

ABSTRACT

Title of dissertation: **DEEP LEARNING-ENABLED INTELLIGENT
GOAL-ORIENTED AND SEMANTIC
COMMUNICATION FOR 6G NETWORKS**

Matin Mortaheb, Doctor of Philosophy, 2025

Dissertation directed by: Professor Sennur Ulukus
Department of Electrical and Computer Engineering

In the emerging sixth generation (6G) wireless networks, two major technological directions are expected to play pivotal roles: semantic communication and federated learning. Although independent in their approaches, both aim to make networks more intelligent, efficient, and adaptive to user and application demands. Semantic communication fundamentally shifts the focus from transmitting raw data to conveying only information that carries meaning and is directly relevant to a given task. This shift enables more efficient use of spectrum and reduces unnecessary transmission overhead, which is critical for supporting services such as augmented reality, holographic telepresence, and real-time control systems. In parallel, federated learning enables distributed edge devices to collaboratively train powerful machine learning models without sharing their raw data, thereby ensuring privacy, reducing communication burdens, and supporting large-scale, decentralized intelligence. Together, these advancements address key 6G challenges such as ultra-low latency, massive connectivity, and the need for real-time, context-aware decision-

making at the edge. By integrating semantic understanding and distributed learning capabilities into the communication fabric, 6G networks will evolve from simple data carriers into intelligent infrastructures that empower a new generation of applications and services. This thesis addresses the design, analysis, and integration of advanced techniques that enable intelligent and efficient communication across multiple emerging 6G applications, including distributed learning, real-time immersive media, and multi-modal AI systems.

A primary contribution of this thesis lies in the development of semantic communication frameworks, where data is encoded and transmitted in a manner that preserves semantically important content while discarding redundant or irrelevant information. We propose an adaptive semantic encoder that utilizes attention mechanisms from vision-language models to dynamically identify and prioritize semantically salient regions in images and videos. By allocating communication resources according to semantic importance, our framework significantly reduces bandwidth consumption and latency, enabling critical 6G use cases such as holographic telepresence, remote surgery, and augmented/virtual reality with enhanced reliability and efficiency.

Parallel to semantic encoding, this thesis investigates federated learning (FL) as a foundational pillar for distributed intelligence in 6G. In FL, multiple devices collaboratively train a shared model without sharing their raw local data, offering strong privacy guarantees. However, practical challenges arise due to severe data heterogeneity across devices and limited wireless resources. To address these, we develop FedGradNorm, a dynamic gradient normalization scheme that harmonizes

learning speeds across heterogeneous tasks, ensuring fair and efficient convergence in personalized federated multi-task learning (PF-MTL) settings. Furthermore, we propose a novel graph-based decentralized learning approach that adaptively adjusts communication topologies by automatically learning inter-client task correlations, thereby minimizing negative interference and enhancing convergence performance. These contributions enable FL frameworks that are communication-efficient, robust to statistical heterogeneity, and tailored for highly dynamic and bandwidth-limited 6G environments.

In addition to semantic encoding and federated learning, this thesis addresses the challenge of adaptive video streaming over wireless networks, a critical service for 6G that supports ultra-high-definition and interactive media applications. Traditional rate control algorithms lack the ability to dynamically adapt to rapidly changing network conditions and user experience requirements. We design a machine learning-based rate control algorithm that employs deep learning to jointly optimize video bitrate selection by considering real-time channel states, buffer dynamics, and semantic content complexity. Our approach achieves superior perceptual quality and stable playback, providing a seamless user experience even under highly variable wireless channel conditions.

Lastly, to ensure reliable and trustworthy information delivery in complex 6G multi-modal systems, we introduce RAG-Check, a novel framework for evaluating and mitigating hallucinations in multi-modal retrieval-augmented generation (RAG) systems. As 6G networks are envisioned to support advanced AI services that integrate visual, textual, and contextual data, ensuring the factual correctness and

semantic alignment of generated responses becomes critical. RAG-Check rigorously analyzes the consistency between retrieved image evidence and generated textual outputs, providing quantitative and interpretable hallucination scores. This framework enhances the robustness and credibility of large-scale vision-language models in real-world deployments.

By cohesively integrating these contributions, this thesis offers a comprehensive design blueprint for goal-oriented and semantic communication systems in 6G networks. The thesis bridges the gaps between intelligent content encoding, collaborative distributed learning, adaptive multimedia delivery, and reliable multi-modal AI, collectively advancing the vision of 6G networks that do not merely transmit data but understand, reason, and act on information to effectively meet diverse user goals and application requirements, thereby laying the foundations for future intelligent communication systems that are efficient, scalable and also contextually aware and user-centric, enabling the next era of ubiquitous and intelligent connectivity.

Deep Learning-Enabled Intelligent Goal-Oriented and Semantic
Communication for 6G Networks

by

Matin Mortaheb

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2025

Advisory Committee:
Professor Sennur Ulukus, Chair/Advisor
Professor Behtash Babadi
Professor Adrian Papamarcou
Professor Dinesh Manocha
Dr. Mohammad A. Khojastepour

© Copyright by
Matin Mortaheb
2025

Dedication

To my friends, family, and parents for always believing in me.

Acknowledgments

I would like to express my deepest gratitude to my advisor, Professor Sennur Ulukus, for her unwavering guidance, support, and insightful mentorship throughout my PhD journey. Her dedication to research and her passion for discovery have continually inspired me and shaped my approach to engineering problems. Over these six years, our countless discussions have taught me how to frame meaningful and technically sound research questions, and have opened my eyes to new and exciting directions. I am especially grateful for her optimism, encouragement, and kind words, which lifted my spirits during challenging times. It has been a privilege and an incredible learning experience to work under her supervision, and I feel truly fortunate to have been her student.

I would also like to extend my sincere thanks to my committee members, Professors Behtash Babadi, Adrian Papamarcou, Amir Khojastepour, and Dinesh Manocha, for their valuable feedback and support. I am grateful as well to all the professors at UMD with whom I had the pleasure to learn and interact over the past six years.

I am truly thankful to Dr. Amir Khojastepour, who guided me as my mentor and advisor during my two-year internship at NEC Laboratories America. His brilliant mind, constant support, and kindness were a great source of motivation, and I deeply appreciate the many papers we wrote and the conferences we attended together. His guidance and encouragement have been invaluable to my professional growth and personal development.

I have been lucky to share this journey with an exceptional group of colleagues and friends in our research group: Yoonkyo Jung, Rustu Karakaya, Stavros Mitrolaris, Arunabh Srivastava, Lei Hu, Ismail Cosandal, Weihang Cao, Priyanka Kaswan, Cemil Vahapoglu, Purbesh Mitra, Mustafa Doger, Subhankar Banerjee, Shreya Meel, Sahan Liyanaarachchi, Mohamed Nomeir, Arunabh Srivastava, Alptug Aytekin, Zhusheng Wang, Batuhan Arasli, Brian Kim, Baturalp Buyukates, and Melih Bastopcu. Their camaraderie and support made this journey all the more meaningful.

I would like to express my sincere gratitude to Yoonkyo Jung, Rustu Karakaya, Pritanka Kaswan, and Cemil Vahapoglu for their invaluable support, motivation, and contributions throughout the process of writing and publishing papers.

Beyond the lab, I am grateful to my friends at UMD — Maryam Maghsoudi, Helya Hosseini, Ali Kamali, Kiyan Amirian, and Parastoo Pilevar — for making College Park feel like home. Thank you for the adventures, late-night talks, and laughter that sustained me through the ups and downs. I am also thankful to all my other friends who helped me stay grounded, healthy, and happy along the way.

Above all, I owe my deepest thanks to my parents, Dr. Hamidreza Mortaheb and Safi Samsamshariat, for their endless sacrifices, unconditional love, and unwavering belief in me. Their constant encouragement gave me the strength to pursue my dreams and persevere through every setback. I am also deeply grateful to my sister, Samin Mortaheb, for her continuous support and for always believing in me. Their faith in me has been the foundation of this journey.

Table of Contents

List of Figures	ix
List of Tables	xiii
1 Introduction	1
2 Adaptive Transformer-Aided Semantic Communication with Multi-Resolution Encoding	12
2.1 Introduction	12
2.2 Alternative Views of Semantics and Semantic Communications	19
2.3 System Model and Problem Formulation	21
2.3.1 Transformer-Aided Semantic Communication	22
2.3.1.1 Projector	22
2.3.1.2 ViT	23
2.3.1.3 Compressor, Decoder, and Classifier	25
2.3.1.4 Main Contributions	25
2.3.2 Multi-resolution Transformer-Aided Semantic Communication	28
2.3.2.1 Attention-Guided Resolution Selector	30
2.3.2.2 Encoders and Decoders	35
2.3.3 Multi-Modal Semantic Communication	36
2.3.3.1 Multi-Modal Semantic Extractor	37
2.3.3.2 Multi-Resolution Encoder/Decoder	41
2.4 Experimental Results	42
2.4.1 Dataset Specifications	42
2.4.2 Model and Hyperparameters	43
2.4.3 Results and Analysis	45
2.4.3.1 Transformer-Aided Semantic Communication	45
2.4.3.2 Multi-Resolution Semantic Communication	49
2.4.3.3 Multi-Modal Semantic Communication	51
2.5 Conclusion	56

3	Semantic Multi-Resolution Communication	58
3.1	Introduction	58
3.2	Multi-Head AE for SMRC	62
3.3	System Model and Problem Formulation	68
3.4	Experimental Results	72
3.4.1	Dataset Specifications	72
3.4.2	Model and Hyperparameters	73
3.4.3	Reconstruction Performance (RP)	73
3.4.4	Semantic Accuracy (SA)	75
3.5	Conclusion	78
4	Personalized Federated Multi-Task Learning Over Networks	79
4.1	Introduction	79
4.2	System Model and Problem Formulation	82
4.2.1	Federated Learning (FL)	82
4.2.2	Personalized Federated Multi-Task Learning (PF-MTL)	82
4.2.3	PF-MTL as Bilevel Optimization Problem	84
4.3	Algorithm Description	87
4.3.1	Definitions and Preliminaries	87
4.3.2	FedGradNorm Description	88
4.4	Convergence Analysis	90
4.5	Experiments	94
4.5.1	Dataset Specifications	94
4.5.2	Hyperparameters and Model Specifications	96
4.5.3	Results and Analysis	96
4.6	Conclusion and Discussion	100
4.7	Appendix	101
5	Personalized Decentralized Multi-Task Learning Over Dynamic Communication Graphs	109
5.1	Introduction	109
5.2	System Model and Problem Formulation	112
5.3	Convergence Analysis	116
5.4	Experimental Results	117
5.4.1	Dataset Specifications	117
5.4.2	Hyperparameters and Model Specifications	119
5.4.3	Results and Analysis	120
5.5	Conclusion	124
6	$rAge-k$: Communication-Efficient Federated Learning Using Age Factor	126
6.1	Introduction	126
6.2	System Model and Problem Formulation	130
6.2.1	Convergence of $rAge-k$	133
6.3	Experimental Results	135
6.3.1	Dataset Specifications	135

6.3.2	Model and Hyperparameters	135
6.3.3	System Performance	136
6.4	Conclusion and Discussion	140
7	Age of Gossip in Networks with Multiple Views of a Source	142
7.1	Introduction	142
7.2	Preliminaries	145
7.3	Main Result	148
7.3.1	Derivation of Average AoI	148
7.3.2	Ring Network	153
7.3.3	Line Network	155
7.3.4	Fully Connected Network	159
7.4	Conclusion	162
8	Evaluating and Improving Multi-Modal RAG Hallucination	163
8.1	Introduction	163
8.2	Multimodal RAG	172
8.3	RAG-Check	174
8.3.1	Partition and Categorization	175
8.3.2	Relevance Score (RS) Model	176
8.3.3	Correctness Score (CS) Model	180
8.4	RS score vs CLIP for RAG selection	182
8.5	Re-ranking with RS Score	184
8.6	Numerical Result	186
8.6.1	Dataset Specification	187
8.6.2	Model and Hyperparameters	188
8.6.3	Relevance Score (RS) Performance	188
8.6.4	Correctness Score (CS) Performance	190
8.6.5	RAG Scores vs Human Evaluation	190
8.6.6	Evaluation of Different RAG Schemes	192
8.6.6.1	RAG Selection Performance:	192
8.6.6.2	RAG Context Generation Performance:	194
8.7	Conclusion	196
9	Deep Learning-Based Real-Time Rate Control for Live Streaming on Wireless Networks	198
9.1	Introduction	198
9.2	System Model	203
9.3	Deep Learning Based Rate Control	206
9.4	Performance Evaluation	209
9.5	Conclusion	215
10	Deep Learning-Based Real-Time Quality Control of Standard Video Compression for Live Streaming	217
10.1	Introduction	217

10.2 System Model	222
10.3 Deep Learning-Based Rate Control	225
10.4 Performance Evaluation	228
10.5 Conclusion	236
11 Conclusions	237
Bibliography	243

List of Figures

2.1	Transformer-aided semantic communication framework.	14
2.2	System model for transformer-aided semantic communication.	23
2.3	Deriving $\tilde{A}_{cls}^{(h)}$ from $A^{(h)}$	26
2.4	Obtaining attention mask from ViT attention unit.	27
2.5	Channel-aware multi-resolution semantic communication framework.	28
2.6	Attention-guided resolution selector block framework.	30
2.7	Attention aggregation block.	31
2.8	Multi-modal semantic communication (MMSC) framework.	36
2.9	Multi-modal semantic encoder framework.	39
2.10	(a) Decoder model structure for ViT-based semantic communication framework (Section 2.3.1). (b) Encoder and decoder structure for different resolutions in multi-resolution transformer-aided semantic communication framework (Section 2.3.2 and 2.3.3).	45
2.11	(a) Masked MSE performance between the original and reconstructed images under varying channel rates. (b) Classification accuracy of the reconstructed image when the most informative patches are selected until the channel bandwidth is fully utilized ($\alpha = 1$).	46
2.12	Example of reconstructed images with compression of $r = 0.5$	46
2.13	(a) Classification accuracy of the reconstructed image in different channel rates when we allow 15% of patches to be selected randomly from low attention score patches along with the most informative patches ($\alpha = 0.85$). (b) Classification accuracy of the reconstructed image in different channel rates using a fine-tuned classifier.	48
2.14	(a) Reconstruction result for three medium resolutions. (b) Accuracy result for three medium resolutions.	50
2.15	(a) Accuracy result for adaptive multi-resolution semantic communication framework in various channel rates. (b) Resolution assignment to patches in different channel rate.	51
2.16	An example image from the dataset, transmitted and reconstructed using the MMSC (Section 2.3.3) and ViT-SC (Section 2.3.2) models under a 50% channel rate constraint.	53

2.17	Average MSE between the original and reconstructed images within the masked area defined by the ground truth segmentation corresponding to the user query.	54
2.18	L1 distance between informative score maps of the original and reconstructed images conditioned on the user query.	55
2.19	CLIP relevancy score between reconstructed image and text.	55
3.1	Application of SMRC on Example 1.	62
3.2	Application of SMRC on Example 2.	63
3.3	Multi-head AE for SMRC.	66
3.4	System model.	69
3.5	Model structure for MNIST dataset.	72
3.6	Model structure for CIFAR10 dataset.	73
3.7	PSNR results for (a) MNIST and (b) CIFAR10 dataset.	75
3.8	Effect of β (weight vector) in precision/recall.	76
3.9	Accuracy (avg. recall) of different layer decoders in different SNR for (a) MNIST (b) CIFAR10 dataset.	77
4.1	Personalized federated learning (PFL) framework with a common network (shown in blue) and small personalized headers (shown in red, green, black).	83
4.2	Comparison of task losses in <i>FedGradNorm</i> and <i>FedRep</i> ; balanced data allocation among tasks (a) task 1 (face landmark), (b) task 2 (gender), (c) task 3 (smile), (d) task 4 (glasses), (e) task 5 (pose), (f) task weights.	98
4.3	Comparison between task accuracy achieved via <i>FedGradNorm</i> and <i>FedRep</i> in RadComDynamic dataset (a) task 1 (modulation classification), (b) task 2 (signal classification), (c) task 3 (anomaly behavior), (d) task weights.	99
5.1	Dynamic communication graph in personalized decentralized learning framework with a common network (shown in blue) and small personalized headers (shown in red, green, black, orange). Users with the same-color circles have positive correlation, and different colors mean negative correlation.	111
5.2	An expected communication graph (correlation) among the chosen attributes from CelebA dataset.	119
5.3	Comparison of task losses achieved via dynamic communication graph using transference and naive fully-connected case for synthetic dataset.	121
5.4	Heatmap of the connectivity matrix achieved by the transference method over different epochs, (a) epoch 1, (b) epoch 6, (c) epoch 11, (d) epoch 16.	122
5.5	Comparison of task losses achieved via dynamic communication graph using transference and naive fully-connected case for celebA dataset.	123
6.1	System model for <i>rAge-k</i> framework.	128

6.2	Heatmap of the connectivity matrix achieved by the DBSCAN method over different epochs, (a) iteration 1, (b) iteration 21, (c) iteration 41, (d) iteration 61.	137
6.3	(a) Accuracy (in percentage) and (b) loss (averaged over all 10 users) over training iterations with MNIST dataset.	138
6.4	Heatmap of the connectivity matrix achieved by the DBSCAN method over different epochs, (a) iteration 1, (b) iteration 201	139
6.5	(a) Accuracy (in percentage) and (b) loss (averaged over all 6 users) over training iterations with CIFAR10 dataset.	140
7.1	(a) An example single source (single sensing node) network, and (b) an example multiple sensing node network.	143
7.2	An example multi-view network with multiple sensing nodes. Nodes 0 and 1 are sensing nodes.	152
7.3	(a) Ring network, and (b) fully connected network, where all nodes are sensing nodes.	155
7.4	Effect of changing q in average AoI in line/ring network.	159
7.5	Effect of changing d in average AoI in line/ring network.	161
8.1	RAG-check example for two different multi-modal RAG schemes. . .	164
8.2	Re-ranking via relevancy score (RS).	169
8.3	RAG scheme structure.	173
8.4	Overlay of RAG-check structure on RAG block diagram.	175
8.5	General structure to quantify relevancy of image to query.	177
8.6	RS model structure.	178
8.7	(a) CLIP-score histogram. (b) RS score histogram.	183
8.8	(a) CDF for relevant (irrelevant) samples in CLIP-score. (b) CDF for relevant (irrelevant) samples in RS score.	185
8.9	A sample entry of dataset used for training/evaluation.	187
8.10	Comparison of different RAG selection mechanisms with RS model. .	193
8.11	RAG retrieval relevancy performance.	194
8.12	Comparison of different RAG context and generation mechanism with CS model.	195
8.13	RAG generated output performance.	196
9.1	Fluctuations of encoded video bitrate and available channel bitrate over chunks.	200
9.2	Consequences of choosing wrong value for QP.	203
9.3	System model for RTRC.	205
9.4	Model structure for the rate control unit (RCU).	207
9.5	Variance of encoded video bitrate over chunks in different QP values. .	210
9.6	Model structure for (a) prediction head and (b) CGN.	211
9.7	PSNR vs. average channel bitrate on fading condition.	212
9.8	Packet drop vs. Average channel bitrate on a fading condition. . . .	213
9.9	Packet success rate vs. channel bandwidth efficiency.	214

10.1	Fluctuations of encoded video bitrate and available channel bitrate over chunks.	219
10.2	Consequences of choosing wrong value for QP	223
10.3	System model for RTQC.	225
10.4	Model structure for the control unit (CU).	227
10.5	Variance in PSNR of encoded video over chunks in different QP values.	229
10.6	Model structure for prediction head.	231
10.7	Model structure for CGN.	233
10.8	Bitrate vs. target video PSNR.	234
10.9	Non-conformance probability vs. targeted video PSNR.	235
10.10	Conformance probability vs. channel bandwidth efficiency.	236

List of Tables

4.1	RadComDynamic: Dynamic settings.	96
4.2	System model hyperparemeters.	97
4.3	Shared network model.	97
4.4	Comparison of task losses after 100 epochs in <i>FedGradNorm</i> and <i>FedRep</i> ; imbalanced data allocation among tasks.	99
5.1	Shared network model.	120
6.1	Network Model	135
8.1	Performance evaluation of scoring models.	189
8.2	Alignment of RS scores with human evaluators.	192

CHAPTER 1

Introduction

As we move toward the 6G era, communication networks are expected to go beyond simply offering higher data rates and better connectivity. The main focus is shifting toward intelligent, goal-oriented, and context-aware communication systems that are deeply integrated with machine learning and artificial intelligence [1–4]. Unlike traditional communication systems, which prioritize reliable delivery of raw bits, future networks aim to understand and convey the semantic meaning behind transmitted information to support task-specific objectives [5, 6]. This transition marks a paradigm shift: advancing from bit-level communication to semantic-level communication, and evolving from emphasizing quality-of-service (QoS) metrics to prioritizing quality-of-experience (QoE) metrics, ultimately placing user experience and task effectiveness at the center of design considerations [7–9].

The key pillars supporting this vision include semantic and context-aware communication, task-oriented end-to-end optimization, and distributed intelligence at the network edge. In 6G systems, devices are expected to interpret, process, and transmit only the most relevant information needed to complete a given task, signif-

icantly reducing bandwidth consumption and latency. Moreover, ensuring the timeliness and freshness of information becomes essential, especially for time-sensitive applications. At the same time, privacy and security must be maintained as data is increasingly processed closer to, or even directly at, the user side in decentralized architectures.

Under this broad 6G framework, this dissertation investigates several disjoint yet complementary research directions that collectively address the key challenges of next-generation communication and learning systems. At its core, this work is driven by the desire to enable efficient, intelligent, and trustworthy information exchange and learning in dynamic, resource-constrained environments.

A central focus of this dissertation is on semantic communication, where the fundamental idea is to transmit only those parts of data that are semantically significant for the receiver’s task [1, 2, 10, 11]. Inspired by the success of deep learning models, especially transformers [12] with their self-attention mechanisms [13], we propose methods to prioritize and compress data based on its semantic importance. This includes developing transformer-aided semantic communication frameworks for images and extending them to multi-modal scenarios, where both visual and textual cues guide the prioritization process. Furthermore, we advance these ideas to semantic multi-resolution communication, in which different data regions are transmitted at varying resolutions aligned with their relevance, enabling adaptive, bandwidth-efficient transmissions under strict latency and channel constraints.

In parallel to semantic communication, this dissertation explores distributed and federated learning frameworks, which play a crucial role in 6G systems by en-

abling on-device intelligence [14–18]. Current machine learning (ML) algorithms heavily depend on a wide range of data sources, including smartphones, IoT devices, and autonomous vehicles, to name just a few. By leveraging this massive dataset, these algorithms can produce accurate predictions and results. However, the current state of ML training is primarily centralized, requiring local datasets from edge sources or clients to be uploaded to a central server for training and prediction. This approach can raise concerns about user privacy violations and incur high costs associated with uploading data to the central server. Hence, new models are shifting to a more decentralized approach, called federated learning (FL) [19], where local models are trained at the clients using their local datasets and only the model parameters or gradients are communicated in multiple iterations by the clients to generate a shared global model. This problem has attracted significant recent interest in deep learning since it allows for improved privacy, faster learning, and real-time predictions.

Despite their many advantages, current FL frameworks, such as FedAvg [19], face two significant challenges. The first is statistical heterogeneity, which arises as the number of clients or tasks in a FL setting increases. This heterogeneity is caused by differences in task complexity and non-i.i.d. data distribution among clients, ultimately leading to degraded system performance [20, 21]. Non-i.i.d. data distribution, in particular, can be especially problematic when clients are responsible for learning different yet related tasks. In this case, synergy between tasks may not always be positive, referred to as a negative transference, which further degrades performance [22]. This motivates the profound need for research on personalized

federated learning (PFL) that can generate customized models for individual clients while still leveraging global knowledge distilled from others.

A second key challenge facing federated learning is the requirement of high bandwidth connections for exchanging model parameters or gradients that often tend to be of the order of hundreds of millions. There are several methods introduced to decrease the communication overhead during training of deep neural networks, specifically in a distributed or federated learning setting [23]. Quantization [18] and sparsification [24, 25] are two main methods proposed as compression methods for communication-efficient distributed learning. These methodologies rely on the fact that gradients of deep networks are naturally sparse. Methods like top- k [24] and its variant r Top- k [25] have been proposed to exploit this sparsity, where r Top- k outperforms top- k by utilizing a combination of exploration and exploitation to mitigate bias from aggressive sparsification, ultimately improving convergence and performance.

Motivated by these challenges, this dissertation explores novel personalized and decentralized federated learning frameworks over wireless fading channels and dynamic communication graphs, which are highly relevant to mobile edge networks in 6G. These contributions aim to enable robust, communication-efficient, and privacy-preserving on-device learning that adapts to rapidly changing network topologies and heterogeneous data distributions.

Beyond learning and semantic transmission, the thesis also addresses the critical challenge of timeliness and freshness of information [26–28], particularly in distributed systems. In future networks, it is crucial not only to deliver correct in-

formation but also to ensure it is fresh and up-to-date. This is addressed through the study of age-of-information-inspired metrics [26], including the age of gossip in networks where different agents hold possibly outdated or inconsistent views of a shared source. Understanding these dynamics helps design protocols that guarantee both consistency and low latency, which are central to 6G’s mission of real-time, trustworthy communication.

Moreover, as intelligent applications increasingly rely on generated content, ensuring the correctness and reliability of such content is paramount. To this end, we develop frameworks to evaluate and improve the performance of multimodal retrieval-augmented generation systems [29], focusing on detecting and mitigating hallucinations and optimizing context selection. These contributions ensure that AI-driven content remains trustworthy and relevant, supporting use cases ranging from virtual assistants to autonomous systems [30–32].

Finally, practical considerations in multimedia communication are addressed through deep learning-based real-time video quality and rate control frameworks [33–35]. In modern live streaming scenarios, especially over bandwidth-limited and fluctuating wireless channels, traditional fixed-parameter video encoders cannot adequately maintain perceptual quality and low latency. By leveraging deep learning to dynamically control encoding parameters and transmission rates, these methods support stable, high-quality experiences crucial for applications such as telepresence and immersive augmented reality, aligning with 6G’s focus on QoE and edge intelligence.

The goal of this dissertation is to develop a comprehensive suite of tools

and frameworks that collectively advance the design of efficient, intelligent, and semantically-aware communication and learning systems, addressing the key pillars of 6G networks. Through separate but thematically connected contributions in semantic communication, federated and decentralized learning, information freshness, trustworthy generation, and real-time media control, this work lays the foundation for future networks capable of delivering beyond connectivity toward truly intelligent, task-driven, and user-centric information systems. In the remainder of this introduction, we summarize the chapters of this dissertation.

In Chapter 2, we develop a transformer-aided semantic communication framework that leverages attention scores from vision transformers to identify and selectively transmit the most semantically important image patches under bandwidth constraints. We first introduce a binary masking approach that transmits only the most relevant patches, preserving critical content for tasks such as classification. Building on this, we propose a multi-resolution extension where patches are encoded at different resolutions based on their semantic importance and channel conditions, enabling finer-grained adaptation and improved reconstruction fidelity. Finally, we extend the framework to a multi-modal setting by integrating user-provided textual queries using cross-modal attention, allowing the system to emphasize regions of an image aligned with both visual content and user intent. Through extensive experiments on TinyImageNet and COCO datasets, we demonstrate significant gains in semantic fidelity, classification accuracy, and bandwidth efficiency, showcasing the flexibility and effectiveness of the proposed approach.

In Chapter 3, we introduce a semantic multi-resolution communication (SMRC)

framework that enables end-to-end communication with multiple resolution levels while preserving semantic information with progressively finer accuracy. Inspired by multi-task learning, we propose a multi-head autoencoder architecture that hierarchically encodes data into successive sub-blocks, each improving reconstruction performance and semantic precision when combined. We demonstrate that each layer can be flexibly tuned to prioritize either reconstruction quality or semantic feature accuracy, allowing for adaptive trade-offs based on application needs such as surveillance or object detection. Through extensive experiments on MNIST and CIFAR-10 datasets, we show that SMRC achieves superior reconstruction quality and semantic accuracy compared to traditional separate source-channel coding methods, highlighting its potential for bandwidth-efficient, goal-oriented communication systems.

In Chapter 4, we propose FedGradNorm, a personalized federated multi-task learning (PF-MTL) framework designed to address statistical heterogeneity and communication constraints in federated learning systems. By dynamically adjusting task-specific gradient weights, FedGradNorm ensures fair and balanced learning across diverse client tasks, improving convergence speed and performance compared to equal-weighted approaches. We provide theoretical convergence guarantees and validate the effectiveness of our method through experiments on the MTFM (multi-task facial landmark) and RadComDynamic datasets. Our results demonstrate superior performance in terms of fairness, learning speed, and robustness to imbalanced data, highlighting the advantages of our proposed approaches in practical federated learning scenarios.

In Chapter 5, we propose a personalized decentralized multi-task learning framework that dynamically adapts the communication topology among clients to address data heterogeneity in decentralized settings. By introducing a transference-based metric that quantifies positive or negative correlations between tasks, we design a dynamic connectivity matrix that links positively correlated clients and disconnects negatively correlated ones. This approach improves learning performance and accelerates convergence by preventing negative task interference and reducing unnecessary communication overhead. We demonstrate the effectiveness of our method on a synthetic Gaussian dataset and the large-scale CelebA dataset, showing significant improvements in convergence speed and task-specific accuracy compared to naive fully connected topologies. Our results highlight the benefits of data-driven dynamic graph adaptation in decentralized personalized learning scenarios.

In Chapter 6, we introduce $r\text{Age-}k$, a novel communication-efficient federated learning framework that leverages the age of information (AoI) metric to simultaneously address data heterogeneity and reduce communication overhead. By incorporating age vectors at the parameter server, our approach selectively requests updates on the most outdated gradient indices, enabling efficient sparsification while preserving model performance. Moreover, $r\text{Age-}k$ employs these age vectors to cluster clients with statistically similar data distributions using DBSCAN, further enhancing communication efficiency and convergence speed. We validate the effectiveness of $r\text{Age-}k$ through extensive experiments on highly non-i.i.d. settings with MNIST and CIFAR-10 datasets, demonstrating that it achieves superior accuracy and faster

convergence compared to existing sparsification methods like r Top- k . Our results highlight the potential of AoI-based strategies to improve both learning performance and resource efficiency in federated learning systems.

In Chapter 7, we investigate the version age of information (AoI) in networks with multiple sensing nodes that maintain distributed views of a single underlying source. By extending the conventional concept of version AoI from a single source node to a distributed sensing scenario, we derive recursive expressions for the average version AoI across arbitrary subsets of nodes. We analyze the asymptotic scaling behaviors for different network topologies, including line, ring, and fully connected networks, and show that distributed sensing does not degrade AoI performance compared to single-source settings. Notably, we demonstrate that in ring and line networks, the average AoI per node can scale as $O(\sqrt{n})$ even when the number of sensing nodes scales only as $O(\sqrt{n})$, highlighting the efficiency of distributed sensing. Through rigorous theoretical analysis and illustrative examples, we reveal the potential of leveraging multiple views to maintain information freshness in large-scale, dynamic networks.

In Chapter 8, we develop a comprehensive framework, RAG-Check, to evaluate and improve multi-modal retrieval-augmented generation (RAG) systems with a focus on mitigating hallucinations. We introduce two key metrics: the relevancy score (RS), which quantifies the relevance of retrieved entries to user queries, and the correctness score (CS), which evaluates the factual accuracy of generated responses with respect to the retrieved context. We design dedicated neural network architectures for both RS and CS, leveraging human-labeled datasets and ChatGPT-derived

samples to achieve high alignment with human judgments. We further show that integrating RS into the retrieval process via a re-ranking mechanism significantly improves selection accuracy over traditional CLIP-based top- k methods. Extensive experiments on the COCO dataset and human-annotated benchmarks demonstrate that our approach reduces hallucinations and enhances overall response reliability across different multi-modal RAG configurations. Our framework provides practical tools for both evaluating and improving the performance of multi-modal RAG systems, addressing critical challenges in modern vision-language applications.

In Chapter 9, we present a deep learning-based real-time rate control (RTRC) framework for live video streaming over wireless networks, addressing the challenges posed by dynamic channel conditions and fluctuating video bitrates. We propose a novel rate control unit (RCU) that leverages a deep neural network architecture to dynamically predict optimal quantization parameters (QP) for each video chunk based on instantaneous channel quality and video content characteristics. Our approach ensures that the encoded video bitrate stays just below the available channel bitrate, minimizing packet drops and maximizing perceived video quality. Extensive experiments conducted on the QCIF dataset and a range of publicly available videos demonstrate substantial gains over state-of-the-art adaptive bitrate streaming methods such as DASH and HLS, achieving up to 20 dB improvements in PSNR and a packet drop rate as low as 0.002. The results highlight the effectiveness of the proposed RTRC framework in achieving high video quality and bandwidth efficiency, making it a promising solution for real-time wireless video streaming applications.

In Chapter 10, building on the deep learning-based real-time rate control

framework introduced in the previous chapter, we propose a deep learning-based real-time quality control (RTQC) framework for live video streaming over wireless networks. Instead of focusing solely on controlling bitrate, the RTQC framework dynamically selects quantization parameters (QP) to ensure that each video chunk meets a specified minimum video quality requirement, measured in terms of peak signal-to-noise ratio (PSNR), while minimizing the encoded bitrate. By leveraging a deep neural network architecture conditioned on both video content and target PSNR, the proposed control unit accurately predicts QP values in real time, achieving high conformance probability and optimal bandwidth efficiency. Extensive experiments on the QCIF dataset and various public video datasets demonstrate that the RTQC approach achieves up to 2.5 times lower bitrate than state-of-the-art adaptive bitrate streaming schemes such as DASH, while maintaining a non-conformance probability below 1%. These results highlight the capability of RTQC to deliver high-quality, bandwidth-efficient live streaming, advancing the state of the art in real-time video transmission.

In Chapter 11, we present conclusions of this dissertation.

CHAPTER 2

Adaptive Transformer-Aided Semantic Communication with Multi-Resolution Encoding

2.1 Introduction

As we move towards 6G communication systems, semantic communication is emerging as a key enabler for advanced, immersive applications. Unlike traditional communication systems that focus on accurate data transmission, semantic communication [1, 2, 4, 11] prioritizes the meaning and purpose behind the transmitted data. This shift is crucial for supporting next-generation services, such as holographic telepresence [36] and haptic/tactile communication [37], where the goal is not just to reconstruct data but to ensure that the transmitted content fulfills a specific objective in real-time, often with stringent bandwidth and latency constraints. In holographic telepresence [36], for example, high-quality 3D holograms must be rendered at the receiver, demanding efficient communication that prioritizes essential information, such as key facial features or body movements. Similarly, in haptic/tactile communication [37], the tactile sensations transmitted for remote surgery or virtual reality interactions must be delivered with ultra-low latency and high accuracy to

ensure precision.

In this work, we present an alternative view for semantic communication, as elaborated in Section 2.2, where the objective extends beyond simple data reconstruction. In the most common view of semantic communication, the aim is to transmit and reconstruct an image at the receiver, which minimizes the inaccuracies in visual and classification performance. In our view of semantic communication, the transmitted data is encoded such that its reconstruction at the receiver has higher fidelity for the parts that are more relevant to its content and the given task, say class label or the query given by the user. This means that, parts of the image that are more relevant to the class label or user query, and hence are more important for accurate classification, are encoded at higher rates, while less relevant parts or elements unrelated to the semantic content, such as the background, can be conveyed at a lower resolution and lower rate. Note that, the semantics can also be adapted during transmission for other objectives, such as anomaly detection, where the set of image class labels, and consequently the semantics, would change. This approach is sometimes referred to as goal-oriented communication [1–4].

In our approach, we extend the concept of semantic communication to address the challenges posed by fading channels and varying channel capacities. Specifically, we use separate source and channel coding. Due to the fading, the channel rate will be varying over time. Hence, for each block of the transmission, we will adapt the source encoding rate to the available channel rate. This real-time, adaptive transmission process ensures that the reconstructed image at the receiver maintains high fidelity in its most important regions, while less critical areas can be represented with

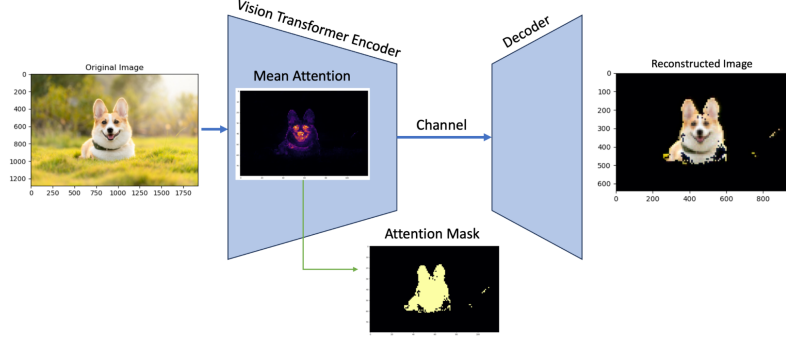


Figure 2.1: Transformer-aided semantic communication framework.

lower resolution or even left blank. By optimizing which image patches to transmit at varying resolutions, based on their semantic importance, we can adhere to the channel constraints dynamically, achieving better overall performance compared to conventional methods.

Several studies [1, 2, 10] have introduced end-to-end communication systems aimed at achieving semantic communication. These systems focus on reconstructing the image at the receiver side while ensuring the retention of semantic information during the encoding process. Typically, this approach incorporates a semantic classifier on the receiver side during training, guiding the encoded data to preserve critical elements of the image for analytical purposes. However, this method faces a limitation due to the encoder’s static size, which cannot adapt to fluctuating bandwidth conditions. Moreover, encoding the entire dataset into a fixed-size format complicates the task of highlighting the most crucial segments of the original data. To overcome this challenge, the authors in [33] introduce a macroblock-wise quantization method. This approach allows for encoding the original data at the macroblock level, tailored to their significance as determined by the receiver’s analysis. Nonetheless, this method necessitates an advanced deep learning module capable of deciding

the optimal encoding quality for each macroblock.

Transformers, as introduced by [13], revolutionize data encoding by leveraging the interconnections within different sections of the input. The effectiveness of transformers primarily stems from their attention unit, which assigns attention scores to various parts of the input, highlighting their relevance to the intended task. Previous studies [3, 4, 38] have explored using transformers for semantic communication. For instance, [4] employs a transformer to create a unified model that encodes multi-modal data for various tasks. [38] introduces an end-to-end framework for wireless image transmission, utilizing transformers to harness the semantics of the source image and feedback signals received in each interaction to generate coded symbols effectively. However, to the best of our knowledge, prior studies have not explored the use of transformers to design efficient semantic communications under limited channel bandwidth constraints.

As first contribution in this work, we focus on image data and utilize the vision transformer (ViT) as introduced in [12]. we introduce a transformer-aided semantic communication framework that employs the attention scores from a ViT to assess the semantic importance of each image patch. These scores act as a proxy for how critical each patch is to the intended task. Using this, we generate a binary attention mask that selects a subset of the most informative patches for transmission [11]. This strategy ensures that even under stringent bandwidth constraints, the key semantic content of an image is preserved, while less relevant patches are discarded.

The transformer’s attention unit excels at evaluating relationships, both among the patches themselves and between each patch and the specific task the transformer

is trained to perform. For example, in classification tasks, the attention unit assesses the interdependency among patches during training. Additionally, it evaluates the contribution of each patch towards accurately predicting the image’s class, thereby determining the relevance of each segment in achieving the task’s objective. In this work, we leverage the attention mechanism within the vision transformer encoder to develop a framework that effectively identifies the most informative patches of an image for a given task and encodes and transmits only those patches to the receiver. These selected regions are the ones most capable of contributing meaningfully toward achieving specific objectives, such as image classification. This method not only conserves bandwidth but also enhances the efficiency and effectiveness of the data analysis process on the receiver’s end.

Building on this foundation, our second contribution proposes a more flexible multi-resolution semantic communication system. Instead of a binary decision to transmit or discard a patch, we introduce a quantized attention-guided compression scheme. Patches are encoded and transmitted at varying levels of resolution based on their attention-derived importance scores and the current real-time channel rate [39]. A novel algorithm is developed to optimally assign each patch to one of several resolution levels, maximizing semantic preservation while adapting to available bandwidth. This finer-grained control leads to improved reconstruction fidelity and task performance, particularly in bandwidth-limited environments by ensuring that each part of the image is reconstructed in alignment with its semantic importance.

The work [40] investigates use of attention scores to pass embeddings from

one layer to the next, aiming to reduce computational overhead during transformer model fine-tuning. This approach can be understood as prioritizing the most relevant embeddings for progression through the layers. While this work falls outside the scope of semantic communication, it shares a conceptual similarity with our approach of encoding the most relevant patches for semantic communication. However, unlike our method, no encoding is performed on the embeddings selected for layer-to-layer transmission. In contrast, we employ multi-resolution encoding based on the value of the attention scores. To the best of our knowledge, our framework is the first to employ multi-resolution encoding using ViT attention scores, adapting to varying channel conditions within the context of semantic communications.

While transformer attention effectively identifies informative regions in many images, it can struggle with complex scenes containing multiple objects or ambiguous context. To overcome this limitation, our third contribution introduces a multi-modal semantic communication framework that leverages cross-modal attention between images and user-provided textual queries [41]. This enables user-guided selection of important regions, refining the system’s focus toward semantically relevant content that aligns with the user’s intent. Using pretrained vision-language models like CLIP [42], we first obtain embeddings for both the image and the user query. To better capture the visual context, we employ a content-dependent transfer [43] to refine the text embedding based on the image content. Then, leveraging the image embedding, the conditioned text embedding, and a mask proposal mechanism [44], we compute an informativeness score for each image patch, quantifying its relevance to the user query. These scores are then used in our multi-resolution

encoding algorithm, enabling the system to transmit patches at varying quality levels while adhering to bandwidth constraints. This approach is especially effective in images where self-attention alone fails to capture the true semantic priority.

Previous work in open-vocabulary semantic segmentation, such as MAFT+ [43], leverages cross-attention mechanisms to fuse image and text representations, enabling pixel-level segmentation for arbitrary text queries. These methods typically generate hard binary masks (0 or 1) indicating the presence or absence of queried concepts in the image. In contrast, our approach computes soft informativeness scores for each image patch, reflecting the degree of relevance to the user query rather than making binary decisions. Furthermore, while open-vocabulary segmentation models often rely on short, single-word queries drawn from predefined datasets, our system supports more expressive, natural-language queries. To the best of our knowledge, this is the first work to use cross-attention-derived multi-modal relevance scores for semantic communication, enabling adaptive transmission of image content based on query relevance.

The main contributions of our work can be summarized as: (I) We present a transformer-based semantic communication framework in which attention scores derived from a vision transformer are utilized to identify the most semantically informative patches of an image. A binary mask is then generated to selectively transmit only these critical regions, allowing for effective communication under stringent bandwidth constraints. (II) We develop a channel-aware compression scheme that extends the binary masking approach by categorizing patches into multiple resolution levels based on their semantic importance and real-time available channel rate.

A novel rate-allocation algorithm is proposed to ensure that patches are encoded at appropriate resolutions while adapting to instantaneous channel capacity. (III) We propose a user-guided semantic communication framework that integrates textual input with visual data via a cross-attention mechanism. This allows users to specify communication goals through natural language queries, which in turn refine the selection of image regions for transmission. The identified patches are then encoded at varying resolutions using our prior allocation strategy, resulting in more contextually relevant and goal-aligned communication.

The rest of the work is organized as follows. In Section 2.2, we introduce an alternative perspective on semantic communication that motivates our approach. Section 2.3 presents the system model and details each of our three main contributions. In Section 2.4, we report our experimental results and provide analysis to validate the effectiveness of our method. Finally, we conclude the work in Section 2.5.

2.2 Alternative Views of Semantics and Semantic Communications

In this section, we formally define semantic meaning in the context of stochastic processes. Let $\mathcal{S}$ denote the context that is defined as the set of all possible semantic classes. For a semantic class $S \in \mathcal{S}$, we observe an instance of a medium (say an image) denoted by X where X is a random variable with distribution $p(X|S)$. The semantic classes follow distribution $p(S)$.

The goal is to compress X into a semantic representation W such that the

joint information between X and S is preserved as much as possible within a bitrate constraint $H(W) \leq r$. Formally, W is chosen to minimize a distortion metric $d(X, S; W)$, where d quantifies the discrepancy between the true pair (X, S) and their reconstructions $(\hat{X}(W), \hat{S}(W))$ derived from the representation W .

A commonly used formulation for the distortion is a weighted sum of two terms:

$$d_{\text{linear}}(X, S; W) = \alpha_s d_s(S, \hat{S}) + \alpha_x d_x(X, \hat{X}), \quad (2.1)$$

where $\alpha_s, \alpha_x > 0$ are weights that trade off between semantic classification error and signal reconstruction fidelity. For instance, when d_s is cross-entropy and d_x is mean squared error (MSE), this distortion function aligns with loss functions frequently used in semantic autoencoders [1, 2].

An alternative formulation employs an information-theoretic view, where distortion is defined as conditional entropy:

$$d_{\text{entropy}}(X, S; W) = H(X, S|W), \quad (2.2)$$

which leads to maximizing the mutual information $I(X, S; W) = H(X, S) - H(X, S|W)$ under the constraint $H(W) \leq r$. Since $H(X, S)$ is fixed, this corresponds to preserving as much semantic information about (X, S) as possible within a limited encoding rate.

Now consider the case where X is a structured observation, e.g., an image

segmented into P distinct patches: $X = (X_1, X_2, \dots, X_P)$. The relevance of each patch to the semantic class may vary. For example, for an image labeled $S = \text{“dogs”}$, the patches containing the dog are clearly more semantically informative than those depicting background elements. From a probabilistic perspective, this heterogeneity is captured by the varying conditional distributions $p(X_i|S)$, which may have different entropies.

We assume a disjoint patch-wise encoding scheme, where the semantic representation W is factorized as $(W_1, W_2, \dots, W_P)$, with each W_i corresponding to an individual patch X_i . The total bitrate constraint becomes $\sum_{i=1}^P H(W_i) \leq r$. Under this model, the overall distortion $d_x(X, \hat{X})$ is the sum of per-patch distortions, each of which depends on how the overall rate r is distributed among patches: $\{r_i = H(W_i)\}_{i=1}^P$. Likewise, the semantic distortion $d_s(S, \hat{S})$ is also influenced by the patch-wise allocation.

In our approach, we adopt a practical two-step solution. First, we estimate the semantic importance of each patch using attention scores derived from a pre-trained vision transformer. These scores guide the allocation of individual rates r_i to patches. Next, we minimize the MSE for each patch i by training encoders and decoders under their respective assigned rate budgets r_i .

2.3 System Model and Problem Formulation

We consider a goal-oriented communication scenario in which an image must be transmitted over a fading channel, with the objective of supporting downstream

tasks, such as classification, rather than merely achieving high-fidelity reconstruction. To enable this, the image is partitioned into fixed-size patches. The transmitter encodes these patches at varying levels of resolutions based on their semantic importance, while adapting to the constraints of the available channel rate. At the receiver side, patches are decoded according to their received resolution, allowing the system to retain the most critical content even under fluctuating bandwidth conditions. In the following sections, we present the system model corresponding to our three main contributions.

2.3.1 Transformer-Aided Semantic Communication

In this section, we consider a simple scheme where we select a subset of patches and use similar rates for encoding each of the selected patches. The selected patches correspond to the section of the image which contains more relevant information to the semantic content, e.g., the pieces of the faces of the dog in Fig. 2.1 where the semantic content is the class “dog”. We build our solution based on a ViT [12]. As shown in Fig. 2.2, the proposed end-to-end semantic communication system is composed of an encoder, a decoder, and a classifier where the encoder consists of three blocks: projector, ViT, and compressor.

2.3.1.1 Projector

The input is an image from a set $\mathcal{X} \in \mathbb{R}^{(3,h,w)}$, where h and w denote the image’s height and width and 3 is the number of image channels, i.e., RGB colors. Each

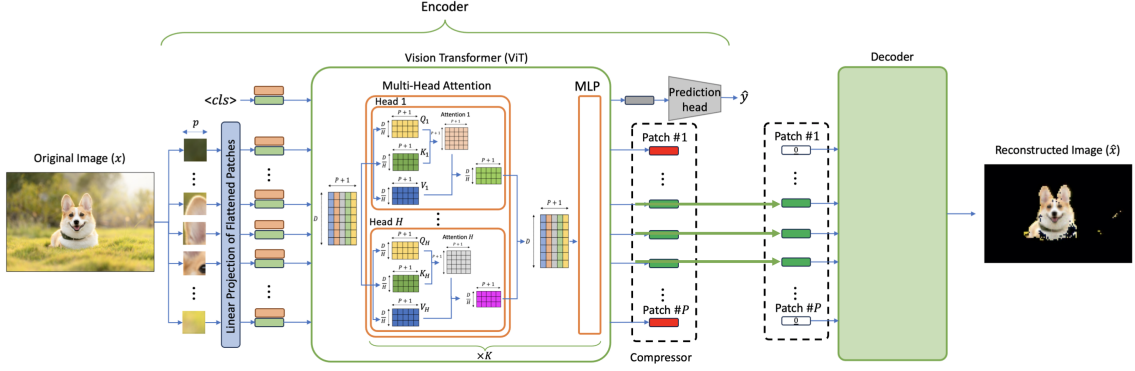


Figure 2.2: System model for transformer-aided semantic communication.

image $X \in \mathcal{X}$ is divided into sequences of patches $X_i, \forall i \in [P]$, each of size $p \times p$, resulting in $P = \frac{hw}{p^2}$ patches per image. We also use a one-dimensional indexing (1D index) for the position of the patches by numbering the patches sequentially. Following this, each patch X_i is flattened to vector $\bar{X}_i \in \mathbb{R}^{(3p^2, 1)}$ which are in turn transformed into vectors $\hat{X}_i$ of dimension D via $W_{proj} \in \mathbb{R}^{(D, 3p^2)}$ that is analogous to tokenization in natural language processing (NLP) tasks.

2.3.1.2 ViT

For each image, ViT first forms the matrix $\tilde{X} = [\tilde{X}_{cls}, \tilde{X}_1, \dots, \tilde{X}_P] \in \mathbb{R}^{(D, P+1)}$ which consist of a random vector $\tilde{X}_{cls}$ and sequence of P vectors $\tilde{X}_i$ for each patch in order, where $\tilde{X}_i$ is the superposition of $\hat{X}_i$ with the positional encoding vectors for the respective patches. After passing through ViT, the transformation of the vector $\tilde{X}_{cls}$ will hold the concise information for classification of the image.

The matrix $\tilde{X}$ undergoes processing through K transformer layers sequentially, each of which containing multi-head attention (MHA) unit followed by a multi-layer perceptron (MLP) layer f_{MLP} with a non-linear activation function. The output of

the last transformer layer $z = [z_{cls}, z_1, \dots, z_P] \in \mathbb{R}^{(D, P+1)}$ comprises z_{cls} which is the transformation of $\tilde{X}_{cls}$ used for classification, followed by P vectors z_i that contain the encoded data for patch i .

Each transformer layer in ViT consists of H parallel attention heads. For simplicity of the notation, we drop the indexing of the layers and in the following we explain a generic layer. The input $\tilde{X}$ is vertically partitioned into H matrices, i.e., $\tilde{X} = [\tilde{X}_1, \dots, \tilde{X}_H]^T$ where $\tilde{X}_h \in \mathbb{R}^{(D/H, P+1)}, \forall h \in [H]$ is the input for the head h .

The query $Q^{(h)}$, key $K^{(h)}$, and value $V^{(h)}$ matrices of size D/H by $P+1$ are produced by using the trainable set of weight matrices $W_Q^{(h)}$, $W_K^{(h)}$, and $W_V^{(h)}$, of size D/H by D/H , respectively. For each head h , the attention score matrix $A^{(h)}(Q, K) \in \mathbb{R}^{(P+1, P+1)}$ for head h is then calculated as follows,

$$A^{(h)}(Q, K) = \text{softmax} \left(\frac{Q^{(h)} K^{(h)T}}{\sqrt{D}} \right) \quad (2.3)$$

The output for head h is obtained by a linear transformation of the value matrix $V^{(h)}$ through attention scores $A^{(h)}$. The output from each attention head $O^{(h)}$ is concatenated to form the output of the MHA unit, represented as $O = [O^{(1)}, \dots, O^{(H)}] \in \mathbb{R}^{(D, P+1)}$. This output O is then fed into an MLP with transfer function $f_{MLP}(O, \phi_{mlp})$ to produce the output for the transformer layer, where ϕ_{mlp} is the MLP parameters. Note that each layer has its own parameters that are learned in the training phase.

ViT predicts the input image class $\hat{y}$ by processing z_{cls} through a single-layer

neural network $f_{predictor}$, i.e., $\hat{y} = f_{predictor}(z_{cls}, \phi_{pred})$, where ϕ_{pred} is the parameters of the prediction head.

2.3.1.3 Compressor, Decoder, and Classifier

The compressor prepares transmission packets based on the output from the ViT and its internal states. We assume that the channel is error-free and it has limited capacity which may fluctuate over time. The compressor’s role is to adapt the packet bitrates to match the instantaneous channel capacity. The received packets are re-arranged in a proper format and passed to the decoder, which reconstructs the image. The classifier processes the reconstructed image to predict its class.

2.3.1.4 Main Contributions

We introduce the use of transformers for the purpose of semantic communications by compressing the output of the transformer blocks of a ViT based on the internal state of the ViT, namely, the attention score matrix. Specifically, we select a portion of the patches based on their attention scores such that the rate of the transmitted packet for the selected image is within the instantaneous bitrate constraint of the channel. The proposed scheme is built on using the attention score of a patch as a representative of its information metric with respect to the semantic content.

First, we perform supervised training of ViT by jointly training the projector, ViT, and its dedicated prediction head $f_{predictor}$. During the training phase, we employ cross-entropy loss $\mathcal{L}_{CE}(\hat{y}, c)$ between the true class label c and $\hat{y}$ to jointly

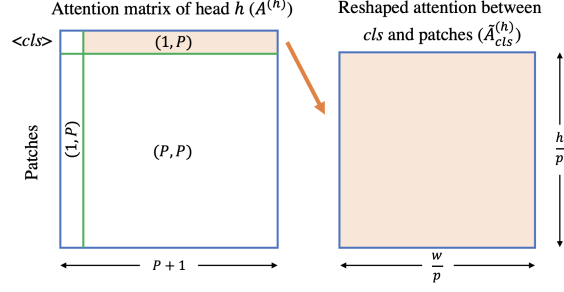


Figure 2.3: Deriving $\tilde{A}_{cls}^{(h)}$ from $A^{(h)}$.

train the projector, transformer, and classifier.

For a trained transformer the attention score matrices $\{A^{(h)}, h \in [H]\}$ in the last transformer layer contain information about the semantic content of the patches. Illustrated in Fig. 2.3, we use the first row of the attention matrix denoted by $A_{cls}^{(h)} = A^{(h)}[0, :], h \in [H]$ as a measure of the relevancy of patches to the semantic content of the input image. More precisely, the value of the score in position i in vector $A_{cls}^{(h)}$ highlights the significance of the patch with 1D index i for accurate classification. It is convenient to reshape $A_{cls}^{(h)}$ into a square matrix of size $(w/p, h/p)$ to form $\tilde{A}_{cls}^{(h)}$ by using the relationship between the two-dimensional position and 1D index of the patches. In order to combine the information from all heads, we calculate the average of $\tilde{A}_{cls}^{(h)}$ for all $h \in [H]$ heads denoted as $\tilde{A}_{cls}$. Fig. 2.4 shows this averaging process and highlights its effect in recognizing the position of the patches that are important in an illustrative example.

We aim to use these scores to create a binary mask, allowing only those patches that positively affect the classification to be transmitted as marked by one as opposed to zero for the patches that are not transmitted.

The compressor uses a design parameter $\alpha \in [0, 1]$ to find a threshold λ such

that by selecting the patches with higher scores than λ their cumulative bitrate is maximized but does not exceed α portion of the bitrate constraint. Next, the compressor selects additional random patches from the remaining ones to maximize the transmission rate without exceeding the bitrate constraint. Hence, the transmitted packet $\hat{z}$ consists of the encoded data for the selected patches, and a positional mask $M \in \mathbb{R}^{(w/p, h/p)}$ which indicates which patches are selected. In our simulations, we have ignored the bitrate required for the positional mask as it is negligible in comparison to the packet bitrate.

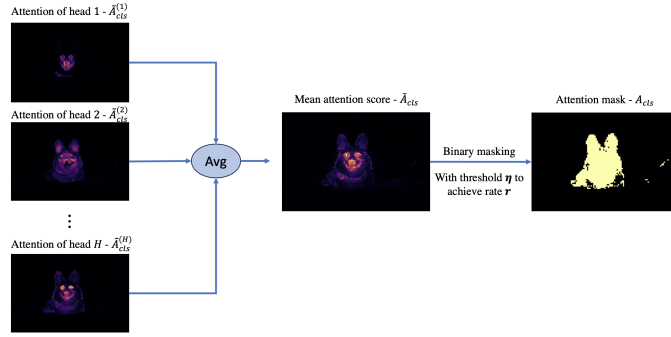


Figure 2.4: Obtaining attention mask from ViT attention unit.

The decoder $f_{decoder}(\hat{z}, \theta)$ is designed to reconstruct the image by minimizing the loss function

$$\mathcal{L}_{decoder} = ||(M \otimes B) \odot (X - \hat{X})||^2 \quad (2.4)$$

by training the parameters θ , which is aimed to minimize the mean squared error (MSE) for the selected patches in M where B is a matrix of size $p \times p$, $\odot$ is the Hadamard product, and $\otimes$ is the Kronecker product. Based on our assumption of attention score as the representative of the information metric with respect to the

semantic content, this design choice can be interpreted as maximizing the reconstruction performance of different parts of the image proportional to their semantic information. The overall algorithm is shown in Algorithm 1.

2.3.2 Multi-resolution Transformer-Aided Semantic Communication

In this section, we extend our initial framework presented in Section 2.3.1 by introducing a multi-resolution semantic communication system. Unlike the previous binary transmit-or-discard approach, the new design employs a quantized compression scheme guided by attention-derived importance scores. Each image patch is assigned an optimal resolution level based on its semantic significance and the instantaneous channel conditions. This enhancement enables the system to make more efficient use of bandwidth while preserving the most meaningful visual information. As shown in Fig. 2.5, our end-to-end system contains three main parts: attention-guided resolution selector, encoder, and decoder.

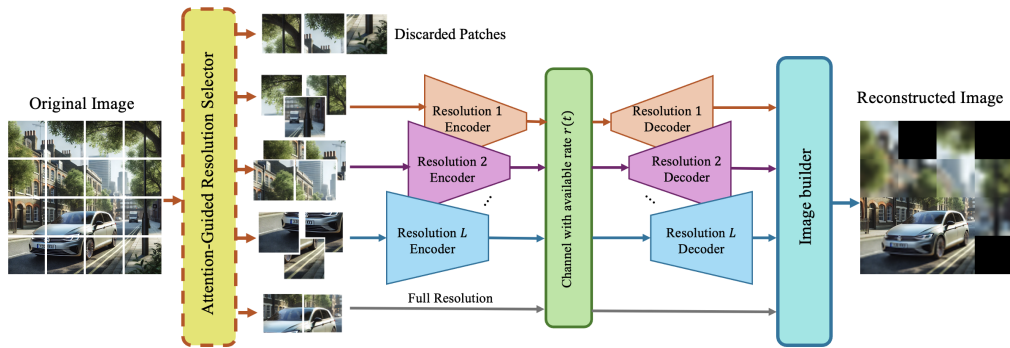


Figure 2.5: Channel-aware multi-resolution semantic communication framework.

Algorithm 1: Training with our proposed algorithm

```

1: Input: Image  $X \in \mathbb{R}^{(3,h,w)}$ ,  $c$ : class of the image
2: Initialization: Embedding dimension:  $D$ , path size:  $p$ , number of heads:  $H$ 
3: Parameters: Transformer encoder parameters at time  $t$ :  $\Theta_t := \{W_Q^{(h)}, W_K^{(h)}, W_V^{(h)}\}_{h=1}^H$ ,
    $\phi_{mlp}$ ,  $\phi_{pred}$ ,  $W_{proj}$ , Decoder parameters at time  $t$ :  $\theta_t$ 

4: Segmenting  $X$  into  $P$  patches each with dimension of  $(3, p, p)$ 
5: Do  $\bar{X}_i \leftarrow \text{flatten}(X_i)$ ,  $\bar{X}_i \in \mathbb{R}^{(3p^2, 1)}$ ,  $\forall i \in [P]$ 
6: Define  $W_{proj} \in \mathbb{R}^{(D, 3p^2)}$  with bias  $b$  ▷ Linear Projection
7: Do  $\hat{X}_i \leftarrow (W_{proj} \cdot \bar{X}_i + b)^T$ ,  $\hat{X}_i \in \mathbb{R}^{(D, 1)}$ ,  $\forall i \in [P]$ 
8: Add positional encoding to  $\hat{X}_i$  to form  $\tilde{X}_i$ 
9:  $\tilde{X} = [\tilde{X}_{cls}, \tilde{X}_1, \dots, \tilde{X}_P]$  s.t.  $\tilde{X} \in \mathbb{R}^{(D, P+1)}$  ▷ Enc. input
10: Partition  $\tilde{X}$  into  $H$  chunks s.t.  $\tilde{X} = [\tilde{X}_1; \tilde{X}_2; \dots; \tilde{X}_H]$ 
11: while  $t < \text{end time}$  do ▷ Encoder training
12:   for each  $h \in [H]$  (in parallel) do ▷ MHA
13:      $Q^{(h)} \leftarrow W_Q^{(h)} \cdot \tilde{X}_h$ ,  $K^{(h)} \leftarrow W_K^{(h)} \cdot \tilde{X}_h$ ,  $V^{(h)} \leftarrow W_V^{(h)} \cdot \tilde{X}_h$ 
14:      $A^{(h)} \leftarrow \text{softmax}\left(\frac{Q^{(h)} \cdot K^{(h)T}}{\sqrt{D}}\right)$  ▷ Attention Score
15:      $O^{(h)} \leftarrow V^{(h)T} \cdot A^{(h)}$ 
16:   end for
17:   MHA output:  $O = [O^{(1)}, \dots, O^{(H)}] \in \mathbb{R}^{(D, P+1)}$ 
18:   Assign  $z \leftarrow f_{MLP}(O, \phi_{mlp})$  ▷ Encoded data  $z$ 
19:   Determine the class  $\hat{y} = f_{predictor}(z_{cls}, \phi_{pred})$ 
20:   Calculate  $\mathcal{L}_{encoder} = \mathcal{L}_{CE}(\hat{y}, c)$ 
21:    $\Theta_{t+1} = \Theta_t - \eta_1 \nabla \mathcal{L}_{encoder}$ 
22: end while
23: while  $t < \text{end time}$  do ▷ Decoder training
24:   for each  $h \in [H]$  (in parallel) do
25:      $\tilde{A}_{cls}^{(h)} = A^{(h)}[0, :]$  and reshape  $\tilde{A}_{cls}^{(h)}: \mathbb{R}^{(1, P)} \rightarrow \mathbb{R}^{(\frac{h}{p}, \frac{w}{p})}$ 
26:   end for
27:    $\tilde{A}_{cls} = \mathbb{E}[\tilde{A}_{cls}^{(h)}]$ 
28:   for  $\forall i \in [\frac{h}{p}]$  and  $\forall j \in [\frac{w}{p}]$  do ▷ Attention mask
29:     if  $\tilde{A}_{cls}[i][j] \leq \lambda$  then  $A_{cls}[i][j] \leftarrow 0$ , else  $A_{cls}[i][j] \leftarrow 1$ 
30:   end for
31:   Reshape  $A_{cls}^{(h)}: \mathbb{R}^{(\frac{h}{p}, \frac{w}{p})} \rightarrow \mathbb{R}^{(P)}$ 
32:   for  $\forall i \in [P]$  do ▷ Transmitted encoded data
33:     if  $A_{cls}[i] == 0$  then  $\hat{z}[i] = 0$ , else  $\hat{z}[i] = z[i]$ 
34:   end for
35:    $\hat{X} = f_{decoder}(\hat{z}, \theta_t)$  ▷ Reconstruct the image  $\hat{x}$ 
36:   Expand  $A_{cls}$  to  $(h, w)$  dimensions
37:    $\mathcal{R}(X) \leftarrow X \cdot A^T$ ,  $\mathcal{R}(\hat{X}) \leftarrow \hat{X} \cdot A^T$ 
38:   Calculate  $\mathcal{L}_{decoder} = \|(M \otimes B) \odot (X - \hat{X})\|^2$ 
39:    $\theta_{t+1} = \theta_t - \eta_2 \nabla \mathcal{L}_{decoder}$ 
40: end while

```

2.3.2.1 Attention-Guided Resolution Selector

The attention-guided resolution selector assigns appropriate resolutions for each patch of the image to be encoded with varying rates, depending on their semantic content and available channel rate. Similar to Section 2.3.1, consider the image $X \in \mathcal{X}$ of size (c, h, w) , where w and h represent the width and height of the image, respectively, and $c = 3$ is the number of color channels (i.e., RGB). The image X is divided into patches of fixed size $p \times p$ for a total of $P = \frac{wh}{p^2}$ patches. Each patch is mapped by using a linear layer to an embedding of size d that is matched with the ViT embedding size. An additional embedding, called the CLS token, of the same size is appended to the beginning of the patch embeddings for each image. This CLS token is trained and used for classification purposes.

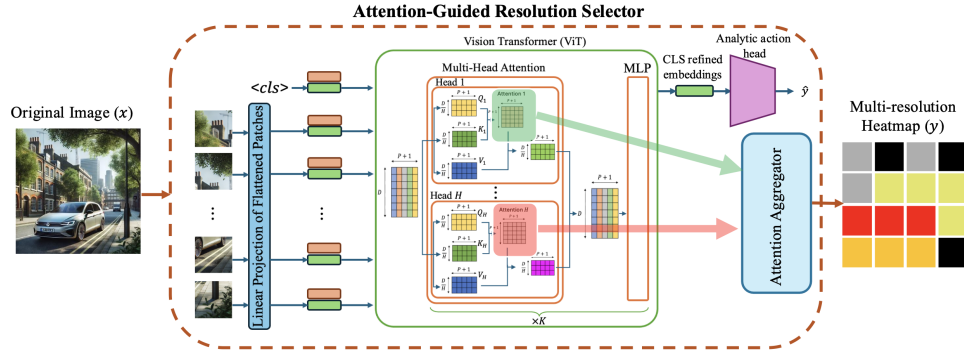


Figure 2.6: Attention-guided resolution selector block framework.

We reuse the ViT architecture from Section 2.3.1, where the network is trained in a supervised manner to perform semantic tasks such as image classification. The attention scores between the CLS token and each patch, extracted from the final transformer layer, serve as indicators of semantic relevance.

Similar to Section 2.3.1, once the vision transformer is trained for the specific

task, the attention matrix of the last transformer block is highly effective in finding the relevancy of each patch to the class label. Specifically, the first row of the attention matrix is a vector of length $1 + P$, where the last P values of the vector correspond to the cross attention between CLS and each image patches. The higher the attention value, the higher the importance of the corresponding patch for the desired analytic action. Using the attention matrix for all the heads in the last transformer block, we create a multi-resolution map of size $\frac{w}{p} \times \frac{h}{p}$ that determines the appropriate encoding level for each patch.

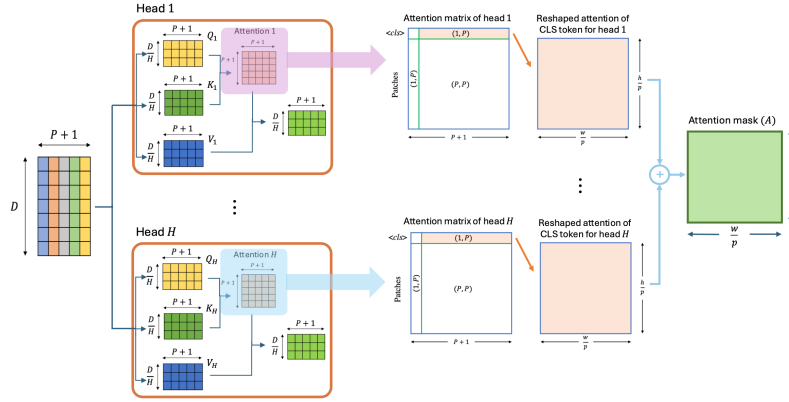


Figure 2.7: Attention aggregation block.

As shown in Fig. 2.7, we first generate an attention mask of size $\frac{w}{p} \times \frac{h}{p}$ based on the first row of the attention matrix for each head in the last transformer block. Then, we find the average attention mask for all H heads. Finally, we quantize the average attention mask to obtain the multi-resolution map. In particular, we solve an optimization problem to quantize the attention scores into appropriate quantization levels that minimizes the quantization error under two constraints: (i) The encoding rate for the multi-resolution map should not exceed the available channel rate, and (ii) the number of patches that are assigned a nonzero-rate is maximized.

Let the available channel rate be r , and assume we have L different resolutions for encoding patches. Algorithm 2 provides a solution to the above optimization problem. Algorithm 2 determines the thresholds for each resolution: First, we check whether the channel rate r is sufficient to encode all patches at the lowest available resolution. If the channel rate is too low to encode all patches at this resolution, we select as many patches as possible until the total bitrate meets the channel rate. If the channel rate exceeds the minimum required to encode all patches at the lowest resolution, we proceed with assigning resolution categories to each patch. We begin by calculating the total sum of the attention scores for all patches. Then, we normalize each attention score by multiplying it by $r/\text{sum}(\text{attention scores})$. After normalization, we apply a function LQ (please see Algorithm 3) that maps the attention scores to their nearest lower resolution. At this stage, the total encoding bitrate will be less than the available channel rate, meaning we have not fully utilized the channel. To make use of the remaining bandwidth, we apply a function UQ (please see Algorithm 3), which maps the attention scores to their nearest higher resolution. We then upgrade the resolution of patches with attention scores that have the smallest gap to their UQ, as these upgrades require the least additional bandwidth. This process is repeated until the total encoding bitrate matches the channel rate. The overall algorithms for LQ, UQ, and resolution selector block are shown in Algorithm 2 and 3.

Algorithm 2: Attention-guided resolution selector algorithm

```

1: Input: Resolution map matrix  $A \in \mathbb{R}^{(w/p, h/p)}$ , available channel rate  $r$ 
2: Parameters:  $p$  is a patch size,  $P$  total number of patches

3: Rate mapping:  $R = \{0 : 0, 1 : 12, 2 : 24, 3 : 48, 4 : 196\}$ 
4:  $A \leftarrow$  Flatten  $A$  to a 1D array of size  $P$ 
5:  $A \leftarrow A \times (r / \text{sum}(A))$  ▷ Normalizing the attention
6: Obtain  $T$  and  $Q$  from  $R$ 
7:  $mask \leftarrow$  Zero array of the same shape as  $A$ 
8:  $A\_perm \leftarrow$  Indices of  $A$  sorted in descending order
9: if  $r \leq R[1] \times P$  then
10:   for  $ind = 1$  to  $\text{int}(r/R[1])$  do  $mask[A\_perm[ind]] = 1$ 
11:   end for
12: else
13:    $LQ\_A \leftarrow LQ(A, T, Q)$  ▷ Calculate LQ of  $A$ 
14:    $UQ\_A \leftarrow UQ(A, T, Q)$  ▷ Calculate UQ of  $A$ 
15:    $A[LQ\_A == 196] = -Inf$ 
16:    $LQ\_A[LQ\_A = 0] = R[1]$ 
17:    $sum\_LQ\_rate = \text{sum}(LQ\_A)$ 
18:   while  $sum\_LQ\_rate < r$  do
19:      $diff = UQ\_A - A$ 
20:      $diff\_perm \leftarrow diff$  sorted in ascending order
21:      $LQ\_A[diff\_perm[0]] \leftarrow UQ\_A[diff\_perm[0]]$ 
22:      $A[diff\_perm[0]] \leftarrow LQ\_A[diff\_perm[0]] + \epsilon$ 
23:     if  $UQ\_A[diff\_perm[0]] = 196$  then
24:        $A[diff\_perm[0]] = -Inf$ 
25:     end if
26:      $sum\_LQ\_rate = \text{sum}(LQ\_A)$ 
27:      $UQ\_A \leftarrow UQ(A, T, Q)$ 
28:   end while
29: end if
30: if  $r > R[1] \times P$  then
31:   Replace values in  $LQ\_attn$ :  $R[1] \rightarrow 1, R[2] \rightarrow 2, R[3] \rightarrow 3, R[4] \rightarrow 4$ 
32: else
33:    $LQ\_attn \leftarrow mask$ 
34: end if
35:  $mask\_attn \leftarrow$  Reshape  $LQ\_attn$  to a  $w/p \times h/p$  array
36: return  $mask\_attn$ 

```

Algorithm 3: Lower and Upper Quantization Functions

Require: Array y , Threshold levels $T = [t_1, t_2, \dots, t_n]$, Quantized values $Q = [q_0, q_1, \dots, q_{n+1}]$

1: **function** LQ(y, T, Q) ▷ Lower Quantization

2: $x \leftarrow y.copy()$

3: **for** $i = 0$ to $\text{len}(x) - 1$ **do**

4: Find the highest index j such that $x[i] < T[j]$

5: $x[i] \leftarrow Q[j]$

6: **end for**

7: **return** x

8: **end function**

9: **function** UQ(y, T, Q) ▷ Upper Quantization

10: $x \leftarrow y.copy()$

11: **for** $i = 0$ to $\text{len}(x) - 1$ **do**

12: Find the lowest index j such that $x[i] > T[j]$

13: $x[i] \leftarrow Q[j + 1]$

14: **end for**

15: **return** x

16: **end function**

2.3.2.2 Encoders and Decoders

The entries of the resolution map indicate the resolutions in which patches are encoded. We train encoder-decoder pairs for each of the resolutions. Each resolution is assigned an encoding size out of a set of possible quantized encoding sizes $\{b_i, 1 \leq i \leq L\}$. The higher the resolution, the higher the assigned rate. For an encoding size b_i , the encoder takes an image patch $X_j, j \in \{1, 2, \dots, P\}$ as an input and produces an embedding of size b_i and the decoder takes this embedding and produces $\hat{X}_j$. The resolution map is designed to satisfy the available channel bitrate constraint, and relying on error-free communication not exceeding the available bitrate, we train the encoder-decoder for all resolutions as follows. Let $\hat{X}$ denote the reconstructed image for all the patches $\hat{X}_j$ at the encoding sizes given by resolution map for the image X , the following loss function can be used for the training,

$$\mathcal{L}_{Encoder-Decoder} = \mathcal{L}_{MSE}(X, \hat{X}). \quad (2.5)$$

However, to design a robust framework that is independent of the analytic action function, we train the encoder-decoder pairs independent of the analytic action function. In this case, we train the encoder-decoder pair for each rate on all image patches available for training using the MSE loss function on individual patch $X_j, j \in \{1, 2, \dots, P\}$ given by,

$$\mathcal{L}_{Encoder-Decoder} = \mathcal{L}_{MSE}(X_j, \hat{X}_j). \quad (2.6)$$

In this case, the encoder-decoder pairs for different resolutions are trained independently. Hence, a key advantage of this approach is that the encoder-decoder pairs do not need to be retrained when the semantic meaning or communication goal changes as imposed by different analytic action functions, i.e., the resolution selector retains the semantic meaning by properly incorporating the analytic action function in generation of the resolution map as the communication goal or task changes.

2.3.3 Multi-Modal Semantic Communication

In this section, we propose a *multi-modal semantic communication* framework that fuses visual and textual modalities to enable semantically guided compression and transmission under bandwidth constraints. As in the previous two sections, the system model includes an encoder and a decoder. The encoder consists of two main components: a multi-modal semantic extractor and a patch-wise multi-resolution encoder. The encoded image patches are transmitted and passed through resolution-specific decoders to reconstruct the final image. Unlike the previous sections, the system now takes two inputs: an image and a user command (See Fig. 2.8).

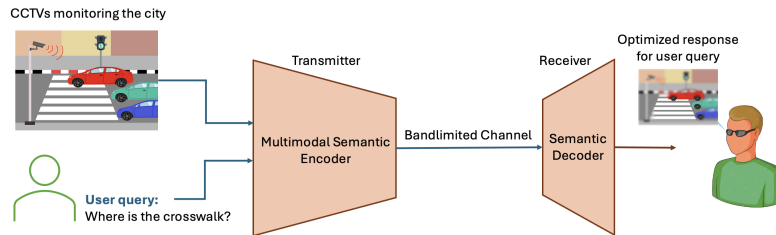


Figure 2.8: Multi-modal semantic communication (MMSC) framework.

2.3.3.1 Multi-Modal Semantic Extractor

The goal of the multi-modal semantic extractor is to fuse the vision and text inputs to produce a semantic score that indicates the most informative regions of an image given the user’s textual query. Our semantic extractor model builds upon the MAFT+ framework [43], originally developed for open-vocabulary semantic segmentation (OSS). First, as shown in Fig. 2.9, the image input $x \in \mathbb{R}^{3 \times h \times w}$ is processed by a convolutional CLIP-Vision (CLIP-V) backbone to extract a feature pyramid $F = \{F^0, F^1, F^2, F^3\}$, where $F^i \in \mathbb{R}^{d \times h_i \times w_i}$ and the spatial strides relative to the input are $\{4, 8, 16, 32\}$ respectively, with a feature dimension d .

These visual features are processed by a *MaskFormer* [44, 45] proposal generator, which first upsamples and fuses the multi-scale features into a dense per-pixel embedding map $E_{\text{pixel}} \in \mathbb{R}^{d \times h \times w}$. A fixed set of N learnable query tokens $\{q_i\}_{i=1}^N$ is passed through a transformer decoder that performs cross-attention with E_{pixel} , producing a set of mask embeddings $E_{\text{mask}} = [E_{\text{mask}}^{(1)}, \dots, E_{\text{mask}}^{(N)}] \in \mathbb{R}^{N \times d}$. These embeddings are then projected back onto the spatial domain via dot product to compute the mask logits $M \in \mathbb{R}^{N \times h \times w}$, where each mask logit is given by:

$$m_i(h, w) = E_{\text{mask}}^{(i)\top} E_{\text{pixel}}[:, h, w] \quad (2.7)$$

The resulting mask logits represent unbounded scores indicating the likelihood of each pixel belonging to each proposal.

During training, a set of class prompts C indicating all possible objects is

embedded via CLIP-Text (CLIP-T) encoder $f_t(\cdot)$, producing text embeddings $T = [t_1, \dots, t_{|C|}] \in \mathbb{R}^{d \times |C|}$. T is refined then via two transformer cross-attention layers called content-dependent transfer (CDT), attending on the flattened highest-level visual feature $F_{\text{flat}}^3 \in \mathbb{R}^{d \times h_3 w_3}$:

$$T_{j+1} = T_j + \text{TransLayer}_j(T_j, F_{\text{flat}}^3) \text{ for } j = 0, 1, \text{ with } T_0 = T \quad (2.8)$$

This would yield a conditioned embedding $\hat{T} \in \mathbb{R}^{d \times |C|}$. Meanwhile, mask pooling on F^3 using each mask logit yields mask embeddings $V \in \mathbb{R}^{N \times d}$. We compute per-mask classification scores:

$$S_{\text{cls}} = \left(V \hat{T} \right)^\top \in \mathbb{R}^{|C| \times N} \quad (2.9)$$

When projecting these scores onto the full image plane, we form dense semantic maps of size $|C| \times h \times w$ via:

$$S(c, h, w) = \sum_{i=1}^N S_{\text{cls}}(c, i) \cdot \sigma(m_i(h, w)) \quad (2.10)$$

where σ is the sigmoid activation. The final pixel-wise cross-entropy is applied against ground-truth labels to train CDT and CLIP-V.

Three losses are used during training. First, the mask proposal loss L_P uses Hungarian matching [46] between the predicted masks M and ground-truth masks, combining binary cross-entropy and Dice losses. This loss updates only the MaskFormer proposal generator, keeping CLIP-V frozen. Second, the mask-aware clas-

sification loss updates both the CDT layers and CLIP-V using ground-truth class labels c_i^* :

$$\mathcal{L}_{ma} = \frac{1}{|\mathcal{M}|} \sum_{i \in \text{matched}} -\log \frac{\exp(S_{\text{cls}}(c_i^*, i))}{\sum_c \exp(S_{\text{cls}}(c, i))} \quad (2.11)$$

Third, the representation compensation loss $\mathcal{L}_{rc}$ preserves original CLIP-V representation by matching multi-scale pooled features between the fine-tuned CLIP-V and a frozen CLIP-V* using SmoothL1 loss (more detailed explanation is available in [43]):

$$\mathcal{L}_{rc} = \sum_{k \in \{1,2,4\}} \text{SmoothL1}(F_k^p, \hat{F}_k^p) \quad (2.12)$$

Gradients from $\mathcal{L}_{ma}$ and $\mathcal{L}_{rc}$ are backpropagated into CLIP-V (with $\mathcal{L}_{ma}$ also updating CDT), while CLIP-T remains entirely frozen throughout.

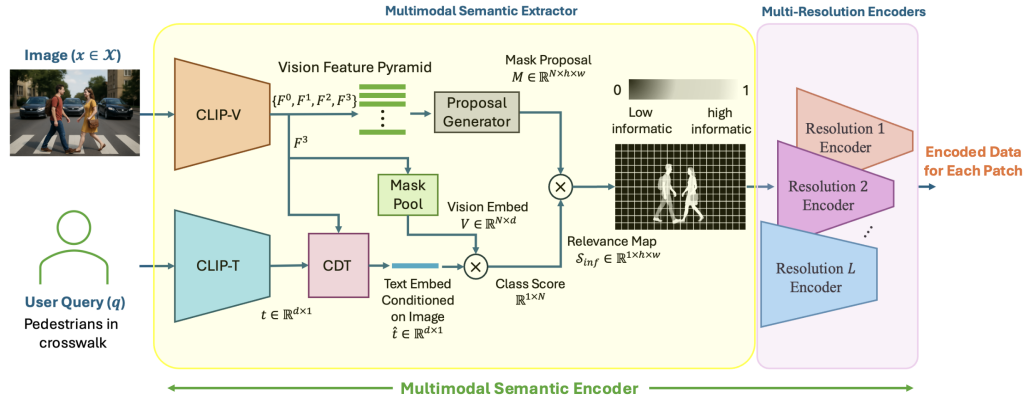


Figure 2.9: Multi-modal semantic encoder framework.

At inference time, unlike training, the framework processes the text input as user command q through the frozen CLIP-T encoder $f_t(\cdot)$ to produce $t \in \mathbb{R}^{d \times 1}$.

The generated text embedding t pass through the same refinement pipeline CDT to generate $\hat{t}$ which is a refined embedding of the text conditioned on the image features. Meanwhile, the image is passed through CLIP-V and MaskFormer to compute the visual features F , mask logits M , and pooled mask embedding V . The pixel-level semantic relevance map is computed as:

$$S_{\text{inf}}(h, w) = \sum_{i=1}^N \langle \hat{t}, v_i \rangle \cdot \sigma(m_i(h, w)) \quad (2.13)$$

This yields a relevance map $S_{\text{inf}} \in \mathbb{R}^{1 \times h \times w}$, which encodes how well each pixel aligns with the user’s textual query.

While our multi-modal semantic extractor is inspired by OSS, there are key differences. OSS typically produces binary segmentation masks indicating whether each object class is present. In contrast, we use a soft relevance score between 0 and 1, representing the confidence that a pixel corresponds to the queried concept. This soft score enables patch-wise variable-resolution encoding, which we detail in the next section. Additionally, in OSS, the text input often consists of a list of single-word object categories. In our case, the user command is a single free-form query, potentially referencing multiple objects. The objective is to generate a mask that captures the degree to which each region of the image matches the intent of the query.

2.3.3.2 Multi-Resolution Encoder/Decoder

Given the relevance map S_{inf} and the available bandwidth budget B , the input image is first partitioned into $P = (h/p)(w/p)$ non-overlapping patches of size $p \times p$. For each patch x_i , we compute a semantic importance score s_i by averaging the per-pixel values within the patch from S_{inf} . Each patch is then assigned to one of L predefined resolution levels, indexed by $\ell_i \in \{1, \dots, L\}$, where each level corresponds to a different encoding bitrate r_{ℓ_i} . Higher semantic scores are mapped to higher resolution levels (i.e., higher r_{ℓ_i}) to preserve critical information, while less relevant patches are compressed more aggressively to save bandwidth.

The patch-to-resolution assignment is performed using a resource allocation algorithm 2, similar to the one described in Section 2.3.2, ensuring that the total bitrate across all patches satisfies the constraint:

$$\sum_{i=1}^P r_{\ell_i} \leq B \quad (2.14)$$

Once the assignments are finalized, each patch x_i is encoded using its designated encoder $\mathcal{E}_{\ell_i}(x_i)$, transmitted, and then decoded at the receiver via the corresponding decoder $\mathcal{D}_{\ell_i}$ to produce the reconstructed patch $\hat{x}_i$. The final reconstructed image $\hat{x} \in \mathbb{R}^{3 \times H \times W}$ is assembled by placing all $\hat{x}_i$ back into their original positions.

This framework enables *task-driven, semantic-aware communication* by strategically allocating communication resources based on the semantic relevance of image content with respect to the user’s text query. As a result, semantically important

regions, those most aligned with the user’s intent, are preserved at higher fidelity, while less relevant regions are encoded more compactly, thus achieving efficient and goal-oriented transmission under bandwidth constraints.

2.4 Experimental Results

We evaluate the performance of our proposed method in terms of reconstruction and accuracy performances. In the case of multi-modal semantic communication, we additionally assess how effectively the reconstructed image reflects the intent of the user query. Specifically, we measure how well the system preserves semantically relevant regions of the image that align with the query, demonstrating the utility of our attention-guided compression framework in capturing user-specific information under bandwidth constraints.

2.4.1 Dataset Specifications

For the training and evaluation of the (multi-resolution) transformer-aided semantic communication framework (Section. 2.3.1 and 2.3.2), we use the TinyImageNet dataset [47], a simplified version of the larger ImageNet dataset [48], to train the encoder-decoder pairs. TinyImageNet consists of 200 classes, with each class containing 500 training images, 50 validation images, and 50 test images. In total, the dataset includes 100,000 training images, 10,000 validation images, and 10,000 test images. All images are originally in RGB format with a resolution of 64×64 pixels. For our application, we scale all these images up to a resolution of 320×480

pixels. The transformer encoder we use has been pre-trained on the full ImageNet dataset [48], providing a strong initialization for fine-tuning on any goal defined later such as image classification or anomaly detection. For evaluation of the multi-modal semantic communication framework (Section 2.3.3), we use the COCO validation dataset [49], a widely used benchmark in computer vision that contains 5,000 images of complex everyday scenes, annotated for object detection, segmentation, and captioning tasks. From this dataset, we select a subset of 200 images for evaluation and adapt the dataset setup to include user queries. For fair comparison with our previous frameworks, all images are resized to a resolution of 320×480 pixels. For each image, we generate a query in one of two ways: (i) by selecting a single object label from the image’s annotated categories, or (ii) by randomly selecting two annotated objects and forming a query in the format “object 1 and object 2.” In both cases, the query is appended to the image sample to simulate a user command. Additionally, the ground-truth segmentation masks corresponding to the queried objects are included to provide reference for evaluating semantic alignment between the query and the reconstructed image.

2.4.2 Model and Hyperparameters

We utilize the DINO model [50] as the transformer encoder in our proposed (multi-resolution) transformer-aided semantic communication frameworks (Sections 2.3.1 and 2.3.2). DINO is a self-supervised vision transformer model pretrained on the ImageNet dataset and available in three variants: *Tiny-ViT*, *Small-ViT*, and *ViT*.

In our framework, we adopt the *Small-ViT* variant with a patch size of $p = 8$, an embedding dimension of $D = 384$, and $K = 12$ transformer blocks, each containing $H = 6$ attention heads.

For the framework presented in Section 2.3.1, we design a deep neural network (DNN) decoder composed of a sequence of convolutional layers, whose architecture is depicted in Fig. 2.10(a). We train the decoder using the Adam optimizer with a learning rate of $\eta = 5 \times 10^{-4}$ and a batch size of 32. To calculate the accuracy result, we employ a pretrained classifier trained on the TinyImageNet dataset.

For the multi-resolution framework in Sections 2.3.2 and 2.3.3, we design encoder-decoder pairs for each resolution level. The detailed architecture of these models is shown in Fig. 2.10(b). Both encoder and decoder are trained independently using the Adam optimizer with a learning rate of $\eta = 2 \times 10^{-5}$ and a batch size of 32.

In Section 2.3.3, we incorporate MAFT+ [43] as the multi-modal semantic extractor. For the CLIP model within MAFT+, we use the ConvNeXt-Large variant from OpenCLIP [42]. The embedding dimension is set to $d = 1024$. The proposal generator component follows the default configuration of Mask2Former [44], with the number of class-agnostic mask proposals set to $N = 100$. For evaluating semantic relevance, we compute text-image similarity scores using the CLIP ViT-Large-Patch14-336 model for both vision and text encoders.

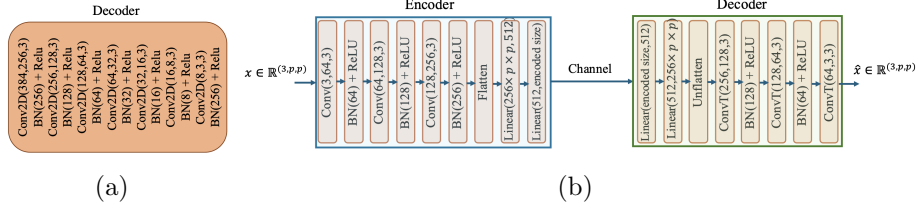


Figure 2.10: (a) Decoder model structure for ViT-based semantic communication framework (Section 2.3.1). (b) Encoder and decoder structure for different resolutions in multi-resolution transformer-aided semantic communication framework (Section 2.3.2 and 2.3.3).

2.4.3 Results and Analysis

2.4.3.1 Transformer-Aided Semantic Communication

We evaluate the performance of our proposed method in terms of reconstruction and accuracy performance. We conduct the training at 4 different rates: $r = 1, 0.75, 0.5, 0.25$. For rate $r = 1$, we send the entire encoded data to the decoder side. Therefore, it can be considered as a traditional semantic communication model which does not use the attention module to compress the encoded data. For rate $r = 0.5$, we use the same framework but we only send 50% of the entire patches based on our derived attention mask. To fairly compare the reconstruction performance on all the rates, we calculate the MSE between the original and reconstructed images on the selected patches which are flagged by a 1 in the attention mask (Eq. 2.4). We show the reconstruction performance in Fig. 2.11(a). The reconstruction performance indicates that the compressed case can be reconstructed as good as (or slightly better) than the uncompressed version on the area requested from the attention mask to be reconstructed.

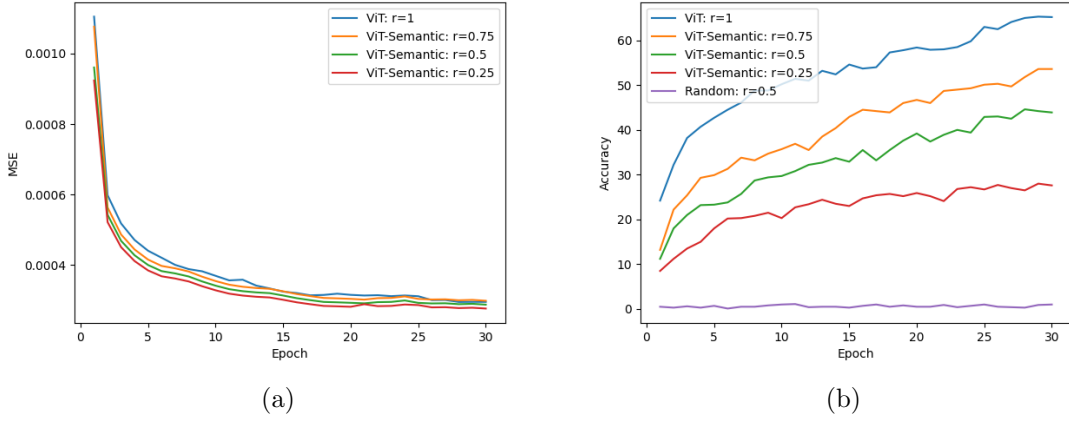


Figure 2.11: (a) Masked MSE performance between the original and reconstructed images under varying channel rates. (b) Classification accuracy of the reconstructed image when the most informative patches are selected until the channel bandwidth is fully utilized ($\alpha = 1$).

Fig. 2.12 demonstrates the original and reconstructed images for some instances of the test data. We observe that the transformer encoder is able to effectively identify and extract the most informative parts of the data, including the essential information needed for classification purposes.

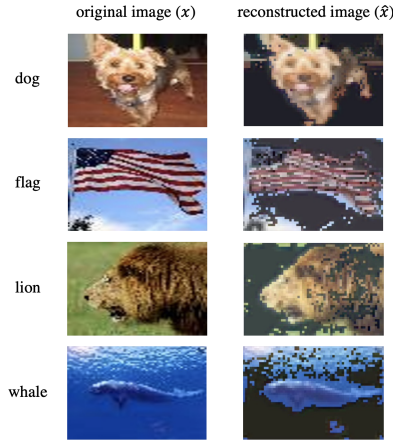


Figure 2.12: Example of reconstructed images with compression of $r = 0.5$.

To evaluate the accuracy of the reconstructed images within our framework, we employ a classifier that was trained on the original TinyImageNet dataset. We

examine the accuracy across all four compression rates by using a compressor with $\alpha = 1$, as shown in Fig. 2.11(b). The findings confirm that our framework is capable of maintaining the semantic integrity of the images, even when reconstructing only a small portion of the patches. This effectiveness is attributed to the capability of the model to preserve the most critical parts of the image because of the attention mask.

Next, we examine the effect of the parameter α used by the compressor in selecting the patches. Fig. 2.13(a) illustrates the effect of using $\alpha = 0.85$ in comparison to the result shown in Fig. 2.11(b). We see that by reducing α by 15%, the system allows 15% of the transmitted patches to be randomly selected from those with lower attention scores. This adjustment leads to improved classification accuracy, particularly at lower compression rates, highlighting the benefit of introducing diversity in patch selection. For example, the accuracy is improved from 24% to 43% over 30 epochs for the compression rate $r = 0.25$. This can be explained by noting that we use the attention score merely as representative of the information in a patch with respect to the semantic content, and even though we believe there is a strong correlation between these two, the true relationship has not been established. Even if the attention score and the semantic information content are almost linearly connected, we are only allowed to remove a patch completely if its semantic information content is zero, otherwise, it cannot be completely ignored and the compressor should still allocate a partial (i.e., non-zero) bitrate to such patches. Since we use a simple binary mask, we seek the remedy by selecting a random number of chunks with lower attention scores (i.e., below the threshold λ).

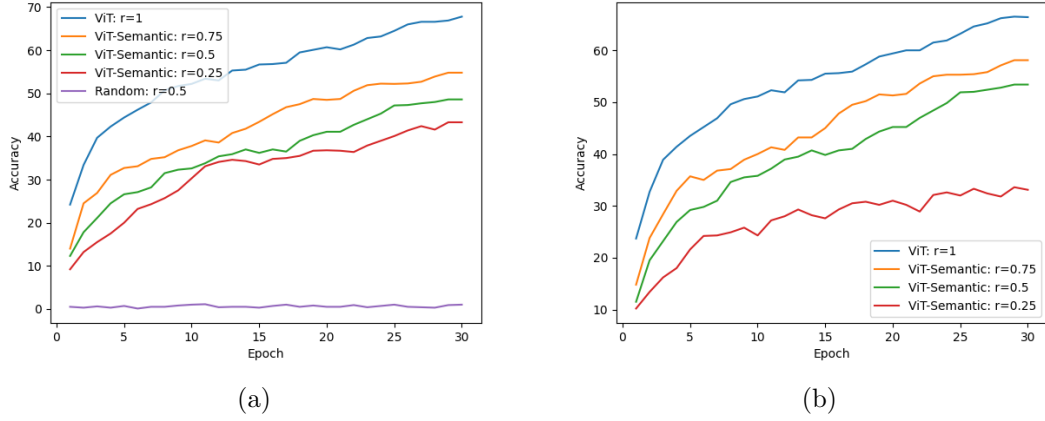


Figure 2.13: (a) Classification accuracy of the reconstructed image in different channel rates when we allow 15% of patches to be selected randomly from low attention score patches along with the most informative patches ($\alpha = 0.85$). (b) Classification accuracy of the reconstructed image in different channel rates using a fine-tuned classifier.

Finally, we note that the classifier which is designed to work on the original images may not be optimized for the classification of the reconstructed images based on the binary mask as we have proposed in this work since as part of the image is going to be missing. The missing (or blackened) background in the reconstructed images can act as an adversarial input for the classifier since the classifier is only seen the original images with backgrounds. Therefore, in order to improve the classification accuracy, we propose to fine-tune the classifier with the following loss function

$$\mathcal{L}_{classifier} = \beta \mathcal{L}_{CE}(y(X), c) + (1 - \beta) \mathcal{L}_{CE}(y(\hat{X}), c) \quad (2.15)$$

where $y(\cdot)$ denotes the pre-trained classifier that we use for TinyImageNet dataset. The loss incorporates the cross entropy for the reconstructed images based on the

compressed data (defined as *compressed images*) in addition to the cross-entropy of the original images (i.e., *uncompressed images*) into the loss function. Fig. 2.13(b) shows the classification accuracy performance for different compression rates using the fined-tuned classifier using the above loss function with $\beta = 0.3$. As shown in Fig. 2.13(b), there is a 10% improvement in accuracy for $r = 0.5$, and approximately, a 6% improvement for both $r = 0.25$ and $r = 0.75$.

2.4.3.2 Multi-Resolution Semantic Communication

The proposed multi-resolution framework consists of five resolutions with encoding sizes of $L = 0, 12, 24, 48, 198$ bytes. The first resolution discards the patch encoding prior to transmission to the receiver, while the last resolution transmits the entire image ($3 \times 8 \times 8$) bytes without further encoding. The three intermediate resolutions involve an encoding and decoding process. Each patch is passed through an encoder to map it to the predefined size of the selected resolution, and the decoder then reconstructs the patch back to its original image size.

The encoder-decoder pairs for each of the three intermediate resolutions are trained on the entire TinyImageNet dataset. Fig. 2.14(a) and Fig. 2.14(b) display the reconstruction quality on the training data and the accuracy on the test dataset, respectively. The reconstruction results indicate that the encoder-decoder pairs are well-trained, and the accuracy results illustrate the performance of the encoder-decoder system across different compression rates. For instance, with a compression rate $q = 0.5$, half of the patches are selected based on the attention

scores, then transmitted to the receiver for decoding and image reconstruction. All selected patches are encoded with a single resolution. As expected, the accuracy results demonstrate that higher encoded sizes (better resolution) significantly improve accuracy across varying compression levels.

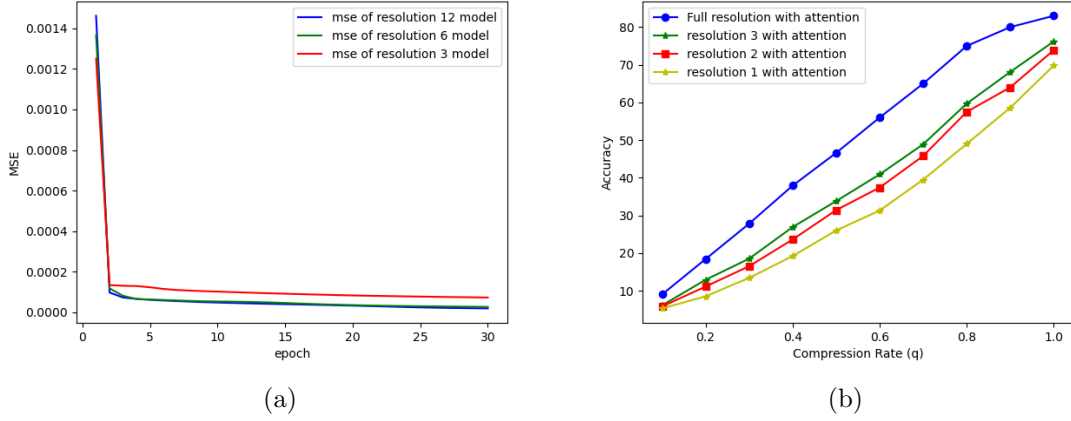


Figure 2.14: (a) Reconstruction result for three medium resolutions. (b) Accuracy result for three medium resolutions.

Next, we evaluate the accuracy performance when using multiple resolutions (using Algorithm 2) based on the available channel rate. We compare this multi-resolution approach to the case where a single fixed resolution is used for transmission. The results show that employing a multi-resolution framework achieves higher accuracy at higher compression rates (i.e., lower available channel rates). As expected, transmitting the most valuable patches at higher resolutions, while sending lower-importance patches at lower resolutions, helps the pretrained classifier achieve higher accuracy even at lower rates. At higher rates, most patches are transmitted at higher resolutions, ultimately matching the performance of full-resolution transmission. The accuracy performance for the multi-resolution framework is shown in

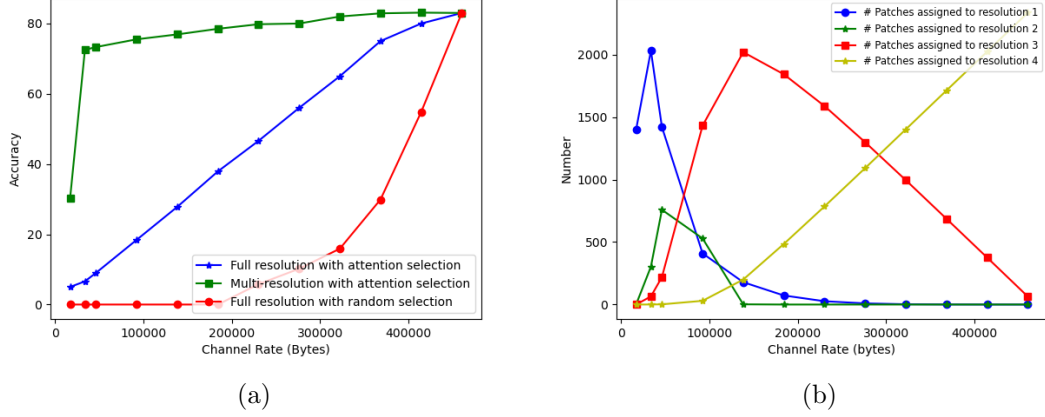


Figure 2.15: (a) Accuracy result for adaptive multi-resolution semantic communication framework in various channel rates. (b) Resolution assignment to patches in different channel rate.

Fig. 2.15(a). Fig. 2.15(b) also illustrates the distribution of image patches transmitted at different resolutions across various channel rates. At lower rates, the resolution selector primarily assigns patches to lower resolution embedding sizes to meet the channel constraints while preserving the semantic content. In contrast, at higher rates, most patches are assigned to higher resolutions to fully utilize the channel capacity.

2.4.3.3 Multi-Modal Semantic Communication

In this framework, we adopt the same five-resolution scheme as before, with encoding sizes of $L = 0, 12, 24, 48, 198$ bytes. The lowest resolution ($L = 0$) skips transmission of the patch, while the highest resolution ($L = 192$) transmits the raw image patch without compression—corresponding to the full $3 \times 8 \times 8$ bytes. The intermediate resolutions use a learned encoder to compress each patch to the desired size, and a

decoder to reconstruct it back to the original dimensions. To standardize our setup, we resize all images to dimensions 320×480 . Given a patch size of 8×8 , this results in exactly 2,400 patches per image. Therefore, the total channel bandwidth for transmitting an entire image ranges from 0 (no patches sent) up to 2400×192 bytes, which corresponds to sending all patches at the highest resolution without any compression. This range defines the limits of our system under varying bandwidth constraints.

Our framework aims to reconstruct the most informative regions of the image based on a user-provided text query. These regions are identified by a vision-language model and prioritized during encoding to receive higher resolution when bandwidth is limited. Fig. 2.16 shows a qualitative comparison between our two semantic communication frameworks: multi-modal semantic communication (MMSC - Section 2.3.3) and multi-resolution transformer-aided semantic communication (ViT-SC - Section 2.3.2). The input image (left) is accompanied by a user query: “Cat and Keyboard”. The ground truth mask highlights the regions corresponding to the queried objects. In the top block, the MMSC framework takes both the image and the user query as input. It generates an informative score map (S_{inf}) that identifies semantically relevant regions aligned with the query and reconstructs the image accordingly under a 50% channel rate. The bottom block shows the output of the ViT-SC framework, which only receives the image as input. It produces an attention score map based on internal attention weights and reconstructs the image under the same bandwidth constraint. As shown, the MMSC framework successfully identifies and reconstructs both queried objects, cat and keyboard, demonstrating

strong alignment with the user’s intent. In contrast, the ViT-SC framework primarily focuses on the most salient object (the cat) and fails to consistently highlight or reconstruct the keyboard. This example underscores the advantage of incorporating multi-modal semantic guidance for query-aware communication and adaptive compression. To further evaluate the reconstruction quality and relevance to the user query, we apply three different approaches over 200 image-text pairs we extracted from COCO validation set.

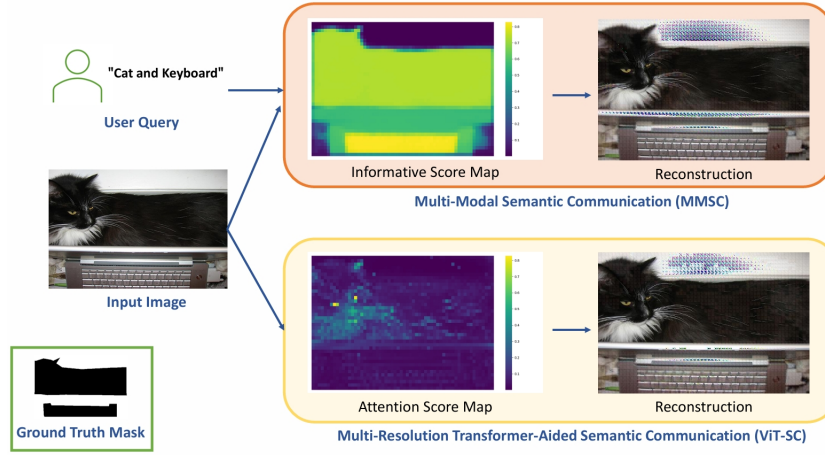


Figure 2.16: An example image from the dataset, transmitted and reconstructed using the MMSC (Section 2.3.3) and ViT-SC (Section 2.3.2) models under a 50% channel rate constraint.

In the first approach, we compute the MSE between the original and reconstructed images, but only within the masked area defined by the ground truth segmentation corresponding to the user query. This region is expected to be reconstructed with the highest fidelity, as it directly relates to the user’s intent. We compare our proposed MMSC framework with the ViT-based framework described in Section 2.3.2, which does not incorporate the user query during encoding. For fairness, both frameworks share the same encoder-decoder and resolution control

components. Fig 2.17 shows that our framework consistently achieves lower masked MSE across all bandwidth levels, indicating better preservation of the semantically important areas.

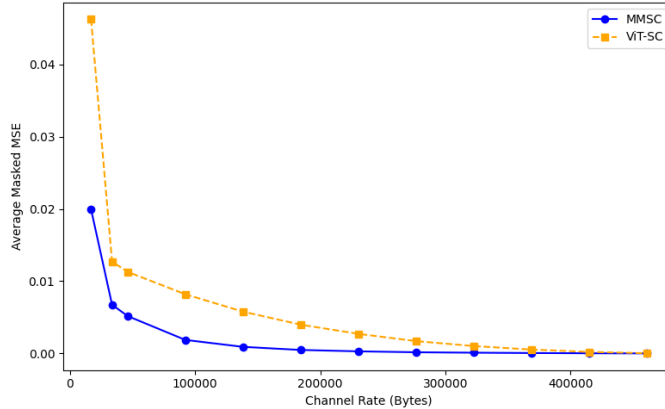


Figure 2.17: Average MSE between the original and reconstructed images within the masked area defined by the ground truth segmentation corresponding to the user query.

In the second approach, we analyze the semantic consistency using the informativeness score S_{inf} , as discussed in Section 2.3.3. We compute S_{inf} between the original image and text query, and $\hat{S}_{\text{inf}}$ between the reconstructed image and the same text. The L1-distance between these scores reflects how well the informative content is preserved. We calculate this distance for both MMSC and ViT-SC frameworks. As illustrated in Fig. 2.18, our MMSC framework demonstrates smaller differences between S_{inf} and $\hat{S}_{\text{inf}}$, outperforming the ViT-based framework across the full range of available bandwidth.

In the third approach, we evaluate the relevance between the reconstructed image and the input text using the CLIP model. The image is encoded using CLIP-V and the text with CLIP-T, and the dot product of their embeddings provides a relevance score, typically ranging from 0.1 to 0.28. A higher score implies

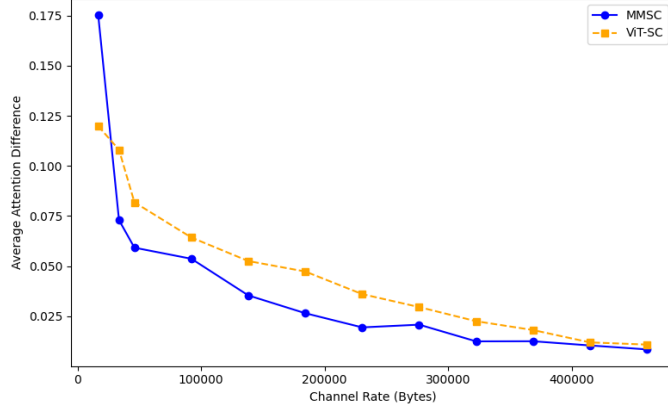


Figure 2.18: L1 distance between informative score maps of the original and reconstructed images conditioned on the user query.

stronger alignment between the visual and textual inputs. We calculate the CLIP score for both our MMSC and ViT-SC reconstructions. Fig. 2.19 indicates that our framework consistently yields higher CLIP scores, particularly at intermediate bandwidths. This is because, under bandwidth constraints, the model is forced to focus on transmitting only the most relevant patches at higher resolutions. At higher bandwidths, however, additional irrelevant regions may also be transmitted and reconstructed, which can dilute semantic focus and slightly reduce the CLIP score.

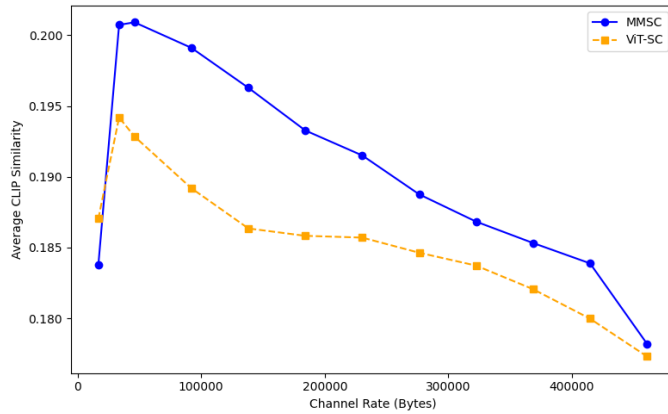


Figure 2.19: CLIP relevancy score between reconstructed image and text.

2.5 Conclusion

We introduced an alternative view of semantic communication, where input data is encoded such that its reconstruction at the receiver preserves higher fidelity in regions that are more relevant to the content and the underlying task. The proposed goal-oriented framework adapts to both bandwidth constraints and task-specific objectives by leveraging transformer-based attention mechanisms. Beginning with a vision-only model, we demonstrated how attention scores from a vision transformer can be used to identify and transmit the most informative image patches, adapting to the available channel rate. We then extended this approach with a novel multi-resolution encoding strategy, which assigns variable bitrates to patches based on their semantic importance and current channel conditions. Finally, we proposed a multi-modal extension that incorporates user queries via cross-attention, enabling the system to emphasize image regions aligned with both the visual content and the user’s intent. Our results show that this multi-modal approach significantly enhances the ability of the semantic communication framework to capture user-intended regions, particularly in complex scenes where vision-only models may overlook relevant details.

Our experiments on TinyImageNet and COCO validate the effectiveness of each component, showing consistent gains in reconstruction quality and classification accuracy under bandwidth constraints. The results highlight the benefits of combining attention-driven patch selection, adaptive bitrate allocation, and user-guided semantic focus. Together, these components form a flexible and extensible

framework for goal-oriented communication in real-world scenarios.

CHAPTER 3

Semantic Multi-Resolution Communication

3.1 Introduction

In traditional communication systems, separate source and channel coding (SSCC) is utilized to express the data as efficiently as possible and to mitigate the effects of noise and interference on the communication channel, respectively. An efficient source encoder generates minimal encoded representation of the source data for a given distortion metric. On the other hand, reliable channel encoding can be achieved by expanding the data via adding redundancies, ensuring that the transmitted data can be decoded by the receiver with negligible error. While Shannon's source-channel separation theorem [51] exists for a single link, SSCC is not optimal for finite block lengths and/or for multi-user scenarios [52]. Indeed, many recent results show the improvement of JSCC over SSCC in such scenarios [3, 53, 54].

Sub-optimality of SSCC in multi-user scenarios, e.g., multi-casting, is further imposed by a limited channel coding rate to support the user with the worst channel. Hence, the users with the better channel will encounter the same reconstruction performance (RP) as the user with the weakest channel. In JSCC, sacrificing a small

portion of RP for weaker users can result in larger gains in RP for better users. This trade-off allows for better overall reconstruction performance across all users. Recently deep learning techniques such as auto-encoder (AE) [55] is introduced to perform JSCC efficiently in an end-to-end communication system [56, 57]. Considering the finite length coding, the drop in weak user RP in comparison to SSCC is shown to be negligible to none in practice.

The idea of multi-resolution (MR) communications is motivated by scenarios such as multimedia-communication where different receivers desire or require different RPs (*defined as tiers*) or when a receiver wishes to improve its RP successively. For example, consider a multi-casting scenario where users are registered to different tiers of RPs. Without MR communications, (a) the transmitter can individually encode the media for each tier and send it using the supported channel rate of the weakest user in that tier, or (b) encode the media with the highest resolution at the best tier and send it with the supported rate of the overall weakest user, in order to satisfy all users' RP requirements and channel rates. In contrast, the MR encoder is designed to provide encoded data for the first (worst) tier and successive (or differential) encoded data for each improved tier. In other words, instead of re-encoding the media for each tier separately, the MR encoder provides differential (or successive) data with considerably smaller sizes to improve the RP of the next better tier. Hence, by using MR encoding, (c) the transmitter is only required to send the successive data for each improved tier at the supported rate of the weakest user in that tier.

A simple calculation may better put the use case of the concept of MR commu-

nication in multi-casting in perspective. Consider 4 tiers of RPs where the encoded block size of the tiers are $B, 2B, 4B$, and $6B$ symbols and the supported rate (for error-free communication) of the lowest user in each tier is $1/3, 1/2, 2/3$, and $3/4$, respectively. Assuming that the MR encoder for each successive layer generates differential output of size ηB , $\eta \geq 1$, the transmission scheme (a), (b), and (c) in the above require transmission of $21B, 18B$, and $(3 + 7.66\eta)B$ symbols, respectively. In practice η is very close to one, say, $\eta = 1.05$ for which $(3 + 7.66\eta)B = 11.04B$.

The works [58, 59] investigate end-to-end multi-resolution communication. In [58], the authors introduce the multi-resolution convolutional AE (MrCAE) approach, which initially processes down-sampled data to capture large-scale, low-dimensional structures. Subsequently, both the data and the neural network undergo iterative refinement using transfer learning at each stage. However, a challenge still exists in transmitting high-resolution data, as the encoder needs to send the encoded data from the beginning. In [59], the authors propose a solution by dividing the encoded data into layers, and each layer’s decoder is designed to utilize the encoded data from previous layers. This technique allows for efficient transmission of encoded data until a certain level of reconstruction quality is achieved, addressing the communication concern more effectively. However, simply dividing the encoded data may not be the best approach in generating multi-resolution representation of the input data.

In this work, we introduce semantic multi-resolution communication (SMRC), where the goal is not only to provide successive refinement of the RP but also to preserve hierarchical semantic information with finer accuracy through successive

encoded outputs. In semantic communication, the goal is to preserve an intended meaning (or semantic) in the encoded data which is otherwise compressed to satisfy a certain RP or distortion value [10, 60]. In SMRC, the encoded output block from the encoder is composed of sub-blocks where each subsequent sub-block provides further information to the decoder in order to (i) improve RP, or (ii) improve the accuracy of a semantic feature or group of semantic features in terms of recall/precision, or both (i) and (ii). In other words, the decoder input consists of a sequential subset of sub-blocks starting from the first sub-block in order to achieve a particular RP and semantic accuracy. Exemplary scenarios discussed in Section 3.2 provide more details and better understanding of the possible use-cases of SMRC.

The main contributions of our work are as follows: (i) We introduce a semantic multi-resolution communication (SMRC) framework, which enables end-to-end communication with various resolutions while preserving semantic features with different accuracy levels (see Section 3.2). (ii) Inspired by vast literature on multi-task learning, we propose a multi-head AE structure which proves to be effective in realizing SMRC (see Section 3.3). (iii) We show the improvement and efficiency of the SMRC framework as well as the effectiveness of the proposed multi-head AE structure through numerical evaluations. We conduct experiments using the MNIST and CIFAR10 datasets to train the models and evaluate the performance of the proposed scheme (see Section 3.4).

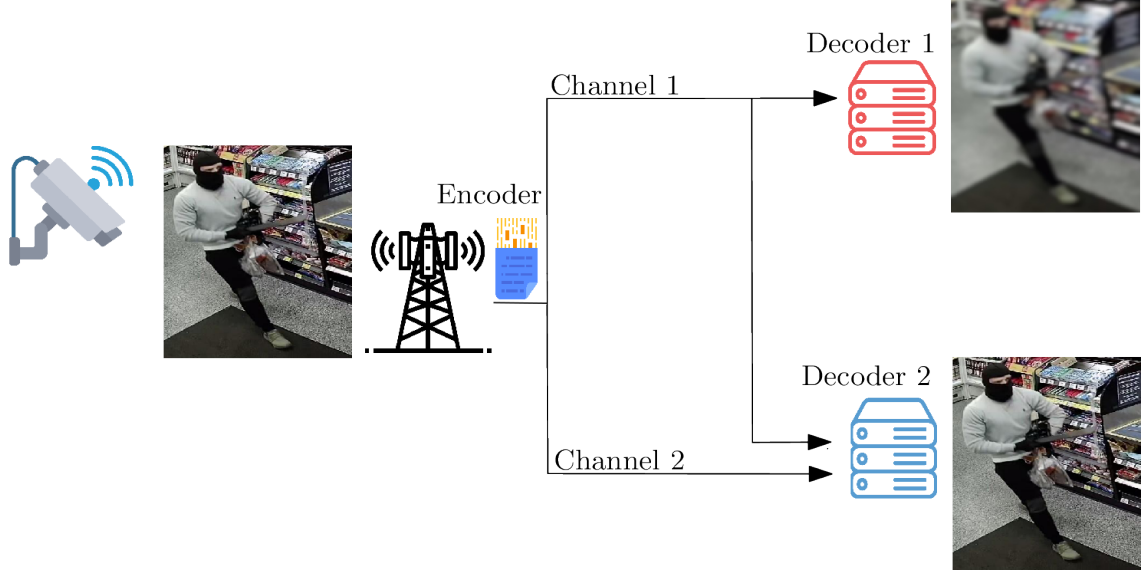


Figure 3.1: Application of SMRC on Example 1.

3.2 Multi-Head AE for SMRC

In the following, we first consider two exemplary scenarios for SMRC. Then, we introduce the general SMRC structure which is built based on the concept of multi-task learning.

Example 1: Consider a scenario where the goal is bandwidth-efficient transmission of the CCTV video of a shop to the server, as depicted in Fig. 3.1. The received signal is not only used by the server to reconstruct the video but also to accurately detect instances of particular emergency situations such as burglary, break-ins, or fire. By employing semantic communication, one can effectively minimize the occurrence of semantic errors along with the reconstruction loss, thereby ensuring the successful preservation of the meaning conveyed by the retrieved information. [10, 60]

In normal conditions, a lower-resolution video is sufficient as long as the se-

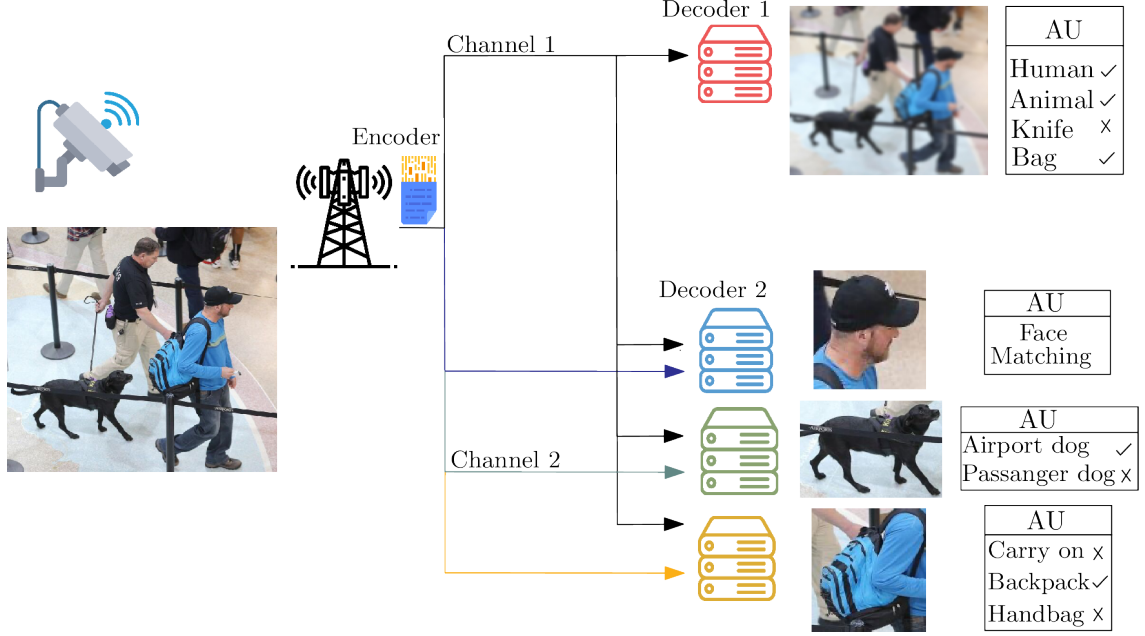


Figure 3.2: Application of SMRC on Example 2.

mantic information, e.g., a possible emergency event, is preserved. However, when the receiver detects an emergency situation such as burglary, one may need to (i) acquire the higher resolution of the event in order to identify specific details, e.g., a human face or (ii) confirm certain semantic information with higher confidence, e.g., with certain recall (probability of detection) and precision (false alarm probability). We note that increasing recall is beneficial to minimize the loss associated with the theft, while precision is important when an incorrect detection incur considerable costs, e.g., when calling 9-1-1 police patrol or other emergency units such as fire-fighters. In such cases, instead of re-sending the entire data from the beginning, our proposed SMRC framework allows for the transmission of successive layers, specifically designed to facilitate the efficient exchange of content with varying resolutions and semantic information with a desired confidence level.

Example 2: Fig. 3.2 illustrates an alternative scenario where a 5G wireless

CCTV camera at the airport terminal gate captures videos of the cars, humans, animals, luggage, etc. The video clips are encoded with a lower resolutions to preserve the bandwidth, nonetheless, encoding is performed to preserve certain semantic information required for running analytic units such as human-detection, animal-detection, and object-detection. Based on a situation, either from time-to-time and at random, or to detect suspicious activities the server-side analytic units may ask for higher resolution video clips, and/or better accuracy of some semantic features, e.g., to run and obtain an improved analytical result, e.g., human-segmentation, object-segmentation, or human-face-recognition. Moreover, for some semantic features, it might be desirable to receive higher resolution video clips, e.g., human-face-detection. The concept of SMRC is helpful in providing incremental information for improving the semantic accuracy and resolution of the reconstruction.

SMRC is composed of an encoder and a decoder. The encoder takes an input block of symbols and produces an encoded block (output block) that is composed of, say L , sub-blocks indexed sequentially from 1 to L . The decoder takes the sequence of sub-blocks 1 to $l \in [L]$ to generate layer l output that achieves a particular resolution determined by the reconstruction performance (RP) and semantic accuracy (SA). The $(l + 1)$ th sub-block provides differential information to successively improve the resolution of the layer l output in terms of RP and SA. Naturally, the resolution defined for different output levels should follow a nested structure. This means that a well-defined layer $(l + 1)$ output has better or equal resolution than the layer l output. Nonetheless, the desired RP for an input block might be related to the semantic content of the block. This means that the RP for layer $(l + 1)$ can

be defined in terms of the semantic content obtained in layer l output. For example, suppose that the semantic preserved in layer l allows to run object-detection, animal-detection, and human-detection analytics and find the corresponding bounding boxes or region-of-interests (RoIs). Then, the RP for the layer $(l + 1)$ may be defined to assign different RP for each of RoIs based on their types, i.e., object, human, and animal.

We also note that the nested structure allows to define a single accuracy metric at the layer l for a group of classes that belong to a semantic type and then further refine the accuracy in the subsequent output levels. This means that one can define a hierarchical structure for semantic types and classes such that all possible cases for the layer $(l + 1)$ are subsets of the cases defined for layer l . The concept of grouping some classes into a single combined class is called *class pooling*. Hence, the hierarchical structure may be defined by allowing class pooling in layer l based on the classes in layer $(l + 1)$, i.e., having a combined class in layer- l by grouping some classes from layer $(l + 1)$. Besides the application of class pooling in defining hierarchical semantic structure, we will later reveal a powerful property of class pooling in joint optimization of recall and precision.

Inspired by the vast literature on multi-task learning (MTL) [61], we adopt a multi-head encoder for learning the related tasks of finding the encoding for each sub-block. It is known that MTL is a powerful approach in learning multiple related tasks together and discovering a shared representation in the process. The fundamental idea behind MTL is that data used for these tasks can have underlying similarities that can be captured in a common representation, and not only the

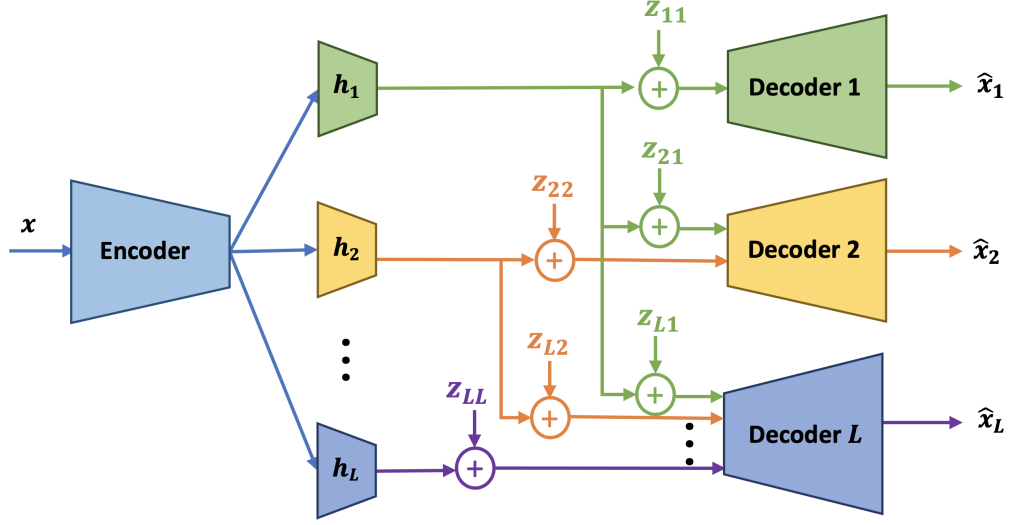


Figure 3.3: Multi-head AE for SMRC.

shared representation but also the individual task performance is improved when they are learned jointly. The MTL is composed of a network’s pre-layer that is shared among all tasks, while task-specific layers are added on top of the pre-layers to perform the function that is specific to each task. To train the model, the overall loss is computed as a weighted sum of individual task losses, where the weights correspond to the importance of each task. This framework allows for efficient and effective learning of multiple tasks simultaneously, leveraging shared information while accommodating task-specific differences.

Fig. 3.3 illustrates the general multi-head structure of our proposed SMRC. By enhancing the existing end-to-end multi-resolution framework introduced by [59], we incorporate a multi-head structure after the encoder stage. The primary objective of SMRC is to efficiently encode data in a hierarchical manner, enabling the decoders to progressively enhance data reconstruction in subsequent levels by leveraging the encoded data from previous layers. Additionally, the multi-head structure in SMRC

allows the transmitter to encode the data in order to preserve semantic features independently for each layer, thereby facilitating effective semantic communication. We note that the head $l \in [L]$ produces the sub-block l in the encoded block.

We normalize the encoder output such that the average power of the transmitted signal is 1. Hence, the SNR of the links is directly controlled by the variance of the noises z_{li} for each link between an encoder head l and decoder i as illustrated in Fig. 3.3. This SNR can also be interpreted as the overall signal-to-noise ratio due to the combined effect of channel gain and additive Gaussian noise at the receiver. This can be used to adapt our SMRC model in a number of different multi-user scenarios and channel fading conditions.

Single AP to multiple users, block fading: Based on this interpretation, Fig. 3.3 illustrates a multi-user scenario in which a single access point (AP) communicates with multiple users where the encoded output for each head is sent as one packet of transmission and the channel between the AP and each user is block fading, i.e., the channel remains constant during transmission of each packet and may change between transmission of packets.

Single AP to multiple users, AWGN channel: If the coherence time of the channel covers the entire transmission of the output from all encoder heads, the channel can be modeled as AWGN channel within such transmission and we have $z_{li} = z_{lj}, \forall i, j \in [L]$. In the multi-user interpretation of the model in Fig. 3.3, the users are served with different resolutions by transmitting varying numbers of packets corresponding to different heads.

Single AP to a single user, block fading: On the other hand, The model

Fig. 3.3 can be used for multi-resolution transmission of data to a single user by setting $z_{ki} = z_{li}, \forall k, l \in [L]$. In this scenario, the encoded data packet generated by each head is successively transmitted to the user which reproduces the data after reception of the first packet and successively improves its reproduction quality by receiving the next packet.

3.3 System Model and Problem Formulation

We consider an end-to-end semantic communication system that incorporates a deep learning model as depicted in Fig. 3.4, consisting of an encoder that produces a semantic multi-resolution encoding of an input data block, denoted as x . The output block generated by the encoder consists of L sub-blocks, indexed as $l = 1, 2, \dots, L$, each containing n_l symbols, where $1 \leq l \leq L$. The encoder exploits a neural network with a multi-head structure. The sub-block l is the output of layer l which is generated by one of the encoder heads, say h_{lk} , that belongs to layer l . This means that there are K_l possible outputs by layer l corresponding to one of its head $h_{lk}, 1 \leq k \leq K_l$ where K_l is the total number of heads in layer l .

In reference to Fig. 3.4, the encoder is responsible for performing joint source and channel encoding of the input data x and the decoder performs both channel and source decoding. The decoder first reconstructs $\hat{x}$ from the received packets transmitted through the communication channel, and then the semantic extractor takes out the semantics $\hat{y}$ from this reconstructed data. The loss function for each

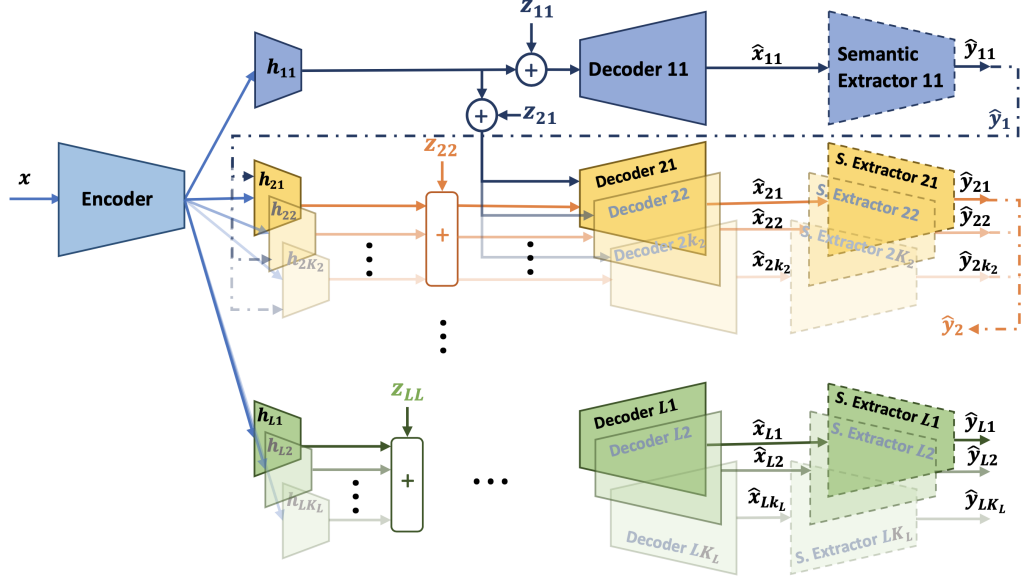


Figure 3.4: System model.

link between an encoder head h_{lk} to the receiver is formulated as

$$\mathcal{L}_{lk} = \alpha_{lk} \overbrace{\|\mathcal{R}_{\hat{Y}_{l-1}}(x - \hat{x}_{lk})\|^2}^{\text{MSE loss}} + (1 - \alpha_{lk}) \underbrace{(\mathbb{1}^T(y_{lk}(x))\beta_{lk} \log(\sigma(\hat{y}_{lk})))}_{\text{semantic loss}} \quad (3.1)$$

where $\mathbb{1}(y_{lk}(x))$ is a one-hot column vector obtained from the class label $y_{lk}(x)$, β_{lk} is a diagonal matrix that models varying degrees of accuracy for different classes, and σ is a softmax operator. $\hat{Y}_{l-1} = [\hat{y}_1, \dots, \hat{y}_{l-1}]$ where $\hat{y}_l = [\hat{y}_{l1}, \dots, \hat{y}_{lK_l}]$, $\forall l \in [L-1]$. $\mathcal{R}_{\hat{Y}_{l-1}}$ is the region of interest (RoI) operator defined as $\mathcal{R}_{\hat{Y}_{l-1}}(x) = M \odot x$ where M is the mask with entries m , $0 \leq m \leq 1$ that is element-wise multiplied by x . The entries of the mask are used to enforce the reconstruction importance of the corresponding element of x . The subscript of the RoI operator $\mathcal{R}_{\hat{Y}_{l-1}}$ denotes the possible dependency of RoI operator to the semantics extracted by previous layers, namely Y_{l-1} . For example, the RoIs in Example 2, correspond to a segment or bounding box which includes a human face, an animal face, etc. which may depend

on the semantic extractor output of the previous layer.

The loss function (3.1) is defined to jointly minimize the reconstruction loss and semantic loss. The parameter $\alpha_{lk} \in [0, 1]$ is used to prioritize reconstruction versus semantic loss. In extreme cases, to maximize the priority of reconstruction loss (semantic loss), we set $\alpha_{lk} = 1$ ($\alpha_{lk} = 0$). For example, when through the reception of the first layer one has identified that a burglary has happened in Example 1, the priority for the second layer might be to increase the resolution of the image in order to identify the face of the burglar in which case we set $\alpha_{21} = 1$ in designing the encoder head h_{21} . Conversely, in the same example, the priority might be to enhance the confidence in the semantic feature of the classifier in order to call the police and stop the loss due to burglary without the need for a better resolution. In this case, one can design a head by setting $\alpha_{22} = 0$.

The parameters β_{lk} are employed to model the significance of the accuracy of the semantic extractor in identifying different classes. Each element of the diagonal matrix β_{lk} is between zero (lowest priority) and one (highest priority). We will see in Section 3.4, that the selection of different values of the diagonal elements of β_{lk} will affect recall and precision in different ways. This is due to the fact that the cross entropy term in semantic loss of (3.1) only tries to maximize the sum of the product of the prior and the posterior probabilities for all classes, which does not correspond to the optimization of either of recall or precision. However, a suitable choice for β will help to effectively balance both recall and precision as different accuracy metrics.

Two notes are in order here. First, the semantics may include the segmentation

or class labels of the input signal. In practice, the class labels have a hierarchical structure, i.e.,

$$\forall c_{lk} \in C_{lk}, \exists c_{l-1,k'} \in C_{l-1,k'} : c_{lk} \subset c_{l-1,k'}$$

where C_{lk} is the set of class labels for the head k of layer l . Second, the length of the vector $\hat{y}_{lk}$, $\mathbb{1}(y_{lk}(x))$, and the order of the square matrix β_{lk} is equal to the size of the set C_{lk} .

Total loss $\mathcal{L}$ is a weighted sum of losses calculated by all layers and all heads inside each layer, namely $\mathcal{L}_{lk}$,

$$\mathcal{L} = \sum_{l=1}^L \sum_{k=1}^{K_l} c_{lk} \mathcal{L}_{lk} \quad (3.2)$$

The sub-block size n_l and weight c_{lk} may be used to tune the performances for different resolutions or accuracy obtained by different encoder heads or sub-blocks. Moreover, the training process may benefit from adaptive adjustment of the weights c_{lk} to enforce faster convergence for some encoder heads prior to optimization of the other heads. For example, this approach may be used to design the differential resolution obtained by an encoder head in layer l conditional on a threshold performance by previous layers.

We use the loss function (3.1) to train the encoder-decoder pair. For faster convergence and ensure low semantic loss from the beginning, the semantic extractors are initially pre-trained with raw data. We use alternative optimization to update the semantic extractors after training the encoder-decoder pair in each epoch. As

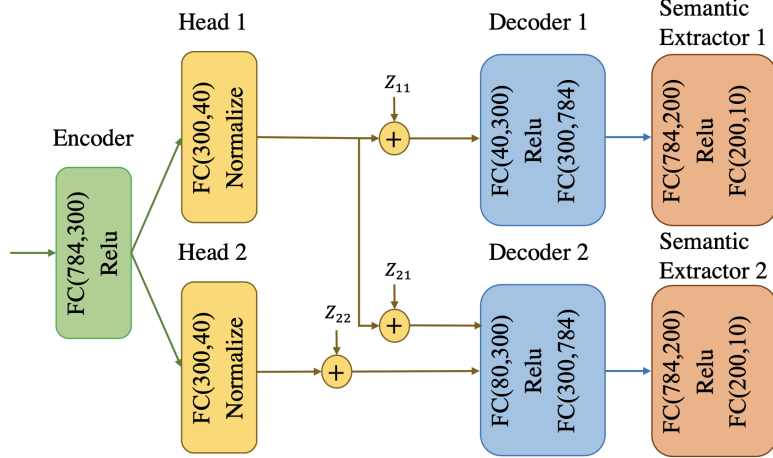


Figure 3.5: Model structure for MNIST dataset.

a result, the semantic extractors will be trained to provide the best accuracy based on the reconstructed data instead of the raw data, hence, incorporating the effect of the designed AE and channel characteristics.

3.4 Experimental Results

3.4.1 Dataset Specifications

We use two datasets in our experiments: I) MNIST dataset that consists of 28×28 gray-scale images of handwritten digits categorized into 10 classes. The training (test) set contains 60,000 (10,000) images. II) The CIFAR-10 dataset that consists of 32×32 RGB images divided into 10 classes with each class representing a specific object category, such as airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks. The training (test) set contains 50,000 (10,000) images.

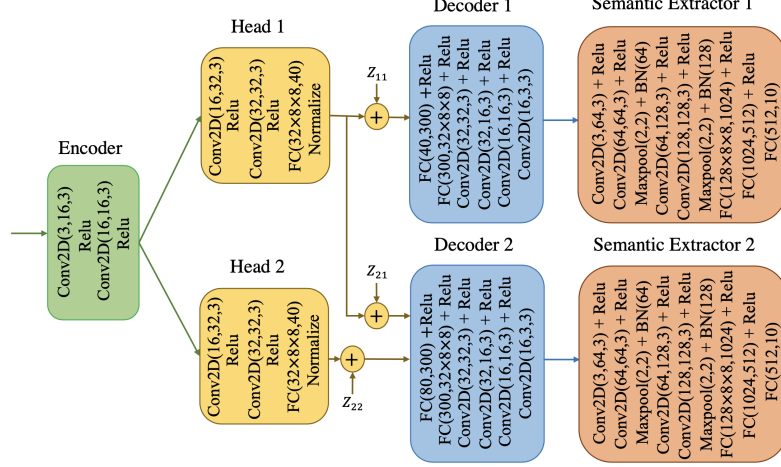


Figure 3.6: Model structure for CIFAR10 dataset.

3.4.2 Model and Hyperparameters

We use Adam optimizer with learning rate $\eta = 1 \times 10^{-3}$ to individually train our proposed SMRC parameters for both datasets. For MNIST dataset, the model for encoder, headers, and decoders can be seen in Fig. 3.5. For CIFAR-10 dataset, the models can be seen in Fig. 3.6. We use equal weights for all $c_{lk}, \forall l, k$. Also, we set $\alpha_{lk} = 0.9, \forall l, k$. In all the simulations, we consider SMRC to a single user, i.e., we set $z_{ki} = z_{li}, \forall k, l \in [L]$.

3.4.3 Reconstruction Performance (RP)

We evaluate the reconstruction performance in terms of peak signal-to-noise ratio (PSNR),

$$\text{PSNR (dB)} = 10 \log_{10} \left(\frac{\max^2}{MSE} \right). \quad (3.3)$$

We employ SMRC as in Fig. 3.5 to encode an image x from MNIST dataset and generate an encoded sub-block of size 40 symbols by each head. The first decoder receives the first sub-block plus noise and reconstructs the image as $\hat{x}_1$, while the second decoder utilizes both sub-blocks to generate $\hat{x}_2$. Fig. 3.7(a) illustrates the PSNR $\hat{x}_1$ and $\hat{x}_2$ versus the SNR of the communication channel (denoted by the training SNR). It can be seen that in comparison to the first decoder, the second decoder improves PSNR by about 2 dB for SNR in the range of 0 to 12 dB. We also plot the RP of single head AE with the same complexity as our two head SMRC encoder that generates an output block of size 80 symbols directly. This will serve as an upper bound on the performance of the second decoder of SMRC with two heads. Fig. 3.7(a) shows that the second decoder of SMRC has negligible loss in comparison to the upper bound that reveals the effectiveness of the successive encoding by SMRC.

In order to assess the performance of our framework in comparison to a SSCC, we employ the BPG (Better Portable Graphics) algorithm as the source coder and LDPC (Low-Density Parity-Check) codes for the channel coding along with MCS tables from 3GPP/LTE standard (Table 7.2.3 in [62]). Fig.3.7(a) shows that using the same number of channel uses, i.e., 80 symbols, our proposed methods consistently outperform SSCC (comprising BPG as a source coding and LDPC as a channel coding scheme) by a significant margin.

We replicate the experiment with CIFAR10 dataset in Fig. 3.7(b). The results again show that our framework improves output quality successively and approaches the upper bound obtained by training a single head AE that encodes the image with

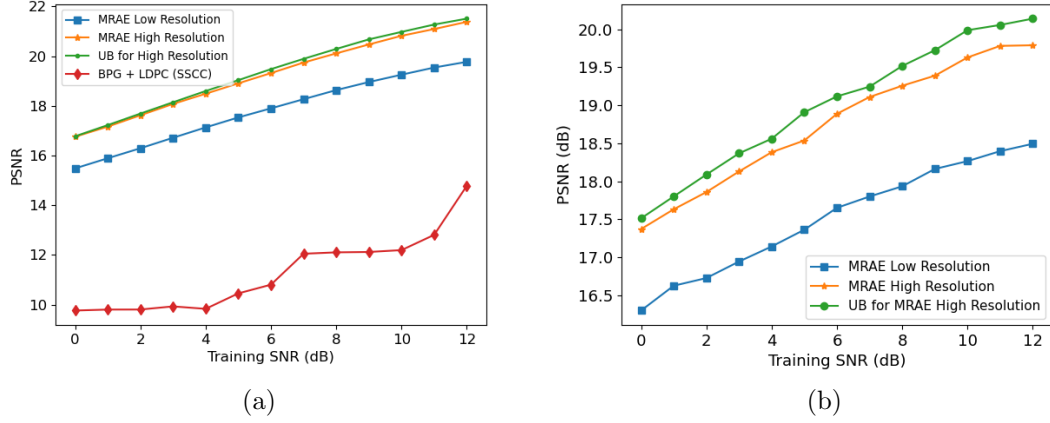


Figure 3.7: PSNR results for (a) MNIST and (b) CIFAR10 dataset.

80 symbols.

3.4.4 Semantic Accuracy (SA)

To demonstrate the efficacy of our framework in preserving intended semantic information, we utilize two key accuracy metrics with respect to a classifier: recall and precision. In the multi-class scenario, we define the accuracy in terms of per-class recall and precision. We define *recall for class i* as $\rho_i \triangleq P(\hat{y} = i | y(x) = i)$, that is the probability of correctly classifying the class of the output $\hat{x}$, i.e., $\hat{y}$, as i given that the class label of the input signal x , i.e., $y(x)$ is i . Also, we define *precision* as $\pi_i \triangleq P(y(x) = i | \hat{y} = i)$ that is the probability of the input label $y(x)$ being equal to i given that the class label of the output, i.e., $\hat{y}$, is equal to i . However, we define *average accuracy* as the average recall for a given input distribution function, i.e., $\sum_i \rho_i P(y(x) = i)$, that is equal to average precision.

We evaluate SA for the same two-layer SMRC used for RP evaluation. Using MNIST dataset, we train the first layer encoder to classify all 10 classes, while the

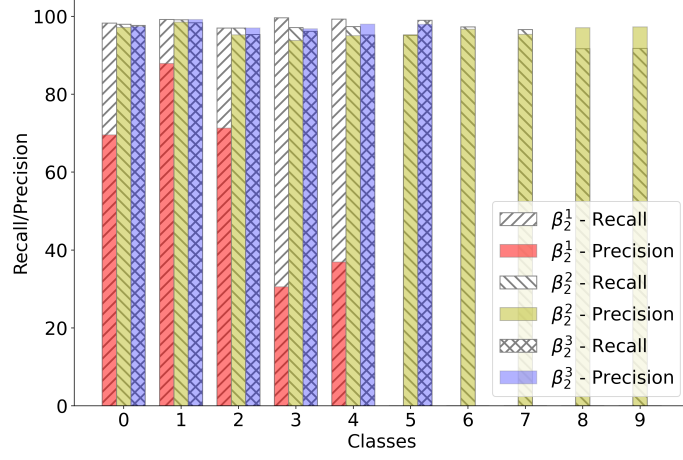


Figure 3.8: Effect of β (weight vector) in precision/recall.

second layer encoder is designed to improve the recall or precision of the first 5 classes by utilizing a weighting vector β in the cross-entropy loss function (3.1). Fig. 3.8 illustrates the recall and precision outcomes obtained for some choices of the weighting vector. In the first case, denoted as β_2^1 , we use weight 1 for each of the first 5 classes and 0 for the rest. In the second weighting choice, β_2^2 , we use equal weight 1 for the first 5 classes and equal weight 1/2 for the second 5 classes. Finally, in the third weighting scheme β_2^3 , we assign weight 1 for the first 5 classes and pool the second 5 classes into one “dummy” class with the weight 1.

These different weighting schemes allow us to assess the impact of varying the importance assigned to different classes in the second layer. Fig. 3.8 depicts the recall (precision) as a measure of accuracy of second layer output for each of the used weighting vector in cross-entropy loss. Using β_2^1 results in the best improvement in recall of successive layer, but it comes at the cost of degrading the precision since the encoder mistakenly learns to map the rest of non-important classes (last 5) into

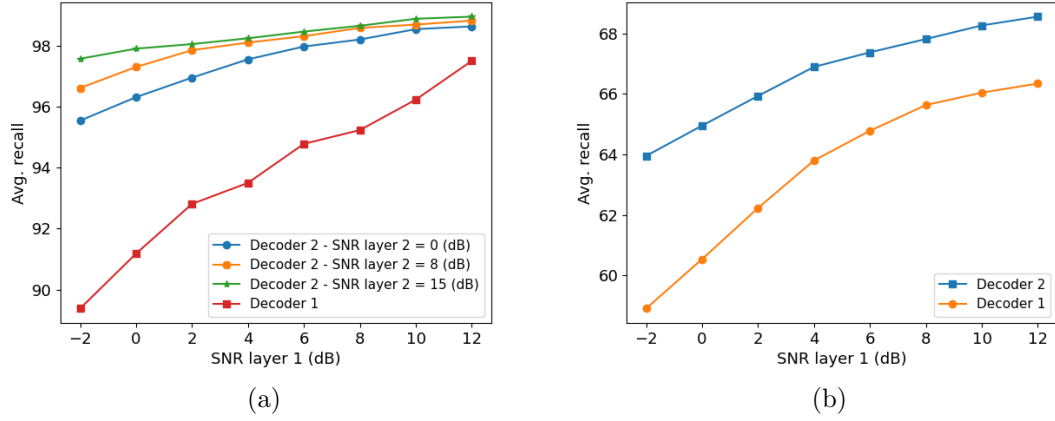


Figure 3.9: Accuracy (avg. recall) of different layer decoders in different SNR for (a) MNIST (b) CIFAR10 dataset.

the important classes (first 5). This can be mitigated by choosing β_2^2 which restores the precision to an acceptable level while still improves the recall performance of the second layer. However, the recall performance with β_2^2 is not as good as that of β_2^1 . Finally, by mapping all unimportant classes into a single class, using β_2^3 provides the best precision performance for the chosen classes at the expense of slightly lower recall than that of β_2^1 or β_2^2 .

The average recall performance of the reconstructed outputs as a function of the first layer channel gain is illustrated in Fig. 3.9(a) for MNIST dataset. We further evaluate the performance of the second layer decoder output across three distinct second layer channel gains: 0 dB, 8 dB, and 15 dB. The results consistently demonstrate that the second decoder output exhibits superior recall performance. We run the same evaluation for CIFAR10 dataset. Fig. 3.9(b) again shows that the successive layer improves the classification performance for CIFAR10 dataset.

3.5 Conclusion

In this work, we introduced semantic multi-resolution communication (SMRC), a novel framework designed to address the limitations of separate source-channel coding (SSCC) and existing multi-resolution communication approaches, particularly in scenarios requiring both progressive reconstruction quality and semantic preservation. Inspired by multi-task learning, our approach employs a deep learning-based multi-head autoencoder architecture that enables hierarchical encoding and decoding of data blocks. Each successive layer enhances both reconstruction performance and semantic accuracy, facilitating efficient adaptation to varying user requirements and channel conditions. SMRC allows for the selective refinement of specific semantic features, making it especially valuable in applications like surveillance, where certain semantic information (e.g., object detection or facial recognition) is prioritized. We proposed a joint loss function that flexibly balances reconstruction fidelity and semantic accuracy, and demonstrated through experiments on MNIST and CIFAR-10 datasets that SMRC achieves significant gains in both PSNR and classification metrics compared to SSCC baselines. The results validate SMRC’s ability to deliver incremental improvements in data quality and semantic correctness, enabling adaptive, bandwidth-efficient, and semantically-aware communication.

CHAPTER 4

Personalized Federated Multi-Task Learning Over Networks

4.1 Introduction

Multi-tasking (MTL) is a powerful technique for learning several related tasks simultaneously [61, 64]. MTL improves the overall system performance, the training speed, and data efficiency, by leveraging the synergy among multiple related tasks. Each task in MTL setting has a single common encoder network to map the raw data into a lower dimensional shared representation space, in addition to a unique client-specific header network to infer task related prediction values from the shared representation. MTL is particularly suitable for distributed learning settings where no single entity has all the data and labels for all of the multiple different tasks.

Federated learning (FL) [19] is a distributed learning paradigm in which many clients train a shared model with the assistance of a central unit called parameter server (PS) by keeping their data private and decentralized. Statistical heterogeneity becomes a major challenge for FL as the size of the distributed setting, namely the number of clients or the number of tasks in a FL setting, increases. This is caused by

This chapter is partially based on joint work with C. Vahapoglu [63].

different task complexities and non-i.i.d. data distribution among clients. Statistical heterogeneity due to non-i.i.d. data distribution degrades the system performance [20, 21]. Further, synergy between the tasks may not always be positive, referred to as a negative transference, again degrading the performance [22].

Personalized federated learning (PFL) is introduced to deal with statistical heterogeneity, where the centralized server and clients learn a shared representation together, while each client trains its own client-specific header network further, referred to as personalization. PFL has the capability of learning user-specific models better while also capturing the distilled common knowledge from other clients [65–67]. As a result, PFL reduces the statistical heterogeneity among clients. Several PFL approaches have been proposed, where different local models are used to fit user-specific data, but also capture the common knowledge distilled from data of other devices for this purpose [14, 65–72]. [73] provides a unified model framework to prove multiple kinds of PFL methods. In this work, we consider MTL in a FL setting, enhanced with personalization, namely, personalized federated multi-task learning (PF-MTL).

PFL works that are closely related to our work are federated representation learning *FedRep* [65] and federated learning with personalization layers *FedPer* [66]. These works use a shared encoder with a unique task-specific header to enhance personalization in a FL setup. *FedRep* and *FedPer* aggregate the common encoder parameters by equal weighting. Our goal in this work is to perform weighted gradient aggregation at the PS dynamically according to gradient updates coming from clients to overcome statistical heterogeneity.

Several approaches involve setting the weights of tasks manually at the beginning of training [74, 75]. Task weights can also be set in an adaptive manner for each iteration. *GradNorm* is a dynamic weighting approach in MTL that normalizes gradient norms by scaling the task loss functions to regulate learning speeds and fairness across tasks [76]. *GradNorm* is proposed for a centralized learning setting. In our work, we introduce *FedGradNorm* framework, where dynamic weighting is incorporated in PFL setting by considering some aspects of [65, 66, 76] together. We provide a theoretical convergence proof for *FedGradNorm*, while *FedPer* [66] and *GradNorm* [76] do not provide a convergence proof, and *FedRep* [65] provides a convergence proof only for the linear learning model setting.

The main contributions of our work can be summarized as:

- We propose the *FedGradNorm* algorithm. The proposed algorithm takes advantage of the *GradNorm* [76] dynamic weighting strategy in a PFL setup for achieving a more effective and fair learning performance when the clients have a diverse set of tasks to perform.
- We provide the convergence analysis for adaptive weighting strategy for MTL in PFL setting. Existing works either do not provide convergence analysis or do it in special cases. We demonstrate that *FedGradNorm* has an exponential convergence rate.
- We conduct several experiments on our framework using Multi-Task Facial Landmark (MTFL) dataset [77], and RadComDynamic dataset on the wireless communication domain [78]. We investigate the changes in task loss during

training to compare the learning speed and fairness of *FedGradNorm* with a similar PFL setting which uses equal weighting technique, namely *FedRep*. Experimental results exhibit a better and faster learning performance for *FedGradNorm* than *FedRep*.

4.2 System Model and Problem Formulation

4.2.1 Federated Learning (FL)

FL [19] is a distributed machine learning approach that enables training on decentralized data in devices such as smart phones, IoT devices, and so on. FL can be described as an approach that brings the training model to the data, instead of bringing data to the training model [79]. In FL, edge devices collaboratively learn a shared model under the orchestration of a PS without sharing their training data.

The generic form of FL with N clients is

$$\min_{\omega} \left\{ F(\omega) \triangleq \frac{1}{N} \sum_{i=1}^N p^{(i)} F^{(i)}(\omega) \right\} \quad (4.1)$$

where $p^{(i)}$ is the loss weight for client i such that $\sum_{i=1}^N p^{(i)} = N$, and $F^{(i)}$ is the local loss function for client i .

4.2.2 Personalized Federated Multi-Task Learning (PF-MTL)

We consider a PFL setting with N clients, in which client i has its own local dataset $D_i = \{(\mathbf{x}_j^{(i)}, y_j^{(i)})\}_{j=1}^{n_i}$ where n_i is the size of the local dataset, and T_i is the task

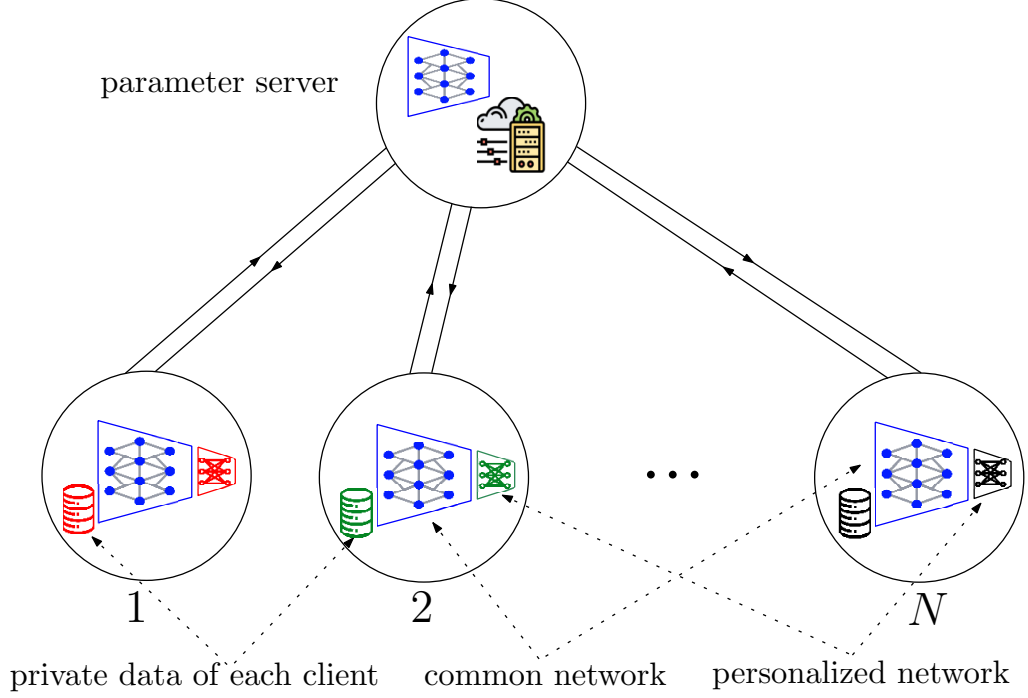


Figure 4.1: Personalized federated learning (PFL) framework with a common network (shown in blue) and small personalized headers (shown in red, green, black).

of client i , $i \in [N]$. The system model consists of a global representation network $q_\omega : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$ which is a function parameterized by $\omega \in \mathcal{W}$ and maps data points to a lower space of size d' . All clients share the same global representation network which is synchronized across clients with global aggregation. Client-specific heads $q_{h^{(i)}} : \mathbb{R}^{d'} \rightarrow \mathcal{Y}$ are functions parameterized by $h^{(i)} \in \mathcal{H}$ for all clients $i \in [N]$ and map from the low dimensional representation space to the label space $\mathcal{Y}$. The system model is shown in Fig. 4.1. Then, the local model for the i th client is the composition of the i th client's global representation model q_ω and the personalized model $q_{h^{(i)}}$,

$$q_i(\cdot) = (q_{h^{(i)}} \circ q_\omega)(\cdot) \quad (4.2)$$

The local loss for the i th client is represented as

$$F^{(i)}(h^{(i)}, \omega) = F^{(i)}(q_i(\cdot)) = F^{(i)}((q_{h^{(i)}} \circ q_\omega)(\cdot)) \quad (4.3)$$

Through alternating minimization, the clients and the centralized server aim to learn a set of global parameters ω together, while each client i learns its own set of client-specific parameters $h^{(i)}$ locally. Specifically, client i performs τ_h local gradient based updates to optimize $h^{(i)}$, $i \in [N]$, while the global network parameters at client i , i.e., $\omega^{(i)}$, are frozen. Thereafter, client i performs τ_ω local updates to optimize the global shared network parameters, while the parameters corresponding to the client-specific head are frozen. Then, the global shared network parameters $\{\omega^{(i)}\}_{i=1}^N$ are aggregated at the PS to obtain a common ω . Thus, the problem is

$$\min_{\omega \in \mathcal{W}} \frac{1}{N} \sum_{i=1}^N p^{(i)} \min_{h^{(i)} \in \mathcal{H}} F^{(i)}(h^{(i)}, \omega) \quad (4.4)$$

FedRep [65] investigates this framework with $p^{(i)} = 1$, $i \in [N]$.

4.2.3 PF-MTL as Bilevel Optimization Problem

The optimization problem in (4.4) can be rewritten as (4.5) because $F^{(i)}(h^{(i)}, \omega)$ relies on only $h^{(i)}$ and ω for all $i \in [N]$, and $h^{(i)}$ are independent of each other

$$\min_{\omega \in \mathcal{W}} \min_{\{h^{(i)} \in \mathcal{H}\}_{i=1}^N} \frac{1}{N} \sum_{i=1}^N p^{(i)} F^{(i)}(h^{(i)}, \omega) \quad (4.5)$$

Equivalently, we have

$$\min_{\omega \in \mathcal{W}, \{h^{(i)} \in \mathcal{H}\}_{i=1}^N} \frac{1}{N} \sum_{i=1}^N p^{(i)} F^{(i)}(h^{(i)}, \omega) \quad (4.6)$$

Note that $p^{(i)}$ values in (4.6) are obtained from our proposed *FedGradNorm* algorithm which will be derived later as a consequence of another optimization problem. As a result, the problem can be expressed as a bilevel optimization problem, which is an optimization problem containing another optimization problem as a constraint

$$\begin{aligned} \min_{x_u \in X_u, x_l \in X_l} \quad & F(x_u, x_l) \\ \text{s.t.} \quad & x_l = \arg \min_{x_l \in X_l} \{g(x_u, x_l), \text{ s.t. } c_j(x_u, x_l) \leq 0, j = 1, \dots, J\} \\ & C_m(x_u, x_l) \leq 0, \quad m = 1, \dots, M \end{aligned} \quad (4.7)$$

where $F(x_u, x_l)$ represents the upper-level objective function and $g(x_u, x_l)$ represents the lower-level objective function. $\{c_j(x_u, x_l) \leq 0, j = 1, \dots, J\}$ are the constraints for the lower-level optimization problem while $\{C_m(x_u, x_l) \leq 0, m = 1, \dots, M\}$ and the lower-level optimization problem itself are the constraints for the upper-level optimization problem.

Multiple algorithms have been developed to solve bilevel optimization problems in the literature [80]. Different reformulations of the bilevel optimization problem have been made by utilizing the optimality conditions of the lower-level optimization problem to formulate the bilevel optimization problem as a single-level constraint problem [81–83]. In addition, there are recently developed gradient-based

bilevel optimization algorithms [84–89]. Our algorithm is based on iterative differentiation (ITD), as explained in Algorithm 4.

Algorithm 4: Iterative differentiation (ITD) algorithm.

Input: K , D , step sizes α , β , initialization $x_u(0)$, $x_l(0)$.

for $k = 0, 1, 2, \dots, K$ **do**

Set $x_l^0(k) = x_l^D(k-1)$ if $k > 0$ otherwise $x_l(0)$.

for $t = 1, \dots, D$ **do**

Update $x_l^t(k) = x_l^{t-1}(k) - \alpha \nabla_{x_l} g(x_u(k), x_l^{t-1}(k))$

end for

Compute $\hat{\nabla}_{x_u} F(x_u(k), x_l^D(k)) = \frac{\partial F(x_u(k), x_l^D(k))}{\partial x_u}$

Update $x_u(k+1) = x_u(k) - \beta \hat{\nabla}_{x_u} F(x_u(k), x_l^D(k))$

end for

Updates to the upper-level optimization take place in the outer loop, while updates to the lower-level optimization are done in the inner loop.

For our problem, x_u and x_l correspond to $(\{h^{(i)}\}_{i=1}^N, \omega)$, $\{p^{(i)}\}_{i=1}^N$, respectively. Additionally, $x_u(k)$ and $x_l(k)$ are denoted as $(\{h^{(i)}\}_{i=1}^N, \omega_k)$, $\{p_k^{(i)}\}_{i=1}^N$ to represent the outer loop iteration index in Algorithm 4 for the rest of the work. Also, i in p_k^i represents the inner loop iteration index, while i in $p_k^{(i)}$ represents the client index. Then, the bilevel optimization problem in our case can be written as

$$\begin{aligned}
& \min_{\omega, \{h^{(i)}\}_{i=1}^N, \{p^{(i)}\}_{i=1}^N} F(\{h^{(i)}\}_{i=1}^N, \omega, \{p^{(i)}\}_{i=1}^N) \\
& \text{s.t.} \quad \{p^{(i)}\}_{i=1}^N \in \arg \min_{\{p^{(i)}\}_{i=1}^N \in \mathbb{R}^N} F_{grad}
\end{aligned} \tag{4.8}$$

The objective function is a weighted sum of local loss functions, $F = \frac{1}{N} \sum_{i=1}^N p^{(i)} F^{(i)}(h^{(i)}, \omega)$, and F_{grad} is the auxiliary loss function defined by the *FedGradNorm* algorithm in the next section.

4.3 Algorithm Description

In this section, we present the *FedGradNorm* algorithm after introducing the definitions and preliminaries.

4.3.1 Definitions and Preliminaries

In *FedGradNorm*, we aim to learn the dynamic loss weights $\{p^{(i)}\}_{i=1}^N$ given in the lower-level optimization problem of (4.8). The main objective of the algorithm is to dynamically adjust the gradient norms so that the different tasks across clients can be trained at similar learning speeds. In the rest of the work, *clients* and *tasks* will be used interchangeably as we assume that each client has its own different task. Before describing the algorithm in detail, we first introduce the notation:

- $\tilde{\omega}$: A subset of the global shared network parameters $\tilde{\omega} \subset \omega$. *FedGradNorm* is applied on $\tilde{\omega}_k^{(i)} \subset \omega_k^{(i)}$, which is a subset of the global shared network parameters at client i at iteration k . $\tilde{\omega}_k^{(i)}$ is generally chosen as the last layer of the global shared network at client i at iteration k .
- $G_{\tilde{\omega}_k^{(i)}}^{(i)}(k) = \|\nabla_{\tilde{\omega}_k^{(i)}} p_k^{(i)} F_k^{(i)}\| = p_k^{(i)} \|\nabla_{\tilde{\omega}_k^{(i)}} F_k^{(i)}\|$: The ℓ_2 norm of the gradient of the weighted task loss at client i at iteration k with respect to the chosen weights $\tilde{\omega}_k^{(i)}$.

- $\bar{G}_{\tilde{\omega}}(k) = \mathbb{E}_{j \sim \text{task}}[G_{\tilde{\omega}_k^{(j)}}^{(j)}(k)]$: The average gradient norm across all clients (tasks) at iteration k .
- $\tilde{F}_k^{(i)} = \frac{F_k^{(i)}}{F_0^{(i)}}$: Inverse training rate of task i (at client i) at iteration k , where $F_k^{(i)}$ is the loss for client i at iteration k , and $F_0^{(i)}$ is the initial loss for client i .
- $r_k^{(i)} = \frac{\tilde{F}_k^{(i)}}{\mathbb{E}_{j \sim \text{task}}[\tilde{F}_k^{(j)}]}$: Relative inverse training rate of task i at iteration k .

Additional notation that is useful in algorithm description:

- $g_k^{(i)} = \frac{1}{\tau_\omega} \sum_{j=1}^{\tau_\omega} g_{k,j}^{(i)}$ is the average of gradient updates at client i at iteration k , where $g_{k,j}^{(i)}$ is the j th local update of the global shared representation at client i at iteration k . Note that $\|\nabla_{\tilde{\omega}_k^{(i)}} F_k^{(i)}\|$ is a subset of $g_k^{(i)}$ since $\tilde{\omega} \subset \omega$.
- $h_{k,j}^{(i)}$ is the client-specific head parameters $h^{(i)}$ after the j th local update on the client-specific network of client i at iteration k , $j = 1, \dots, \tau_h$.
- $\omega_{k,j}^{(i)}$ is the global shared network parameters of client i after the j th local update at iteration k , $j = 1, \dots, \tau_\omega$. Additionally, $\omega_k^{(i)}$ denotes $\omega_{k,\tau_\omega}^{(i)}$ for brevity.

4.3.2 FedGradNorm Description

FedGradNorm adjusts gradient magnitudes to balance training rates between different tasks across clients. *FedGradNorm* is distributed across the clients and the parameter server. $\bar{G}_{\tilde{\omega}}$ is used to have a common scale for the gradient sizes while the gradient norms are adjusted according to the relative inverse training rates $r_k^{(i)}$. A higher value of $r_k^{(i)}$ leads to a larger gradient magnitude for task i , which encourages the task to train faster. Each client i sends its inverse training rate $\tilde{F}_k^{(i)}$ at time k to

the PS, so that the PS can construct $r_k^{(i)}$, $i \in [N]$. Therefore, given the common scale of gradient magnitudes, and the relative inverse training rate, the desired gradient norm of task i at iteration k is calculated as $\bar{G}_{\bar{\omega}}(k) \times [r_k^{(i)}]^\gamma$, where γ represents the strength of the restoring force which pulls tasks back to a common training rate, which can also be considered as a metric of task asymmetry across different tasks. In cases where tasks have different learning dynamics, a larger γ would be a better choice for a stronger balancing.

In order to shift the gradient norms towards the desired norm, the loss weights $p_k^{(i)}$ are updated by minimizing an auxiliary loss function $F_{\text{grad}}\left(k; \{p_k^{(i)}\}_{i=1}^N\right)$ which is the sum of ℓ_2 distances between the actual gradient norm and the desired gradient norm across all tasks for each iteration k , i.e.,

$$F_{\text{grad}}\left(k; \{p_k^{(i)}\}_{i=1}^N\right) = \sum_{i=1}^N F_{\text{grad}}^{(i)}\left(k; p_k^{(i)}\right) \quad (4.9)$$

$$= \sum_{i=1}^N \left\| p_k^{(i)} \|\nabla_{\bar{\omega}_k^{(i)}} F_k^{(i)}\| - \bar{G}_{\bar{\omega}}(k) \times [r_k^{(i)}]^\gamma \right\| \quad (4.10)$$

The auxiliary loss function $F_{\text{grad}}\left(k; \{p_k^{(i)}\}_{i=1}^N\right)$ is constructed by the parameter server at each global iteration k by using $\nabla_{\bar{\omega}_k^{(i)}} F_k^{(i)}$, which is a subset of the whole gradient of the global shared network sent by client i at iteration k for the global aggregation.

Next, the parameter server performs the differentiation of $F_{\text{grad}}\left(k; \{p_k^{(i)}\}_{i=1}^N\right)$ with respect to each element of $\{p^{(i)}\}_{i=1}^N$ so that $\nabla_{p^{(i)}} F_{\text{grad}}$ is applied via gradient descent to update $p^{(i)}$. The desired gradient norm terms, $\bar{G}_{\bar{\omega}}(k) \times [r_k^{(i)}]^\alpha$, are treated as constant to prevent loss weights $\{p^{(i)}\}_{i=1}^N$ from drifting towards zero while differ-

entiating $F_{\text{grad}}\left(k; \{p_k^{(i)}\}_{i=1}^N\right)$ with respect to each loss weight $p_k^{(i)}$. The weights are updated as,

$$p^{(i)} \leftarrow p^{(i)} - \alpha \nabla_{p^{(i)}} F_{\text{grad}}, \quad i \in [N] \quad (4.11)$$

The updated $\{p^{(i)}\}_{i=1}^N$ are normalized such that $\sum_{i=1}^N p^{(i)} = N$. Finally, the parameter server obtains the global aggregated gradient $g_k = \frac{1}{N} \sum_{i=1}^N p_k^{(i)} g_k^{(i)}$ to update the global shared network parameters ω via $\omega_{k+1} = \omega_k - \beta g_k$ and broadcasts the updated parameters to the clients for the next iteration. The overall *FedGradNorm* algorithm is summarized in Algorithm 5. In *FedGradNorm*, $\text{Update}(f, h)$ represents the generic notation for the update of the variable h by using the gradient of f function with respect to the variable h .

4.4 Convergence Analysis

In this section, we provide the convergence analysis for *FedGradNorm* along with necessary assumptions and lemmas.

Assumption 4.1 *The following strong convexity assumptions hold for upper-level optimization function $F(\cdot)$ and lower-level optimization function $g(\cdot)$ given in (4.7),*

- $g(x, p)$ is μ -strongly convex with respect to $p \in \mathbb{R}^N$
- $F^{(i)}(x, p(x))$ is μ -strongly convex with respect to $x \in \mathcal{H}^N \times \mathcal{W}, \forall i \in [N]$, where $x = (\{h^{(i)}\}_{i=1}^N, \omega)$, and $p^*(x) = \arg \min_{p \in \mathbb{R}^N} g(x)$.

Algorithm 5: Training with *FedGradNorm*

Initialize $\omega_0, \{p_0^{(i)}\}_{i=1}^N, \{h_0^{(i)}\}_{i=1}^N$

for $k=1$ **to** K **do**

The parameter server sends the current global shared network parameters ω_k to the clients.

for Each client $i \in [N]$ **do**

Initialize global shared network parameters for local updates by $\omega_{k,0}^{(i)} \leftarrow \omega_k$

for $j = 1, \dots, \tau_h$ **do**

$h_{k,j}^{(i)} = \text{Update}(F^{(i)}(h_{k,j-1}^{(i)}, \omega_{k,0}^{(i)}), h_{k,j-1}^{(i)})$

end for

$F_k^{(i)} = 0$

for $j = 1, \dots, \tau_\omega$ **do**

$\omega_{k,j}^{(i)} \leftarrow \omega_{k,j-1}^{(i)} - \beta g_{k,j}^{(i)}$

$F_k^{(i)} += F^{(i)}(h_{k,\tau_h}^{(i)}, \omega_{k,j}^{(i)})$

end for

$F_k^{(i)} \leftarrow \frac{1}{\tau_\omega} F_k^{(i)}$

Client i sends $g_k^{(i)} = \frac{1}{\tau_\omega} \sum_{j=1}^{\tau_\omega} g_{k,j}^{(i)}$, and $\tilde{F}_k^{(i)} = \frac{F_k^{(i)}}{F_0^{(i)}}$ to the parameter server

end for

After collecting $g_k^{(i)}$, and $\tilde{F}_k^{(i)}$ for active clients $i \in [N]$, the parameter server performs the following operations in the order:

- Constructs $F_{\text{grad}}(k; \{p_k^{(i)}\}_{i=1}^N)$ using $\{g_k^{(i)}\}_{i=1}^N$ and $\{\tilde{F}_k^{(i)}\}_{i=1}^N$ as given in eq. (4.10).

- Updates $p_k^{(i)} \leftarrow p_{k-1}^{(i)} - \alpha \nabla_{p^{(i)}} F_{\text{grad}}, \forall i \in [N]$.

- Aggregates the gradient for the global shared network by $g_k = \frac{1}{N} \sum_{i=1}^N p_k^{(i)} g_k^{(i)}$.

- Updates the global shared network parameters with the aggregated gradient by $\omega_{k+1} = \omega_k - \beta g_k$.

- Broadcasts ω_{k+1} to clients for the next global iteration.

end for

Assumption 4.2 $\nabla F(z)$ and $\nabla g(z)$ are L -Lipschitz, i.e., for any z, z' ,

$$\|\nabla F(z) - \nabla F(z')\| \leq L\|z - z'\| \quad (4.12)$$

$$\|\nabla g(z) - \nabla g(z')\| \leq L\|z - z'\| \quad (4.13)$$

Assumption 4.3 The derivatives $\nabla_x \nabla_p g(z)$ and $\nabla_p^2 g(z)$ are τ - and ρ -Lipschitz,

i.e., for any z, z' ,

$$\|\nabla_x \nabla_p g(z) - \nabla_x \nabla_p g(z')\| \leq \tau \|z - z'\| \quad (4.14)$$

$$\|\nabla_p^2 g(z) - \nabla_p^2 g(z')\| \leq \rho \|z - z'\| \quad (4.15)$$

Assumption 4.4 *The expected value of the squared ℓ_2 norm of stochastic gradient of $F(\cdot)$ with respect to p is bounded, i.e.,*

$$\mathbb{E}_\xi [\|\nabla_p F(x, p; \xi)\|^2] \leq M^2 \quad (4.16)$$

The expectation is taken over the randomness of stochasticity of gradient descent, where ξ represents the stochastic data samples.

Assumption 4.5 *The stochastic gradient of $F(\cdot)$ with respect to x is an unbiased estimator of the gradient, i.e.,*

$$\mathbb{E}_\xi [\nabla_p F(x, p; \xi)] = \nabla_p F(x, p) \quad (4.17)$$

where ξ represents the stochastic data samples.

The following lemma characterizes the Lipschitz properties of the upper-level objective function $F(\cdot)$. It is adapted from [85, Lemma 2.2].

Lemma 4.1 *Suppose Assumptions 4.1 to 4.5 hold. For any $x, x' \in \mathcal{W} \times \mathcal{H}^N$, we*

have

$$\|\nabla F(x, p) - \nabla F(x', p)\| \leq L_F \|x - x'\| \quad (4.18)$$

where the constant L_F is given by

$$L_F = L + \frac{2L^2 + \tau M^2}{\mu} + \frac{\rho LM + L^3 + \tau ML}{\mu^2} + \frac{\rho L^2 M}{\mu^3} \quad (4.19)$$

Lemma 4.2 Suppose Assumptions 4.1 to 4.5 hold and let $\alpha \leq \frac{1}{L}$. Define $\hat{\nabla}_{x_k} F(x_k, p_k) = \frac{\partial F(x_k, p_k^D)}{\partial x_k}$ and $\nabla_{x_k} F(x_k, p_k) = \frac{\partial F(x_k, p_k^*)}{\partial x_k}$. Then,

$$\begin{aligned} \|\hat{\nabla}_{x_k} F(x_k, p_k) - \nabla_{x_k} F(x_k, p_k)\| &\leq \left(\frac{L(L + \mu)(1 - \alpha\mu)^{\frac{D}{2}}}{\mu} \right. \\ &\quad \left. + \frac{2M(\tau\mu + L\rho)}{\mu^2} (1 - \alpha\mu)^{\frac{D-1}{2}} \right) \|p_k^0 - p^*(x_k)\| \\ &\quad + \frac{LM(1 - \alpha\mu)^D}{\mu}. \end{aligned} \quad (4.20)$$

The proof of Lemma 4.2 is provided in Appendix 4.7.

Lemma 4.3 Suppose Assumptions 4.1 to 4.5 hold. Then, the following holds,

$$\begin{aligned} F(x_{k+1}, p_{k+1}) &\leq F(x_k, p_k) - \left(\frac{\beta}{2} - \beta^2 L_F \right) \|\nabla F(x_k, p_k)\|^2 \\ &\quad + \left(\frac{\beta}{2} + \beta^2 L_F \right) \|\hat{\nabla} F(x_k, p_k) - \nabla F(x_k, p_k)\|^2 \end{aligned} \quad (4.21)$$

The proof of Lemma 4.3 is provided in Appendix 4.7.

Theorem 4.1 *Let Assumptions 4.1 to 4.5. Then, the algorithm satisfies*

$$\begin{aligned} F(x_k, p_k) - F(x^*, p^*(x^*)) &\leq \left(\frac{L_F}{2} - \frac{\mu^2 \beta}{2} + \mu^2 \beta^2 L_F \right) (1 - \beta \mu)^{k-1} \|x_0 - x^*\|^2 \\ &\quad + \left(\frac{\beta}{2} + \beta^2 L_F \right) B \end{aligned} \quad (4.22)$$

where

$$\begin{aligned} B = & 3\Delta \left(\frac{L^2(L + \mu)^2(1 - \alpha\mu)^D}{\mu^2} + \frac{4M^2(\tau\mu + L\rho)^2}{\mu^4}(1 - \alpha\mu)^{D-1} \right) \\ & + 3 \frac{L^2 M^2 (1 - \alpha\mu)^{2D}}{\mu^2} \end{aligned} \quad (4.23)$$

Theorem 4.1 shows that *FedGradNorm* algorithm converges exponentially over the iterations. The proof of Theorem 4.1 is provided in e Appendix 4.7.

4.5 Experiments

4.5.1 Dataset Specifications

The following two datasets are used for experiments:

- **Multi-Task Facial Landmark (MTFL) [77]:** This dataset contains 10,000 training and 3,000 test images, which are human face images annotated by 1) five facial landmarks, 2) gender, 3) smiling, 4) wearing glasses, and 5) head pose. The first task (five facial landmarks) is a regression task, and other tasks are classification tasks.
- **RadComDynamic [78]:** This dataset is a multi-class wireless signal dataset

which contains 125,000 samples. Samples are radar and communication signals from GNU Radio Companion derived for different SNR values. The dataset contains 6 modulation types and 8 signal types. Dynamic parameters for samples are listed in Table 4.1. We perform 3 different tasks over RadComDynamic dataset, 1) modulation classification, 2) signal type classification, and 3) anomaly detection.

- *Task 1. Modulation classification:* The modulation classes are amdsb, amssb, ask, bpsk, fmcw, and pulsed continuous wave (PCW).
- *Task 2. Signal type classification:* The signal classes are AM radio, short-range, Radar-Altimeter, Air-Ground-MTI, Airborne-detection, Airborne-range, Ground-mapping.
- *Task 3. Anomaly behavior:* Signal to noise ratio (SNR) can be considered as a proxy for geo-location information. We define anomaly behavior as having SNR lower than -4 dB.

Each data-point in this dataset is a normalized signal vector of size 256 which is obtained by vectorizing the real and complex parts of the signal, $x = x_I + jx_Q$ where $x_I, x_Q \in \mathbb{R}^{128}$, as follows,

$$\hat{x} = \begin{bmatrix} x_I \\ x_Q \end{bmatrix} \in \mathbb{R}^{256} \quad (4.24)$$

4.5.2 Hyperparameters and Model Specifications

A detailed description of the hyperparameters of the system model for *FedGradNorm* is given in Table 4.2. Note that γ is a hyperparameter that should be determined with respect to the task asymmetry in the system. We use Adam optimizer for both network training and F_{grad} optimization. β is a learning rate that optimizer uses to update the global shared network as well as the personalized network on the client side, and α is a learning rate used for F_{grad} optimization. The shared network model is explained in Table 4.3. Each client also has a simple linear layer that maps the shared network’s output to the corresponding prediction value for a personalized network. Cross-entropy and mean squared error (MSE) are used as the loss functions for classification and regression tasks, respectively.

Table 4.1: RadComDynamic: Dynamic settings.

Dynamic Parameters	Value
Carrier frequency offset std. dev/sample	0.05 Hz
Maximum carrier frequency offset	250 Hz
Sample rate offset std. dev/sample	0.05 Hz
Maximum sample rate offset	60 Hz
Num. of sinusoids in freq. selective fading	5
Maximum doppler frequency	2 Hz
Rician K-factor	3
Fractional sample delays comprising PDP	[0.2, 0.3, 0.1]
Number of multipath taps	5
List of magnitudes corresponding to each delay in PDP	[1, 0.5, 0.5]

4.5.3 Results and Analysis

In the experiments with MTFD dataset, we observe that task 1 (facial landmark regression task) has a higher gradient norm compared to all other tasks, which are

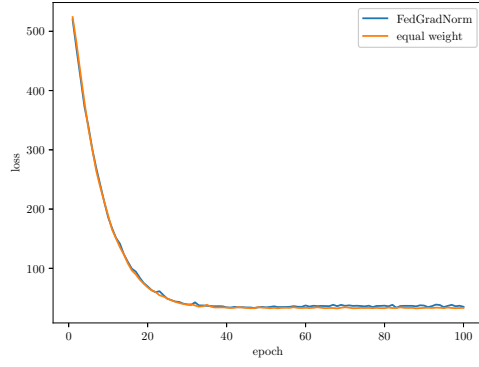
Table 4.2: System model hyperparameters.

Hyperparameter	Value
Optimizer	Adam
γ	0.9
Learning rate (β)	0.0002
Learning rate (α)	0.004

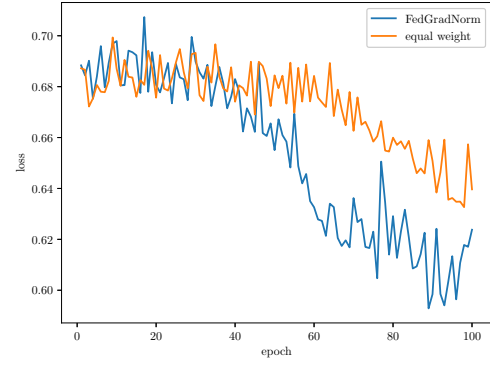
Table 4.3: Shared network model.

NETWORK 1	NETWORK 2
CONV2D(1, 16, 5)	FC(256, 512)
MAXPOOL2D(2, 2)	FC(512, 1024)
CONV2D(16, 48, 3)	FC(1024, 2048)
MAXPOOL2D(2, 2)	FC(2048, 512)
CONV2D(48, 64, 3)	FC(512, 256)
MAXPOOL2D(2, 2)	
CONV2D(64, 64, 2)	

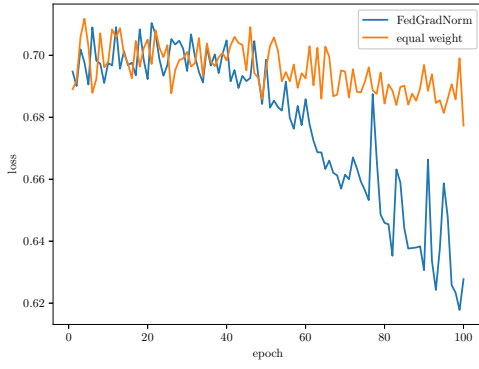
classification tasks. Fig. 4.2 illustrates how *FedGradNorm* gradually decreases the loss weight of the first task to balance the learning speed among tasks. At epoch 70, when tasks 2 and 3 finally can reduce their loss with a higher rate, the weight of their corresponding tasks decreases to help improve the two remaining tasks. Tasks 2 and 3 could not be improved without dynamic-weighting since task 1 would mask the gradient updates for the remaining tasks. As a result of reducing the weight of tasks 2 and 3, the weight of tasks 4 and 5 would then be increased with a similar slope in order to improve the training performance if possible. Unlike other tasks, task 4 (detecting glasses on human faces) and task 5 (pose estimation) reach the minimum very quickly on the first epochs as they are easy tasks. Thus, as indicated by Fig. 4.2, the performance does not improve much for tasks 4 and 5. Although the performance of tasks 1 and 5 are also quite the same in the long-run, *FedGradNorm* helps to learn faster at the early stages. For Fig. 4.2, the data allocation is balanced.



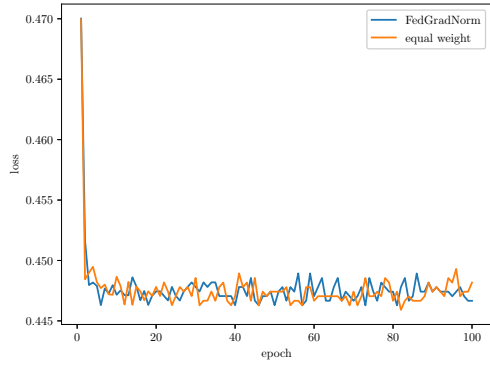
(a)



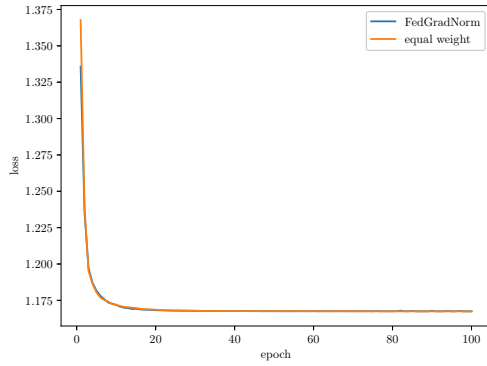
(b)



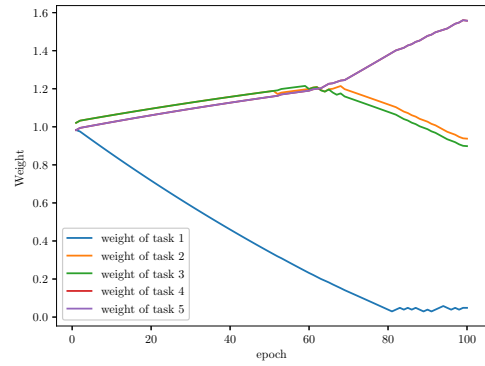
(c)



(d)



(e)



(f)

Figure 4.2: Comparison of task losses in *FedGradNorm* and *FedRep*; balanced data allocation among tasks (a) task 1 (face landmark), (b) task 2 (gender), (c) task 3 (smile), (d) task 4 (glasses), (e) task 5 (pose), (f) task weights.

We also perform experiments with the imbalanced data distribution. Table 4.4 exhibits the loss comparisons between *FedGradNorm* and *FedRep* when task 2 and task 4 have access to 500 data points, whereas other tasks have 3000 data points. The *FedGradNorm* performs better than *FedRep*.

Table 4.4: Comparison of task losses after 100 epochs in *FedGradNorm* and *FedRep*; imbalanced data allocation among tasks.

Tasks	face landmark	gender	smile	glass	pose
<i>FedRep</i> loss	33.28	0.66	0.60	0.44	1.1
<i>FedGradNorm</i> loss	33.25	0.56	0.57	0.43	1.1

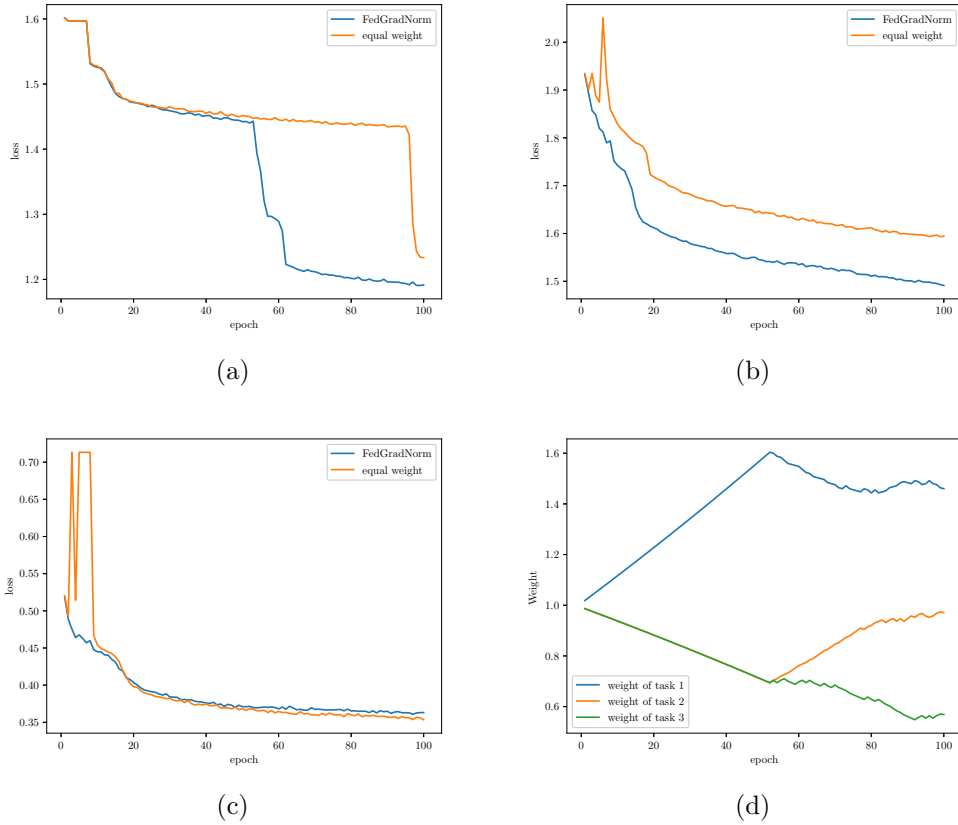


Figure 4.3: Comparison between task accuracy achieved via *FedGradNorm* and *FedRep* in RadComDynamic dataset (a) task 1 (modulation classification), (b) task 2 (signal classification), (c) task 3 (anomaly behavior), (d) task weights.

Furthermore, we conduct experiments with the RadComDynamic dataset us-

ing Network 2 given in Table 4.3. *FedGradNorm* outperforms *FedRep* on modulation detection and signal detection tasks, as illustrated in Fig. 4.3. The modulation detection task and the signal detection task have slower training than the anomaly detection task with respect to the change of the loss. By employing *FedGradNorm*, we demonstrate that the learning speed of signal and modulation detection tasks are balanced against the anomaly detection task. Moreover, we observe that the loss weight for task 1 is increased to speed up its training at the beginning of the training since the loss for task 2 and task 3 decreases faster compared to the loss of task 1 initially. In epoch 55, the loss of task 1 decreases significantly. Therefore, the loss weight of task 1 is decreased to prevent task 1 from dominating the training.

4.6 Conclusion and Discussion

We proposed *FedGradNorm*, a distributed version of the *GradNorm* dynamic weighting algorithm for the personalized FL setting. We provided the convergence analysis for *FedGradNorm* and showed that it has an exponential convergence rate. In the experiments with *FedGradNorm*, the learning speed and task performance of *FedGradNorm* were compared with the naive equal weighting strategy. In contrast to naively assigning equal weights to each task, we observed that *FedGradNorm* could ensure faster training and more consistent performance. Additionally, *FedGradNorm* could compensate for the effects of imbalanced allocation of data among the clients. Tasks with insufficient data are also eligible for fair training since the weights of task losses are adjusted with respect to training speeds to encourage the

slow learning tasks. An extension of this work is presented in [63, 90], where we proposed *HOTA-FedGradNorm*, a modified version of *FedGradNorm* that incorporates over-the-air aggregation in a hierarchical FL setting, explicitly considering wireless channel characteristics.

4.7 Appendix

Proof: [Proof of Lemma 4.2] Since p_k depends on x_k , the following holds by the chain rule,

$$\frac{\partial F(x_k, p_k^i)}{\partial x_k} = \nabla_x F(x_k, p_k^i) + \frac{\partial p_k^i}{\partial x_k} \nabla_p F(x_k, p_k^i)$$

Then,

$$\begin{aligned} \left\| \frac{\partial F(x_k, p_k^D)}{\partial x_k} - \frac{\partial F(x_k, p^*(x_k))}{\partial x_k} \right\| &= \left\| \nabla_x F(x_k, p_k^D) + \frac{\partial p_k^D}{\partial x_k} \nabla_p F(x_k, p_k^D) - \nabla_x F(x_k, p^*(x_k)) \right. \\ &\quad \left. - \frac{\partial p^*(x_k)}{\partial x_k} \nabla_p F(x_k, p^*(x_k)) + \frac{\partial p^*(x_k)}{\partial x_k} \nabla_p F(x_k, p_k^D) \right. \\ &\quad \left. - \frac{\partial p^*(x_k)}{\partial x_k} \nabla_p F(x_k, p_k^D) \right\| \end{aligned} \quad (4.25)$$

$$\begin{aligned} &\leq \left\| \nabla_x F(x_k, p_k^D) - \nabla_x F(x_k, p^*(x_k)) \right\| \\ &\quad + \left\| \frac{\partial p_k^D}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} \right\| \left\| \nabla_p F(x_k, p_k^D) \right\| \\ &\quad + \left\| \frac{\partial p^*(x_k)}{\partial x_k} \right\| \left\| \nabla_p F(x_k, p_k^D) - \nabla_p F(x_k, p^*(x_k)) \right\| \end{aligned} \quad (4.26)$$

where the last inequality follows from the triangle inequality. The first term of (4.26) is bounded by Assumption 4.2 as follows,

$$\left\| \nabla_x F(x_k, p_k^D) - \nabla_x F(x_k, p^*(x_k)) \right\| \leq L \|p_k^D - p^*(x_k)\| \quad (4.27)$$

Similarly, the third term of (4.26) is bounded by Assumption 4.1

$$\left\| \frac{\partial p^*(x_k)}{\partial x_k} \right\| \left\| \nabla_p F(x_k, p_k^D) - \nabla_p F(x_k, p^*(x_k)) \right\| \leq L \left\| \frac{\partial p^*(x_k)}{\partial x_k} \right\| \|p_k^D - p^*(x_k)\| \quad (4.28)$$

Furthermore, the second term of (4.26) is bounded as

$$\left\| \frac{\partial p_k^D}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} \right\| \left\| \nabla_p F(x_k, p_k^D) \right\| \leq M \left\| \frac{\partial p_k^D}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} \right\| \quad (4.29)$$

since we have $\|\nabla_p F(x, p)\| \leq M$ by Assumptions 4.4, 4.5. Then, (4.26) is upper bounded as

$$\begin{aligned} \left\| \frac{\partial F(x_k, p_k^D)}{\partial x_k} - \frac{\partial F(x_k, p^*(x_k))}{\partial x_k} \right\| &\leq L \|p_k^D - p^*(x_k)\| + L \left\| \frac{\partial p^*(x_k)}{\partial x_k} \right\| \|p_k^D - p^*(x_k)\| \\ &\quad + M \left\| \frac{\partial p_k^D}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} \right\| \end{aligned} \quad (4.30)$$

To upper bound (4.30) further, we bound $\left\| \frac{\partial p_k^D}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} \right\|$.

By the gradient descent update of p , i.e., $p_k^t = p_k^{t-1} - \alpha \nabla_p g(x_k, p_k^{t-1})$ for $t =$

$1, \dots, D$, and by the chain rule,

$$\frac{\partial p_k^t}{\partial x_k} = \frac{\partial p_k^{t-1}}{\partial x_k} - \alpha \left(\nabla_x \nabla_p g(x_k, p_k^{t-1}) + \frac{\partial p_k^{t-1}}{\partial x_k} \nabla_p^2 g(x_k, p_k^{t-1}) \right) \quad (4.31)$$

Additionally, based on the optimality of $p^*(x_k)$, we have $\nabla_p g(x_k, p^*(x_k)) = 0$. Then, by taking the partial derivative with respect to x_k ,

$$\nabla_x \nabla_p g(x_k, p^*(x_k)) + \frac{\partial p^*(x_k)}{\partial x_k} \nabla_p^2 g(x_k, p^*(x_k)) = 0 \quad (4.32)$$

By combining (4.31) and (4.32),

$$\begin{aligned} \frac{\partial p_k^t}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} &= \frac{\partial p_k^{t-1}}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} \\ &\quad - \alpha \left(\nabla_x \nabla_p g(x_k, p_k^{t-1}) + \frac{\partial p_k^{t-1}}{\partial x_k} \nabla_p^2 g(x_k, p_k^{t-1}) \right) \\ &\quad + \alpha \left(\nabla_x \nabla_p g(x_k, p^*(x_k)) + \frac{\partial p^*(x_k)}{\partial x_k} \nabla_p^2 g(x_k, p^*(x_k)) \right) \\ &= \frac{\partial p_k^{t-1}}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} - \alpha (\nabla_x \nabla_p g(x_k, p_k^{t-1}) - \nabla_x \nabla_p g(x_k, p^*(x_k))) \\ &\quad - \alpha \left(\frac{\partial p_k^{t-1}}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} \right) \nabla_p^2 g(x_k, p_k^{t-1}) \\ &\quad + \alpha \frac{\partial p^*(x_k)}{\partial x_k} (\nabla_p^2 g(x_k, p^*(x_k)) - \nabla_p^2 g(x_k, p_k^{t-1})) \end{aligned} \quad (4.33)$$

Also, by (4.32) and Assumptions 4.1, 4.2,

$$\left\| \frac{\partial p^*(x_k)}{\partial x_k} \right\| = \left\| \nabla_x \nabla_p g(x_k, p^*(x_k)) [\nabla_p^2 g(x_k, p^*(x_k))]^{-1} \right\| \leq \frac{L}{\mu} \quad (4.34)$$

By (4.33), (4.34), and Assumption 4.2, we have

$$\begin{aligned} \left\| \frac{\partial p_k^t}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} \right\| &\leq \|I - \alpha \nabla_p^2 g(x_k, p_k^{t-1})\| \times \left\| \frac{\partial p_k^{t-1}}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} \right\| \\ &\quad + \alpha \left(\tau + \frac{L\rho}{\mu} \right) \|p_k^{t-1} - p^*(x_k)\| \end{aligned} \quad (4.35)$$

Furthermore, based on μ -strong convexity of $g(x, \cdot)$ with respect to p , we have

$\nabla_p^2 g(x, \cdot) \geq \mu$ for any $x \in \mathcal{H}^N \times \mathcal{W}$. Then, (4.35) can be simplified as

$$\left\| \frac{\partial p_k^t}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} \right\| \leq (1 - \alpha\mu) \left\| \frac{\partial p_k^{t-1}}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} \right\| + \alpha \left(\tau + \frac{L\rho}{\mu} \right) \|p_k^{t-1} - p^*(x_k)\| \quad (4.36)$$

In addition, the following equation is obtained from μ -strong convexity of $g(x, \cdot)$ with respect to p as well,

$$\|p_k^{t-1} - p^*(x_k)\| \leq (1 - \alpha\mu)^{\frac{t-1}{2}} \|p_k^0 - p^*(x_k)\| \quad (4.37)$$

Inserting (4.37) into (4.36) and telescoping results as

$$\begin{aligned} \left\| \frac{\partial p_k^D}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} \right\| &\leq (1 - \alpha\mu)^D \left\| \frac{\partial p_k^0}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} \right\| \\ &\quad + \alpha \left(\tau + \frac{L\rho}{\mu} \right) \sum_{t=0}^{D-1} (1 - \alpha\mu)^{D-1-t} (1 - \alpha\mu)^{\frac{t}{2}} \times \|p_k^0 - p^*(x_k)\| \end{aligned} \quad (4.38)$$

$$\begin{aligned} &= (1 - \alpha\mu)^D \left\| \frac{\partial p_k^0}{\partial x_k} - \frac{\partial p^*(x_k)}{\partial x_k} \right\| + \frac{2(\tau\mu + L\rho)}{\mu^2} (1 - \alpha\mu)^{\frac{D-1}{2}} \|p_k^0 - p^*(x_k)\| \end{aligned} \quad (4.39)$$

$$\leq \frac{L(1-\alpha\mu)^D}{\mu} + \frac{2(\tau\mu + L\rho)}{\mu^2}(1-\alpha\mu)^{\frac{D-1}{2}} \|p_k^0 - p^*(x_k)\| \quad (4.40)$$

where the last inequality comes from $\frac{\partial p_k^0}{\partial x_k} = 0$ and (4.34). Then, the proof is completed by using (4.40) in (4.30) in addition to upper bounding $\left\| \frac{\partial p^*(x_k)}{\partial x_k} \right\|$ and $\|p_k^D - p^*(x_k)\|$ in (4.30) with (4.34) and (4.37), respectively. ■

Proof: [Proof of Lemma 4.3] Based on L_F -smoothness of $F(\cdot)$,

$$F(x_{k+1}, p_{k+1}) \leq F(x_k, p_k) + (x_{k+1} - x_k)^T \nabla F(x_k, p_k) + \frac{L_F}{2} \|x_{k+1} - x_k\|^2 \quad (4.41)$$

By the gradient descent update of x

$$x_{k+1} - x_k = -\beta \hat{\nabla} F(x_k, p_k) \quad (4.42)$$

By inserting (4.42) into (4.41), the following holds,

$$\begin{aligned} F(x_{k+1}, p_{k+1}) &\leq F(x_k, p_k) + (-\beta \hat{\nabla} F(x_k, p_k) + \beta \nabla F(x_k, p_k) - \beta \nabla F(x_k, p_k))^T \nabla F(x_k, p_k) \\ &\quad + \frac{L_F}{2} \left\| -\beta \hat{\nabla} F(x_k, p_k) + \beta \nabla F(x_k, p_k) - \beta \nabla F(x_k, p_k) \right\|^2 \end{aligned} \quad (4.43)$$

$$\begin{aligned} &= F(x_k, p_k) - \beta \langle \hat{\nabla} F(x_k, p_k) - \nabla F(x_k, p_k), \nabla F(x_k, p_k) \rangle - \beta \|\nabla F(x_k, p_k)\|^2 \\ &\quad + \beta^2 L_F \left\| \hat{\nabla} F(x_k, p_k) - \nabla F(x_k, p_k) \right\|^2 + \beta^2 L_F \|\nabla F(x_k, p_k)\|^2 \end{aligned} \quad (4.44)$$

$$\begin{aligned} &\leq F(x_k, p_k) - \left(\frac{\beta}{2} - \beta^2 L_F \right) \|\nabla F(x_k, p_k)\|^2 \\ &\quad + \left(\frac{\beta}{2} + \beta^2 L_F \right) \left\| \hat{\nabla} F(x_k, p_k) - \nabla F(x_k, p_k) \right\|^2 \end{aligned} \quad (4.45)$$

where the last inequality comes from

$$\|x - y\|^2 = \|x\|^2 + \|y\|^2 - 2\langle x, y \rangle \geq -\|x\|^2 - 2\langle x, y - x \rangle \quad (4.46)$$

by substituting $x = \nabla F(x_k, p_k)$ and $y = \hat{\nabla} F(x_k, p_k)$. ■

Proof: [Proof of Theorem 4.1] By Lemma 4.3, we have

$$\begin{aligned} F(x_{k+1}, p_{k+1}) \leq & F(x_k, p_k) - \underbrace{\left(\frac{\beta}{2} - \beta^2 L_F \right) \|\nabla F(x_k, p_k)\|^2}_{A_1} \\ & + \underbrace{\left(\frac{\beta}{2} + \beta^2 L_F \right) \|\hat{\nabla} F(x_k, p_k) - \nabla F(x_k, p_k)\|^2}_{A_2} \end{aligned} \quad (4.47)$$

To upper bound A_2 , we use Lemma 4.2 and the fact that $(a + b + c)^2 \leq 3a^2 + 3b^2 + 3c^2, \forall a, b, c \in \mathbb{R}$, while also assuming $\|p_k^0 - p_k^*(x_k)\|^2 \leq \Delta$. By choosing $a = \frac{L(L+\mu)(1-\alpha\mu)^{\frac{D}{2}}}{\mu} \Delta^{\frac{1}{2}}$, $b = \frac{2M(\tau\mu+L\rho)}{\mu^2} (1-\alpha\mu)^{\frac{D-1}{2}} \Delta^{\frac{1}{2}}$, $c = \frac{LM(1-\alpha\mu)^D}{\mu}$, we have

$$\begin{aligned} \|\hat{\nabla} F(x_k, p_k) - \nabla F(x_k, p_k)\|^2 \leq & 3\Delta \left(\frac{L^2(L+\mu)^2(1-\alpha\mu)^D}{\mu^2} + \frac{4M^2(\tau\mu+L\rho)^2}{\mu^4} (1-\alpha\mu)^{D-1} \right) \\ & + 3 \frac{L^2 M^2 (1-\alpha\mu)^{2D}}{\mu^2} \end{aligned} \quad (4.48)$$

where constant B is defined as

$$\begin{aligned} B \triangleq & 3\Delta \left(\frac{L^2(L+\mu)^2(1-\alpha\mu)^D}{\mu^2} + \frac{4M^2(\tau\mu+L\rho)^2}{\mu^4} (1-\alpha\mu)^{D-1} \right) \\ & + 3 \frac{L^2 M^2 (1-\alpha\mu)^{2D}}{\mu^2} \end{aligned} \quad (4.49)$$

To upper bound the A_1 , we use the μ -strong convexity of $F(x, p)$ with respect to x by Assumption 4.1. By μ -strong convexity of $F(x, p)$, for any fixed $p \in \mathcal{P}^N$, we have,

$$\nabla F(x_k, p) \geq \mu(x_k - x^*) \quad (4.50)$$

Then,

$$\|\nabla F(x_k)\|^2 \geq \mu^2 \|x_k - x^*\|^2 \quad (4.51)$$

By substituting (4.48) and (4.51) in (4.47), we have

$$F(x_{k+1}, p_{k+1}) \leq F(x_k, p_k) - \mu^2 \left(\frac{\beta}{2} - \beta^2 L_F \right) \|x_k - x^*\|^2 + \left(\frac{\beta}{2} + \beta^2 L_F \right) B \quad (4.52)$$

By subtracting $F(x^*, p^*(x^*))$ from both sides, we have

$$\begin{aligned} F(x_{k+1}, p_{k+1}) - F(x^*, p^*(x^*)) &\leq F(x_k, p_k) - F(x^*, p^*(x^*)) - \mu^2 \left(\frac{\beta}{2} - \beta^2 L_F \right) \|x_k - x^*\|^2 \\ &\quad + \left(\frac{\beta}{2} + \beta^2 L_F \right) B \end{aligned} \quad (4.53)$$

By L_F -smoothness of $F(x, p)$ from Lemma 4.1, and the fact that $\nabla_x F(x, p)|_{\{x=x^*, p=p^*\}} =$

0

$$F(x_k, p_k) - F(x^*, p^*(x^*)) \leq \frac{L_F}{2} \|x_k - x^*\|^2 \quad (4.54)$$

for any x_k . By substituting (4.54) in (4.53), we have

$$F(x_{k+1}, p_{k+1}) - F(x^*, p^*(x^*)) \leq \left(\frac{L_F}{2} - \frac{\mu^2 \beta}{2} + \mu^2 \beta^2 L_F \right) \times \|x_k - x^*\|^2 + \left(\frac{\beta}{2} + \beta^2 L_F \right) B \quad (4.55)$$

Additionally, by the μ -strong convexity of $F(x, p)$ with respect to x , we have

$$\|x_{k+1} - x^*\| \leq (1 - \beta\mu)^{\frac{1}{2}} \|x_k - x^*\| \quad (4.56)$$

$$\|x_k - x^*\| \leq (1 - \beta\mu)^{\frac{k}{2}} \|x_0 - x^*\| \quad (4.57)$$

Then,

$$\begin{aligned} F(x_k, p_k) - F(x^*, p^*(x^*)) &\leq \left(\frac{L_F}{2} - \frac{\mu^2 \beta}{2} + \mu^2 \beta^2 L_F \right) \times (1 - \beta\mu)^{k-1} \|x_0 - x^*\|^2 \\ &\quad + \left(\frac{\beta}{2} + \beta^2 L_F \right) B \end{aligned} \quad (4.58)$$

completing the proof. ■

CHAPTER 5

Personalized Decentralized Multi-Task Learning Over Dynamic Communication Graphs

5.1 Introduction

Decentralized learning (DL) algorithms are able to operate over arbitrary network topologies, in which participants communicate only with their immediate neighbors without a need for communication with a central server. A key challenge in DL is to deal with data heterogeneity: as each agent has its own data, local datasets typically exhibit different distributions. This is especially true for multi-task learning (MTL), where tasks are distributed across different users.

Data heterogeneity is addressed through personalization in federated learning (FL), in which the parameter server and clients train a common base model, and each client additionally trains a small header for its own specific task; shown in Fig. 5.1 for the DL setting. By using personalization, users can obtain essentially different learning models that are better fitted to their unique data while capturing the common knowledge distilled from other devices' data [65, 66, 91].

Prior to the consideration of heterogeneity as a factor in DL convergence, com-

munication topologies have been entirely characterized by their spectral gaps [92]. The choice of topology, however, has a large impact on heterogeneous settings as observed in [93–95]. In the presence of data heterogeneity, choosing a good topology for DL is important. On one hand, clients (tasks) may have adverse effects on each other if the topology of the graph is chosen inappropriately. On the other hand, having a fully connected network can result in a high communication cost. Using a time-varying and data-aware design of the communication network, [96,97] investigate how decentralized SGD performance can be improved in the presence of data heterogeneity. They propose an algorithm to adapt the connectivity matrix (network topology) by minimizing the relative heterogeneity in each round. However, solving an optimization problem that finds the optimal topology in each round results in high computational complexity.

In order to increase user performance, the topology should be configured to connect similar tasks and to isolate unrelated ones. To identify related tasks, we use a technique called *transference* [98]. In MTL [61], transference is a metric to quantify the positive/negative effect of a task’s gradient update to the parameters of a shared encoder on another task’s loss during training. In other words, transference metric of i to j , Z_{ij} , measures the loss of task j before and after applying the gradient update of task i on the shared network. A positive value of Z_{ij} indicates that the update on the shared parameters (by task i) results in a smaller loss on task j than the original parameter values. In contrast, a negative value of transference indicates the negative impact of tasks on each other.

To efficiently design the communication graph, we calculate an approximate

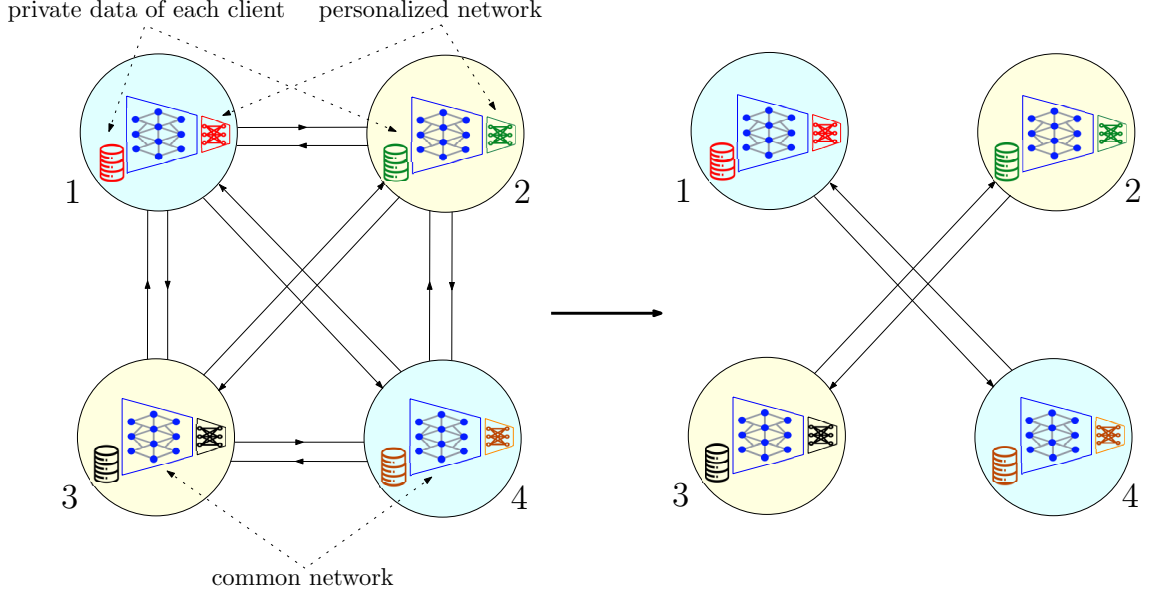


Figure 5.1: Dynamic communication graph in personalized decentralized learning framework with a common network (shown in blue) and small personalized headers (shown in red, green, black, orange). Users with the same-color circles have positive correlation, and different colors mean negative correlation.

transference metric using the gradient updates exchanged among the neighbors. Based on the approximated transference, we dynamically adjust the connectivity matrix W so that the positively correlated tasks are connected together and the negatively correlated tasks are disconnected from each other, thereby, preventing performance degradation. We generate the connectivity matrix by applying the spectral clustering technique [99] to the approximate transference matrix. Our experiments demonstrate that our framework is capable of achieving better performance faster than a fully-connected network in which all users are connected to each other.

The main contributions of our work are as follows: 1) We propose a novel algorithm, which dynamically changes the network topology of the users according to their positive/negative correlations. 2) We conduct several experiments on a syn-

thetic Gaussian dataset and a large-scale celebrity attributes (CelebA) dataset [100]. By deliberately designing the covariance matrix for the synthetic Gaussian datasets, we show that the framework can detect and cluster correlated tasks correctly. In addition, the experiments with CelebA dataset exhibit faster convergence and improved performance in our framework compared to the fully-connected network.

5.2 System Model and Problem Formulation

We consider a personalized distributed learning (PDL) setting [66,91], with N clients which are connected via a directed graph. The network of clients aim to solve the following optimization problem associated with an ML problem,

$$\min_{\{\theta_{s,i}\}_{i=1}^N, \{\theta_i\}_{i=1}^N} \left\{ F(\{\theta_{s,i}\}, \{\theta_i\}) \triangleq \frac{1}{N} \sum_{i=1}^N \mathcal{L}_i(\mathcal{X}_i^t, \theta_{s,i}^t, \theta_i^t) \right\} \quad (5.1)$$

where $\mathcal{X}_i^t$ is the i th user's data at time t , and $\mathcal{L}_i(\mathcal{X}_i^t, \theta_{s,i}^t, \theta_i^t)$ is the task-specific loss obtained by using $\mathcal{X}_i^t$ as an input over the shared network and private head of task i . Each client contains a shared network and a private network. The shared network parameters are represented by $\theta_{s,i}^t$ and the private head parameters are represented by θ_i^t , for user i at time t .

The matrix W represents the graph topology among the nodes: $W_{ij} > 0$ means that agent i can communicate with agent j , while $W_{ij} = 0$ means that agent i cannot communicate with agent j . The matrix W is a mixing matrix, which is composed of values between 0 and 1, and is doubly stochastic, i.e., $1^T W = 1$ and $W 1 = 1$, where 1 is a vector of all ones. Mixing property of the topology matrix W is needed

to ensure convergence of an iterative algorithm to solve problem (5.1).

Similar to the centralized case, the problem in (5.1) can be solved through an alternating minimization approach in the decentralized case, using decentralized SGD. First, client i performs K_1 local gradient based updates to optimize θ_i^t , while the global network parameters at client i , i.e., $\theta_{s,i}^t$, are frozen,

$$\theta_i^{t+1} = \theta_i^t - \eta \nabla_{\theta_i} \mathcal{L}_i(\mathcal{X}_i^t, \theta_{s,i}^t, \theta_i^t) \quad (5.2)$$

Then, by keeping the parameters corresponding to the client-specific head frozen, each node sends the gradient update corresponding to the shared network to its neighbors based on the network topology at time t and aggregates the received updates again based on the network topology specified by the connectivity matrix W , K_2 times as follows,

$$\theta_{s,i}^{t+1} = \sum_{j=1}^N W_{ij}^t (\theta_{s,j}^t - \eta \nabla_{\theta_{s,j}} \mathcal{L}_j(\mathcal{X}_j^t, \theta_{s,j}^t, \theta_j^t)) \quad (5.3)$$

Now, the objective is to automatically capture the correlations among the users and to dynamically modify the mixing matrix W so that the positively correlated tasks are connected and uncorrelated tasks are disconnected to prevent negative transference. To that end, let us define the quantity $\theta_{s,j|i}^{t+1}$ to represent the updated j th shared parameters after a gradient step with respect to loss of task i ,

$$\theta_{s,j|i}^{t+1} = \theta_{s,j}^t - \eta \nabla_{\theta_{s,j}} \mathcal{L}_i(\mathcal{X}^t, \theta_{s,j}^t, \theta_i^t) \quad (5.4)$$

where X^t is a shared data that each client has for calculating the transference value. As a result, the transference of task i on a single task j at time t can be calculated as,

$$Z_{i \rightarrow j}^t = 1 - \frac{\mathcal{L}_j(\mathcal{X}^t, \theta_{s,j|i}^{t+1}, \theta_j^{t+1})}{\mathcal{L}_j(\mathcal{X}^t, \theta_{s,j}^t, \theta_j^t)} \quad (5.5)$$

A positive value for $Z_{i \rightarrow j}^t$ indicates that using task i loss values to update the shared parameters would result in a lower loss for task j . Thus, a more positive value for $Z_{i \rightarrow j}^t$ means more correlation among these two tasks. Conversely, having a negative value means that learning task i and j simultaneously would deteriorate the performance of both of the tasks.

Instead of calculating $\theta_{s,j|i}^{t+1}$ for each user at each time to obtain transference, we can use its first order Taylor series expansion to obtain transference directly from the exchanged gradients among the clients. Let us define the gradient update for the shared network of user i at time t as $g_{s,i}^t = \nabla_{\theta_{s,i}} \mathcal{L}_i(\mathcal{X}^t, \theta_{s,i}^t, \theta_i^t)$. Then, the first order Taylor series expansion for the transference can be written as,

$$Z_{i \rightarrow j}^t = 1 - \frac{\mathcal{L}_j(\mathcal{X}^t, \theta_{s,j|i}^{t+1}, \theta_j^{t+1})}{\mathcal{L}_j(\mathcal{X}^t, \theta_{s,j}^t, \theta_j^t)} \quad (5.6)$$

$$\approx 1 - \frac{\mathcal{L}_j(\mathcal{X}^t, \theta_{s,j}^t, \theta_j^t) - \langle g_{s,j}^t, g_{s,i}^t \rangle}{\mathcal{L}_j(\mathcal{X}^t, \theta_{s,j}^t, \theta_j^t)} \quad (5.7)$$

$$= \frac{\langle g_{s,j}^t, g_{s,i}^t \rangle}{\mathcal{L}_j(\mathcal{X}^t, \theta_{s,j}^t, \theta_j^t)} \quad (5.8)$$

Therefore, instead of calculating $\theta_{s,j|i}^{t+1}$ in each round to obtain $Z_{i \rightarrow j}^t$ value, users can exchange their gradient updates obtained via the shared data X^t to more

efficiently compute $Z_{i \rightarrow j}^t$. After calculating $Z^t \in \mathbb{R}^{N \times N}$ matrix, we must convert it to a doubly stochastic mixing matrix W^t to ensure the convergence of the algorithm. We convert Z^t to a doubly stochastic matrix by clustering the Z^t matrix.

We use the *spectral clustering* technique [99] to perform clustering over a communication graph expressed by matrix Z^t . In spectral clustering, we use the eigenvalues of the graph Laplacian to find the appropriate clusters. To calculate the graph Laplacian, let us define matrix D as the degree matrix, which is a diagonal matrix where the (i, i) th entry indicates the degree of node i (the number of edges connected to node i). Then, the graph Laplacian can be calculated as $L = D - Z$. In the graph Laplacian matrix, diagonal entries are the degrees of the nodes, and off-diagonal entries are the negative edge weights. Finally, we calculate the eigenvalues of the graph Laplacian matrix L .

By sorting the eigenvalues, we see that the number of 0 eigenvalues corresponds to the number of connected components in the graph. Also, an eigenvalue with a small value, e.g., $\text{eig}(L) \leq 1$, indicates that there is almost a separation of the two components. Therefore, we can determine the number of clusters by calculating the number of graph Laplacian eigenvalues which have values less than 1. The vectors associated with those eigenvalues contain information as to how to segment the network. Finally, we perform k -means on those vectors in order to obtain the labels for the nodes. The spectral clustering algorithm is given in Algorithm 6.

Let D_l be the number of nodes in cluster A_l for $l \in 1, \dots, k$. Then, W^t can be

Algorithm 6: Unnormalized spectral clustering [99]

- 1: **Input:** Similarity matrix Z , number k of clusters to construct.
 - 2: Compute the unnormalized Laplacian L .
 - 3: Compute the first k eigenvectors $u_1, \dots, u_k$ of L .
 - 4: Let $U \in \mathcal{R}^{n \times k}$ be the matrix containing the vectors $u_1, \dots, u_k$ as columns.
 - 5: For $i = 1, \dots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i th row of U .
 - 6: Cluster the points $\{y_i\}_{i=1, \dots, n}$ in $\mathbb{R}^k$ with the k -means algorithm into clusters $C_1, \dots, C_k$.
 - 7: **Output:** Clusters $A_1, \dots, A_k$ with $A_i = \{j | y_j \in C_i\}$.
-

calculated as,

$$W^t(i, j) = \begin{cases} \frac{1}{d_i}, & \text{if } i, j \in A_l \\ 0, & \text{otherwise} \end{cases} \quad (5.9)$$

This method ensures that W^t remains doubly-stochastic. To prevent error due to the tolerance of calculating transference, the Z^t matrix is averaged over each H epochs and then W^t is calculated by using spectral clustering.

Finally, in each training step, we perform a decentralized SGD step to exchange and update the shared network across users connected via an edge in the derived topology at time t , W^t . The overall algorithm is summarized in Algorithm 7.

5.3 Convergence Analysis

Our convergence analysis follows [101, 102]. We assume that each worker's objective function $f_i : \mathbb{R}^{d+d_i} \rightarrow \mathbb{R}$ for all i is L -smooth and μ -strongly convex and that the variance on each worker is bounded. We also assume that the connectivity matrix (mixing matrix) is doubly-stochastic. Therefore, the convergence analysis

Algorithm 7: Training with our proposed algorithm

```

1: Input: step sizes  $\eta$ , initialization  $\theta_s, \{\theta_i | i \in N\}, H, K_2$ .
2: for  $t = 1, \dots, T$  do
3:   for each  $i \in N$  (in parallel) do
4:     Compute task-loss  $\mathcal{L}_i(\mathcal{X}_i^t, \theta_{s,i}^t, \theta_i^t)$ 
5:      $\theta_i^{t+1} \leftarrow \theta_i^t - \eta \nabla_{\theta_i} \mathcal{L}_i(\mathcal{X}_i^t, \theta_{s,i}^t, \theta_i^t)$ 
6:   end for
7:   for each  $i, j \in N$  (in parallel) do
8:     Calculating  $Z_{i \rightarrow j}^t$  using (5.6)
9:   end for
10:  Calculating mixing matrix  $V^t$  from  $Z^t$  using (5.9)
11:   $W_{temp}^t = W_{temp}^t + V^t$ 
12:  if  $t = H$  then
13:     $W^t = W_{temp}^t / H$ 
14:     $W_{temp} = 0$ 
15:  end if
16:  for  $k = 1, \dots, K_2$  do
17:    for each  $i \in N$  (in parallel) do
18:       $\theta_{s,i}^{t+\frac{1}{2}} \leftarrow \theta_{s,i}^t - \eta \nabla_{\theta_s} \mathcal{L}_i(\mathcal{X}_i^t, \theta_{s,i}^t, \theta_i^t)$ 
19:       $\theta_{s,i}^{t+1} \leftarrow \sum_{j=1}^N W_{ij}^t \theta_{s,j}^{t+\frac{1}{2}}$ 
20:    end for
21:  end for
22: end for

```

follows from [101, Thm. 4]. Further, [102] uses much weaker assumptions to prove convergence.

5.4 Experimental Results

We compare the task losses achieved by naive fully connected network and our proposed dynamic communication graph algorithm.

5.4.1 Dataset Specifications

We use the following two datasets for our experiments:

Synthetic Gaussian vector dataset that contains 30,000 training data, 10,000 test data, and 10,000 shared data. Each data point in this dataset is a Gaussian vector of size 10, i.e., $x_i \in \mathbb{R}^{10}$, with mean of $\mu \in \mathbb{R}^{10}$ and covariance of $\Sigma \in \mathbb{R}^{10 \times 10}$. Same as Fig. 5.1, the covariance matrix is purposefully designed in such a way that the first and fourth coordinates are positively correlated, and they are negatively correlated with the second and third coordinates. Conversely, the second and third coordinates have a positive correlation, while they both have negative correlations with the first and fourth tasks. We consider four attributes for each sample in our experiments. The first four coordinates with purposefully assigned correlations are selected as those four attributes. For each sample, the attributes are 1 if the corresponding coordinate is greater than the assigned mean and 0 otherwise. This synthetic dataset is designed to test the effectiveness of our proposed algorithm to connect the tasks who have positive correlation among themselves and disconnect those who have negative correlations. The training data points are distributed among clients, but each client has access only to one of the attributes (coordinates).

CelebA dataset [100] that contains 200,000 images, where each image contains 40 attributes. In our experiment, we only use 6 of the attributes, namely, male, mustache, high-cheekbones, smiling, heavy-makeup, and wearing lipsticks. Therefore, the dataset is divided into 6 parts, and each user has access to a single attribute of the given images. The chosen attributes and the expected correlation among attributes is shown in Fig. 5.2.

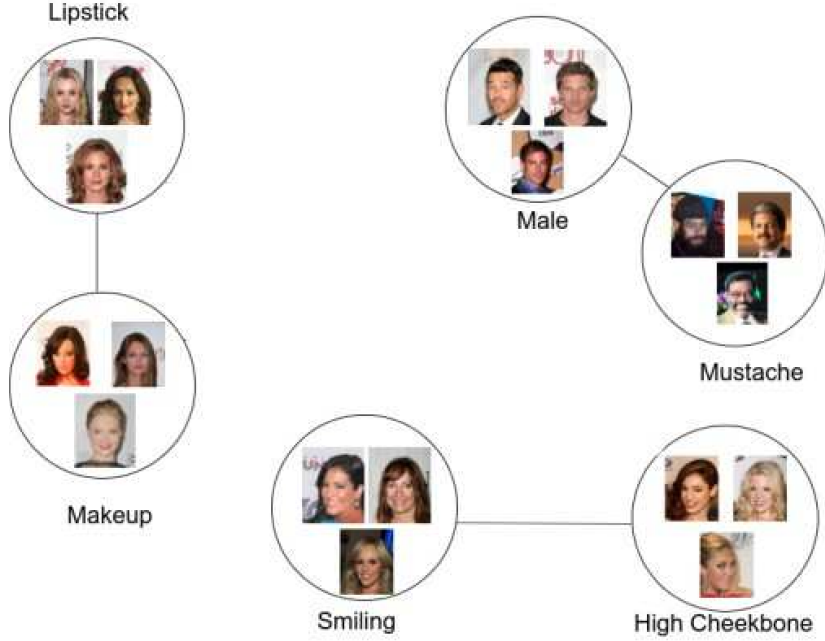


Figure 5.2: An expected communication graph (correlation) among the chosen attributes from CelebA dataset.

5.4.2 Hyperparameters and Model Specifications

We use Adam optimizer with learning rate $\eta = 2 \times 10^{-5}$ to train the shared and personalized networks for both datasets. In the synthetic Gaussian dataset, we use a shared encoder as explained in Table 5.1, and each client also has a simple linear layer as a personalized network that maps the shared network’s output to the corresponding prediction value. We use cross-entropy as a loss function for the classification task.

In CelebA dataset, we use resnet-18 as a shared network for all the users, and a simple 2-layer network for each user to map the output of the shared network

SHARED NETWORK
FC(10, 64)
FC(64, 128)
FC(128, 256)
FC(256, 512)
FC(512, 256)
FC(256, 128)

Table 5.1: Shared network model.

to the corresponding classification task. In both experiments, we perform $K_2 = 2$ decentralized-SGD protocols in each epoch. Also, we average the transference metric through 5 epochs before modifying the connectivity matrix, i.e., $H = 5$.

5.4.3 Results and Analysis

We begin with the synthetic dataset. As shown in Fig. 5.3, by capturing the true correlation among users with the transference method at epoch 15, the connectivity matrix clusters users properly based on their correlation, which results in improved performance as compared to the case where all users are connected regardless of their correlation to the other tasks. For tasks 2-4, our proposed method converges approximately at epoch 30, while the naive fully-connected method converges at epoch 60. A faster convergence is more evident in task 1 where our method converges after 45 epochs, while a naive fully-connected network does not converge satisfactorily even after 80 epochs.

The heatmap of the dynamic topology is shown in Fig. 5.4. According to our proposed method, the connectivity matrix initially starts with a fully-connected network, and the topology changes every five epochs based on the calculated trans-

ference. In epoch 15, the topology finally captures the correct transference as purposefully designed as in Fig. 5.1. Ultimately, the weights are uniformly distributed among tasks 1, 4 and tasks 2, 3 that have positive correlations between them.

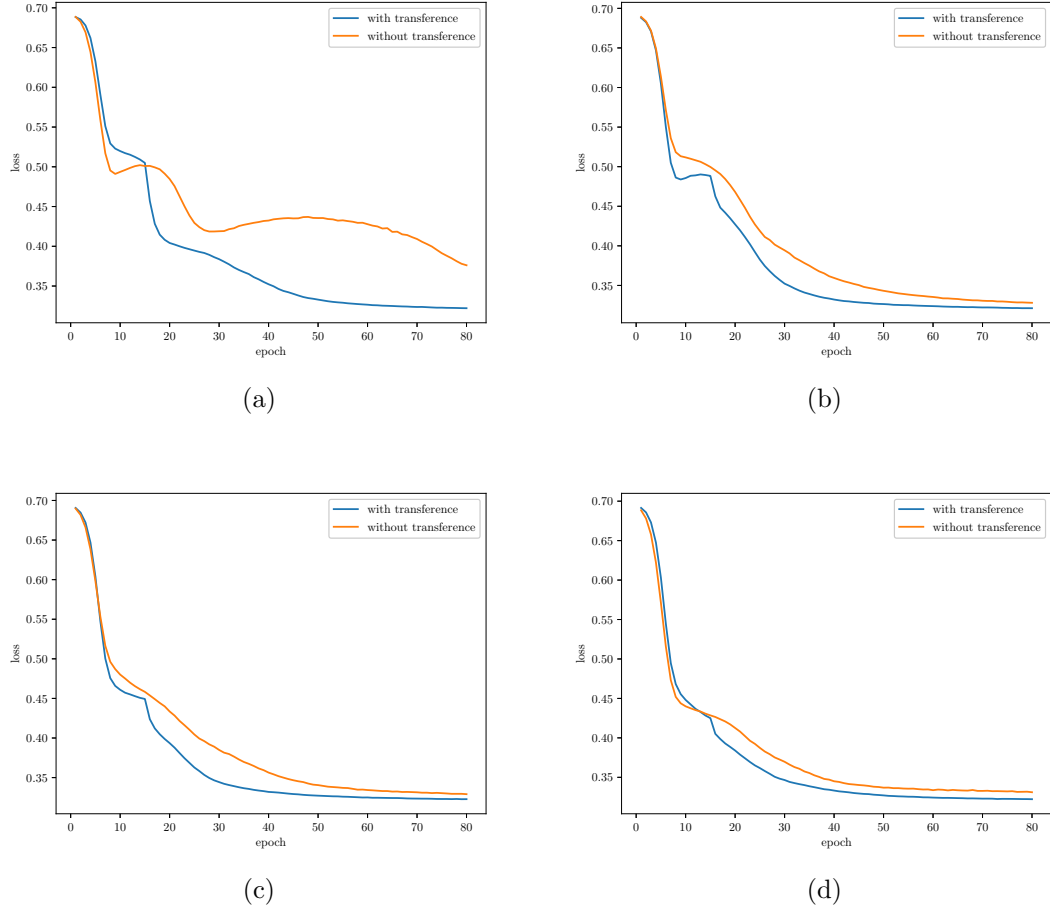


Figure 5.3: Comparison of task losses achieved via dynamic communication graph using transference and naive fully-connected case for synthetic dataset.

Next we examine the proposed method on a larger and more realistic dataset, CelebA. As shown in Fig. 5.5, the results demonstrate the superiority of our method compared to a naive fully-connected network. As part of this experiment, we stopped changing the connectivity matrix at epoch 21 to observe the effect of changing the

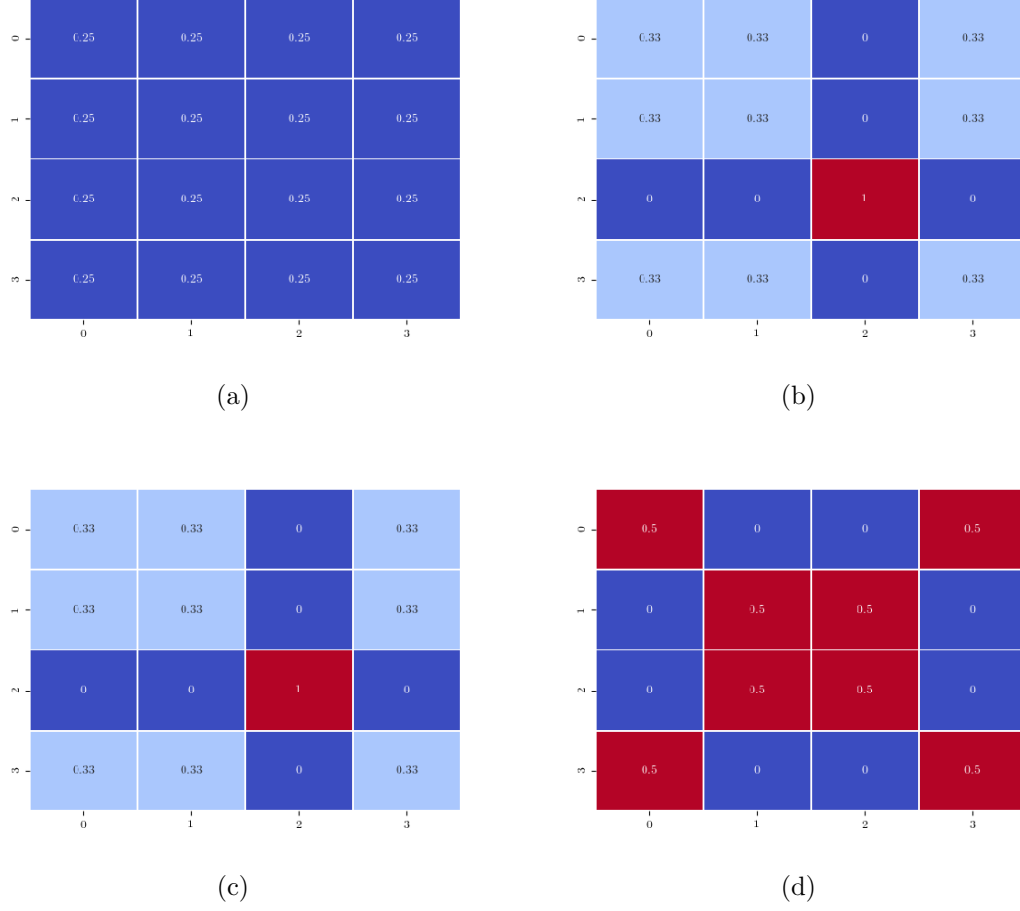
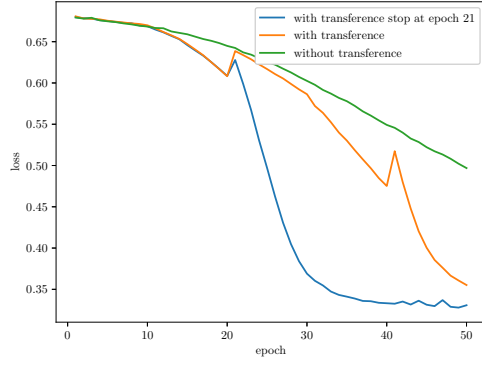
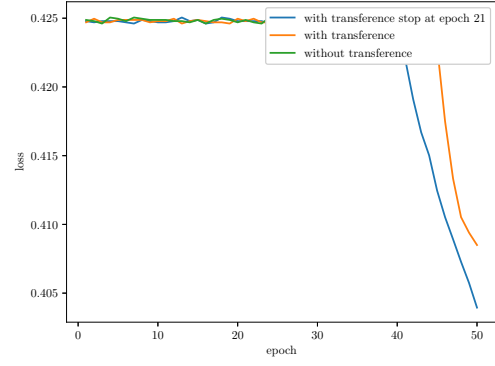


Figure 5.4: Heatmap of the connectivity matrix achieved by the transference method over different epochs, (a) epoch 1, (b) epoch 6, (c) epoch 11, (d) epoch 16.

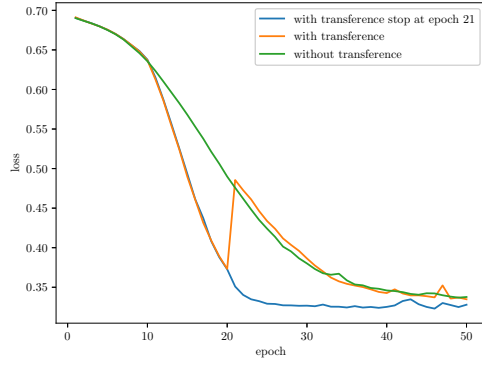
topology on the initial training phase compared with the whole training process. Our results indicate that the training speed increases when the topology is not modified after sufficiently capturing the correlation of tasks during the initial training phase. Therefore, changing the topology until the end of the training may result in drifting from the optimal performance and slower convergence. However, in the majority of our experiments, our proposed method converged faster in both early stopping and permanent topology change cases.



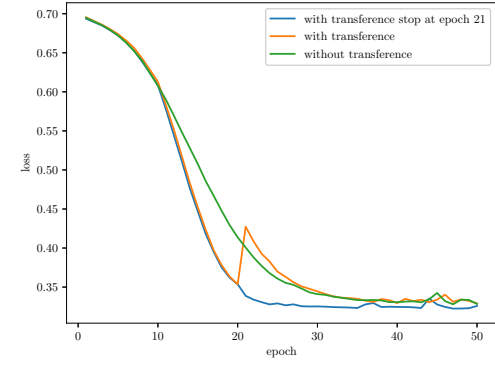
(a)



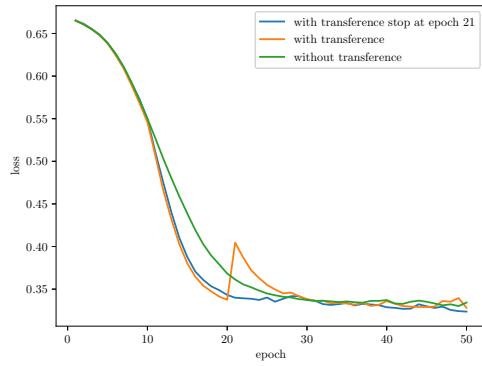
(b)



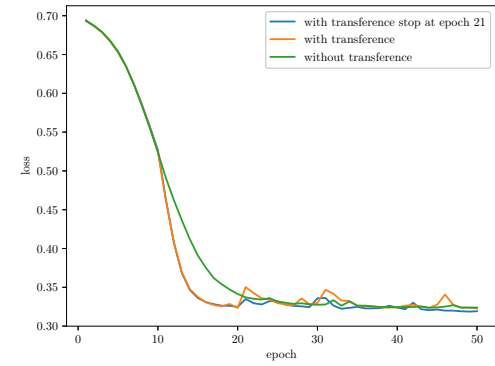
(c)



(d)



(e)



(f)

Figure 5.5: Comparison of task losses achieved via dynamic communication graph using transference and naive fully-connected case for celebA dataset.

As shown in Fig. 5.5, as a result of our proposed method, the loss of task 1 drops with a higher slope after epoch 20, while further topology change can reduce the slope of dropping losses due to the sudden changes that may occur in the topology of the graph. Hence, our method with early stopping topology change converges at approximately epoch 40, while the transference method with a permanent topology change converges at epoch 50, and the naive fully-connected network does not converge satisfactorily even at epoch 50. The second task (learning mustache) converges quickly even from the very first epochs since it is a simple task to learn. Nevertheless, after around epoch 40, when the other correlated task converges appropriately, the loss decreases due to the changed communication topology. As a result of our proposed method, we have observed that task 3-6 have converged faster at around epoch 10, while the losses for the fully-connected tasks have started to decline much more slowly. The permanent modification of the topology matrix, however, may result in sudden divergence from the convergence and may delay the pace of convergence.

5.5 Conclusion

In this work, we proposed a novel personalized decentralized multi-task learning framework designed to address data heterogeneity in decentralized learning environments, where each user aims to learn a personalized task. Unlike prior approaches that rely on fixed fully-connected network topologies, our method dynamically adapts the communication graph among users by leveraging task cor-

relations inferred from exchanged gradient updates. Specifically, we introduced a transference-based metric that quantifies the positive or negative influence of one task’s update on another, enabling us to cluster positively correlated tasks and isolate those with negative interactions. Using spectral clustering on these transference scores, we construct adaptive, data-driven communication topologies that promote beneficial knowledge sharing while mitigating negative transfer. We validated our approach on a synthetic Gaussian dataset and the CelebA large-scale facial attribute dataset, demonstrating that our method converges significantly faster and achieves lower task-specific losses compared to naive fully-connected approaches. The results highlight the potential of dynamically optimized communication graphs in personalized decentralized learning settings, offering improved scalability and robustness in heterogeneous data environments.

CHAPTER 6

***r*Age-*k*: Communication-Efficient Federated Learning Using Age Factor**

6.1 Introduction

Federated learning (FL) [103] is a distributed learning paradigm that enables multiple clients to collaboratively train a shared model while keeping the training data decentralized and private at the clients. The training with the clients is coordinated here by a centralized server known as the parameter server (PS). However, FL algorithms suffer from two major issues: data heterogeneity [20,21] and communication overhead [16, 18, 23–25, 104–107]. The statistical data heterogeneity stems from each client collecting its own data, with the local datasets in FL typically exhibiting different distributions, which can cause model divergence and degrade the FL performance. In addition, exchanging the hundreds of millions of parameters by clients with the PS in modern deep learning (DL) models introduces significant communication overhead on the clients.

In this respect, we review the existing works and commonly used techniques that address these issues. To overcome the challenge of communication overhead,

the gradients can be compressed using quantization [18, 104–106] or sparsification [16, 24, 25, 107] techniques. The success of these compression techniques hinges on the sparse nature of gradient vectors intrinsic to neural networks (NN) during training. We take particular note of two compression strategies popular in the literature: *top-k* sparsification [24] and *rTop-k* sparsification [25]. Under *top-k* sparsification algorithm, clients transmit the indices with the top k largest magnitudes from their gradient vector to the PS in each global iteration. Improving upon *top-k*, *rTop-k* sparsification algorithm entails clients first selecting r ($r > k$) indices with the largest magnitudes and then randomly picking a subset of k indices from the selected indices to send to PS. The superior performance of *rTop-k* over *top-k* sparsification is attributed to the reduction in bias introduced by *top-k* towards exploitation of only the significant entries of the gradient vector, thereby bringing in a strategic mix of exploration and exploitation of the gradient vector during training.

On the other hand, data heterogeneity problem has been addressed in the literature with personalization [65, 66, 91] and clustering [63, 108, 109] methods. Personalization calls for the PS and the clients to initially train a common base model, with each client subsequently training a small header network tailored for its specific task. Through personalization, users can obtain distinct learning models that better fit their unique data, while still incorporating the common knowledge distilled from other devices’ data. Clustering, on the other hand, copes with data heterogeneity by grouping users with similar data distribution into clusters and training a specialized model for the clients in each cluster.

In this work, we address these limitations, hitherto handled separately in the

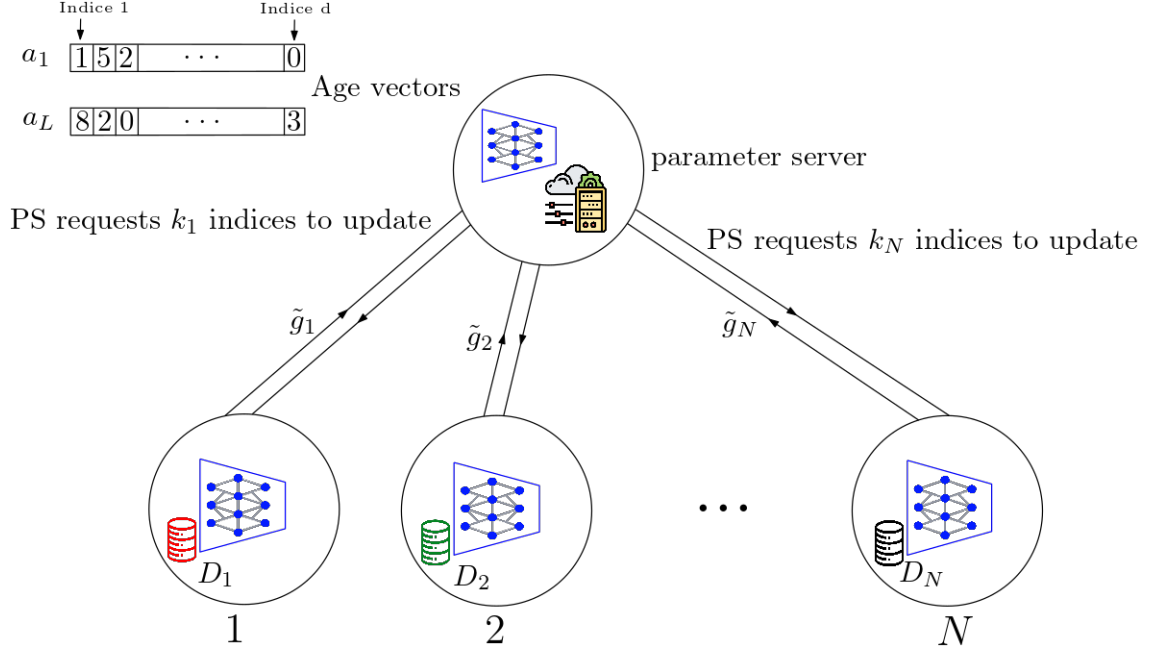


Figure 6.1: System model for $rAge-k$ framework.

literature, simultaneously within a single comprehensive algorithm. We refer to this algorithm as $rAge-k$, for it employs the age of information (AoI) metric to obtain sparsified gradient vectors for training the global model. AoI is defined as the time elapsed since the most recently received information was generated at its source, reflecting the staleness or outdatedness of information [26]. In the context of $rAge-k$ sparsification, this information corresponds to the value in the gradient vector on a particular index, received from any client of a specific cluster, i.e., the source of information. While functional aspects of AoI have been explored in distributed computing and learning [110–114], these works have no overlap with data heterogeneity or gradient sparsification based two-pronged objective of this work.

The pivot of the $rAge-k$ algorithm is the introduction of age vectors at the PS for all the clusters. These age vectors have the same dimension as the number of

parameters in the model, i.e., the dimension of the full gradient vectors, and aid in monitoring the age of gradient updates received on all the indices of the gradient vector. An age vector will not store the same age for all indices, since $rAge-k$, being a sparsification based algorithm, involves communicating only a subset of indices in each round by any client to the PS, inducing nonuniform staleness with respect to different indices. With these age vectors, we can track the most and least frequently updated indices from each cluster, which serves several purposes. First, the age vectors help PS choose a small subset of indices to request updates on in each training round, thereby reducing the communication overhead of training. Second, the updating frequency of the indices tracked under this algorithm helps in identifying clients with statistically similar data, which can be grouped into clusters. The PS can exploit the cluster formations by merging the age vectors of these clients in the cluster, thereby enhancing the coordination of index updates within a cluster among clients. In scenarios where the PS is limited to requesting few updates from the clients, the merged vectors can be used by the PS to strategically choose a disjoint set of indices to request updates on from each individual client within the same cluster.

The main contributions of our work are as follows: 1) We introduce a novel FL framework that integrates the AoI metric to simultaneously address the two major limitations in FL, i.e., communication efficiency and data heterogeneity. 2) We utilize age vectors at the PS to strategically select subsets of indices in each training round to request gradient updates on from each client, and also to identify clients with statistically similar data and group them into clusters. 3) We conduct

extensive experiments on our framework using the MNIST and CIFAR-10 datasets. The experimental results demonstrate the effectiveness of our framework in correctly clustering users with similar types of data, and achieving faster convergence and enhanced accuracy performance.

6.2 System Model and Problem Formulation

We consider an FL setting involving N clients, where each client i possesses a unique local dataset $D_i = \{(\mathbf{x}_j^{(i)})\}_{j=1}^{n_i}$, with n_i representing the size of the local dataset at client i . The clients aim to collaboratively train a model composed of d parameters (with the d -dimensional parameter vector denoted by θ) using their local datasets. The general FL objective with N clients is expressed as,

$$\min_{\theta} \left\{ F(\theta) \triangleq \frac{1}{N} \sum_{i=1}^N \mathcal{L}_i(D_i, \theta_i^t) \right\}, \quad (6.1)$$

where $\mathcal{L}_i(D_i, \theta_i^t)$ represents the loss specific to each client i obtained by using their dataset D_i with their local model parameters θ_i^t at time t .

In our framework, the parameter server maintains a d -dimensional age vector a_l for each cluster $l \in [L]$ within the network, considering each client as an individual cluster upon initially joining the network of clients.

Each client performs H local iterations before participating in a global iteration, i.e., sending its sparse gradients updates, composed of requested indices, to the PS. During global iterations, clients inform the PS of the indices in their gradient vector having the r largest magnitudes. As mentioned earlier, the PS associates

each client with a unique cluster, such that the number of clusters and the cluster constituents are redefined after every M iterations. The PS uses the age vector associated with client-specific cluster to request updates from the client on a subset of k_i ($k_i \ll d$) indices with the highest age out of the client-reported r indices. Subsequently, clients create $\tilde{g}_i^t$ by sparsifying their gradient vectors based on the requested k_i indices, and transmit the non-zero elements of $\tilde{g}_i^t$ and their index location to PS for a global model aggregation. The PS aggregates the received sparse gradient updates ($\tilde{g}^t = \sum_{i=1}^N \tilde{g}_i^t$) to update the global model θ and distributes the updated model back to all the clients.

As shown in Algorithm 9, upon obtaining updates on the requested k_i indices from each client i by the PS, the age vector a_l , where $i \in \text{cluster } l$, will get updated according to the following protocol,

$$a_l^t(j) = \begin{cases} 0, & \text{if } j \in [k_l], \\ a_l^{t-1}(j) + 1, & \text{otherwise,} \end{cases} \quad (6.2)$$

where $a_l^t(j)$ is age of cluster l at time t corresponding to index j . In other words, the age of the requested indices are reset to 0 since they got updated recently, however the age of the remaining indices increase by one, indicating that they are further outdated at the PS.

At intervals of every M iterations, upon receiving the sparsified gradient updates from the clients, the PS initiates an algorithm to identify clients with similar types of data, aiming to group them into the same cluster. We associate a frequency

vector $f^t[i]$ of size d with client i , in which the coordinate j indicates how many times the index j was requested from client i by PS until time t . The PS carries out the clustering process by calculating a similarity matrix for each pair of clients from the dot product of their frequency vectors as follows,

$$d^t[i_1, i_2] = \frac{\langle f^t[i_1], f^t[i_2] \rangle}{\langle f^t[i_1], f^t[i_1] \rangle}. \quad (6.3)$$

Next, the PS employs the DBSCAN (density-based spatial clustering of applications with noise) method [115] to cluster clients based on the similarity matrix computed earlier. DBSCAN distinguishes itself from its counterpart clustering methods in that it does not require the number of clusters to be predetermined. DBSCAN relies on a density-based notion of clustering to group data points, where it identifies clusters as areas of high density of points separated by areas of low density. DBSCAN takes in two parameters as input: *eps*, the maximum distance between two points for them to be considered in the same neighborhood, and *minPts*, the minimum number of points required to form a dense region. Points inside the dense regions are labelled as core points, while the points close to dense regions but not part of them are labelled as border points, while all other points are considered noise. This method is adept at discovering clusters of arbitrary shapes and is robust to outliers. Note that when a client is added to an existing cluster, its age vector is merged with that of the cluster. Conversely, if DBSCAN determines that a client significantly deviates from its current cluster and the client is reassigned to a different group, the age vector relevant for that client is automatically reset due to the changed cluster identity.

The overall training process and the age-function are detailed in Algorithm 8 and Algorithm 9, respectively.

Algorithm 8: Training with our proposed algorithm

```

1: Input: step sizes  $\eta$ , initialization  $\theta$ ,  $H$ ,  $k$ ,  $r$ .
2: for  $t = 1, \dots, T$  do
3:   for each  $i \in N$  (in parallel) do ▷ Local Iteration
4:     Compute client-loss  $\mathcal{L}_i(D_i, \theta_i^t)$ 
5:      $\theta_i^{t+1} \leftarrow \theta_i^t - \eta \nabla_{\theta_i} \mathcal{L}_i(D_i, \theta_i^t)$ 
6:   end for
7:   if  $t \% H = 0$  then ▷ Global Iteration
8:      $\tilde{g}_i^t = \text{rAge-k}(\nabla_{\theta_i} \mathcal{L}_i(D_i, \theta_i^t), a_i, k, r)$ 
9:     Client  $i$  sends non-zero elements of  $\tilde{g}_i^t$  and its location to PS
10:  end if
11:  PS: Update the age vector
12:  PS:  $\tilde{g}^t = \sum_{i=0}^N \tilde{g}_i^t$ 
13:  PS: Update global model  $\theta^t$  based on  $\tilde{g}^t$ 
14:  Send global model  $\theta^t$  to all clients
15:  if  $t \% M = 0$  then ▷ Cluster Checking at PS
16:    for each  $i, j \in N$  (in parallel) do
17:      Calculating  $d^t[i, j]$  using (6.3) to form  $D$ 
18:    end for
19:    Clustering = DBSCAN( $eps, minPts$ ).fit( $D$ )
20:  end if
21: end for

```

6.2.1 Convergence of $rAge-k$

Define $f^{(i)}(\theta) = \mathbb{E}_{X \sim \mathcal{D}_i} [\mathcal{L}_i(X; \theta)]$. We make the following assumptions:

1. Smoothness: The loss functions $f^{(i)}$ are L -smooth:

$$\begin{aligned}
f^{(i)}(\theta) &\leq f^{(i)}(\theta') + \langle \nabla f^{(i)}(\theta'), \theta - \theta' \rangle \\
&\quad + \frac{L}{2} \|\theta - \theta'\|_2^2, \quad \text{for any pair } \theta, \theta'.
\end{aligned} \tag{6.4}$$

Algorithm 9: *rAge-k* function

```

1: rAge-k (gradient-vector( $g$ ), age-vector( $a$ ),  $k$ ,  $r$ ):
2:    $\tilde{g} = \text{Zeros}(g.\text{shape})$ 
3:   (Top-val, Top-ind) = topk(abs( $g$ ),  $r$ )
4:   (Topage-val, Topage-ind) = topk(age[Top-ind],  $k$ )
5:   Top-ind = Top-ind[Topage-ind]
6:    $\tilde{g}[\text{Top-ind}] = g[\text{Top-ind}]$ 
7:    $a += 1$ 
8:    $a[\text{Top-ind}] = 0$ 
9:   return  $\tilde{g}$ , Top-ind,  $a$ .

```

2. Bounded second moment of gradients: For some constant $0 \leq G < \infty$, we have

$$\mathbb{E}_{X \sim \mathcal{D}_i} \|\nabla \mathcal{L}(X; \theta)\|_2^2 \leq G^2. \quad (6.5)$$

3. The ratio of the magnitudes of the largest magnitude index to the r th largest magnitude index is bounded by β at all clients in all iterations.

Definition (Compression operator): A (potentially randomized) function Comp_k is a compression operator if there exists a constant $\gamma \in (0, 1]$ (depending on sparsification parameter k and gradient dimension d) such that

$$\mathbb{E} \|\theta - \text{Comp}_k(\theta)\|_2^2 \leq (1 - \gamma) \|\theta\|_2^2. \quad (6.6)$$

By Assumption 3, *rAge-k* can be shown to be a compression operator with $\gamma = \frac{k}{k + (r-k)\beta + (d-r)}$, and *rAge-k* therefore enjoys the convergence guarantees of $O(\frac{1}{\sqrt{T}})$ from [106, Thm. 1]. Note that when $k = r$, γ becomes simply $\frac{k}{d}$. Loosening r increases β , hence higher exploration of gradient indices in our framework comes

with less strict convergence guarantees.

6.3 Experimental Results

6.3.1 Dataset Specifications

In our experiments, we utilize two datasets: I) MNIST dataset comprising 28×28 gray-scale images of handwritten digits categorized into 10 classes, with the training (test) set of 60,000 (10,000) images, II) CIFAR10 dataset comprising 32×32 RGB images divided into 10 classes, each representing a specific object category such as airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks, with the training (test) set of 50,000 (10,000) images.

Table 6.1: Network Model

NETWORK 1	NETWORK 2
TOTAL PAR (39760)	TOTAL PAR (2515338)
FC(784,50)	CONV2D(3, 64, 3)+BN(64)
RELU	MAXPOOL2D(2, 2)
FC(50,10)	CONV2D(64, 128, 3)+BN(64)
SOFTMAX	CONV2D(128, 256, 3)+BN(256)
	CONV2D(256, 512, 3)+BN(512)
	FC(2048, 128)
	FC(128, 256)
	FC(256, 512)
	FC(512, 1024)
	FC(1024, 10)

6.3.2 Model and Hyperparameters

We use Adam optimizer with learning rate $\eta = 1 \times 10^{-4}$ to train the client models for both datasets. The model architectures for the MNIST and CIFAR10 datasets are

detailed in Table 6.1. For MNIST dataset, the model contains 39,760 parameters, and we set $r = 75$ and $k = 10$. We perform $H = 4$ local iterations before a global iteration wherein clients send their sparse gradient updates to the PS. The PS updates the clusters in the network every $M = 20$ iterations. For CIFAR10 dataset, the model contains 2,515,338 parameters, and we set $r = 2500$ and $k = 100$. We perform $H = 100$ local iterations before the global iteration, and the PS updates the clusters every $M = 200$ iterations. For both MNIST and CIFAR10 datasets, we consider batch size to be 256 during training.

6.3.3 System Performance

With MNIST dataset, to create a highly non-i.i.d. scenario and evaluate the effectiveness of DBSCAN in clustering clients with similar data, we structure our distributed learning system with 10 clients. Each client contains the images from two categories, and further, there are five pairs of clients such that the clients of every pair have data from the same categories. Specifically, the local datasets of clients 1 and 2 come from the same categories/distribution as both handle images with label 0 and 1, likewise clients 3 and 4 handle images with label 2 and 3, and so on.

The heatmap of the weight assigned to pairs of clients inside the connectivity matrix at epochs 1, 21, 41, 61, shown in Fig. 6.2, illustrates the effectiveness of DBSCAN in identifying users with similar image categories/data distributions. The evolution of heatmap in Fig. 6.2 demonstrates that the PS successfully detects

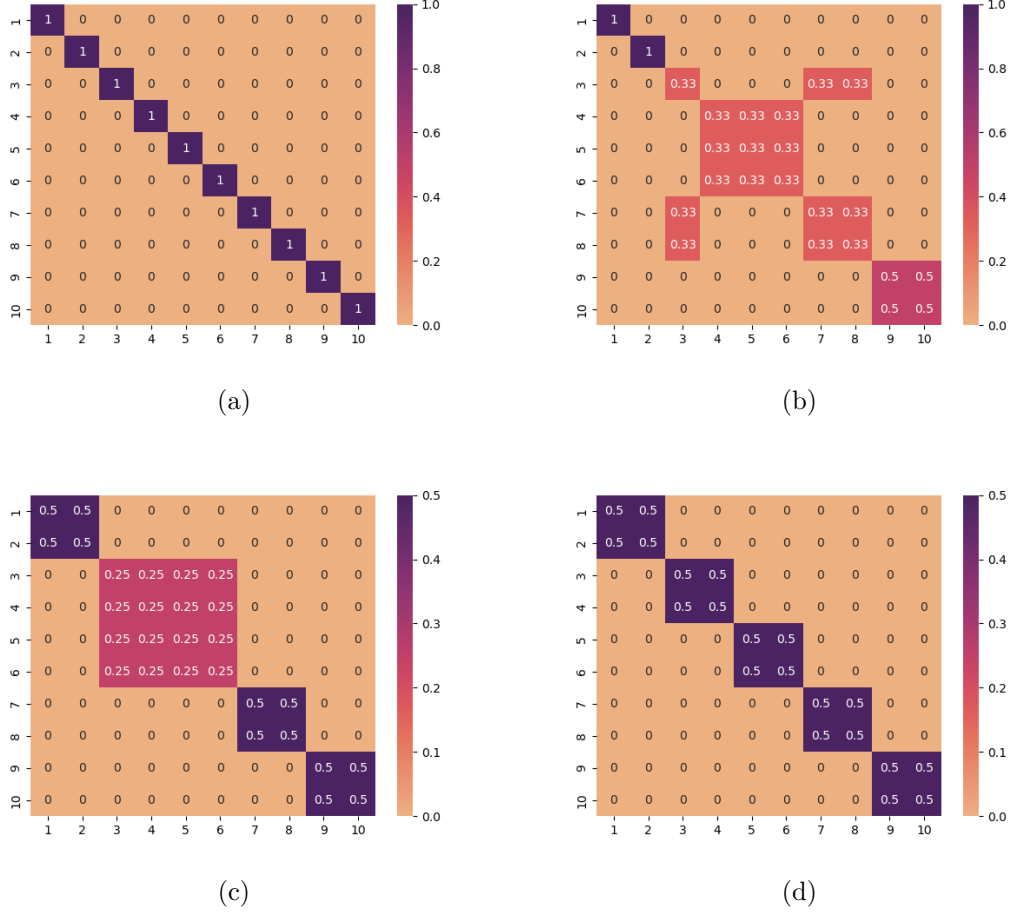


Figure 6.2: Heatmap of the connectivity matrix achieved by the DBSCAN method over different epochs, (a) iteration 1, (b) iteration 21, (c) iteration 41, (d) iteration 61.

similar clients, such that the clustering remains broadly the same until the end of the training.

To test the effectiveness of our communication-efficient framework in terms of accuracy performance, we conduct a comparative analysis with $rTop-k$, a popular communication-efficient algorithm. In $rTop-k$, the client first selects the r indices with the largest magnitudes in its gradient vector and randomly picks k indices from the selection to send updates on to the PS. The aggregation mechanism at the PS

in $rTop-k$ remains consistent with our framework. For a meaningful comparison of $rAge-k$ with $rTop-k$, in our simulations we use the same r and k values in both algorithms.

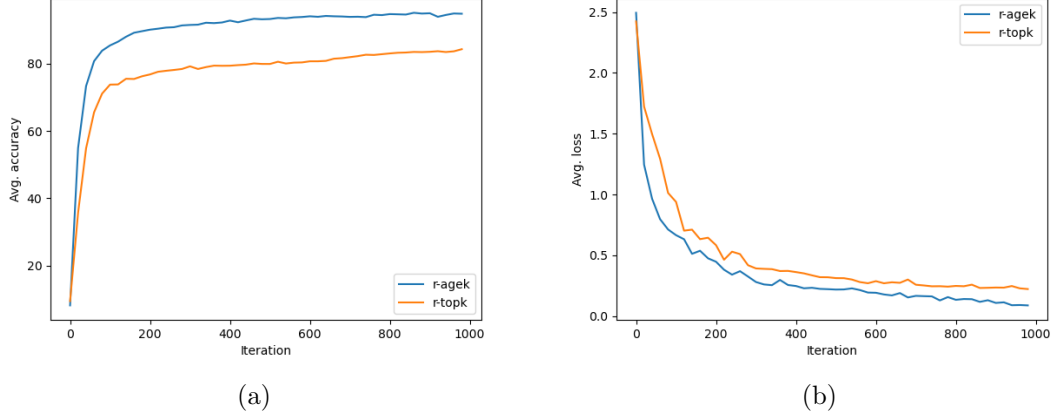


Figure 6.3: (a) Accuracy (in percentage) and (b) loss (averaged over all 10 users) over training iterations with MNIST dataset.

The accuracy performance (in percentage) and the loss corresponding to the two frameworks, with the MNIST dataset, averaged over all 10 users, is plotted in Fig. 6.3(a) and Fig. 6.3(b), respectively. As shown in Fig. 6.3(a), our framework demonstrates faster convergence compared to the $rTop-k$ algorithm, attributed to the ability of $rAge-k$ to cluster users with similar data types. We also note that $rAge-k$ achieves better accuracy performance by the end of the training process. The comparison of loss values achieved by our age-based approach and the $rTop-k$ algorithm is shown in Fig. 6.3(b). Our method exhibits a faster reduction in loss compared to the $rTop-k$ framework where clients independently transmit sparse updates to the PS in a non-i.i.d. client data setting. This improvement underscores our framework’s capability to address both data heterogeneity and communication

overhead issues through accurate clustering of users with similar data distribution via their age vectors at the PS.

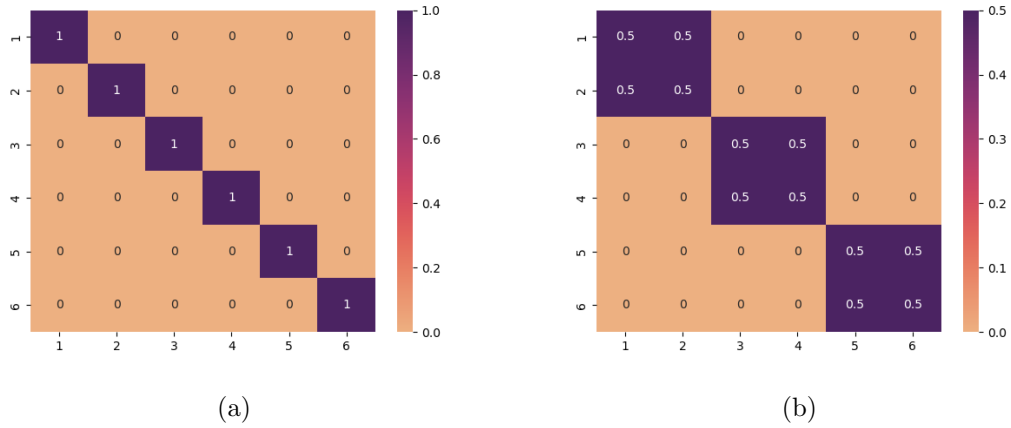


Figure 6.4: Heatmap of the connectivity matrix achieved by the DBSCAN method over different epochs, (a) iteration 1, (b) iteration 201

We similarly evaluate the two frameworks on the CIFAR10 dataset, this time with a total of 6 clients. We distribute the categories amongst clients to create non-i.i.d. client datasets in a similar fashion to the MNIST case, i.e., we assign statistically similar datasets to pairs of clients. To elaborate on the category assignment among clients, clients 1 and 2 handle images with label 1, 2, 3, clients 3 and 4 handle images with label 4, 5, 6, and clients 5 and 6 handle images with label 7, 8, 9, 10. Fig. 6.4 illustrates the clustering effectiveness of DBSCAN, successfully grouping users with similar classes after 200 iterations. The accuracy performance and loss performance for both the frameworks is illustrated in Fig. 6.5(a) and Fig. 6.5(b). Our framework provides a significant boost to the accuracy performance for the CIFAR10 dataset, reaching 80% accuracy by iteration 400, while the $rTop-k$ algorithm only achieves 70% accuracy by iteration 1400. Similarly, our framework

attains lower loss values much sooner compared to the $rTop-k$ algorithm as shown in Fig. 6.5(b).

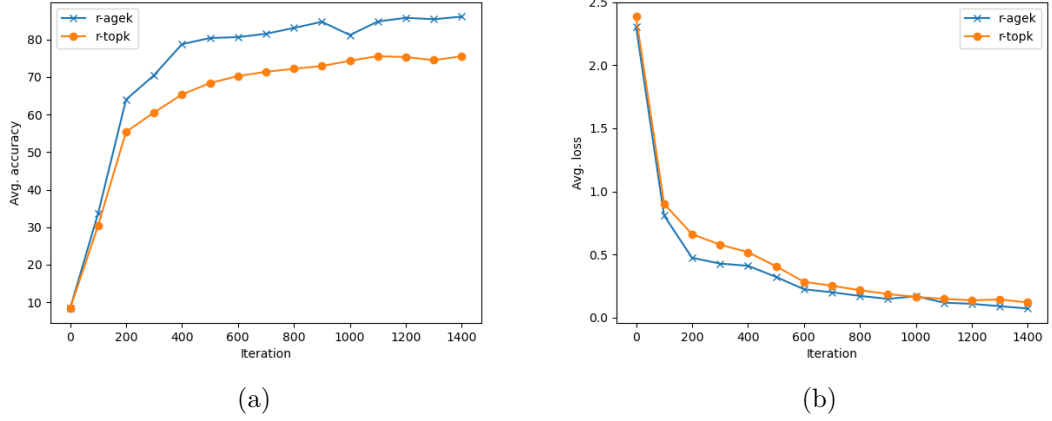


Figure 6.5: (a) Accuracy (in percentage) and (b) loss (averaged over all 6 users) over training iterations with CIFAR10 dataset.

6.4 Conclusion and Discussion

We proposed $rAge-k$, a novel communication-efficient FL framework that utilizes the AoI metric to sparsify gradient vectors transmitted from clients to the PS. Our framework not only reduces communication overhead but also effectively addresses data heterogeneity by clustering users with similar data types. Our experimental results showed that the $rAge-k$ framework outperforms conventional compression algorithm $rTop-k$, achieving faster and more accurate results under the same bandwidth.

Moving forward, this framework can be extended to enhance model performance on more complex datasets, such as CelebA and ImageNet, by incorporating

personalization layers. In this extended setup, all users could collaborate on a shared base model via the PS, while clients within the same cluster could exchange personalized models among themselves. This approach transforms our framework into a form of semi-supervised FL, showcasing its potential for broader applicability and improved model adaptability in heterogeneous data environments.

CHAPTER 7

Age of Gossip in Networks with Multiple Views of a Source

7.1 Introduction

Consider a scenario where a source holds a time-varying information that is crucial to a user or a group of users. The users need to track this information as closely as possible in real-time to achieve their objectives. The age of information (AoI) [26,28,116,117] is a novel metric introduced to quantify how fresh this information is within communication systems. Specifically, AoI measures the time that has elapsed since the last received update was generated at the source, allowing to evaluate how current the information is at the receiver's end.

This metric has garnered significant attention as the demand for real-time data processing and timely information delivery have become critical in various applications, such as, sensor networks, autonomous systems, and the internet of things (IoT) in 6G systems. For instance, in the context of autonomous vehicles, to prevent collisions and enhance overall traffic safety, it is essential that multiple vehicles receive real-time data about the positions of other vehicles. AoI is particularly important in systems where the relevance of information diminishes over time, such as

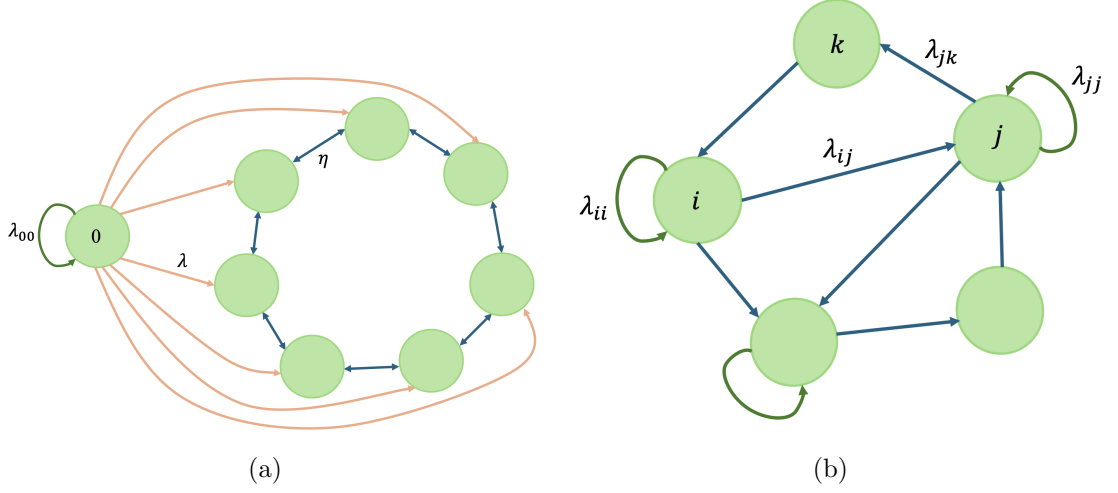


Figure 7.1: (a) An example single source (single sensing node) network, and (b) an example multiple sensing node network.

monitoring networks, where outdated data can lead to incorrect decisions or system inefficiencies.

Another significant advancement in this domain is the concept of *version age*, where freshness of information is measured by a discrete quantity, i.e., version of the last update, instead of the absolute time [118–120]. Every time a new reading from the source becomes available to a node (that is the “source node” in the current state-of-the-art), the version age of all nodes in the network is incremented with the exception of the source node. The performance of version AoI has been analyzed across different network architectures, such as fully connected networks [118], ring networks [118, 121], and grid networks [122]; see [123] for a recent survey. However, despite these advances, the current literature only deals with a single source node in the network (as shown in Fig. 7.1(a)) which generates all the updates according to a Poisson process.

In this work, we build on a novel interpretation of the source node in the

current literature, and introduce a new model. The so-called source node can be understood as a node of the network which in fact samples the information source, possibly with continuous distribution, such as the temperature or location, at times that follow a Poisson point process (PPP). Given this interpretation, one can immediately recognize that there is no need to limit such a node with sensing capability to be unique. Indeed, we consider having *multiple views* of the information source via a subset of the nodes in the network (see Fig. 7.1(b)), which we call the *sensing nodes*.

For instance, consider a fleet of drones deployed to monitor a rapidly changing environment, such as during a disaster response operation. Each drone is responsible for gathering critical data, such as temperature, air quality, or the location of people in need of rescue, within the same area and relaying the most recent information back to a central user or system. In this work, we will consider the scenario where there is a single underlying source, but there are multiple sensing nodes. Therefore, we will make a distinction between a source node in the prior art, where the underlying random source and the unique sensing node were both called the “source node”, here we will call the underlying random source as the source and the nodes which sample/sense it the sensing nodes.

In comparison between single- and multiple-view scenarios, we note that in a multiple-view scenario: (i) the AoI for a particular sensing node might be non-zero because another sensing node could have more up-to-date information, and (ii) the sensing node can also receive updates from other nodes of the network whether they are sensing or not-a-sensing nodes. We calculate the average version AoI for a

network with multiple sensing nodes and demonstrate that the scaling behavior of the version AoI, in two well-known extreme cases of the system networks, i.e., fully connected networks and ring networks, is consistent with that observed in these networks with single-views under similar network configurations.

The scenarios with multiple source updates have also been considered, e.g., see [27, 124]. However, analysis of the multiple-source scenarios is quite different from the treatment of the network with multiple views of a single source that we aim to analyze in this work.

The rest of the work is organized as follows. In Section 7.2, we introduce the preliminaries, including definitions of version and version AoI for a multiple sensing nodes structure. In Section 7.3, we present our proposed distributed sensing structure. Specifically, in Section 7.3.1, we derive the average version AoI for the proposed scheme. Then, in Sections 7.3.2, 7.3.3, and 7.3.4, we analyze the scaling behavior of the version AoI for three different network structures: ring networks, line networks, and fully connected networks, respectively.

7.2 Preliminaries

We consider a network of n nodes labelled from 1 to n , i.e., $[n] = \{1, 2, \dots, n\}$. In this network, e.g., a sensor network, an accurate clock between the nodes may not be available, and consequently, the timeline at each node is measured by version updates. The set of all nodes is denoted by $\mathcal{N}$. A subset of the nodes $I \subset \mathcal{N}$ sense a time-varying source. To incorporate most physical phenomena such as location,

temperature, air quality, and so on, we consider a source with a continuous distribution in general. A view of the source is available to node i , $i \in I$, by direct sampling of the source at times that is determined by a Poisson point process (PPP) of rate λ_{ii} denoted in short hand notation as $\text{PPP}(\lambda_{ii})$. Any update from a node $i \in I$ constitutes a new version of the information that is available to the whole network. The information at node $i \in \mathcal{N}$ is shared with node j in the network at times that are determined by a $\text{PPP}(\lambda_{ij})$. We note that the nodes in set I may also receive an update of the source through the network from an adjacent node. By definition, the nodes and their updates through the $\text{PPP}(\lambda_{ij})$ for $i, j \in \mathcal{N}$ form a network that can be represented by a directed graph where the weight of the edge from node i to node j is given by λ_{ij} . For simplicity of derivation and presentation of the results, in this work, we restrict our attention to the case that all PPPs are independent.

Let us define the current version of the source sample available at node i and time t by $V_i(t)$. We note that $V_i(t)$ could merely be an abstract value that is used for calculation of the version AoI and it is not necessarily known to the nodes. However, one can consider a shared memory that is accessible to the nodes which contains the latest version of the source sample known to the whole network. In which case, when a sensing node performs a new sampling of the source, it increments this shared memory and marks its new packet with a time stamp and the new version.

Due to the stochastic nature of the PPP used in sampling at the nodes in I and communications between the nodes in the network, $V_i(t)$ is also a random process which possibly has discontinuities only at the points that one of the $\text{PPP}(\lambda_{ij})$, $i, j \in \mathcal{N}$ occurs, where at such a point, $V_i(t^-)$ and $V_i(t)$ indicate the value of V_i

before and after the possible update, respectively. We assume that the nodes use gossip to send their updates to their neighbors. The node i sends the update to the node j when the event corresponding to $\text{PPP}(\lambda_{ij})$ happens. The update from a node is composed of the current version of the information at that node and its corresponding time stamp. The time stamp is added to the information packet along with the source sample at each sensing node, i.e., the nodes $i \in I$. Each node only keeps the latest view of the source available at that node. Hence, the receiving node j updates its version to the latest available version, i.e., $V_i(t^-)$ is replaced by $V_i(t) = \max(V_i(t^-), V_j(t))$. To clearly distinguish the value of $V_i(t)$ before and after update, we use the notation $V_i(t^-)$ to denote the available version to the node i right before the update from node j is applied. Let $\pi_{ij}(Y(t))$ denote the operator which provides the value of a random process $Y(t)$ at time t after the $\text{PPP}(\lambda_{ij})$ has occurred. Using this notation, we write $V_i(t) = \pi_{ij}(V_i(t)) = \max(V_i(t^-), V_j(t))$. Without loss of generality, we can assume that the two PPPs do not happen at exactly the same time and differ by at least a small ϵ time difference due to the continuous (exponential) distribution between the points and the discrete nature of the PPP.

The version for a subset of the nodes $S \subset \mathcal{N}$ at time t is defined as $V_S(t) = \max_{i \in S} V_i(t)$. Formally the version of the network at time t is defined as $V_{\mathcal{N}}(t)$ which is the best (highest) version that can be available at any node in the network. We define the AoI at node i as $X_i(t) = V_{\mathcal{N}}(t) - V_i(t)$. Hence, the version age of information (version AoI) at a subset $S \subset \mathcal{N}$ may be written as $X_S(t) = \min_{i \in S} X_i(t)$.

7.3 Main Result

In this section, we borrow the results from well-developed theory of stochastic hybrid and switched systems (SHS). This theory has applications mainly in modern control theory, where the control system comprises a hybrid of continuous and discrete time components, inputs (control actuators), and outputs. In this work, we aim to characterize the average version AoI $v_S(t) = \mathbb{E}[X_S(t)]$, particularly, for an arbitrary node i , i.e., $S = \{i\}$. The application in our problem is immediate, as we are interested in average (in continuous time) version age of information (which takes discrete values corresponding to discrete version updates) and Poisson point process which happens in discrete time intervals with exponential distribution (continuous) between its sample points.

7.3.1 Derivation of Average AoI

For a piece-wise linear SHS system where the derivative of the measurement function $Y(t)$ only has discontinuities as a result of a jump through a point process in time, the measurement function satisfies Dynkin's formula [125],

$$\frac{d}{dt}\mathbb{E}[Y(t)] = \mathbb{E}\left[\sum_{(i,j)\in\mathcal{L}}\lambda_{ij}(\pi_{ij}(Y(t)) - Y(t^-))\right], \quad (7.1)$$

where $Y(t^-)$ denotes the value of Y right before the update at time t . For $Y(t) = X_S(t)$, we have,

$$\pi_{ij}(X_S(t)) = \begin{cases} X_S(t^-) + 1, & j = i, j \in I, j \notin S, \\ 0, & j = i, j \in I, j \in S, \\ X_S(t^-), & j = i, j \notin I, \\ X_S(t^-), & j \neq i, j \notin S, \\ X_S(t^-), & j \neq i, j \in S, i \in S, \\ X_{S \cup \{i\}}(t^-), & j \neq i, j \in S, i \notin S. \end{cases} \quad (7.2)$$

Combining (7.2) for $Y(t) = X_S(t)$ with (7.1), the right hand side of (7.1) can be rewritten as,

$$\frac{d}{dt} \mathbb{E}[X_S] = \mathbb{E} \left[\sum_{j \notin S, j \in I} \lambda_{jj} (X_S + 1 - X_S) - \sum_{j \in S, j \in I} \lambda_{jj} X_S + \sum_{j \in S, i \notin S, j \neq i} \lambda_{ij} (X_{S \cup \{i\}} - X_S) \right]. \quad (7.3)$$

We note that $\lambda_{jj} = 0$ where $j \notin I$, hence, the corresponding term in calculation of (7.1) does not appear. Also, if $j \in S$ and $i \notin S$ then trivially $j \neq i$, hence, the condition $j \in S, i \notin S, j \neq i$ can be reduced to $j \in S, i \notin S$. By setting $\frac{d}{dt} \mathbb{E}[X_S] = 0$ in (7.3), the stationary point must satisfy

$$\sum_{j \notin S, j \in I} \lambda_{jj} - \sum_{j \in S, j \in I} \lambda_{jj} \mathbb{E}[X_S] + \sum_{j \in S, i \notin S, j \neq i} \lambda_{ij} (\mathbb{E}[X_{S \cup \{i\}}] - \mathbb{E}[X_S]) = 0, \quad (7.4)$$

and by setting $v_S = \mathbb{E}[X_S]$ and $v_{S \cup \{i\}} = \mathbb{E}[X_{S \cup \{i\}}]$ after rearranging, we have the following result.

Proposition 7.1 *The average version AoI for a subset S in a network with sensing nodes $i \in I$ that sample a view of the source at times characterized by $PPP(\lambda_{ii})$, and update received at node j from node i at the times given by $PPP(\lambda_{ij})$, $i \neq j$ is given by*

$$v_S(t) = \frac{\sum_{j \notin S, j \in I} \lambda_{jj} + \sum_{j \in S, i \notin S} \lambda_{ij} v_{S \cup \{i\}}}{\sum_{j \in S, j \in I} \lambda_{jj} + \sum_{j \in S, i \notin S} \lambda_{ij}}. \quad (7.5)$$

Here, a few remarks are in order: First, we note that the average renewal rate for the whole network is given by $\sum_{j \in I} \lambda_{jj}$. In a network with a single sensing node $I = \{0\}$, the average renewal rate is λ_{00} . The formulation in [118] derives the average AoI in a network with single source node 0 with Poisson distribution λ_{00} and n nodes, $i \in [n]$. If we interpret the renewal of the view of the node 0 as the arrival (or generation) of the new information by node 0 directly as a source, the average AoI given by (7.5) for the special case of single view of the source reduces to the result of [118, Thm. 1], in which case (7.5) reduces to,

$$v_S(t) = \frac{\lambda_{00} + \sum_{j \in S, i \notin S} \lambda_{ij} v_{S \cup \{i\}}}{\sum_{j \in S} \lambda_{0j} + \sum_{j \in S, i \notin S} \lambda_{ij}}. \quad (7.6)$$

Second, the minimum average version AoI taken over all possible nodes of the network, $v_{\{i\}}$, $i \in \mathcal{N}$ achieves its minimum, that is equal to zero for $v_I = 0$, in the case of a network with single view at its unique viewing point $|I| = 1$. However,

the same is not true for the network with multiple views of the source. Example 7.1 reveals a sharp contrast between a network with a single view of the source versus a network with multiple views of the source.

Example 7.1 *Consider the network in Fig. 7.2. Let us use the short hand notation v_{ijk} by directly specifying the elements of the set $S = \{i, j, k\}$ in the subscript to denote v_S . We have $v_{012} = 0$, and*

$$v_2 = \frac{\lambda_{00} + \lambda_{11} + \lambda_{12}v_{12} + \lambda_{02}v_{02}}{\lambda_{12} + \lambda_{02}}, \quad (7.7)$$

$$v_{12} = \frac{\lambda_{00} + \lambda_{02}v_{012}}{\lambda_{11} + \lambda_{02}}, \quad (7.8)$$

$$v_{02} = \frac{\lambda_{11} + \lambda_{12}v_{012}}{\lambda_{00} + \lambda_{12}}. \quad (7.9)$$

Hence, the average version AoI for each node is given by

$$\begin{aligned} v_2 &= \frac{\lambda_{00} + \lambda_{11}}{\lambda_{12} + \lambda_{02}} + \frac{\lambda_{00}\lambda_{12}}{(\lambda_{12} + \lambda_{02})(\lambda_{11} + \lambda_{02})} \\ &\quad + \frac{\lambda_{11}\lambda_{12}}{(\lambda_{12} + \lambda_{02})(\lambda_{00} + \lambda_{12})}, \end{aligned} \quad (7.10)$$

$$v_1 = \frac{\lambda_{00}}{\lambda_{11}}, \quad (7.11)$$

$$v_0 = \frac{\lambda_{11}}{\lambda_{00}}. \quad (7.12)$$

If we choose $\lambda_{11} = \lambda_{00} = 1$ and $\lambda_{12} = \lambda_{02} = 10$, in Example 7.1, we obtain $v_0 = v_1 = 1$, and $v_2 = 0.19$. Thus, Example 7.1 shows that the average AoI in neither of the viewing points (sensing nodes) is zero. It also shows that the node

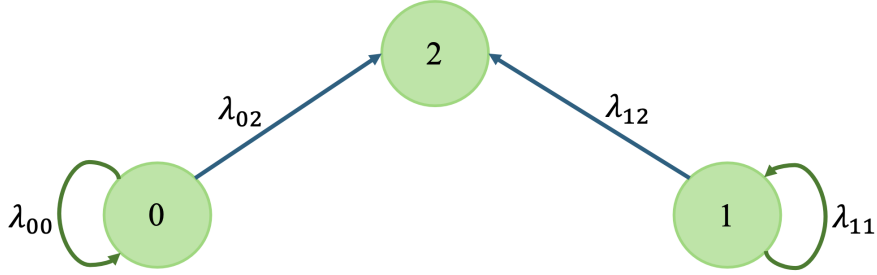


Figure 7.2: An example multi-view network with multiple sensing nodes. Nodes 0 and 1 are sensing nodes.

with the minimum AoI is not necessarily one of the viewing points. We next state the following proposition.

Proposition 7.2 *In a network with multiple views, i.e., $|I| > 1$ and PPP of finite rate, the average AoI for any node is always nonzero.*

Even though Proposition 7.2 could provide a pessimistic observation of the network with multiple views, we note that the average version AoI can indeed be a constant, i.e., $O(1)$ even as the network size grows large.

The average AoI in a network provides a way to compare the freshness of information at different parts of the network. In a network with a single view of the source, the average AoI of the node indeed provides additional insight when the transition probabilities between the nodes change, i.e., when the network topology evolves. However, if the sampling rate for the view of the source changes from an instance of the network to another, the average AoI of a nodes does not directly provide a fair comparison to the freshness of the information at this node. Moreover, in comparing two networks with different numbers of viewing points, the average AoI is a fair measure if the total renewal rate of the network in comparison is a constant.

Hence, we propose to use the Normalize aVerage AoI, i.e., NoVAI, defined as the ratio of the average AoI to the average renewal rate for the view of the source by all nodes in the network as a fair measure. NoVAI can also provide a fair measure in comparison between two networks, when the number of sensing nodes changes in the network, or when the sampling rates of the sensing nodes change. In the following, we investigate the scaling of the average version AoI in the network for various important network typologies, namely, ring, line, and fully connected networks. We present the results for the average version AoI of any single node in the network, as in [118]. Other measures such as the average version AoI taken over all single nodes of the network has also been considered [126] which does not necessarily enforce the scaling on any single node.

7.3.2 Ring Network

Consider a network of n nodes labeled as $i \in [n]$. Using $\text{PPP}(\lambda)$, the node i receives a renewal sample of the information source and keeps the last version and its time stamp; see Fig. 7.3(a). The node i also receives updates from adjacent nodes $(i-1) \bmod n$ and $(i+1) \bmod n$ at a rate governed by $\text{PPP}(\eta)$ and keeps the version with the latest time stamp. Let v_k denote the average AoI for a connected subset of nodes of size k , i.e., the nodes with indices $j+1 \bmod n, j+2 \bmod n, \dots, j+k \bmod n$. Due to the symmetry of the network, $v_k, k \in [n]$ is well defined as any subset of k connected nodes are equal. Using (7.5) we can write v_k in terms of v_{k+1} as

$$v_k = \frac{(n-k)\lambda + 2\eta v_{k+1}}{k\lambda + 2\eta}. \quad (7.13)$$

Defining $\beta = 2\eta/\lambda$, we have

$$v_k = \frac{(n-k)}{k+\beta} + \frac{\beta}{k+\beta}v_{k+1}. \quad (7.14)$$

Using this recursion, we find the AoI for a single node v_1 as

$$v_1 = \frac{(n-1)}{1+\beta} + \frac{\beta}{1+\beta} \left(\frac{(n-2)}{2+\beta} + \frac{\beta}{2+\beta} \left(\dots \right. \right. \quad (7.15)$$

$$= \Gamma(1+\beta) \sum_{k=1}^{n-1} \frac{n-k}{\Gamma(1+k+\beta)}. \quad (7.16)$$

For $\lambda = \lambda_{00}/n$, $\eta = \eta_{00}/2$, and $\lambda_{00} = \eta_{00}$, we have $\beta = n$, and using the fact that $v_n = 0$, we have

$$v_1 = \sum_{k=1}^{n-1} \frac{n^{k-1}(n-k)}{\prod_{i=1}^k (n+i)} = n! \sum_{k=1}^{n-1} \frac{n^{k-1}(n-k)}{(n+k)!} \quad (7.17)$$

$$\leq \frac{n!}{n^n} \sum_{k=1}^{n-1} \frac{n^{n+k}}{(n+k)!} \leq \frac{n!}{n^n} e^n. \quad (7.18)$$

Using Sterling's formula for large n , $n! \sim \sqrt{2\pi n} \frac{n^n}{e^n}$, we conclude that $v_1 \sim O(\sqrt{n})$.

Even though we derived this scaling results for the network with distributed views, following similar derivation, we can derive the same scaling results for the network with an additional node 0 connected to the n nodes on the ring with the link following the activation through PPP(λ/n). This scaling of $O(\sqrt{n})$ was conjectured in [118] as experimentally observed in [118, Fig. 4], and mathematically proved to be true in [121, Lemma 2] for the single view case. The above derivation proves it for the multi-view case.

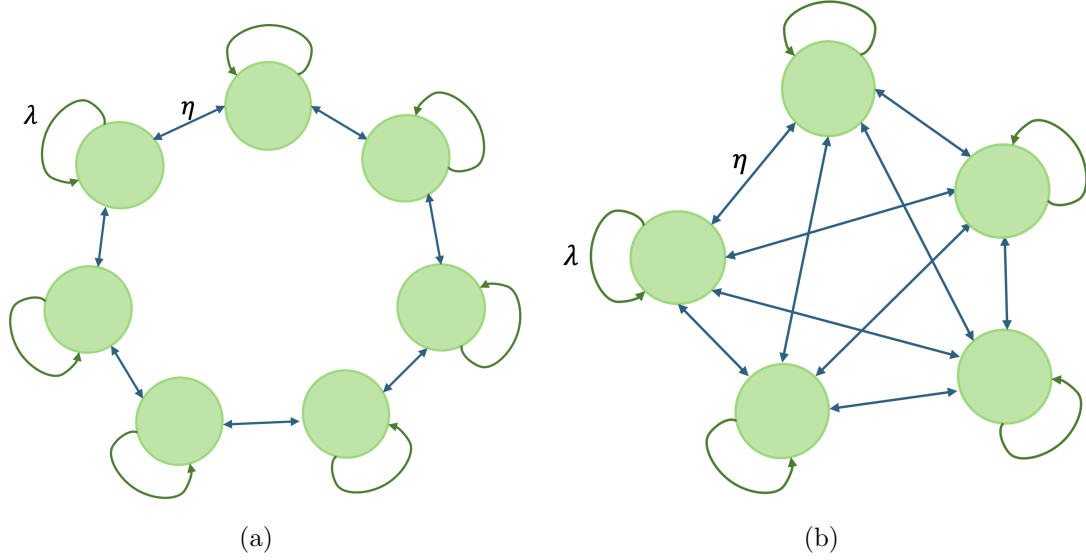


Figure 7.3: (a) Ring network, and (b) fully connected network, where all nodes are sensing nodes.

7.3.3 Line Network

The line network is composed of n nodes labeled as $i \in [n]$ where the node $1 < i < n$ only receives updates from adjacent nodes $(i - 1)$ and $(i + 1)$ and node 1 and n only receive updates from node 2 and $n - 1$, respectively, at a rate governed by $\text{PPP}(\eta)$ and keeps the version with the latest time stamp. Consider a general case where there are only q sensing nodes which sample the source corresponding to $\text{PPP}(\lambda)$. Let d be the maximum number of consecutive nodes where none of the i is a sensing node, i.e., we have $i \notin I$ for any node i in this set of consecutive nodes.

Let $v_{j,k}$ denote the average version AoI for a connected subset of nodes of size k which starts with the node index $j + 1$, i.e., the nodes with indices from the set $\mathcal{S}_{j,k} = \{(j + 1) \bmod n, (j + 2) \bmod n, \dots, (j + k) \bmod n\}$. Since the position of the sampling points is not known, the value of v_k is indeed a function of j , however,

we can define an upper bound on $v_{j,k}$ for all k as $\bar{v}_k$ which is obviously not less than $\max_j v_{j,k}$, i.e., maximum of average AoI over all possible connected subset of nodes. In the following, we derive recursive expression for $\bar{v}_k$. We use the following properties to reduce the number of recursive expression for $\bar{v}_k$. (i) We note that by starting from the largest set, i.e., $k = n$, we have $\bar{v}_n = \bar{v}_{\mathcal{N}} = 0$. (ii) For any $k < n$, we note that $\bar{v}_k$ only depends on $v_{j,k+1}$ and if in any such expression $v_{j,k+1}$ is replaced by $\bar{v}_{k+1}$ then $\bar{v}_k$ is not reduced and still is an upper bound for $v_{j,k}$ for all j . (iii) The worst case update for any of such expressions for update from the $v_{j,k}$ is also when it comes from a single node with PPP(η). This statement is intuitively true, and the formal proof relies on the fact that for any $T \subset S \subset \mathcal{N}$, we have $v_T \geq v_S$, and therefore $\bar{v}_{k+1} < \bar{v}_k$. (iv) Consider $v_{j,k}$ for any j, k . There is $n - k$ nodes outside $\mathcal{S}_{j,k}$ out of which at least $\lfloor \frac{n-k-q}{d+1} \rfloor^+$ nodes are sensing nodes, where $\lfloor x \rfloor^+$ is non-negative part of the floor $x \in \mathbb{R}$. For example, when $k = n - 2q$, there could be exactly q non-sensing nodes in each side of the set $\mathcal{S}_{j,k}$, but if $k = n - 2q - 1$ there is at least one sensing node in the set $\mathcal{S}_{j,k}$. Hence, using (7.5) and the above reduction technique, we have

$$\bar{v}_k = \frac{(q - \lfloor \frac{n-k-q}{d+1} \rfloor^+) \lambda + \eta \bar{v}_{k+1}}{\lfloor \frac{n-k-q}{d+1} \rfloor^+ \lambda + \eta}, \quad k \geq n - q - md, \quad (7.19)$$

$$\bar{v}_k = \frac{k \lambda + \eta \bar{v}_{k+1}}{(q - k) \lambda + \eta}, \quad k < n - q - md, \quad (7.20)$$

where $n - q = dm + p$, $p < d$ for some positive integers m, p . Using definition of d and q it is immediate that $(d + 1)q \geq n$, otherwise it contradicts the fact that d is the maximum number of (not sensing) nodes between any two consecutive sensing

nodes. The worst case happens when $(d+1)q \geq n$ which means that $m = q$ and $p = 0$. Define $\beta = \eta/\lambda$, the expression for $\bar{v}_1$ is given by

$$\bar{v}_1 = \frac{2qd}{\beta} + \sum_{j=1}^{m-1} \sum_{i=1}^{d+1} \frac{(q-j)\beta^{(j-1)(d+1)+i-1}}{(1+\beta)^i \left(\prod_{l=1}^{j-1} (l+\beta) \right)^{d+1}}. \quad (7.21)$$

Using geometric series sum, we have

$$\bar{v}_1 \leq \frac{2qd}{\beta} + q + \frac{q}{\beta} \sum_{j=1}^{m-1} \left(\frac{\beta^j}{\prod_{l=1}^j (l+\beta)} \right)^{d+1} \quad (7.22)$$

$$= \frac{2qd}{\beta} + q + \frac{q\Gamma(1+\beta)^{(d+1)}}{\beta^{\beta(d+1)+1}} \sum_{j=1}^{m-1} \left(\frac{\beta^{(j+\beta)}}{\Gamma(1+j+\beta)} \right)^{d+1}. \quad (7.23)$$

Using gamma function approximation $\Gamma(x) = x^x \sqrt{(2\pi x)}/e^x$, and using the fact that λ should be scaled down to λ/q , we find that the upper bound scales as $\bar{v}_1 \sim O(d+q)$.

It is immediate that by converting line network to a ring by adding a link between the two ends of the line, the average AoI for any subset would not be worse, hence the scaling results in the line network will hold true in the ring network.

The asymptotic behavior predicted for the upper bound reveals that in order to achieve $O(\sqrt{n})$ bound on the average AoI of any single node in the line network, it is enough to have $q \sim O(\sqrt{n})$ and $d \sim O(\sqrt{n})$. We note that the asymptotical bound is not necessarily tight in all cases and may even fall short of some simple bounds. Nonetheless, the achievement of $O(\sqrt{n})$ for average AoI of any single node in the network with only $q \sim O(\sqrt{n})$ at any position is quite interesting, while the best known result for a ring network requires $q \sim O(n)$ [118]. We note that scaling of average AoI is also a function of scaling of d and the condition $d \sim O(\sqrt{n})$ is

crucial to achieve $O(\sqrt{n})$ scaling for the average version AoI.

Two notes are in order here: First, the bound is not a function of particular structure or pattern for the placement of the sensing nodes. Second, this bound may be used to emphasize the importance of the distributed or multi-view of a source. More specifically, if a single node, say in the middle of the line network or any node on a ring, samples the source corresponding to $\text{PPP}(\lambda_0)$, the average AoI per node scales as $O(n)$, while by using distributed sensing with $q = \lceil \sqrt{n} \rceil$ that sample the source corresponding to $\text{PPP}(\lambda_0/q)$ collectively at the same rate λ_0 , the average AoI per node scales as $O(\sqrt{n})$. Comparison of the average AoI scaling per node for various scaling of d is shown in Fig. 7.5 and discussed in the following.

Fig. 7.4 illustrates the numerical evaluation of the upper bound $\bar{v}_1$ in (7.19)-(7.20) for different scaling of the q and d . We observe that if the number of sensing nodes q scales as $O(\sqrt{n})$, $O(\log(n))$, and $O(n^{1/5})$ and d accordingly scales as $O(n/q)$, i.e., $O(\sqrt{n})$, $O(n/\log(n))$, and $O(n^{4/5})$, the corresponding curve scales as $O(d+q)$, i.e., $O(\sqrt{n})$, $O(n/\log(n))$, and $O(n^{4/5})$, respectively. Moreover, we note that $q(d+1) \geq n$ forces $d \geq -1 + n/q$ and on the other hand we have $n \geq q+d$ which provides an upper bound on q as $d \leq n - q$ which means that $O(qd)$ can be indeed larger than $O(n)$. Fig. 7.5 illustrates the bound (7.19)-(7.20) for different values of d and q and the numerical values show that the scaling for large n is $O(d+q)$. In particular, even though $q \sim O(\sqrt{n})$, as d scales with $O(n^{2/3})$, $O(n^{3/4})$, and $O(n^{4/5})$, the upper bound on the average AoI per a single node, i.e., $\bar{v}_1$ scales as $O(n^{2/3})$, $O(n^{3/4})$, and $O(n^{4/5})$, respectively.

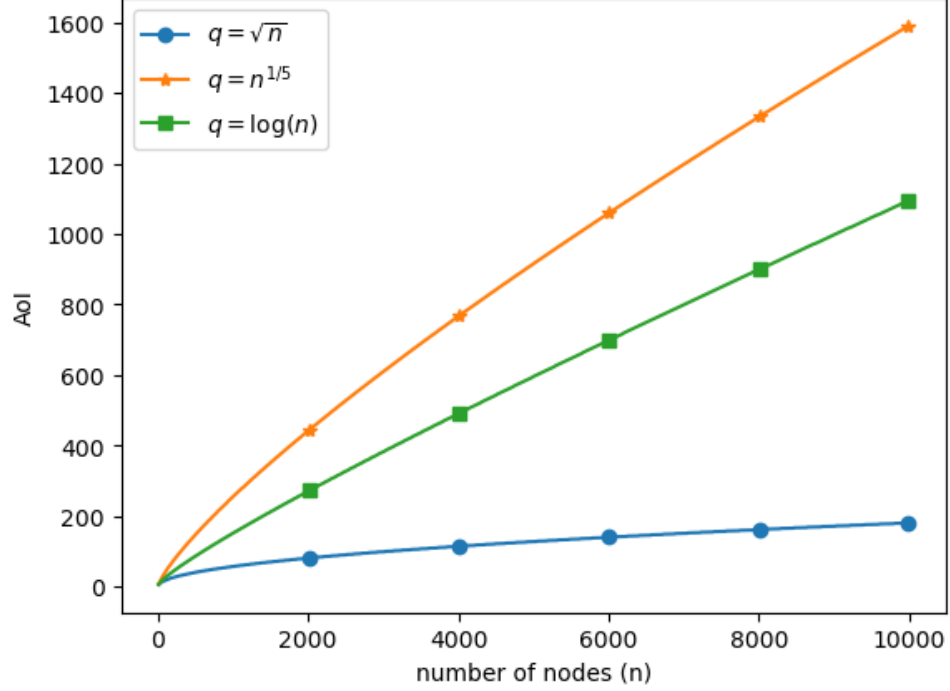


Figure 7.4: Effect of changing q in average AoI in line/ring network.

7.3.4 Fully Connected Network

Consider a fully connected network of n nodes labeled as $i \in [n]$. As shown in Fig. 7.3(b), each node samples the information source using $\text{PPP}(\lambda)$ and only keeps the last view of sample from the information source as the current version with its time stamp. Each node receives the version available at another node governed by $\text{PPP}(\eta)$ and only keeps the version with the latest time stamp. We note that in general η and λ could indeed be a function of n , e.g., $\eta = \eta_0/n$, $\lambda = \lambda_0/n$ and so on. Let v_k denote the average version AoI for a subset of nodes of size k . Due to the symmetry, v_k , $k \in [n]$ is well defined as any subset of k nodes are equal. Using

(7.5) we write v_k in terms of v_{k+1} as

$$v_k = \frac{(n-k)\lambda + (n-k)k\eta v_{k+1}}{k\lambda + (n-k)k\eta}. \quad (7.24)$$

Define $\alpha = \lambda/\eta$, we have

$$v_k = \frac{(n-k)}{n-k+\alpha} \left(\frac{\alpha}{k} + v_{k+1} \right). \quad (7.25)$$

Using the above recursion, we can find the AoI for a single node v_1 as

$$v_1 = \frac{(n-1)}{n-1+\alpha} \left(\frac{\alpha}{1} + \frac{(n-2)}{n-2+\alpha} \left(\frac{\alpha}{2} + \cdots + \frac{1}{1+\alpha} \left(\frac{\alpha}{n-1} + v_n \right) \cdots \right) \right) \quad (7.26)$$

We note that $v_n = 0$, hence for $\alpha = 1$, i.e., $\lambda = \eta$ we have

$$v_1 = \frac{(n-1)}{n} \left(1 + \frac{(n-2)}{n-1} \left(\frac{1}{2} + \cdots + \frac{1}{2} \left(\frac{1}{n-1} \right) \right) \right) \quad (7.27)$$

$$= \sum_{k=1}^{n-1} \frac{n-k}{nk} \leq \log(n) \quad (7.28)$$

This means that if both λ and η scale as $O(\frac{1}{n})$, e.g., $\lambda = \lambda_{00}/n$ and $\eta = \eta_{00}/n$, the AoI for a single node in the network scale as $O(\log(n))$. Note that the same scaling is true if, e.g., $\eta = \frac{\eta_{00}}{n-1}$. This means that for fully connected network, the scaling of AoI using distributed sampling at the nodes is the same as that of the case where the version updates from a single view node are provided to the nodes of the network at the similar rate as derived in [118]. In other words, there is no loss in terms of AoI if we perform distributed sampling. Note that the average renewal

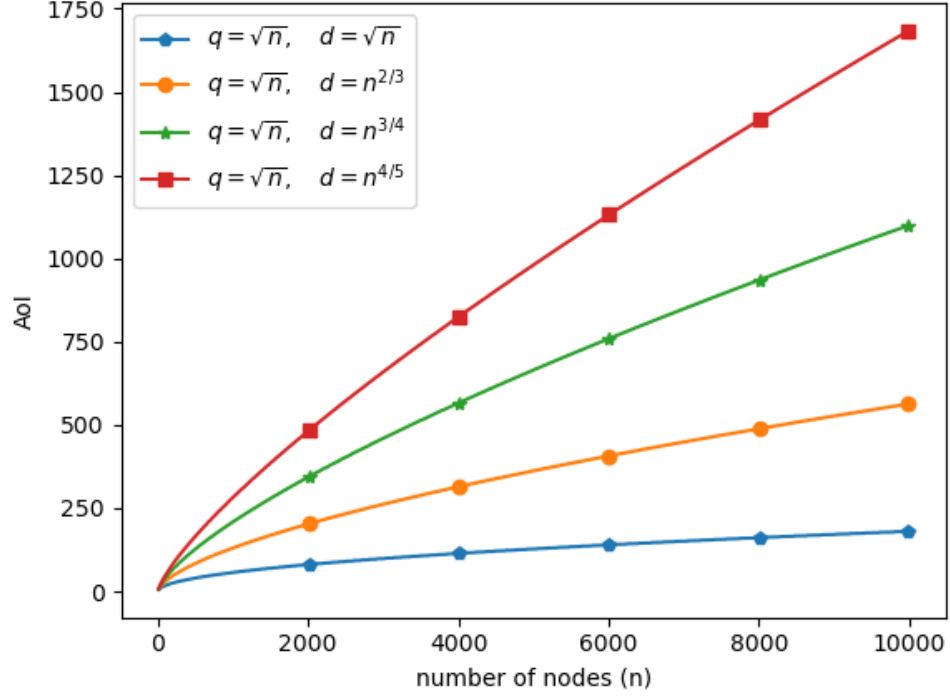


Figure 7.5: Effect of changing d in average AoI in line/ring network.

of the information in the network in both cases is equal to λ_{00} . However, we note that if for some constants λ_{00} and η_{00} , if $\lambda = \lambda_{00}/n$ and $\eta = \eta_{00}$ then the average AoI scales as $O(\frac{\log(n)}{n})$. Also, if $\lambda = \lambda_{00}/n$ and $\eta = \eta_{00}/n^2$, the average AoI scales as $O(n \log(n))$. These results are true for either of case of a network with multiple view or single view as in [118]. This emphasizes the fact that the scaling of AoI varies as the scaling behavior of the mean of PPP for the links of the network varies. Hence, even though not stated directly in the theorem statement, the strong result of [118, Thm. 1] should be carefully interpreted only for the case that the mean of the PPP for the link of the network is of order $o(1/n)$.

7.4 Conclusion

We introduced distributed sensing for version AoI in networks. We derived iterative expressions for the average AoI for any subset of the nodes. We established the scaling of the average AoI as the number of nodes in the network grows for multiple fundamental network topologies.

CHAPTER 8

Evaluating and Improving Multi-Modal RAG Hallucination

8.1 Introduction

Generative models have seen significant improvements with recent advancements in large language models (LLMs) [127]. Although the generated responses often reach human-like quality, hallucinations, generating incorrect or irrelevant responses, remain an issue [128]. This problem is particularly concerning for applications where accuracy is critical, such as in medical evaluations, processing insurance claims, and autonomous decision-making. The hallucination issue also persists in vision-language models (VLMs) [129–132], which process both images and user queries to generate text responses. Several robust VLMs, such as LLaVA [129], InstructBLIP [130], and VILA [131], exist. However, these models sometimes produce incorrect responses based on the provided images and user queries.

On the other hand, retrieval-augmented generation (RAG) [29] offers a promising solution to improve the relevance of responses generated by LLMs. RAG systems enhance LLMs by incorporating external knowledge, on which the user seeks to base their responses. In a RAG system, external knowledge such as enterprise data is

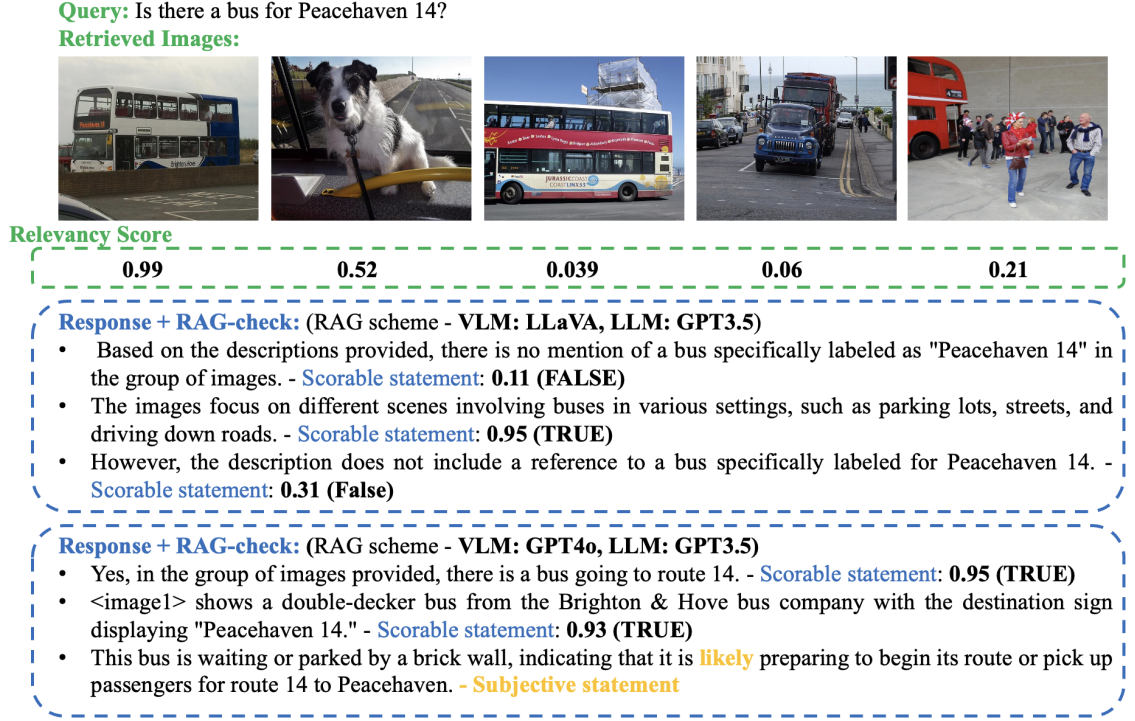


Figure 8.1: RAG-check example for two different multi-modal RAG schemes.

stored in a database. When a user submits a query, the RAG system retrieves a few relevant and similar pieces of data from the database. The LLM then generates a coherent response based on this curated information. This approach reduces hallucinations by constraining the LLM to generate responses grounded in the provided external knowledge, thereby increasing accuracy and relevance. However, in addition to the possible hallucinations in the response, the RAG scheme introduces some new sources of hallucinations during the retrieval process including the selection of the information pieces and the context generation. During the retrieval process, depending on the selection algorithm, RAG selects a few, say k , entries from the database that have the highest similarity to the user query. A common approach is to select the k entries whose embeddings have the highest cosine similarity to the query embedding, referred to as the top- k entries. However, this method of

selection may not always retrieve the most relevant data for the user query. The wrong selection can result in an incorrect response since the LLM would never see the correct information. We refer to this as *selection-hallucination*. Another source of hallucination is during context generation in the RAG scheme. In particular, multi-modal RAG systems process each selected piece of information to text and generate a text-based context by concatenating the text-based context for all the selected pieces. The text-based context is then provided to the LLM in the final stage of the RAG system along with the query to produce the response. For example, if the selected piece is an image, VLM may be used, say with a prompt “Describe the image” (DTI) to narrate the images into the text, and it is well-known that VLMs may hallucinate and produce wrong context in this case. We refer to this as *context-generation-hallucination*. In the final stage, LLM may also hallucinate when producing the response which is referred to as *response-generation-hallucination*.

Prior works [133–135] provide means to evaluate LLMs and measure the correctness of text-based responses by LLM given a text query. FactScore [133] demonstrates that breaking long statements into fine-grained atomic statements can improve the identification of hallucinations. This method allows for more precise verification of each individual statement against the source material. Lookback Lens [134] detects context hallucination by analyzing the attention scores of the corresponding LLM. It calculates a ratio attention scores between query tokens and response tokens to identify parts of the response that may not be adequately supported by the context. MARS [135] builds on the idea that some parts of a statement are more crucial than others in determining hallucination. The work highlights such impor-

tant part by assigning more weight to these significant parts, thereby improving the detection of inaccuracies by focusing on the most relevant segments. Despite these advancements, a few works have specifically addressed hallucination detection in text-based RAG. Among these, RAGAS [30] and LlamaIndex evaluation stand out. RAGAS focuses on evaluating the accuracy and relevance of responses generated by RAG systems. It assesses how well the retrieved documents support the generated response and whether the information is correctly incorporated. RAGAS typically involves human evaluators who rate the coherence and factual accuracy of the responses. LlamaIndex relies on GPT4 [127] to evaluate the faithfulness of the generated response and the relevancy of retrieved documents. Note that both RAGAS and LlamaIndex are designed for text-based queries and datasets.

On the other hand, some works have focused on detecting hallucination in VLMs [136–138]. The work in [136] presents a model inspired by InstructBLIP to detect hallucinations occurring in single-image scenarios with user queries. This approach leverages the capabilities of InstructBLIP to scrutinize the generated responses for inaccuracies related to the provided image and query. FaithScore [137] is another tool trained to identify the correctness of VLM-generated statements with an image input. FaithScore first extracts a comprehensive list of atomic facts from the generated response, then verifies the accuracy of these fine-grained atomic facts against the input image. However, to the best of our knowledge, there are no existing works that provide hallucination scores specifically for multi-modal RAG systems, where the contexts include multiple pieces of multi-modal data (say multiple images) which is retrieved from a database and then the inference is performed

on such retrieved multi-modal context. Multi-modal RAG systems present unique challenges, as they require integrating information from various visual sources along with the text to generate coherent and accurate responses.

Our goal is to develop RAG-check, a method to evaluate the performance of multi-modal RAG schemes. RAG-check comprises three main components: (i) First, we design and train a neural network structure that takes the selected pieces of data by the RAG scheme by tapping into the internal components of a RAG system and producing a relevancy score (RS) between each selected piece and the query. For example, when an image (or text) is selected, RS evaluates how relevant the retrieved image (or text) is to the user query. By assigning a relevancy score to each image, we can determine how well the visual data aligns with the user’s intent and the specific information requested. (ii) Second, we design an algorithm to partition the output response by the RAG into segments, namely *spans*, and categorize each span. The *span* may be defined as statements, phrases, etc. Some spans may not be scorable, e.g., if it is based on personal opinions or feelings rather than on facts. Alternatively, a span is not scorable if it states an analysis say in a conditional statement, or expresses the uncertainty such as the possibility or probability of something being true. We label such statements as “subjective” which have been also referred to as “analysis” statement type in the prior-art [136]. A span that is not “subjective”, i.e., it is scorable, is labeled as “objective”. (iii) Third, we design and train a neural network structure which assesses the correctness of each objective span in the view of the *raw context* defined as the selected pieces of the data by the RAG. We note that in multi-modal RAG, the raw context may be

first converted to *text-based context*, e.g., by using VLM for the images, and then text-based context will be used by an LLM to generate the response based on the query. Alternatively, the raw context may be directly fed to a multi-modal large language models (MLLM) to generate the response based on the query. Irrespective of the internal structure of the RAG system, the CS assesses the accuracy of each span of the generated text with the raw context which include the original set of the selected pieces. Indeed, the CS measures how correctly each part of the response reflects the information presented in the raw context. By evaluating the correctness of the text spans, we ensure that the generated responses are not only relevant but also factually accurate based on the multi-modal data. Fig. 8.1 presents an example result from RAG-Check for the given query.

Using our proposed RS model, we also develop a re-ranking algorithm to improve the retrieval stage and mitigate *RAG selection-hallucination*. In multi-modal RAG systems, a common approach is to use top- k selection based on CLIP similarity scores [139], as CLIP is widely used for its strong performance in aligning image and text embeddings across modalities. However, while effective in capturing general semantic similarity, CLIP struggles to detect irrelevance as accurately as relevance, making it less reliable for context-sensitive tasks. One fundamental limitation is its tendency to assign high similarity scores to visually or semantically generic content, even when such content is not truly relevant to the query. For instance, given a query like “a child playing soccer,” CLIP might assign high similarity to images of a child or a soccer field individually, even if neither fully matches the query. This inability to filter out irrelevant content can lead to suboptimal retrieval, ultimately

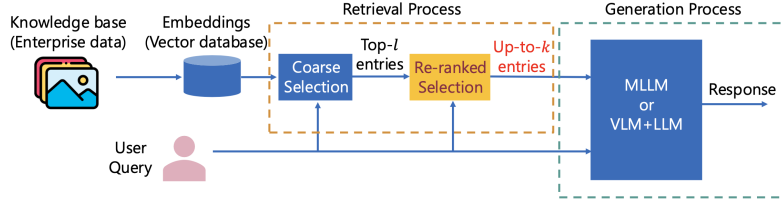


Figure 8.2: Re-ranking via relevancy score (RS).

degrading the reliability of the RAG system. In multi-modal RAG systems, where accuracy depends on precise alignment between the retrieved data and the user query, such limitations can significantly degrade the system’s reliability.

Re-ranking techniques offer a promising solution to refine initial retrieval results by re-evaluating the relevance of retrieved entries before passing them to the response generation phase. Recent works in multimodal retrieval have introduced advanced re-ranking methods to enhance relevance estimation [31, 32, 140–142]. For example, the work in [141] reconstructs data samples based on top-ranked intra- and inter-modal neighbors (referred to as “pillars”) to improve retrieval accuracy. The work in [140] employs MLLMs with knowledge-enhanced re-ranking and noise-injected training to refine retrieval results. Also, the work in [32] demonstrates the potential of vision-language models for relevance evaluation, but it highlights these models’ tendency to over-rely on semantic similarity, often failing in cases that require precise contextual understanding. A common limitation of most re-ranking approaches is their reduced ability to detect irrelevant information, as MLLMs are primarily trained on datasets containing relevant image-query pairs. Moreover, using VLM or MLLM in prior art for re-ranking is limited to a binary decision, i.e., “YES/NO”. In contrast, our RS model addresses these challenges by being explic-

itly trained to assess context-specific relevance, effectively filtering out irrelevant content and ensuring that retrieved entries align with the user’s intent. Moreover, analogous to similarity score, the RS model provides a quantitative value that can be used to explicitly rank entries based on their relevancy as well as eliminating the entries due to irrelevancy.

In this work, we address the shortcomings of CLIP-based retrieval by proposing the use of our developed RS model for re-ranking retrieved entries in multi-modal RAG systems. Unlike CLIP, the RS model is specifically trained to assess query-specific relevance while penalizing irrelevant entries effectively. For example, given the query “a doctor holding a medical instrument”, RS can prioritize images depicting this exact context over generic depictions of doctors or medical tools, which CLIP might incorrectly rank highly. By replacing the initial CLIP-based top- k selection process with RS-based re-ranking as shown in Fig. 8.2, our method ensures that retrieved data aligns more closely with the user’s intent. Our evaluation, conducted on a dataset derived from the COCO benchmark, demonstrates that RS-based re-ranking significantly improves the quality of selected images, leading to more accurate and factually grounded responses as measured by our CS. Furthermore, we highlight the broader implications of using advanced relevance metrics in retrieval refinement for multi-modal RAG systems, showcasing their potential to mitigate hallucinations and enhance response reliability.

The contributions of this work are as follows: (i) We design and train a neural network structure to find the RS between the query and each selected piece of data by RAG scheme. (ii) We perform partitioning the RAG response into spans and

categorizing each span as “subjective” or “objective”. The partitioning and categorization is performed based on developing a dedicated algorithm as well as using GPT3.5 API. (iii) We design and train a neural network structure to find the CS between the raw context, i.e., the selected pieces of information from the database by RAG, and each objective span of the generated response by RAG. (iv) In addition to the original training, validation, and testing dataset used for developing CS and RS, we also build a database containing 5000 samples by gathering human feedback (HF) on the RAG selection process and the accuracy of the RAG output spans. This collected HF was then used to assess the performance of the designed RS and CS scores with human evaluation for a RAG system. (v) While CS and RS can be used to evaluate the reliability of an instance of the response generated by RAG, we show a different use of the tools developed in this work by comparing the average performance of different RAG systems in terms of the relevancy of their retrieval and the correctness of their responses. (vi) We apply the RS model in a re-ranking scheme to replace CLIP-based top- k selection. Our RS model provides continuous-valued relevancy scores, which allow for more reliable ranking and filtering of retrieved entries.

The rest of the work is organized as follows. Section 8.2 introduces how various multi-modal RAG schemes operate. Section 8.3 describes the proposed RAG-Check system model and provides details on the training of each component. In Section 8.4, we discuss the limitations of the CLIP score in multi-modal RAG retrieval and show how the RS score more effectively distinguishes relevant from irrelevant entries. Section 8.5 presents a re-ranking scheme based on the RS score. In Section 8.6, we

report experimental results and provide analysis to validate the effectiveness of our method, and we use the proposed scores to compare different multi-modal RAG schemes. Finally, we conclude the work in Section 8.7.

8.2 Multimodal RAG

A RAG system is composed of a database, namely, enterprise database, which is usually pre-processed by partitioning the data into pieces and generating an embedding for each piece. The use of embeddings serves two main purposes: (i) To make a compact representation of each piece of information in order to enhance the retrieval process. (ii) To have speedy retrieval usually by using a similarity search using dot product (i.e., cosine similarity).

In a multi-modal RAG, depending on the type of data, embeddings are derived via corresponding encoders. Nonetheless, all embeddings should use the same embedding space in order to facilitate the search. In our work, we use CLIP embedding space to generate embeddings. Each image in the dataset is embedded using the CLIP vision encoder (CLIP-VE), and each piece of the text is embedded using the CLIP text encoder (CLIP-TE). The same goes for the query. Then, the cosine similarity is found between the embedding of the query and the embeddings in the vector database, irrespective of its original data type.

Based on such preprocessing, a vector database is produced that is composed of a pair of embeddings and the reference to each piece of information (original data) in the enterprise database. The RAG system is also composed of several blocks

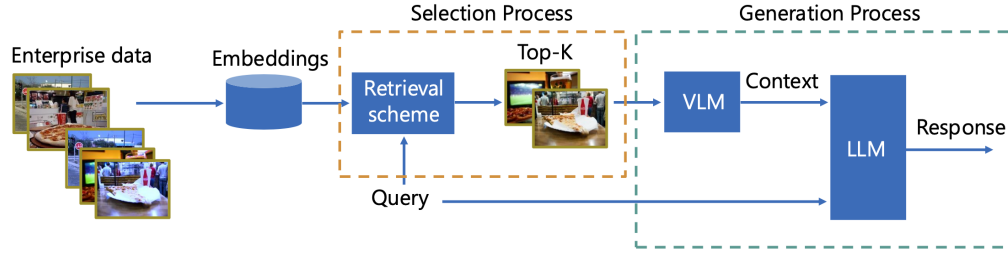


Figure 8.3: RAG scheme structure.

which are used during the operation. A selection block retrieves the relevant pieces of the enterprise data by performing a cosine similarity search using the embeddings for each piece of the data. The final stage of the selection process is determining the *raw context* which is composed of the corresponding original pieces of the data for the top- k entries in terms of the cosine similarity. The last block of the RAG system is composed of a generation block which takes the retrieved context and generates a RAG response based on the query. In multi-modal RAG, this block may be a MLLM (such as GPT4o), capable of directly handling multi-modal data. Alternatively, this block can consist of engines that generate a text-based context for each piece of retrieved data, which is then collectively used as input to an LLM (such as LLAMA or GPT3.5) to generate the response. For instance, if the original data is in text form, it is used as is, whereas a VLM is employed to convert any images into text descriptions. Both “selection block” and “generation block” are illustrated in Fig. 8.3.

8.3 RAG-Check

Our main objective in RAG-check is to evaluate both the selection process (i.e., how relevant the “retrieved documents” are to the query) and the generation process (i.e., which parts of the LLM response are supported by the RAG context). Note that the generation process comprises response generation and possibly text-based context generation. In particular, we introduce two quantitative measures: the relevance score (RS) and the correctness score (CS) for multi-modal RAG schemes to determine the performance of the selection and generation process, respectively. The RS provides a quantitative measure between zero (least relevant) and one (most relevant) for each retrieved piece of information. In order to assess the quality of the generated output in more details, we break down the generated response by RAG into smaller pieces, namely *spans*, and calculate the CS for each span. Therefore, our RAG-check model consists of three main blocks: 1) Splitting the RAG-generated response into spans and categorizing each span as either a subjective or factual statement. 2) RS model. 3) CS model. Fig. 8.4 shows the RAG-check model overlaid on the RAG system model. Fig. 8.4 also illustrates an example scenario where the query “Is beer more popular to drink with pizza or Coca-Cola?” is submitted to a RAG system, and the RAG selects the 5 most relevant images from the enterprise dataset and produces a response containing 3 statements. The RS block calculates the relevancy score for each image shown in Fig. 8.4. Also, the generated response is broken into 3 spans where one is “subjective” and the other two are marked with the respective correctness score calculated by CS block. In the following, we further

explain each of these three blocks of the RAG-check system.

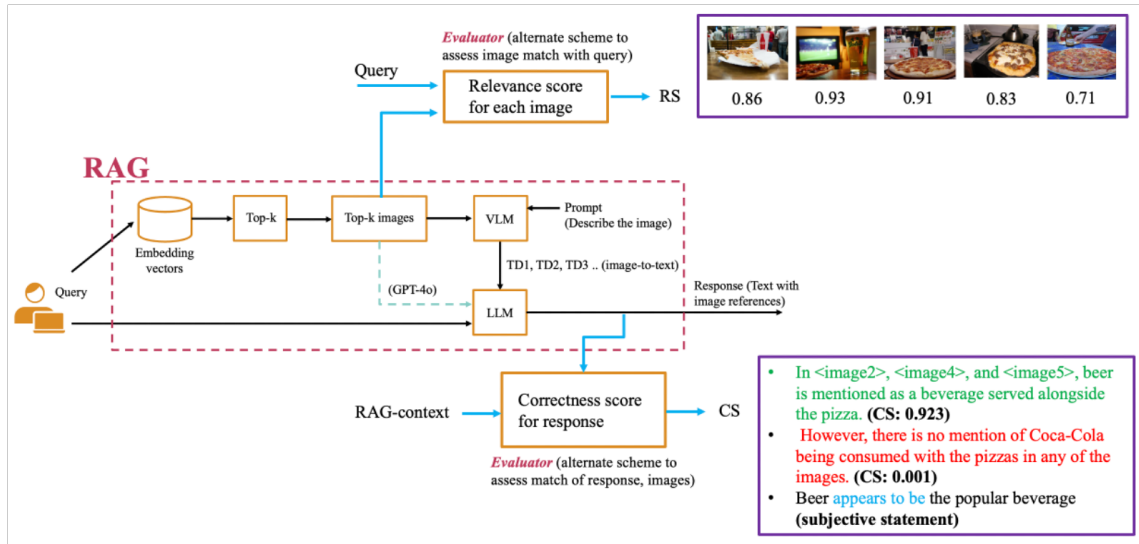


Figure 8.4: Overlay of RAG-check structure on RAG block diagram.

8.3.1 Partition and Categorization

Depending on the type of span that we define (phrase, atomic statement, sentence, or paragraph), we split the generated response of the RAG scheme into those spans. We define *atomic statements* as full sentences that are self-sufficient in expressing a meaning without the need to be evaluated along with another sentence or part of the original text. For example, consider the response as: “In the image, the desk is red and shiny. It is made of wood that is decorated with nice inlays.” In breaking down the second sentence in the response, the personal pronoun “It” has to be replaced with “The desk” to make this statement self-sufficient without the need to be evaluated with the first statement. We use GPT3.5 with a proper prompt to partition the response to atomic statements. The prompt is engineered in a way that whenever it is needed, the personal pronouns (such as ‘she’, ‘he’, ‘it’, and ‘they’),

demonstrative pronouns (such as ‘this’ and ‘that’), progressive pronouns (such as ‘his’), etc. are replaced with the proper reference from the original text.

Next, we identify whether the given span is a subjective statement or an objective statement. A subjective statement is one that may depend on human feelings, or is subject to personal viewpoints, experiences, or perspectives. Subjective statements are not scorable and are usually difficult to infer directly from the image or they are debatable. We perform an algorithm to label the subjective statements. In particular, we search for modal verbs (e.g., “could”, “might”), opinion indicators (e.g., “believe”, “feel”), hedging phrases (e.g., “it seems,”), uncertain quantifiers (e.g., “some”, “many”), adverbs of frequency and degree (e.g., “often”, “usually”), judgmental adjectives (e.g., “important”, “useful”), conjectures (e.g., “it is possible that”), and comparisons or preferences (e.g., “better”, “prefer”) in our algorithm. Note that GPT3.5 may also be used for categorization of the spans. Nonetheless, we build our system based on our custom-tailored algorithm since it had better match with human labeling of the statements as “objective” and “subjective” based on our evaluation.

8.3.2 Relevance Score (RS) Model

As defined RS provides a measure of relevance between the query and each piece of the raw context from the enterprise data. The relevancy in RAG systems is usually measured through cosine similarity between the embeddings. In this work, we address multi-modal RAG by focusing on the similarity between different data

modalities. Specifically, we focus on text-based queries and enterprise data which consists of images. We aim to obtain the RS between the query and each of the raw context (image) from the enterprise database separately. Therefore, we design and train a model that receives an image and query as an input and returns a score as a measure of the relevancy between an image and the query.

Even though the relevancy is measured by cosine similarity between the embeddings in RAG, this approach is arguably not optimal. Based on the current state-of-the-art, relevancy can be better examined by performing cross-attention between the query and the image in a transformer as in Fig. 8.5. Successive cross-attention units in the transformer integrate the query with the information from the image patches, and through training, such structure can outperform the use of cosine similarity between the embeddings of the query and image. Nonetheless, this approach incurs significant complexity, e.g., the calculation of RS in our design using the same GPU machine is 35 times slower than the computation of the cosine similarity. This is the primary reason that the RAG system relies on cosine similarity to search the enterprise data (through stored embedding in a vector database that is produced in a preprocessing stage).

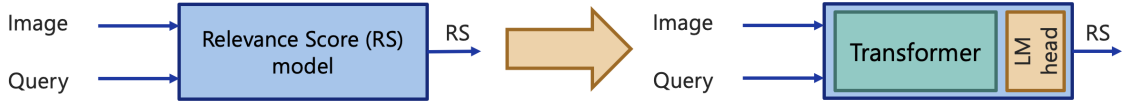


Figure 8.5: General structure to quantify relevancy of image to query.

Once the transformer module is designed properly, its output contains valuable information about the relevancy between the query and the image. At this point,

a neural network head is trained to extract this information out of the produced embedding by the cross-attention module in the form of a single real number between zero (representing no relevance) and one (representing complete relevance).

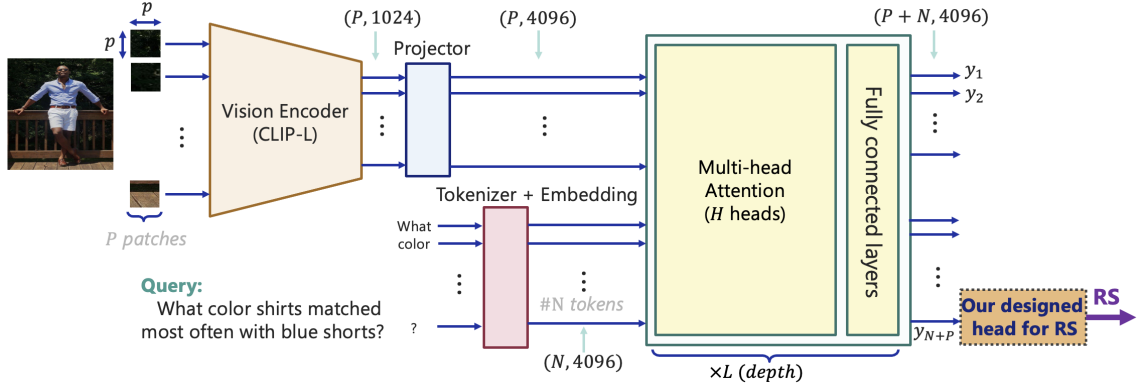


Figure 8.6: RS model structure.

Hence, to train for RS, we propose the model as shown in Fig. 8.6. This model is composed of 5 blocks discussed in the following in more details. (i) Vision encoder, which encodes patches of images separately. We use CLIP large as a vision encoder which has a transformer architecture to encode image patches. (ii) Projector: The embeddings of patches obtained from the vision encoder need to be translated to a language that is known by the transformer block, and the projector performs this conversion between the output embeddings created by the vision encoder and the input embeddings to the transformer. (iii) Tokenization and embedding: Each word in the text query has a corresponding token that will be mapped to an embedding. Some tokens require special treatment. For example, if a user query contains a special image token ($\langle \text{Image} \rangle$), the token will be replaced with the embeddings of the patches. If there is no reference to an image, the system will add image embeddings to the beginning of user query embeddings. As a result, we have $N + P$

embeddings in total where N is the number of user query tokens (with the exception of special tokens) and P is the total number of patches. The embedding space is d dimensional. (iv) Transformer block: The entire $N + P$ embeddings are processed by the transformer, which contains L transformer blocks each containing multi-head attention (MHA) unit with H attention heads and a fully-connected layers. The attention mechanism in the transformer is used to find the relation between different patches of image and user query. The output of the transformer unit is a vector of $N + P$ embeddings each with dimension d . (v) LM head: The last generated token by the transformer, i.e., y_{N+P} in Fig. 8.6, is an embedding of size d which is given to the LM head as input. The LM head is a fully connected layer that maps dimension d to 1 which is trained to represent RS.

Training the entire model from scratch to learn both language and the relationship between language and images requires vast amounts of data and computational power. Therefore, for the backbone of our system, we leverage the weights from the current state-of-the-art model. Specifically, we use LLaVA [129] weights, which include the clip-vit-large-patch14-336p as the vision encoder and LLAMA [143] as the LLM decoder, to combine the image patches and query tokens. We modify the final head of LLaVA, which originally converts embeddings into vocabulary, and replace it with a dedicated head that maps the LLM embedding dimension d to a 1 (single output). To train the RS model (specifically our head), we use a training dataset consisting of triplets $(\mathcal{I}, s_p, s_n)$, where $\mathcal{I}$ is the image, s_p is a positive statement about the image $\mathcal{I}$, and s_n is a negative statement about the image $\mathcal{I}$. We define RS model as $\mathcal{M}$. The output of our model with the given statement s is a vector

$y = M(\mathcal{I}, s)$ of dimension $N + P$. For the sake of short notation, we use $y_{-1}(\mathcal{I}, s)$ to represent the last embedding output of the LLM decoder (in Fig. 8.6) given an image and a statement. We use the following template in processing the user query s by the RS model: “Evaluate the relevancy of the given statement with the image <image>. Evaluate by either ‘relevant’ or ‘irrelevant’. The statement is: $\{s\}$.”

We use a modified version of the reinforcement learning with human feedback (RLHF) loss function [144] to train our RS model. In the original RLHF model, even though we have data indicating both highly preferable and less preferable instances, the loss function only ensures that the highly preferable instance receives a higher score than the less preferable one and there is no lower or upper bound of the loss function. However, since we want to assign a score that falls within the range of $[0, 1]$ for any given statement and image, we modify the RLHF loss function as follows:

$$\mathcal{L} = -\log(\sigma(y_{-1}(\mathcal{I}, s_p)) - \sigma(y_{-1}(\mathcal{I}, s_n))), \quad (8.1)$$

where σ is our softmax operator. During the inference, given a pair of $(\mathcal{I}, q)$, we can obtain the RS as:

$$\text{RS} = \sigma(y_{-1}(\mathcal{I}, q)). \quad (8.2)$$

8.3.3 Correctness Score (CS) Model

When the RAG response is generated, we apply the partition mechanism above to break the entire generated response r into the spans, in here atomic statements $\{s_i\}_{i=1}^L$ where L is the number of spans. For each of the atomic statements, we use the

categorization algorithms mentioned above and mark the statements as subjective or objective. For objective statements, we use CS models to obtain its correctness score. For each atomic statement, the CS model has access to all retrieved images along with the statement for the correctness measure evaluation.

In terms of structure, the difference between the CS model and the RS model is its ability to work with multiple images rather than a single image. Hence, we exploit VILA instead of the LLaVA model and to simplify training, we adopt the weight from the VILA model in Fig. 8.6 for the CS model. The reason for this is that LLaVA is not originally trained on multiple images, which limits its performance when it makes inferences on multiple images. This in turn affects the performance of the CS score which is the fine-tuned model with a dedicated LM head. In contrast, VILA has a similar structure to the LLaVA model but is trained specifically for multi-image inference.

The training process for the dedicated LM head in our CS model is similar to that of the RS model. The template that we add to the beginning of each statement s_i is as follows: “I am giving you k images. Evaluate this statement with these images and answer by either ‘correct’ or ‘incorrect’: $\{s_i\}$ ”. When there is no reference to a particular piece of context, CS is found between the statement and the entire pieces in the raw context. However, in the calculation of CS for a statement that has particular references to pieces of the context, CS is found between the statement and only the referred pieces of the context in the statement. For example, if an evaluation of the statement: $s_k = \text{“A boy with a cowboy hat is riding a white horse in <image1>”}$, the CS is only computed by using the template: “I am giving you a

statement. Evaluate this statement and answer by either ‘correct’ or ‘incorrect’: s_k ”, where the embeddings of image1 are inserted in the position of the token <image1>.

8.4 RS score vs CLIP for RAG selection

Compared to CLIP-based methods, which rely on cosine similarity between embeddings to rank image-text pairs, RS is designed to better detect both relevance and irrelevance. While CLIP is effective at capturing general semantic similarity, it often assigns high scores to visually or conceptually generic data, even if it is not contextually relevant to the query. For instance, given a query like “a person reading a book in a park,” CLIP may retrieve images of people in a park or people reading indoors, failing to prioritize those that align with the full context of the query. In contrast, RS is specifically tuned to penalize such mismatches, enabling more precise selection of relevant entries.

We evaluate both CLIP and our RS model by using 2000 pairs of images and positive statements as well as 2000 pairs of the same images and negative statements drawn from evaluation dataset. Fig. 8.7(a) shows the histograms of similarity scores based on CLIP for 2000 evaluation dataset. The histogram for the similarity scores between the CLIP embeddings of the positive (negative) statements and the image are depicted by blue (orange) color and labeled as relevant (irrelevant) statements. Let us define *CLIP-score* as the similarity score based on CLIP embeddings. From the histogram, it is evident that (i) the CLIP-score between image-text pairs has limited range, e.g., about (0.13, 0.35), and (ii) the distribution of the CLIP-score for

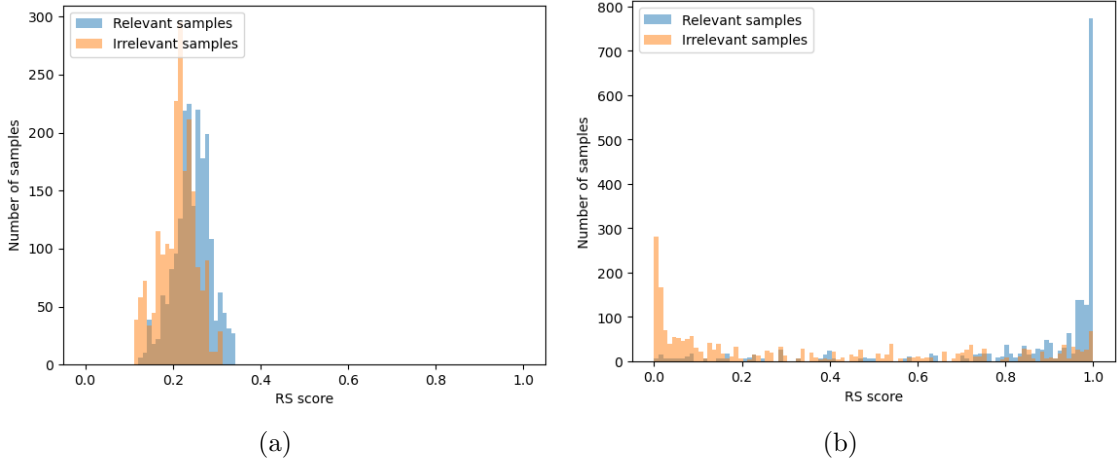


Figure 8.7: (a) CLIP-score histogram. (b) RS score histogram.

relevant and irrelevant statements has considerable overlap which means that such similarity score struggles to adequately distinguish between relevant and irrelevant pieces of data.

In contrast, as shown in Fig. 8.7(b), the RS score on the same evaluation dataset effectively shifts the distribution, creating a clearer separation between relevant and irrelevant samples. In particular, the RS score (i) exploits the full output range of $(0, 1)$ and (ii) in comparison to the similarity score based on the CLIP, the two distributions for relevant and irrelevant samples based on RS are well separated. As a simple measure of separation one can consider the distance between the mean of relevant and irrelevant samples for CLIP-score and RS that are calculated as 0.03 and 0.41 for the distribution given in Fig. 8.7(a) and Fig. 8.7(b), respectively, while the maximum separation between the mean of such scores that are limited in interval $[0, 1]$ is at most 1. A more suitable measure of separation between two discrete distribution is Jensen-Shannon divergence (JSD) which is calculated as 0.32

and 0.58 for CLIP-score and RS, respectively, with the upper bound 0.83 for scores that are limited in interval $[0,1]$.

The threshold may be used on the value obtained by CLIP-score in order to make a binary classification of the statements into relevant/irrelevant. In order to easily interpret the performance of CLIP-score based on the different thresholds we plot the cumulative density function (CDF) of the distribution of the irrelevant samples depicted in Fig. 8.7(a) and 1-cdf of the distribution of the relevant samples as illustrated in Fig. 8.8(a). By selecting a threshold value on the CLIP-score in the horizontal axis, one can easily see the probability of correct labeling of positive and negative statements by reading the corresponding values of the plotted curves. The overall accuracy is also the average between these two curves denoted by the legend ‘accuracy’. It is observed that the maximum accuracy for the CLIP-score on the evaluated dataset is about 0.65 occurring at the threshold of about 0.2.

Similarly, we plot the cdf and 1-cdf of the distributions of the irrelevant and relevant samples, respectively, for the RS in Fig. 8.8(b). By selecting an optimized threshold, RS achieves significantly higher accuracy of about 0.88 occurring at threshold 0.75.

8.5 Re-ranking with RS Score

In multi-modal RAG systems, the retrieval process plays a critical role in ensuring that selected data aligns with the user query. While CLIP-based methods are efficient for initial retrieval, they often fall short in distinguishing between truly

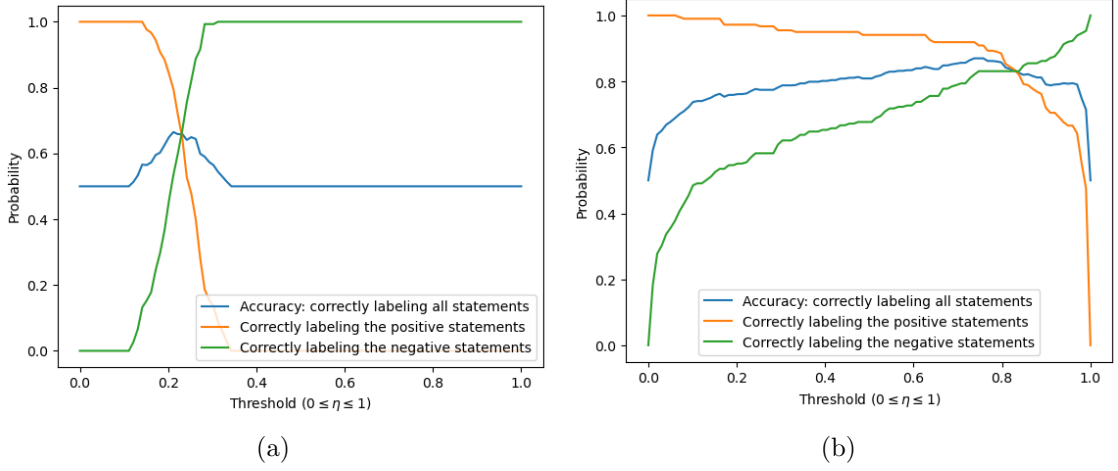


Figure 8.8: (a) CDF for relevant (irrelevant) samples in CLIP-score. (b) CDF for relevant (irrelevant) samples in RS score.

relevant and irrelevant entries. One may directly use RS score to enhance retrieval process. However, this would elongate the retrieval by a factor of $\times 35$. To address this, we propose the following re-ranking mechanism.

For selecting up-to- k entries, we first retrieve a larger candidate set of size l , $l > k$, using the computationally efficient CLIP-score. These l candidates are then re-ranked based on their RS scores, which incorporate the relevance of each candidate to the query more precisely. Finally, the up-to- k entries from the re-ranked list are selected as the final retrieved entries.

Our proposed up-to- k algorithm works as follows. From the list of l candidates, we eliminate the entries with RS below threshold τ_{lo} and keep the entries with RS above τ_{hi} . Let τ be the median of the RS for the rest of the entries. We discard the entries below τ and keep the entries above τ . If the number of kept entries exceeds k , we select k entries with highest RS, otherwise we select only the kept entries.

This adaptive re-ranking method not only improves the alignment of retrieved

entries with the user query but also reduces the likelihood of irrelevant data that could cause hallucinations in downstream tasks. By combining the time efficiency of CLIP for fast initial retrieval with enhanced capability of RS in measuring the relevancy, our approach proposes a balance between computational complexity and retrieval quality. Our experimental results illustrates significant enhancement not only in retrieval but also in the accuracy of the final output generated by multi-modal RAG systems.

8.6 Numerical Result

In this section, we present numerical results on the training and evaluation of our proposed RS and CS models. The intended use case of the RS and CS score in this work is to evaluate any particular instances of response invoked by a query when using a given RAG. In this process, RS assesses the performance of the selection scheme of the RAG and CS assesses the performance of the generation scheme (response and possibly combined with context generation) of the RAG. In the following, we first discuss the training database, the specifics of the model and hyper-parameters, and then provide the evaluation results for the RS and CS models. We also evaluate the matching and alignment of our proposed scores with the data from human evaluation. We finally use our score to compare the average performance of different RAG schemes and demonstrate how our proposed re-ranking with RS can enhance retrieval performance.

8.6.1 Dataset Specification

In this subsection, we describe the dataset used for training, validation, and evaluation of the RS and CS models. We create and utilize a dataset comprising 121,000 samples partitioned randomly for training (101K samples), validation (10K samples) and evaluation (10k samples). This dataset is a mixture of two sources: a ChatGPT-derived database and a database containing human evaluator samples [136]. The human evaluator database consists of image-statement pairs from the COCO image dataset [49], where the statements were marked as either positive or negative by human evaluators. Additionally, we task GPT-4o with generating positive and negative statements for an additional 2,000 COCO images. The statements are then checked by human for accuracy of generation. Since we aim to use the RLHF loss function for training, These two datasets (GPT derived and the datasets in [136]) are then combined such that each sample in the final database contains a triplet: an image, a positive statement, and a negative statement. An example of the dataset is given in the following.

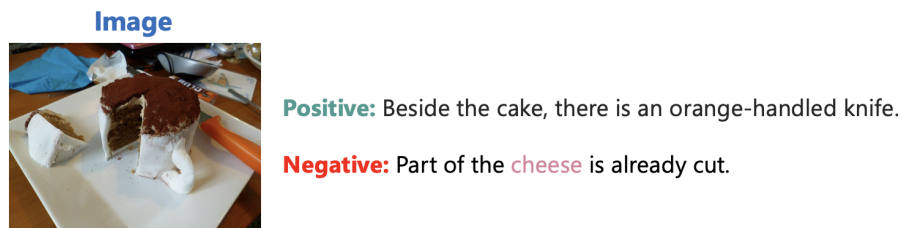


Figure 8.9: A sample entry of dataset used for training/evaluation.

To evaluate the entire RAG system and measure its alignment with human evaluators, we created two datasets (described above) by collecting human prefer-

ences regarding both RAG selection and generation processes.

8.6.2 Model and Hyperparameters

For the structure of the RS model, we use “clip-vit-large-patch14-336” [139] as a vision encoder, and a projector to map the vision encoder output to proper embedding. We use llama-1.5v [143] as a language decoder model and replace the language model final layer head with our custom head which maps from dimension 4096 to 1. To benefit from the pre-training, we use the trained weight of the LLaVA model for the vision encoder projector and language model. We train our RS model on GPU A100. For the CS model, the vision encoder is Siglip [145] and the language model is llama-3. Similarly, the last layer of the llama-3 model is replaced with our custom head which maps a layer of size 4096 to 1 and the weights are taken from the VILA model. The learning rate for both training CS and RS models are $\alpha = 10^{-4}$.

8.6.3 Relevance Score (RS) Performance

To evaluate our RS score, we use test data to determine how accurately the RS model can detect samples with relevant text to a given image. Fig. 8.7(b) shows the score histogram of 2,000 test samples for either of positive and negative statements. The RS score is a number in interval $[0, 1]$ and it is not straightforward to evaluate it with respect to an image and query since the dataset entries are only labeled as ‘relevant’ or ‘irrelevant’. Instead, we can form a *RS-labeler* by using a threshold $0 \leq \eta \leq 1$ to make a hard decision as ‘relevant/irrelevant’ in order to evaluate its

performance based on the evaluation dataset entries. One may also use the original LLaVA model with the same prompt used by RS model to form *LLaVA-labeler*.

Table 8.1 displays the values for true detection (TD) of relevant samples (denoted as true0) and irrelevant samples (denoted as true1), as well as overall accuracy of the labeling for both RS-labeler with optimized threshold $\eta = 0.7$ and LLaVA-labeler. Table 8.1 indicates that our model compared to the original LLaVA has greater confidence in determining the relevance of the image to the query as the probability of true detection for both relevant (true0) and irrelevant (true1) samples and hence the overall accuracy for RS-labeler is higher.

Table 8.1: Performance evaluation of scoring models.

Model	Accuracy	true0	true1
LLaVA	0.724	0.695	0.746
RS model	0.865	0.909	0.831
VILA	0.732	0.710	0.734
CS model	0.875	0.940	0.806

Fig. 8.8(b) illustrates the effect of changing the threshold on true0, true1, and accuracy. As shown, there is a trade-off: for example, if the threshold is decreased to near 0, almost all samples are detected as relevant, increasing the probability of correctly identifying relevant samples at the expense of incorrectly labeling irrelevant samples. We optimize the threshold η where the probabilities of detecting each type of sample are equal to balance the detection of both relevant and irrelevant samples. This threshold may be fine-tuned after training the RS model.

8.6.4 Correctness Score (CS) Performance

Using similar approach as before, we define *CS-labeler* using CS score with threshold $\eta = 0.7$ and *VILA-labeler* to evaluate the performance of CS score based on the evaluation dataset. Table 8.1 shows the evaluation result of CS-labeler and VILA-labeler in detecting the positive and negative statements as correct (denoted as true0) and incorrect (denoted as true1) depending to the given image, respectively. Similar conclusions hold for CS score as Table 8.1 indicates that CS model has greater confidence in detecting the correctness of the statement with respect to the given image in comparison to the original VILA as the probability of true detection for both correct (true0) and incorrect (true1) samples and hence the overall accuracy for CS-labeler is higher.

8.6.5 RAG Scores vs Human Evaluation

In this section, we aim to show the alignment between RS and CS with human evaluators. For RS, we randomly select 1,000 questions from the test set of the COCO-QA [146] dataset. We also gather 1,281 random images from the COCO validation set and use the CLIP-ViT-large-patch14-334p model to encode both the images and the selected questions. We obtain the top-5 images for each question that had the highest cosine similarity with the encoded query data. We collect human evaluators' opinions on the relevance of each retrieved image for each question. Evaluators could choose from five options: unsure, no relevance, mild relevance, high relevance, and complete relevance corresponding to a rating 0, 1, ..., 4, respectively.

For each question and retrieved image pair, we calculate the relevancy score with different relevancy scoring algorithms including RS, clip-vit-large-patch14-336, clip-vit-base-patch16, clip-vit-base-patch32, blip [147]. The following algorithm was used to assess how well the human evaluators’ judgments match with relevance scoring measures: We consider all possible combinations among the five retrieved images for each question. For each combination, if any image is marked as ‘unsure’ by the human evaluator, we disregard the sample. In cases where there is a difference in human evaluator’s opinion between two images, we evaluate how closely the score aligns with this difference.

Specifically, for two nonzero ratings r_1 and r_2 , where $r_1 < r_2$, given two images for a single query, if the score corresponding to the second image is higher, we consider this a match and assign a reward $r_2 - r_1$. Otherwise, it is considered unmatched, and the reward is zero. We then calculate the average reward across all pairs of images for each question and across all questions. Table 8.2 shows the average reward for different scores including RS, clip-vit-large-patch14, clip-vit-base-patch16, clip-vit-large-patch32, and BLIP. The result indicates that RS has more than 20% improvement in average reward in comparison to the other scores used for relevancy and has a good match with human evaluator’s opinion.

We use similar methodology to evaluate CS. We use the same test set of 1,000 questions. We fix the selection scheme by using the CLIP-ViT-large-patch14-336 as the vision encoder and cosine similarity to select the top-5 relevant images. Next, we employ LLaVA as the VLM and llama-1.5v as the LLM. The VLM extracts text descriptions from the selected images, while the LLM generates the final re-

sponse. To evaluate the responses, we partition the generated response into atomic statements using our partitioning algorithm. A human evaluator then assesses each atomic statement for each question, choosing one of three options: correct, incorrect, or subjective. To measure the alignment between our proposed CS and human judgment, we calculate the overlap between our CS and human evaluation. The results show a 91% match between the CS and the human evaluator’s responses.

Table 8.2: Alignment of RS scores with human evaluators.

Relevance Scoring Method	Value
Proposed Relevance Score (RS)	0.879
CLIP-vit-large-patch14-336	0.689
CLIP-vit-base-patch32	0.620
CLIP-vit-base-patch16	0.611
BLIP-large	0.491

8.6.6 Evaluation of Different RAG Schemes

In this section, we use our proposed RS and CS scores to evaluate the performance of different RAG schemes in terms of the relevancy of retrieved entries and the correctness of the generated responses.

8.6.6.1 RAG Selection Performance:

We compare the performance of different selection mechanisms in multi-modal RAG systems. Fig. 8.10 illustrates the average relevancy score for each of the top-5 retrieved images across 1,000 questions from the COCO-QA dataset and use the same set of 1,281 images from the COCO dataset. As shown, when a RAG system uses CLIP models (clip-vit-large-patch14-336, clip-vit-base-patch32, clip-vit-

base-patch16) as the vision encoder to process the images and queries, and cosine similarity to select the top-5 images, the average relevancy of the retrieved images ranges from 41% to 30%. In contrast, when using the RS model to calculate the RS score for all possible query-image pairs before selecting the top-5 images, the average relevancy scores significantly improve, ranging from 89.5% to 71%. However, using the RS model to score all possible pairs results in a $35\times$ increase in computation compared to the CLIP dot product, even when using a GPU (A100).

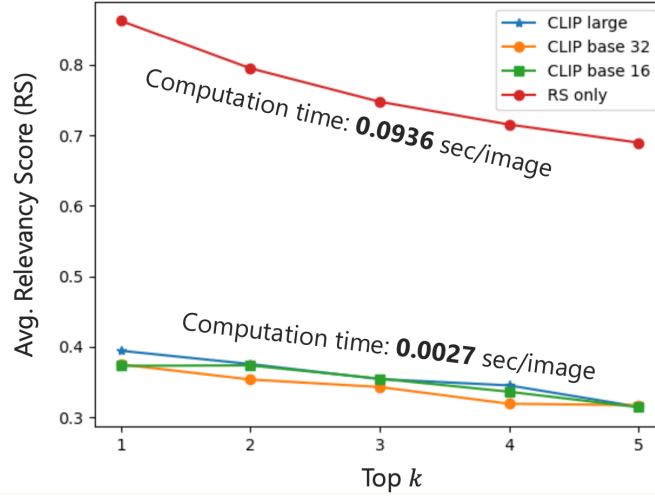


Figure 8.10: Comparison of different RAG selection mechanisms with RS model.

The results show a significant improvement in RAG selection performance compared to using CLIP alone, demonstrating how the RS model can enhance the RAG system beyond simply evaluating its performance. Fig. 8.11 illustrates that our proposed re-ranking mechanism with $l = 20$ and $l = 10$ provide a significant boost in relevancy performance of about 85% and 71%, at cost of 1.55 and 1.27 times slower retrieval process, respectively.

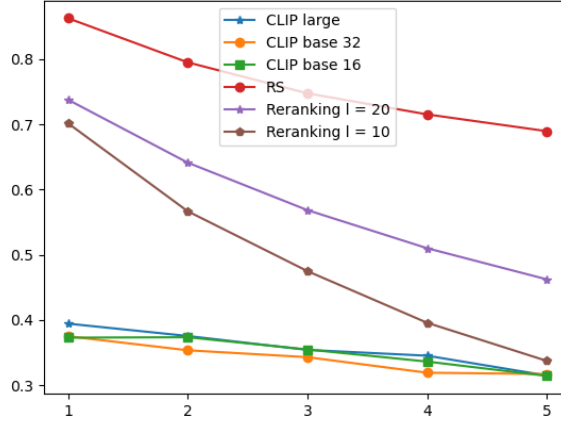


Figure 8.11: RAG retrieval relevancy performance.

8.6.6.2 RAG Context Generation Performance:

We use our proposed CS to compare various RAG schemes that incorporate different VLMs and LLMs, or in general different MLLMs. For the VLM component, one can select LLaVA or GPT-4, and for the LLM, options include LLAMA or GPT-3.5. Alternatively, one can combine both VLM and LLM into a single MLLM, such as GPT-4o, to directly generate responses from retrieved images. Fig. 8.12 illustrates the comparison between the five RAG configurations in terms of context and generation error. The results show that GPT-4o outperforms the other schemes by approximately 20%, while the remaining RAG schemes exhibit performance within the 60-68% range.

Next, we compare the generated output performance of different RAG implementations, i.e., four RAG schemes with combinations of different VLMs and LLMs as well as a RAG scheme directly based on GPT-4o that processes multi-modal context, e.g., multiple images, and text query simultaneously. For each RAG

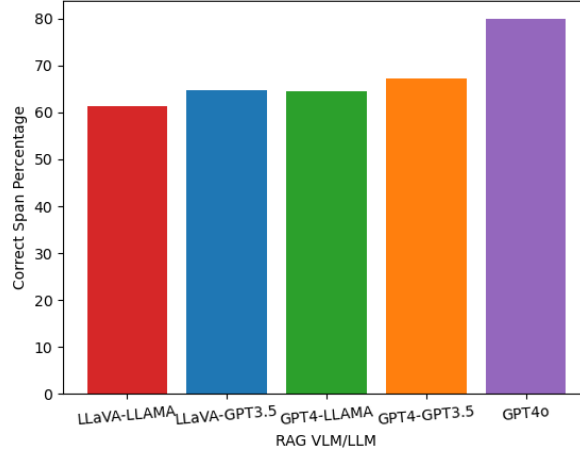


Figure 8.12: Comparison of different RAG context and generation mechanism with CS model.

implementation, we use 4 retrieval methods: top- k selection for $k = 5$ using CLIP-large, up-to- k selection directly using RS, and up-to- k selection using re-ranking with $l = 10, 20$. We use $\tau_{lo} = 0.3$ and $\tau_{hi} = 0.75$ for the up-to- k algorithm. The performance of the generated output is measured in terms of confidence score that is calculated as the geometric mean of CS and RS. The confidence score incorporates both the relevancy of the retrieved context and accuracy of the output in the view of the context in a single measure. It is important to note that a RAG may generate an output that is correct within the selected context, i.e., CS is high, but the retrieved context is not relevant, i.e., RS is low, hence, the output may not contain the answer to the user query. Fig. 8.13 indicates that applying RS directly or through re-ranking improves response quality across all evaluated RAG structures.

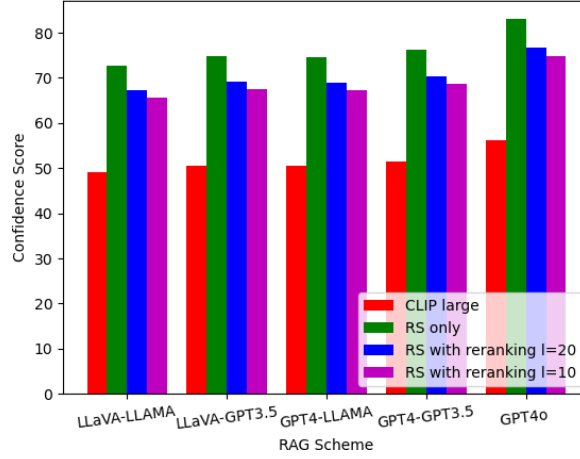


Figure 8.13: RAG generated output performance.

8.7 Conclusion

In this work, we introduced a comprehensive framework to evaluate and improve the reliability of multi-modal retrieval-augmented generation (RAG) systems. While RAG enhances large language models by incorporating external knowledge, multi-modal settings introduce unique challenges, particularly in retrieving truly relevant context and avoiding hallucinated outputs. To tackle these issues, we proposed two new performance measures: the relevancy score (RS), which quantifies how well retrieved entries match the user query, and the correctness score (CS), which evaluates the factual accuracy of generated responses. Both models were trained using a combination of a ChatGPT-derived database and human evaluator samples, achieving approximately 88% accuracy on test data. Our evaluations on a large human-annotated dataset and the COCO benchmark confirmed that the RS model aligns with human judgments 20% more frequently than existing CLIP-based

methods, while the CS model matches human preferences around 91% of the time. Furthermore, we showed that RS can be used beyond evaluation to directly improve retrieval quality. By integrating RS into the retrieval process with an adaptive up-to- k selection strategy, we significantly reduced irrelevant context and improved both selection and generation performance compared to traditional embedding-based approaches. Overall, our framework offers practical and effective tools for diagnosing and enhancing multi-modal RAG systems.

CHAPTER 9

Deep Learning-Based Real-Time Rate Control for Live Streaming on Wireless Networks

9.1 Introduction

The evolution of live streaming and video transmission has fundamentally reshaped how we connect, communicate, and consume content. High-speed internet and advanced technology empower individuals and organizations to instantly share real-time experiences, from live concerts and sports events to educational seminars and personal vlogs, transcending geographical boundaries and fostering immediacy and interactivity. Consequently, there is a growing emphasis on video transmission technologies that guarantee seamless, low-latency, high-quality live streaming over wireless channels.

A compelling use case for such technology is exemplified in the transmission of closed-circuit television (CCTV) video feeds via wireless 5G systems. CCTV serves diverse purposes, from providing multiple angles for live events like concerts and sports matches to securing large public areas. Leveraging the remarkable attributes of 5G wireless networks, such as, high speed, low latency, and large bandwidth,

CCTV video is now exceptionally clear and responsive. However, dynamic wireless channels introduce challenges, leading to potential issues like freezing and color distortions due to varying channel quality. Therefore, adapting video bitrate to match the available channel bitrate has become imperative.

In this work, we consider standard compliant systems where video compression is executed by a video encoder (VE), such as an H.264 encoder [35]. The encoded video stream is then encapsulated into container frames, which are subsequently transmitted via the physical layer. Employing standard compliant VE and communication systems offers advantages such as streamlined deployment and interoperability, alongside harnessing the efficiency of specialized hardware accelerations.

Adaptive bitrate (ABR) streaming effectively addresses these challenges by dynamically optimizing the bitrate based on network conditions, ensuring a smooth and consistent viewing experience despite data rate fluctuations. However, aligning the encoded video bitrate with the available channel bitrate is highly complex. Video bitrate fluctuates due to numerous factors, including visual complexity, color dynamics, motion, and scene variations. Simultaneously, channel bitrate changes occur due to wireless channel fading. Despite the ability to control video bitrate via VE parameter adjustments, the exact relationship between output bitrate and parameters remains elusive. This complexity is especially evident in live video streams, necessitating real-time VE parameter adjustments to optimize video quality within target encoding bitrates.

The illustration presented in Fig. 9.1 captures the inherent variations in both

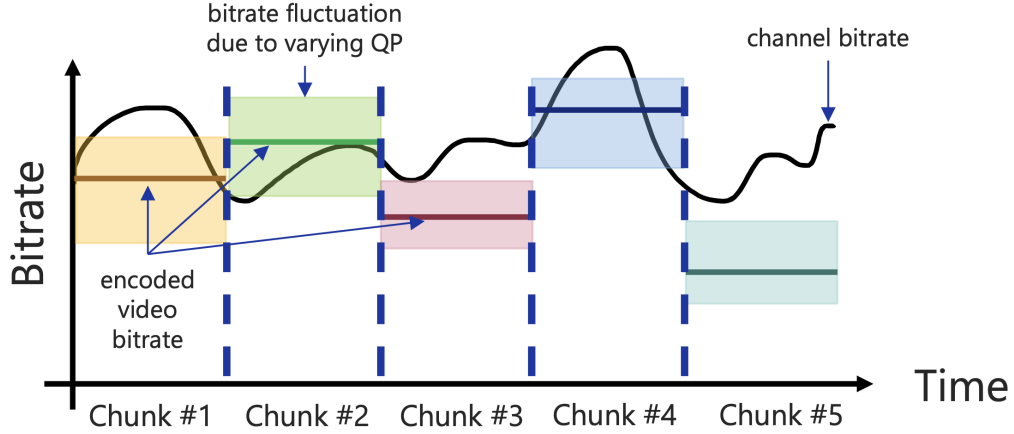


Figure 9.1: Fluctuations of encoded video bitrate and available channel bitrate over chunks.

video and channel bitrates. As input data is processed in chunks by the VE, the encoded video bitrate is established on a chunk-by-chunk basis. Fig. 9.1 brings into focus the dynamic range of encoded bitrates (depicted as solid lines) across distinct chunks, showcasing this diversity despite the uniform employment of quantization parameter (QP) at $QP = 20$ by the VE. Further, the encircling shaded region adjacent to the solid encoded bitrate line outlines the extent of fluctuations in the encoded video bitrate within the range of $15 < QP < 25$. Lastly, Fig. 9.1 represents the channel bitrate using a curve, reflecting the reality that channel quality index (CQI) feedback from users is often provided within a shorter time frame than that of a single chunk.

Several widely recognized adaptive bitrate streaming implementations are prevalent today, including Dynamic Adaptive Streaming over HTTP (DASH) [148], Apple’s HTTP Live Streaming (HLS) [149], Microsoft’s Live Smooth Streaming (Smooth) [150], and Adobe’s Adaptive Streaming (HDS) [151]. These technologies have attained significant popularity, serving as vital conduits for delivering video content

over the internet. The state-of-the-art in ABR transmission, exemplified by DASH and HLS, has indubitably revolutionized content delivery over the internet. These schemes seamlessly transition between various video quality levels, culminating in uninterrupted playback experiences. However, these shifts are confined to occur between video segments, and the necessity for pre-encoded video qualities restricts their applicability in live streaming and real-time encoding scenarios.

Compounded by the differences in video segment sizes and chunk durations, DASH's adaptability is hindered by the rapid changes in channel bitrate precipitated by wireless channel fading. Additionally, the volatile nature of video bitrates for DASH or HLS video segments complicates the adaptation of encoded bitrates to channel bandwidth variations. Consequently, the efficacy of DASH and HLS technologies in wired networks does not translate seamlessly to wireless environments, where the dynamic fluctuations of channels pose a challenge for consistent streaming experiences.

Despite their wide adoption, DASH and HLS protocols possess certain limitations that compromise their suitability for wireless channels and real-time applications, particularly live streaming scenarios. The ABR technology in DASH and HLS hinges on user-initiated and controlled selections, based on past packet receptions, device capabilities, and estimated available bitrates. This reactive approach, while delivering requested video streams, falls short in actively providing the optimal quality based on real-time network conditions. Therefore, while these technologies offer valuable adaptive bitrate streaming capabilities, their limitations warrant careful consideration when selecting streaming solutions for live streaming

over wireless channels. Our research underscores that, within 5G networks, DASH and HLS incur significant penalties in video quality (about 10-20db PSNR penalty) and artifact occurrence (about 1%-10% of the video chunks), underpinning the need for alternative approaches.

In response, this work introduces a network-aware real-time rate control (RTRC) framework, where an RTRC controller dynamically adjusts encoding parameters in real-time to encode individual video chunks with negligible delay. This controller accurately forecasts the encoded video bitrate, factoring in both “input video” and QP parameter. As a result, QP is strategically chosen to ensure that the encoded video bitrate for each chunk remains below the anticipated user bitrate, preventing bitrate overshoot and packet drops while minimizing the likelihood of user-visible artifacts. Simultaneously, optimized QP harnesses available bandwidth, thereby maximizing received video quality.

RTRC’s foundation rests on three pivotal pillars: (i) accessing user channel quality through the physical layer, (ii) forecasting the encoded video bitrates based on “input video” and QP and adapting QP to maximize the encoded video quality and satisfy bitrate constraints in harmony with the available channel bitrate, and (iii) encapsulating encoded video chunks into container packets and transmitting them via the physical layer. The proposed technology finely tailors the video stream to user capabilities, gracefully accommodating network quality fluctuations and ensuring the highest attainable video quality.

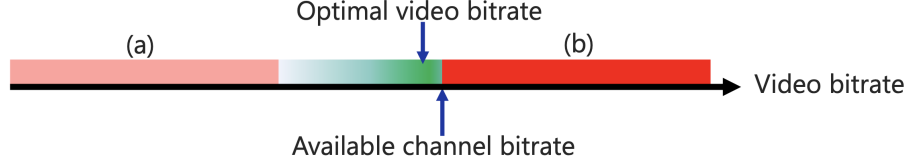


Figure 9.2: Consequences of choosing wrong value for QP.

9.2 System Model

We consider a transmitter that aims to send a live video feed to a receiver with the best possible resolution and minimal to no delay. The transmitter uses an standard compliant VE, signal modulator and channel encoder (MCE). We consider H.264 codec for VE and measure the video resolution in terms of average PSNR which is obtained as the mean of the PSNR of the reconstructed video frames. We assume that the video is taken and encoded with a fixed number of video frames per second (fps) where each video frame has constant width W and height H . The VE has minimal delay, hence the video buffer before and after VE only contains active video frames over which the temporal encoding/decoding is performed.

We assume that VE takes a *chunk* of consecutive video frames and produces the encoded video, hence a chunk contains one or more group of pictures (GOP) over which the temporal encoding/decoding happens. The encoded video $\tilde{V}$ for a chunk V is generated by VE with particular H , W , and fps . The bitrate of $\tilde{V}$, i.e., the encoded bitrate $b(V, QP)$, is a function of quantization parameter QP using H.264 codec. The encoded video is then encapsulated into *container packets* which are sent through the channel.

We consider MCE that is compliant with modulation and coding scheme table from 3GPP/LTE standard [62, Table 7.2.3]). The physical layer performs MCE for

each container packet, ensuring error-free communication as long as the encoded rate stays below the available channel rate. We assume a Rayleigh fading channel with the channel coefficient h following a circularly symmetric complex Gaussian distribution (zero mean and variance σ_n^2). The supported transmission rate for an instantaneous channel h is then given by $r(h) = \log\left(1 + \frac{P|h|^2}{\sigma_n^2 G}\right)$, where G models the gap to the capacity due to the channel coding. Please note that in this work we assume G is a fixed number and do not consider its dependency on the modulation and coding scheme (MCS). Also, we do not consider discrete values for $r(h)$ due to MCS.

We assume that the channel coherence time is longer than the duration of a chunk and the channel estimate is available for each chunk. Hence, if the encoded video rate for a chunk is not higher than the available channel rate, all the container packets will be received without error. However, if the encoded video rate exceeds the channel's supported rate, the physical layer encodes and transmits the container packets with the supported rate of the channel, hence, a number of container packets in such chunks cannot be transmitted on time by the network and they will be dropped. The number of successful packets transmitted in the duration of one chunk will be proportional to the instantaneous channel rate for that chunk.

This variability of the channel bitrate $r(h)$ due to the fading nature of the wireless channel, i.e., h , as well as the variations in the video bitrates per video chunk have to be carefully considered in the design of an RTRC system. In live streaming, the RTRC should be able to identify a correct value of QP that maximizes the video quality while satisfies the constraint on the encoded bitrate $b(V, QP)$ in real-time. If

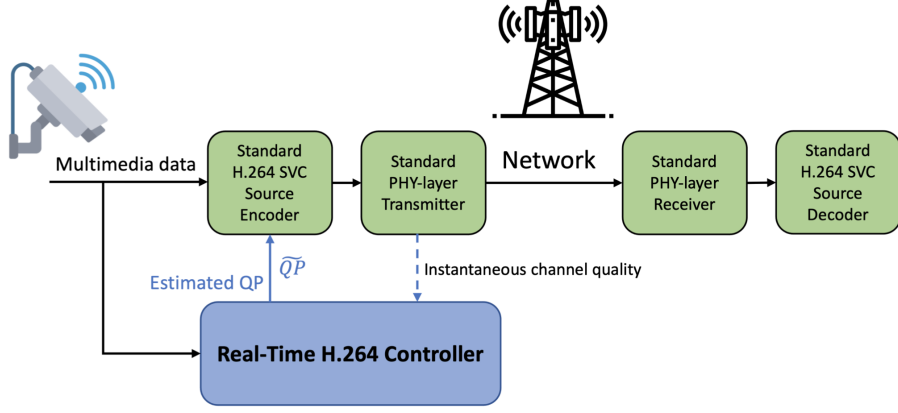


Figure 9.3: System model for RTRC.

the encoder selects a QP value that is too high, the resulting encoded video bitrate is too low and the constraint on the encoded video bitrate can be easily satisfied. However, a high value of QP leads to poor video quality on the receiver's end and an inefficient utilization of the available bandwidth. Conversely, if the selected QP is too low, the encoded video bitrate exceeds the channel bitrate, i.e., $b(V, QP) > r(h)$, that results in dropped video frames due to the channel bitrate limitations. This drop in frames leads to a noticeable decline in the received video quality due to artifacts such as freezing, blank screens, color distortions, flickering, and so on.

Fig. 9.2 illustrates two regions for encoded video bitrates with respect to an available channel bitrate for which either the bandwidth is underutilized hence video quality is bad (denoted by (a) in Fig. 9.2), or some packets are dropped and the video has artifacts (denoted by (b) in Fig. 9.2). Finally, Fig. 9.2 illustrates a region for the encoded video bitrate between regions (a) and (b) where good video quality may be achieved without packet drop and artifacts. Of course, the closer the encoded bitrate to the available channel bitrate, the better the video quality.

Hence, our goal is to design a deep learning (DL) based network-aware RTRC

which estimates the QP value in real-time (i.e., negligible delay) that maximizes the encoded video quality for a given available channel bitrate. This QP value is then used by the H.264 video encoder to generate the encoded video. The complete system model, where the proposed RTRC is highlighted with a blue bounding box, is shown in Fig. 9.3.

9.3 Deep Learning Based Rate Control

Considering that video codecs commonly possess standardized implementations and can leverage existing hardware acceleration, the central objective of a rate control unit (RCU) resides in the precise selection of the appropriate quantization parameter, QP , for each video chunk that will be applied to the encoder. This meticulous calibration endeavor is directed towards attaining an optimal video quality while adhering to maximum bitrate constraint for individual video chunks.

As discussed earlier the main challenge of the RCU is correctly forecasting the encoded video bitrates based on “input video” and QP so that the rate control (RC) module can select the QP to maximize the encoded video quality and satisfy bitrate constraints in harmony with the available channel bitrate. In this work, we exploit a DL network which is proven very effective in capturing the dynamic aspects and essential information within the video, which influences the encoded bitrate of H.264 codec for different values of QP .

Fig. 9.4 illustrates the architecture of the RCU model. The RCU receives an input video chunk along with a specified bitrate cap, denoted as BR_{max} . The

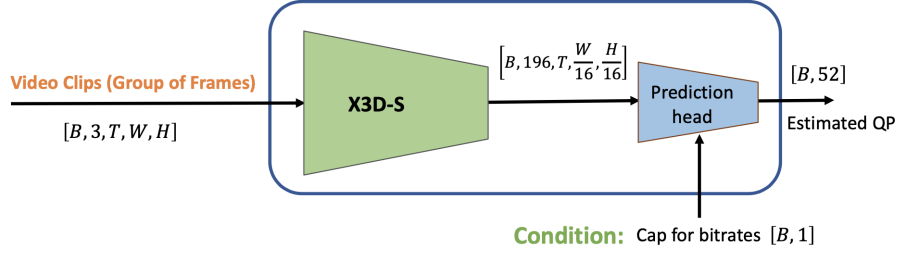


Figure 9.4: Model structure for the rate control unit (RCU).

resulting output is the QP value, which falls within the range of $[0, 51]$. This QP value is subsequently employed in the following encoding stage.

The architecture of RCU is given as follows. We employ X3D-S [152] as the base network for capturing video dynamics. X3D-S incorporates a multitude of convolutional layers, significantly augmenting its capability to recognize and comprehend video content. X3D-S processes video content with dimensions of $[B, 3, T, W, H]$, yielding corresponding features sized as $[B, 196, T, W/16, H/16]$, where B represents the batch size, T denotes the number of frames in a segment, and W and H stand for the width and height of the video segment.

The resulting feature is then further processed through a DNN prediction head comprising several convolutional layers, each followed by a conditional group normalization (CGN) block [153]. The CGN blocks normalize the output from the previous layer. Additionally, it takes a tensor of $\log_{10}(BR_{max})$ with a size of $[B, 1]$ as a conditioning factor. Each element in this tensor represents the bitrate cap for each video within the batch. CGN processes this condition through 3 linear layers, each incorporating the Gelu activation function [154], which transforms the tensor size to $[2B, 1]$. Subsequently, this tensor is split into two tensors of size $[B, 1]$, referred

to as γ and β . These tensors are then employed to adjust the normalized output from the preceding layer using the formula $\gamma \times output + \beta$. This approach effectively trains the video feature to discern QP values across various scenarios with varying BR_{max} magnitudes.

Our training-set comprises three key elements: A video chunk V , a specified target quantization parameter QP_{target} , and the corresponding bitrate BR_{target} linked to the designated QP_{target} . In our RCU, we utilize V as an input for the X3D-S model and employ the BR_{target} value as a conditioning factor for the CGN blocks. The outcome generated by the RC unit is an estimated QP represented as $\tilde{QP}$. We employ the following loss function,

$$L = L_{CE}(\tilde{QP}, QP_{target}) \quad (9.1)$$

where L_{CE} represents a cross-entropy loss function.

It is worth noting that during the training of the RCU, BR_{target} is not strictly treated as an upper bitrate limit. Therefore, during the testing phase, there is a possibility that the estimated QP is slightly lower (usually by one or two steps) than the QP value required to ensure a certain bitrate caps. To mitigate this rounding effect, we use slightly higher values for QP than the estimated QP value for encoding the data. This adjustment effectively prevents the occurrence of unwanted and destructive artifacts in the corresponding video chunk, while the resulting drop in video quality due to incremental change in QP by one or two steps remains negligible. We note that while using a higher value of QP than the estimated value

by RCU improves the packet success rate and minimizes the probability of seeing an artifact, it will also result in decrease in bandwidth efficiency and lowers the overall transmitted video quality.

9.4 Performance Evaluation

The encoded video bitrate in state-of-the-art video coding standards, such as H.264 [35] and H.265 [34], exhibits notable fluctuations. This phenomenon is well-documented, with I frames requiring significantly higher bitrates compared to P or B frames. However, these fluctuations are not limited to individual frames; they extend to larger GOPs as well.

Our extensive studies, involving hundreds of random videos from public datasets reveal that the coefficient of variation, $CV = std/mean$, computed over 10-100 consecutive chunks of the same video has a mean of about 0.3 which remains relatively high. In this study, the duration of the video chunks is 1 second that is equivalent to 30 frames when $fps = 30$; hence, a chunk corresponds to multiple GOPs or a considerably large GOP. To visually illustrate this encoded bitrate fluctuation within a single video, we use a typical video taken from a football match and partition it into 100 one-second chunks and encode each chunk using 6 values of $QP = (1, 20, 33, 37, 44, 51)$.

Fig. 9.5 illustrates the variation of the encoded video bitrate across each quality level. This observation underscores the necessity of employing deep learning techniques, as they offer an effective means to understand video dynamics within

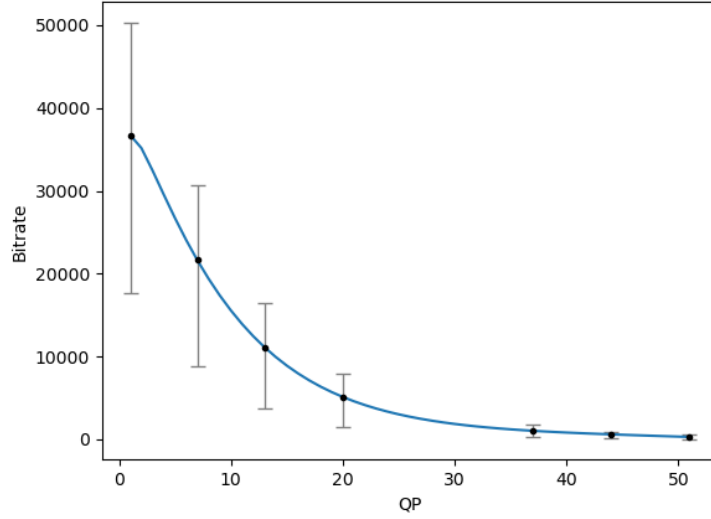


Figure 9.5: Variance of encoded video bitrate over chunks in different QP values.

a chunk and precisely adjust parameters like QP while adhering to the available channel bandwidth constraints.

To train our RCU, we assume that chunk size is equal to one GOP of size T of 8 frames. For videos with $fps = 25$, the chunk size is 0.32 seconds. We employ the QCIF dataset [155], comprising videos presented in an uncompressed YUV format. This dataset contains 25 types of videos, all in the QCIF video format, defined by dimensions of 176 pixels in width W and 140 pixels in height H . We decompose these videos into uniform chunks of 8 consecutive frames, utilizing the FFmpeg tool for this purpose. We further encode all chunked videos with FFmpeg to all 52 video qualities, resulting in a total of 66,768 video chunks. Out of the complete chunk set, we allocate 60,000 chunks for training purposes, while the remaining chunks are reserved for the testing phase. The batch size B considered is 32. In our simulations, we use Adam optimizer with a learning rate of $\eta = 10^{-4}$. The model used for the

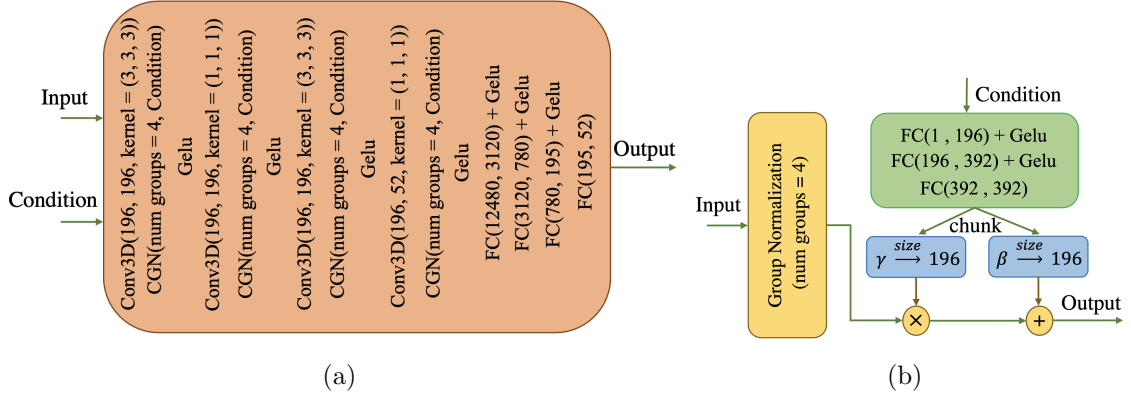


Figure 9.6: Model structure for (a) prediction head and (b) CGN.

prediction head and CGN in RCU are shown in Fig. 9.6.

We evaluate reconstruction performance with peak signal-to-noise ratio, calculated as $\text{PSNR (dB)} = 10 \log_{10} \left(\frac{\max^2}{\text{MSE}} \right)$.

During the training, the RCU model takes an input pair comprising a video chunk (from the training-set) and its corresponding encoded bitrates and the output of the RCU is compared against the input label comprising the corresponding QP value for the selected training video chunk. We evaluate the performance of the trained RCU model using the test-set. As detailed in Section 9.3, we address the inherent rounding effect in our RTRC design by incriminating the output QP from the DL RC model for desired accuracy. While the estimated output by DL model yeilds accuracy of 95.7%, adjusting the QP value one (two) step higher than the estimated output by the DL model yields an accuracy of 99.12% (99.87%).

Fig. 9.7 provides a comparative analysis of the proposed RTRC framework and DASH algorithm in terms of reconstructed video quality (measured in tems of PSNR) versus the average channel bitrate across scenario of wireless fading channels.

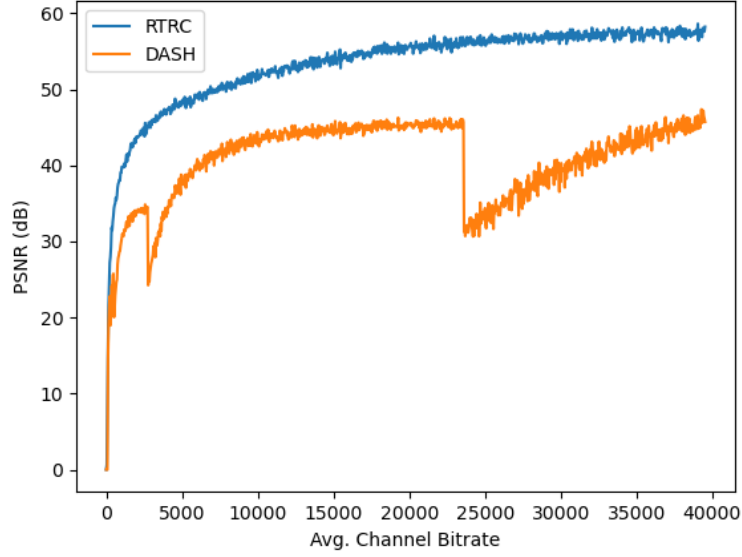


Figure 9.7: PSNR vs. average channel bitrate on fading condition.

The results demonstrate that our novel RTRC method consistently outperforms DASH in all channel circumstances, with particularly pronounced gains in fading scenarios where RTRC showcases 10-20% higher PSNR values.

The considerable improvement achieved by our innovative RTRC technology lies within two key attributes. First, the ability of RTRC to dynamically adapt the encoding bitrates encompassing an expansive spectrum of 52 distinct video quality configurations in real-time. In contrast, DASH is confined to a limited assortment of 5 pre-established video quality levels.

Second, within the RTRC framework, a strategic configuration of QP ensures that allocated bandwidth remains marginally below the predetermined threshold of the channel bitrate. In contrast, DASH adopts an alternative approach, striving to sustain consistent quality until the average channel bitrate surpasses the average bitrate associated with the subsequent available video quality level. To elaborate,

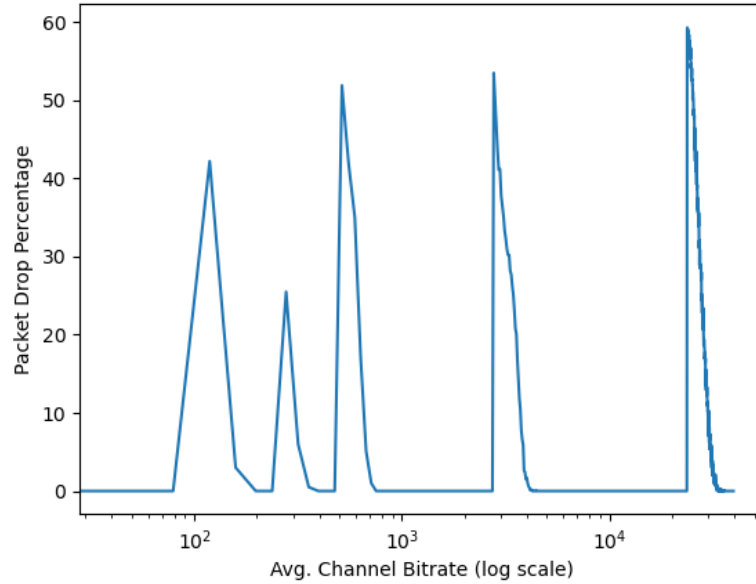


Figure 9.8: Packet drop vs. Average channel bitrate on a fading condition.

DASH makes quality determinations grounded in average quality bitrates observed over multiple chunks (i.e., segments lasting 10 seconds). Hence, DASH may suffer from noticeable artifacts due to (i) variation of the bitrates of a segment across its chunks and (ii) variation of the channel bitrate due to fading.

Fig. 9.8 shows the packet drop percentage as a function of average channel bitrate for DASH, indicating the likelihood of video artifacts per chunk. We observe that particularly at the transitions between the next video, the video bitrate is matched with the average channel quality, and hence there is considerable probability that DASH suffers from the packet drop. However, for the channel qualities which fall between the average bitrates of two available DASH resolutions, packet drop probabilities are usually very small and close to zero. These transitional artifacts are visually illustrated in Fig. 9.7.

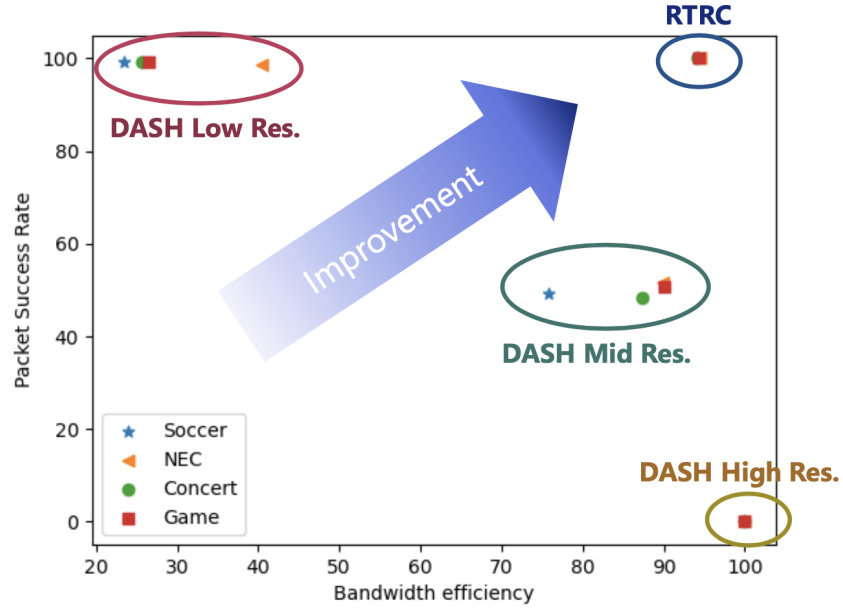


Figure 9.9: Packet success rate vs. channel bandwidth efficiency.

We define bandwidth efficiency as the variance between the video bitrate and the available channel bitrate. Meanwhile, we define the packet success rate as the proportion of chunks that do not exhibit artifacts, meaning the encoded video bitrate is lower than the available channel bitrate.

The interplay between packet success rate and channel bandwidth efficiency is shown in Fig. 9.9. In this experiment, we select four random internet videos and individually encode them at 3 distinct DASH resolutions: high, medium, and low. For all videos, we assume that the available channel bitrate is equal to the average bitrate of DASH's mid-resolution video.

Employing a lower resolution DASH configuration results in nearly 100% packet success rate because the video bitrate falls below the available channel bitrate. However, this approach underutilizes the available channel bandwidth, achieving only about 25% bandwidth efficiency. On the other hand, using a higher resolution

DASH encoding causes the encoded bitrate to exceed the available channel bitrate, resulting in a 0% packet success rate but maximizing channel bandwidth utilization.

With mid-resolution DASH encoding, the packet drop rate approaches 40% due to the closely matched encoded video bitrate and available channel bitrate. However, this approach achieves approximately 80% bandwidth efficiency.

In contrast, the RTRC outperforms all DASH resolution scenarios where combined effect of packet success rate and channel bitrate utilization is considered. This superiority is due to the strategic selection of an encoded video bitrate just below the available channel bitrate by the RTRC, leading to nearly 99.7% packet success rate and 95.6% bandwidth efficiency.

9.5 Conclusion

In this work, we proposed a deep learning-based real-time rate control framework for live video streaming over wireless networks using standard video compression schemes such as H.264. By incorporating instantaneous channel quality information from the physical layer and dynamically analyzing video content, our approach accurately predicts and selects optimal QP value to maximize video quality while ensuring that the encoded bitrate remains slightly below the available channel bitrate. This design prevents packet drops and minimizes visible artifacts, thus delivering smooth and high-quality streaming even under challenging wireless fading conditions. Experimental results on diverse video datasets demonstrate that the proposed RTRC significantly outperforms state-of-the-art ABR streaming approaches

like DASH, achieving up to 20 dB improvements in PSNR and reducing packet drop rates to as low as 0.3%. Moreover, our framework maintains near-optimal bandwidth efficiency and packet success rates above 99%, showcasing its capability to effectively utilize available channel resources while providing reliable and high-quality live streaming. Overall, the RTRC framework offers a robust and practical solution for enhancing live video delivery in dynamic wireless environments.

CHAPTER 10

Deep Learning-Based Real-Time Quality Control of Standard Video Compression for Live Streaming

10.1 Introduction

The landscape of digital communication has been profoundly reshaped by the evolution of live streaming and video transmission. This transformation has altered how we connect, communicate, and consume content. Thanks to high-speed internet and advanced technologies, individuals and organizations can now effortlessly share real-time experiences, events, and information with a global audience through live streaming. This technology knows no geographical bounds, enabling the immediate exchange of content, whether it is live concerts, sporting events, educational seminars, or personal vlogs. The primary objective is to maintain a balance between video quality and the necessary bandwidth. In certain situations, video quality requirements vary based on the service tier, while in others, these requirements change over time due to user demand. Consequently, the pursuit of video transmission technologies that can deliver seamless live streams with minimal latency and the desired quality has been paramount.

A compelling application for such technology is evident in the transmission of closed-circuit television (CCTV) video feeds. This use case is relevant in a variety of settings, ranging from providing multiple perspectives of live events like concerts and sports matches to monitoring extensive public spaces for security reasons. Typically, each video feed is required to be transmitted at a specified video quality level, often indicated by a service tier. Different video feeds may demand varying service tiers. However, a specific video feed may transition to a different service tier on demand in a relatively stable manner, triggered by significant events or suspicious activities, for example. It is important to note that higher video quality necessitates a greater bandwidth for transmission. Therefore, the goal is to minimize the encoded bitrate for each video stream to make efficient use of the available channel bandwidth. Given that the encoded video quality (e.g., in terms of PSNR) depends on the video content such as scene, action speed, etc., adjusting the encoder parameters to meet a designated video quality threshold for each video chunk while simultaneously reducing the average video bitrate is a challenging task.

In this work, we focus on systems that adhere to industry standards, where video compression is carried out by a video encoder (VE), e.g., an H.264 encoder [35]. After compression, the resulting video stream is enclosed within container frames and then sent through the physical layer. Utilizing video encoders and communication systems that conform to established standards offers various benefits, including simplified deployment, compatibility with other systems, and the utilization of specialized hardware accelerations for enhanced efficiency.

Adaptive bitrate (ABR) streaming technologies are intended to dynamically

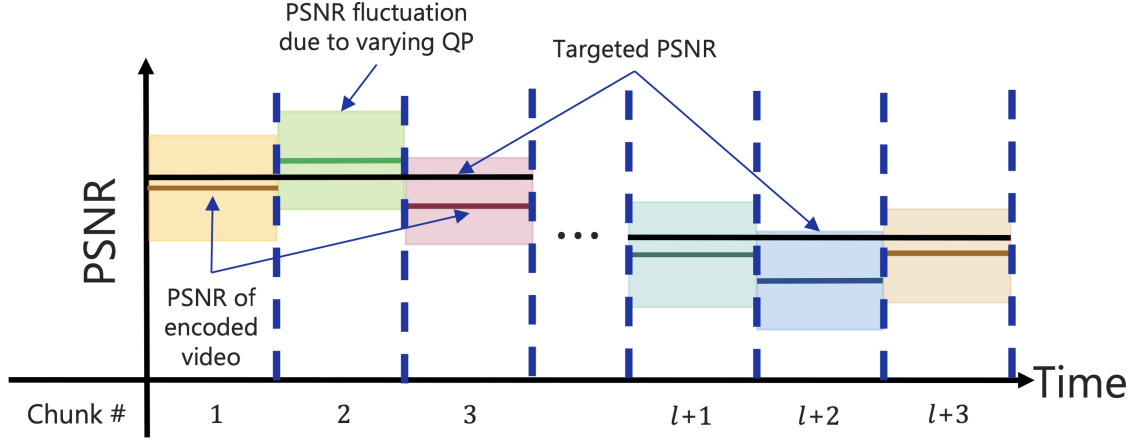


Figure 10.1: Fluctuations of encoded video bitrate and available channel bitrate over chunks.

adjust the average video bitrate. Nevertheless, the quality of the compressed video is contingent on the video content itself, including factors like the visual scene’s complexity, color dynamics, motion within frames, and scene variations. Even though we can influence the encoded video quality by tweaking encoder parameters, the exact relation between the output quality and these parameters is unknown. This intricacy is particularly pronounced in live video streaming, making it imperative to make real-time adjustments to video encoder parameters to optimize video quality while meeting the minimum encoding video quality.

The visual representation in Fig. 10.1 captures the inherent fluctuations in video quality, as measured by PSNR, stemming from the dynamic and complex nature of the scene. As data is processed in chunks by the VE, the PSNR of the encoded video is assessed for each chunk individually. In Fig. 10.1, the shaded region surrounding the solid line for each chunk showcases how the PSNR of the encoded video varies with quantization parameters QP within the range of $10 < QP < 30$.

It also illustrates the variation between different chunks, as well as the variability in the dynamic range itself across different chunks. This can be observed by examining the solid lines for each chunk, which represent the PSNR associated with $QP = 20$ for all chunks.

To achieve a target PSNR value, indicated by black solid lines in Fig. 10.1, QP values for each chunk must be dynamically adjusted before encoding to meet the PSNR requirement on a chunk-by-chunk basis. Fig. 10.1 also illustrates that the target PSNR may be quasi-stationary emphasizing the fact that the user or system requirement may vary over time.

Several well-known adaptive bitrate streaming implementations are prevalent today, such as Dynamic Adaptive Streaming over HTTP (DASH) [148], Apple's HTTP Live Streaming (HLS) [149], Microsoft's Live Smooth Streaming (Smooth) [150], and Adobe's Adaptive Streaming (HDS) [151]. These technologies have gained significant prevalence and play crucial roles in delivering video content over the internet. The state-of-the-art in ABR transmission, represented by DASH and HLS, has unquestionably transformed internet content delivery. These systems seamlessly switch between different video quality levels, ensuring uninterrupted playback experiences.

Despite their widespread use, both DASH and HLS protocols exhibit specific shortcomings that undermine their suitability for live streaming and real-time applications. ABR technology in DASH and HLS relies on user-initiated and controlled selections, driven by factors like past packet receptions, device capabilities, and estimated available bitrates. This reactive approach, while effective for delivering

requested video streams, falls short in actively providing the necessary quality for live streaming. Consequently, while these technologies offer valuable ABR streaming capabilities, it is crucial to consider their limitations when selecting streaming solutions for live streaming scenarios. DASH and HLS result in significant bandwidth penalties up to 250% and an increased likelihood of non-conformance with the required PSNR constraint, affecting about 10% to 20% of the video chunks. These findings emphasize the need for alternative approaches.

In this work, we present a network-aware real-time quality control (RTQC) framework. Within this framework, an RTQC controller dynamically adjusts encoding parameters in real-time to encode individual video chunks with minimal delay. This controller accurately predicts the encoded video quality, considering both the “input video” and the QP parameter. Consequently, QP is strategically selected to ensure that the encoded video quality for each chunk meets the minimum required PSNR specified by the user.

The intricate operation of the RTQC controller ensures to satisfy the required video quality for each video chunk with very high probability (≥ 0.99), while simultaneously minimizing the necessary bitrate to achieve such PSNR performance. The proposed technology finely customizes the video stream to accommodate user capabilities and requirements, effectively meeting the needs of all users within the network bandwidth constraints.

The rest of the work is organized as follows. We introduce the system model in Section 10.2. Section 10.3 contains the design of RTQC scheme which is the heart of the proposed system. In Section 10.4, we provide numerical evaluations of the

proposed system model and DL-based RTQC. We conclude the work in Section 10.5.

10.2 System Model

We examine a transmitter tasked with encoding a real-time video feed, ensuring that the quality of each video segment exceeds a predefined threshold while minimizing the average bitrate of the encoded video stream. The transmitter employs a standard-compliant VE, a signal modulator and channel encoder (MCE). In this context, we adopt the H.264 codec for the VE and quantify video quality based on the PSNR of the reconstructed video segments.

Our scenario assumes a fixed number of video frames per second (fps) during the video capture and encoding process. Each video frame maintains a constant width (W) and height (H). With minimal delay in the VE, the video buffer preceding and following the VE exclusively contains active video frames, facilitating temporal encoding/decoding processes.

We assume that VE takes a *chunk* of consecutive video frames, hence, a chunk contains one or more GOP defined as a group of pictures, i.e., video frames, over which the temporal encoding/decoding takes place. The encoded video $\tilde{V}$ for a chunk V is generated by VE with particular H , W , and fps . Both the PSNR $p(V, QP)$ and the bitrate of $\tilde{V}$, i.e., the encoded bitrate $b(V, QP)$, are a function of quantization parameter QP using H.264 codec.

The live video stream is continuously partitioned into video chunks and then each chunk is encoded using a QP . The encoded video is then encapsulated into

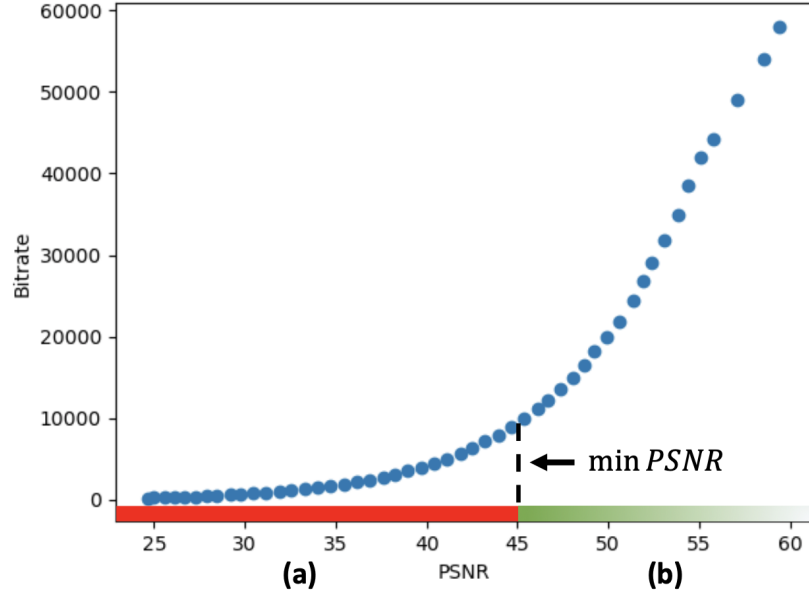


Figure 10.2: Consequences of choosing wrong value for QP .

container packets which are sent through the channel. Assuming error-free reception of the container packets through the error control behavior of the physical layer, the video is then decoded. The goal is to satisfy a minimum level of the video quality of the decoded video chunks in terms of PSNR. The PSNR for each video chunk is controlled during the video compression though setting the QP of a standard VE. The QP which takes values from 0 to 51. The lower the QP the higher the quality and essentially $QP = 0$ is a near lossless compression.

The video quality (PSNR) and the bitrate are functions of the video content, e.g., the scene complexity and dynamics of the video frames. Hence, even though the value of QP can be changed to increase or decrease the PSNR or the bitrate, the same value of QP produces considerably different video quality or bitrate depending on the video content. Later in the evaluation section (Section 10.4) we illustrate the extent of such dependency (e.g., in terms of PSNR) as a function of QP for a few

typical videos. The real-time nature of live streaming makes controlling the QP to satisfy PSNR and minimize bitrate even more challenging.

In live streaming, the RTQC technology should be able to identify a correct value of QP that is sufficient to satisfy the constraint on video quality $p(V, QP)$ while minimizes the encoded bitrate $b(V, QP)$ in real-time. If the encoder selects a QP value that is too low, the resulting encoded video quality is too high and the constraint on the encoded video quality (PSNR) can be easily satisfied. However, a low value of QP leads to unnecessary high bitrate and an inefficient utilization of the available bandwidth. Conversely, if the selected QP is too high, the encoded video quality (PSNR) falls short of the target PSNR threshold (i.e., $p(V, QP) > \lambda(t)$), that results in non-conforming video chunks with respect to the system PSNR threshold $\lambda(t)$ at time t .

Fig. 10.2 depicts the bitrate as a function of PSNR for a typical video. It is observed that the higher the requirement on PSNR the higher the bitrate required to satisfy the requirement. In the same figure and just above the PSNR axis, we illustrate two regions for encoded video quality with respect to a constraint on PSNR. The region (a) in the left marked with red is where we should avoid since the PSNR is not sufficient and hence the encoded video chunk is non-conforming. The region (b) in the right marked with gradient-filled green is where the PSNR is satisfied, but in order to minimize the bitrate we would like to operate as close to the PSNR threshold as possible without crossing the PSNR threshold.

Hence, our goal is to design a deep learning-based (DL-based) network-aware RTQC which can estimate the QP value in a real-time (i.e., with negligible delay) that

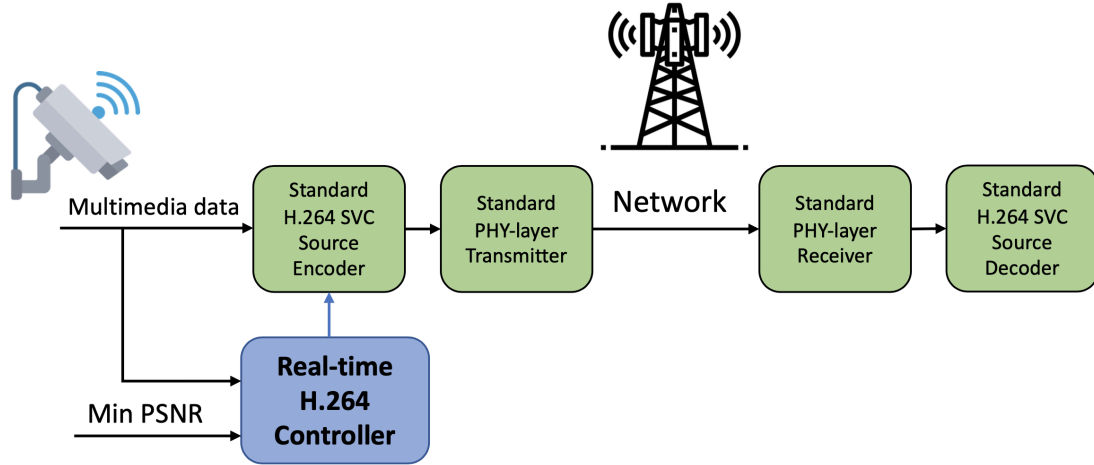


Figure 10.3: System model for RTQC.

minimizes the encoded video bitrate and satisfies a given video quality, i.e., $\text{PSNR} > \lambda(t)$. This value of QP is then used by the H.264 video encoder to generate the encoded video. The complete system model can be seen in Fig. 10.3, where the proposed RTQC is highlighted with blue bounding box.

10.3 Deep Learning-Based Rate Control

Our objective is to develop a control unit (CU) capable of accurately predicting the encoded video quality, measured in terms of PSNR. This prediction relies on both the “input video” and the QP value. The ultimate goal is to enable the CU to intelligently select the QP value, minimizing the encoded video bitrate while adhering to specified PSNR constraints. In this study, we leverage the effectiveness of a DL network. This approach has demonstrated considerable efficacy in capturing the dynamic aspects and crucial information inherent in the video, influencing the encoded PSNR of the H.264 codec across various QP values.

The architecture of the CU model is illustrated in Fig. 10.4. In this setup, the CU is provided with an input video chunk, along with a specified minimum PSNR requirement denoted as `min PSNR`. The value of `min PSNR` is represented by $\lambda(t)$ highlighting that it may be time varying due to the change in the user demand or other system functions. The resulting output of the CU is the QP value, which falls within the range of $[0, 51]$. This QP value is subsequently utilized in the subsequent encoding stage.

The CU architecture is described as follows. We adopt X3D-S [152] as the foundational network for capturing video dynamics. X3D-S incorporates a diverse set of convolutional layers, significantly enhancing its ability to recognize and comprehend video content. Processing video content with dimensions $[B, 3, T, W, H]$, X3D-S produces corresponding features sized as $[B, 196, T, W/16, H/16]$, where B represents the batch size, T denotes the number of frames in a segment, and W and H signify the width and height of the video segment, respectively.

The resulting feature is further processed through a deep neural network (DNN) prediction head, which includes multiple convolutional layers, each followed by a conditional group normalization (CGN) block [153]. The CGN blocks serve to normalize the output from the preceding layer. Notably, the CGN block takes a tensor of $\log_{10}(\lambda(t))$ with a size of $[B, 1]$ as a conditioning factor. Each element in this tensor represents the required minimum PSNR for each video within the batch.

The CGN processes this condition through three linear layers, each incorporating the Gelu activation function [154], thereby transforming the tensor size to $[2B, 1]$. Subsequently, this tensor is split into two tensors of size $[B, 1]$, denoted as

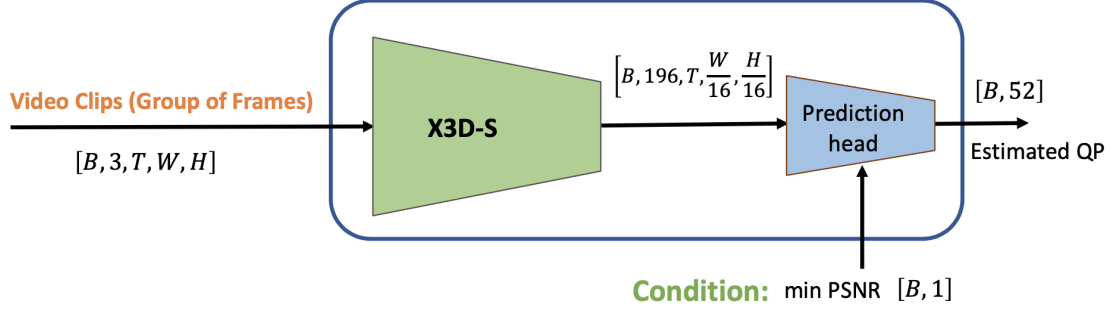


Figure 10.4: Model structure for the control unit (CU).

γ and β . These tensors are then utilized to adjust the normalized output from the preceding layer using the formula $\gamma \times output + \beta$. This approach effectively trains the video feature to discern QP values across various scenarios with varying magnitudes of $\lambda(t)$.

Our training set comprises three essential elements: a video chunk (V), a specified target quantization parameter (QP_{target}), and the corresponding minimum PSNR (λ_{target}) linked to the designated QP_{target} . Within our CU, we utilize V as input for the X3D-S model and employ the λ_{target} value as a conditioning factor for the CGN blocks. The outcome generated by the CU is an estimated QP , represented as $\widetilde{QP}$. For training, we employ the following loss function:

$$L = L_{CE}(\widetilde{QP}, QP_{\text{target}}). \quad (10.1)$$

Here, L_{CE} denotes a cross-entropy loss function. During CU training, λ_{target} is not strictly treated as a minimum PSNR limit. Consequently, during testing, there is a possibility that the estimated QP is slightly higher (usually by one or two steps) than the QP value required to ensure a certain minimum PSNR. To counteract this

rounding effect, we utilize slightly lower values for QP than the estimated $\widetilde{QP}$ for encoding the data. This adjustment effectively prevents the occurrence of unwanted non-conforming compressed video output for the corresponding video chunk. While the resulting increase in the compressed video bitrate due to a slight decrement in QP by one or two steps is tolerable, it is crucial to note that using a lower value of QP than the estimated value by CU improves the packet success rate in terms of adhering to the minimum PSNR requirement. However, this approach may decrease bandwidth efficiency and increase the overall transmitted video bitrate.

10.4 Performance Evaluation

In this section, we commence by assessing the variation of PSNR as a function of the QP values. Our extensive studies, covering hundreds of randomly selected videos from public datasets, reveal that the coefficient of variation ($CV = \frac{\text{std}}{\text{mean}}$), defined as the ratio of the standard deviation over the mean of the PSNR, computed over 10 to 100 consecutive chunks of the same video has a mean of approximately 0.23, indicating a relatively high variability. For this study, the duration of the video chunks is set to 1 second, equivalent to 30 frames when $fps = 30$; thus, a chunk corresponds to multiple GOPs or a considerably large GOP.

To visually depict the encoded PSNR fluctuations within a single video and across different videos, we select three representative videos, one from a football match, one from a live interview, and one from a concert show. Each of these three videos are partitioned into 100 one-second chunks, and each chunk is en-

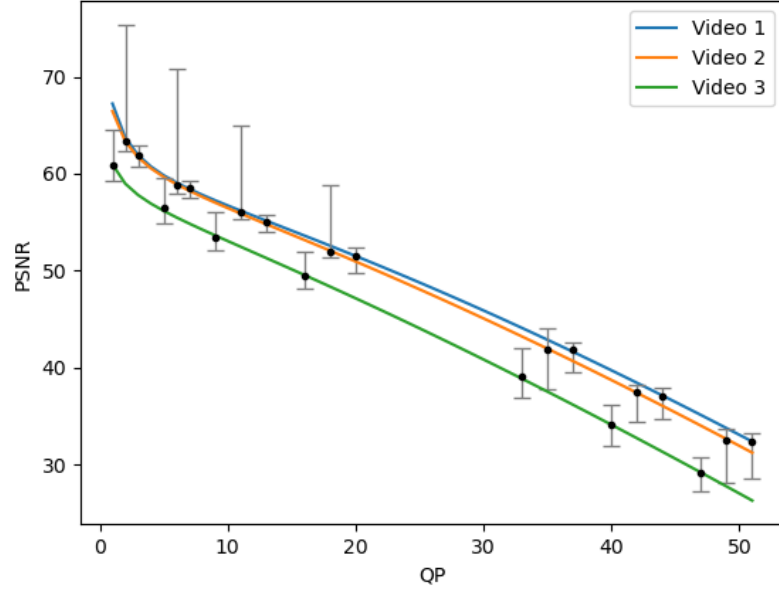


Figure 10.5: Variance in PSNR of encoded video over chunks in different QP values.

coded using seven values of QP such as $QP_1 = (1, 5, 9, 16, 33, 40, 47)$, $QP_2 = (2, 6, 11, 18, 35, 42, 49)$, and $QP_3 = (3, 7, 13, 20, 37, 44, 51)$, respectively.

Fig. 10.5 illustrates the variation in encoded video quality, measured in terms of PSNR, across each quality level. The curves denote the average PSNR for the entire video using the corresponding value of QP for each of the three selected videos. The vertical bars at seven different QP values in each curve mark the minimum and the maximum PSNR achieved in one of the 100 chunks for the corresponding video. We intentionally use slightly different QP values for different videos so that the vertical bars do not interfere.

We observe significant PSNR variation for each QP among the video chunks within each video and also across different videos. This observation underscores the importance of employing a technique, such as the DL approach presented in

this work, that can effectively understand video dynamics for each individual video chunk. This understanding enables precise adjustment of encoder parameters to meet the minimum PSNR requirement while minimizing the average encoded bitrate.

To facilitate the training of our CU, we assume that the chunk size is equivalent to one GOP with a size (T) of 8 frames. For videos with a frame rate of $fps = 25$, the resulting chunk size is 0.32 seconds. We make use of the QCIF dataset [155], which consists of videos in an uncompressed YUV format. This dataset encompasses 25 types of videos, all in the QCIF video format, characterized by dimensions of 176 pixels in width (W) and 140 pixels in height (H). These videos are decomposed into uniform chunks of 8 consecutive frames, a process facilitated by the FFmpeg tool. Subsequently, we encode all the chunked videos with FFmpeg across 52 video qualities, resulting in a total of 66,768 video chunks.

For training purposes, we allocate 60,000 chunks from the complete set, reserving the remaining chunks for the testing phase. The batch size (B) is set to 32. In our simulation, we employ the Adam optimizer with a learning rate of $\eta = 10^{-4}$.

We evaluate the reconstruction performance in terms of PSNR defined as the average PSNR of the video frames defined as

$$\text{PSNR (dB)} = 10 \log_{10} \left(\frac{\max^2}{MSE} \right). \quad (10.2)$$

The model used for CU is comprises two parts the prediction head and the CGN as shown in Fig. 10.6 and Fig. 10.7, respectively.

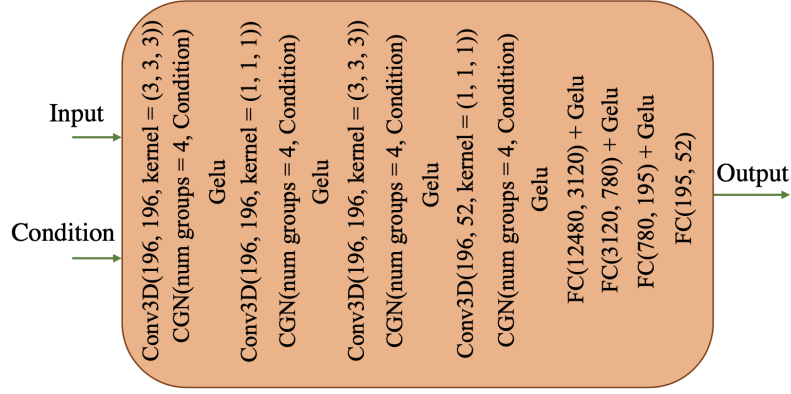


Figure 10.6: Model structure for prediction head.

During the training, the CU model takes an input pair comprising a video chunk (from the training-set) and its corresponding encoded PSNR and the output of the CU is compared against the input label comprising the corresponding QP value for the selected training video chunk. Subsequently, we evaluate the performance of the trained CU model using the test-set. As detailed in Section 10.3, we address the inherent rounding effect in our RTQC design by decrementing the output QP from the DL RC model by one or two steps for the desired accuracy. While the estimated output by DL model yields accuracy of 93.8%, adjusting the QP value one (two) step lower than the estimated output by the DL model yields an accuracy of 98.10% (99.70%).

Fig. 10.8 provides a comparative analysis of the proposed RTQC framework and DASH algorithm in terms of average bitrate versus the reconstructed video quality (measured in terms of PSNR). In this simulation, the CU has no prior knowledge of the live video streams and for each video chunk it forecasts the proper QP to satisfy the minimum PSNR and minimize the average bitrate. However, we use a very favorable situation for the DASH algorithm by assuming that DASH has

prior knowledge about the average of the PSNR over the next 100 video chunk for each of its resolutions, and hence DASH picks the resolution based on this knowledge. The results demonstrate that our novel RTQC method consistently outperforms DASH by achieving considerably (up to 2.5 times) lower bitrate depending on the target PSNR value.

The considerable improvement achieved by our novel RTQC technology lies within two key attributes. First, the ability of RTQC to dynamically adapt the encoding PSNR using 52 distinct video quality configurations in real-time, while DASH is limited to an assortment of 5 pre-established video quality levels. Second, the delicate configuration of QP in RTQC ensures that the PSNR of the encoded frame marginally surpasses the predetermined threshold with high probability. In contrast, DASH adopts an alternative approach, striving to sustain consistent quality until the average channel bitrate surpasses the average bitrate associated with the subsequent available video quality level. To elaborate, DASH makes quality determinations grounded in average video quality observed over multiple chunks (in this context we assume a non-causal information about a future video segment of duration 10 seconds). Hence, DASH may suffer from high probability of non-conformance with the required PSNR for each chunk due to variation of the PSNR of a segment across its chunks.

Fig. 10.9 illustrates the non-conformance probability of the DASH algorithm as a function of target PSNR for DASH. It is observed that particularly at the transitions between the next video, the video PSNR is matched with the average PSNR over multiple chunks, and hence, there is considerable chance that DASH

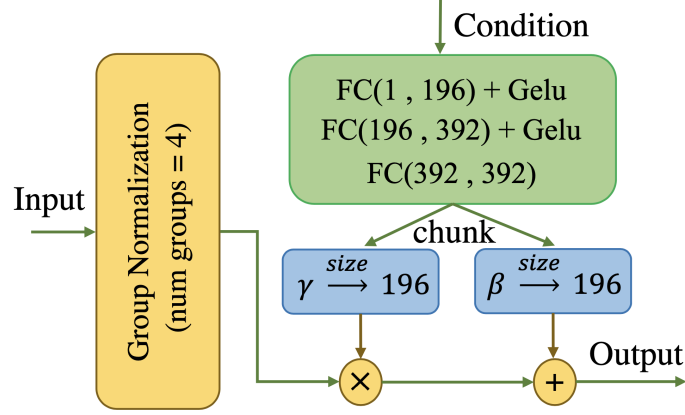


Figure 10.7: Model structure for CGN.

suffers from the non-conformance probability of up to 40%. However, for the channel qualities which fall between the average bitrates of two available DASH resolutions, the non-conformance probability remains very small and close to zero. The non-conformance probability of the RTQC is omitted from Fig. 10.9 as it is below 10^{-2} and does not add valuable illustration in this figure.

We define bandwidth efficiency between 0 to 1 (0 being the worst and 1 being the best) which measures how close an algorithm performs in terms of average encoded bitrate comparison to the optimal solution. Let $b_{opt}(V)$ denote the optimal bitrate for a video chunk V . The bandwidth efficiency is then defined as an average of the bandwidth efficiency $BW_{eff}(V)$ of the chunk V defined as

$$BW_{eff}(V) = 1 - \frac{\max(0, b(V, \tilde{QP}) - b_{opt}(V))}{b(V, \tilde{QP})} \quad (10.3)$$

where $b(V, \tilde{QP})$ denotes the bitrate of the algorithm for the compressed video chunk V using the $\tilde{QP}$ value for quantization parameter QP . The value 0 indicates the worst and 1 denotes the best bandwidth efficiency.

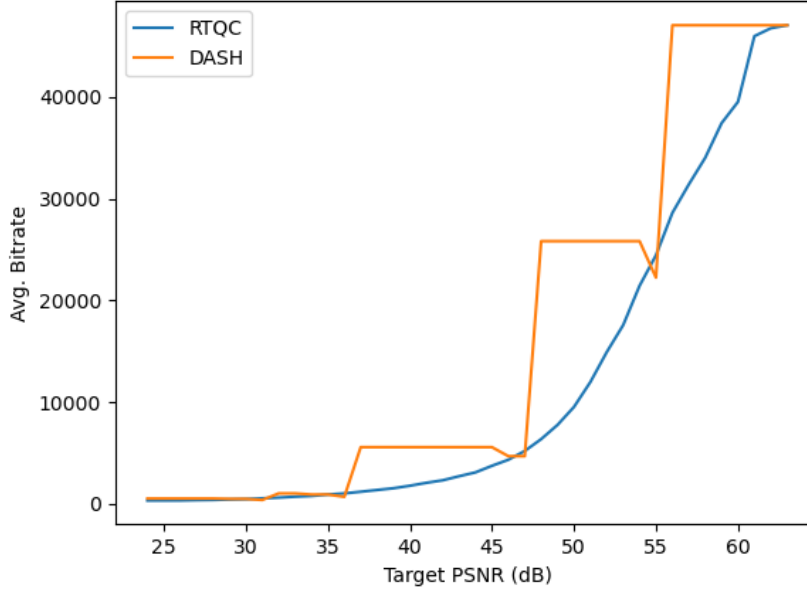


Figure 10.8: Bitrate vs. target video PSNR.

The interplay between conformance probability and bandwidth efficiency is depicted in Fig. 10.10. In this experiment, we select four random internet videos and individually encode them at three distinct DASH resolutions: high, medium, and low. For all videos, we assume the target PSNR is equivalent to the average PSNR of the corresponding video.

Employing a high-resolution DASH configuration results in nearly 100% conformance probability because the encoded video PSNR most likely satisfies the target PSNR. However, this approach demands high bitrate and hence it achieves only about 28% bandwidth efficiency. On the other hand, using a lower resolution DASH encoding causes the encoded video PSNR to fall short of the target PSNR with high probability resulting in very low conformance probability but maximizing bandwidth efficiency. With mid-resolution DASH encoding, the conformance approaches 68%

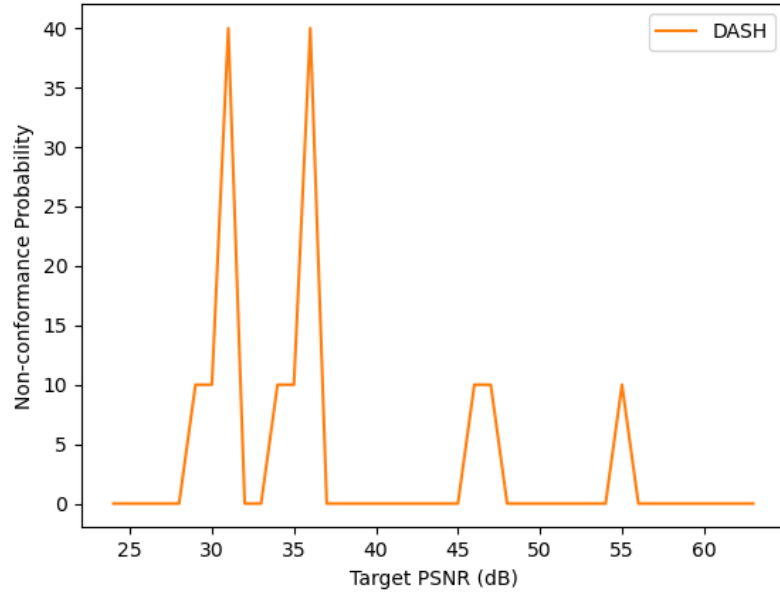


Figure 10.9: Non-conformance probability vs. targeted video PSNR.

due to the closely matched encoded video quality with the target PSNR. Meanwhile, this approach achieves approximately 75% bandwidth efficiency. This phenomenon by itself reveals the fundamental trade-off between the bandwidth efficiency and packet conformance.

In contrast, the RTQC outperforms all DASH resolution scenarios in both aspects of the conformance probability and bandwidth efficiency. This superiority is due to the precise selection of QP to ensure the target PSNR for the encoded video while minimizing the average bitrate, leading to nearly 98.7% packet success rate and 99.1% bandwidth efficiency.

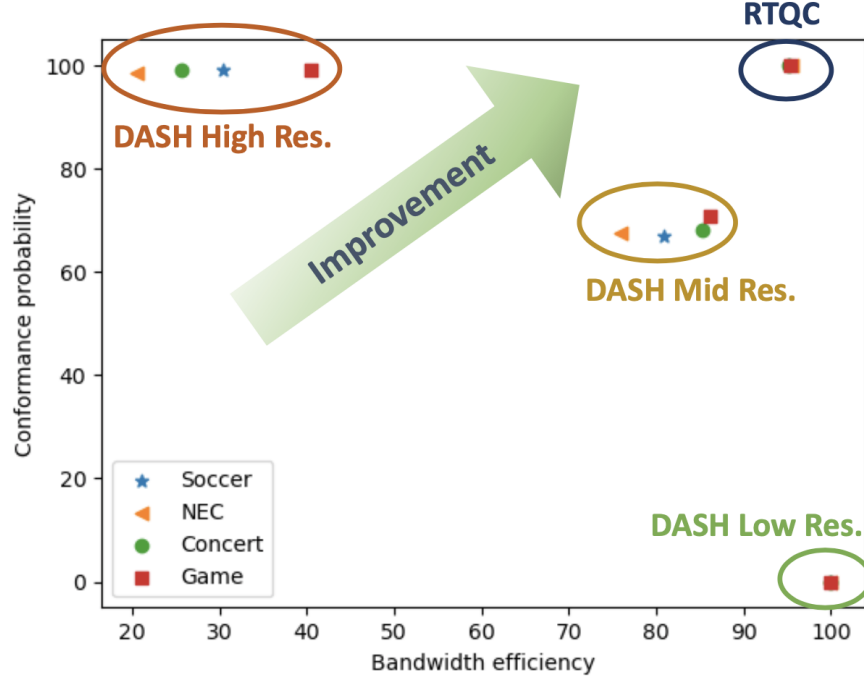


Figure 10.10: Conformance probability vs. channel bandwidth efficiency.

10.5 Conclusion

We proposed a DL-based real-time quality control scheme that works with existing standard video compression schemes for live-streaming of video content. The proposed DL-based real-time H.264 controller offers a promising solution by dynamically adjusting encoder parameter, namely, QP values. For encoding the live stream feeds, our proposed scheme results in reconstructed video quality that conforms with target PSNR while achieving significant reduction in required bitrate in comparison to the state-of-the-art schemes such as DASH.

CHAPTER 11

Conclusions

In this dissertation, we explored 6G communication systems through the lenses of semantic communication, federated learning, RAG-Check, age of information, and real-time rate control.

In Chapter 2, we developed a comprehensive transformer-aided semantic communication framework that prioritizes transmitting the most semantically important content under bandwidth constraints. We first proposed a vision-only approach using attention scores from a vision transformer to identify and transmit the most informative image patches. We then extended this to a multi-resolution encoding strategy, where patches are assigned different resolutions based on their semantic importance and the available channel capacity, enabling fine-grained adaptation to varying communication conditions. Finally, we introduced a multi-modal extension that incorporates user-provided textual queries through cross-attention mechanisms, allowing the system to focus on image regions that align with both visual content and user intent. From the experimental results, we showed that the proposed framework effectively preserves semantic fidelity and task-relevant information while adapting

to bandwidth limitations.

In Chapter 3, we introduced SMRC framework that enables progressive refinement of both reconstruction quality and semantic accuracy through hierarchical, end-to-end joint source-channel coding. Inspired by multi-task learning principles, we designed a novel multi-head autoencoder architecture that hierarchically encodes input data into successive sub-blocks, each providing incremental improvements in resolution and semantic feature accuracy. This structure allows the decoder to flexibly combine different layers based on application needs, preserving critical semantic attributes while adapting to varying channel and user requirements. Our framework also introduced a joint loss formulation that balances reconstruction and semantic objectives, enabling fine-grained control over trade-offs such as precision versus recall for specific semantic tasks. From the experimental results, we showed that the proposed SMRC significantly outperforms traditional separate source-channel coding approaches, validating its effectiveness for bandwidth-efficient, task-oriented communication.

In Chapter 4, we proposed FedGradNorm, a personalized federated multi-task learning framework that dynamically adjusts task-specific gradient weights to address statistical heterogeneity and ensure fair and balanced learning across diverse clients. We introduced a bilevel optimization formulation to derive adaptive weighting strategies, enabling tasks with different learning dynamics and data distributions to converge at similar rates. We also provided a theoretical analysis establishing the exponential convergence of our proposed methods, supporting their practical effectiveness. From the experimental results, we showed that FedGrad-

Norm improves learning fairness and convergence speed while offering robustness to imbalanced data.

In Chapter 5, we proposed a personalized decentralized multi-task learning framework that adapts the communication topology among clients to mitigate the adverse effects of data heterogeneity. By introducing a novel transference-based metric to quantify the positive and negative impacts of task updates on each other, we designed an algorithm that dynamically adjusts the network connectivity to connect positively correlated clients and disconnect negatively correlated ones. Our framework leverages spectral clustering to systematically form these adaptive topologies, ensuring that each client benefits from shared knowledge while avoiding detrimental interference. Additionally, we provided a convergence analysis of the proposed method under dynamic graph structures. From the experimental results, we showed that our approach achieves faster convergence and improved task-specific performance compared to fully connected decentralized learning setups.

In Chapter 6, we proposed *rAge-k*, a novel communication-efficient federated learning framework that integrates the AoI metric to simultaneously address data heterogeneity and reduce communication overhead. We introduced age vectors at the parameter server to track the staleness of gradient indices and selectively request updates on the most outdated parameters, thereby enabling effective gradient sparsification. Additionally, we designed a clustering mechanism using DBSCAN to group clients with similar data distributions based on their update patterns, allowing for more personalized and efficient coordination within clusters. We also provided theoretical convergence analysis under standard assumptions, establishing rigorous

guarantees for our approach. From the experimental results, we showed that $r\text{Age-}k$ achieves faster convergence and higher accuracy compared to existing sparsification-based methods, confirming its potential for scalable and robust federated learning in heterogeneous and bandwidth-limited environments.

In Chapter 7, we introduced a novel distributed sensing framework for analyzing the version age of information (AoI) in networks with multiple sensing nodes providing views of a common source. By generalizing the traditional single-source model, we developed recursive expressions to characterize the average version AoI for arbitrary node subsets and derived scaling laws for key network topologies, including ring, line, and fully connected networks. Our analysis showed that distributed sensing can achieve comparable or even superior AoI scaling behavior compared to centralized single-source setups, particularly when the number of sensing nodes and their placement are appropriately chosen. These theoretical insights demonstrate the fundamental potential of multi-view sensing to maintain information freshness in large-scale networks, offering practical guidelines for the design of future real-time monitoring and distributed sensing systems.

In Chapter 8, we introduced RAG-Check, a comprehensive framework to evaluate and improve the reliability of multi-modal RAG systems. We designed two novel performance measures: the relevancy score (RS), which quantifies the alignment between retrieved entries and the user query, and the correctness score (CS), which assesses the factual accuracy of generated responses with respect to the raw retrieved context. Our framework includes dedicated neural network architectures for both RS and CS, as well as a span partitioning and categorization algorithm

to distinguish objective from subjective statements in generated responses. Additionally, we demonstrated how RS can be used not only as an evaluation metric but also to enhance retrieval performance via a re-ranking mechanism, addressing limitations of traditional CLIP-based selection. From the experimental results, we showed that our approach effectively reduces hallucinations and improves overall response reliability, confirming the practical impact of the proposed framework for advancing multi-modal RAG systems.

In Chapter 9, we proposed a deep learning-based real-time rate control framework for live video streaming over wireless networks to address the challenges posed by highly dynamic video content and rapidly fluctuating channel conditions. We designed a novel RCU that uses a deep neural network architecture, integrating video content features and instantaneous channel bitrate information to accurately predict the optimal QP value for each video chunk in real time. This approach ensures that the encoded video bitrate remains just below the available channel capacity, effectively minimizing packet drops while maximizing video quality and bandwidth efficiency. From the experimental results, we showed that the RTRC framework significantly improves video quality and reliability compared to state-of-the-art adaptive bitrate streaming methods, demonstrating its potential to enable high-quality, artifact-free live streaming even under challenging wireless conditions.

In Chapter 10, we proposed a deep learning-based real-time quality control framework for live video streaming over wireless networks to ensure high-quality video delivery under strict bandwidth constraints. We designed a novel CU architecture that leverages deep neural networks to dynamically predict the optimal QP

for each video chunk, based on both the video content and a user-defined minimum PSNR requirement. This allows the encoder to satisfy target quality levels with high confidence while minimizing the transmitted bitrate. By integrating conditional group normalization and carefully conditioned feature extraction, our RTQC framework adaptively adjusts encoding parameters in real time, enabling fine-grained control over video quality without sacrificing bandwidth efficiency. From the experimental results, we showed that our RTQC approach consistently outperforms state-of-the-art adaptive bitrate methods such as DASH, achieving significantly lower bitrates while maintaining very high conformance to quality constraints.

The contents of Chapter 2 are published in [11,39,41], Chapter 3 in [2], Chapter 4 in [63,90,91], Chapter 5 in [156], Chapter 6 in [157], Chapter 7 in [158], Chapter 8 in [159,160], Chapter 9 in [161], Chapter 10 in [162].

Bibliography

- [1] Y. E. Sagduyu, S. Ulukus, and A. Yener. Task-oriented communications for nextg: End-to-end deep learning and ai security aspects. *IEEE Wireless Communications*, 30(3):52–60, June 2023.
- [2] M. Mortaheb, M. A. Khojastepour, S. T. Chakradhar, and S. Ulukus. Semantic multi-resolution communications. In *IEEE Globecom Workshops*, December 2023.
- [3] K. Yang, S. Wang, J. Dai, K. Tan, K. Niu, and P. Zhang. Witt: A wireless image transmission transformer for semantic communications. In *IEEE ICASSP*, June 2023.
- [4] G. Zhang, Q. Hu, Z. Qin, Y. Cai, G. Yu, and X. Tao. A unified multi-task semantic communication system for multimodal data. *IEEE Transactions on Communications*, Feb 2024.
- [5] Z. Qin, X. Tao, J. Lu, W. Tong, and G. Y. Li. Semantic communications: Principles and challenges. Available online at arXiv:2201.01389.
- [6] G. Shi, Y. Xiao, Y. Li, and X. Xie. From semantic communication to semantic-aware networking: Model, architecture, and open problems. *IEEE Communications Magazine*, 59(8):44–50, September 2021.
- [7] K. B. Letaief, W. Chen, Y. Shi, J. Zhang, and Y. A. Zhang. The roadmap to 6g: Ai empowered wireless networks. *IEEE communications magazine*, 57(8):84–90, August 2019.
- [8] W. Yang, H. Du, Z. Q. Liew, W. Y. B. Lim, Z. Xiong, D. Niyato, X. Chi, X. Shen, and C. Miao. Semantic communications for future internet: Fundamentals, applications, and challenges. *IEEE Communications Surveys & Tutorials*, 25(1):213–250, November 2022.
- [9] G. Zhu, D. Liu, Y. Du, C. You, J. Zhang, and K. Huang. Toward an intelligent edge: Wireless communication meets machine learning. *IEEE communications magazine*, 58(1):19–25, January 2020.

- [10] Y. E. Sagduyu, T. Erpek, S. Ulukus, and A. Yener. Vulnerabilities of deep learning-driven semantic communications to backdoor (trojan) attacks. In *IEEE CISS*, March 2023.
- [11] M. Mortaheb, E. Karakaya, M. A. Khojastepour, and S. Ulukus. Transformer-aided semantic communications. In *IEEE Asilomar*, October 2024.
- [12] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, May 2021.
- [13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *NeurIPS*, December 2017.
- [14] F. Hanzely and P. Richtárik. Federated learning of a mixture of global and local models. Available online at arXiv:2002.05516.
- [15] Z. Yang, M. Chen, K. Wong, H. V. Poor, and S. Cui. Federated learning for 6g: Applications, challenges, and opportunities. *Engineering*, 8:33–41, January 2022.
- [16] B. Isik, F. Pase, D. Gunduz, S. Koyejo, T. Weissman, and M. Zorzi. Communication-efficient federated learning through importance sampling. Available online at arXiv:2306.12625.
- [17] L. Liu, J. Zhang, S. H. Song, and K. B. Letaief. Client-edge-cloud hierarchical federated learning. In *IEEE ICC*, June 2020.
- [18] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In *AISTATS*, August 2020.
- [19] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *AISTATS*, April 2017.
- [20] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith. Federated optimization in heterogeneous networks. In *MLSys*, March 2020.
- [21] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. In *ICML*, July 2020.
- [22] C. Fifty, E. Amid, Z. Zhao, T. Yu, R. Anil, and C. Finn. Measuring and harnessing transference in multi-task learning. Available online at arXiv:2010.15413.

- [23] F. Sattler, S. Wiedemann, K. R. Müller, and W. Samek. Robust and communication-efficient federated learning from non-iid data. *IEEE transactions on neural networks and learning systems*, 31(9):3400–3413, September 2019.
- [24] Y. Lin, S. Han, H. Mao, Y. Wang, and B. Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *ICLR*, May 2018.
- [25] L. P. Barnes, H. A. Inan, Be. Isik, and A. Özgür. rtop-k: A statistical estimation approach to distributed sg. *IEEE Journal on Selected Areas in Information Theory*, 1(3):897–907, November 2020.
- [26] R. D. Yates, Y. Sun, R. Brown, S. K. Kaul, E. Modiano, and S. Ulukus. Age of information: An introduction and survey. *IEEE Journal on Selected Areas in Communications*, 39(5):1183–1210, May 2021.
- [27] R. D. Yates and S. K. Kaul. The age of information: Real-time status updating by multiple sources. *IEEE Transactions on Information Theory*, 65(3):1807–1827, March 2019.
- [28] A. Kosta, N. Pappas, and V. Angelakis. Age of information: A new concept, metric, and tool. *Foundations and Trends in Networking*, 12(3):162–259, November 2017.
- [29] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in NIPS*, 2020.
- [30] S. Es, J. James, L. Espinosa-Anke, and S. Schockaert. Ragas: Automated evaluation of retrieval augmented generation. *arXiv preprint arXiv:2309.15217*, 2023.
- [31] N. Messina, M. Stefanini, M. Cornia, L. Baraldi, F. Falchi, G. Amato, and R. Cucchiara. Aladin: distilling fine-grained alignment scores for efficient image-text matching and retrieval. In *CBMI*, pages 64–70, 2022.
- [32] J. H. Yang and J. Lin. Toward automatic relevance judgment using vision-language models for image-text retrieval evaluation. *arXiv preprint arXiv:2408.01363*, 2024.
- [33] C. Reich, B. Debnath, D. Patel, T. Prangemeier, and S. T. Chakradhar. Deep video codec control. Available online at arXiv:2308.16215.
- [34] G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand. Overview of the high efficiency video coding (HEVC) standard. *IEEE Transactions on Circ. and Sys. for Video Tech.*, 22(12):1649–1668, September 2012.

- [35] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the h. 264/avc video coding standard. *IEEE Transactions on Circ. and Sys. for Video Tech.*, 13(7):560–576, July 2003.
- [36] R. Cheng, K. Liu, N. Wu, and B. Han. Enriching telepresence with semantic-driven holographic communication. In *ACM HotNets*, November 2023.
- [37] Y. Qiao, Q. Zheng, Y. Lin, Y. Fang, Y. Xu, and T. Zhao. Haptic communication: Toward 5G tactile internet. In *IEEE CSRSWTC*, pages 1–3, 2020.
- [38] H. Wu, Y. Shao, E. Ozfatura, K. Mikolajczyk, and D. Gündüz. Transformer-aided wireless image transmission with channel feedback. Available online at arXiv:2306.09101.
- [39] M. Mortaheb, M. A. Khojastepour, and S. Ulukus. Efficient semantic communication through transformer-aided compression. In *IEEE ICMLCN*, May 2025.
- [40] A. Devoto, F. Alvetreti, J. Pomponi, P. Di Lorenzo, P. Minervini, and S. Scardapane. Adaptive layer selection for efficient vision transformer fine-tuning. Available online at arXiv:2408.08670.
- [41] M. Mortaheb, E. Karakaya, and S. Ulukus. Multi-modal semantic communication. In *IEEE Asilomar*, October 2025.
- [42] G. Ilharco, M. Wortsman, N. Carlini, R. Taori, A. Dave, V. Shankar, H. Namkoong, J. Miller, H. Hajishirzi, A. Farhadi, and L. Schmidt. Open clip, June 2021.
- [43] S. Jiao, H. Zhu, J. Huang, Y. Zhao, Y. Wei, and S. Humphrey. Collaborative vision-text representation optimizing for open-vocabulary segmentation. In *ECCV*, October 2024.
- [44] B. Cheng, A. Choudhuri, I. Misra, A. Kirillov, R. Girdhar, and A. G. Schwing. Mask2former for video instance segmentation. Available online at arXiv:2112.10764.
- [45] B. Cheng, A. Schwing, and A. Kirillov. Per-pixel classification is not all you need for semantic segmentation. *NeurIPS*, December 2021.
- [46] H. W. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.
- [47] H. Pouransari and S. Ghili. Tiny imagenet visual recognition challenge. 2014.
- [48] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, April 2015.

- [49] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, September 2014.
- [50] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *IEEE ICCV*, Oct 2021.
- [51] C. E. Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, Oct 1948.
- [52] D. Gunduz, E. Erkip, A. Goldsmith, and H. V. Poor. Source and channel coding for correlated sources over multiuser channels. *IEEE Transactions on Information Theory*, 55(9):3927–3944, Sep 2009.
- [53] D. B. Kurka and D. Gündüz. Deepjssc-f: Deep joint source-channel coding of images with feedback. *IEEE Journal on Selected Areas in Information Theory*, 1(1):178–193, May 2020.
- [54] M. Yang, C. Bian, and H. Kim. Deep joint source channel coding for wireless image transmission with ofdm. Available online at arXiv:2101.03909.
- [55] P. Baldi. Autoencoders, unsupervised learning, and deep architectures. In *ICML*, pages 37–49, June 2012.
- [56] F. A. Aoudia and J. Hoydis. Model-free training of end-to-end communication systems. *IEEE Journal on Selected Areas in Communications*, 37(11):2503–2516, Nov 2019.
- [57] F. A. Aoudia and J. Hoydis. End-to-end learning for ofdm: From neural receivers to pilotless communication. *IEEE Transactions on Wireless Communications*, 21(2):1049–1063, Feb 2021.
- [58] Y. Liu, C. Ponce, S. L. Brunton, and J. N. Kutz. Multiresolution convolutional autoencoders. *Journal of Computational Physics*, 474:111801, Feb 2023.
- [59] D. B. Kurka and D. Gündüz. Successive refinement of images with deep joint source-channel coding. In *IEEE SPAWC*, pages 1–5, July 2019.
- [60] Y. E. Sagduyu, T. Erpek, S. Ulukus, and A. Yener. Is semantic communications secure? A tale of multi-domain adversarial attacks. Available online at arXiv:2212.10438.
- [61] Y. Zhang and Q. Yang. A survey on multi-task learning. Available online at arXiv:1707.08114.
- [62] 3GPP TS36.213 V12.3.0. Evolved universal terrestrial radio access (e-utra); physical layer procedures.

- [63] M. Mortezaei, C. Vahapoglu, and S. Ulukus. Personalized federated multi-task learning over wireless fading channels. *Algorithms*, 15(11):421, November 2022.
- [64] R. Caruana. Multitask learning. *Machine Learning*, 28:41–75, July 1997.
- [65] L. Collins, H. Hassani, A. Mokhtari, and S. Shakkottai. Exploiting shared representations for personalized federated learning. In *ICML*, July 2021.
- [66] M. G. Arivazhagan, V. Aggarwal, A. K. Singh, and S. Choudhary. Federated learning with personalization layers. Available online at arXiv:1912.00818.
- [67] Y. Deng, MM. Kamani, and M. Mahdavi. Adaptive personalized federated learning. Available online at arXiv:2003.13461.
- [68] A. Fallah, A. Mokhtari, and A. Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *NeurIPS*, December 2020.
- [69] G. Lan and Y. Zhou. An optimal randomized incremental gradient method. *Math. program.*, 171:167–215, September 2018.
- [70] V. Smith, C. K. Chiang, M. Sanjabi, and A. S. Talwalkar. Federated multi-task learning. *NeurIPS*, December 2017.
- [71] P. P. Liang, T. Liu, L. Ziyin, N. B. Allen, R. P. Auerbach, D. Brent, R. Salakhutdinov, and L. P. Morency. Think locally, act globally: Federated learning with local and global representations. Available online at arXiv:2001.01523.
- [72] A. Agarwal, J. Langford, and C. Y. Wei. Federated residual learning. Available online at arXiv:2003.12880.
- [73] F. Hanzely, B. Zhao, and M. Kolar. Personalized federated learning: A unified framework and universal optimization techniques. Available online at arXiv:2102.09743.
- [74] A. Kendall, Y. Gal, and R. Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, June 2018.
- [75] W. Qian, B. Chen, Y. Zhang, G. Wen, and F. Gechter. Multi-task variational information bottleneck. Available online at arXiv:2007.00339.
- [76] Z. Chen, V. Badrinarayanan, C. Lee, and A. Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *ICML*, July 2018.
- [77] Z. Zhang, P. Luo, C.C. Loy, and X. Tang. Facial landmark detection by deep multi-task learning. In *ECCV*, September 2014.

- [78] A. Jagannath and J. Jagannath. Multi-task learning approach for automatic modulation and wireless signal classification. In *IEEE ICC*, June 2021.
- [79] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konecný, S. Mazzocchi, H. McMahan, T. V. Overveldt, D. Petrou, D. Ramage, and J. Roselander. Towards federated learning at scale: System design. April 2019.
- [80] A. Sinha, P. Malo, and K. Deb. A review on bilevel optimization: from classical to evolutionary approaches and applications. *IEEE Transactions on Evolutionary Computation*, 22(2):276–295, 2017.
- [81] P. Hansen, B. Jaumard, and G. Savard. New branch-and-bound rules for linear bilevel programming. *SIAM J. Sci. Comput.*, 13:1194–1217, September 1992.
- [82] C. Shi, J. Lu, and G. Zhang. An extended kuhn-tucker approach for linear bilevel programming. *Applied Mathematics and Computation*, 162:51–63, March 2005.
- [83] K. P. Bennett and G. M. Moore. Bilevel programming algorithms for machine learning model selection. Rensselaer Polytechnic Institute, 2010.
- [84] J. Domke. Generic methods for optimization-based modeling. In *AISTATS*, April 2012.
- [85] S. Ghadimi and M. Wang. Approximation methods for bilevel programming. Available online at arXiv:1802.02246.
- [86] R. Grazzi, L. Franceschi, M. Pontil, and S. Salzo. On the iteration complexity of hypergradient computation. In *ICML*, July 2020.
- [87] A. Shaban, C. A. Cheng, N. Hatch, and B. Boots. Truncated back-propagation for bilevel optimization. In *AISTATS*, April 2019.
- [88] D. Maclaurin, D. Duvenaud, and R. Adams. Gradient-based hyperparameter optimization through reversible learning. In *ICML*, September 2015.
- [89] K. Ji, J. Yang, and Y. Liang. Bilevel optimization: Convergence analysis and enhanced design. In *ICML*, July 2021.
- [90] C. Vahapoglu, M. Morteheb, and S. Ulukus. Hierarchical over-the-air fedgrad-norm. In *IEEE Asilomar*, October 2022.
- [91] M. Morteheb, C. Vahapoglu, and S. Ulukus. Fedgradnorm: Personalized federated gradient-normalized multi-task learning. In *IEEE SPAWC*, July 2022.
- [92] J. Wang, A. K. Sahu, Z. Yang, G. Joshi, and S. Kar. Matcha: Speeding up decentralized sgd via matching decomposition sampling. In *Sixth Indian Control Conference*, December 2019.

- [93] A. Bellet, A. M. Kermarrec, and E. Lavoie. D-cliques: Compensating non-idness in decentralized federated learning with topology. Available online at arXiv:2104.07365.
- [94] G. Neglia, C. Xu, D. Towsley, and G. Calbi. Decentralized gradient methods: does topology matter? In *AISTATS*, August 2020.
- [95] T. Vogels, H. Hendrikx, and M. Jaggi. Beyond spectral gap: The role of the topology in decentralized learning. Available online at arXiv:2206.03093.
- [96] B. Le Bars, A. Bellet, M. Tommasi, E. Lavoie, and A. M. Kermarrec. Refined convergence and topology learning for decentralized optimization with heterogeneous data. Available online at arXiv:2204.04452.
- [97] Y. Dandi, A. Koloskova, M. Jaggi, and S. U. Stich. Data-heterogeneity-aware mixing for decentralized learning. Available online at arXiv:2204.06477.
- [98] C. Fifty, E. Amid, Z. Zhao, T. Yu, R. Anil, and C. Finn. Efficiently identifying task groupings for multi-task learning. In *NeurIPS*, December 2021.
- [99] U. V. Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, August 2007.
- [100] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *ICCV*, December 2015.
- [101] A. Koloskova, S. Stich, and M. Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *ICML*, June 2019.
- [102] A. Koloskova, N. Loizou, S. Boreiri, M. Jaggi, and S. Stich. A unified theory of decentralized sgd with changing topology and local updates. In *ICML*, July 2020.
- [103] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.
- [104] J. Bernstein, Y. X. Wang, K. Azizzadenesheli, and A. Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *ICML*, July 2018.
- [105] W. Wen, C. Xu, F. Yan, C. Wu, Y. Wang, Y. Chen, and H. Li. Terngrad: Ternary gradients to reduce communication in distributed deep learning. *Neurips*, December 2017.
- [106] D. Basu, D. Data, C. Karakus, and S. Diggavi. Qsparse-local-sgd: Distributed sgd with quantization, sparsification and local computations. *Neurips*, December 2019.
- [107] S. U. Stich, J. B. Cordonnier, and M. Jaggi. Sparsified sgd with memory. *Neurips*, December 2018.

- [108] M. Morafah, S. Vahidian, W. Wang, and B. Lin. Flis: Clustered federated learning via inference similarity for non-iid data distribution. *Available online at arXiv:2208.09754*.
- [109] F. Sattler, K. R. Müller, and W. Samek. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE Transactions on Neural Networks and Learning Systems*, 32(8):3710–3722, August 2020.
- [110] H. H. Yang, A. Arafa, T. Q. Quek, and H. V. Poor. Age-based scheduling policy for federated learning in mobile edge networks. In *IEEE ICASSP*, May 2020.
- [111] E. Ozfatura, B. Buyukates, D. Gündüz, and S. Ulukus. Age-based coded computation for bias reduction in distributed learning. In *IEEE Globecom*, December 2020.
- [112] B. Buyukates and S. Ulukus. Timely distributed computation with stragglers. *IEEE Transactions on Communications*, 68(9):5273–5282, September 2020.
- [113] K. Wang, Y. Ma, M. B. Mashhadi, C. H. Foh, R. Tafazolli, and Z. Ding. Age of information in federated learning over wireless networks. *Available online at arXiv:2209.06623*.
- [114] B. Buyukates and S. Ulukus. Timely communication in federated learning. In *IEEE Infocom*, May 2021.
- [115] M. Ester, H. Kriegel, J. Sander, X. Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, volume 96, pages 226–231, 1996.
- [116] S. Kaul, M. Gruteser, V. Rai, and J. Kenney. Minimizing age of information in vehicular networks. In *IEEE SECON*, June 2011.
- [117] Y. Sun, I. Kadota, R. Talak, and E. Modiano. Age of information: A new metric for information freshness. *Synthesis Lectures on Communication Networks*, 12(2):1–224, December 2019.
- [118] R. D. Yates. The age of gossip in networks. In *IEEE ISIT*, July 2021.
- [119] B. Abolhassani, J. Tadrous, A. Eryilmaz, and E. Yeh. Fresh caching for dynamic content. In *IEEE Infocom*, May 2021.
- [120] M. Bastopcu and S. Ulukus. Who should Google Scholar update more often? In *IEEE Infocom*, July 2020.
- [121] B. Buyukates, M. Bastopcu, and S. Ulukus. Age of gossip in networks with community structure. In *IEEE SPAWC*, September 2021.

- [122] A. Srivastava and S. Ulukus. Age of gossip on a grid. In *Allerton Conference*, September 2023.
- [123] P. Kaswan, P. Mitra, A. Srivastava, and S. Ulukus. Age of information in gossip networks: A friendly introduction and literature survey. 2023. Available at arXiv:2312.16163.
- [124] A. Javani, M. Zorgui, and Z. Wang. Age of information for multiple-source multiple-server networks. arXiv preprint arXiv:2106.07247.
- [125] R. D. Yates and S. Kaul. Real-time status updating: Multiple sources. In *IEEE ISIT*, pages 2666–2670, July 2012.
- [126] P. Kaswan and S. Ulukus. Age of gossip in ring networks in the presence of jamming attacks. In *IEEE Asilomar*, October 2022.
- [127] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- [128] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.
- [129] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. *Advances in NIPS*, 2024.
- [130] W. Dai, J. Li, D. LI, A. Tiong, J. Zhao, W. Wang, B. Li, P. N Fung, and S. Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. In *Advances in NIPS*, 2023.
- [131] J. Lin, H. Yin, W. Ping, P. Molchanov, M. Shoeybi, and S. Han. Vila: On pre-training for visual language models. In *CVPR*, 2024.
- [132] G. Team, R. Anil, S. Borgeaud, Y. Wu, J. B. Alayrac, J. Yu, R. Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.
- [133] S. Min, K. Krishna, X. Lyu, M. Lewis, W. Yih, P. W. Koh, M. Iyyer, L. Zettlemoyer, and H. Hajishirzi. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. *arXiv preprint arXiv:2305.14251*, 2023.
- [134] Y. S. Chuang, L. Qiu, C. Y. Hsieh, R. Krishna, Y. Kim, and J. Glass. Lookback lens: Detecting and mitigating contextual hallucinations in large language models using only attention maps. *arXiv preprint arXiv:2407.07071*, 2024.
- [135] Y. F. Bakman, D. N. Yaldiz, B. Buyukates, C. Tao, D. Dimitriadis, and S. Avestimehr. Mars: Meaning-aware response scoring for uncertainty estimation in generative llms. *arXiv preprint arXiv:2402.11756*, 2024.

- [136] A. Gunjal, J. Yin, and E. Bas. Detecting and preventing hallucinations in large vision language models. In *AAAI*, 2024.
- [137] L. Jing, R. Li, Y. Chen, M. Jia, and X. Du. Faithscore: Evaluating hallucinations in large vision-language models. *arXiv preprint arXiv:2311.01477*, 2023.
- [138] C. Jiang, W. Ye, M. Dong, H. Jia, H. Xu, M. Yan, J. Zhang, and S. Zhang. Hal-eval: A universal and fine-grained hallucination evaluation framework for large vision language models. *arXiv preprint arXiv:2402.15721*, 2024.
- [139] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.
- [140] Z. Chen, C. Xu, Y. Qi, and J. Guo. Mllm is a strong reranker: Advancing multimodal retrieval-augmented generation via knowledge-enhanced reranking and noise-injected training. *arXiv preprint arXiv:2407.21439*, 2024.
- [141] L. Qu, M. Liu, W. Wang, Z. Zheng, L. Nie, and T. S. Chua. Learnable pillar-based re-ranking for image-text retrieval. In *ACM SIGIR*, 2023.
- [142] S. Xu, D. Hou, L. Pang, J. Deng, J. Xu, H. Shen, and X. Cheng. Ai-generated images introduce invisible relevance bias to text-image retrieval. *arXiv preprint arXiv:2311.14084*, 2023.
- [143] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M. A. Lachaux, T. Lacroix, B. Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
- [144] N. Ziegler, D. M. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.
- [145] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer. Sigmoid loss for language image pre-training. In *IEEE/CVF ICCV*, 2023.
- [146] M. Ren, R. Kiros, and R. Zemel. Exploring models and data for image question answering. *Advances in NIPS*, 2015.
- [147] J. Li, D. Li, C. Xiong, and S. Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, 2022.
- [148] T. Stockhammer. Dynamic adaptive streaming over http— standards and design principles. In *Proceedings of the second annual ACM conference on Multimedia systems*, pages 133–144, 2011.

- [149] *Apple HTTP Live Streaming*. Accessed: Sep. 25, 2014. [Online]. Available: <https://developer.apple.com/resources/http-streaming/>.
- [150] *Microsoft Smooth Streaming*. Accessed: Sep. 25, 2014. [Online]. Available: <http://www.iis.net/downloads/microsoft/smooth-streaming>.
- [151] *Adobe HTTP Dynamic Streaming*. Accessed: Sep. 25, 2014. [Online]. Available: <http://www.adobe.com/products/hds-dynamic-streaming.html>.
- [152] C. Feichtenhofer. X3d: Expanding architectures for efficient video recognition. In *CVPR*, pages 203–213, June 2020.
- [153] Y. Wu and K. He. Group normalization. In *ECCV*, September 2018.
- [154] D. Hendrycks and K. Gimpel. Gaussian error linear units (gelus). Available online at [arXiv:1606.08415](https://arxiv.org/abs/1606.08415).
- [155] *QCIF dataset*. Available: http://vip.cs.nctu.edu.tw/resource_seq.html.
- [156] M. Mortaheb and S. Ulukus. Personalized decentralized multi-task learning over dynamic communication graphs. In *IEEE CISS*, March 2023.
- [157] M. Mortaheb, P. Kaswan, and S. Ulukus. r age- k : Communication-efficient federated learning using age factor. In *IEEE Asilomar*, October 2024.
- [158] K. J. Khojastepour, M. Mortaheb, and S. Ulukus. Age of gossip in networks with multiple views of a source. In *IEEE WCNC*, March 2025.
- [159] M. Mortaheb, M. A. Khojastepour, S. T. Chakradhar, and S. Ulukus. Rag-check: Evaluating multimodal retrieval augmented generation performance. In *AAAI workshop*, March 2025.
- [160] M. Mortaheb, M. A. Khojastepour, S. T. Chakradhar, and S. Ulukus. Re-ranking the context for multimodal retrieval augmented generation. In *ACM SIGIR workshop*, July 2025.
- [161] M. Mortaheb, M. A. Khojastepour, S. T. Chakradhar, and S. Ulukus. Deep learning-based real-time rate control for live streaming on wireless networks. In *IEEE ICMLCN*, May 2024.
- [162] M. Mortaheb, M. A. Khojastepour, S. T. Chakradhar, and S. Ulukus. Deep learning-based real-time quality control of standard video compression for live streaming. In *IEEE ICC*, June 2024.