as the buffer occupancy changes, i.e., scenarios where the chunk download time exceeds the buffer occupancy at the start of the download. In scenarios where the playback buffer cannot accommodate video from an additional chunk download, the simulator pauses requests for 500 ms before retrying, which is the default behaviour of DASH player. After each chunk download, the simulator sends relevant state information to the ABR agent for the bitrate decision of the next chunk. In our objective evaluation, the streaming simulator is configured to emulate the network conditions from our corpus of network traces, along with an 80 ms RTT, between the client and server.

Model Training: We randomly split the streaming video contents/bitrate ladders/throughput traces into 60% training, 20% validation, and 20% testing sets. The throughput traces are split according to the client id to facilitate connection-level optimization. We follow the experimental setup in Section 5.3.1 to train the proposed MetaTP, suggesting that 20% of the traces from each client in the testing set is used for fast adaptation. For fairness, these traces are also included in the training set for other data-driven ABR algorithms.

Evaluation Criterion: In this work, we are not only interested in the streaming video QoE, but also the amount of bitrate resource it takes to achieve a certain QoE level. For two ABR algorithms resulting in the same QoE, the algorithm consumes less bitrate should be

considered better than the other. In order to capture this property, we propose to generalize the Bjøntegaard-Delta bitrate measure [20], which has been widely used to evaluate the performance of video encoders. Specifically, the performance of an ABR algorithm on a specific video content can be characterized by its rate-distortion curve generated from a corpus of network traces. The average bitrate differences between rate-distortion curves for the same QoE is given by

$$R_{\mathrm{BD}} = \left\{ \exp \left\{ \frac{\int_{q_{\mathrm{L}}}^{q_{\mathrm{H}}} \left[r_{\mathrm{B}}(q) - r_{\mathrm{A}}(q) \right] dq}{q_{\mathrm{H}} - q_{\mathrm{L}}} \right\} - 1 \right\} \times 100, \tag{7.1}$$

where q, r_{A}, and r_{B} are the QoE, the logarithmic-scale bitrate of streaming videos generated by the reference and test ABR algorithms, respectively. $[q_{\mathrm{L}},\ q_{\mathrm{H}}]$ is the effective range covered by the rate-distortion curves under test. The overall rate-distortion performance can be obtained by taking the average R_{BD} across all video contents, video encoders, and viewing devices, leading to a comprehensive evaluation of ABR algorithms.

7.2.2 Rate Distortion Optimization Paradigm vs. Bitrate Maximization Paradigm

We directly compare the proposed RDOS paradigm with the traditional bitrate maximization paradigm by adapting the existing ABR algorithms to optimize the rate-distortion reward. Each RDOS adapted ABR algorithm is compared against its original counterpart in terms of the bitrate saving in (7.1). In this study, we adopt the proposed BSQI as the QoE measure in (3.2). Nevertheless, the proposed framework is general enough to incorporate better QoE measures once they become available.

Exhaustive evaluation of all ABR algorithms is difficult as it involves optimizing over an infinite-dimensional functional space. To this end, we evaluate the following ABR algorithms, ranging from the naïve de facto rate-based algorithm to the state-of-the-art algorithms, and optimize the free parameters by empirical simulations based on the training dataset:

- RB: RB ABR algorithm [63] employs a variant of greedy algorithm to optimize a linear bitrate-based reward function. The available future throughput is predicted

by the arithmetic mean of observed throughput over the past five chunks. The algorithm makes a conservative bitrate decision without using the buffer occupancy status.

- BOLA: BOLA [191] makes greedy optimization to maximize a reward function that encourages higher bitrate, shorter rebuffering duration, and lower buffer occupancy. Under some mild assumptions, the problem can be solved by Lyapunov optimization, which select optimal bitrates solely based on buffer occupancy observations.

- FastMPC: FastMPC [241] uses both buffer occupancy observations and throughput predictions using harmonic mean of the past 5 chunks to select bitrate. In each step, the algorithm maximizes a linear bitrate-based QoE measure over a horizon of five future chunks, and re-plans the trajectory as new state variables become available. The optimization problem is solved offline and its solution is stored as a lookup table. We use 100 bins for throughput prediction, 100 bins for buffer level, and 13 bins for the past bitrate level.

- Fugu: Fugu [237] is a variant of FastMPC. The major improvement is a data-driven throughput predictor over the model-based harmonic mean.

- Pensieve: Pensieve [131] is a model-free reinforcement learning-based ABR algorithm that learns to select optimal bitrate in terms of a linear bitrate-based QoE model. The algorithm learns a CNN-based policy that takes throughput observations and buffer occupancy of previous eight chunks as input and produces the optimal bitrate decision. One of the biggest advantages of the algorithm is that it can efficiently optimize the long-term reward.

To derive the RDOS counterpart of these ABR algorithms, we made the following modifications:

- RB+RDOS: The adaptation of RB to the new RDOS paradigm is straight-forward. We simply replace the original reward function by the linear combination of BSQI and bitrate.

- BOLA+RDOS: To lift the need for throughput prediction, the original BOLA algorithm makes a prior assumption about the bandwidth characteristics. The throughput distribution is a implicit function of the model hyper-parameters, making the conversion to RDOS non-trivial. To this end, we use the simplest throughput predictor, *i.e.*, the nearest neighbourhood predictor, in the adapted version of BOLA. Similarly, we perform the greedy optimization scheme to maximize the RDOS reward function.

- FastMPC+RDOS: To obtain the necessary input to perform rate-distortion optimization, we look ahead both the quality scores and chunk-level bitrate in the next five chunks. We also expand the input space by incorporating the presentation video quality. We set the quantization step in the lookup table to 10. In the end, we optimize the RDOS reward function at each state to obtain the revised lookup table.

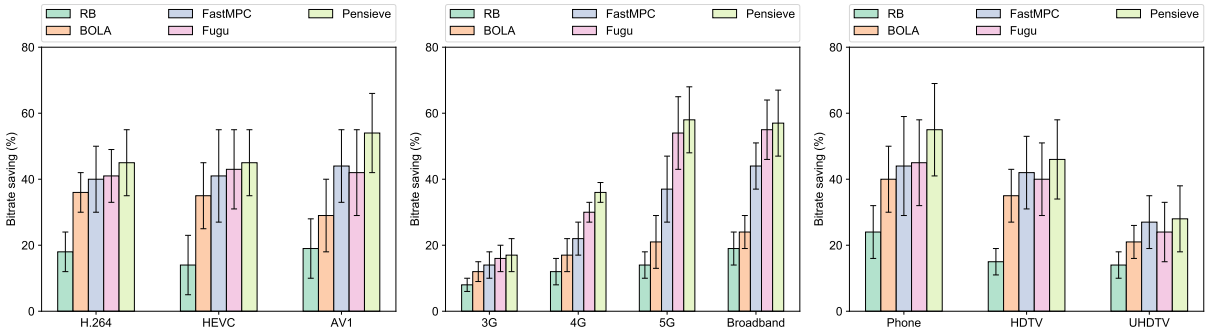
- Fugu+RDOS: The RDOS optimized CS2P can be obtained in a similar fashion to the FastMPC.

- Pensieve+RDOS: We extend the input space of Pensive to integrate the presentation video quality, and change the neural network architecture accordingly. We utilize Advantage Actor Critic Algorithm (A2C) as the learning algorithm to optimize the RDOS reward function.

We set the rate-distortion tradeoff parameter $\lambda = 0.001$ in the experiment.

Experimental Results

Figure 7.4 demonstrates the average bitrate saving of RDOS schemes over its bitrate maximization counterpart for different encoders, throughput traces, and viewing conditions. Figure 7.5 shows the QoE change in BSQI of each bitrate maximization algorithm introduced by employing the RDOS objective function. There are three key takeaways from these results. First, we find that the RDOS framework can achieve 12%-45% bitrate saving without sacrificing QoE for all baseline ABR schemes. In general, RDOS achieves a



(a) Encoder (b) Network (c) Device

Figure 7.4: The bitrate saving of rate-distortion optimized ABR algorithms over its default version. Error bars span ± one standard deviation from the average.

higher bitrate saving with a more accurate throughput prediction model and a better bitrate adaptation function. For example, the rate-distortion optimized Pensieve produces a bitrate saving of 18%-58% over the original bitrate maximization Pensieve across all scenarios considered while obtaining the highest average QoE of 67.

Second, due to the cross-video, cross-encoder, and cross-device adaptation capability, RDOS attains different bitrate savings at different operating conditions. Specifically, the maximum bitrate saving is obtained on smartphone and the AV1 encoded video streams, where the presentation QoE score predicted by SSIMplus or VMAF usually saturates at relatively low bitrate levels. Being aware of the device-specific rate-distortion characteristics, the proposed framework usually picks the intermediate bitrate level of sufficiently high QoE, effectively maximizing the presentation QoE scores while minimizing the bitrate usage. The bitrate conservative strategy also has extra benefit in reducing the probability of rebuffering, especially in high performance but high variability network environment such as 5G. In particular, RDOS-based algorithms reduce rebuffering duration by 12% on average comparing to the original implementations.

Third, RDOS not only earns the maximum bitrate saving when there are abundant bandwidth resources, it also demonstrates notable performance gain at low-bandwidth conditions. By having a closer look at the streaming logs, we find that RDOS is able to learn a policy that starts with low bitrate level, gradually switches up, and stays at an

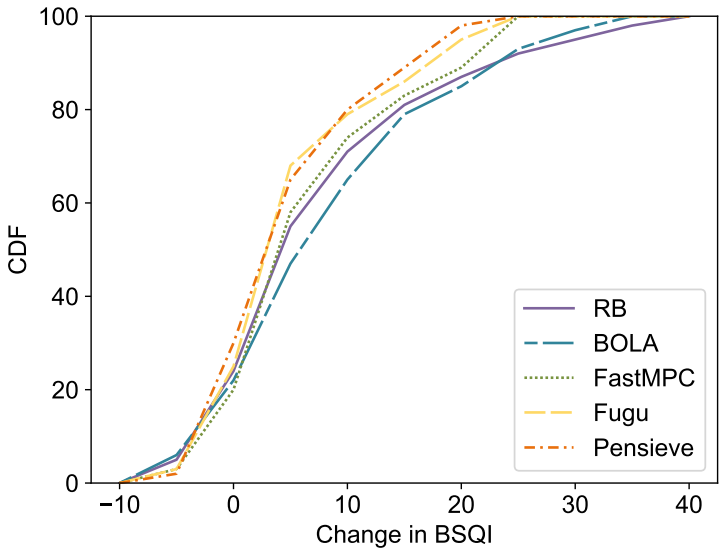


Figure 7.5: The cumulative density function of QoE change in terms of BSQI introduced by applying the RDOS reward function.

intermediate bitrate level at poor bandwidth conditions, while other ABR algorithms either constantly makes conservative decisions or erratically switches up and down according to the instantaneous bandwidth estimate or buffer occupancy observations. The difference may be explained by the perceptually motivated QoE model employed by RDOS, whereby positive adaptations are preferred over negative adaptations. On the other hand, FastMPC and Pensieve, which penalize switching up and switching down equally, usually stay at the minimum bitrate level even when temporary switching up is possible.

7.2.3 Full System Validation

In this section, we evaluate the performance of the full ABR system, which is a combination of RDOS (with BSQI as the QoE measure), MetaTP, and EERO. For simplicity, we name the full algorithm as RDOS. We compare RDOS with a comprehensive list of existing ABR algorithms, including

- RB: Refer to 7.2.2.

- BB: We employed the function suggested by Huang *et al.* [93], where bitrate is chosen as a piece-wise linear function of buffer occupancy. The algorithm always starts with the lowest bitrate till the buffer occupancy reaches a certain threshold called reservoir. Once reservoir is filled up, a higher bitrate is selected as the buffer occupancy increases till there is enough video segment in the buffer (upper reservoir) to absorb the variation caused by the varying capacity and by the finite chunk size, where the range from the lower to upper reservoir is defined as cushion. We set a lower reservoir and cushion to be 5 and 10 seconds, respectively.

- AIMD: The heuristic algorithm picks the representation according to the bandwidth estimation using the previous downloaded chunk in an additive increase and multiplicative decrease manner [122]. When the two thresholds for switching are not met, the algorithm keeps the selected bitrate.

- ELASTIC: This algorithm incorporates a PI-controller to maintain a constant duration of video in the buffer (10 seconds in the experiment). Since the bandwidth estimation module is not specified in the original implementation, we adopt the throughput prediction using harmonic mean of the past five chunks, because it is shown to be effective in previous studies [102].

- QDASH: QDASH picks an intermediate bitrate when there is a bandwidth drop to mitigate the negative impact of abrupt quality degradation [141]. Without impacting the performance, we replace the proxy service for bandwidth estimation in the original implementation with the throughput prediction using harmonic mean of past five chunks for simplicity.

- FESTIVE: This rate-based algorithm balances both efficiency and stability, and incorporates fairness across players, which is not a concern of this paper [102]. We assume there is no wait time between consecutive chunk downloads, and implement FESTIVE without the randomized chunk scheduling. Note that this does not negatively impact the player QoE. Specifically, FESTIVE calculates the efficiency score depending on the throughput prediction using harmonic mean of the past five chunks, as well as a stability score as a function of the bitrate switches in the past five chunks.

The bitrate is chosen to be the minimal stability score plus $\alpha = 12$ times efficiency score.

- BOLA: Refer to 7.2.2.
- Robust Model Predictive Control (RobustMPC): The algorithm applies dynamic programming to solve the QoE optimization problem online. In accordance to the recommendation in [102], RobustMPC employs the harmonic mean-based throughput predictor to estimate the bitrate resource. In each step, the algorithm maximizes a linear bitrate-based QoE measure over a horizon of five future chunks, and re-makes the bitrate decision at the next state.
- FastMPC: Refer to 7.2.2.
- SDNDASH: SDNDASH is a centralized ABR algorithm that is dedicated to optimize the expected group-level QoE [16]. Without impacting the performance, we employ a client-side implementation of the algorithm, which is composed of a model-based throughput predictor, a linear VQA-based reward function, and a greedy bitrate selector.
- CS2P: Refer to 7.2.2.
- Q-Learning: Back in 2013, Claeys *et al.* [32] presented the first reinforcement learning-based ABR algorithm. The algorithm, dubbed Q-Learning, learns a mapping between each (state, action) pair and the corresponding expected total reward during the interaction with a trace-driven environment simulator. The reward in the study is a non-linear function of bitrate level, rebuffering duration, and bitrate oscillation.
- Pensieve: Refer to 7.2.2.
- Fugu: Fugu is a variant of RobustMPC which employs advanced machine learning techniques in throughput prediction. Specifically, it learns a multi-layer perceptron to predict the throughput in the next five chunks by using a very-large scale realistic throughput dataset. By default, this ABR optimizes a SSIM-based reward function. In this study, we replace the default reward function by a linear function of VMAF,

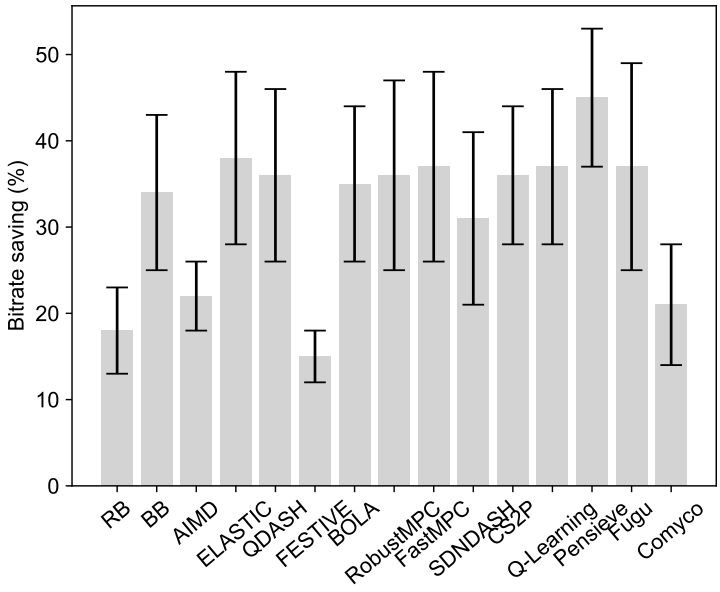


Figure 7.6: The bitrate saving of full RDOS over its competing algorithms.

rebuffering duration, and VMAF variation, because 1) the model exhibits a higher correlation with subjective QoE measurement and 2) both of these models rely on the same assumptions.

- Comyco: This ABR algorithm performs imitation learning [92] to mimic the behavior of an offline optimal adaptive streaming algorithm. We use the default settings suggested in the original paper to train the RNN-based ABR controller.

Experimental Results

Figure 7.6 shows the average bitrate saving achieved by the full RDOS algorithm over each competing ABR algorithms on our entire test corpus. Figure 7.7 compares RDOS to all test ABR algorithms in terms of the utility from the average quality, bitrate consumption, and rate-distortion reward in (3.2). The key observations are summarized as follows. First, we find that RDOS significant reduces the bitrate assumption without inducing the loss in QoE, achieving an average bitrate saving of 18%-46% over the existing ABR

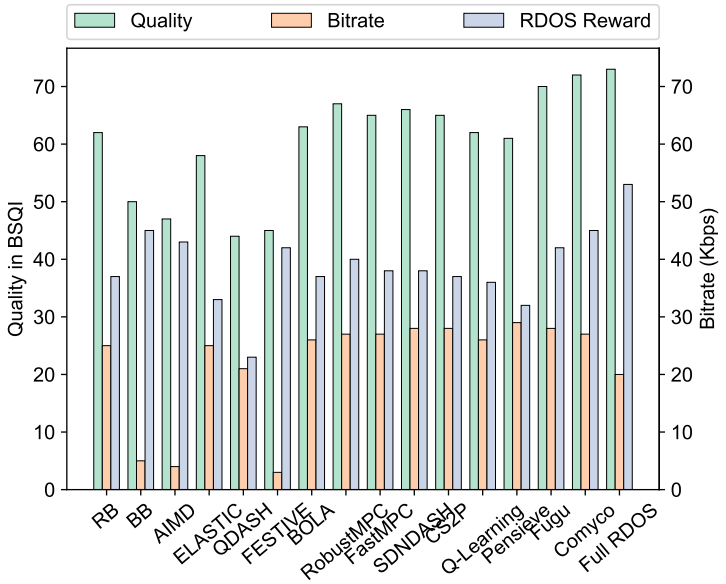


Figure 7.7: The composition of rate-distortion reward obtained by the test ABR algorithms.

algorithms. The closest competing algorithm is Comyco, which employs a variant of BSQI as reward function and a pure data-driven adaptation policy. Thanks to the novel rate-distortion paradigm, the accuracy throughput predictor, and the robust control policy, RDOS obtains a comparable QoE to Comyco but reduces the bitrate usage by 19%. Second, the heuristic ABR algorithms generally receive the lowest QoE because they always apply conservative strategy, regardless of the throughput condition. For example, FESTIVE selects the maximum bitrate level at a probability lower than 2% even with sufficient bandwidth resource. In contrast, RDOS can quickly switch to the best representation when necessary, spending 2%, 5%, and 12% time in playing the highest bitrate level on Phone, HDTV, and UHDTV, respectively. Third, despite consuming the most bitrate, Pensieve does not produce the highest QoE. In fact, the two tabular ABR algorithms Q-learning and FastMPC that optimize the same linear bitrate reward function outperform Pensieve in terms of BSQI. On the other hand, a close variant of Pensieve, namely Comyco, can effectively alleviate this problem. This suggests that there exists significant gap between bitrate and QoE, and thus simply maximizing bitrate as a proxy of subjective QoE may

(a) Slides (b) Game (c) Movie (d) Nature (e) Sport

Figure 7.8: Snapshots of video sequences.

Table 7.2: Spatial Information (SI), Temporal Information (TI), Frame Rate (FPS), and Description of Reference Videos

Name	FPS	SI	TI	Description
Slides	30	97.7	1.1	screen content, static
Game	30	25.6	12.4	animation, high motion
Movie	24	31.2	28.2	computer generated, high motion
Nature	30	46.5	2.8	natural, animal
Sport	30	32.5	22.0	human, high motion

lead to sub-optimal performance.

7.3 Subjective Evaluation

In this section, we first describe the construction of the new WaterlooSQoE-IV database including the source material collection and the simulation experiment setup. We then present the details of the subjective experiment for collecting human annotations.

7.3.1 Database Construction

Source videos: We select five high-quality 4K creative commons licensed videos from the Internet, which span a diverse set of content genres, including screen content, video game, movie, natural scene and sport. There are different types of camera motion, including static (*e.g.* Slides, Game and Nature) and complex scenes taken with a moving camera,

with panning and zooming (*e.g.* Movie and Sport). To make sure that the videos are of pristine quality, we carefully inspect each of the videos multiple times by zooming in and remove those videos with visible distortions. The detailed specifications of those videos are listed in Table 7.2 and a screenshot from each video is included in Figure 7.8. Ideally, the sampled videos should come from the real-world streaming video distribution, suggesting an average video duration of 11.7 minutes [204]. However, there exist several challenges in the development of subjective experiment for long streaming videos in practice. First, given the limited capacity for subjective experiment, the design of QoE database has to trade off the number of source content, the number of influencing factors, and the duration of each video. Second, the generalization capability of existing subjective experiment protocols on such long video sequences has not been examined in the literature. To alleviate these problems, we cut a 32-second video clip from each source content. This choice is in accordance with many recent studies [11, 27], which suggest that longer videos of up to 30 seconds may be required to be able to test the impact of switching patterns.

Encoding profiles: Using the aforementioned sequences as the source, each video is encoded with H.264 [233] and High Efficiency Video Coding (HEVC) [198] encoders into 13 representations using the bitrate ladder shown in Table 6.1 to cover different quality levels. The choices of bitrate levels are based on Netflix's recommendation [147] while the last two representations are appended to the original bitrate ladder to cover the high-quality representations suggested in Apple's recommendation [95]. Despite the recent development in content adaptive bitrate ladder generation [36, 49, 208], there has been no widely accepted per-title encoding strategy. Furthermore, some ABR algorithms only accept a fixed set of encoding profiles as input [131]. We segment the test sequences with GPAC's MP4Box [112] with a segment length of 4 seconds. Since some testing ABR algorithms rely on chunk-level bitrate and presentation quality scores in the bitrate selection, we precompute and embed them as the attributes of SegmentURL [63] in the manifest file that describes the specifications of the video.

Network traces: We pick nine network traces from the test set in the simulation experiment, as shown in Figure 7.9. To cover different network conditions, we choose traces at a variety of mean and variance. Some network traces are likely to cause sudden bitrate/quality changes and rebufferings even if the average bandwidth is relatively high.

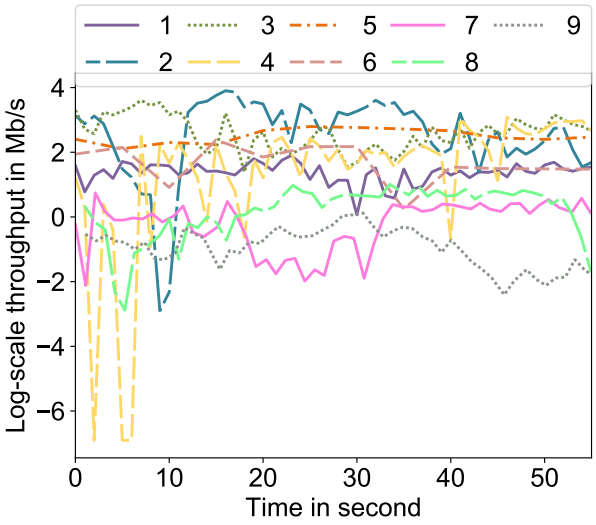


Figure 7.9: Network traces used in the subjective experiment.

ABR algorithms: We evaluate the five representative ABR algorithms including RB, BB, FastMPC, Pensieve, and RDOS. We implement the ABR algorithms in dash.js (version 2.9.2) [63]. We optimize the free parameters of BB, FastMPC, Pensieve, and RDOS across an independent database of training videos generated from 250 high-quality 4K videos and 3040 network traces. The training data is generated in a similar fashion to ensure the optimality of ABR algorithms on the test set. For FastMPC, we compress the javascript code directly instead of performing run-length coding on the lookup table. For RDOS, we set the tradeoff parameter to 0.001. In our experiment, we find the simplification introduces minimum overhead and the code size is close to the original implementation [241]. We implemented a simplified version of RDOS by replacing the reward function of the Pensieve (with integrating MetaTP and EERO[1]). To perform feed-forward prediction in the browser, we convert the actor networks of Pensieve and RDOS to Tensorflow.js [97] and save the models in the client local storage via IndexedDB [143].

Viewing devices: The ultimate receivers of streaming videos are human beings, who

[1]We carried out the subjective experiment before the development of MetaTP and EERO. Due to the time complexity of an user study, the subjective evaluation of the full system implementation is considered as a future work.

consume multimedia on a large variety of viewing devices. In this thesis, we consider three mostly used viewing devices according to [104], including HDTV, smartphone, and UHDTV. Note that the presentation quality is a function of viewing device, which we take into account with device adaptive VMAF scores.

Experimental setup: In order to generate meaningful and representative test videos for subjective experiment, we conduct a set of DASH video streaming experiments, recorded the relevant streaming activities, and reconstructed the streaming sessions using video processing tools. DASH videos were pre-encoded and hosted on an Apache Web server. We used Mahimahi [148] to emulate the network conditions from our corpus of network traces, along with an 80 ms round-trip time, between the client and server. The client video player is a customized Chromium browser (version 73) supporting H.264 and HEVC playback. Both the client and server run on the same computer with an Intel i7-6900K 3.2GHz CPU. Before a streaming session is initialized, the player selects one viewing device from HDTV, Phone, and UHDTV, and parses presentation QoE scores from the manifest file accordingly. After each video streaming session, a log file was generated on the client device, including selected bitrates, duration of initial buffering, and the duration of each stalling event. We then reconstructed each streaming video with FFmpeg [206]. Aiming to evaluate the performance at steady status, we force all ABR algorithms to start with the same quality level. We remove the initial buffering and the first chunk from the streaming videos for presentation.

The simulation with realistic network traces and ABR systems ensures the generated streaming videos come from the real-world distribution. Furthermore, the end-to-end treatment from server, network to client viewing device enables controlled data analysis, which is not possible with only the streaming videos in the wild.

Summary: A total of 1,350 streaming videos (5 source videos $\times$ 2 encoders $\times$ 9 network traces $\times$ 5 ABR algorithms $\times$ 3 viewing devices) are generated for presentation. The mean and standard deviation of the video duration are 30.7 and 1.8 seconds, respectively.

7.3.2 Subjective Testing

Choice of testing methodology: Given the large-scale streaming videos and the limited capacity of subjective testing, it is prohibitively difficult to employ the pairwise comparison subjective testing method, which arguably produces more reliable ratings [100, 240]. There exist two alternatives in the literature including Single Stimulus (SS) and Single Stimulus Continuous Quality Evaluation (SSCQE) methods. In SS methods, a single streaming video is presented and the assessor provides an index of the entire presentation. The approach has become the standard subjective testing method in the field of visual communication and has been applied in several streaming video QoE datasets [53, 54, 70]. By contrast, the SSCQE scheme records not only continuous-time QoE scores while participants are viewing test stimulus, but also retrospective scores at the end of the presentation. The past decades has witnessed an increasing trend towards the usage of single stimulus continuous quality evaluation in streaming video QoE assessment[11, 12, 142], thanks to its capability to provide scene-dependent and time-varying quality evaluation. We conduct a small-scale pilot study to investigate the feasibility of SS and SSCQE in the QoE assessment of streaming videos. There are 200 streaming videos in the experiment, which are generated in a similar fashion to the WaterlooSQoE-IV dataset. 10 participants take part in the pilot study, who are uniformly assign to the SS experiment and the SSCQE experiment. To investigate the reliability of each experiment, each of the participant performs the same experiment twice. The repeated experiments are scheduled on different date to reduce the fatigue effect and memory effect. In the SS experiment, a single streaming video is presented and the assessor provides an index of the entire presentation, while in the SSCQE experiment, we record not only continuous-time QoE scores while participants are viewing test stimulus, but also retrospective scores at the end of the presentation. After the experiment, the participants in the SSCQE group report that they frequently encounter difficulties in recalling retrospective scores due to the limited mental capacity. The phenomenon is evident by the low repeatability and reliability of the SSCQE experiment. Specifically, we obtain an inner-subject correlation and inter-subject correlation of 0.73 and 0.65 in the SSCQE experiment, respectively, which are significantly lower than the inner-subject correlation and inter-subject correlation of 0.82 and 0.79 in the SS experiment. Furthermore, there is time delay between the recorded instantaneous quality and

the video content, and such delay varies between subjects and is also a function of slider "stiffness". This is an unresolved issue of the general SSCQE methodology, but is avoided when only a single score is acquired. On the other hand, the long duration of test videos in SS as opposed to the international recommendation [100] comes with a cost. We find that participants gradually loss interests in viewing test stimuli, merely paying attention to the first few segments. To overcome these problems, we propose a variant of SS by introducing an auxiliary task in the experiment. In particular, each subject is asked to (1) perform a keystroke whenever a rebuffering event occurs and (2) provide the overall QoE score at the end of each presentation. The auxiliary task not only motivates participants focusing on the experiment materials, but also helps us identify outliers who do not attend to the full test stimuli. We empirically observe a better inner subject correlation in the proposed experiment. As a result, we adopt the dual-task SS as the subjective testing methodology in the development of WaterlooSQoE-IV.

Experiment procedure: The subjective experiment is carried out over a period of eight weeks at the University of Waterloo at Image and Vision Computing subjective testing lab. The environment is setup as a normal indoor home settings with an ordinary illumination level, with no reflecting ceiling walls and floors. A customized graphical user interface is used to render the videos on the screen and to record the individual subject ratings on the database. In order to remove any memory effects, we randomly shuffle the source contents and the corresponding playout patterns while ensuring that the same content is not consecutively displayed to a subject in any session. Furthermore, the play order is kept different for each participant to minimize the impact of context on the subjective experiment. A total of 97 naïve subjects, including 50 males and 47 females aged between 18 and 38, participate in the subjective test. Given the time constraint, each subject is randomly assigned a viewing device from HDTV (24 inch ViewSonic VA2452SM), Phone (5.8 inch Apple iPhone XS Max), and UHDTV (55 inch Sony XBR55X800H). In the end, the Phone, HDTV, and UHDTV studies received ratings from 33, 32, and 32 participants, respectively. All videos are displayed at full-screen on each of the devices. The monitors are calibrated in accordance with the ITU-R BT.500 recommendation [100]. Observers are seated at a distance of 3 $\times$ the height of the viewing device [100]. Visual acuity and color vision are confirmed from each subject before the subjective test. A training session

is performed, during which, 8 videos that are different from the videos in the testing set are presented to the subjects. We used the same methods to generate the videos used in the training and testing sessions. Therefore, subjects knew what distortion types would be expected before the test session, and thus learning effects are kept minimal in the subjective experiment. Subjects were instructed with sample videos to judge the overall QoE considering all types of streaming activities in the session. For each subject, the whole study takes about 5.5 hours, which is divided into eleven sessions spanning over three days. In order to minimize the influence of fatigue effect, the length of a session was limited to 25 minutes. The choice of a 100-point continuous scale as opposed to a discrete 5-point ITU-R Absolute Category Scale (ACR) has advantages: expanded range, finer distinctions between ratings, and demonstrated prior efficacy [186]. Since the eleven sessions were conducted independently, there is a possibility of misalignment of their quality scales. In order to alleviate the problem, we performed a separate experiment for realignment, where ten videos from each session were collected as test stimuli. The videos chosen from each session roughly covered the entire quality range for that session.

Post-processing: The raw subjective scores are converted to Z-scores. We remove the ratings of streaming videos where each rebuffering event is not associated with an keystroke. In addition to the outlier removal scheme suggested in [100], we remove subjects who failed to accurately perform 10% of the auxiliary task, leaving 92 valid subjects. The results of the realignment experiment were used to map the Z-scores to MOS in accordance with [186]. Specifically, we assume a linear mapping between Z-scores and MOS. The coefficients are learnt by minimizing the prediction residual. One mapping is learnt for the experiment on each day and applied to the Z-scores of all videos in the respective sessions to produce the realigned MOS for the whole database.

The standard deviation of opinion scores and the mean SRCC between individual subject ratings and the MOSs are 17.35 and 0.67, respectively. The mean SRCC between individual ratings and the MOSs in the WaterlooSQoE-IV database is relatively lower than standard subjective video quality assessment database. To quantitatively analyze the behaviour, we compare the mean SRCC in the WaterlooSQoE-IV database to the ones in the WaterlooSQoE-I, WaterlooSQoE-II, and WaterlooSQoE-III databases. Although we would like to extend our analysis to other databases, neither the individual subjective ratings nor

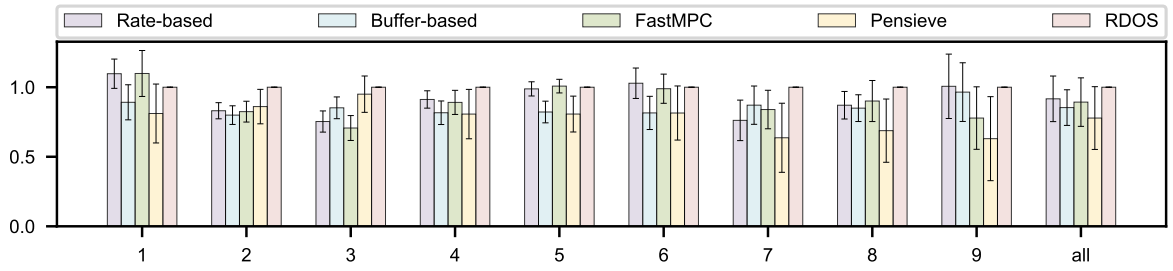


Figure 7.10: Performance of ABR algorithms on each testing network trace. Results are normalized against the performance of RDOS. Error bars span ± one standard deviation from the average.

existing data analysis are available in other streaming video QoE datasets. We believe that the low inter-subject consistency arises from two aspects. First, it has been observed that the inter-subject consistency negatively correlates with the number of stimuli under the same experimental protocol [53, 54, 199]. The claim is supported by our small-scale pilot study, which exhibits an inter-subject correlation of 0.79 under the same experimental protocol. Nevertheless, we find that the mean SRCC in the WaterlooSQoE-IV database is comparable to the WaterlooSQoE-III (mean SRCC of 0.667), which is considerably smaller in size. This suggests that the auxiliary task and additional alignment experiment helped mitigate the fatigue effect to some extent. Second, the low inter-subject consistency may also be a consequence of the intrinsic heterogeneity in the presence of diverse distortion patterns. The argument is evident by the high inter-subject consistency discrepancy between the databases with hand-crafted distortion (WaterlooSQoE-I and WaterlooSQoE-II) and the databases with realistic distortions (WaterlooSQoE-I and WaterlooSQoE-II).

7.3.3 Performance of ABR Algorithms

We evaluate the performance of ABR algorithms in two aspects. First, we are interests in how RDOS compare to other ABR algorithms in terms of subjective QoE, which is typically considered as the ultimate goal of ABR streaming. To this regard, we evaluate the performance of ABR algorithms with average MOS. Second, we are not only interested in the streaming video QoE, but also the amount of bitrate resource it takes to achieve

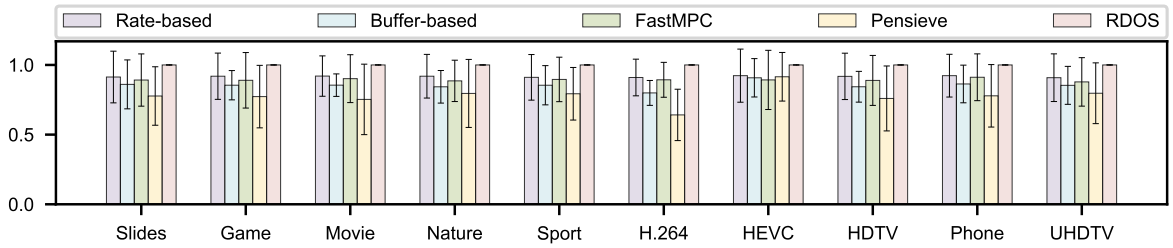


Figure 7.11: Performance of ABR algorithms on each content, video codec, and viewing device. Results are normalized against the performance of RDOS. Error bars span ± one standard deviation from the average.

a certain QoE level. For two ABR algorithms resulting in the same QoE, the algorithm consumes less bitrate should be considered better than the other. In order to capture this property, we adopt the bitrate saving measure in (7.1) as the evaluation criteria.

Performance in MOS

Figure 7.10 and 7.11 show the MOS that each ABR scheme receives across each dimension. Figure 7.12 provides the cumulative distribution of MOS attained by the ABR algorithms on the WaterlooSQoE-IV database. There are three key takeaways from these results. First, we find that RDOS exceeds the performance of the best existing ABR algorithm with a sizable margin on almost all scenarios considered. RDOS achieves 10% and 30% performance gain over the second best algorithm rate-based and its bitrate-driven counterpart Pensieve, respectively. Second, it is surprising that buffer-based, FastMPC, and Pensieve are inferior to the de facto rate-based algorithm on average. This is in sharp contrast to the significant gains claimed in existing studies using one or few test samples of hand-picked video clips and network traces, and verified with casual testing. Our results suggest that a better QoE model, or a better understanding of the human perceptual experiences, is an essential and dominating factor in improving ABR algorithms, as opposed to advanced optimization frameworks, machine learning strategies, or bandwidth predictors, where a majority of ABR research has been focused on in the past decade. Third, FastMPC outperforms another data-driven algorithm Pensieve, despite the common objective func-

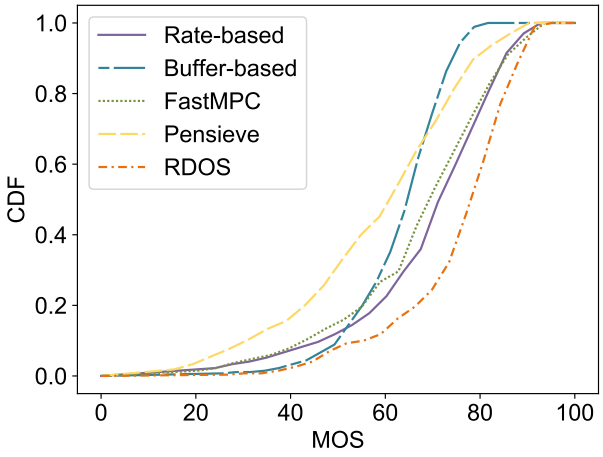


Figure 7.12: Cumulative distribution function of MOS generated from five competing ABR algorithms.

tion. The first interpretation of the phenomenon is that the convolutional neural network architecture cannot well characterize the vast diversity of network conditions, leading to sub-optimal bitrate selection. Another possible explanation could be that despite a more accurate throughput prediction, the enormous difference between the objective QoE prediction and subjective QoE response results in misplacement of bitrate resources. The real cause remains unclear because Pensieve only provides implicit throughput prediction. Fourth, RDOS demonstrates the most notable performance gain at low-bandwidth conditions. The difference may be explained by the perceptually motivated QoE model employed by RDOS, whereby positive adaptations are preferred over negative adaptations. Fifth, the rate-based algorithm and FastMPC perform at least on par with the best algorithm RDOS on network traces with small variation such as traces 5, 6, and 9, suggesting the (implicit) data-driven throughput prediction does not always lead to the optimal bitrate selection. In such cases, future ABR algorithms may exploit the connection-level information to reduce the uncertainty of future throughput [3, 237]. Sixth, although source content and viewing device have relatively little influences, the performance of ABR algorithms varies significantly over different video codecs. Consequently, the reported gain in the existing studies obtained on a single encoder does not generalize to other settings. At last, not a single

Table 7.3: Statistical Significance Matrix Based on Wilcoxon-Statistics on the WaterlooSQoE-IV Dataset. A Symbol "1" Means That the Performance of the Row Algorithm Is Statistically Better Than That of the Column Algorithm, A Symbol "0" Means That the Row Algorithm Is Statistically Worse, and A Symbol "-" Means That the Row and Column Algorithms Are Statistically Indistinguishable

	Rate-based	Buffer-based	FastMPC	Pensieve	RDOS
Rate-based	-	1	-	1	0
Buffer-based	0	-	0	1	0
FastMPC	-	1	-	1	0
Pensieve	0	0	0	-	0
RDOS	1	1	1	1	-

algorithm provides the best perceptual quality under all network profiles. This suggests that there is still room for future improvement.

To ascertain the performance difference among ABR algorithms is statistically significant, we carry out a statistical significance analysis. The evaluation statistic is the Wilcoxon signed-rank test. The null hypothesis is that the sample produced by a pair of ABR algorithms come from the same distribution. In particular, it tests whether the distribution of the differences is symmetric about zero (with 95% confidence). The results are summarized in Table 7.3, where a symbol '1' means the row algorithm performs significantly better than the column algorithm, a symbol '0' means the opposite, and a symbol '-' indicates that the row and column schemes are statistically indistinguishable. It can be observed that buffer-based and Pensieve algorithms are statistically inferior to the naïve rate-based algorithm, while RDOS is significantly better than all competing algorithms, confirming the importance of perceptual QoE modeling.

Performance in Bitrate Saving

Table 7.4 summarizes the bitrate saving of RDOS with MOS as the QoE measure. In general, our results echo the outcomes from the objective evaluation. We observe a slight

Table 7.4: Percentage of bitrate saving of column model vs. row model with MOS as the QoE measure.

	BB	FastMPC	RB	Pensieve	RDOS
BB	0	-9.91	-7.02	-47.42	**-51.95**
FastMPC	-	0	21.30	-25.41	**-39.65**
RB	-	-	0	-24.45	**-39.56**
Pensieve	-	-	-	0	**-6.17**
RDOS	-	-	-	-	0

decrease in the bitrate saving of RDOS over the existing ABR algorithms. This is not surprising because we use the proposed objective QoE index both in training and testing, which inevitably introduces biases favoring RDOS. Nevertheless, RDOS outperforms the best competitor algorithm by 6.17% on the aforementioned test set.

Chapter 8

Conclusion

We present rate-distortion optimization as a motivating principle for the design of ABR algorithms. The new RDOS paradigm not only delivers better resource allocation efficiency, but also uncovers a secret trade between video consumers and service providers, based on which streaming video business are made possible. In a free market, consumers pay the video service provider for their desired QoE, while the latter invests money on bitrate resources, which are in turn traded for perceptual quality of digital videos. Lying at the heart of the virtual market, ABR algorithms have a much profound impact in the video delivery ecosystem than what the bitrate maximization paradigm suggests. We believe that the proposed RDOS principle paves the way to the next-generation network resource allocation methods that are economically efficient and theoretically grounded.

To fully exploit the potential of RDOS, we delve into the three functional components underpinning ABR algorithms, including objective QoE model, throughput predictor, and bitrate selector. We take a Bayesian approach to formulate the three problems, effectively resolving the strong inductive bias induced by traditional expert models and the overfitting problem that comes along with the recently popularized data-driven methods. Depending on the amount of available data in each domain, the prior distribution is either extracted from expert knowledge or learnt by the empirical Bayes' method. The resulting system achieves highly competitive performance against the state-of-the-art, which is supported by so-far the largest objective and subjection evaluations.

Throughput history, most seminal scientific works challenge very fundamentals of conventional wisdom, propose novel perspective to a long standing problem, and develop mathematical theorems/models around the basic intuition. When it comes to the field of engineering, this research principle usually leads to a more comprehensive and meaningful design objective, which is the main lesson that I have learnt through my research. In particular, I have been inspired to think through the true objective of a system before blindly pursuing a "better" solution within a traditional framework. I have tried to apply the principle in the design of ABR systems, and the solution advanced in this dissertation is RDOS.

The second lesson that I have learnt during my investigation is to appreciate the importance of prior knowledge in the design of visual communication systems. This does not suggest the value of training data should be diminished. Rather, I hope to make the point that the key is to adopt the most appropriate prior knowledge according to the problem setting. In situations that there is a major conflict between the limited training data and the huge dimension of input space, meaningful subjective prior knowledge can provide strong regularization effect to the maximum likelihood-based solution, which is evident by the state-of-the-art performance of BSQI. Even in the domain with abundant training data, an objective prior empowered by empirical Bayes' method could lead to significant improvements as what have been observed in MetaTP and EERO.

In adaptive streaming, one of the most promising areas for future work is developing better objective QoE models, where we only have very limited understanding of HVS and sparsely distributed training data. The BSQI is an instantiation of the bottom-up approach, simulating the function of relevant early-stage components in the HVS. Future approaches should explore the top-down approach, mimicking the hypothesized functionality of the overall HVS. For example, it is highly promising to adapt the feature extractor optimized for other visual tasks to the QoE prediction according to our previous success with image/video quality assessment [50, 123, 128].

There is little doubt that accurate throughput prediction could lead to improved bitrate decision. The MetaTP makes one of the first attempt in this direction based on the combination of connection-level fast adaptation and empirical Bayes prior learning. However, the proposed model is by no means the optimal throughput predictor. Specifically, MetaTP

makes use of the connection-level throughput data across a couple of streaming sessions to adapt the model to the target network characteristics, during which ABR algorithms have to fall back to the user agnostic throughput predictor. The new throughput observation required in the fast adaptation stage may be minimized in future works.

With the rise of the data economy, data privacy has become a major concern in multimedia communication applications. Despite the potential benefits of customized services, video consumers are inclined not to release their private information such as the geometry, IP address, and throughput observations to service providers. In practice, many video content providers have to deliver personalized services without invasion of privacy, which strongly affects the application scope of the proposed MetaTP. To this end, federated learning [109] may be a promising alternative to generate more flexible models that adapt to individual users while protecting their privacy.

Future ABR research may also benefit from a better evaluation procedure, which can reliably evaluate the performance of ABR algorithms in the wild and at scale. To this end, crowd-sourcing has demonstrated strong promise in very large-scale data collection involving human responses such as image recognition [39], scene parsing [177], and image quality assessment [69, 90]. It remains to be seen how these techniques can be effectively adapted to QoE evaluation.

Although several efforts attempt to simultaneously maximize the overall QoE and fairness across multiple adaptive streaming players that compete at bottleneck links [102, 119], our economic interpretation of RDOS suggests that there exists a fundamental conflict between optimality and equity [130]. It would be interesting to build a quantitatively relationship between efficiency and equity, and implement practical solutions to achieve desired tradeoff.

We present a decentralized instantiation of the RDOS system in the thesis, which performs QoE assessment, throughput prediction, and bitrate selection in the video player. Nevertheless, the general RDOS paradigm can also be applied to a server-side or hybrid adaptive streaming system. Such collaborative designs have already begun to make some impacts in the practical video delivery frameworks [15]. We expect future works to extend the new design paradigm at each operational points across the transmission pipeline, where

different information is available to facilitate more efficient resource allocation.

The study of ABR algorithms in the thesis is a promising start for scientifically investigating many longstanding and emerging problems in the perceptually oriented resource management. Both rate-distortion theory and Bayesian theory may find their application in many other network resource allocation problems such as TCP congestion control, workstation management, and general web services. With the challenge of meeting the growing consumer demand using limited resources, I hope the study in the thesis can shed light on the long-neglected problem, and provide a new perspective to the research community and the industry.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[2] S. Akhshabi, A. C. Begen, and C. Dovrolis. An experimental evaluation of rate-adaptation algorithms in adaptive streaming over HTTP. In *Proc. ACM Conf. Multimedia Systems*, pages 157–168, 2011.

[3] Z. Akhtar, Y. S. Nam, R. Govindan, S. Rao, J. Chen, E. Katz-Bassett, B. Ribeiro, J. Zhan, and H. Zhang. Oboe: Auto-tuning video ABR algorithms to network conditions. In *Proc. ACM SIGCOMM*, pages 44–58, Budapest, Hungary, Aug. 2018.

[4] S. Athar, T. Costa, K. Zeng, and Z. Wang. Perceptual quality assessment of UHD-HDR-WCG videos. In *Proc. IEEE Int. Conf. Image Proc.*, pages 1740–1744, 2019.

[5] S. Athar and Z. Wang. A comprehensive performance evaluation of image quality assessment algorithms. *IEEE Access*, 7:140030–140070, 2019.

[6] L. Atzori, A. Floris, G. Ginesu, and D. D. Giusto. Quality perception when streaming video on tablet devices. *Journal of Visual Comm. and Image Representation*, 25(3):586–595, Apr. 2014.

[7] B. Bakker. Reinforcement learning with long short-term memory. In *Nerual Information Processing Systems*, pages 1475–1482, Vancouver, BC, Canada, Dec. 2002.

[8] C. G. Bampis and A. C. Bovik. Learning to predict streaming video QoE: Distortions, rebuffering and memory. *ArXiv*, abs/1703.00633, Mar. 2017.

[9] C. G Bampis, A. C. Bovik, and Z. Li. A simple prediction fusion improves data-driven full-reference video quality assessment models. In *Picture Coding Symposium*, pages 298–302, San Francisco, CA, USA, Jun. 2018.

[10] C. G. Bampis, Z. Li, and A. C. Bovik. Continuous prediction of streaming video QoE using dynamic networks. *IEEE Signal Processing Letters*, 24(7):1083–1087, Jul. 2017.

[11] C. G. Bampis, Z. Li, I. Katsavounidis, T. Y. Huang, C. Ekanadham, and A. C. Bovik. Towards perceptually optimized end-to-end adaptive video streaming. *ArXiv preprint arXiv:1808.03898*, Aug. 2018.

[12] C. G. Bampis, Z. Li, A. K. Moorthy, I. Katsavounidis, A. Aaron, and A. C. Bovik. Study of temporal effects on subjective video Quality of Experience. *IEEE Trans. Image Processing*, 26(11):5217–5231, Nov. 2017.

[13] H. B. Barlow et al. Possible principles underlying the transformation of sensory messages. *Sensory Communication*, 1(01), 1961.

[14] P. G. Barten. *Contrast sensitivity of the human eye and its effects on image quality*. SPIE press, 1999.

[15] A. Bentaleb. *Enabling optimization of video delivery in HTTP adaptive streaming*. PhD thesis, National University of Singapore, 2019.

[16] A. Bentaleb, A. C. Begen, and R. Zimmermann. SDNDASH: Improving QoE of HTTP adaptive streaming using software defined networking. In *Proc. ACM Int. Conf. Multimedia*, pages 1296–1305, Amsterdam, The Netherlands, Oct. 2016.

[17] T. Berger. *Rate Distortion Theory: A Mathematical Basis for Data Compression*. Prentice-Hall electrical engineering series. Prentice-Hall, Upper Saddle River, NJ, USA, 1971.

[18] J. M. Bernardo and A. F. Smith. *Bayesian theory*, volume 405. John Wiley & Sons, 2009.

[19] C. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, Berlin, Heidelberg, 2006.

[20] G. Bjøntegaard. Calculation of average PSNR differences between RD-curves. Video Coding Experts Group, VCEG-M33, ITU-T SG 16/Q6, 13th VCEG Meeting, Apr. 2001.

[21] Y. Blau and T. Michaeli. The perception-distortion tradeoff. In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, pages 6228–6237, 2018.

[22] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, Jul. 2011.

[23] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang. JAX: Composable transformations of Python+NumPy programs, 2018.

[24] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, Oct. 2001.

[25] C. Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, Jun. 1998.

[26] CDN77. CDN pricing & features comparison 2021, May 2021. https://www.cdn77.com/compare-cdn-providers?gclid=Cj0KCQjw2NyFBhDoARIsAMtHtZ6oVCSvWBscY6yXmE6MCKJofp8g_ZZjR9EbrJ4UYglIJcj5wnD3Q3YaAkUqEALw_wcB.

[27] C. Chen, L. K. Choi, G. de Veciana, C. Caramanis, R. W. Heath, and A. C. Bovik. Modeling the time-varying subjective quality of HTTP video streams with rate adaptations. *IEEE Trans. Image Processing*, 23(5):2206–2221, Mar. 2014.

[28] Y. Chen, D. Murherjee, J. Han, A. Grange, Y. Xu, Z. Liu, S. Parker, C. Chen, H. Su, U. Joshi, et al. An overview of core coding tools in the AV1 video codec. In *Proc. IEEE Picture Coding Symp.*, pages 41–45, 2018.

[29] M. Cheon and J. Lee. Subjective and objective quality assessment of compressed 4K UHD videos for immersive experience. *IEEE Trans. Circuits and Systems for Video Tech.*, 28(7):1467–1480, Jul. 2018.

[30] F. Chiariotti, S. D'Aronco, L. Toni, and P. Frossard. Online learning adaptation strategy for DASH clients. In *Proc. ACM Conf. Multimedia Systems*, pages 1–12, Klagenfurt, Austria, May 2016.

[31] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. In *Syntax, Semantics and Structure in Statistical Translation Workshop*, pages 103–111, Doha, Qatar, Sep. 2014.

[32] M. Claeys, S. Latré, J. Famaey, T. Wu, W. Van Leekwijck, and F. De Turck. Design of a Q-learning-based client quality selection algorithm for HTTP adaptive video streaming. In *Adaptive and Learning Agents Workshop*, pages 30–37, Saint Paul, Minn., USA, May 2013.

[33] M. Claeys, S. Latré, J. Famaey, T. Wu, W. Van Leekwijck, and F. De Turck. Design and optimisation of a (FA) Q-learning-based HTTP adaptive streaming client. *Connection Science*, 26(1):25–43, 2014.

[34] R. Coulom. Efficient selectivity and backup operators in Monte-Carlo tree search. In *Proc. Int. Conf. Computers and Games*, pages 72–83, 2006.

[35] L. De Cicco, V. Caldaralo, V. Palmisano, and S. Mascolo. Elastic: A client-side controller for dynamic adaptive streaming over HTTP (DASH). In *Proc. IEEE Int. Packet Video Workshop*, pages 1–8, San Jose, CA, USA, Dec. 2013.

[36] J. De Cock, Z. Li, M. Manohara, and A. Aaron. Complexity-based consistent-quality encoding in the cloud. In *Proc. IEEE Int. Conf. Image Proc.*, pages 1484–1488, Phoenix, AZ, USA, Sep. 2016.

[37] B. De Finetti. *Theory of probability: A critical introductory treatment*, volume 6. John Wiley & Sons, 2017.

[38] J. Dean. Designs, lessons and advice from building large distributed systems. *Keynote from LADIS*, 1, 2009.

[39] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li. Imagenet: A large-scale hierarchical image database. In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[40] S. Deshpande. Subjective and objective visual quality evaluation of 4K video using AVC and HEVC compression. In *SID Symposium Digest of Technical Papers*, pages 481–484, 2012.

[41] P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, Y. Wu, and P. Zhokhov. Openai baselines. https://github.com/openai/baselines, May 2017.

[42] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. Joseph, A. Ganjam, J. Zhan, and H. Zhang. Understanding the impact of video quality on user engagement. *ACM SIGCOMM Computer Comm. Review*, 41(4):362–373, Aug. 2011.

[43] Z. Duanmu, D. Chen, Z. Li, W. Liu, Z. Wang, Y. Wang, and W. Gao. Waterloo streaming Quality-of-Experience database-IV, Jun. 2019. http://ece.uwaterloo.ca/~zduanmu/waterloosqoe4.

[44] Z. Duanmu, W. Liu, D. Chen, Z. Li, , Z. Wang, Y. Wang, and W. Gao. Pairwise comparison of objective QoE models via analysis-by-synthesis, 2019. http://ece.uwaterloo.ca/~zduanmu/ksqi/demo.

[45] Z. Duanmu, W. Liu, D. Chen, Z. Li, Z. Wang, Y. Wang, and W. Gao. A knowledge-driven Quality-of-Experience model for adaptive streaming videos. *arXiv preprint arXiv:1911.07944*, 2019.

[46] Z. Duanmu, W. Liu, Z. Li, D. Chen, Z. Wang, Y. Wang, and W. Gao. Assessing the Quality-of-Experience of adaptive bitrate video streaming. *arXiv preprint arXiv:2008.08804*, 2020.

[47] Z. Duanmu, W. Liu, Z. Li, K. Ma, and Z. Wang. Characterizing generalized rate-distortion performance of video coding: An eigen analysis approach. *IEEE Trans. Image Processing*, 29:6180–6193, 2020.

[48] Z. Duanmu, W. Liu, Z. Li, and Z. Wang. Modeling generalized rate-distortion functions. *IEEE Trans. Image Processing*, 29:7331–7344, 2020.

[49] Z. Duanmu, W. Liu, Z. Li, and Z. Wang. Modeling generalized rate-distortion functions. *IEEE Trans. Image Processing*, 29:7331 – 7344, Jun. 2020.

[50] Z. Duanmu, W. Liu, Z. Wang, and Z. Wang. Quantifying visual image quality: A Bayesian view. *Annual Review of Vision Science*, Sep. 2021, To Appear.

[51] Z. Duanmu, K. Ma, and Z. Wang. Quality-of-Experience of adaptive video streaming: Exploring the space of adaptations. In *Proc. ACM Int. Conf. Multimedia*, pages 1752–1760, Mountain View, CA, USA, Oct. 2017.

[52] Z. Duanmu, K. Ma, and Z. Wang. Quality-of-Experience for adaptive streaming videos: An expectation confirmation theory motivated approach. *IEEE Trans. Image Processing*, 27(12):6135–6146, Dec. 2018.

[53] Z. Duanmu, A. Rehman, and Z. Wang. A Quality-of-Experience database for adaptive video streaming. *IEEE Trans. Broadcasting*, 64(2):474–487, Jun. 2018.

[54] Z. Duanmu, K. Zeng, K. Ma, A. Rehman, and Z. Wang. A Quality-of-Experience index for streaming video. *IEEE Journal of Selected Topics in Signal Processing*, 11(1):154–166, Sep. 2017.

[55] H. Ebbinghaus. *Memory: A contribution to experimental psychology.* Teachers college, Columbia university, Oct. 1913.

[56] S. Egger and A. Raake. *Quality and Quality of Experience.* Springer-Verlag, 2014.

[57] Encoding.com. Microsoft smooth streaming, 2016.

[58] N. Eswara, S. Ashique, A. Panchbhai, S. Chakraborty, H. P. Sethuram, K. Kuchi, A. Kumar, and S. S. Channappayya. Streaming video QoE modeling and prediction: A long short-term memory approach. *IEEE Trans. Circuits and Systems for Video Tech.*, 30(3):661–673, Mar. 2020.

[59] N. Eswara, K. Manasa, A. Kommineni, S. Chakraborty, H. P. Sethuram, K. Kuchi, A. Kumar, and S. S. Channappayya. A continuous QoE evaluation framework for video streaming over HTTP. *IEEE Trans. Circuits and Systems for Video Tech.*, 28(11):3236–3250, Aug. 2017.

[60] Federal Communications Commission. Raw data - measuring broadband america, Aug. 2016. https://www.fcc.gov/reports-research/reports/.

[61] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proc. Int. Conf. on Machine Learning*, pages 1126–1135, 2017.

[62] A. Floris, L. Atzori, G. Ginesu, and D.D. Giusto. QoE assessment of multimedia video consumption on tablet devices. In *IEEE Globecom Workshops*, pages 1329–1334, Anaheim, CA, USA, Dec. 2012.

[63] DASH Industry Forum. For promotion of MPEG-DASH 2012, Apr. 2012. http://dashif.org.

[64] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept drift adaptation. *ACM computing surveys*, 46(4):44.1–44.37, Apr. 2014.

[65] A. Ganjam, F. Siddiqui, J. Zhan, X. Liu, I. Stoica, J. Jiang, V. Sekar, and H. Zhang. C3: Internet-scale control plane for video quality optimization. In *USENIX Symp. on Networked Systems Design and Implementation*, pages 131–144, 2015.

[66] Y. Gao, X. Wei, and L. Zhou. Personalized QoE improvement for networking video service. *IEEE Journal on Selected Areas in Communications*, 2020, To Appear.

[67] M. N. Garcia, F. De Simone, S. Tavakoli, N. Staelens, S. Egger, K. Brunnström, and A. Raake. Quality of Experience and HTTP adaptive streaming: A review of subjective studies. In *Proc. IEEE Int. Workshop Quality of Multimedia Experience*, pages 141–146, Singapore, Singapore, Sep. 2014.

[68] M. N. Garcia, D. Dytko, and A. Raake. Quality impact due to initial loading, stalling, and video bitrate in progressive download video services. In *Proc. IEEE Int. Workshop Quality of Multimedia Experience*, pages 129–134, Singapore, Singapore, Sep. 2014.

[69] D. Ghadiyaram and A. C. Bovik. Massive online crowdsourced study of subjective and objective picture quality. *IEEE Trans. Image Processing*, 25(1):372–387, 2015.

[70] D. Ghadiyaram, A. C. Bovik, H. Yeganeh, R. Kordasiewicz, and M. Gallant. Study of the effects of stalling events on the Quality of Experience of mobile streaming videos. In *Proc. IEEE Global Conf. Signal and Information Processing*, pages 989–993, Atlanta, GA, USA, Dec. 2014.

[71] D. Ghadiyaram, J. Pan, and A. C. Bovik. A subjective and objective study of stalling events in mobile streaming videos. *IEEE Trans. Circuits and Systems for Video Tech.*, 29(1):183–197, 2017.

[72] M. Grafl and C. Timmerer. Representation switch smoothing for adaptive HTTP streaming. In *Proc. IEEE Int. Workshop Perceptual Qualtiy of Systems*, pages 178–183, Vienna, Austria, Sep. 2013.

[73] E. Grant, C. Finn, S. Levine, T. Darrell, and T. Griffiths. Recasting gradient-based meta-learning as hierarchical bayes. *arXiv preprint arXiv:1801.08930*, 2018.

[74] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.1, Mar. 2014. http://cvxr.com/cvx.

[75] L. M. Graves. Riemann integration and Taylor's theorem in general analysis. *Transactions of the American Mathematical Society*, 29(1):163–177, 1927.

[76] D. Grois, D. Marpe, A. Mulayoff, B. Itzhaky, and O. Hadar. Performance comparison of H.265/MPEG-HEVC, VP9, and H.264/MPEG-AVC encoders. In *Picture Coding Symposium*, pages 394–397, 2013.

[77] Video Quality Experts Group. Final report from the video quality experts group on the validation of objective models of video quality assessment, phase ii. *2003 VQEG*, 2003.

[78] M. T. Hagan, H. B. Demuth, M. H. Beale, and R. De Jesús. *Neural network design*. PWS Publishing Co., Boston, MA, USA, 1996.

[79] D. S. Hands and S. E. Avons. Recency and duration neglect in subjective assessment of television picture quality. *Applied Cognitive Psychology*, 15(6):639–657, Nov. 2001.

[80] Q. He, C. Dovrolis, and M. Ammar. On the predictability of large transfer TCP throughput. *ACM SIGCOMM Computer Comm. Review*, 35(4):145–156, 2005.

[81] D. J. Heeger. Normalization of cell responses in cat striate cortex. *Visual Neuroscience*, 9(2):181–197, 1992.

[82] R. S. Hiller. Profitably bundling information goods: Evidence from the evolving video library of netflix. *Journal of Media Economics*, 30(2):65–81, 2017.

[83] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[84] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, Jan. 1989.

[85] T. Hoßfeld, S. Egger, R. Schatz, M. Fiedler, K. Masuch, and C. Lorentzen. Initial delay vs. interruptions: Between the devil and the deep blue sea. In *Proc. IEEE Int. Workshop Quality of Multimedia Experience*, pages 1–6, Yarra Valley, VIC, Australia, Jul. 2012.

[86] T. Hoßfeld, R. Schatz, E. Biersack, and L. Plissonneau. Internet video delivery in YouTube: From traffic measurements to Quality of Experience. In *Data Traffic Monitoring and Analysis*, pages 264–301. Springer, Berlin, Heidelberg, Jan. 2013.

[87] T. Hoßfeld, M. Seufert, M. Hirth, T. Zinner, P. Tran-Gia, and R. Schatz. Quantification of YouTube QoE via crowdsourcing. In *Proc. IEEE Int. Sym. Multimedia*, pages 494–499, Dana Point, CA, USA, Dec. 2011.

[88] T. Hoßfeld, M. Seufert, C. Sieber, and T. Zinner. Assessing effect sizes of influence factors towards a QoE model for HTTP adaptive streaming. In *Proc. IEEE Int. Workshop Quality of Multimedia Experience*, pages 111–116, Singapore, Singapore, Sep. 2014.

[89] T. Hossfeld, D. Strohmeier, A. Raake, and R. Schatz. Pippi longstocking calculus for temporal stimuli pattern on Youtube QoE: 1+ 1= 3 and $1 \cdot 4 \neq 4 \cdot 1$. In *Proc. Workshop on Mobile Video*, pages 37–42, 2013.

[90] V. Hosu, H. Lin, T. Sziranyi, and D. Saupe. KonIQ-10k: An ecologically valid database for deep learning of blind image quality assessment. *IEEE Trans. Image Processing*, 29:4041–4056, 2020.

[91] T. Huang, R. Zhang, C. Zhou, and L. Sun. QARC: Video quality aware rate control for real-time video streaming based on deep reinforcement learning. In *Proc. ACM Int. Conf. Multimedia*, pages 1208–1216, Seoul, Republic of Korea, May 2018.

[92] T. Huang, C. Zhou, R. Zhang, C. Wu, X. Yao, and L. Sun. Comyco: Quality-aware adaptive video streaming via imitation learning. In *Proc. ACM Int. Conf. Multimedia*, pages 429–437, 2019.

[93] T. Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson. A buffer-based approach to rate adaptation: Evidence from a large video streaming service. *ACM SIGCOMM Computer Comm. Review*, 44(4):187–198, Feb. 2015.

[94] L. Huo, Z. Wang, M. Xu, Y. Li, Z. Ding, and H. Wang. A meta-learning framework for learning multi-user preferences in QoE optimization of DASH. *IEEE Trans. Circuits and Systems for Video Tech.*, 2020, To Appear.

[95] Apple Inc. Best practices for creating and deploying HTTP live streaming media for iPhone and iPad, Jul. 2013. http://is.gd/LBOdpz.

[96] Apple Inc. Apple unleashes M1, May 2021. https://www.apple.com/ca/newsroom/2020/11/apple-unleashes-m1/.

[97] Google Inc. Tensorflow.js: A JavaScript library for training and deploying ML models in the browser and on Node.js, 2018.

[98] S. Ioffe and C.n Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456, 2015.

[99] I. Ishii, T. Tatebe, Q. Gu, Y. Moriue, T. Takaki, and K. Tajima. 2000 fps real-time vision system with high-frame-rate video recording. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 1536–1541, Anchorage, AK, USA, May 2010.

[100] ITU-R BT.500-12. Recommendation: Methodology for the subjective assessment of the quality of television pictures, Nov. 1993. https://forum.ddigest.com/attachments/f4/16001d1276428339-subjective-video-quality-need-help-recomendation_500_itu-r.pdf.

[101] J. Jiang, V. Sekar, H. Milner, D. Shepherd, I. Stoica, and H. Zhang. CFA: A practical prediction system for video QoE optimization. In *USENIX Symp. on Networked Systems Design and Implementation*, pages 137–150, 2016.

[102] J. Jiang, V. Sekar, and H. Zhang. Improving fairness, efficiency, and stability in HTTP-based adaptive video streaming with FESTIVE. *IEEE/ACM Transactions on Networking*, 22(1):326–340, Feb. 2014.

[103] P. Juluri, V. Tamarapalli, and D. Medhi. SARA: Segment aware rate adaptation algorithm for dynamic adaptive streaming over HTTP. In *Proc. IEEE Int. Conf. Comm. Workshop*, pages 1765–1770, 2015.

[104] P. Kafka. You can watch Netflix on any screen you want, but you're probably watching it on a TV, Mar. 2018. https://www.recode.net/2018/3/7/17094610/netflix-70-percent-tv-viewing-statistics.

[105] S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski. High dynamic range video. *ACM Trans. of Graphics*, 22(3):319–325, Jul. 2003.

[106] H. J. Kim, D. G. Yun, H. Kim, K. S. Cho, and S. G. Choi. QoE assessment model for video streaming service using QoS parameters in wired-wireless network. In *Proc. IEEE Int. Conf. Advanced Comm. Technology*, pages 459–464, 2012.

[107] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *ArXiv preprint arXiv:1412.6980*, Dec. 2014.

[108] D. C. Knill and W. Richards. *Perception as Bayesian inference*. Cambridge University Press, 1996.

[109] J. Konečnỳ, B. McMahan, and D. Ramage. Federated optimization: Distributed optimization beyond the datacenter. *arXiv preprint arXiv:1511.03575*, 2015.

[110] C. Kreuzberger, B. Rainer, H. Hellwagner, L. Toni, and P. Frossard. A comparative study of DASH representation sets using real user characteristics. In *Int. Workshop Network and OS Support for Digital Audio and Video*, pages 1–4, 2016.

[111] P. S. Laplace. Memoir on the probability of the causes of events. *Statistical science*, 1(3):364–378, 1986.

[112] J. Le Feuvre, C. Concolato, and J. Moissinac. GPAC: Open source multimedia framework. In *Proc. ACM Int. Conf. Multimedia*, pages 1009–1012, Augsburg, Bavaria, Germany, Sep. 2007.

[113] S. Lederer. Optimal adaptive streaming formats mpeg-dash & hls segment length, 2015.

[114] S. Lederer. Optimal adaptive streaming formats MPEG-DASH & HLS segment length, Apr. 2020. https://bitmovin.com/mpeg-dash-hls-segment-length/.

[115] S. Lederer, C. Müller, and C. Timmerer. Dynamic adaptive streaming over HTTP dataset. In *Proc. ACM Conf. Multimedia Systems*, pages 89–94, Chapel Hill, NC, USA, Feb. 2012.

[116] B. Lewcio, B. Belmudez, A. Mehmood, M. Wältermann, and S. Möller. Video quality in next generation mobile networks-perception of time-varying transmission. In *IEEE Int. Workshop Technical Committee on Comm. Quality and Reliability*, pages 1–6, Naples, FL, May 2011.

[117] Z. Li, A. Aaron, L. Katsavounidis, A. Moorthy, and M. Manohara. Toward a practical perceptual video quality metric, Jun. 2016. http://techblog.netflix.com/2016/06/toward-practical-perceptual-video.html.

[118] Z. Li, Z. Duanmu, W. Liu, and Z. Wang. AVC, HEVC, VP9, AVS2, or AV1? - A comparative study of state-of-the-art video encoders on 4K videos. In *Proc. Int. Conf. Image Analysis and Recognition*, Waterloo, ON, Canada, 2019.

[119] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. C. Begen, and D. Oran. Probe and adapt: Rate adaptation for HTTP video streaming at scale. *IEEE Journal on Selected Areas in Comm.*, 32(4):719–733, Apr. 2014.

[120] J. Lin, R. Song, C. Wu, T. Liu, H. Wang, and C. J. Kuo. MCL-V: A streaming video quality assessment database. *Journal of Visual Comm. and Image Representation*, 30:1–9, 2015.

[121] L. Lin, S. Yu, L. Zhou, W. Chen, T. Zhao, and Z. Wang. PEA265: Perceptual assessment of video compression artifacts. *IEEE Trans. Circuits and Systems for Video Tech.*, 30(11):3898–3910, 2020.

[122] C. Liu, I. Bouazizi, and M. Gabbouj. Rate adaptation for adaptive HTTP streaming. In *Proc. ACM Conf. Multimedia Systems*, pages 169–174, 2011.

[123] W. Liu, Z. Duanmu, and Z. Wang. End-to-end blind quality assessment of compressed videos using deep neural networks. In *Proc. ACM Int. Conf. Multimedia*, pages 546–554, Seoul, Republic of Korea, Oct. 2018.

[124] X. Liu, F. Dobrian, H. Milner, J. Jiang, V. Sekar, I. Stoica, and H. Zhang. A case for a coordinated internet video control plane. *ACM SIGCOMM Computer Comm. Review*, 42(4):359–370, Sep. 2012.

[125] Y. Liu, S. Dey, F. Ulupinar, M. Luby, and Y. Mao. Deriving and validating user experience model for DASH video streaming. *IEEE Trans. Broadcasting*, 61(4):651–665, Dec. 2015.

[126] Y. Liu, Y. Shen, Y. Mao, J. Liu, Q. Lin, and D. Yang. A study on Quality of Experience for adaptive streaming service. In *Proc. IEEE Int. Conf. Comm. Workshop*, pages 682–686, Budapest, Hungary, Jun. 2013.

[127] K. Ma, Z. Duanmu, Z. Wang, Q. Wu, W. Liu, H. Yong, H. Li, and L. Zhang. Group maximum differentiation competition: Model comparison with few samples. *IEEE Trans. Pattern Analysis and Machine Intelligence*, pages DOI: 10.1109/TPAMI.2018.2889948, IEEE Xplore early access, 2019.

[128] K. Ma, W. Liu, K. Zhang, Z. Duanmu, Z. Wang, and W. Zuo. End-to-end blind image quality assessment using deep neural networks. *IEEE Trans. Image Processing*, 27(3):1202–1213, 2017.

[129] Z. Mai, C. Yang, K. Kuang, and L. Po. A novel motion estimation method based on structural similarity for H.264 inter prediction. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Processing*, pages 913–916, 2006.

[130] N. G. Mankiw. *Principles of Microeconomics*, chapter 14, pages 289–307. South-Western Cengage Learning, Boston, MA, United States, 5 edition, 2008.

[131] H. Mao, R. Netravali, and M. Alizadeh. Neural adaptive video streaming with Pensieve. In *Proc. ACM SIGCOMM*, pages 197–210, Los Angeles, CA, USA, Aug. 2017.

[132] Z. Meng, M. Wang, J. Bai, M. Xu, H. Mao, and H. Hu. Interpreting deep learning-based networking systems. In *Proc. ACM SIGCOMM*, pages 154–171, 2020.

[133] V. Menkovski and A. Liotta. Intelligent control for adaptive video streaming. In *Proc. IEEE Int. Conf. Cnsumer Electronics*, pages 127–128, 2013.

[134] K. Miller, E. Quacchio, G. Gennari, and A. Wolisz. Adaptation algorithm for adaptive streaming over HTTP. In *Proc. IEEE Int. Packet Video Workshop*, pages 173–178, 2012.

[135] M. Mirza, J. Sommers, P. Barford, and X. Zhu. A machine learning approach to TCP throughput prediction. *ACM SIGMETRICS Perform. Eval. Rev.*, 35(1):97–108, 2007.

[136] A. M. Mishra. *Quality of Service in Communications Networks*. Wiley, 2001.

[137] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proc. Int. Conf. on Machine Learning*, pages 1928–1937, 2016.

[138] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[139] R. K. Mok, E. W. Chan, and R. K. Chang. Measuring the Quality of Experience of HTTP video streaming. In *Proc. IFIP/IEEE Int. Sym. Integrated Network Management*, pages 485–492, Dublin, Ireland, May 2011.

[140] R. K. Mok, E. W. Chan, X. Luo, and R. K. Chang. Inferring the QoE of HTTP video streaming from user-viewing activities. In *Proc. ACM SIGCOMM Workshop*, pages 31–36, 2011.

[141] R. K. Mok, X. Luo, E. W. Chan, and R. K. Chang. QDASH: A QoE-aware DASH system. In *Proc. ACM Conf. Multimedia Systems*, pages 11–22, Chapel Hill, NC, USA, Feb. 2012.

[142] A. K. Moorthy, L. K. Choi, A. C. Bovik, and G. De Veciana. Video quality assessment on mobile devices: Subjective, behavioral and objective studies. *IEEE Journal of Selected Topics in Signal Processing*, 6(6):652–671, Oct. 2012.

[143] Mozilla. IndexedDB API, Apr. 2019. https://developer.mozilla.org/en-US/docs/Web/API/IndexedDB_API#Specifications.

[144] C. Müller, S. Lederer, and C. Timmerer. An evaluation of dynamic adaptive streaming over HTTP in vehicular environments. In *Proc. ACM Workshop on Mobile Video*, pages 37–42, 2012.

[145] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proc. Int. Conf. on Machine Learning*, page 807–814, 2010.

[146] R. M. Nasiri, J. Wang, A. Rehman, S. Wang, and Z. Wang. Perceptual quality assessment of high frame rate video. In *Proc. IEEE Int. Conf. Multimedia and Signal Processing*, pages 1–6, Xiamen, China, Oct. 2015.

[147] Netflix Inc. Per-title encode optimization, Dec. 2015. http://techblog.netflix.com/2015/12/per-title-encode-optimization.html.

[148] R. Netravali, A. Sivaraman, K. Winstein, S. Das, A. Goyal, and H. Balakrishnan. Mahimahi: A lightweight toolkit for reproducible web measurement. *ACM SIGCOMM Computer Comm. Review*, 44(4):129–130, Aug. 2014.

[149] R. Ng, M. Levoy, M. Brédif, G. Duval, M. Horowitz, and P. Hanrahan. Light field photography with a hand-held plenoptic camera. Computer Science Technical Report, Apr. 2005.

[150] P. Ni, R. Eg, A. Eichhorn, C. Griwodz, and P. Halvorsen. Flicker effects in adaptive video streaming to handheld devices. In *Proc. ACM Int. Conf. Multimedia*, pages 463–472, Scottsdale, AZ, USA, Nov. 2011.

[151] A. Nichol, J. Achiam, and J. Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[152] J. Nightingale, Q. Wang, C. Grecos, and S. Goma. The impact of network impairment on quality of experience (QoE) in H.265/HEVC video streaming. *IEEE Trans. Consumer Electronics*, 60(2):242–250, 2014.

[153] R. L. Oliver. A cognitive model of the antecedents and consequences of satisfaction decisions. *Journal of Marketing Research*, 17(4):460–469, Nov. 1980.

[154] B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Research*, 37(23):3311–3325, 1997.

[155] A. Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu. Conditional image generation with pixelcnn decoders. *arXiv preprint arXiv:1606.05328*, 2016.

[156] Y. Ou, Y. Xue, and Y. Wang. Q-STAR: A perceptual video quality model considering impact of spatial, temporal, and amplitude resolutions. *IEEE Trans. Image Processing*, 23(6):2473–2486, Jun. 2014.

[157] ITU-T P.1203. Parametric bitstream-based quality assessment of progressive download and adaptive audiovisual streaming services over reliable transport, Oct. 2017. https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-P.1203-201710-I!!PDF-E&type=items.

[158] ITU-T P.910. Recommendation: Subjective video quality assessment methods for multimedia applications, Sep. 1999. https://www.itu.int/rec/T-REC-P.910-199909-S.

[159] R. Pastrana-Vidal, J. C. Gicquel, C. Colomes, and H. Cherifi. Sporadic frame dropping impact on quality perception. In *Human Vision and Electronic Imaging*, pages 182–194, San Jose, CA, USA, Jun. 2004.

[160] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. https://github.com/pytorch/pytorch, Oct. 2017.

[161] P. Paudyal, F. Battisti, and M. Carli. Impact of video content and transmission impairments on Quality of Experience. *Multimedia Tools and Applications*, 75(23):16461–16485, 2016.

[162] M. H. Pinson and S. Wolf. A new standardized method for objectively measuring video quality. *IEEE Trans. Broadcasting*, 50(3):312–322, Sept. 2004.

[163] N. Ponomarenko, L. Jin, O. Ieremeiev, V. Lukin, K. Egiazarian, J. Astola, B. Vozel, K. Chehdi, M. Carli, and F. Battisti. Image database TID2013: Peculiarities, results and perspectives. *Signal Processing: Image Communication*, 30:57–77, Jan. 2015.

[164] S. J. Prince. *Computer vision: Models, learning, and inference*. Cambridge University Press, 2012.

[165] Y. Qi and M. Dai. The effect of frame freezing and frame skipping on video quality. In *Proc. IEEE Int. Conf. Intelligent Information Hiding and Multimedia Signal Processing*, pages 423–426, Pasadena, CA, USA, Dec. 2006.

[166] D. Raca, D. Leahy, C. J. Sreenan, and J. J. Quinlan. Beyond throughput, the next generation: A 5G dataset with channel and context metrics. In *Proc. ACM Conf. Multimedia Systems*, pages 303–308, 2020.

[167] D. Raca, J. J. Quinlan, A. H. Zahran, and C. J. Sreenan. Beyond throughput: A 4G LTE dataset with channel and context metrics. In *Proc. ACM Conf. Multimedia Systems*, pages 460–465, 2018.

[168] A. Rehman and Z. Wang. Perceptual experience of time-varying video quality. In *Proc. IEEE Int. Workshop Quality of Multimedia Experience*, pages 218–223, Klagenfurt am Worthersee, Jul. 2013.

[169] A. Rehman, K. Zeng, and Z. Wang. Display device-adapted video Quality-of-Experience assessment. In *Proc. SPIE*, pages 939406.1–939406.11, San Francisco, CA, USA, Feb. 2015.

[170] M. Řeřábek and T. Ebrahimi. Comparison of compression efficiency between HEVC/H.265 and VP9 based on subjective assessments. In *Applications of Digital Image Processing*, volume 9217, page 92170U, 2014.

[171] H. Riiser, P. Vigmostad, C. Griwodz, and P. Halvorsen. Commute path bandwidth traces from 3G networks: Analysis and applications. In *Proc. ACM Conf. Multimedia Systems*, pages 114–118, Oslo, Norway, Feb. 2013.

[172] D. C. Robinson, Y. Jutras, and V. Craciun. Subjective video quality assessment of HTTP adaptive streaming technologies. *Bell Labs Technical Journal*, 16(4):5–23, Mar. 2012.

[173] J. G. Robson. Spatial and temporal contrast-sensitivity functions of the visual system. *Journal of Optical Society of America*, 56(8):1141–1142, 1966.

[174] D. Z. Rodríguez, Z. Wang, R. L. Rosa, and G. Bressan. The impact of video-quality-level switching on user quality of experience in dynamic adaptive streaming over HTTP. *EURASIP Journal on Wireless Communications and Networking*, 2014(1):1–15, 2014.

[175] A. M. Rohaly, P. J. Corriveau, J. M. Libert, A. A. Webster, V. Baroncini, J. Beerends, J. Blin, L. Contin, T. Hamada, D. Harrison, et al. Video quality experts group: Current results and future directions. In *Proc. SPIE Visual Comm. and Image Processing*, volume 4067, pages 742–753, 2000.

[176] S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proc. Int. Conf. Artificial Intelligence and Statistics*, pages 627–635, 2011.

[177] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and F. Li. ImageNet large scale visual recognition challenge. *Int. Journal Computer Vision*, 115(3):211–252, 2015.

[178] A. Sackl, S. Egger, and R. Schatz. Where's the music? Comparing the QoE impact of temporal impairments between music and video streaming. In *Proc. IEEE Int. Workshop Quality of Multimedia Experience*, pages 64–69, Klagenfurt am Wörthersee, Austria, Jul. 2013.

[179] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging*, 2017(19):70–76, 2017.

[180] R. J. Santos. Equivalence of regularization and truncated iteration for general ill-posed problems. *Linear Algebra and Its Applications*, 236:25–33, 1996.

[181] K. Seshadrinathan and A. C. Bovik. Temporal hysteresis model of time varying subjective video quality. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Processing*, pages 1153–1156, Prague, Czech Republic, May 2011.

[182] K. Seshadrinathan, R. Soundararajan, A. C. Bovik, and L. K. Cormack. Study of subjective and objective quality assessment of video. *IEEE Trans. Image Processing*, 19(6):1427–1441, 2010.

[183] K. Seshadrinathan, R. Soundararajan, A. C. Bovik, and L. K. Cormack. A subjective study to evaluate video quality assessment algorithms. In *Human Vision and Electronic Imaging*, page 75270H, 2010.

[184] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hoßfeld, and P. Tran-Gia. A survey on Quality of Experience of HTTP adaptive streaming. *IEEE Comm. Surveys & Tutorials*, 17(1):469–492, Sep. 2014.

[185] H. R. Sheikh, A. C. Bovik, and G. De Veciana. An information fidelity criterion for image quality assessment using natural scene statistics. *IEEE Trans. Image Processing*, 14(12):2117–2128, 2005.

[186] H. R. Sheikh, M. F. Sabir, and A. C. Bovik. A statistical evaluation of recent full reference image quality assessment algorithms. *IEEE Trans. Image Processing*, 15(11):3440–3451, Nov. 2006.

[187] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.

[188] E. P. Simoncelli and B. A. Olshausen. Natural image statistics and neural representation. *Annual Review of Neuroscience*, 24(1):1193–1216, 2001.

[189] K. Singh, Y. Hadjadj-Aoul, and G. Rubino. Quality of Experience estimation for adaptive HTTP/TCP video streaming using H.264/AVC. In *CCNC-IEEE Consumer Communications & Networking Conference*, pages 1–6, Las Vegas, NV, USA, Jan. 2012.

[190] D. Smilkov, N. Thorat, Y. Assogba, A. Yuan, N. Kreeger, P. Yu, K. Zhang, S. Cai, E. Nielsen, and D. Soergel. Tensorflow.js: Machine learning for the web and beyond. *ArXiv preprint arXiv:1901.05350*, Jan. 2019.

[191] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman. BOLA: Near-optimal bitrate adaptation for online videos. In *Proc. IEEE Int. Conf. Computer Comm.*, pages 1–9, San Francisco, CA, USA, Apr. 2016.

[192] N. Staelens, S. Moens, W. Van den Broeck, I. Marien, B. Vermeulen, P. Lambert, R. Van de Walle, and P. Demeester. Assessing Quality of Experience of IPTV and video on demand services in real-life environments. *IEEE Trans. Broadcasting*, 56(4):458–466, Dec. 2010.

[193] ANSI Standard. Digital transport of one-way video signals-parameters for objective performance assessment. *ANSI Standard ATIS-0100801.03*, 2003.

[194] Statista. Number of downloads of selected video streaming mobile apps worldwide in june 2021, Jun. 2021. https://www.statista.com/statistics/1251791/selected-streaming-apps-downloads/.

[195] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. OSQP: An operator splitting solver for quadratic programs. *ArXiv preprint arXiv:1711.08013*, Nov. 2017.

[196] T. Stockhammer. Dynamic adaptive streaming over HTTP: Standards and design principles. In *Proc. ACM Conf. Multimedia Systems*, pages 133–144, San Jose, CA, USA, Feb. 2011.

[197] A. Storkey. When training and test sets are different: Characterizing learning transfer. *Dataset Shift in Machine Learning*, 30:3–28, 2009.

[198] G. J. Sullivan, J. Ohm, W. Han, and T. Wiegand. Overview of the high efficiency video coding (HEVC) standard. *IEEE Trans. Circuits and Systems for Video Tech.*, 22(12):1649–1668, Sep. 2012.

[199] W. Sun, Fei Zhou, and Q. Liao. MDID: A multiply distorted image database for image quality assessment. *Pattern Recognit.*, 61:153–168, Jul. 2017.

[200] Y. Sun, X. Yin, J. Jiang, v. Sekar, F. Lin, N. Wang, T. Liu, and B. Sinopoli. CS2P: Improving video bitrate selection and adaptation with data-driven throughput prediction. In *Proc. ACM SIGCOMM*, pages 272–285, Florianópolis, Brazil, Aug. 2016.

[201] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, Massachusetts, 1998.

[202] Christian T., Matteo M., and Benjamin R. Which adaptation logic? An objective and subjective performance evaluation of HTTP-based adaptive media streaming systems, 2016.

[203] J. Talens-Noguera, W. Zhang, and H. Liu. Studying human behavioural responses to time-varying distortions for video quality assessment. In *Proc. IEEE Int. Conf. Image Proc.*, pages 651–655, Quebec City, QC, Sep. 2015.

[204] H. Tankovska. Average youtube video length 2018, by category, 2018.

[205] S. Tavakoli, K. Brunnström, K. Wang, B. Andrén, M. Shahid, and N. Garcia. Subjective quality assessment of an adaptive video streaming model. In *IS&T/SPIE Electronic Imaging*, pages 9016.1–9016.13, San Francisco, CA, Feb. 2014.

[206] FFmpeg Team. FFmpeg, Nov. 2018. https://www.ffmpeg.org/.

[207] L. L. Thurstone. A law of comparative judgment. *Psychological Review*, 34(4):273, 1927.

[208] L. Toni, R. Aparicio-Pardo, K. Pires, W. Simon, A. Blanc, and P. Frossard. Optimal selection of adaptive streaming representations. *ACM Trans. Multimedia Computing, Comm., and Applications*, 11(2s):1–43, Feb. 2015.

[209] D. Tsai, Y. Lee, and E. Matsuyama. Information entropy measure for evaluation of image quality. *Journal of Digital Imaging*, 21(3):338–347, 2008.

[210] K. Tsukida and M. R. Gupta. How to analyze paired comparison data. University of Washington, Tech. Rep. UWEETR-2011-0004, May 2011.

[211] J. van der Hooft, S. Petrangeli, T. Wauters, R. Huysegems, P. R. Alface, T. Bostoen, and F. De Turck. HTTP/2-based adaptive streaming of HEVC video over 4G/LTE networks. *IEEE Communications Letters*, 20(11):2177–2180, Aug. 2016.

[212] J. V. van der Hooft, S. Petrangeli, M. Claeys, J. Famaey, and F. De Turck. A learning-based algorithm for improved bandwidth-awareness of adaptive streaming clients. In *Proc. IFIP/IEEE Int. Sym. Integrated Network Management*, pages 131–138, Ottawa, ON, Canada, May 2015.

[213] J. V. van der Hooft, S. Petrangeli, T. Wauters, R. Huysegems, P. R. Alface, T. Bostoen, and F. De Turck. HTTP/2-Based adaptive streaming of HEVC video over 4G/LTE networks. *IEEE Comm. Letters*, 20(11):2177–2180, Jun. 2016.

[214] B. J. Villa, K. De Moor, P. E. Heegaard, and A. Instefjord. Investigating Quality of Experience in the context of adaptive video streaming: Findings from an experimental user study. In *Proceedings of Norsk Informatikkonferanse*, pages 122–133, Stavanger, Norway, Nov. 2013.

[215] Cisco Mobile VNI. Cisco visual networking index: Global mobile data traffic forecast update, 2016-2021 white paper, Mar. 2017. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html.

[216] VQEG. Final report from the video quality experts group on the validation of objective models of video quality assessment. Technical Report, Apr. 2000. http://www.vqeg.org/.

[217] M. J. Wainwright and E. P. Simoncelli. Scale mixtures of gaussians and the statistics of natural images. In *Nerual Information Processing Systems*, volume 12, pages 855–861, 1999.

[218] S. Wang, A. Rehman, K. Zeng, J.g Wang, and Z. Wang. SSIM-motivated two-pass VBR coding for HEVC. *IEEE Trans. Circuits and Systems for Video Tech.*, 27(10):2189–2203, 2016.

[219] Y. Wang, P. Li, L. Jiao, Z. Su, N. Cheng, X. Shen, and P. Zhang. A data-driven architecture for personalized QoE management in 5G wireless networks. *IEEE Wireless Communications*, 24(1):102–110, Nov. 2016.

[220] Z. Wang. *Internet QoS: Architectures and mechanisms for quality of service*. Morgan Kaufmann, 2001.

[221] Z. Wang and A. C. Bovik. Modern image quality assessment. *Synthesis Lectures on Image, Video, and Multimedia Processing*, 2(1):1–156, Dec. 2006.

[222] Z. Wang and A. C. Bovik. Mean squared error: Love it or leave it? A new look at signal fidelity measures. *IEEE Signal Processing Magazine*, 26(1):98–117, Jan. 2009.

[223] Z. Wang and A. C. Bovik. Reduced-and no-reference image quality assessment. *IEEE Signal Processing Magazine*, 28(6):29–40, 2011.

[224] Z. Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Processing*, 13(4):600–612, Apr. 2004.

[225] Z. Wang, Z. Duanmu, A. Rehman, and K. Zeng. Method and system for automatic user Quality-of-Experience measurement of streaming video, Aug. 2017. US Patent WO/2017/152274.

[226] Z. Wang and Q. Li. Video quality assessment using a statistical model of human visual speed perception. *Journal of Optical Society of America A*, 24(12):B61–B69, 2007.

[227] Z. Wang and A. Rehman. Begin with the end in mind: A unified end-to-end Quality-of-Experience monitoring, optimization and management framework. In *SMPTE Annual Technical Conference and Exhibition*, pages 1–11, Hollywood, CA, USA, Oct. 2017.

[228] Z. Wang and E. P. Simoncelli. Maximum differentiation (MAD) competition: A methodology for comparing computational models of perceptual quantities. *Journal of Vision*, 8(12):8.1–8.13, Sep. 2008.

[229] Z. Wang, K. Zeng, and A. Rehman. Method and system for smart adaptive video streaming driven by perceptual Quality-of-Experience estimations, Aug. 2 2016. US Patent WO/2016/123721.

[230] Z. Wang, K. Zeng, A. Rehman, H. Yeganeh, and S. Wang. Objective video presentation QoE predictor for smart adaptive video streaming. In *Proc. SPIE Optical Engineering+Applications*, pages 95990Y.1–95990Y.13, Sep. 2015.

[231] K. Watanabe, J. Okamoto, and T. Kurita. Objective video quality assessment method for evaluating effects of freeze distortion in arbitrary video scenes. In *Image Quality and System Performance IV*, pages 64940.1–64940.8, San Jose, CA, United States, Jan. 2007.

[232] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the H.264/AVC video coding standard. *IEEE Trans. Circuits and Systems for Video Tech.*, 13(7):560–576, 2003.

[233] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the H.264/AVC video coding standard. *IEEE Trans. Circuits and Systems for Video Tech.*, 13(7):560–576, Aug. 2003.

[234] S. Winkler. Analysis of public image and video databases for quality assessment. *IEEE Journal of Selected Topics in Signal Processing*, 6(6):616–625, 2012.

[235] S. Winkler and R. Campos. Video quality evaluation for internet streaming applications. In *Human Vision and Electronic Imaging*, pages 104–115, 2003.

[236] J. Xue, D. Zhang, H. Yu, and C. W. Chen. Assessing Quality of Experience for adaptive HTTP video streaming. In *Proc. IEEE Int. Conf. Multimedia and Expo Workshop*, pages 1–6, Chengdu, China, Jul. 2014.

[237] F. Y. Yan, H. Ayers, C. Zhu, S. Fouladi, J. Hong, K. Zhang, P. Levis, and K. Winstein. Learning in situ: A randomized experiment in video streaming. In *USENIX Symp. on Networked Systems Design and Implementation*, pages 495–511, 2020.

[238] K. Yang, C. C. Guest, K. El-Maleh, and P. K. Das. Perceptual temporal quality metric for compressed video. *IEEE Trans. Multimedia*, 9(7):1528–1535, 2007.

[239] J. Yao, S. S. Kanhere, I. Hossain, and M. Hassan. Empirical evaluation of HTTP adaptive streaming under vehicular mobility. In *Int. Conf. Research in Networking*, pages 92–105, 2011.

[240] P. Ye and D. Doermann. Active sampling for subjective image quality assessment. In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, pages 4249–4256, Columbus, OH, USA, Jun. 2014.

[241] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli. A control-theoretic approach for dynamic adaptive video streaming over HTTP. *ACM SIGCOMM Computer Comm. Review*, 45(4):325–338, Apr. 2015.

[242] YouTube. Recommended upload encoding settings, May 2020. https://support.google.com/youtube/answer/1722171.

[243] X. Zhang, S. Sen, D. Kurniawan, H. Gunawi, and J. Jiang. E2E: Embracing user heterogeneity to improve Quality of experience on the web. In *Proc. ACM SIGCOMM*, pages 289–302, 2019.

[244] Y. Zhong, I. Richardson, A. Miller, and Y. Zhao. Perceptual quality of H.264/AVC deblocking filter. In *Proc. IET Int. Conf. Visual Info. Engineering*, pages 379–384, 2005.

[245] M. Zink, O. Künzel, J. Schmitt, and R. Steinmetz. Subjective impression of variations in layer encoded videos. In *Proc. IEEE/ACM Int. Workshop Quality of Service*, pages 137–154, Berlin, Germany, Jun. 2003.

[246] M. Zink, J. Schmitt, and R. Steinmetz. Layer-encoded video in scalable adaptive streaming. *IEEE Trans. Multimedia*, 7(1):75–84, Feb. 2005.

[247] X. Zou, J. Erman, V. Gopalakrishnan, E. Halepovic, R. Jana, X. Jin, J. Rexford, and R. K. Sinha. Can accurate predictions improve video streaming in cellular networks? In *HotMobile*, pages 57–62, Santa Fe, New Mexico, USA, Feb. 2015.

Appendix A

Detailed Experimental Setup of Meta Learning-Based Throughput Predictor

A.1 Multi-Layer Perceptron Architecture

We learn a MLP to using the proposed learning scheme, and use it as a baseline to validate the effectiveness of the proposed network architecture in throughput prediction. The parametrization of MLP used in the ablation experiment is shown in A.1.

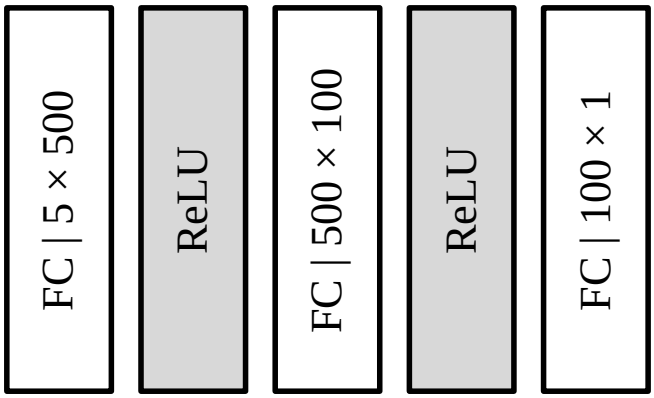


Figure A.1: Illustration of the multi-layer perceptron configurations for MetaTP. We denote the parameterization of the fully connected layer as "input channel × output channel".

Appendix B

Construction of the Waterloo Streaming Video Database

B.1 Encoding Configurations

We use the standard FFmpeg video processing library to encode the streaming videos in the WaterlooSV database. More details of each encoder is given by Table B.1.

Table B.1: Encoder version and package.

Encoder	Version	Package URL
H.264	ffmpeg v4.4	https://trac.ffmpeg.org/wiki/Encode/H.264
HEVC	ffmpeg v4.4	https://trac.ffmpeg.org/wiki/Encode/H.265
AV1	ffmpeg v4.4	https://trac.ffmpeg.org/wiki/Encode/AV1

The encoding configurations used in the WaterlooSV database are illustrated in Table B.2.

B.2 Packaging

Table B.2: Encoder configuration.

Encoder	Pass	Command
H.264	1	ffmpeg INPUT -an -c:v libx264 -width=WIDTH -height=HEIGHT -preset veryfast -pass 1 -r FRAMERATE -b:v BITRATE -maxrate BITRATE*2 -bufsize BITRATE*4 -x264opts -keyint=FRAMERATE*4:min-keyint=FRAMERATE*4:no-scenecut -f null /dev/null && /
	2	ffmpeg INPUT -an -c:v libx264 -width=WIDTH -height=HEIGHT -preset veryfast -pass 2 -r FRAMERATE -b:v BITRATE -maxrate BITRATE*2 -bufsize BITRATE*4 -x264opts -keyint=FRAMERATE*4:min-keyint=FRAMERATE*4:no-scenecut OUTPUT
HEVC	1	ffmpeg INPUT -an -c:v libx265 -width=WIDTH -height=HEIGHT -preset veryfast -pass 1 -r FRAMERATE -b:v BITRATE -maxrate BITRATE*2 -bufsize BITRATE*4 -x265-params -keyint=FRAMERATE*4:min-keyint=FRAMERATE*4:open-gop=0 -f null /dev/null && /
	2	ffmpeg INPUT -an -c:v libx265 -width=WIDTH -height=HEIGHT -preset veryfast -pass 2 -r FRAMERATE -b:v BITRATE -maxrate BITRATE*2 -bufsize BITRATE*4 -x265-params -keyint=FRAMERATE*4:min-keyint=FRAMERATE*4:open-gop=0 OUTPUT
AV1	1	ffmpeg INPUT -an -c:v libaom-av1 -width=WIDTH -height=HEIGHT -preset veryfast -pass 1 -r FRAMERATE -b:v BITRATE -maxrate BITRATE*2 -bufsize BITRATE*4 -aom-params -keyint_min=FRAMERATE*4 -f null /dev/null && /
	2	ffmpeg INPUT -an -c:v libaom-av1 -width=WIDTH -height=HEIGHT -preset veryfast -pass 2 -r FRAMERATE -b:v BITRATE -maxrate BITRATE*2 -bufsize BITRATE*4 -aom-params -keyint_min=FRAMERATE*4 OUTPUT

Listing B.1: Sample manifest file

```
<?xml version='1.0' encoding='utf-8'?>
<MPD xmlns="urn:mpeg:dash:schema:mpd:2011" maxSegmentDuration="PT0H0M4
    .000S" mediaPresentationDuration="PT0H0M32.000S" minBufferTime="PT1
    .500S" profiles="urn:mpeg:dash:profile:full:2011" type="static">
 <ProgramInformation moreInformationURL="http://gpac.io">
  <Title>manifest.mpd generated by GPAC</Title>
 </ProgramInformation>

 <Period duration="PT0H0M32.000S">
  <AdaptationSet bitstreamSwitching="true" lang="eng" maxFrameRate="30"
      maxHeight="2160" maxWidth="3840" par="16:9" segmentAlignment="true">
   <SegmentList>
    <Initialization sourceURL="manifest_init.mp4" />
   </SegmentList>
   <Representation bandwidth="241372" codecs="avc3.64000D" frameRate="30"
       height="180" id="1" mimeType="video/mp4" sar="1:1" width="320">
```

```
13      <SegmentList duration="61440" timescale="15360">
14       <SegmentURL bitrate="149204" hdtv="34" media="v01/1.m4s" phone="40"
             uhdtv="22" />
15       <SegmentURL bitrate="76864" hdtv="1" media="v01/2.m4s" phone="0"
             uhdtv="0" />
16       <SegmentURL bitrate="163823" hdtv="5" media="v01/3.m4s" phone="2"
             uhdtv="1" />
17       <SegmentURL bitrate="139866" hdtv="4" media="v01/4.m4s" phone="0"
             uhdtv="0" />
18       <SegmentURL bitrate="143724" hdtv="6" media="v01/5.m4s" phone="5"
             uhdtv="1" />
19       <SegmentURL bitrate="98406" hdtv="3" media="v01/6.m4s" phone="2"
             uhdtv="1" />
20       <SegmentURL bitrate="114561" hdtv="3" media="v01/7.m4s" phone="1"
             uhdtv="0" />
21       <SegmentURL bitrate="88342" hdtv="4" media="v01/8.m4s" phone="2"
             uhdtv="0" />
22      </SegmentList>
23     </Representation>
24     <Representation bandwidth="1070223" codecs="avc3.64001E" frameRate="30
         " height="360" id="5" mimeType="video/mp4" sar="1:1" width="640">
25      <SegmentList duration="61440" timescale="15360">
26       <SegmentURL bitrate="624912" hdtv="73" media="v05/1.m4s" phone="85"
             uhdtv="57" />
27       <SegmentURL bitrate="415939" hdtv="53" media="v05/2.m4s" phone="68"
             uhdtv="30" />
28       <SegmentURL bitrate="723362" hdtv="56" media="v05/3.m4s" phone="70"
             uhdtv="33" />
29       <SegmentURL bitrate="575269" hdtv="58" media="v05/4.m4s" phone="70"
             uhdtv="34" />
30       <SegmentURL bitrate="561296" hdtv="57" media="v05/5.m4s" phone="70"
             uhdtv="34" />
31       <SegmentURL bitrate="439411" hdtv="55" media="v05/6.m4s" phone="68"
             uhdtv="31" />
32       <SegmentURL bitrate="544505" hdtv="56" media="v05/7.m4s" phone="67"
             uhdtv="34" />
33       <SegmentURL bitrate="405499" hdtv="56" media="v05/8.m4s" phone="68"
             uhdtv="35" />
```

```
34      </SegmentList>
35    </Representation>
36   </AdaptationSet>
37  </Period>
38 </MPD>
```

B.3 Viewing Device

In the construction of the WaterlooSV database, we employ VMAF to account for the device-dependent video quality prediction. VMAF [117] is a state-of-the-art presentation video quality measure which is capable to predict the quality of a video displayed on HDTV, smartphone, and UHDTV. Each device specific model is calibrated by a subject video quality assessment experiment, in which streaming videos are displayed on a certain viewing device to viewers at the typical distance. Following to the recommendation in [117], we re-sample the test and reference videos using bicubic filter to match the display resolution followed by objective quality evaluation.