

# A Journey of Personalization-Aware Collaborative Multimodal Machine Learning

---

A

*Dissertation*

Presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

---

in partial fulfillment

of the requirements for the degree

Doctor of Philosophy

by

Jiayi Chen

May 2024

# APPROVAL SHEET

This  
Dissertation  
is submitted in partial fulfillment of the requirements  
for the degree of  
Doctor of Philosophy

Author: Jiayi Chen

Advisor: Dr. Aidong Zhang

Advisor:

Committee Member: Dr. Hongning Wang

Committee Member: Dr. Yangfeng Ji

Committee Member: Dr. Jundong Li

Committee Member: Dr. Stefan Bekiranov

Committee Member:

Committee Member:

Accepted for the School of Engineering and Applied Science:



Jennifer L. West, School of Engineering and Applied Science

May 2024



© Copyright by

Jiayi Chen

2024

# Acknowledgments

I would like to thank many people who have helped me along my path of pursuing a Ph.D. degree in computer science.

First and foremost, I would especially like to thank my advisor Professor Aidong Zhang, for taking me under her supervision, and for running a great lab where everyone works on cutting-edge research problems. She has been a role model researcher who is always passionate about new challenges. She is very knowledgeable and extremely professional. What makes her an even greater advisor is that she truly cares about her students. I consider myself very fortunate to have her as my advisor. Her knowledge, advice, support and encouragement have greatly helped me during my Ph.D. study.

I would like to thank my committee members Professor Hongning Wang, Professor Yangfeng Ji, Professor Jundong Li, and Professor Stefan Bekiranov for their valuable suggestions for my research and for shaping my dissertation.

I also appreciate the support and help from my collaborators, lab mates, parents, and friends. I would like to thank my collaborators and mentors during my internships, Wei Wei, Hanjun Dai, Bo Dai, Zhe Zhao, Shuo Yang, and Hussein Hazimeh, for being very instructive and providing insightful advice. I would like to thank my lab mates, Guangxu Xun, Mengdi Huai, Kishlay Jha, Qiuling Suo, Guangtao Zheng, Jianhui Sun, Mia Shu, Hyun Jae Cho, Guangzhi Xiong, Sanchit Sinha, and many more whose names are not included here. I would especially like to thank my parents, Huibo Chen and Ying Kuang, for raising me to value education and always being supportive on my Ph.D. journey. I would also like to thank my friends, Biying Wang, Shili Sheng, Xin Shu, Jing Ma, and many others, for their supports and encouragements.

## Abstract

There has been a surge of interest in Multimodal Machine Learning (MML) in recent years. MML focuses on building frameworks for understanding or synthesizing the multimodal world and is one of significant fields paving the way toward artificial general intelligence. However, in the field of MML, the practice of learning through multi-agent collaboration, namely Collaborative Multimodal Machine Learning (CoMML), remains relatively underexplored. CoMML draws the inspiration from the peer-to-peer learning behavior observed in humans and aims to build frameworks that enable multiple MML agents to collaborate with each other efficiently, achieving a comprehensive understanding of the multimodal world that surpasses what any individual agent could achieve alone. This thesis conducts a systematic study on CoMML, and particularly, we emphasize the importance of Personalization in CoMML focusing on supporting the special needs and prerequisites of individual MML agents throughout all stages of collaborative learning.

However, achieving effective personalization while encouraging efficient collaboration in the multimodal learning system poses many practical challenges. This thesis specifically endeavors to pursue the following research goals in Personalization-aware CoMML. (1) First, we investigate a variety of technical challenges during collaboration brought about by user personalization. The various user needs and capabilities among learning agents lead to increased knowledge diversity, which complicates the discovery of shareable knowledge among individual agents. We aim to achieve a good balance between personalization and collaboration under such knowledge heterogeneity. (2) Second, the patterns of personalization may vary dramatically depending on application scenarios, users' unique attributes and roles, locality constraints, system budgets, and other factors. This thesis investigates different personalization patterns, including modality, task, concept, and architecture preferences, and further explores the necessary technical efforts and proposes novel approaches for dealing with each pattern. (3) Third, we aim to advance the compatibility and applicability of personalization-aware CoMML frameworks in multimodal general intelligence. A broad range of modality types, including language, image, audio, video, 3d shapes, and diverse types of multimodal learning tasks, will be studied within our frameworks.

# Contents

Acknowledgments . . . . .	i
Abstract . . . . .	ii
List of Tables . . . . .	vii
List of Figures . . . . .	viii
<b>1 Introduction . . . . .</b>	<b>1</b>
1.1 Motivations and Research Problem . . . . .	1
1.2 Thesis Overview . . . . .	4
1.3 Contributions . . . . .	6
<b>2 Background and Literature Review . . . . .</b>	<b>7</b>
2.1 Notations . . . . .	7
2.2 Multimodal Machine Learning (MML) . . . . .	8
2.2.1 Fundamental Principles of Multi-modality . . . . .	8
2.2.2 Basic Network Components . . . . .	9
2.3 Collaborative Machine Learning . . . . .	11
2.3.1 Learning Paradigms . . . . .	11
2.3.2 Collaborative MML (CoMML) . . . . .	13
2.3.3 Personalization in Collaboration . . . . .	14
2.4 Literature Review . . . . .	16
<b>I Collaboration with Graph and Meta Learning . . . . .</b>	<b>21</b>
<b>3 Heterogeneous Hypernode Graph Learning for Collaborative In-</b>	
<b>formation Fusion with Modality Gap . . . . .</b>	<b>22</b>
3.1 Introduction . . . . .	22
3.2 Related Work . . . . .	25

3.3	Problem Formulation . . . . .	26
3.4	Methodology . . . . .	27
3.4.1	Overview . . . . .	27
3.4.2	Modeling Collaboration in Heterogeneous Graph . . . . .	28
3.4.3	Intra-hypernode Encoder . . . . .	31
3.4.4	Multi-fold bilevel Graph Attentional Layer . . . . .	34
3.5	Experiments . . . . .	39
3.5.1	Simulations . . . . .	39
3.5.2	Baseline Models . . . . .	41
3.5.3	Experimental Setup . . . . .	42
3.5.4	Results and Analysis . . . . .	44
3.6	Conclusion . . . . .	46
<b>4</b>	<b>Meta-learning based Collaborative Visual-language Understanding with Concept Personalization . . . . .</b>	<b>47</b>
4.1	Introduction . . . . .	47
4.2	Related Works . . . . .	49
4.3	Problem Formulation . . . . .	51
4.3.1	Local Task Setting on Each Agent . . . . .	51
4.3.2	Multi-agent Collaborative EpVDER . . . . .	53
4.4	Methodology . . . . .	54
4.4.1	Transformer-based Multimodal Encoder . . . . .	55
4.4.2	Task-dependent Embedding Space . . . . .	56
4.4.3	Token Labelling . . . . .	56
4.4.4	Self-aware Meta Learners . . . . .	56
4.5	Experiments . . . . .	60
4.5.1	Dataset . . . . .	60
4.5.2	Setups and Evaluation Metrics . . . . .	62
4.5.3	Main Results . . . . .	63
4.5.4	Class Structure Disentanglement . . . . .	65
4.6	Conclusions . . . . .	67
<b>5</b>	<b>Meta Graph Learning for Handling Coexistence of Modality Gap and Concept Shift . . . . .</b>	<b>68</b>
5.1	Introduction . . . . .	68
5.2	Related Work . . . . .	71
5.3	Problem Formulation . . . . .	73
5.3.1	Local Task Definition on Each Agent . . . . .	73

5.3.2	Multi-agent Collaborative hFSL . . . . .	74
5.4	TopoNet: Graph Transductive Meta-learner . . . . .	75
5.4.1	Feature Embedding . . . . .	76
5.4.2	Topological Transductive Learning . . . . .	77
5.4.3	Optimization . . . . .	82
5.5	Experiments . . . . .	84
5.5.1	Dataset . . . . .	84
5.5.2	Baseline Methods . . . . .	85
5.5.3	Setups . . . . .	85
5.5.4	Results and Analysis . . . . .	86
5.6	Conclusion . . . . .	87

## II Collaboration with Explicit Knowledge Transfer 88

<b>6</b>	<b>Splittable and Adaptive Transfer for Fast Collaboration . . . . .</b>	<b>89</b>
6.1	Introduction . . . . .	89
6.2	Problem Formulation . . . . .	92
6.3	Proposed FedMSplit . . . . .	95
6.3.1	Correlation-adaptive Model Update . . . . .	96
6.3.2	Federated Training . . . . .	99
6.4	Experiments . . . . .	103
6.4.1	Simulations . . . . .	103
6.4.2	Baselines . . . . .	105
6.4.3	Main Results . . . . .	105
6.4.4	Ablation Study . . . . .	106
6.5	Conclusion . . . . .	108
<b>7</b>	<b>Disentangled and Gated Transfer for Asymmetrical Collaboration</b>	<b>109</b>
7.1	Introduction . . . . .	109
7.2	Related Works . . . . .	111
7.3	Problem Formulation . . . . .	112
7.3.1	Modality-task Agnostic Federated Learning (AFL) . . . . .	112
7.3.2	Four Heterogeneity Patterns in AFL . . . . .	113
7.4	Asymmetrical Knowledge Transfer . . . . .	114
7.5	Proposed DisentAFL . . . . .	116
7.5.1	Stage One: Coarse-grained Disentanglement . . . . .	118
7.5.2	Stage Two: Fine-grained Disentanglement . . . . .	119

7.5.3	Sparsely-gated Collaboration via Network Routing . . . . .	124
7.6	Details and Proofs . . . . .	126
7.6.1	Global Knowledge Type Design . . . . .	126
7.6.2	Proofs . . . . .	128
7.6.3	Pseudocode and Workflow . . . . .	133
7.7	Experiments . . . . .	134
7.7.1	Simulations . . . . .	134
7.7.2	Baseline Methods . . . . .	136
7.7.3	Setups . . . . .	137
7.7.4	Results . . . . .	137
7.8	Conclusions . . . . .	139

### **III Collaboration with Implicit Knowledge Transfer 140**

<b>8</b>	<b>Latent Transfer for Architecture-free Collaboration . . . . .</b>	<b>141</b>
8.1	Introduction . . . . .	141
8.2	Related Works . . . . .	144
8.3	Problem Formulation . . . . .	146
8.3.1	Local Task Setting on Each Agent . . . . .	146
8.3.2	Multi-agent Collaboration with Architecture Gap . . . . .	146
8.4	Proposed FedMBridge . . . . .	148
8.4.1	Main Idea . . . . .	149
8.4.2	Multimodal Neural Architectures as Graphs . . . . .	150
8.4.3	Topology-aware HyperNetwork . . . . .	155
8.4.4	Workflow . . . . .	157
8.5	Experiments . . . . .	158
8.5.1	Simulations . . . . .	158
8.5.2	Baseline Methods . . . . .	159
8.5.3	Setups . . . . .	163
8.5.4	Main Results . . . . .	164
8.6	Conclusion . . . . .	167
<b>9</b>	<b>Conclusion and Future Directions . . . . .</b>	<b>168</b>
9.1	Conclusion . . . . .	168
9.2	Future Directions . . . . .	169
9.3	Broader Impact . . . . .	170

<b>Bibliography . . . . .</b>	<b>173</b>
-------------------------------	------------

# List of Tables

1.1.1 Thesis Overview . . . . .	5
2.1.1 Notations . . . . .	7
3.5.1 Experimental Simulation of Modality Gap . . . . .	40
3.5.2 Hyperparameters . . . . .	42
3.5.3 Main Results P1 . . . . .	44
3.5.4 Main Results P2 . . . . .	45
4.5.1 Simulation of Fine-grained Concept Shift. . . . .	61
4.5.2 Main Results on FewVEX . . . . .	64
5.5.1 Results on Hybrid Multimodal Few-shot Classification . . . . .	86
5.5.2 Hyperparameter and ablation studies on 5-way 1-shot h-CUB-200. . . . .	87
6.4.1 Statistics of Multimodal Federated Simulations. . . . .	104
6.4.2 Main Results . . . . .	106
6.4.3 Ablation study of FedMSplit . . . . .	107
7.7.1 Statistics of 5 Source Datasets. . . . .	135
7.7.2 Statistics of the three AFL Simulations . . . . .	135
7.7.3 Average testing accuracy over all agents' classification tasks at $T$ . . . . .	138
8.5.1 Summary of the 4 simulations of AMFL . . . . .	159
8.5.2 Illustrations of Statistical and Architectural Heterogeneity . . . . .	160
8.5.3 Main Results . . . . .	165
8.5.4 Ablation Study for FedMBridge using MnistAMF dataset. . . . .	166



# List of Figures

2.2.1 Principles of Multi-modality . . . . .	9
2.2.2 An example pipeline of Multimodal Learning . . . . .	10
3.1.1 An Illustration of Modality Preferences . . . . .	23
3.4.1 Overview of Heterogeneous Graph-based Multimodal Fusion (HGMF)	28
3.4.2 HHG graph construction in HGMF. . . . .	30
3.4.3 Architecture of intra-hypernode encoder in HGMF. . . . .	32
3.4.4 Multi-fold bilevel Graph Attentional Layers in HGMF. . . . .	35
4.3.1 Visually-rich Document Entity Retrieval via Multi-agent Collabora- tion with Concept Preferences . . . . .	53
4.4.1 The proposed self-aware meta-learning framework . . . . .	55
4.5.1 tSNE visualization of the learned embedding space . . . . .	65
4.5.2 Overall Visualization . . . . .	66
5.1.1 Illustration of simultaneous modality and concept preferences . . . .	69
5.4.1 The proposed TopoNet framework. . . . .	76
6.2.1 Multimodal federated learning setting. . . . .	92
6.2.2 A graph view of CoMML among different splittable models . . . . .	95
6.3.1 Overview of FedMSplit workflow . . . . .	101
7.1.1 Modality-task Agnostic Federated Learning (AFL) . . . . .	110
7.4.1 Comparison between symmetrical and asymmetrical inter-agent knowl- edge relationships . . . . .	115
7.5.1 Overview of the proposed DisentAFL. . . . .	117
7.5.2 The two-stage Knowledge Disentanglement with Gated Collabora- tion mechanism in DisentAFL. . . . .	118
7.5.3 Central Supernetwork Architecture. . . . .	120
7.5.4 An example of Disentangled and Gated Collaboration. . . . .	125
7.7.1 Baseline Models. . . . .	136

8.1.1 Architecture-personalized Multimodal Federated Learning (AMFL)	142
8.4.1 The proposed FedMBridge framework . . . . .	152
8.5.1 Baseline model . . . . .	161

# Chapter 1

## Introduction

### 1.1 Motivations and Research Problem

The world is multimodal—things happen and are perceived by us in different ways. For example, when learning the concept of an object, we hear its sounds, observe its shapes, feel its texture, smell its odors, and more. In this **multimodal world**, humans develop our cognitive abilities of analyzing multimodal information and advance our interaction experiences with the environments. In recent years, a broad community of researchers has emerged in the field of Artificial General Intelligence (AGI) focusing on the ambitious goal of emulating human-like cognitive abilities in the multimodal world [1, 2, 3]. **Multimodal Machine Learning (MML)** is one of significant fields paving the way toward AGI, focusing on building frameworks for understanding or synthesizing the multimodal world. The general MML research covers a broad range of modality types, including but not limited to language, image, audio, video, 3d motions, and graphs, and there are a variety of research subfields associated with MML, scattered across different domains depending on the modalities involved and how they combine to form task settings [4]. For instance, in cases involving audios and images, tasks may include audio-visual object recognition or retrieval, where both modalities serve as inputs, or image-to-audio synthesis when images serve as inputs and audios are the outputs.

Humans are not isolated in the world. We are members in a larger community, wherein we learn knowledge not only based on our own experiences (i.e., direct

learning) but also from the experiences of others in a peer-to-peer learning manner (i.e., indirect learning) [5]. Such a **peer-to-peer learning** community is essential for humans due to the limited learning capabilities and diverse backgrounds of individuals—i.e., individuals’ learning scopes are restricted by where we are (how we perceive the world) and the historical era to which we belong. Similar to the learning limitations of individual humans, individual machine learning agents also encounter constraints such as data limitations and system constraints, which impedes the ability of a single AGI agent to generalize the diverse knowledge in the world. In response to this challenge, recent advancements of Multi-agent Collaborative Machine Learning [6, 7, 8, 9] has drawn inspiration from the peer-to-peer learning behavior observed in humans: the goal is to leverage the collaboration of multiple AI agents, each of which have they own views and learning behaviors, to potentially expand the learning abilities of individual AI agents. However, in the field of AGI and MML, the practice of learning through multi-agent collaboration remains relatively underexplored. To address this gap, this thesis focuses on a systematic study of **Collaborative Multimodal Machine Learning (CoMML)**, which aims to build frameworks that enable multiple MML agents to collaborate with each other efficiently, achieving a comprehensive understanding of the multimodal world that surpasses what any individual agent could achieve alone.

While humans collaborate within groups, the satisfaction of our individual needs is also important. Preserving our individuality during collaboration is an inherent human ability [10]. Likewise, in CoMML, individuality also plays a significant role, particularly when individual learning agents are utilized to serve specific users that have unique and strong preferences and abilities related to data, concepts, and model performance. Therefore, it is essential to emphasize the importance of **Personalization in CoMML**, which focuses on supporting the special needs and prerequisites of individual MML agents throughout all stages of collaborative learning. However, achieving effective personalization within a collaborative multimodal learning system poses many practical challenges. (1) First, personalization implies *heterogeneity*—the various user needs and capabilities among learning agents lead to increased knowledge diversity. This heterogeneity among agents complicates the discovery of shareable knowledge among individual agents, mak-

ing it challenging to achieve an appropriate balance between personalization and collaboration performance. (2) Second, the *patterns of personalization* may vary dramatically depending on application scenarios, users’ unique attributes and roles, locality constraints, system budgets, and other factors. For example, in some cases, agents may have different device setups, resulting in varied input modality types. In other cases, agents might be used to serve different target tasks. Given different personalization patterns, the algorithms governing how agents collaborate may have various objectives and evaluation tradeoffs, highlighting the necessity for systematic exploration of which collaboration mode fits each pattern.

Motivated by the above challenges, this thesis endeavors to pursue the following research goals in Personalization-aware CoMML. (1) The first is to tackle the **heterogeneity** challenge caused by personalization, aiming to find appropriate strategies for knowledge sharing among diversified MML agents to achieve a good balance between personalization and collaboration. (2) The second is to investigate **different personalization patterns** in real-world scenarios and to further determine the necessary technical efforts required for each pattern. Specifically, we will consider the following five possible personalization patterns between agents:

- **Modality Preference:** Agents have specific input modality types or construction of multiple modalities due to their distinct device setups, the job environments, network connections, sensor affordability, and use cases. For example, in a driving scenario with two vehicles connected and collaborating with each other; a vehicle may use its onboard camera to capture videos to predict traffics, and another vehicle may use both video and RADAR signals to predict traffics.
- **Task Preference:** Agents may target different downstream tasks based on their use cases. For example, while an agent deals with image classification, another agent may focus on audio synthesis conditioned on the input image.
- **Domain Preference:** Agents are situated in diverse environments, resulting in unique perceptions shaped by specific biases in what they see, hear, or feel. This includes two cases: i) Domain preferences within each modality type, such as object rotations or environmental lighting in images, and background noise or speaker identity in audios. ii) Varied multimodal interaction behaviors among agents. For example, given two agents equipped with both image and text modal-

ities, the common knowledge shared by a pair of image and text may be within the background for one agent but around the contours of objects for the other.

- **Concept Preference:** Agents have their own target class categories, distinct definitions for concepts, or prior knowledge during decision-making. For example, many users are only interested in a minority of classes. Additionally, different agents might assign different label IDs to the same object. These concept preferences result in shifts in label spaces among agents.
- **Architecture Preference:** Agents have the freedom to specify their model architectures, resulting in differences in model depth, width, or topology (i.e., the family of neural network architecture). There are four real-world reasons motivating the necessity of considering architecture preference. i) *Task complexity* varies among agents, leading some to require larger neural network sizes or even varied network families to address more complex problems. ii) *Inter-modal interaction mechanism* can vary across agents, necessitating distinct network connections (e.g., concatenation, element-wise product, cross-attention fusion, tensor fusion) for effectively capturing such interactions [11]. iii) The *computation resource budget* can also vary greatly across agents, and the computation device might range from mobile phones and tablets to personal computers. This imposes restrictions on the sizes of model architectures for agents. iv) In continual development workflows, the *model architecture can be modified* over time [12] or even embedded in neural architecture search loops [13]. In such cases, consideration of architecture preference supports such extension flexibility.

Finally, **(3)** we aim to advance the compatibility and applicability of Personalization-aware CoMML frameworks in multimodal general intelligence. A broad range of modality types, including language, image, audio, video, 3D shapes, and diverse types of multimodal learning tasks, will be studied within our frameworks.

## 1.2 Thesis Overview

An overview of this thesis is summarized in Table 1.1.1. We unfold 9 research topics centered around Personalization-aware CoMML, which are categorised into three parts: We **begin** with relatively simple personalization patterns—the modality

Table 1.1.1: Thesis Overview. Research topics are centered around Personalization-aware Collaborative Multimodal Machine Learning, addressing various technical challenges arising from the existence (or potential coexistence) of five personalization patterns: Modality Preference (**M**), Concept Preference (**C**), Domain Preference (**D**), Task Preference (**T**), and Architecture Preference (**A**).

Individual Preference	Collaboration Mode		Chapters	Keywords
	Shareability	Final Goal		
M	Data	General Model	Chapter 3	Heterogeneous Graph
C	Meta-gradients		Chapter 4	Contrastive, OOD
M+C			Chapter 5	Graph Meta Learner
M+C+D	Parameter Weights	General Model + Personal Models	Chapter 6	Split Transfer
M+C+D+T			Chapter 6	Agent Sampling
			Chapter 7	Gated Transfer
			Chapter 7	Disentanglement
M+C+D+A	Inference Outputs		Chapter 8	Co-Distillation
	Latent Knowledge		Chapter 8	Hypernetwork

preferences and concept preferences, in Part I (Chapter 3, Chapter 4, and Chapter 5), wherein we address the heterogeneity by formulating the collaboration problem within a *centralized* global training paradigm and explore graph-based, contrastive, and meta-learning approaches. **Then**, in Part II (Chapter 6 and Chapter 7), we turn to the *global-local* personalized collaborative learning paradigm to more effectively address complex agent heterogeneity, particularly when introducing two additional personalization patterns—domain preferences and task type preferences. We explore approaches for *explicitly* optimizing knowledge transfer among heterogeneous agents. **Finally**, in Part III (Chapter 8), we incorporate architecture preferences into the system, wherein we continue to utilize the global-local decentralized collaborative learning paradigm but switch to explore *implicit* information sharing approaches to optimize knowledge transfer among heterogeneous agents. In addition to the aforementioned three parts, Chapter 2 offers background information and a literature review relevant to all these chapters. Chapter 9 delves into broader impacts, future directions in CoMML and personalization approaches, and provide a summary of this thesis.

### 1.3 Contributions

The contributions of this thesis are summarized as follows:

- Exploring **multi-agent collaboration** to advance the frontier of understanding the multimodal world remains an underexplored research direction. To the best of our knowledge, this thesis is one of the **first** endeavors to systematically study in this direction, namely Collaborative Multimodal Machine Learning (CoMML).
- To improve applicability, we formulate and address a variety of real-world scenarios of CoMML by considering **multiple personalization patterns**, ranging from modality preferences to architecture preferences. These scenarios might involve the existence of one of these patterns as well as the simultaneous existence of several personalization patterns. We hope such a comprehensive investigation will provide valuable insights for future research in this field.
- We propose several novel approaches to tackle the heterogeneity challenge and strike a balance between personalization and collaboration. We discuss challenges associated with each specific personalization patterns and introduce novel approaches to solve each of them.
  - We introduce graph-based, contrastive, and meta-learning approaches to handle the modality gap and adapt to concept shifts (Part I).
  - To address the more complex statistical heterogeneity, along with modality gap and downstream task differences, we propose *explicitly* optimizing positive knowledge transfer among agents through techniques such as split learning, gated knowledge sharing, and disentanglement (Part II).
  - To address the coexistence of statistical and architectural heterogeneity, we explore *implicit* collaboration, investigating feature-sharing methods and proposing a novel agent-bridgeable latent knowledge sharing approach (Part III).
- We conduct experiments on diverse simulated scenarios of personalization-aware CoMML, which involve various modality types (e.g., images, texts, 3D point clouds, videos, and audio signals) and different multimodal learning tasks (e.g., audio-video classification, scanned document span retrieval, multimedia emotion analysis, and image-conditioned audio generation). Experimental results demonstrate the efficacy of our proposed approach compared to baselines.



# Chapter2

## Background and Literature Review

### 2.1 Notations

Table 2.1.1 present the notations used throughout all chapters.

Table 2.1.1: Notations.

Notations	Descriptions
$N$	Number of Agents
$M$	Number of Modality Types across Agents
$O$	Number of Downstream Task Categories across Agents
$i$	Agent ID $i \in [N]$
$m$	Modality ID $m \in [M]$
$o$	Task ID $o \in [O]$
$\mathcal{X}^{(m)}$	The feature space associated with the modality $m$ , i.e., $\mathcal{X}^{(m)} = \mathbb{R}^{F_m}$
$\mathcal{I}_i$	A set of input modality categories at agent $i$ . In practice, $\mathcal{I}_i \subseteq [M]$
$\mathcal{O}_i$	A set of target task categories at agent $i$ . In practice, $\mathcal{O}_i \subseteq [O]$
$n_i$	Number of samples on each agent $i$
$\mathcal{D}_i$	Local dataset at agent $i$ , consisting of $n_i$ samples
$j$	Sample ID $j$ in the agent $i$
$\mathbf{x}_{ij}^{(m)}$	The modality $m$ within the input of sample $j$ on agent $i$
$\mathbf{y}_{ij}^{(o)}$	The target $o$ within the label of sample $j$ on agent $i$
$\tilde{\mathbf{x}}_{ij}$	The input of the sample $j$ on agent $i$ , where $\tilde{\mathbf{x}}_{ij} = \text{Join}(\mathbf{x}_{ij}^{(m)}   m \in \mathcal{I}_i)$
$\tilde{\mathbf{y}}_{ij}$	The label of the sample $j$ on agent $i$ , where $\tilde{\mathbf{y}}_{ij} = \{\mathbf{y}_{ij}^{(o)}   \mathbf{y} \in \mathcal{O}_i\}$
$\mathcal{A}_i$	The model architecture of agent $i$
$\boldsymbol{\theta}_i$	Personal model weights of agent $i$ , from the weight space $\boldsymbol{\theta}_i \in \mathbb{R}^{d_i}$
$f_{\mathcal{A}_i}(\cdot; \boldsymbol{\theta}_i)$	The input-to-output mapping function at agent $i$
$\widehat{\mathcal{D}}_{\text{colab}}$	A collection of all agents' data

## 2.2 Multimodal Machine Learning (MML)

### 2.2.1 Fundamental Principles of Multi-modality

Multi-modality refers to the various representational modes of a real-world instance or phenomenon. There are the two fundamental principles that have driven technical challenges and subsequent research innovations of MML [14, 15].

- **Principle 1 (Discrepancy between Modalities):** Multiple modalities (e.g., images, texts, audios, videos, 3D point clouds, 3D motions, graphs) are gathered from diverse sources. Consequently, when these different modalities represent the same real-world subject, they demonstrate **discrepancies** not only in their **(1) data structures** but also in the **(2) cognitive information conveyed by the representations**. For instance, when representing an apple, an image displaying an apple on a plate conveys its color, shape, and potential container holding it, whereas the written or spoken word “apple” contains linguistic knowledge and phonetic information.
- **Principle 2 (Connection between Modalities):** Multiple modalities, when representing the same real-world subject, are also **connected**; they often share partial cognitive information or are causally related to each other. **(1) Inter-modality correlation** exists when the values of one variable relate to the values of another. Statistically, this could lead to correlation—the degree in which elements are linearly related, or other non-linear associations. For example, two cognitive concepts may co-occur in two or more modalities. **(2) Inter-modal dependency** goes beyond correlation and requires an understanding of the statistical causality between modalities.

Figure 2.2.1 illustrate the two basic principles. With the two principles, multiple modalities that represent the same real-world subject provide **complementary information** that unveil the fundamental characteristics of the subject.

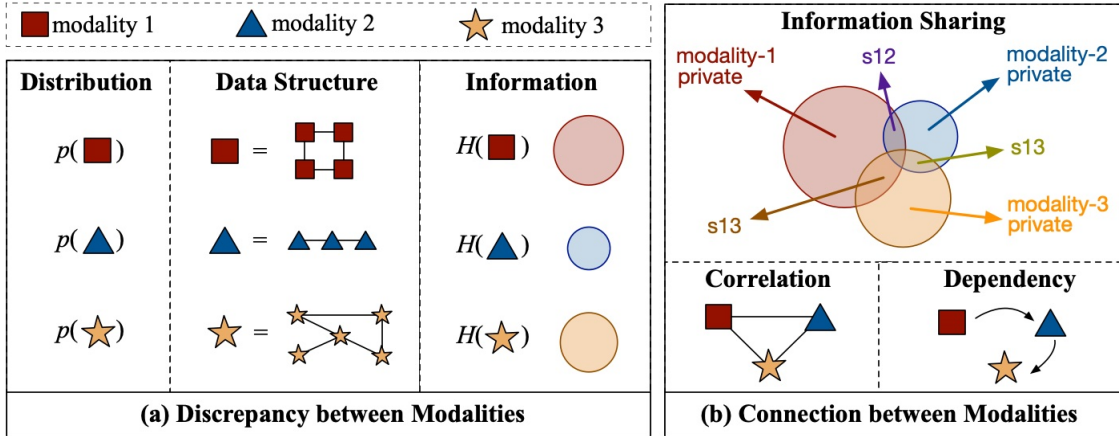


Figure 2.2.1: Principles of Multi-modality. Here shows a 3-modality case. The three modalities have discrepant properties as well as connected with each other.

## 2.2.2 Basic Network Components

MML is a broad field that studies a variety of tasks related to multimodal modalities, such as multimodal fusion, cross-modal data generation, cross-modal alignment, multimodal representation learning, cross-modal retrieval, multimodal grounding, and so on. Different MML tasks may employ distinct neural network architectures or leverage a unified general Transformer-based architecture [16, 17].

Based on the two principles mentioned in Section 2.2.1, there are several basic network components that are utilized to form various MML task networks.

- The Combinatorial Input.** Depending on task types and hardware setups, a subject or phenomenon can be represented by *any single* modality type or *jointly by a combination* of several modality types that provide complementary information. We denote the input of a sample as  $\tilde{\mathbf{x}} = \text{Join}(\mathbf{x}^{(m)} | m \in \mathcal{I})$ , where  $\mathbf{x}^{(m)} \in \mathbb{R}^{F_m}$  represents the modality  $m$  in  $\mathbf{x}$ ,  $F_m$  is the input dimension of the modality  $m$ , and  $\mathcal{I}$  denotes a set of input modality categories.  $|\mathcal{I}| = 1$  if the model is intended for a single-modal deterministic task or a cross-modal generation task, and  $|\mathcal{I}| = 2$  for a bimodal deterministic task or bimodal grounding.
- Unimodal Embedding & Encoder.** (1) *Unimodal embedding*, denoted as  $f_{\text{emb}}^{(m)}: \mathbb{R}^{F_m} \rightarrow \mathbb{R}^{L_m \times D_m}$ ,  $\forall m \in \mathcal{I}$ , aims for low-level feature extraction, or discretization, incorporating a trainable codebook. Each modality type has its own

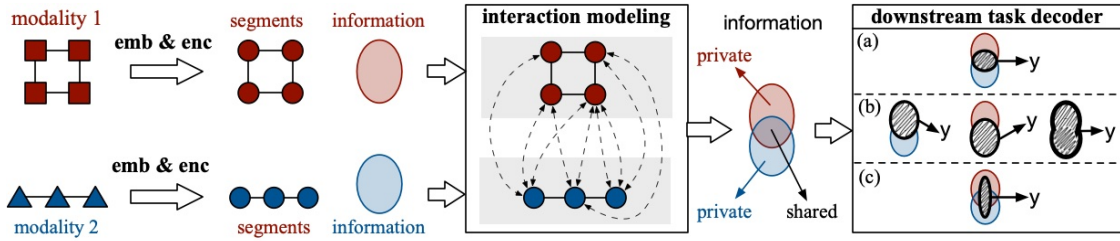


Figure 2.2.2: An example pipeline of Multimodal Learning for inference tasks on two input modalities.

representational structure so that different modalities are discretized in different ways [17]. For example, text data is tokenized using a vocabulary, while images are processed using patch embedding with a visual codebook. The output of the embedding consists of  $L_m$  ordered segments, each represented by an  $d_m$ -dimensional embedding vector, such as the token sequence in the text modality [18], the multi-scale spatially-ordered image patches of the image modality [19], and the multi-scale spatiotemporally-ordered 3d patches of the video modality. Note that multimodal networks that do not employ attention mechanisms typically may not incorporate the embedding process (i.e.,  $L_m = 1, F_m = D_m$ ). (2) *Unimodal Encoder*, denoted as  $f_{\text{enc}}^{(m)}: \mathbb{R}^{L_m \times D_m} \rightarrow \mathbb{R}^{L_m \times D'_m}, \forall m \in \mathcal{I}$ , aims for high-level feature extraction by modeling the intra-modal interactions. For example, the self-attention mechanism of Transformer-based encoders models the relationships between every pair of discretized segments [19]. If the Transformer model is too expansive, it is possible to resort to traditional modeling networks.

- **Inter-modality Interaction Modeling.** In scenarios where the input comprises multiple modalities, which complementarily demonstrate the fundamental characteristics of the subject, Inter-modality Interaction Modeling, denoted as  $f_{\text{inter}}: \mathbb{R}^{L_1 \times D'_1} \times \mathbb{R}^{L_2 \times D'_2} \dots \times \mathbb{R}^{L_{|\mathcal{I}|} \times D'_{|\mathcal{I}|}} \rightarrow \mathbb{R}^{L' \times \tilde{D}}$ , aims to model how different modalities are interconnected with each other. Addressing such high-order relationships is a key technical challenge that has sparked numerous research innovations in MML. Inter-modality interaction modeling typically consider the relationships between every pair of segments (or neurons) from two different modalities. Traditional inter-modality interaction models were leveraged manually-designed or non-parametric strategies like addition, element-wise product, cross

product, or low-rank product [20, 21]. Modern Transformer-based alternatives have explored cross-attention mechanisms to adaptively learn how modalities are connected [22]. The output format depends on the task for which the model is utilized:  $L' = 1$  for multimodal fusion tasks or simple non-autoregressive generative tasks, and  $L' > 1$  for sequential data generation tasks, where  $L'$  is the number of segments in the outputs needed by the downstream task.

- **Downstream Task Decoder.** A decision making function  $f_{\text{dec}}: \mathbb{R}^{L' \times \tilde{D}} \rightarrow \mathbb{R}^{L' \times F}$ , where  $F$  represents the feature space aligned with user needs. Again,  $L' = 1$  for multimodal fusion tasks or simple non-autoregressive generative tasks, and  $L' > 1$  for sequential data generation tasks, where  $L'$  denotes the number of segments in the outputs.

Figure 2.2.2 illustrates an example pipeline for a multimodal inference task from two modalities, consisting of the network components mentioned above.

## 2.3 Collaborative Machine Learning

### 2.3.1 Learning Paradigms

Collaborative Machine Learning emulates the **peer-to-peer learning** society of humans. It harnesses the collaboration between multiple AI agents, each of which have their own perspectives and learning behaviors, with the aim of potentially expanding the learning abilities of individual AI agents [6, 7, 8, 9].

**Setup of Agents:** Assuming there are  $N$  agents, each agent  $i = 1, 2, \dots, N$

- has its observations for the world within a **local dataset**  $\mathcal{D}_i = \{(\tilde{\mathbf{x}}_{ij}, \tilde{\mathbf{y}}_{ij})\}_{j=1}^{n_i}$ , sampled from an input distribution  $\tilde{\mathbf{x}}_{ij} \sim \mathcal{P}_i(\tilde{\mathbf{x}})$  and the conditional output distribution  $\tilde{\mathbf{y}}_{ij} \sim \mathcal{Q}_i(\tilde{\mathbf{y}}|\tilde{\mathbf{x}}_{ij})$ , where  $\mathcal{P}_i, \mathcal{Q}_i$  are *agent-specific distributions* over the *agent-specific input and output spaces*  $\mathcal{X}_i, \mathcal{Y}_i$ , respectively;
- has a **local mapping function**  $f_{\mathcal{A}_i}(\cdot; \boldsymbol{\theta}_i)$  characterized by its agent-specific model architecture  $\mathcal{A}_i$  parameterized by trainable weights  $\boldsymbol{\theta}_i \in \mathbb{R}^{d_i}$ , where  $d_i$  indicates the structure of the weight space associated with  $\mathcal{A}_i$ ; and
- aims to solve its **local objective**  $\min_{\boldsymbol{\theta}_i} \mathcal{L}_i(\boldsymbol{\theta}_i) := \mathbb{E}_{(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) \sim \mathcal{D}_i} l(\tilde{\mathbf{y}}, f_{\mathcal{A}_i}(\tilde{\mathbf{x}}; \boldsymbol{\theta}_i))$ , where  $l(\cdot, \cdot)$  is the loss function for each sample.

**Multi-agent Collaboration Paradigms:** Information sharing among agents can boost individual agents’ performance by providing a broader view of global generalization, allowing local unseen data to benefit from the knowledge of others [6, 7, 8, 9]. The collaboration procedure, which facilitates beneficial information sharing among agents, has often been formulated within two learning paradigms:

- **Global Centralized Paradigm** aims to learn a single *global* model that can easily *adapt to individual* models of each agent [23, 24, 25, 26, 27]. The **global objective** is to obtain a single global model  $\theta \in \mathbb{R}^d$  and an auxiliary adaption model  $\phi \in \mathbb{R}^{d'}$  to jointly minimize the local objectives using adapted models

$$\min_{\theta, \phi} \left[ \frac{1}{N} \sum_{i=1}^N \mathcal{L}_i(\text{Adapt}(\mathcal{D}_i, \theta; \phi)) \right] \quad (2.1)$$

where  $\text{Adapt}(\cdot, \cdot; \phi)$  is the process of adapting global model to agent-specific preferences.  $\text{Adapt}(\cdot, \cdot; \phi)$  takes two inputs—the global model and the personal task, and its role is to find a personal model  $\theta_i^* = \text{Adapt}(\mathcal{D}_i, \theta^*; \phi^*)$  with a good generalization-personalization balance based on the two inputs.

- **Global-local Decentralized Paradigm** [28, 29, 30] simultaneously learn the *individual* models  $\theta_1 \in \mathbb{R}^{d_1}, \theta_2 \in \mathbb{R}^{d_2}, \dots, \theta_N \in \mathbb{R}^{d_N}$  of each agent while encouraging *global knowledge transfer* among their learned knowledge. The **global objective** is

$$\min_{\theta_1, \theta_2, \dots, \theta_N, \phi} \left[ \frac{1}{N} \sum_{i=1}^N \mathcal{L}_i(\theta_i) \right] + \mathcal{R}(\theta_1, \theta_2, \dots, \theta_N; \phi), \quad (2.2)$$

which **(1)** jointly optimizes the *local objectives* of all agents and meanwhile **(2)** encourages a *global knowledge sharing scheme* among agents  $\mathcal{R}$  in order to boost each agent’s local model performance. A good balance between generalization and personalization is managed by  $\mathcal{R}$ , which controls the impact of knowledge transfer on local knowledge. Depending on the method of conducting knowledge sharing,  $\mathcal{R}$  can be either parametric or non-parametric ( $\phi = \emptyset$ ).

The two learning paradigms fit different application scenarios and have distinct advantages and disadvantages depending on the context. For instance, the Global-

local paradigm is better suited to situations with high-order agent heterogeneity, whereas the global paradigm might encounter challenges when the personal models need to be highly distinguished from the global model. Furthermore, Global-local paradigm can be used in privacy-preserving learning such as Federated Learning. In Section 2.4, we will review the existing works in the two paradigms.

### 2.3.2 Collaborative MML (CoMML)

CoMML focuses on the multi-agent collaboration for multimodal intelligence. Following the setup in the previous section, CoMML specifies  $M$  **modality types** (e.g., image, video, text, audio, tabular) and  $O$  **types of downstream tasks** across the  $N$  agents. An agent  $i$  may solve any or multiple MML tasks, taking either single modality or multimodal modalities as inputs. Accordingly, each agent has the following properties:

- Every agent  $i$  is associated with a subset of modality types  $\mathcal{I}_i \subseteq \{1, 2, \dots, M\}$ , leading to a **combinatorial input space**  $\mathcal{X}_i := \mathcal{X}_{\mathcal{I}_i} = \text{Join}(\mathcal{X}^{(m)} | \forall m \in \mathcal{I}_i)$ , where  $\mathcal{X}^{(m)}$  is the subspace associated with the modality type  $m$ .
- The agent-specific input distribution  $\mathcal{P}_i(\tilde{\mathbf{x}})$  is defined over the space  $\mathcal{X}_{\mathcal{I}_i}$ , where  $\tilde{\mathbf{x}} = \text{Join}(\mathbf{x}^{(m)} | m \in \mathcal{I}_i)$ . In the local dataset  $\mathcal{D}_i$ , every sample’s input can be written as  $\tilde{\mathbf{x}}_{ij} \sim \mathcal{P}_i(\tilde{\mathbf{x}})$  where  $\tilde{\mathbf{x}}_{ij} = \text{Join}(\mathbf{x}_{ij}^{(m)} | m \in \mathcal{I}_i)$ .
- Likewise, Every agent  $i$  has a subset of downstream task types  $\mathcal{O}_i \subseteq \{1, 2, \dots, O\}$ , which leads to an **output space** consisting of multiple independent label spaces of each local tasks  $\mathcal{Y}_i := \mathcal{Y}_{\mathcal{O}_i} = \{\mathcal{Y}_i^{(o)} | \forall o \in \mathcal{O}_i\}$ , where  $\mathcal{Y}_i^{(o)}$  is the subspace for the task type  $o$  and has its agent-specific concept definitions.
- The agent-specific conditional output distribution  $\mathcal{Q}_i(\tilde{\mathbf{y}} | \tilde{\mathbf{x}})$  is defined over  $\mathcal{Y}_{\mathcal{O}_i}$ . In the local dataset  $\mathcal{D}_i$ , every sample’s target can be written as  $\tilde{\mathbf{y}}_{ij} \sim \mathcal{Q}_i(\tilde{\mathbf{y}} | \tilde{\mathbf{x}}_{ij})$ , where  $\tilde{\mathbf{y}}_{ij} = \{\mathbf{y}_{ij}^{(o)} | o \in \mathcal{O}_i\}$ .
- The agent-specific **neural architecture**  $\mathcal{A}_i$ , which determines the weight space topology  $\mathbb{R}^{d_i}$  for trainable parameters  $\boldsymbol{\theta}_i$ , typically consists of unimodal embedding and encoder, inter-modality interaction module, and decoders. We will separately define the **trainable weights** of each module. For example, we will use  $\boldsymbol{\theta}_{\text{enc},i}^{(m)}$  to denote the trainable weights for modality- $m$ ’s encoder module

of agent  $i$ , and the **decomposed forward function** for this module will be denoted as  $f_{\mathcal{A}_{\text{enc},i}^{(m)}}(\cdot; \boldsymbol{\theta}_{\text{enc},i}^{(m)})$ . If all agents use the same sub-architecture in this module, we will simplify the notations as  $f_{\mathcal{A}_{\text{enc},1}^{(m)}} = f_{\mathcal{A}_{\text{enc},2}^{(m)}} = \dots = f_{\mathcal{A}_{\text{enc},N}^{(m)}} = f_{\text{enc}}^{(m)}$ . Details will be introduced in each chapter.

- The **local objective** can be rewritten as the multitask objective  $\min_{\boldsymbol{\theta}_i} \mathcal{L}_i(\boldsymbol{\theta}_i) := \mathbb{E}_{(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}) \sim \mathcal{D}_i} \frac{1}{|\mathcal{O}_i|} \sum_{o \in \mathcal{O}_i} \mathcal{L}^{(o)}(\mathbf{y}^{(o)}, f_{\mathcal{A}_i}(\tilde{\boldsymbol{x}}; \boldsymbol{\theta}_i)_o)$ .

The aforementioned properties, along with Section 2.3.1, delineate the general setups of CoMML. Every subsequent chapter of this thesis will adhere to these setups or concentrate on specific cases thereof.

### 2.3.3 Personalization in Collaboration

This thesis will focus on *five* real-world **personalization patterns** between agents in Personalization-aware CoMML, including *modality* preferences (M), *concept* preferences (C), *domain* preferences (D), *task* preferences (T), and *architecture* preferences (A). Each of these personalization patterns will result in specific heterogeneity patterns among collaborative agents. Therefore, we provide terminology and definitions for these **heterogeneity patterns** here.

- **Modality Gap.** Due to modality preferences, agents vary in their *input spaces*. That is, for any pair of agents  $(i, i')$ , it is possible that  $\mathcal{X}_{\mathcal{I}_i} \neq \mathcal{X}_{\mathcal{I}_{i'}}$  due to their different input modality types  $\mathcal{I}_i \neq \mathcal{I}_{i'}$ . For example, in a multi-vehicle collaboration system, a vehicle may use its onboard camera to capture videos to predict traffics, while another vehicle may use both video and RADAR signals to predict traffics.
- **Task Difference.** Due to task preferences, agents might vary in their *output spaces*. That is, for any pair of agents  $(i, i')$ , it is possible that  $\mathcal{Y}_{\mathcal{O}_i} \neq \mathcal{Y}_{\mathcal{O}_{i'}}$  since they target at different downstream tasks  $\mathcal{O}_i \neq \mathcal{O}_{i'}$ . For example, in a multi-purpose surveillance system where all camera-based agents take videos as input, agents installed at conferences may focus on video activity classification tasks, while other agents intended for daily life observations may prioritize object detection from video.



- **Domain Drift.** Due to the different environments from which agents collect observations, agents may vary in their *input distributions*. That is,  $\mathcal{P}_i \neq \mathcal{P}_{i'}$ . Specifically, we consider the joint distribution drifts, meaning that the multimodal interaction behaviors can vary between agents. For example, if there are two agents that both have image and text modalities, the image-text relationships can happen in the background contexts at one agent but occur around the contour of objects at the other agent.
- **Concept Shift.** Due to concept preferences, agents have different interested concepts or different definitions for the same concept. That is, they vary in their conditional *output distributions*  $\mathcal{Q}_i \neq \mathcal{Q}_{i'}$ . The label space of classification can be represented at different concept granularities: **(1)** *instance-level* concept shifts and **(2)** *segment-level* concept shifts. For instance, in computer vision, image classification labels correspond to each sample, image segmentation labels correspond to each pixel, and video event recognition labels correspond to each frame. Likewise, text classification labels are associated with each text sequence, while named entity recognition segmentation labels are associated with each word, and so on.
- **Architecture Gap.** Due to architecture preferences, agents vary in their *weight spaces*. That is, for any pair of agents  $(i, i')$ , it is possible that  $\mathbb{R}^{d_i} \neq \mathbb{R}^{d_{i'}}$  since they use different neural architectures  $\mathcal{A}_i \neq \mathcal{A}_{i'}$ . Specifically, there could be four cases of architecture gap. **(1)** Different input modalities or target tasks lead to variations in the *input channels* (i.e., construction of unimodal encoders) and *output channels* (i.e., task-specific decoders). **(2)** *Topology Difference* is a common situation in multimodal neural architecture. Two agents might use different model families (e.g. one agent is based on Transformer while the other is based on ResNet) or use different multimodal fusion strategies for different input modality types (e.g. one agent uses alignment while the other uses concatenation). **(3)** *Depth Difference* refers that two agents having the same topology (e.g. both are based on ResNet) but their numbers of layers/modules are different. **(4)** *Width Difference* describes a situation where two agents having the same topology and same depth, but their numbers of neurons at each layer are different.

Domain drifts and concept shifts have also been referred to as statistical heterogeneity, as discussed in [31, 12], while the other three heterogeneity patterns (modality gap, task difference, architecture gap) have received less attention previously. We will review the previous works addressing these heterogeneities as follows.

## 2.4 Literature Review

**Review on Collaboration Paradigms:** (1) *Federated Learning (FL)*. FL is a distributed collaborative AI approach [32]. By pushing AI data training to the network edge at Internet-of-Things devices where the data reside, FL benefits both network operators and IoT user agents in terms of network resource savings and privacy enhancement. Early works such as FedAvg [33] builds the global model based on averaging the local Stochastic Gradient Descent (SGD) [34, 35] updates. Then, Various methods [24, 36, 37, 38] are introduced to improve the robustness of the global model under non-IID settings. For example, FedProx [24] adds a proximal term to the local objective. Personalized FL [25, 26, 27] has been proposed as an alternative to deal with non-IID data, where the global model plays the role of a meta-model to be used as initialization for few-shot adaptation at each client. For example, pFedMe [26] used Moreau envelopes, while PerFedAvg [27] took advances of meta learning approaches [39]. In the context of multi-modality, [40] learns the correlated alignment information from multiple modalities in the unsupervised manner, and [41] uses the co-attention mechanism in personalized FL to fuse the complementary information of different modalities. FL approaches can be further categorised into vertical FL, horizontal FL, Federated transfer learning, and so on. (2) *Mutual learning*. The idea of collaborative learning has been explored in Mutual Learning, also referred to as Online Knowledge Distillation (OKD). DML [42] shows that a group of models can benefit from mutual learning of predictive class probability distributions. CL [43] further extends this idea to a hierarchical architecture with multiple classifier heads. PCL [9] introduces an extra temporal mean network for each peer as the teacher role. In contrast to mutual mimicry, KDCL [44] construct an online teacher via a weighted ensemble logit distribution but differ in various aggregation strategies. Mutual Contrastive Learning [45, 46]

proposes mutual transfer of self-supervision augmented distributions and perform mutual interaction and transfer of contrastive distributions. Mutual Learning has recently been integrated with FL to scale up to a large number of participating agents [47, 48, 49, 50]. **(3) *Meta Learning*** can be utilized in the multi-agent collaboration scenarios due to its generalization ability over a distribution of multiple tasks. Recent meta-learning approaches can be divided into two categories: *inductive* and *transductive* few-shot classification. Inductive few-shot learning has been more widely studied than transductive few-shot learning. Inductive methods mainly includes metric-based and optimization-based algorithms. *Metric-based approaches* learn an embedding metric space shared by all tasks, on which data samples of different classes can distinguish with each other based on distance measurements [51, 52, 53, 54]. While metric-based methods directly measure data similarities of all tasks in a common metric space, *optimization-based approaches* train a meta-learner as an optimizer to fine-tune the meta-prior, thus adapt the class distribution to each specific task [39, 55, 56, 57]. Further, [58, 59] incorporated meta learning into multimodal fusion tasks by employing recurrent neural networks or modality modulators. The three families of frameworks have been actively integrated with each other, and each of them can be further categorised into either the global or the global-local paradigm defined in Section 2.3.1. We will review personalized approaches within each of the three frameworks as follows.

**Handling Statistical Heterogeneity in Collaboration:** In recent years, the fields of Federated Learning, Mutual Learning, and Meta Learning, have been actively investigating the challenges posed by domain shifts and concept shifts across tasks. **(1) *Personalized Federated Learning (PFL)***. Statistical heterogeneity, or Non-IIDness, has been widely discussed in Federated Learning (FL). While global-only FL has addressed this challenge [24, 36, 37, 38] by modification of local objective or gradient update process, PFL [25, 26, 27], an alternative family of FL methods that allow agents to train their own local models, have employed fine-tuning methods (e.g., FedPer [60], pFedMe [61], Per-FedAvg [27]), meta-learning methods (e.g., MAML [39]), mixture methods (e.g., APFL [62], PFL-MoE [29], Factorized-FL [63]), multi-task learning methods (e.g., MOCHA [28]), graph based

methods [64, 65], and hypernetwork based methods (e.g., HyperPFL [30, 66]).

**(2) *Task-adaptive Meta Learning.*** The Learning-to-learn framework of meta learning provides freedom of injecting agent-specific domain or concept knowledge into inner-loop adaptation. [52, 67, 68] have improved the metric- or optimization-based methods in terms of task adaptability by introducing task-adaptive metric space [52], modulation network [67], or explicitly modeling structured task relationships [68]. Among existing works, approaches addressing concept shifts have primarily focused on an instance-level granularity. However, finer-grained concept shifts, such as variations in label definitions for tasks like object detection and token labeling, remain relatively less explored.

**(3) *Domain-aware Mutual Learning.*** DaFKD [69] discerns the importance of each agent’s model to each distillation sample, enabling optimization of the ensemble of soft predictions from diverse domains. CCDistill [70] explores fine-grained cross-agent domain correlations by utilizing comprehensive knowledge extracted from features, such as content and style knowledge from images, and then conducts correlation-aware distillation.

**Handling Modality Gap and Task Difference in Collaboration:** We review two families of approaches that have been used to address modality gaps and task differences across collaborative agents.

**(1) *Single-path Alignment Approaches.*** At first glance, straightforwardly applying single-modal single-head collaborative learning approaches to our modality-heterogeneous multi-task settings [33, 24, 28, 71] is feasible. Specifically, by unifying all possible input and output spaces using agent-specific adaptors, one can fit single-modal single-task models for multimodal multi-tower tasks. For example, [40] employ cross-modal alignment on an agent-shared latent representation space, enabling the global model to serve both unimodal agents and agents employing various combinations of multiple modalities. This strategy often suffers from erroneous information sharing during collaboration especially when different modality types entail highly discrepant information or when task types are highly distinguished.

**(2) *Multi-path Transfer Approaches.*** Alternatively, recent works [72, 73] encourage agents’ models to partially share input and output channels, leveraging a unified global network where each agent-specific input-output mapping function uses only a subset of pathways

within this model. The computational pathways of each agent can be predefined or dynamically searched using techniques like SparseMoE [74, 75, 76] and Reinforcement Learning [77]. The different computational pathways of agent models share knowledge with each other through either weight average (e.g., FedMoE [78]) or feature distillation (e.g., [79, 80, 81]). For example, Cross-modal Mutual Learning employs distillation on an agent-shared modality-shared latent space [82, 83, 84, 85], or distill on a learned latent modality structures [86]. FedMoE [78] employs Mixture of Experts to dynamically assign gated weights to merge global models and private models. More works on Multimodal FL, Multimodal Meta Learning, and Mutual Co-Learning for multimodal tasks, have been surveyed in [87, 88, 89]. These lines of works have demonstrated limitations in ensuring efficient and accurate knowledge transfer when both modality gaps and task differences exist, and have also shown training inefficiency with highly discrepant agents.

**Handling Architecture Gap in Collaboration:** Given the recent real-world application requirements for collaboration across diverse computation platforms with varying resource budgets, the knowledge transfer across proprietary model architectures has attracted significant research attention in collaborative learning. **(1) *Feature-sharing Approaches via Distillation.*** In the field of Federated Learning (FL), Federated Mutual Learning utilizes Knowledge Distillation to share predictions or intermediate features of local models among agents, making it a natural choice for facilitating knowledge transfer across diverse architectures. For example, FedDistill [90], FML [49], PFML [91], and FedGKD [92], adopt collaborative knowledge distillation to align predictions among diverse client models on the server. Without requiring a public dataset at server, Data-free Federated Distillation learns an additional data generator to simulate local data on the server [93, 94]. **(2) *Weight-sharing Approaches via Supernetwork and Hypernetwork.*** Another line of works in Federated Learning (FL) has explored direct weight-sharing approaches to handle diversified network architectures. HeteroFL [95] allows participants to share parameters among prototypes. Another line of works, such as Split-Mix [72], DepthFL [96], and DisPFL [97], leverages a large supernetwork, where each agent is aligned to partial parameters in this supernetwork indicated

by a supermask learned by pruning techniques, and/or adopts progressive distillation to handle depth difference. While supernet-based methods have shown their unsuitability for topology differences across architectures, other approaches have explored hypernetworks to address these topology differences by generalizing architecture graphs [98, 66]. However, these approaches have limitations in multi-modal scenarios and cases involving joint architectural-statistical heterogeneity.

**Collaborative Learning for Large Foundation Models:** There has been a growing concern regarding data privacy during the pre-training and fine-tuning phases of Multimodal Foundation Models. For example, the massive multimodal corpora for pre-training might include *sensitive or personal information*, thus centralizing these data is not possible; also, commercial competition tend to isolate users’ feedbacks, hindering direct collaboration and knowledge sharing for downstream task fine tuning. **(1) Collaborative Pre-training.** Recent works have shown that text-only LLMs can be pre-trained with Federated Learning. For example, FedBERT [99] enables agents with limited computing capability to participate in pre-training a large language model. Subsequently, FedBFPT [100] employs multiple smaller local models to further pre-train a global model tailored for specific tasks. **(2) Collaborative Fine-tuning.** [101, 102, 103, 104, 105, 106] have investigated fine-tuning text-only LLMs using FL. Federated Instruction Tuning (FedIT) [107] adopts FL as the learning framework for instruction tuning of LLMs, effectively utilizing users’ diverse instructions stored on local devices while addressing concerns related to data sensitivity and transmission costs. OpenFedLLM [108] builds a codebase for collaborative and privacy-preserving LLM training on distributed private data via FL, allowing multiple data owners to train a shared model without transmitting raw data. GPT-FedRec [109] introduces a federated recommendation framework leveraging ChatGPT and a hybrid Retrieval Augmented Generation (RAG) mechanism. Despite these initial explorations, the field is still in its early stages. More significantly, there is a lack of discussions on decentralized Artificial General Intelligence (AGI) focusing on Multimodal LLMs, and no existing work has addressed complex inter-agent heterogeneity in this area.

# Part I

## Collaboration with Graph and Meta Learning

# Chapter 3

## Heterogeneous Hypernode Graph Learning for Collaborative Information Fusion with Modality Gap

### 3.1 Introduction

In this chapter, we address situations where merely **modality preferences** of individual MML agents are allowed, assuming all agents solve the same classification task with a shared label space from a shared domain. One real-world application scenario of such a setting is the task of *multimodal information fusion where some modalities may be missing* [110]. Each agent solves an information fusion task, which takes multiple modalities as input, fuses their complementary information, and then makes a decision. Yet it's common that not all modality types are available; modalities are often missing due to reasons such as agents' disuse, sensor damage, and data corruption. This chapter aims to study and experiment on this specific use case—*incomplete multimodal information fusion via multi-agent collaboration*, to delve into the research on “CoMML with modality preferences”.

Given the modality preferences of individual agents, different agents might have different combinations of input modalities, as illustrated in Figure 3.1.1. Such a heterogeneity pattern caused by modality preference is called **modality gap**. The modality gap among agents poses two technical challenges. (1) First, they bring



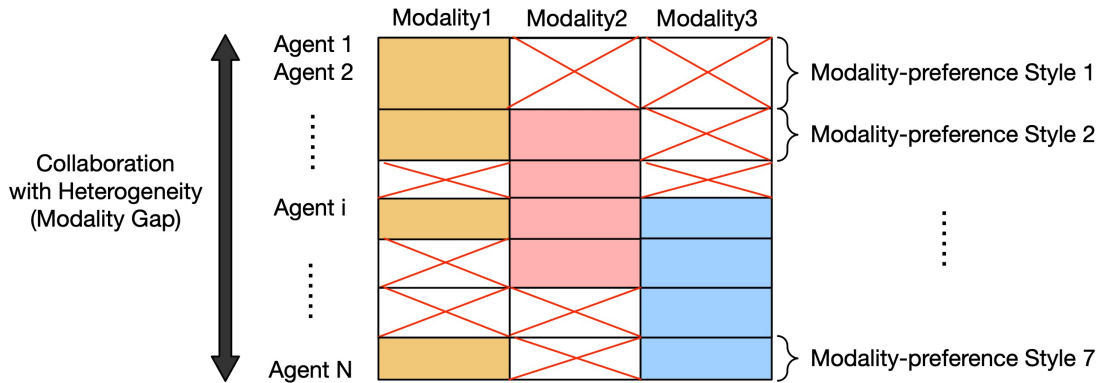


Figure 3.1.1: An illustration of modality preferences in a trimodal multi-agent collaboration scenario. There are at most seven input feature spaces.

about inconsistent input data structures, and thus introduce difficulties in applying complete multimodal fusion models [20, 111, 112, 113] that require the same input structure, consisting of all modalities, to be fed to their model architecture. (2) Second, modality gap might lead to inconsistent input information. Effective multimodal fusion requires learning about complementary information, the modality-specific information as well as the multimodal interactions [15]. However, agents that take fewer modalities as inputs (e.g., only a single modality) may lack relevant information compared to other agents with additional input modalities.

Prior works have addressed these challenges by imputing the absent modalities and then conducting collaboration with the **imputed homogeneous** information. Traditional imputation techniques include zero/average imputation and matrix completion. Deep learning based imputation approaches employ generative models to synthesis those modalities that cannot be seen at the agent based on the observable modalities [114, 115, 116]. Then, multimodal fusion can be achieved simultaneously [117, 118, 119] or after imputation [116, 114]. However, these methods may introduce extra noise to the original data which has negative impacts on the performances, and they are sometimes associated with complex auxiliary models such as deep generative models.

To avoid potential noises, we choose to pursue an alternative line of research, which skips the imputation and **directly collaborates with heterogeneous information** [120, 121, 122]. Multi-source feature learning method [120] partitions the agents into multiple complete subgroups, and then integrates representations of

the subgroups as a sparse multi-task learning problem. Multi-hypergraph learning method [123] incorporates high-order relationships of subgroups and learn directly on the output. However, these methods ignore the complex intra- and inter-modal interactions and fail to learn the fine-grained relationships among all agents.

Motivated by the limitations of existing approaches mentioned above, in this chapter, we introduce a new fundamental framework that facilitates both (1) the complex *intra-agent* information extraction as well as (2) the *inter-agent* information sharing between modality-heterogeneous agents without the need for imputation. The proposed framework, namely **H**eterogeneous **G**raph-based **M**ultimodal **F**usion (HGMF), models the multi-agent collaboration in a heterogeneous graph structure, and then exploits graph neural network-based transductive learning to extract complementary information from the highly interacted incomplete multimodalities and fuse the information from different subspaces into a unified space. The proposed approach also tackles a series of technical challenges posed by the graph construction, the exploitation of multimodal interactions, and the information sharing schemes between modality-heterogeneous agents through graph learning. Here is a summary of this chapter.

- We propose leveraging a Heterogeneous Hypernode Graph (HHG) to model the collaboration behavior between modality-heterogeneous agents. Such graph representation connects the agents that have various types of input modality combinations, and will help to explore the complex multimodal interactions and inter-agent relationships.
- We propose a novel transductive learning framework based on graph neural network, applied to the constructed HHG, which handles both intra-agent feature extraction and inter-agent information sharing. The key idea is to derive relevant information from the agents having observations of specific modalities, to enhance those without them.
- We conduct experiments in multiple levels of modality gap and demonstrate that our method can deal with real scenarios with a high degree of freedom in modality preferences. We show the effectiveness of our model by comparing it with both inductive and transductive baselines on three datasets.

## 3.2 Related Work

In addition to the literature reviewed in Section 2.4, our proposed techniques are related to and inspired by three lines of work: 1) multimodal fusion; 2) multimodal data analysis with incomplete modalities; 3) graph-based transductive learning.

**Multimodal Information Fusion.** Data from multiple sources can provide complementary information that reveals the fundamental characteristics of subjects and phenomena [15]. Integrating multimodal information thus can promote the concept understanding. Therefore, multimodal data fusion has become a widely-studied topic across various application scenarios, such as object detection [124, 125], sentiment analysis [126, 112, 111], emotion recognition [112, 123, 20], and disease diagnosis [123]. Extensive work [127, 4, 128] has been developed to combine different modalities and perform predictions. These approaches focus on learning representations to exploit the complementarity and redundancy of multiple modalities [129, 130, 131, 132, 133] and fusing information to perform the prediction tasks [134, 135, 136, 15, 137]. The majority of prior studies on deep multimodal fusion assume complete feature sets (i.e., all modalities are available for the inputs). Early fusion methods refer to concatenating multimodal data at the input level [138, 139], while late fusion methods [140, 113] integrate unimodal outputs. Graph-based methods such as hypergraph neural networks (HGNN) [141] perform early fusion (concatenation) as well as late fusion which exploits graph structural relationships among unimodal representations to integrate outputs. However, these methods have limited capabilities in exploring complementary information from high-order modality interactions, and cannot deal with missing uni-modalities. Recent methods that perform intermediate fusion include multimodal sequential learning [111, 112] for sequential data (time series, language, audio and video), and post-dynamic learning for general data [20, 21]. However, these works cannot model multimodal interactions with the presence of modality preferences.

**Incomplete Multimodal Data Analysis.** Imputation methods [119, 114, 116] that complete or generate incomplete modalities may introduce extra noise

to the fusion process. Non-imputation methods such as multi-source learning [120] and multi-hypergraph learning [123] first partition the incomplete data to multiple complete subgroups, and then integrate subgroup representations using multi-task learning or shallow graph learning through graph Laplacian. However, these works fail to effectively model the interactions between modalities with missingness, and fail to explore the relationships between different incompleteness patterns.

**Graph-based Transductive Learning.** Several graph-based transductive learning models designed generally are also related to our work. Graph Attention Networks (GATs) [142] compute attention coefficients using an edge-wise mechanism, which is extended in our work to learn a heterogeneous hypernode graph where there is not an immediately-obvious structure. We employ the graph attention mechanism to learn the unknown relationships between different incompleteness patterns within a heterogeneous hypernode graph. In addition, several GNN variants [143, 144] propose to handle node embedding in heterogeneous graphs. HGNN [141] and multi-hypergraph learning [123] perform late fusion on graphs constructed from complete or incomplete multimodal data, through traditional graph Laplacian or graph neural networks. These methods cannot deal with the complex multimodal hypernode structure formulated in this chapter.

### 3.3 Problem Formulation

**Definition 3.1 (Collaborative Multimodal Fusion with Modality Gap).** Suppose there are  $M$  modality types across  $N$  agents. For simplicity, in this chapter, we assume that each agent  $i = 1, 2, \dots, N$  has only one sample, i.e.,  $n_i = 1$ . We define  $\widehat{\mathcal{D}}_{\text{colab}}$  consisting of a collection of  $N_{\text{trn}}$  training agents and  $N_{\text{tst}}$  testing agents, where each agent has one labeled or unlabeled sample, along with an indicator specifying its preferred modality set

$$\begin{aligned}
 \widehat{\mathcal{D}}_{\text{colab}} &:= \{\mathcal{D}_1, \mathcal{D}_2, \dots, \mathcal{D}_{N_{\text{trn}}}, \mathcal{D}_1^*, \mathcal{D}_2^*, \dots, \mathcal{D}_{N_{\text{tst}}}^*\}, \\
 \mathcal{D}_i &:= (\tilde{\mathbf{x}}_i, \mathcal{I}_i, y_i), \quad i = 1, 2, \dots, N_{\text{trn}} \\
 \mathcal{D}_i^* &:= (\tilde{\mathbf{x}}_i^*, \mathcal{I}_i^*), \quad i = 1, 2, \dots, N_{\text{tst}}
 \end{aligned} \tag{3.1}$$

where  $\tilde{\mathbf{x}}_i = \{\mathbf{x}_i^{(m)}\}_{m \in \mathcal{I}_i}$ . Recall that  $\mathcal{I}_i \subseteq [M]$  is the modality-preference style—a set of preferable or available modalities—on agent  $i$ . The goal is to train a global model  $\boldsymbol{\theta}^*$  from the training agents, which can capture modality interactions from all agents with different styles.  $\boldsymbol{\theta}^*$  should perform well on testing agents, even if they have different modality-preference styles compared to the training agents.

## 3.4 Methodology

### 3.4.1 Overview

Given the above setup and objective, we will formulate a **Transductive Learning** framework [123] to handle multi-agent collaboration with modality gap. Different from inductive learning, transductive learning directly incorporates the feature information implicit in other samples [145]. Our key idea is that an agent lacking certain input modality can derive relevant information from other agents possessing it within the transductive learning framework. Agents with different modality-preference styles can effectively exchange their modality-specific and inter-modal interaction information, and multimodal fusion can be achieved along the way.

Among transductive learning variants, graph-based transductive learning methods achieved promising performance in practice [145, 141]. Recent advancements in graph neural networks (GNNs) also allow high-level features and high-dimensional representations to be learned from graph structural original data. Since graphs are powerful representations to model data relationships, in this chapter, we use graphs to exchange significant information between multimodal agents, and formulate our problem in a novel GNN-based transductive learning framework, HGMF.

In this section, we present our HGMF method which is built based on a GNN transductive learning framework. The HGMF has three stages: 1) modeling incomplete multimodal data in a proposed heterogeneous hypernode graph structure; 2) encoding the highly interacted multimodal data with the presence of missingness into more explicit modality-specific and cross-modal interaction information; and 3) aggregating and exchanging information among multimodal agents across different modality-preference styles, through which all data can be fused into the same

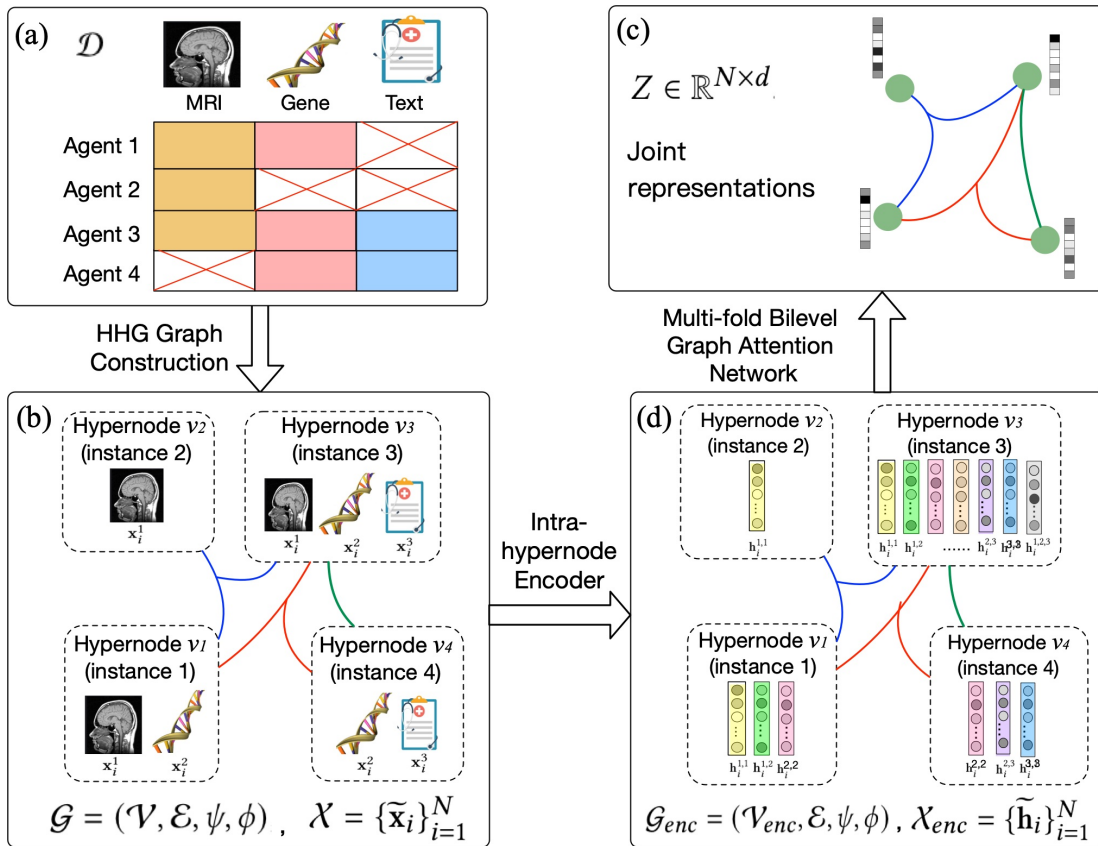


Figure 3.4.1: The three-stage workflow of Heterogeneous Graph-based Multimodal Fusion (HGMF). It shows a four-agent example over three modalities in total, but in real tasks, the graph can be larger in scale and more complex. First,  $\mathcal{D}$  include four agents with four styles of modality preferences.  $\mathcal{G}$  and  $\mathcal{X}$  are constructed after HHG graph construction stage.  $\mathcal{G}_{enc}$  and  $\mathcal{X}_{enc}$  are obtained through intra-hypernode encoding stage. Multimodal agents are finally fused into joint representations  $Z$  through multiple MBGAT layers' graph embedding stage.

embedding space. Figure 3.4.1 illustrates the three-stage HGMF workflow using a simple four-data example. Note that in real scenarios, graphs can be larger and more complex than those shown in Figure 3.4.1. In the following, we will introduce the technical details in each stage.

### 3.4.2 Modeling Collaboration in Heterogeneous Graph

The multi-agent collaboration with multiple modality-preference styles can be modeled as a  $k$ -NN affinity graph structure, where each node is an agent. However, as

each agent contains multiple data sources, belongs to different modality-preference styles, and has different feature spaces, we cannot use a simple affinity graph to model our problem. To this end, we first define a new family of graph structures, namely **H**eterogeneous **H**ypernode **G**raphs (HHG) as follows.

**Definition 3.2 (Heterogeneous Hypernode Graph).** A Heterogeneous Hypernode Graph (HHG) is denoted as  $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \psi, \phi)$ , containing the following components and properties.

- $\mathcal{V} = \{v_i\}_{i=1}^N$  is the *hypernode* set, where each hypernode is self-interacted. Different from simple graphs in which each node is associated with the same dimensional feature, hypernodes contains different numbers and dimensions of features, and each hypernode’s features may be implicitly or explicitly interacted. A hypernode refers to a multimodal agent, and we define the feature set of graph as  $\mathcal{X} = \{\tilde{\mathbf{x}}_i | i = 1, 2, \dots, N_{\text{trn}}\} \cup \{\tilde{\mathbf{x}}_i^* | i = 1, 2, \dots, N_{\text{tst}}\}$ .
- $\mathcal{E} = \{e_j\}_{j=1}^{|\mathcal{E}|}$  is the edge set. As we construct a  $k$ -NN affinity graph for agents, to more efficiently represent the high-order connections among nodes, we use the *hyperedges*<sup>1</sup> of hypergraphs instead of pairwise edges [123, 141]. A hyperedge is a subset of (hyper)nodes, connecting  $k$  agents who share some similar information, and showing a  $k$ -NN relationship among some nodes. Hyperedges  $\mathcal{E}$  can be represented by an incidence matrix  $\mathbf{H} \in \{0, 1\}^{|\mathcal{V}| \times |\mathcal{E}|}$ , where each row represents a hypernode  $v_i$  and each column represents a hyperedge  $e_j$ . For each entry in the incidence matrix,  $\mathbf{H}(v_i, e_j) = 1$  indicates that the hypernode  $v_i$  is connected with some others through the hyperedge  $e_j$ . Each hyperedge  $e_j$  is associated with a single-valued weight  $w_j$ . In this chapter, all edge weights equal to one.
- $\psi : \mathcal{V} \mapsto \mathcal{U}$  is the node type-mapping function, where  $\mathcal{U} = \{1, 2, \dots, 2^M - 1\}$  is all possible modality combinations in a multi-agent collaboration system with individual preferences over  $M$  modality types. Given  $M$  modality types, there could have up to  $(2^M - 1)$  modality-preference styles. Figure 3.1.1 illustrates a trimodal ( $M = 3$ ) case with seven modality-preference styles

---

<sup>1</sup>hyperedges: edges in hypergraphs are called hyperedges or hyperlinks. A hyperedge connects two or more than two vertices.

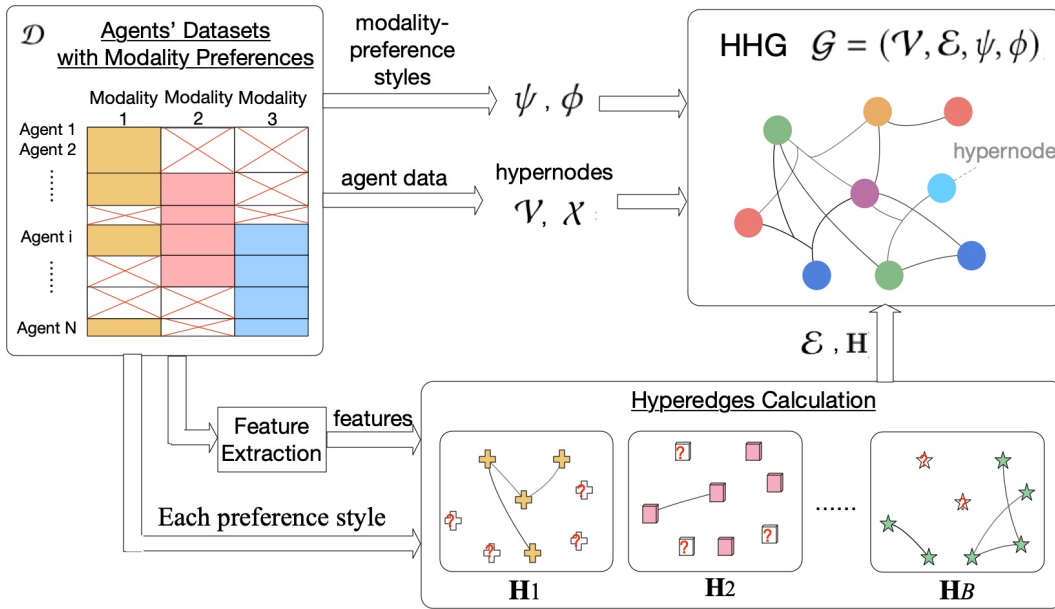


Figure 3.4.2: Pipeline of HHG graph construction from the multi-agent dataset with individual modality preferences.

[120]. Those with solid colors indicate those preferable or available modalities on each agent, and the other blocks represent unpreferable or non-available modalities. This figure also shows that agents can be divided into separate groups such that in each group all agents have the same modality-preference style and each agent only belongs to one style. The graph is *heterogeneous* because hypernodes (agents) have different modality-preference styles, so we define  $\psi$  to distinguish multimodal agents.

- $\phi : \mathcal{U} \mapsto \mathcal{P}(\mathcal{M}) \setminus \emptyset$  defines a function that maps a style to a combination of modalities, where  $\mathcal{P}(\mathcal{M})$  denotes the power set (all subsets) of the set  $\mathcal{M} = \{1, 2, \dots, M\}$ .

**Heterogeneous Hypernode Graph Construction.** Figure 3.4.2 shows an overview of the HHG graph construction process. On the left of the figure shows an trimodal incomplete dataset, where columns denote modalities and rows denote agents. Given such a multimodal dataset  $\mathcal{D}$ , one can easily obtain  $\psi(\cdot)$ ,  $\phi(\cdot)$ ,  $\mathcal{V}$ , and  $\mathcal{X}$  based on data availability and corresponding features. On the right in Figure 3.4.2 is the seven-style HHG constructed from the input, where different



colors denote different styles. Note that each node here is a hypernode containing multi-modalities. We also provide an illustration of heterogeneous hypernodes in Figure 3.4.1(b), where each agent constructs a hypernode who contains at least one modality. The bottom region in Figure 3.4.2 illustrates the hyperedge calculation process.  $\mathcal{E}$  can be calculated from  $\mathcal{D}$  as follows. As multimodal agents (hypernodes) have complex connections, each modality can provide a certain view of the data relationships. Motivated by [141, 123, 120], to capture the multi-view and high-order relationships among multimodal agents, we first reconstruct the blockwise incomplete dataset into  $B$  blocks according to different combinations of available modalities, and then calculate a set of hyperedges among all agents involved in each block. Let  $\mathcal{V}_b$  and  $\mathcal{M}_b$  denote the hypernodes and the modality set involved in the block  $b$ , respectively. We calculate the normalized distance between each pair of agents in a block ( $\forall v_i, v_{i'} \in \mathcal{V}_b$ ) as follows:

$$d^{(b)}(v_i, v_{i'}) \triangleq \frac{1}{|\mathcal{M}_b|} \sqrt{\sum_{m \in \mathcal{M}_b} \|f_{\text{emb}}^{(m)}(\mathbf{x}_i^{(m)}) - f_{\text{emb}}^{(m)}(\mathbf{x}_{i'}^{(m)})\|_2^2 / Z_m}, \quad (3.2)$$

where  $Z_m = \sum_{i, i' \in \mathcal{V}_b} \|f_{\text{emb}}^{(m)}(\mathbf{x}_i^{(m)}) - f_{\text{emb}}^{(m)}(\mathbf{x}_{i', m})\|_2^2$  and  $\{f_{\text{emb}}^{(m)}(\cdot), m = 1 \dots M\}$  are the pre-trained unimodal embedding models that are used to initialise shallow unimodal features. After calculating all distances, each hyperedge is calculated using  $k$  nearest neighbor method centered with each node [123]. As shown at the bottom region in Figure 3.4.2, for all pre-defined blocks,  $B$  sets of hyperedges are calculated independently depending on the feature extraction models' parameters. Suppose their incidence matrix are  $\{\mathbf{H}_1, \mathbf{H}_2, \dots, \mathbf{H}_B\}$ , the final incidence matrix for HHG is the concatenation form  $\mathbf{H} = [\mathbf{H}_1; \mathbf{H}_2; \dots; \mathbf{H}_B]$ . In this way, agents that do not have certain modalities can be connected with those that have the modalities, and then the incomplete data problem can be alleviated. In Figure 3.4.1(b), the hyperedges in different colors connects agents according to different blocks.

### 3.4.3 Intra-hypernode Encoder

After constructing the input graph  $\mathcal{G}$  and feature set  $\mathcal{X}$ , we propose an intra-hypernode encoder to capture complementary information [15] from the highly

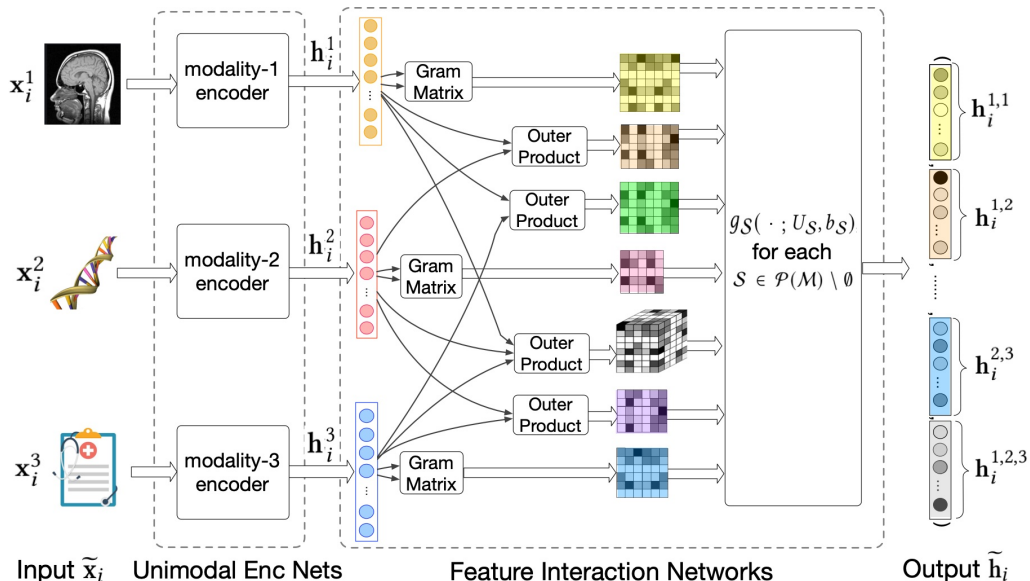


Figure 3.4.3: Architecture of intra-hypernode encoder, which takes as input each hypernode feature set  $\tilde{x}_i$  and output new feature set  $\tilde{h}_i$ . Partial neurons are blocked based on which modalities are unpreferable/non-available at each agent.

interacted modalities with the presence of missing data. The intra-hypernode encoder, whose architecture is shown in Figure 3.4.3, consists of two components: 1) *Unimodal Encoder Networks* take unimodal data as input, and output unimodal embeddings, and 2) *Feature Interaction Networks* captures the modality interactions among these embeddings, and extract complementary information (modality-specific and cross-modal interaction information) from them.

### 3.4.3.1 Unimodal Encoder Networks

Since the original data in  $\mathcal{X}$  is very high-dimensional, sparse, and inconsistent with respect to data structure, it is hard to calculate interactions among original modalities. We thus setup a series of *source-specific* Deep Neural Networks (DNNs) to learn compressed and rich feature representations from unimodal original data.

Based on real data structures in the datasets on which we perform our models, we mainly consider three types of architectures to build unimodal encoder networks: 1) Convolutional Neural Networks (CNN) for embedding image modalities; 2) Bidirectional Long-short Term Memory (BI-LSTM) for embedding sequential

modalities, such as video, free texts (e.g., clinical records) and spoken language; and 3) Stacked fully connected layers followed by nonlinear activation functions, for embedding high-dimensional or sparse feature-based modalities, such as gene expressions. Suppose  $f_{\text{enc}}^{(m)}(\cdot; \boldsymbol{\theta}_{\text{enc}}^{(m)})$  be the  $m$ 's unimodal encoder network with learnable parameter  $\boldsymbol{\theta}_{\text{enc}}^{(m)}$ . For each hypernode  $v_i$  whose content is  $\tilde{\mathbf{x}}_i = \{\mathbf{x}_i^{(m)}\}_{m \in \phi(\psi(i))}$ , its modality- $m$  is embedded as

$$\mathbf{h}_i^m = f_{\text{enc}}^{(m)}(\mathbf{x}_i^{(m)}; \boldsymbol{\theta}_{\text{enc}}^{(m)}), \quad (3.3)$$

where  $\mathbf{h}_i^m \in \mathbb{R}^{F_m}$ .  $F_m$  is the embedding dimension of modality- $m$ .

### 3.4.3.2 Feature Interaction Networks

A hypernode contains unimodal components that are highly interacted. Such modality interactions are high-order and implicit, and therefore difficult to be represented. The goal of feature interaction networks is to capture such interactions, to extract modality-specific and cross-modal interaction information from them.

The high-order modality interactions can appear individually, between each pair of modality and among more than two modalities. Let  $\mathcal{P}(\cdot)$  denote the power set operation and  $\mathcal{M} = \{1, 2, \dots, M\}$ . As each subset  $\forall \mathcal{S} \in \mathcal{P}(\mathcal{M}) \setminus \emptyset$  denotes a combination of modalities, we can learn from each  $\mathcal{S}$  one type of multimodal interaction and a piece of information, namely factor.

A hypernode's complementary information consists of many factors. Let each factor be represented by a  $F'$ -dimensional vector. A factor can be calculated as follows. If there is only one element in  $\mathcal{S}$  (i.e.,  $|\mathcal{S}| = 1$ ), meaning that we are calculating *modality-specific information* for the modality  $m \in \mathcal{S}$ , we can calculate modality-specific information  $\mathbf{h}_i^{m,m}$  as

$$\begin{aligned} \bar{\mathbf{h}}_i^m &= g_m(\mathbf{h}_i^m; \mathbf{U}_m, \mathbf{b}_m) \\ \mathbf{G}_i^m &= (\mathbf{h}_i^m)(\mathbf{h}_i^m)^T \\ \mathbf{h}_i^{m,m} &= g_{m,m}(\mathbf{G}_i^m; \mathbf{U}_{m,m}, \mathbf{b}_{m,m}) + \bar{\mathbf{h}}_i^m, \end{aligned} \quad (3.4)$$

where  $\mathbf{U}_m \in \mathbb{R}^{F' \times F_m}$ ,  $\mathbf{b}_m \in \mathbb{R}^{F'}$ ,  $\mathbf{U}_{m,m} \in \mathbb{R}^{F' \times (F_m)^2}$  and  $\mathbf{b}_{m,m} \in \mathbb{R}^{F'}$  are parameters

of the neural networks  $g_m(\cdot)$  and  $g_{m,m}(\cdot)$ , respectively.  $\mathbf{G}_i^m \in \mathbb{R}^{F_m \times F_m}$  is the Gram matrix of unimodal encoder  $\mathbf{h}_i^m$ , which represents the covariance, the feature self-interaction information; and  $\bar{\mathbf{h}}_i^m$  can be viewed as the mean, low-dimensional, and specific information for modality- $m$ .

If there are more than one elements in  $\mathcal{S}$  (i.e.,  $|\mathcal{S}| > 1$ ), meaning that we are calculating *cross-modal interaction information* among all unimodal encoders  $\{\mathbf{h}_i^m | \forall m \in \mathcal{S}\}$ . Inspired by [20], we can calculate cross-modal interaction information  $\mathbf{h}_i^{\mathcal{S}}$  as

$$\begin{aligned} \mathbf{C}_i^{\mathcal{S}} &= \otimes_{m \in \mathcal{S}} \mathbf{h}_i^m \\ \mathbf{h}_i^{\mathcal{S}} &= g_{\mathcal{S}}(\mathbf{C}_i^{\mathcal{S}}; \mathbf{U}_{\mathcal{S}}, \mathbf{b}_{\mathcal{S}}), \end{aligned} \quad (3.5)$$

where  $\mathbf{C}_i^{\mathcal{S}}$  represents the  $|\mathcal{S}|$ -fold cross-product of the involved unimodal encoders, and  $\mathbf{U}_{\mathcal{S}} \in \mathbb{R}^{F' \times (\prod_{m \in \mathcal{S}} F_m)}$  and  $\mathbf{b}_{\mathcal{S}} \in \mathbb{R}^{F'}$  is the learnable weights of the neural network  $g_{\mathcal{S}}(\cdot)$ . A special case is that, for example, the bimodal interaction information between  $m$  and  $m'$  can be extracted as  $\mathbf{h}_i^{m,m'} = g_{m,m'}((\mathbf{h}_i^m)(\mathbf{h}_i^{m'})^T; \mathbf{U}_{m,m'}, \mathbf{b}_{m,m'})$ .

### 3.4.3.3 Summary

In this section, our intra-hypernode encoder leverages all combinations of unimodal-specific and cross-modal interactions, and extracts pieces of complementary information from multi-modalities with the presence of missing data. We let  $\boldsymbol{\theta}_e$  consist of a collection of all parameter weights in Eq.(3.4) and Eq.(3.5).

The architecture shown in Figure 3.4.3 is shared by all hypernodes. Intra-hypernode encoder takes as input each hypernode feature set  $\tilde{\mathbf{x}}_i = \{\mathbf{x}_i^{(m)}\}_{m \in \phi(\psi(i))}$  and output new feature set  $\tilde{\mathbf{h}}_i = \{\mathbf{h}_i^{\mathcal{S}}\}_{\mathcal{S} \in \mathcal{P}(\phi(\psi(i))) \setminus \emptyset}$ . Then, we obtain a new heterogeneous hypernode graph  $\mathcal{G}_{enc} = (\mathcal{V}_{enc}, \mathcal{E}, \psi, \phi)$  associated with a new feature set  $\mathcal{X}_{enc} = \{\tilde{\mathbf{h}}_i\}_{i=1}^N$ , which is illustrated in Figure 3.4.1(c).

## 3.4.4 Multi-fold bilevel Graph Attentional Layer

In the previous step, the intra-hypernode encoder outputs new HHG  $\mathcal{G}_{enc}$  and feature set  $\mathcal{X}_{enc}$ , which contains more explicit modality-specific and cross-modal interaction information than the input feature set. The hypernodes in  $\mathcal{G}_{enc}$  are also heterogeneous, because hypernodes with different modality-preference styles

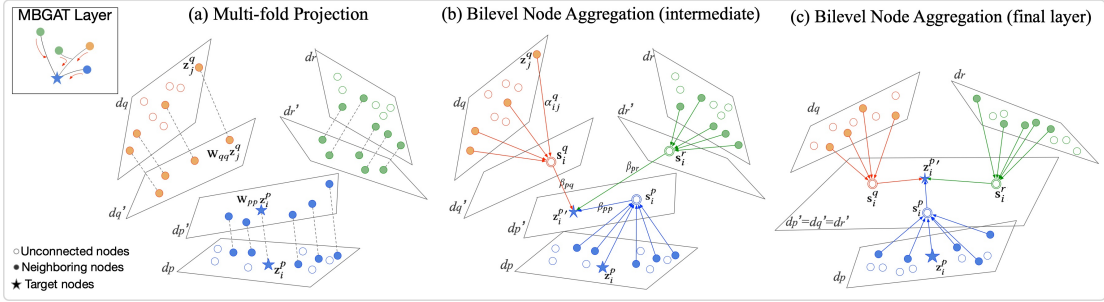


Figure 3.4.4: Multi-fold bilevel Graph Attentional Layers (a) Multi-fold embedding space projection. (b) Bilevel node aggregation at any intermediate layer. (c) Bilevel node aggregation at the final layer.

contain different numbers and categories of factors; there are a total of  $(2^{|\phi(\psi(i))|} - 1)$  factors calculated for hypernode  $v_i$ .

In this section, we focus on learning the interactions between different incompleteness styles, and propose to simultaneously fuse multimodal data with different missingness within a graph-based transductive learning architecture. Specifically, we propose to solve a sub-problem described as follows.

**Definition 3.3.** ( $(2^M - 1)$ -fold Heterogeneous Graph Embedding). Given the heterogeneous hypernode graph  $\mathcal{G}_{enc} = (\mathcal{V}_{enc}, \mathcal{E}, \psi, \phi)$ , where the node set can be divided into  $|\mathcal{U}| = 2^M - 1$  non-overlapping subsets  $\mathcal{V}_{enc} = \{\mathcal{V}_p | \forall p \in \mathcal{U}\}$  based on  $\psi(\cdot)$ , and each node  $v_i \in \mathcal{V}_p$  is associated with a set of  $F'$ -dimensional vectors  $\tilde{\mathbf{h}}_i = \{\mathbf{h}_i^{\mathcal{S}} | \forall \mathcal{S} \in \mathcal{P}(\psi(p)) \setminus \emptyset\}$ , the task is to learn to map the heterogeneous hypernodes in  $2^M - 1$  embedding spaces, into a homogeneous embedding space  $\mathbf{Z} \in \mathbb{R}^{N \times d}$ .

By solving this sub-problem, hypernodes with different modality-preference styles can be projected onto the same feature space; multimodal agents with missing information (missing factors) can derive such information from others; and, the final node embeddings can be the fused multimodal representations.

To solve this sub-problem, in this section, we propose **Multi-fold Bilevel Graph Attention Networks (MBGAT)** inspired by graph attention networks (GATs) [142]. In the following, we state the overall goal of each MBGAT layer, and then introduce technical details.

### 3.4.4.1 Main Idea

At each MBGAT layer, the goal is to project the existing features in  $2^M - 1$  spaces onto  $2^M - 1$  new spaces (see Figure 3.4.4) that are close to each other. However, aggregating information from heterogeneous nodes is challenging as the relationships between different feature spaces are unknown. In the context of multimodal fusion, such difficulty comes from the unknown relationships between different modality-preference styles.

To tackle this problem, inspired by self-attention mechanism [18] and GATs [142], we design a bilevel attention strategy to aggregate neighborhood information among different styles. An MBGAT layer consists of two components: *multi-fold intra-style aggregation* that aggregates nodes in the same space independently, and *inter-style aggregation* that learns style relationships, and fuses all neighbors into one target space.

At each MBGAT layer, we represent the multi-space inputs as  $\mathbf{z} = \{\{\mathbf{z}_i^p | \forall v_i \in \mathcal{V}_p\} | \forall p \in \mathcal{U}\}$ , where  $\mathbf{z}_i^p \in \mathbb{R}^{d_p}$  is the  $d_p$ -dimensional feature associated with the style- $p$  node  $v_i$ . The layer’s multi-space outputs are represented as  $\mathbf{z}' = \{\{\mathbf{z}_i^{p'} | \forall v_i \in \mathcal{V}_p\} | \forall p \in \mathcal{U}\}$ ,  $\mathbf{z}_i^{p'} \in \mathbb{R}^{d'_p}$ , where  $d'_p$  is the dimension of the new feature space of style- $p$ . Note that at the first layer, we initialize the input node features  $\mathbf{z}^{(0)}$  by concatenating all feature vectors in hypernodes, since the previously extracted features within a hypernode are separate pieces of information. In the following, we present the technical details on how to aggregate neighborhood information for a target node  $v_i$  whose style is  $p = \psi(i)$ .

### 3.4.4.2 Multi-fold Intra-style Aggregation

As the lower-level aggregation, we focus on aggregating neighbors in the same feature space (multimodal agents that miss the same modalities).

**Multi-fold Projection.** Each node should be projected onto its new and lower-dimensional feature space to get prepared for aggregation. As we prepare all nodes at the same time, each node in any style’s feature space will need to be combined with nodes in any other style’s space. We therefore define  $\{\mathbf{W}_{pq} | \forall p, q \in \mathcal{U}\}$  as our  $|\mathcal{U}|$ -fold projection scheme, where  $\mathbf{W}_{pq} \in \mathbb{R}^{d'_q \times d_p}$  is learnable matrix that

projects nodes from the style- $p$ 's feature space to the style- $q$ 's *new* feature space. Figure 3.4.4(a) illustrates the multi-fold projection, where nodes in different colors (different styles) are projected onto different spaces.

**Intra-patten Aggregation.** Suppose  $\mathcal{N}_q(i)$  denote the style- $q$  neighboring node set of  $v_i$ , which can be defined as  $\mathcal{N}_q(i) = \{v_{i'} | \forall v_{i'} \in \mathcal{V}_q \wedge (\mathbf{H}\mathbf{H}^T)_{i,i'} > 0\}$ , where  $\mathbf{H}$  is the incidence matrix (constructed in Section 3.1). We then calculate the importance of each neighboring node in  $\mathcal{N}_q(i)$  to the target node  $v_i$  by performing the attention mechanism,  $\vec{\mathbf{a}}_q \in \mathbb{R}^{2d'_q \times 1}$ . The calculated attention coefficients for each style- $q$  neighbor  $v_{i'}$  is

$$\alpha_{i,i'}^q = \frac{\exp(\text{LeakyReLU}(\vec{\mathbf{a}}_q[\mathbf{W}_{pq}\mathbf{z}_i^p; \mathbf{W}_{qq}\mathbf{z}_{i'}^q]))}{\sum_{k \in \mathcal{N}_q(i)} \exp(\text{LeakyReLU}(\vec{\mathbf{a}}_q[\mathbf{W}_{pq}\mathbf{z}_i^p; \mathbf{W}_{qq}\mathbf{z}_k^q]))}. \quad (3.6)$$

Finally, the intra-style aggregation result from all style- $q$  neighbors of node  $v_i$  is

$$\mathbf{s}_i^q = \sigma \left( \sum_{j \in \mathcal{N}_q(i)} \alpha_{i,i'}^q \mathbf{W}_{qq}\mathbf{z}_{i'}^q \right) \quad (3.7)$$

where  $\sigma(\cdot)$  denotes the sigmoid function. In Figure 3.4.4(a) and (b), we can see that nodes in the same colors (same style) are aggregated to a certain double-circled feature points on the new space.

### 3.4.4.3 Inter-style Aggregation

After aggregating the neighborhood information within each style, we aim to learn the relationships between different styles, so that multimodal agents that have different modality preferences can derive information from each other. To achieve the goal, we perform inter-style aggregation as the higher-level aggregation.

Similarly, given the intra-style aggregation results  $\{\mathbf{s}_i^1, \mathbf{s}_i^2, \dots, \mathbf{s}_i^{2^M-1}\}$ ,  $\mathbf{s}_i^q \in \mathbb{R}^{d'_q}$ , we can calculate the importance of the style- $q$  neighbors to the style- $p$  target by performing the attention mechanism,  $\vec{\mathbf{b}}_p \in \mathbb{R}^{2d'_p \times 1}$ . The calculated attention coefficients is:

$$\beta_{pq} = \frac{\exp(\text{LeakyReLU}(\vec{\mathbf{b}}_p[\mathbf{V}_{pp}\mathbf{s}_i^p; \mathbf{V}_{qp}\mathbf{s}_j^q]))}{\sum_{r \in \mathcal{U}} \exp(\text{LeakyReLU}(\vec{\mathbf{b}}_p[\mathbf{V}_{pp}\mathbf{s}_i^p; \mathbf{V}_{rp}\mathbf{s}_i^r]))}, \quad (3.8)$$

---

**Algorithm 1: HGMF**


---

- 1: **Input data:**  $\mathcal{M}, \mathcal{U}, \mathcal{V}, \phi, \psi, \mathcal{D} = \{\tilde{\mathbf{x}}_i\}_{i=1}^N, \mathcal{Y}_{trn}$
  - 2: **Input parameters:**  $k, \eta$
  - 3: Initialise networks with random parameters  $\boldsymbol{\theta}_e, \boldsymbol{\theta}_g, \boldsymbol{\theta}_p$ .  
 $\mathbf{H} \leftarrow kNN(\mathcal{D}, k)$  using Eq.(3.2)  
 $\mathcal{X} = \{\{\mathbf{x}_i^{(m)}\}_{m \in \phi(\psi(i)) \subseteq \mathcal{M}}\}_{i=1}^N \leftarrow \mathcal{D}$   
 $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \psi, \phi) \leftarrow \mathbf{H}, \mathcal{X}$
  - 4:  $t \leftarrow 0$
  - 5: **while** stopping condition is not met **do**
  - 6:   **for**  $v_i \in \mathcal{V}$  **do**
  - 7:      $\tilde{\mathbf{h}}_i \leftarrow \text{InHyNEnc}_{\boldsymbol{\theta}_e}(\tilde{\mathbf{x}}_i)$  using Eq.(3.2)–Eq.(3.4)
  - 8:   **end for**
  - 9:    $\mathcal{G}_{enc}, \mathcal{X}_{enc} \leftarrow \{\tilde{\mathbf{h}}_i\}_{v_i \in \mathcal{V}}$
  - 10:    $\mathbf{Z} \leftarrow \text{MBGAT}_{\boldsymbol{\theta}_g}(\mathcal{G}_{enc}, \mathcal{X}_{enc})$  using Eq.(3.5)–Eq.(3.8)
  - 11:    $\hat{\mathbf{Y}} \leftarrow \text{Prediction}_{\boldsymbol{\theta}_p}(\mathbf{Z})$
  - 12:    $\mathcal{L} \leftarrow \hat{\mathbf{Y}}_{trn}, \mathcal{Y}_{trn}$  using Eq.(3.9)
  - 13:   Update  $\boldsymbol{\theta}_e, \boldsymbol{\theta}_g, \boldsymbol{\theta}_p \leftarrow \min_{\boldsymbol{\theta}_e, \boldsymbol{\theta}_g, \boldsymbol{\theta}_p} \mathcal{L}$
  - 14:    $t \leftarrow t + 1$
  - 15: **end while**
  - 16: **return:**  $\boldsymbol{\theta}_e^*, \boldsymbol{\theta}_g^*, \boldsymbol{\theta}_p^*$ , and  $\hat{\mathbf{Y}}_{test}$
- 

where  $\mathbf{V}_{qp} \in \mathbb{R}^{d'_p \times d'_q}$  denotes the space-to-space transformation from style- $q$  to style- $p$ . Once obtained, the attention coefficients are used to compute a linear combination of intra-style aggregation results. Finally, we can update the embedding for target node  $v_i$  as:

$$\mathbf{z}_i^{p'} = \sigma \left( \sum_{q \in \mathcal{U}} \beta_{pq} \mathbf{V}_{qp} \mathbf{s}_i^q \right). \quad (3.9)$$

As shown in Figure 3.4.4(b) and (c), double-circled points (intra-aggregation results) are aggregated to the blue star node on  $d'_p$  space (the node  $v_i$ 's new embedding point on style- $p$ 's new space).

#### 3.4.4.4 Summary

In this section, we proposed MBGAT, which performs multimodal fusion through an  $2^M - 1$ -fold heterogeneous graph embedding procedure. At each MBGAT layer, the two levels of aggregation enable each node to receive information from its  $2^M - 1$



styles of neighboring nodes, in which learned attention coefficients are responsible to handle how different modality interaction information exchanges between agents.

We stack multiple MBGAT layers, so that the heterogeneous multimodal nodes can be embedded and fused layer by layer. In this chapter, we let  $L = 2$  in all experiments. Note that at the final layer, all styles of nodes are projected to a consistent feature space (see Figure 3.4.4(b)). In other words, we let  $d = d_1^{(L)} = d_2^{(L)} = d_3^{(L)} = \dots = d_{2^M-1}^{(L)}$ . Finally, the output embedding  $\mathbf{Z} = \mathbf{Z}^{(L)} \in \mathbb{R}^{N \times d}$  is the fused representations for all multimodal agents.

## 3.5 Experiments

We conduct experiments with the aim of answering two questions: 1) How does HGMF perform for multimodal classification tasks with different percentages of missingness? And, 2) How does HGMF perform compared with inductive and transductive baselines?

### 3.5.1 Simulations

**Datasets:** We perform experiments in both bimodal and trimodal settings, considering three datasets. (1) **ModelNet40** [146] is a 3D CAD dataset, containing 12,311 3D shapes covering 40 common categories, including airplane, bathtub, bed, bench, bookshelf, bottle, bowl, cone, cup, and so on. (2) **NTU** [147] dataset is composed of 2,012 3D shapes from 67 categories, including car, chair, chess, chip, clock, cup, pen, plant leaf and so on. ModelNet40 and NTU are used as bimodal datasets. Following [141], the two input modalities are the two views of shape representations extracted from Multi-view Convolutional Neural Network (MVCNN) [124] and Group-View Convolutional Neural Network (GVCNN) [125]. Both the MVCNN and the GVCNN features are calculated by employing 12 virtual cameras to capture views with a interval angle of 30 degree. (3) **IEMOCAP** [148] dataset consists of a collection of 151 videos of recorded dialogues, with 2 speakers per session for a total of 302 videos across the dataset. Each segment is annotated for the presence of 9 emotions (angry, excited, fear, sad, surprised, frustrated, happy, disappointed and neutral). IEMOCAP is used as trimodal dataset. Following [20, 21],

Table 3.5.1: Simulation Datasets of Modality Gap. NoIS: Number of Modality Preference Styles.  $M$ : number of modalities; and  $|\mathcal{C}|$ : number of classes)

Dataset	#Agents ( $N_{\text{trn}}/N_{\text{val}}/N_{\text{tst}}$ )	#Modalities ( $M$ )	NoIS	$ \mathcal{C} $
ModelNet40	7,387/1,231/3,693	2	4	40
NTU	1,207/201/604	2	4	67
IEMOCAP	2,680/447/1,340	3	7	2

we adopted the same feature extraction scheme for language, visual and acoustic modalities. Language features are obtained from the pre-trained 300-dimensional Glove word embeddings [149], which encode a sequence of transcribed words into a sequence of word vectors; *visual* features are extracted as indicators of facial muscle movement, using Facet [150], include 20 facial action units, 68 facial landmarks, head pose, gaze tracking and HOG features; and, *acoustic* features are obtained from time-series audio using the COVAREP acoustic analysis framework [151], including 12 Mel frequency cepstral coefficients (MFCCs), pitch, voiced/unvoiced segmentation, glottal source, peak slope, and so on. We obtain the above features from CMU Multimodal SDK [112]. Samples in the SDK are annotated according to the presence of four emotion categories (i.e., happy, sad, neutral and angry). For each emotion, we conduct a binary classification task.

**Simulation of Modality Gap:** We evaluate the performance of HGMF under different percentages of data incompleteness. From each multimodal dataset, we prepare the input data by creating several blockwise incomplete multimodal scenarios. Given a Multimodal Incompleteness Ratio (MIR)  $\rho\%$  and suppose the dataset is  $M$ -modal, we randomly delete data from the original complete datasets such that a total of  $\rho\%$  instances have different conditions of missing modalities. In particular, for each incomplete scenario, we let each modality-preference mode has  $N \times \rho / (2^M - 1)\%$  instances. For example, given a bimodal dataset, for each class, we randomly sample  $N \times \rho / 2\%$  instances to remove their first modality, and sample  $N \times \rho / 2\%$  from the rest to remove the second modality.

**Data Split:** All datasets are split into training, validation, and testing sets. In general, to ensure balanced datasets, for each class and each modality-preference mode, 60% data are used for training, 10% for validation, and 30% for testing. Table 3.5.1 shows a summary of the datasets and data split information.

### 3.5.2 Baseline Models

To achieve a comprehensive and comparative analysis of HGMF, we compare it with previously proposed neural multimodal fusion models, which can be divided into two categories. (1) Inductive multimodal fusion models. **Concat** baseline performs fusion by concatenating unimodal features before a fully connected classifier. Our model use the same unimodal embedding networks as this baseline. **Tensor Fusion Network (TFN)** [20] introduces tensor product mechanisms to model unimodal interactions. It is an inductive multimodal fusion model which is proposed without considering the existence of unexpected missing modalities. **Low-rank Fusion Network (LFM)** [21] tries to approximate the expensive tensor products by performing efficient multimodal fusion with modality-specific low-rank factors, which is also an inductive learning method not designed for handling incompleteness. **Multi-task multimodal learning (MTL)** combines the proposed intra-hypernode encoder and pattern-specific classifiers. We directly use the original incomplete data to train this baseline without imputation. All multimodal instances share the same intra-hypernode encoder, but different patterns of instances use different classifiers. This baseline aims to test the impact of graph fusion strategy. (2) Transductive models. **Hypergraph Neural Network (HGNN)** [141] studies deep graph learning and node classification in traditional hypergraph structures. It also applies the approach to multimodal prediction tasks, but simply concatenates unimodal features as the input node features. This method cannot deal with the heterogeneous and highly-interacted incomplete multimodal data.

**Reproducibility:** Details to implement baselines are as follows. First, for the TFN, LFM, Concat and MTL baselines, the model sizes of unimodal networks and final-layer classifiers are the same as those in the proposed model. Other hyperparameters follow their original settings. Second, for the baselines that cannot deal

Table 3.5.2: Hyperparameters for HHG graphs and MGMF models.

Hyper-parameters	HGMF ( $M=2$ )	HGMF ( $M=3$ )
$k$	10	10
$L$	2	2
$\mathcal{M}$	{1, 2}	{1, 2, 3}
$\mathcal{T}$	{1, 2, 3}	{1, 2, ..., 7}
$F_m, \forall m \in \mathcal{M}$	{128, 128}	{128, 128, 128}
$F'$	128	128
$d_p^0, \forall p \in \mathcal{T}$	{512  $\forall p \in \mathcal{T}$ }	{512  $\forall p \in \mathcal{T}$ }
$d_p^1, \forall p \in \mathcal{T}$	{128  $\forall p \in \mathcal{T}$ }	{128  $\forall p \in \mathcal{T}$ }
$d$	64	64
learning rate	1e-4	1e-3

with missing modalities (i.e., Concat, TFN, LFM, HGNN), we preprocess the input data by imputing the missing modalities with zero or values. We have also tested average imputation, but the performances of the baselines using average imputation are worse than those using zero imputation. Thus, we use zero imputation to perform all baselines in this chapter. Third, for HGNN baseline, we preprocess the input data by concatenating all modalities in a node to shape proper feature vectors as input feature matrix. Note that for multimodal dataset may comes with 2D- or 3D-tensor sequential data/features (e.g., image, video, and audio features), we cannot concatenate them using the same way as 1D-tensor data. In order to apply HGNN on such tasks, we take the sum value over the additional dimensions, and then modalities can be concatenated. Graph edges were constructed using the same way in our work. Similar to the proposed model, we also stack two HGNN layers in all experiments.

### 3.5.3 Experimental Setup

**Hyperparameters:** We employed Pytorch3 to implement HGMF and all baselines, and conducted experiments on a single-core GPU. During graph construction, the hypernodes are associated with the original pre-extracted features. Each element in a hypernode can be of different dimension and are not concatenated; the language modality is in 3D-tensor format and others are 2D-tensor. The  $k$  for con-

structuring the high-order hyperedges (in Section 3.1) equals 10 in each experiment. Note that as we let edge weights be 1, we construct a graph that only reflect data connection information. The intra-hypernode encoder parameter set in Algorithm 1 can be summarised as  $\theta_e = \{\Theta_m, \mathbf{U}_S, \mathbf{b}_S | \forall m \in \mathcal{M}, \forall S \in \mathcal{P}(\mathcal{M}) \setminus \emptyset\}$ , Similarly, the MBGAT’s parameter set can be represented as  $\theta_g = \{\vec{\mathbf{a}}_p^{(l)}, \vec{\mathbf{b}}_p^{(l)}, \mathbf{W}_{pq}^{(l)}, \mathbf{V}_{pq}^{(l)} | \forall p, q \in \mathcal{T}, 0 \leq l < L\}$ . We built up HGMF models in bimodal and trimodal settings. Table 3.5.2 summarises the hyperparameters of the HGMF models used in our experiments, including both graph structure and neural network hyperparameters. For intra-hypernode encoder, the embedding dimension of unimodal hidden representations are 64 for visual and language modalities, and 128 for other modalities, which are similar to those in baseline models. Encoded feature dimensions between different patterns can be significantly different. We let the dimension of each factor (an extracted modality-specific or interaction information) equal to 128. We stack two MBGAT layers as the transductive fusion stage of HGMF. At the first layer, we let each pattern’s new feature space dimension is half of input dimension. the final embedding dimension for all patterns equal to 64, meaning that they are encoded into the same space.

**Model Training and Optimization:** The overall training procedure is in Algorithm 1. Since we construct multimodal instances in an HHG graph structure, we formulate the training of our data fusion system HGMF as a semi-supervised node classification task [152, 142]. After embedding the hypernodes in Section 3.3, the fused representations of incomplete multimodal instances is  $Z^{(L)} \in \mathbb{R}^{N \times d}$ . For  $|\mathcal{C}|$ -class classification tasks, given the set of labels for training data  $\mathcal{Y}_{trn} = \{y_i | \forall v_i \in \mathcal{V}_{trn} \subset \mathcal{V}\}$  where  $y_i \in \{c_1, c_2, \dots, c_{|\mathcal{C}|}\}$ , we minimize the cross-entropy loss defined as follows:

$$\mathcal{L} = - \sum_{i \in \mathcal{V}_{trn}} \sum_{c=1}^{|\mathcal{C}|} y_i \cdot \log(\text{softmax}(p(\mathbf{z}_i^{(L)}; \theta_p))). \quad (3.10)$$

where  $\theta_p$  is the parameter of the classifier  $p(\cdot)$ , a fully connected deep neural network shared by all nodes’ fused representation. Parameters of intra-hypernode encoder and multi-fold bilevel graph attention network are initialized with uniform

Table 3.5.3: Test Accuracy (%) on ModelNet40 and NTU ( $M=2$ ) datasets compared with baselines with various MIRs (NMG: No Modality Gap).

Method	ModelNet40					NTU				
	NMG	30%	45%	60%	75%	NMG	30%	45%	60%	75%
Concat	96.54	95.72	94.89	92.33	91.41	89.34	88.45	86.65	83.67	81.56
TFN [20]	98.16	96.02	95.48	93.81	93.3	<b>93.03</b>	<b>91.40</b>	87.97	85.07	84.72
LMF [21]	97.62	96.10	95.94	93.30	92.47	90.94	89.73	85.62	82.25	78.72
HGNN [141]	97.80	96.52	<b>96.10</b>	93.80	91.83	92.73	91.13	87.58	85.41	84.24
MTL	97.40	96.13	95.12	94.12	93.2	90.70	89.91	86.36	83.08	82.39
<b>HGMF (ours)</b>	<b>98.29</b>	<b>97.20</b>	96.02	<b>94.78</b>	<b>93.87</b>	<b>92.38</b>	91.22	<b>88.77</b>	<b>86.41</b>	<b>85.89</b>

distribution. Before training the entire network, we do not consider node connections, treating each node independently to pre-train the intra-hypernode encoder. Then, we train the whole model parameters via the Adam optimizer [153] with tuned learning rates. We repeat the training iterations until the validation set’s accuracy change between two consecutive iterations is sufficiently small.

### 3.5.4 Results and Analysis

We perform classification tasks to evaluate our model against baselines. For multi-class datasets, we report classification accuracy  $\text{Acc-}K$  where  $K$  denotes the number of classes. For binary classification datasets, we report F1 score. The results of our comparative evaluation experiments are summarized in Table 3.5.3 and Table 3.5.4. We examine the efficacy of the proposed HGMF model on both complete scenarios and incomplete scenarios.

#### 3.5.4.1 Comparison on no modality gap

We first evaluate the effectiveness of the proposed method on complete multimodal data. In such scenarios, there is only one pattern in the dataset.

In Table 3.5.3, the results (columns NMG) on both datasets are higher than baselines. It proves that the proposed model can also be used in complete multi-modal fusion. Compared with inductive learning, our improvement on NTU dataset is higher than that on ModelNet40 dataset. Our model performs only slightly better than GNN-based methods because the two modalities that are also used in HGNN may not highly-interacted.

In Table 3.5.4, we only compare with non-graph based inductive learning meth-

Table 3.5.4: F1 scores on three emotion categories in IEMOCAP dataset ( $M = 3$ ) compared with baselines in different incompleteness scenarios (EMO&IS: emotion categories and incompleteness scenarios; NMG: complete).

EMO&IS		Concat	TFN	LMF	MTL	<b>HGMF</b> (ours)
Happy	NMG	86.26	88.72	<b>89.69</b>	87.52	88.87
	30%	86.54	<b>88.74</b>	88.56	87.43	88.70
	45%	85.92	87.51	86.38	86.21	<b>87.93</b>
	60%	85.73	87.00	85.87	86.74	<b>87.24</b>
	75%	83.27	86.53	84.56	86.02	<b>86.02</b>
Sad	NMG	83.69	85.09	85.45	84.71	<b>85.72</b>
	30%	83.23	<b>85.25</b>	84.35	83.97	84.67
	45%	82.61	<b>84.58</b>	83.47	83.88	84.55
	60%	81.35	82.04	81.26	81.97	<b>83.33</b>
	75%	80.69	81.95	79.89	81.06	<b>82.32</b>
Angry	NMG	86.74	88.22	<b>88.74</b>	87.75	88.38
	30%	85.93	88.02	87.59	87.66	<b>88.14</b>
	45%	84.86	87.42	86.38	86.03	<b>87.81</b>
	60%	83.29	86.17	85.25	85.26	<b>87.34</b>
	75%	83.71	85.46	84.68	85.02	<b>86.89</b>

ods, because the graph-based baseline HGNN cannot deal with modalities in different dimensional tensor in IEMOCAP dataset, and also cannot handle the modality interactions. From the comparisons with inductive learning, our model’s results outperform Concat and MTL, and either higher or similar to TMF and LMF. It is because baselines do not need to impute data in complete scenarios, so that our model and baselines are under the same circumstances.

### 3.5.4.2 Comparison on different degrees of modality gap

Now we consider the more realistic setting where blockwise modality gap is present. We evaluate the influences of modality gap by changing the multimodal incompleteness ratio from 30% to 75% with a intermittent 15%. As shown in both Table 3.5.3 and Table 3.5.4, our method tends to outperform baselines while there are higher modality gap.

From Tables 3.5.3 and Table 3.5.4, as more modalities are missing, the performances of Concat and LMF drop dramatically, while TFN, MTL and the proposed

HGMF do not drop too much. It is because Concat and LMF do not explore much inter-modality interactions, and their network neurons can be significantly affected by attacked values at the beginning. Also, when the missing ratio is not lower enough (i.e., less than 45%), the results show that TFN does not drop too much as the proposed method. It may be due to that the zero imputation can be viewed as dropout layer at the beginning, and the dropout at a low rate does not influence the higher-level neurons too much.

### 3.6 Conclusion

In this chapter, we have studied the multi-agent collaborative multimodal fusion problem, focusing on a specific personalization pattern—the *modality preferences* of individual MML agents. To address the *modality gap* challenge during inter-agent collaboration, we have presented the heterogeneous graph-based multimodal fusion (HGMF) framework. HGMF exploits a heterogeneous hypernode graph (HHG) structure to capture modality interactions from the intra-agent available modalities (intra-hypernode encoder) as well as learns the relationships among different modality-preference styles (MBGAT). The idea is to exploit the powerful graphs representations to enable the agents having certain unavailable modalities to derive relevant missing information from other agents who have such information. Through the node-to-node information propagation within HHG, the proposed HGMF framework effectively fuses multimodal data into joint representations and makes decisions based on them. Our experimental results demonstrated the significance of our approach.



# Chapter4

## Meta-learning based Collaborative Visual-language Understanding with Concept Personalization

### 4.1 Introduction

In this chapter, we address situations where only **concept preferences** of individual MML agents are allowed: all agents deal with the same classification task and take the same set of input modalities, but different agents might have their own interested class categories or their own definitions for various class categories, i.e., *personalized label spaces*. It is worth noting that, the label space of classification can be represented at different **concept granularities**. For instance, in computer vision, image classification labels correspond to each sample, image segmentation labels correspond to each pixel, and video event recognition labels correspond to each frame; likewise, text classification labels are associated with each text sequence, while named entity recognition segmentation labels are associated with each word, and so on. Sample-wise classification has been more widely explored in CoMML, whereas tasks with smaller concept granularity (e.g., pixelwise or token-wise) remain less investigated. Therefore, in this chapter, we focus on exploring agents' concept preferences at a small granularity.

A real-world example of such a setting is *Visually-rich Document Entity Re-*

*trieval (VDER) where users have their own interested sets of entities* [154, 155, 156]. VDER is a visual-language bimodal task that aims to extract key information (e.g., date, address, signatures) from the document images (i.e., scanned documents composed of structured and organized information, such as invoices and receipts). These key information are the target *concepts in document images*, and extracting them from document images is a fine-grained concept classification task (i.e., the token tagging or image segmentation task). In real-world VDER systems, concept personalization plays a critical role as new document types continuously emerge at a constant pace and each of them might have its unique set of target entity categories. This leads to the practical Entity-personalized VDER task. This chapter will study and experiment on this special use case, **Entity-personalized VDER (EpVDER)**, via Multi-agent Collaboration, as an endeavor to delve into the research on “CoMML with concept preferences”.

Existing efforts in VDER have leveraged pre-trained language models [157] or prompt tuning [158] to obtain transferable knowledge from a source domain and apply it to a target domain, where a small number of document images are labeled for fine-tuning. These prior works address the label space in a granularity of **document level**, assuming a *globally predefined* entity space and *balanced* entity occurrences across documents. However, Entity-personalized VDER presents distinctive **challenges** that cannot be addressed by these document-level VDER approaches. (1) *Limited occurrences* of certain entities in labeled documents (*few-shot scenarios*). (2) The numbers of occurrences of each entity category maintain a significant *imbalance* across documents. (3) The small number of personally-relevant concepts increases the complexity and prevalence of out-of-distribution contents. Additionally, (4) prior methods face difficulties in quickly adapting the model to each agent’s specific label space.

To address these challenges, we initiate the investigation for the unexplored EpVDER. We adopt a meta-learning based framework built upon pretrained language models, along with several proposed techniques for achieving *task personalization* and handling *out-of-task distribution* contents. With the help of the meta-learning paradigm, **(1)** the learning experiences on some example tasks could be effectively utilized and **(2)** the domain gap between the pre-trained model and novel

EpVDER tasks is largely reduced, promoting quicker and more effective fine-tuning on future novel entity types. Yet we found popular meta-learning algorithms [39, 51, 159] are still not robust to the specific challenges of EpVDER tasks. Specifically, the background context that does not belong to the agent’s personalized entity types occupies most of the predictive efforts, and also, such noisy contextual information varies a lot across tasks and documents. To address this, we propose **self-aware meta-learning** techniques (ContrastProtoNet, ANIL+HC, etc.) to allow the meta-learners to be aware of those multi-mode out-of-task distribution background and achieve fast adaptation to the agent’s personalized entity types. Here is a summary of the contributions of this chapter.

- Within the realm of CoMML, this is one of the first attempts that investigate concept shifts among agents at a *fine-grained concept level*, departing from prior works that primarily concentrate on sample-wise classification.
- In the context of visually-rich document understanding, this is the first work examining the VDER problem through the lens of *entity personalization*, which offers a supplementary research perspective alongside the majority of document-level studies.
- We propose a meta-learning based framework for solving the newly introduced task. While vanilla meta-learning approaches have limitations on this task, we propose several self-aware meta-learners to enhance task personalization by dealing with out-of-task distribution.
- Experiment results demonstrate our proposed approaches significantly improve the performance of baseline methods.

## 4.2 Related Works

In addition to the literature reviewed in Section 2.4, since our proposed techniques will be applied to Visually-rich Documents (VD) in NLP, we will also review three lines of work: 1) models for *general* VD understanding; 2) the *particular* Entity Retrieval (ER) task for VD; and 3) related meta-learning approaches for VD-like problems in CV, NLP, and Multimodal domains.

**General VD Understanding.** Pretrained LLMs for *general* VD understanding have shown strong performance in general understanding of visually-rich multi-modal documents, and therefore, can serve as *pretrained prior* for Few-shot VDER. There are many LLM candidates our framework can use as the pretrained encoder, such as LayoutLM [160], which extends the standard BERT [161], and the recent LayoutLMv3 [162] and DocGraphLM [163], which show improvements by using advanced cross-modal alignment or local-global position embeddings. In this chapter, we use the basic BERT model for experiments since our focus is how to improve the *post* fine-tuning on few-shot downstream tasks, without a restrict on the specification of LLM type. Extending this research to other pretrained Document Understanding LLMs could be one of future works.

**Entity Retrieval from VD.** The *particular* Entity Retrieval (ER) tasks for VD have been studied for many years using Deep Neural Networks, Graph Neural Networks, or traditional models [164, 165], or empowered by the contextual prior knowledge provided by VD-understanding LLMs [154, 156, 166]. Recent advancements in Few-shot VDER predominantly rely on pretrained LLMs and prompt design, followed by fine tuning on a small number of VD documents [167, 157]. Despite their success, these works address the situation where the entity label space is fixed over tasks and entity occurrences do not shift a lot. However, VDER specializing entity-space personalization, where the concepts or entity categories learned on each agent is user-specific, has garnered comparatively less attention in prior research.

**General Meta-learning for VD-like Problems.** Meta-learning [51, 39] has been exploited in various AI/ML domains in problems like Multitask Learning (MTL), Few-shot Learning (FSL), Federated Learning (FL), and Optimization [168]. In CV or NLP domains, there are two tasks closely related to Entity-personalized VDER that also have been addressed using meta-learning: **(1)** Few-shot object detection or segmentation [169, 170] aims at localizing objects in visual data, where each object can be treated as an entity in VDER; and, **(2)** Few-shot Named Entity Recognition (NER) aims at labelling tokens within a contextual

text sequence [171, 172]. While few-shot NER and object detection algorithms can provide inspirations for few-shot VDER, the challenges we face and methodology details are relatively different. Beyond them, the scope of this chapter falls within the field of Multimodal Learning. Multimodal Few-shot Learning has historically been addressed by meta-learning approaches [173, 174]. Our work differs from these methods in two key aspects: first, we combine the advantages of both Large Language Model’s prior knowledge and meta-learning for balancing generalization and concept personalization; second, to enhance the personalization performance of EpVDER, we incorporate Out-of-distribution detection techniques in few-shot context [175].

## 4.3 Problem Formulation

### 4.3.1 Local Task Setting on Each Agent

**General VDER.** A document image is processed through Optical Character Recognition (OCR) [176] to form a sequence of tokens  $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_L]$ , where  $L$  is the sequence length and each token  $\mathbf{x}_l = \{\mathbf{x}_l^{(v)}, \mathbf{x}_l^{(p)}, \mathbf{x}_l^{(b)}, \dots\}$  is composed of various information from the image and text modalities: the token id ( $v$ ), the 1d position ( $p$ ) of the token in the sequence, the bounding box ( $b$ ) representing the token’s relative 2d position, scale in the image, and so on<sup>1</sup>. The goal is to predict  $\mathbf{Y} = [y_1, y_2, \dots, y_L]$ , which assigns each multimodal token  $\mathbf{x}_l$  a categorical label  $y_l$  to indicate either the token is one of entities in a set of *predefined* entity types or does not belong to any entity (denoted as 0 class).

**Definition 4.1 (Entity-personalized VDER (EpVDER)).** Each agent deals with an EpVDER task specialized by its own defined target entity types. The EpVDER task at each agent is formulated as  $\mathcal{D} = \{S, Q, \mathcal{E}\}$ , which consists of a train (*support*) set  $S$  containing  $M_s$  documents, a test (*query*) set  $Q$  containing  $M_q$

---

<sup>1</sup>In this chapter, we assume modality uniformity. All the samples across different agents have the same input modality sets so that we omit the tilde symbol from the input notations.

documents, and a personalized *target* class set  $\mathcal{E}$  containing  $U$  target entity types

$$\begin{aligned} S &= \{(\mathbf{X}_1, \mathbf{Y}_1), (\mathbf{X}_2, \mathbf{Y}_2), \dots, (\mathbf{X}_{M_s}, \mathbf{Y}_{M_s})\} \\ Q &= \{\mathbf{X}_1^*, \mathbf{X}_2^*, \dots, \mathbf{X}_{M_q}^*\} \\ \mathcal{E} &= \{e_1, e_2, \dots, e_U\}, \end{aligned} \tag{4.1}$$

where  $\mathbf{X}_j = [\mathbf{x}_{j1}, \mathbf{x}_{j2}, \dots, \mathbf{x}_{jL}]$  is the sequence of multimodal token features of document  $j$  and  $\mathbf{Y}_j = [y_{j1}, y_{j2}, \dots, y_{jL}]$  is the sequence of token labels corresponding to  $\mathbf{X}_j$ , and  $e_c$  denotes the  $c$ -th entity type in  $\mathcal{D}$ . Note that we have referenced the standard few-shot learning setup [39] when defining Eq.(4.1). Though our definitions have many differences from the standard definition as follows. “***U*-way**” refers to the  $U$  unique entity types the user is interested in, reflecting task personalization. It is important to highlight that within  $S$  and  $Q$  documents, there may exist entities that fall outside the  $U$  target classes ( $e' \notin \mathcal{E}$ ). These entity types come from the *out-of-distribution* in contrast to what the task  $\mathcal{D}$  aims to train on, which do not attract user interest, remain unlabeled, and thus are treated as the background 0 class. “**Soft- $K$ -shot**” refers that, among the  $M_s$  labelled documents in  $S$ , the total number of *occurrences* of each entity type  $e \in \mathcal{E}$  is within a range  $K \sim \rho K$ , where  $\rho > 1$  is the softening hyperparameter. An *entity occurrence* is defined as a contiguous subsequence in the document with the same entity type as labels. We do not impose a strict constraint on the exact count  $K$  since the entity-level personalization scenario implies that the frequency of entity occurrences may vary dramatically from one document to the other, which makes it difficult to set a strict limit. For instance, an entity type may occur more frequent in some documents and less so in others. The right area of Figure 4.3.1 shows an example  $U$ -way soft- $K$ -shot VDER task. The **local objective** of  $\mathcal{D}$  is to obtain a **personal model**  $f(\cdot; \theta)$  that assigns each token as either one of  $\mathcal{E}$  (task-personalized entity types) or 0 (background or out-of-task entity types), based on the few labeled entity occurrences for those in  $\mathcal{E}$  in support set  $S$ , such that the model achieves high performance on the query set  $Q$ .

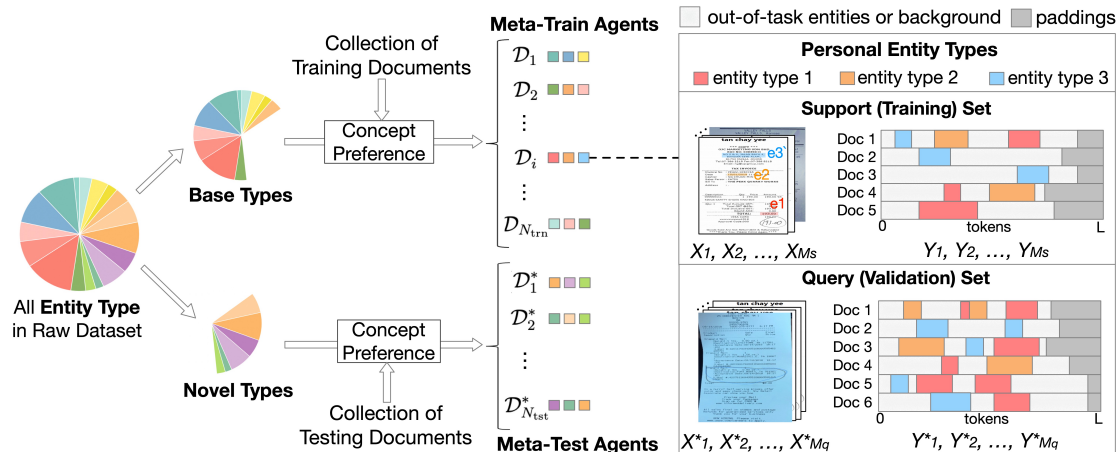


Figure 4.3.1: Problem setting of Visually-rich Document Entity Retrieval formulated in a meta-learning based multi-agent collaboration framework, with personally-defined entity categories for each agent. Different colors represent different entity categories. The pie chart split on the left indicates that the target classes in testing tasks are not seen in training tasks. On the right area, we show an example 3-way soft-2-shot task. In this example,  $\rho = 2$ .

### 4.3.2 Multi-agent Collaborative EpVDER

**Distribution over EpVDER Tasks.** Based on the above formulation for a single EpVDER task, we further formulate a task distribution  $P(\mathcal{D})$  over EpVDER tasks. Assume there is a large pool of entity types  $\mathcal{C}$  corresponding to the domain of  $P(\mathcal{D})$ . The task on any agent  $i \in [N]$  can be sampled as,

$$\mathcal{D}_i = \{S_i, Q_i, \mathcal{E}_i\} \sim P(\mathcal{D}), \quad (4.2)$$

where its target entity types come from the class pool  $\mathcal{E}_i \subset \mathcal{C}$ . A collection of tasks  $\widehat{\mathcal{D}}_{\text{colab}}$  can be viewed as a simulation of multiple EpVDER agents.  $\widehat{\mathcal{D}}_{\text{colab}}$  is *heterogeneous* according to *concept shifts*  $\mathcal{E}_1 \neq \mathcal{E}_2 \dots \neq \mathcal{E}_N$  across EpVDER agents.

**Definition 4.2 (Collaborative EpVDER with Entity Shifts).** We formulate a multi-agent systems as  $\widehat{\mathcal{D}}_{\text{colab}} = \{\widehat{\mathcal{D}}_{\text{colab}}^{\text{trn}}, \widehat{\mathcal{D}}_{\text{colab}}^{\text{tst}}\}$  consisting of  $N_{\text{trn}}$  agents that

have labelled validation set and  $N_{\text{tst}}$  agents that only have a few labelled examples

$$\begin{aligned} \widehat{\mathcal{D}}_{\text{colab}}^{\text{trn}} &= \{\mathcal{D}_1, \mathcal{D}_2 \dots \mathcal{D}_{N_{\text{trn}}}\} \text{ where } \mathcal{D}_i = \{S_i, Q_i, \mathcal{E}_i\} \sim P(\mathcal{D}) \\ \widehat{\mathcal{D}}_{\text{colab}}^{\text{tst}} &= \{\mathcal{D}_1^*, \mathcal{D}_2^* \dots, \mathcal{D}_{N_{\text{tst}}}^*\} \text{ where } \mathcal{D}_i^* = \{S_i^*, Q_i^*, \mathcal{E}_i^*\} \sim P(\mathcal{D}) \end{aligned} \quad (4.3)$$

where any training agent focuses on personal entity types (i.e., concepts)  $\mathcal{E}_i \subset \mathcal{C}_{\text{base}}$  from a set of base classes  $\mathcal{C}_{\text{base}}$  and any testing agent focuses on personal concepts  $\mathcal{E}_i^* \subset \mathcal{C}_{\text{novel}}$  sampled from  $\mathcal{C}_{\text{novel}}$ . The query sets of training agents are treated as validation sets,  $Q_i = \{(\mathbf{X}_{ij}^*, \mathbf{Y}_{ij}^*)\}_{j=1}^{M_{qi}}$  for  $\forall \mathcal{D}_i \in \widehat{\mathcal{D}}_{\text{colab}}^{\text{trn}}$ . The query sets of testing agents are unlabelled testing data, that is,  $Q_i^* = \{\mathbf{X}_{ij}^*\}_{j=1}^{M_{qi}^*}$ ,  $\forall \mathcal{D}_i^* \in \widehat{\mathcal{D}}_{\text{colab}}^{\text{tst}}$ . Figure 4.3.1 shows an overview of the simulation of the multi-EpVDER-agent collaboration. We define the **global objective** following [39]. That is, we aim to train a global generalization model  $\theta^* = \arg \min_{\theta} \mathbb{E}_{\mathcal{D}_i \sim P(\mathcal{D})} \mathcal{L}_i(\text{Adapt}(\mathcal{D}_i, \theta))$  for  $P(\mathcal{D})$  such that any agent’s task  $\mathcal{D}_i$  can take advantage of it to quickly obtain a good **personal** model  $\theta_i^* = \text{Adapt}(\mathcal{D}_i, \theta^*)$ , where  $\mathcal{L}_i(\cdot)$  is the local loss on query set and  $\text{Adapt}(\cdot, \cdot)$  is the local training result from the support set.

## 4.4 Methodology

We propose a meta-learning (i.e., learning-to-learn) framework to facilitate the collaborative learning between the heterogeneous EpVDER agents with concept shifts. Different from the recent advancements based on pre-training or prompts [157, 158], meta-learning helps to significantly promote *quick* adaptation and improve model *personalization* on task-specific entity types.

The proposed framework consists of three components: **(1)** a multimodal *encoder* (Section 4.4.1) that encodes the document images within a task into a *task-dependent embedding space* (Section 4.4.2); **(2)** a *token labelling* function (Section 4.4.3); and **(3)** a *meta-learner* built upon the encoder-decoder model, where we propose two task-personalized meta-learning methods (Section 4.4.4). Figure 4.4.1 shows an overview of the framework.



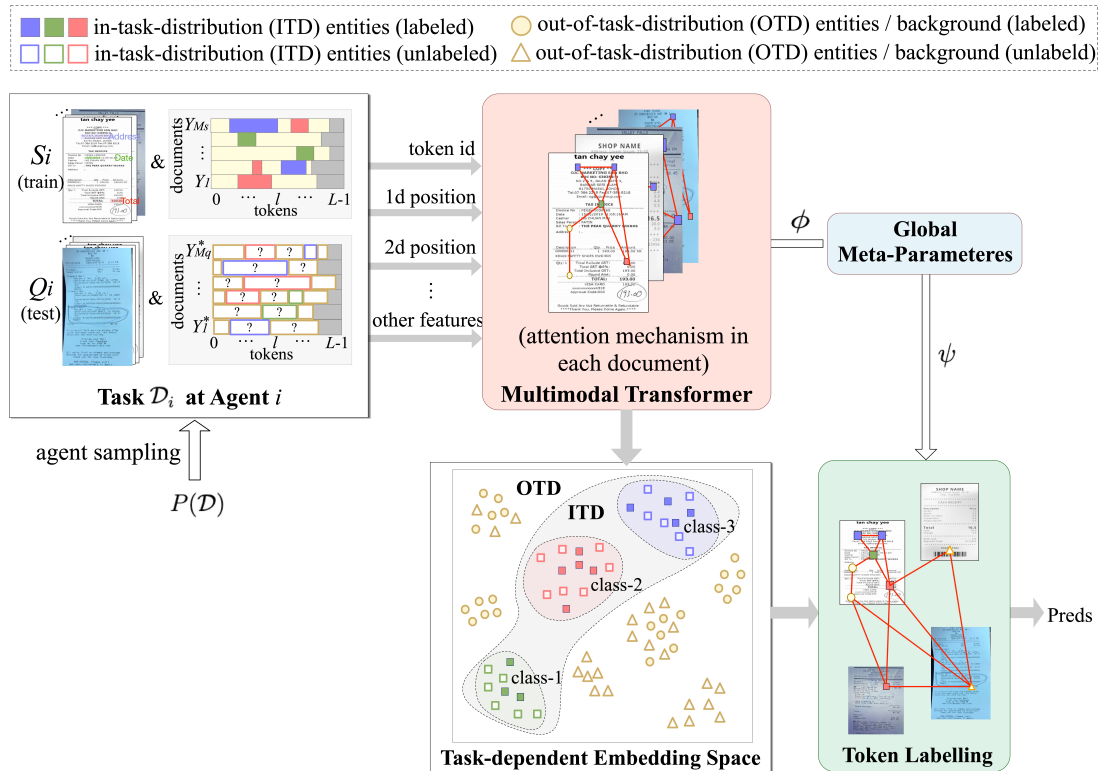


Figure 4.4.1: The proposed self-aware meta-learning framework. The framework is applicable to both the metric-based method (aiming to learn global parameters  $\theta = \phi$ ) and gradient-based method (aiming to learn global parameters  $\theta = \{\phi, \psi\}$ ).

#### 4.4.1 Transformer-based Multimodal Encoder

We consider an encoder network represented by a parameterized function  $f_{\text{enc}}(\cdot; \phi)$  with parameters  $\phi$ . The encoder aims to capture the cross-modal semantic relationships between tokens in a document image. To achieve this, we employ a BERT-base Transformer [161] with an additional positional embedding layer for the 2d position of each input token, through which the complex spatial structure of the input document can be incorporated and then interacted with the textual contents via attention mechanisms. The embedding of token  $l$  in the document image  $j$  of task  $\mathcal{D}_i$  is computed as  $\mathbf{h}_{ijl} = f_{\text{enc}}(\mathbf{x}_{ij}; \phi)_l$ . In practice, before meta-training, the multimodal Transformer is pretrained on the IIT-CDIP dataset [177].

### 4.4.2 Task-dependent Embedding Space

Through the multimodal encoder, each task  $\mathcal{D}_i$  is encoded to a *task-dependent embedding space*. As illustrated in Figure 4.4.1, on the task-dependent embedding space, there are all the token embeddings in the task:

$$\mathcal{H}_i = \{\mathbf{h}_{ijl} | l \in [L], (\mathbf{x}_{ij}, \mathbf{y}_{ij}) \in S_i \cup Q_i\}. \quad (4.4)$$

There are several properties on the task’s embedding space: **(1) First**, in addition to *in-task distribution* (ITD) entities from the target classes, there exists a large portion (nearly 90% as observed in our dataset FewVEX) of *out-of-task distribution* (OTD) entities or background, which serve as the context for ITD entities but dominate the task’s embedding space. **(2) Second**, the OTD entities follows a *multi-mode* distribution  $P_i^{\text{OTD}}$  that consists of several unimodal distributions, each of which represents an outlier entity type aside from ITD. **(3) Finally**, it is not guaranteed that each unimodal component of  $P_i^{\text{OTD}}$  is observable in the train set  $S_i$ —in many cases, an OTD entity type could occur in the query documents but is absent in the support documents. To sum up, the OTD distribution in a EpVDER task is complex, dominates the entire task, and may vary between documents.

### 4.4.3 Token Labelling

On the basis of the task-dependent embedding space, the token labelling or decoding process can either leverage a *parameterized* decoder  $f_{\text{dec}}(\cdot; \psi)$  that acts as the classification head, or rely on *non-parametric* methods, like nearest neighbors.

### 4.4.4 Self-aware Meta Learners

We consider two main categories of the meta-learning approaches: the gradient-based and the metric-based meta-learning, on each of which we propose our own methods. We specifically pay attention to two properties when solving the Entity-personalized VDER tasks: **1) Few-shot out-of-task distribution detection**, which aims to distinguish the ITD (i.e., the target  $U$  entity types) against the OTD (i.e., background or any outlier entity type). **2) Few-shot token labelling**

for **in-task distribution tokens**, which assigns each ITD token to one of the  $U$  in-task entity types.

#### 4.4.4.1 Self-aware ContrastProtoNet

We first focus on *metric-based* meta-learning [51, 52]. The goal is to learn *meta-parameters*  $\phi$  for the encoder network, generally shared by all tasks  $\mathcal{D}_i \sim P(\mathcal{D})$ , such that, on each task’s specific embedding space, the distances between token points in  $S_i$  and  $Q_i$  are measured by some metrics, e.g., Euclidean distances.

**ProtoNet with or without Estimated OTD.** One of the most popular and effective metric-based meta-learning methods is the Prototypical Network (ProtoNet) [51]. For each EpVDER task  $\mathcal{D}_i = \{S_i, Q_i, \mathcal{E}_i\}$ , the prototype for each entity type  $e \in \mathcal{E}_i$  can be computed as the mean embedding of the tokens from  $S_i$  belonging to that entity type, that is,  $\mu_{i,e} = 1/|I_e^{\text{trn}}| \sum_{(j,l) \in I_e^{\text{trn}}} \mathbf{h}_{ijl}$ , where  $I_e^{\text{trn}}$  is a collection of the token indices for the type- $e$  tokens in the support set. For the out-of-task distribution (OTD), one may consider to estimate its mean embedding as an extra 0-type prototype:  $\bar{\mu}_i = 1/|I_{\text{OTD}}^{\text{trn}}| \sum_{(j,l) \in I_{\text{OTD}}^{\text{trn}}} \mathbf{h}_{ijl}$ .

A problem of the vanilla methods is that there is no specific mechanism distinguishing the ITD entities against the OTD entities, which are weakly-supervised and partially observed from a multi-mode distribution  $P_i^{\text{OTD}}$ . The prototype  $\bar{\mu}_i$  is a biased estimation of the mean of  $P_i^{\text{OTD}}$  and the covariance of  $P_i^{\text{OTD}}$  can be larger than any of the ITD classes. In consequence, the task-specific ITD classes may not be clearly distinguished from the OTD classes on the task-dependent embedding space and most of tokens will be misclassified.

Regarding the above challenges, we propose a self-aware method that adopts two techniques to boost the performance.

**Meta Contrastive Loss.** During meta-training, we encourage the  $U$  ITD entity types to be distinguished from each other as well as far away from any unimodal component of OTD. To achieve this, we adopt the idea from supervised contrastive learning [178] to compute a *meta contrastive loss* (MCON) from each task, which will be further used to compute meta-gradients for updating the meta-parameters

$\phi$ . Intuitively, our meta-objective is that the query tokens from the ITD type- $e$  should be pushed away from any OTD tokens and other types of ITD tokens within the same task, and should be pulled towards the prototype  $\boldsymbol{\mu}_{i,e}$  of support tokens and the other query tokens belonging to the same entity type. Formally, let  $I_{\text{ITD}}^{\text{val}} = \{(j, l) | l \in [L], (\mathbf{x}_j^*, \mathbf{y}_j^*) \in Q_i, \mathbf{y}_{ijl}^* \in \mathcal{E}_i\}$  denote a collection of ITD validation tokens. The meta contrastive loss computed from  $\mathcal{D}_i$  is

$$\begin{aligned} \mathcal{L}_i^{\text{MCON}} &= \sum_{(j,l) \in I_{\text{ITD}}^{\text{val}}} \frac{-1}{|A^+(j,l)|} \sum_{\mathbf{v} \in A^+(j,l)} a_{ijl}(\mathbf{v}) \\ a_{ijl}(\mathbf{v}) &= \log \frac{\exp(\mathbf{h}_{ijl}^\top \mathbf{v})}{\sum_{\mathbf{u} \in A(j,l)} \exp(\mathbf{h}_{ijl}^\top \mathbf{u})}. \end{aligned} \quad (4.5)$$

For each *anchor*, i.e., the ITD validation token  $l$  in document  $j$ , we let  $A^+(j, l) = \{\mathbf{h}_{irm} | (r, m) \in I_{\text{ITD}}^{\text{val}} \setminus \{(j, l)\}, \mathbf{y}_{ijl}^* = \mathbf{y}_{irm}^*\} \cup \{\boldsymbol{\mu}_{i,e} | e \in \mathcal{E}_i, \mathbf{y}_{ijl}^* = e\}$  denote a collection of the *positive* embeddings/prototype for the anchor and let  $A(j, l) = \{\mathbf{h}_{irm} | (r, m) \in I_{\text{ALL}} \setminus \{(j, l)\}\} \cup \{\boldsymbol{\mu}_{i,e} | e \in \mathcal{E}_i\}$  contain all the ITD/OTD embeddings and prototypes ( $I_{\text{ALL}} = \{(j, l) | l \in [L], (\mathbf{x}_j, \mathbf{y}_j) \in S_i \cup Q_i\}$ ) in  $\mathcal{D}_i$ .

**Unsupervised OTD Detector.** During the inference time for novel entity types, we adopt the nonparametric token-level nearest neighbor classifier, which assigns  $\mathbf{x}_{ijl}$  the same label as the support token that is nearest in the task’s embedding space:

$$\hat{y}_{ijl}^{\text{nn}} = \operatorname{argmax}_{y_{irm} \text{ where } (r,m) \in I_{\text{ALL}}^{\text{trn}}} \mathbf{h}_{ijl}^\top \mathbf{h}_{irm}, \quad (4.6)$$

where  $I_{\text{ALL}}^{\text{trn}} = \{(r, m) | m \in [L], (\mathbf{x}_r, \mathbf{y}_r) \in S_i\}$ . The ITD or OTD entity tokens in  $Q_i$  should be closer to the corresponding ITD or OTD tokens in  $S_i$  that belong to the same entity type. However, since the embedding space dependent on the support set is not sufficiently rich, the network may be blind to properties of the out-of-task distribution  $P_i^{\text{OTD}}$  that turn out to be necessary for accurate entity retrieval. To tackle this, we exploit an unsupervised out-of-distribution detector [179] operating on the task-dependent embedding space, in assistance with the classifier. Specifically, we define an OTD detector:  $\hat{y}_{ijl} = 0$  if  $r(\mathbf{h}_{ijl}) \geq R_i$ ; otherwise,  $\hat{y}_{ijl} = \hat{y}_{ijl}^{\text{nn}}$ , where  $R_i$  is the task-dependent uncertainty threshold and  $r(\mathbf{h}_{ijl})$  is defined as the OTD score of each token computed as its minimum Mahalanobis

distance among the  $U$  ITD classes:  $r(\mathbf{h}_{ijl}) = \min_{e \in \mathcal{E}_i} (\mathbf{h}_{ijl} - \boldsymbol{\mu}_{i,e})^\top \Omega_{i,e}^{-1} (\mathbf{h}_{ijl} - \boldsymbol{\mu}_{i,e})$ . Here,  $\Omega_{i,e} = \sum_{(j,l) \in I_e^{\text{trn}}} (\mathbf{h}_{ijl} - \boldsymbol{\mu}_{i,e})^\top (\mathbf{h}_{ijl} - \boldsymbol{\mu}_{i,e})$  is the covariance matrix for entity type  $e$  computed from the type- $e$  tokens in the support set ( $I_e^{\text{trn}}$ ). The higher OTD score indicates the more likely the token belongs to the background.

#### 4.4.4.2 Gradient-based Meta-learning with OTD Detection

For *gradient-based meta learning*, the goal is to learn the meta-parameters  $\boldsymbol{\theta} = \{\boldsymbol{\phi}, \boldsymbol{\psi}\}$  globally shared over the task distribution  $P(\mathcal{D})$ , which can be fast fine-tuned for any given individual task  $\mathcal{D}_i$ .

**Computation-efficient Meta Optimization.** Although MAML [39] is the most widely adopted approach, the fact that it needs to differentiate through the fine-tuning optimization process makes it a bad candidate for Transformer-based encoder-decoder model, where we need to save a large number of high-order gradients for the encoder. Instead, we consider two alternatives which require less computing resources and more efficient. **ANIL** [180] employs the same bilevel optimization framework as MAML but the encoder is not fine-tuned during the inner loop. The features from the encoder are reused in different tasks, to enable the rapid fine tuning of the decoder. **Reptile** [57] is a first-order gradient based approach that avoids the high-order meta-gradients. To further boost training efficiency, we exploit Federated Learning [181, 73] for meta-optimization of Transformer.

**Self-aware Hierarchical Classifier (HC).** A vanilla classifier can achieve high performance in the label-sufficient VDER. However, it turns out to be not robust in few-shot EpVDER tasks because of the existence of the complicated out-of-task entities—the models usually either get overconfident on the  $U$  IID entity types or fail to distinguish target entities from the OTD background. For this reason, we incorporate OTD detection into the decoder and propose a hierarchical classifier, which has two classifiers  $\boldsymbol{\psi} = \{\boldsymbol{\psi}_1, \boldsymbol{\psi}_2\}$ : 1) *binary* classifier  $f_{\boldsymbol{\psi}_1}^{\text{bin}}$ , so that all ITD tokens are classified against OTD ones, and 2) *entity* classifier  $f_{\boldsymbol{\psi}_2}^{\text{ent}}$ , so that ITD tokens are classified to one of the  $U$  entity types of the task. Specifically, suppose  $P_i^{\text{OTD}}$  and  $P_i^{\text{ITD}}$  denotes the OTD and ITD of the task  $\mathcal{D}_i$ , respectively. The probability that the token  $\mathbf{h}_{ijl}$  is from OTD is denoted as  $P(y_{ijl} = 0) = f_{\boldsymbol{\psi}'_1}^{\text{ent}}(\mathbf{h}_l)$ , which is used as

the OTD score to weight the entity prediction. The probability that the token is the entity type- $e$  is computed as  $P(y_{ijl} = e | \mathbf{x}_{ijl} \in P_i^{\text{ITD}}) = (1 - P(y_{ijl} = 0)) f_{\psi'_{i2}}^{\text{ent}}(\mathbf{h}_{ijl})_e$ .

## 4.5 Experiments

### 4.5.1 Dataset

We setup a novel simulation, FewVEX, to evaluate Few-shot VDER tasks.

**Collection of Entity Types and Documents.** First, we collect the entity types  $\mathcal{C}$  associated with the task distribution  $P(\mathcal{D})$  and a set of document images  $\mathcal{D}$  annotated by these entity types. We use a source dataset, Consolidated Receipt Dataset for post-OCR parsing (CORD) dataset [182], that are widely used in normal large-scale document understanding tasks such as entity recognition, parsing, and information extraction. CORD consists of 1000 receipt images of texts and contains 6 superclasses (menu, void menu, subtotal, void total, total, and etc) which are divided into 30 fine-grained subclasses. For different entity types, the total numbers of entity occurrences over the CORD images are highly imbalanced, ranging from 1 occurrence of entity "void menu (nm)" to 997 occurrences of "menu (price)". From the two datasets, we obtain a combined source dataset denoted as  $\mathcal{D}$ , which contains 1199 unique document images with original annotations on 33 classes. However, we observe that some fine-grained classes in CORD occurs in less than  $\max_i(M_{si} + M_{qi})$  images, the maximum number of documents within individual tasks. This will result in a large amount of repetitive usage of the same documents within one task and between different tasks. Therefore, we further sort the 33 classes by the number of unique document images where they occur and then discard three entity types that occurs in low frequency. To sum up, we finally have a total of  $|\mathcal{C}| = 30$  entity types and  $|\mathcal{D}| = 1199$  unique document images annotated by these entity types. The pie chart (on the left) in Figure 4.3.1 illustrates the number of occurrences of the final entity types.

**Simulation of Agent Entity Personalization.** To ensure that testing tasks in  $\widehat{\mathcal{D}}_{\text{colab}}^{\text{tst}}$  focus on novel classes that are unseen during meta-training  $\widehat{\mathcal{D}}_{\text{colab}}^{\text{trn}}$ , we should

Table 4.5.1: Simulation of fine-grained concept shifts as Few-shot VDER tasks.

Meta Training (from $\mathcal{C}_{base}$ )		Meta Testing (from $\mathcal{C}_{novel}$ )		Range of $U$
# Entity Types	# Tasks	# Entity Types	# Tasks	
18	3000	5	128	[1, 5]

split the total entity types  $\mathcal{C}$  into two separate sets  $\mathcal{C} = \mathcal{C}_{base} \cup \mathcal{C}_{novel}$ ,  $\mathcal{C}_{base} \cap \mathcal{C}_{novel} = \emptyset$  such that  $\mathcal{C}_{base}$  is used for meta-training and  $\mathcal{C}_{novel}$  for meta-testing. Specifically, we use a split ratio  $\gamma$  to control the number of novel classes and randomly choose  $\gamma|\mathcal{C}|$  entity types from  $\mathcal{C}$  as  $\mathcal{C}_{novel}$ . Then,  $\mathcal{C}_{base} = \mathcal{C} \setminus \mathcal{C}_{novel}$ . Note that for the cases that some entity types occurs in less number of documents than the others, we set a threshold  $U$  and any entity type that occurs in less than  $U$  documents are forced to be one of the novel classes. Each individual task  $\mathcal{D} = \{S, Q, \mathcal{E}\}$  in either  $\widehat{\mathcal{D}}_{colab}^{trn}$  or  $\widehat{\mathcal{D}}_{colab}^{tst}$  can be generated by the following steps. (2) *Personalized Class Sampling*. The target classes of task  $\mathcal{E}$  is generated by randomly sampling  $U$  entity types from either  $\mathcal{C}_{base}$  (for the training task) or  $\mathcal{C}_{novel}$  (for the testing task). (3) *Document Sampling*. Given the  $U$  target classes, we then collect document images that satisfies the few-shot setting defined in Section 5.1.1. However, one problem of document sampling from the original corpus is the *inefficiency*. It is because, for each task, only a small number of documents that contain the corresponding classes can be the candidate documents of the task. For example, if each document contains only a small number of entity types, the majority of documents would be rejected. To improve sampling efficiency, one strategy is to count entities in each document in advance and, for each entity type, all the candidate documents that contain this type are temporally stored in a new dataset. We only look at the task-specific candidate datasets  $\mathcal{D}^{\mathcal{E}} = \{\mathcal{D}^e | \forall e \in \mathcal{E}\}$ , where  $\mathcal{D}^e = \{(\mathbf{X}, \mathbf{Y}) | \forall (\mathbf{X}, \mathbf{Y}) \in \mathcal{D} \text{ if } e \in \mathbf{Y}\}$ . We randomly sample  $M_s$  documents such that the total number of entity instances is satisfied—that is,  $K \sim \rho K$  shots per entity type. Likewise, we sample  $M_q$  documents for  $Q$ , such that there are  $K_q \sim \rho K_q$  shots per entity type. We keep track a table to record the current count of occurrences of each type of entity types in the task. (4) *Label Conversion*. In the few-shot setting, the majority region of an document does not follow the *in-task distribution* (ITD) of  $\mathcal{E}$ . These regions’ tokens are treated as either background or

the other types of entities from the *out-of-task distribution* (OTD), whose original labels should be arbitrarily converted into 0 label. In addition, we map the original labels of ITD tokens to relative labels. For example, if we use I/O schema, the relative labels should range from label id 0 to label id  $(U - 1)$ .

## 4.5.2 Setups and Evaluation Metrics

We compare the proposed framework with aforementioned meta-learning baselines. Data generation and methods are implemented using JAX and Tensorflow. All experiments ran on 32 TPU devices. We use the Adam optimizer to update the meta-parameters. For gradient based methods, we use vanilla SGD for the inner-loop optimization and fix 15 SGD updates with a constant learning rate of 0.015.

**LLM-based Multimodal Encoder:** We pre-train the multimodal Transformer on the IIT-CDIP dataset [177]. It should be noting that this paper does not focus on the pre-training technique. In fact, our framework does not require a well pre-trained encoder, since the meta-learning will further meta-tune the pre-trained encoder to capture the domain knowledge of  $P(\mathcal{D})$ . Thus, we stop the pre-training until an 81.5% token classification accuracy.

**Training Parallelism:** We employ the episodic training pipeline to learn the meta-parameters from training tasks (i.e., episodes). At each meta-training step, a total of  $\tau$  episodes are trained and then validated to obtain the meta-gradients used for updating meta-parameters. Both meta-training and meta-testing were run in a multi-process manner. Each of our experiments was run on a total of 4 machines and on each machine there are 8 local TPU devices. Since the parameter size of the Transformer-based encoder is large, we use the 8 devices of each machine to train one single episode in parallel. That is saying, at each meta-training step, a total of 4 tasks are used to compute the meta-gradients. Both the support (train) and query (test) documents in one task are divided and assigned to 8 devices. The prototypes, the nearest neighbors of data points, or the adapted parameters trained on the local support set, are computed on each local device. For validation on the query set, however, we should consider, the scope of the entire task over



different local devices. Therefore, we employ Federated Learning techniques [183, 184, 73, 181] operating on multiple devices for a distributed within-task adaptation, where we collect the locally adapted parameters (at each inner-loop step) or the prototypes from the 8 devices of a single episode and average their parameters. Specifically, for training parallelism of each episode/task, there are 4 steps: (1) on each device, we first adapt a model based on the partial support documents located on the device; (2) then, we collect the adapted knowledge from each of the 8 local devices and aggregate them; (3) on each device, we apply the collected adapted knowledge to the partial query documents; (4) the validation loss on the query subset on each devices are collected and we take an average of them.

**Evaluation Metrics:** We consider two types of quantitative metrics. **(1) Overall Performance:** following [160], we use the precision (**P**), recall (**R**), and micro **F1**-score over meta-testing tasks to measure the accuracy of entity retrieval. **(2) Task Specificity (TS):** to evaluate how well the trained meta-learners can distinguish in-task distribution (ITD) from out-of-task distribution (OTD) for any novel given task, we plot ROC curves and calculate **AUROC** [185] using the ITD scores over meta-testing tasks. A random guessing detector outputs an AUROC of 0.5. A higher AUROC indicates better TS performance.

**Visualization:** To visualize the TS, we plot the **ROC curves** of all the meta-testing tasks, where each curve represent one task. Another visualization for TS is to show how **ITD and OOD** are distinguished against each other. We randomly select a testing task and exploit **tSNE** [186] to visualize the learned embeddings of all the tokens in the task, where ITD tokens are denoted as red points and OTD tokens are blue points. Furthermore, we use tSNE to visualize the learned embeddings of only the **ITD** token instances in the task, where different colors represent different entity types.

### 4.5.3 Main Results

Table 4.5.2 reports the results on FewVEX. Under the same  $U$  and  $K$  setups, traditional meta-learning methods fail to balance the precision and recall performances:

Table 4.5.2: Performance on 4-way 1-shot, 4-way 4-shot, 5-way 2-shot FewVEX.

Methods	4-way 1-shot				4-way 4-shot				5-way 2-shot			
	Overall			TS	Overall			TS	Overall			TS
	P	R	F1	AUROC	P	R	F1	AUROC	P	R	F1	AUROC
ProtoNet	0.02	0.10	0.03	N/A	0.02	0.09	0.03	N/A	0.02	0.09	0.03	N/A
ProtoNet+EOD	0.13	0.47	0.21	N/A	0.11	0.58	0.23	N/A	0.11	0.35	0.17	N/A
<b>ContrastProtoNet</b>	0.54	0.43	<b>0.47</b>	<b>0.59</b>	0.61	0.59	<b>0.60</b>	<b>0.89</b>	0.49	0.41	<b>0.44</b>	<b>0.62</b>
Reptile	0.48	0.10	0.15	0.58	0.62	0.44	0.51	0.67	0.39	0.09	0.14	0.59
ANIL	0.39	0.19	0.25	0.56	0.54	0.44	0.50	0.87	0.35	0.13	0.19	0.61
<b>Reptile+HC</b>	0.35	0.13	0.20	<u>0.63</u>	0.63	0.65	<b>0.64</b>	<b>0.98</b>	0.34	0.12	0.18	<u>0.65</u>
<b>ANIL+HC</b>	0.40	0.58	<b>0.50</b>	<b>0.95</b>	0.47	0.59	<u>0.51</u>	<u>0.98</u>	0.38	0.56	<b>0.46</b>	<b>0.92</b>

ANIL and Reptile using vanilla decoders achieved high precision but tended to perform low recall; the vanilla Prototypical Networks tended to be opposite: low precision but high recall. In contrast, ANIL+HC, Reptile+HC and ContrastProtoNet, achieved better precision-recall balance and thus higher F1 scores and TS, proving that detecting and alleviating the influence of out-of-task distribution can improve task personalization and accuracy. Such phenomenon is also illustrated in Figure 4.5.1 and Figure 4.5.2, where we plot ROC curves and tSNE visualizations of token embeddings after task adaptation. Comparing our methods against baselines, we observe an elevation in the curves and more distinct boundaries between OTD and ITD and between ITD classes.

The reasons are as follows. *First*, ANIL and Reptile treat the dominant OTD instances as an extra class as well. The problem turns out the imbalanced classification in meta-learning, one of the challenges in few-shot VDER tasks. By using an OTD detector, ANIL+HC and Reptile+HC can faster adapt to the task-specific boundary between OTD and ITD. Overall, this potentially increase the recall and task specificity score and the overall F1 score. *Second*, for the vanilla metric-based methods, where OTD instances are treated as one extra class, the ITD testing instances tend to be close to ITD class centers so that we have high recall. However, OTD instances dominate the task. It is possible that some OTD testing instances are closer to ITD centers than the OTD class center (the average center of multiple OTD classes) so that most of them are misclassified as one of ITD classes, i.e., low precision. In opposite, ContrastProtoNet does not make any assumption on the OTD distribution; instead, we enforce OTD to be far away from ITD classes and

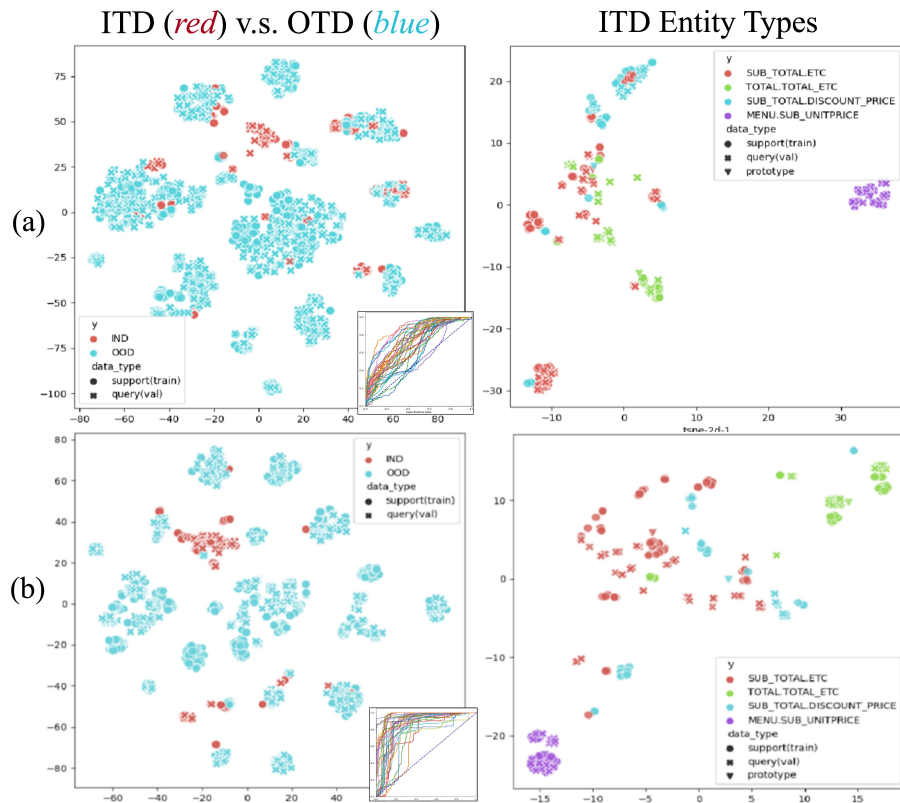
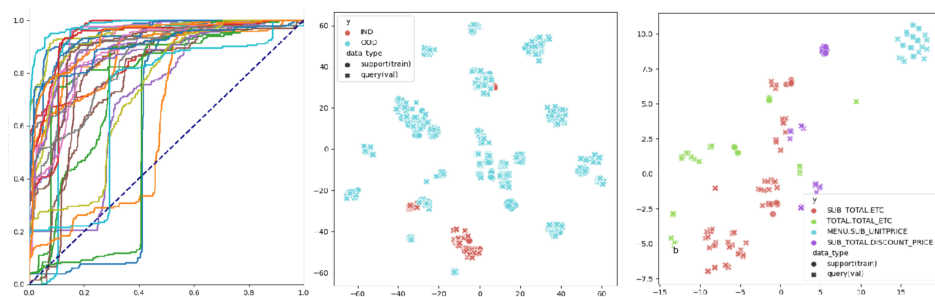


Figure 4.5.1: tSNE visualization of the learned embedding space for a randomly-selected meta-testing task, comparing (a) vanilla ProtoNet and (b) ContrastProtoNet methods, under the 4-way 4-shot setting of FewVEX.

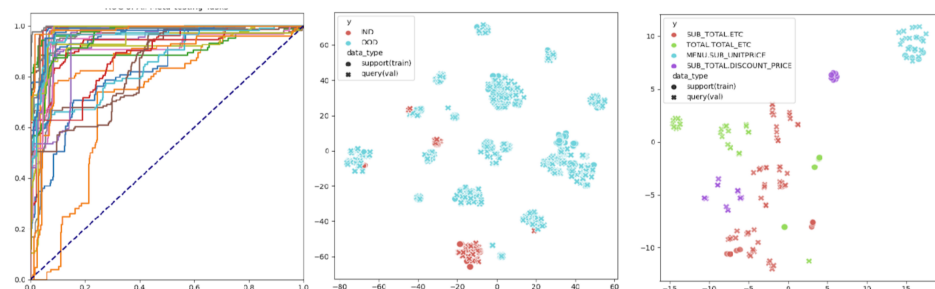
classify via token-level similarities while considering probabilistic uncertainty.

#### 4.5.4 Class Structure Disentanglement

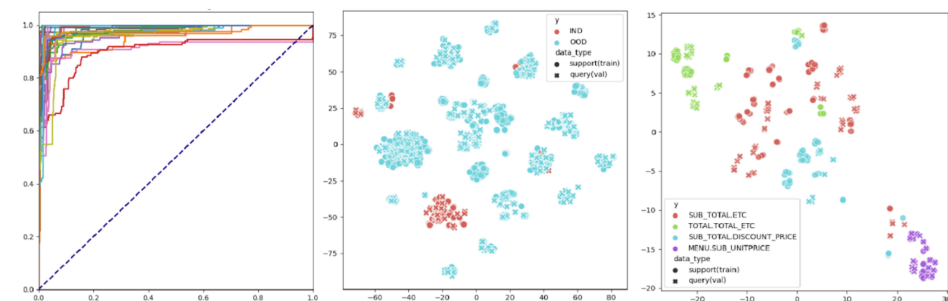
We examine the explainability and disentanglement of the learned representations (generated by the meta-parameters of encoder). Figure 4.5.1 shows tSNE visualizations of the learned embedding space of a selected task. Overall, by comparing Figure 4.5.1 to Table 4.5.2, the higher performance appears to be consistent with more disentangled clusters. Moreover, from the first column containing ITD (*red*) tokens and OTD (*blue*) tokens, we observe that the blue points dominate the embedding space and comprises multiple clusters, which demonstrates the out-of-task distribution is multimodal, making it hard to identify in-task entities. Further, we try to understand the disentangled structure of classes from the clusters. In the right col-



(a) ANIL (4-way 1-shot)



(b) ANIL + HC (4-way 1-shot)



(c) ANIL + HC (4-way 4-shot)

Figure 4.5.2: Visualization under 4-way 4-shot and 4-way 1-shot settings of FewVEX, for ANIL and ANIL+HC.

umn in Figure 4.5.1, we zoom into the four ITD classes, where the 4 entity types are represented by different colors: “menu (sub\_uniprice)” (violet), “sub\_total (etc)” (red), “total (total\_etc)” (green), and “sub\_total (discount\_price)” (blue). We observe that “menu (sub\_uniprice)” (violet) is far away from the other three classes, while the other three classes are slightly entangled. Such class structure represents the relationships between these entity types, which is explainable: the red and blue classes belong to the same superclass `sub_total`; the green and red are both `etc` information.

## 4.6 Conclusions

In this chapter, we studied the entity-personalized multimodal learning problem for VDER (EpVDER) as a case study for CoMML with agent individual concept preferences. We exploited both metric-based and gradient-based meta-learning paradigms, along with a new technique we proposed to enhance task personalization via out-of-task-distribution awareness. The experiments showed that the proposed methods achieve major improvements over the baselines for EpVDER. For future works, our approaches might be improved in the following directions: (1) A more robust algorithm that distinguishes between the OTD and ITD. (2) An advanced decoding process considering graphical structures or implicit correlations between entity instance within each task. (3) Exploring the causal role of pretrained models.

# Chapter 5

## Meta Graph Learning for Handling Coexistence of Modality Gap and Concept Shift

### 5.1 Introduction

In this chapter, we focus on scenarios where individual MML agents are permitted to have both **modality preferences** and **concept preferences**. In such cases, there exist simultaneous modality gaps and concept shifts among agents. While Chapter 3 and Chapter 4 have addressed each of the two personalization patterns, respectively, the **coexistence of modality gaps and concept shifts** introduces additional complexity to the problem: the personally-defined concepts can be represented by various combinations of modalities combining, which further exacerbates the information gap when agents exchange shareable information. Simply combining their solutions may not straightforwardly resolve this challenge.

A real-world example of such a coexistence setting can be found in *Hybrid Few-shot Learning* (hFSL) problem. The hFSL problem, distinguished from standard few-shot learning [39, 187], takes into account that personally-defined concepts can be represented by various combinations of modalities. That is, the few-shot examples of the personal concepts in an agent can be further divided and distributed across separate feature spaces. Figure 5.1.1(b) shows two hybrid 5-way 1-shot clas-

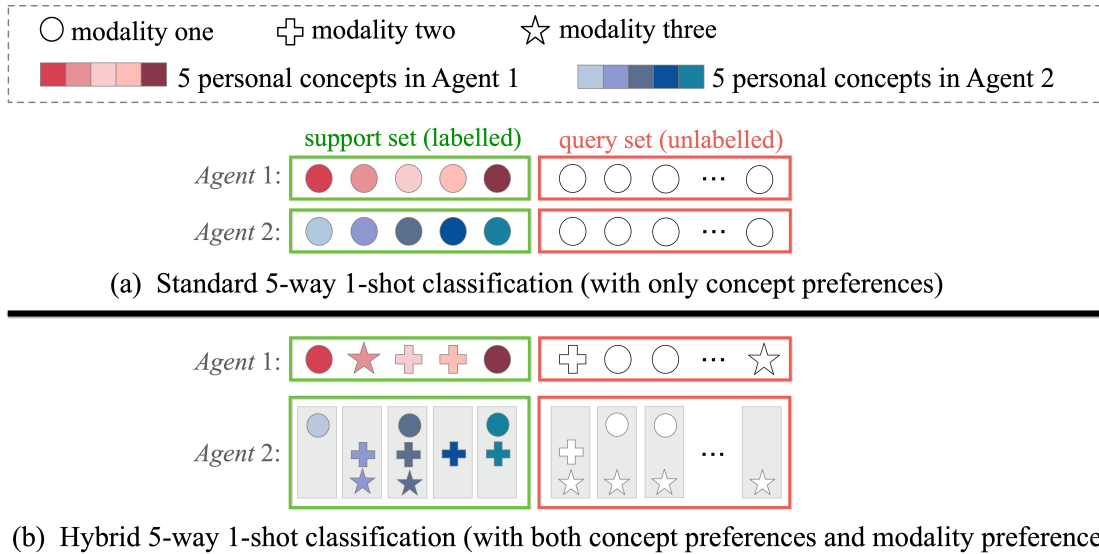


Figure 5.1.1: Comparison between uniform and hybrid FSL. Colored shapes within green rectangles denote training samples. In each task, different colors indicate the different classes (concepts). Hollow shapes within red rectangles denote unlabeled testing points. Note that in case-2, each of the grey regions denotes a data sample, where a sample may contain more than one modality.

sification tasks, which can be regarded as two participating agents, where samples may be diverged from each other in terms of their feature spaces. Therefore, in this chapter, we aim to study and experiment on this specific use case—**hFSL via multi-agent collaboration**, to delve into the research on “CoMML with both modality preferences and concept preferences”.

The coexistence of modality gaps and concepts shifts in Multi-agent Collaborative hFSL leads to two technical challenges. (1) First, compared with uniform FSL, the **data scarcity** problem would be escalated in hFSL. Specifically, since the few labeled samples per class may be spread in different feature spaces, in *each space*, there would be less labeled data per class (i.e., *less shots*) or even no training data available in some classes (i.e., *zero shot*). That is, the number of training samples in each space may be reduced. For example, consider the task shown in Figure 5.1.1(b) agent-1, there is no modality-one training data in class-2, thus the agent-2 for modality-one is a zero-shot case. (2) Second, for a hybrid  $K$ -shot classification task, the *uneven* split of  $K$  examples per class would result in a hybrid number of labeled samples per class (i.e., **hybrid shots**) in *each space*. Typically, a model

is designed for training and testing data that has the same input space. However, the decreased and hybrid number of training examples in each space may bring difficulties to the model training. Although one may consider training an alignment function to unify the training data from heterogeneous spaces, the accuracy of such a task-specific alignment function still relies on the limited number of support examples in each input space.

To alleviate the data-scarcity and hybrid-shot problems of hFSL, we propose to formulate the hybrid few-shot task as a transductive learning task, which maximally leverages available information in the task to enrich our knowledge about the target concepts, while learning the potential relationships between heterogeneous data. Transductive inference for few-shot learning typically utilizes the query samples to improve the task-specific knowledge distillation [188, 189, 190]. Inspired by this, we propose a transductive meta-learner which can incorporate some unlabeled data containing information that is not possessed in the labeled samples. Intuitively, our key idea is to jointly learn all the samples in the task with heterogeneous spaces so that the model can obtain extra information (from unlabeled data) about the relationships between spaces and the data distribution to make better predictions. In particular, we aim to learn the task-specific relationships 1) between heterogeneous input spaces and 2) between samples within the same class (intra-class samples) or belonging to different classes (inter-class samples), where the underlying data relationships within a task are complicated and hard to be learned due to data heterogeneity.

To achieve these goals, we propose **Task-adaptive Topological Transduction Network (TopoNet)**, a graph neural network-based transductive few-shot learning framework for hFSL. Basically, we introduce a topological transductive meta-learner, which can learn the task’s class distribution by simultaneously exploring relationships between concepts as well as relationships between the heterogeneous feature spaces of data. We explicitly model a graph structure to connect all the samples in a task to perform the transduction; edges expressively connect inter- and intra-class samples as well as bridge heterogeneous samples, which helps to leverage multi-space relationships and data semantic similarities. To capture both the multi-space relationships and the inter- and intra-class data relationships, we



first construct a *node-heterogeneous multi-relation graph* from the original multi-space features, and then we propose the *edge-enhanced heterogeneous graph neural network* to alternatively update edge and node features layer by layer, where heterogeneous input spaces are gradually unified while leveraging edge features to incorporate inter- and intra-class relationships. An overview of this chapter:

- In the era of few-shot learning, this chapter studies a novel hybrid few-shot learning problem, allowing a task to incorporate multiple feature spaces and contains a hybrid number of shots per class in each space.
- In the era of CoMML, this chapter explores a solution for facilitating multi-agent collaboration amidst simultaneous modality gaps and concept shifts.
- We propose to train a graph meta-learner to overcome the data-scarcity and hybrid-shot challenges in hFSL by modeling a learnable and generalizable topological structure.
- The experimental results on Collaborative hFSL simulations demonstrate that our framework is superior to existing approaches.

## 5.2 Related Work

In addition to the literature reviewed in Section 2.4, our proposed techniques are related to and inspired by three lines of work: 1) Meta-Learning for Few-shot Classification; 2) Transductive Inference; and 3) Meta-learning for Graphs.

**Meta-Learning for Few-shot Classification.** Recent meta-learning approaches can be divided into two categories: *inductive* and *transductive* few-shot classification. Inductive few-shot learning has been more widely studied than transductive few-shot learning. Inductive methods mainly includes metric-based and optimization-based algorithms. *Metric-based approaches* learn an embedding metric space shared by all tasks, on which data samples of different classes can distinguish with each other based on distance measurements [51, 52, 53, 54]. *Optimization-based approaches* train a meta-learner as an optimizer to fine-tune the meta-prior, thus adapt the class distribution to each specific task [39, 55, 56, 57]. Further, several works [52, 67, 68] improved the metric- or optimization-based methods in

terms of task adaptability. While these approaches presume the samples in a task share a uniform input space, we assume a hybrid task setting involving a mixture of different input spaces. Some recent works studied *multimodal few-shot* learning [58, 59, 191]. Although we also use multimodal few-shot datasets, we allow for the frequent occurrence and different conditions of missing modalities in real-world multimodal few labeled data scenarios.

**Transductive Inference.** Transductive learning was first introduced in [192]. A family of transductive methods were built upon graph learning frameworks, such as graph propagation [193] and graph neural networks (GNN) [194, 195]. Transductive inference has been recently used to solve few-shot tasks, which has shown substantial improvements over inductive counterparts as it utilizes unlabeled query data to obtain more representative class distribution. Based on how the model incorporates unlabeled data, existing transductive approaches can be separated into implicit and explicit methods. *Implicit transductive* methods directly use the entire unlabeled feature information to enhance the classification boundaries [57, 189, 196, 197]. While implicit methods do not leverage data relationships during transduction, *explicit transductive* methods measures the underlying relationships between data to enrich class features [198, 199, 200, 201, 188, 190] Our method follows the explicit transductive paradigm in the sense that we also explore data relationships during within-task transductive adaption. However, existing transductive methods rely on a common metric space to measure data relationships. Yet this assumption does not hold in the hybrid few-shot setting with heterogeneous input spaces. This chapter mainly deals with the difficulties from the division of samples, where the relationships between data could be more complicated and unclear.

**Meta-learning for Graphs.** Our framework utilizes Graph Neural Networks (GNNs) [194, 202] for solving hybrid few-shot tasks. Yet we focus on jointly learning the graph structure and node representations, as well as how to generalize and adapt the learnable structure over tasks. Some works [203, 204] proposed techniques for optimizing graph structures together with GNN parameters using meta-gradients, reinforcement learning, or discrete edge probabilities, but stud-

ied different problems (e.g., completing corrupted edges and adversarial attacks) on a single large-scale graph. Recent works incorporated graph structured data into meta-learning [205, 206, 207, 208]. We also formulate our problem as graph-structured semi-supervised node classification tasks plugged into meta-learning. However, these methods assume a single large-scale graph whose structure is given. In contrast, the graph structure of our task is not given, and moreover, we generalize the graph structure knowledge across unlimited graphs and adapt the graph learning procedure over different tasks.

## 5.3 Problem Formulation

### 5.3.1 Local Task Definition on Each Agent

Chapter 4 focused on Concept-shifted Collaboration. While there are concept shifts among agents, the input modality types remain uniform across agents (e.g., both text and modalities are fed to the input at all times). In this chapter, we consider simultaneous **modality preferences** and concept shifts. Further, while Chapter 4 focuses on the fine-grained concept preferences, this chapter will resort to a simpler version of Chapter 4—the **instance-level concept preferences**.

**Definition 5.1 (Modality- and Concept-personalized Few-shot Learning).**

An agent is defined as a  $U$ -way  $K$ -shot instance classification task  $\mathcal{D} = \{\mathcal{S}, \mathcal{Q}, \mathcal{C}, \mathcal{I}\}$ , which consists of a *support* set  $\mathcal{S}$  containing  $U \times K$  samples, a *query* set  $\mathcal{Q}$  containing  $n$  samples, a personal *label* space  $\mathcal{C}$ , and an indicator  $\mathcal{I}$  that specifies its personal and sample-unified modality set

$$\begin{aligned} \mathcal{S} &= \{(\tilde{\mathbf{x}}_1, y_1), (\tilde{\mathbf{x}}_2, y_2), \dots, (\tilde{\mathbf{x}}_{U \times K}, y_{U \times K})\} \\ \mathcal{Q} &= \{(\tilde{\mathbf{x}}_1^*, y_1^*), (\tilde{\mathbf{x}}_2^*, y_2^*) \dots, (\tilde{\mathbf{x}}_n^*, y_n^*)\} \\ \mathcal{C} &= \{c_1, c_2, \dots, c_U\} \\ \mathcal{I} &\subseteq [M], \end{aligned} \tag{5.1}$$

where the input of each sample  $j$  is an ordered tuple  $\tilde{\mathbf{x}}_j = (\mathbf{x}_j^{(m)}) \in \mathbb{R}^{F_m} | m \in \mathcal{I}$  containing the data of modalities specified by  $\mathcal{I} \subseteq [M]$ .  $M$  is the number of all modality types across agents. The goal is to obtain a personal model  $\theta$  by

optimizing the local objective  $\min_{\theta} \mathbb{E}_{(\tilde{x}, y) \sim \mathcal{S}} l(y, f(\tilde{x}; \theta))$ , where  $l(\cdot, \cdot)$  is the loss function for each sample and  $f(\cdot; \theta)$  is the trainable forward function.

**Definition 5.2 (Modality-hybrid Concept-personalized Few-shot Learning (hFSL)).** A more general case of Definition 5.1 is that the  $(UK + n)$  samples within an agent might also have different input modality sets. This situation results in the support/query samples of a concept in the task being further separated into different feature spaces. An agent is defined as a *hybrid*  $U$ -way  $K$ -shot instance classification task  $\mathcal{D} = \{\mathcal{S}, \mathcal{Q}, \mathcal{C}\}$  which consists of a *support* set  $\mathcal{S}$  containing  $U \times K$  samples, a *query* set  $\mathcal{Q}$  containing  $n$  samples, and a personal *label* space  $\mathcal{C}$

$$\begin{aligned} \mathcal{S} &= \{(\tilde{\mathbf{x}}_1, y_1, \mathcal{I}_1), (\tilde{\mathbf{x}}_2, y_2, \mathcal{I}_2), \dots, (\tilde{\mathbf{x}}_{U \times K}, y_{U \times K}, \mathcal{I}_{U \times K})\} \\ \mathcal{Q} &= \{(\tilde{\mathbf{x}}_1^*, y_1^*, \mathcal{I}_1^*), (\tilde{\mathbf{x}}_2^*, y_2^*, \mathcal{I}_2^*), \dots, (\tilde{\mathbf{x}}_n^*, y_n^*, \mathcal{I}_n^*)\} \\ \mathcal{C} &= \{c_1, c_2, \dots, c_U\} \end{aligned} \quad (5.2)$$

The modality set indicator  $\mathcal{I}_j \subseteq [M]$  is associated with each sample  $j$ . The input of each sample  $j$  is an ordered tuple  $\tilde{\mathbf{x}}_j = (\mathbf{x}_j^{(m)}) \in \mathbb{R}^{F_m} | m \in \mathcal{I}_j$  containing the data of modalities specified by  $\mathcal{I}_j$ . The goal is to obtain a personal model  $\theta$  by optimizing the local objective  $\min_{\theta} \mathbb{E}_{(\tilde{x}, y, \mathcal{I}) \sim \mathcal{S}} l(y, f(\tilde{x}, \mathcal{I}; \theta))$ .

### 5.3.2 Multi-agent Collaborative hFSL

We will deal with the more general case in Definition 5.2. Similar to Chapter 4, we consider a task distribution  $P(\mathcal{D})$  over few-shot learning tasks. The task of any agent  $i \in [N]$  can be sampled as  $\mathcal{D}_i = \{\mathcal{S}_i, \mathcal{Q}_i, \mathcal{C}_i\} \sim P(\mathcal{D})$ . We formulate a multi-agent systems as  $\hat{\mathcal{D}}_{\text{colab}} = \{\hat{\mathcal{D}}_{\text{colab}}^{\text{trn}}, \hat{\mathcal{D}}_{\text{colab}}^{\text{tst}}\}$  consisting of  $N_{\text{trn}}$  agents that have labelled validation set and  $N_{\text{tst}}$  agents that only have a few labelled examples

$$\begin{aligned} \hat{\mathcal{D}}_{\text{colab}}^{\text{trn}} &= \{\mathcal{D}_1, \mathcal{D}_2, \dots, \mathcal{D}_{N_{\text{trn}}}\} \text{ where } \mathcal{D}_i = \{\mathcal{S}_i, \mathcal{Q}_i, \mathcal{C}_i\} \sim P(\mathcal{D}) \\ \hat{\mathcal{D}}_{\text{colab}}^{\text{tst}} &= \{\mathcal{D}_1^*, \mathcal{D}_2^*, \dots, \mathcal{D}_{N_{\text{tst}}}^*\} \text{ where } \mathcal{D}_i^* = \{\mathcal{S}_i^*, \mathcal{Q}_i^*, \mathcal{C}_i^*\} \sim P(\mathcal{D}) \end{aligned} \quad (5.3)$$

where any training agent focuses on personal concepts  $\mathcal{C}_i \subset \mathcal{C}_{\text{base}}$  from a set of base classes  $\mathcal{C}_{\text{base}}$  and any testing agent focuses on personal concepts  $\mathcal{C}_i^* \subset \mathcal{C}_{\text{novel}}$  sampled from  $\mathcal{C}_{\text{novel}}$ . The query sets of training agents are treated as validation sets,  $\mathcal{Q}_i =$

$\{(\tilde{\mathbf{x}}_{ij}^*, y_{ij}^*, \mathcal{I}_{ij})\}_{j=1}^n$  for  $\forall \mathcal{D}_i \in \widehat{\mathcal{D}}_{\text{colab}}^{\text{trn}}$ . The query sets of testing agents are unlabelled testing data, that is,  $Q_i^* = \{\tilde{\mathbf{x}}_{ij}^*, \mathcal{I}_{ij}^*\}_{j=1}^n$ ,  $\forall \mathcal{D}_i^* \in \widehat{\mathcal{D}}_{\text{colab}}^{\text{tst}}$ . We define the **global objective** following [39]. We aim to train a meta-learner consisting of the global generalization parameters  $\theta$  and a personalization-aware adaptor  $\text{Adapt}(\cdot, \cdot; \phi)$

$$\theta^*, \phi^* = \arg \min_{\theta, \phi} \frac{1}{N} \sum_{i=1}^{N_{\text{trn}}} \mathcal{L}_i(\text{Adapt}(\mathcal{D}_i, \theta; \phi)) \quad (5.4)$$

such that any seen or unseen agent  $\mathcal{D}_i$  can take advantage of it to quickly obtain a good **personal** model  $\theta_i^* = \text{Adapt}(\mathcal{D}_i, \theta^*; \phi^*)$ , where  $\mathcal{L}_i$  is the local loss on query set and  $\text{Adapt}(\cdot, \cdot; \phi)$  is the local training result from the support set.

## 5.4 TopoNet: Graph Transductive Meta-learner

In a hybrid few-shot classification task, as defined in Eq.(5.2), data are heterogeneous in terms of the inconsistent feature spaces of data. That is, the limited labeled samples per class (i.e.,  $K$  shots) can be further partitioned by the different feature spaces. Therefore, each space  $u$  only contains partial labeled samples for each class, which leads to two subproblems: 1) the **data-scarcity** problem is aggravated such that the number of training samples per class in each space is reduced to less shots or zero shot; 2) the **hybrid-shot** problem, where different classes have different number of training samples in each space  $u$ , as the  $K$  examples per class have been *unevenly* split.

To overcome these challenges, we propose to employ the transductive inference for task adaption. We aim to train a *transductive meta-learner* that jointly considers the knowledge about heterogeneous data in both  $\mathcal{S}$  and  $\mathcal{Q}$ :

$$p_{\theta}(y^* | x^*, \mathcal{S}, \mathcal{Q} \setminus \mathcal{Y}_Q), \quad (5.5)$$

where  $\mathcal{Y}_Q$  denotes the ground-truth labels of query samples in  $\mathcal{D}$ , which means the labels of query set are not required for solving each task, which is the truth in reality. An assumption underlying Eq.(5.5) is that we know partial testing (query) samples for solving a task. This assumption yet holds in meta-learning framework

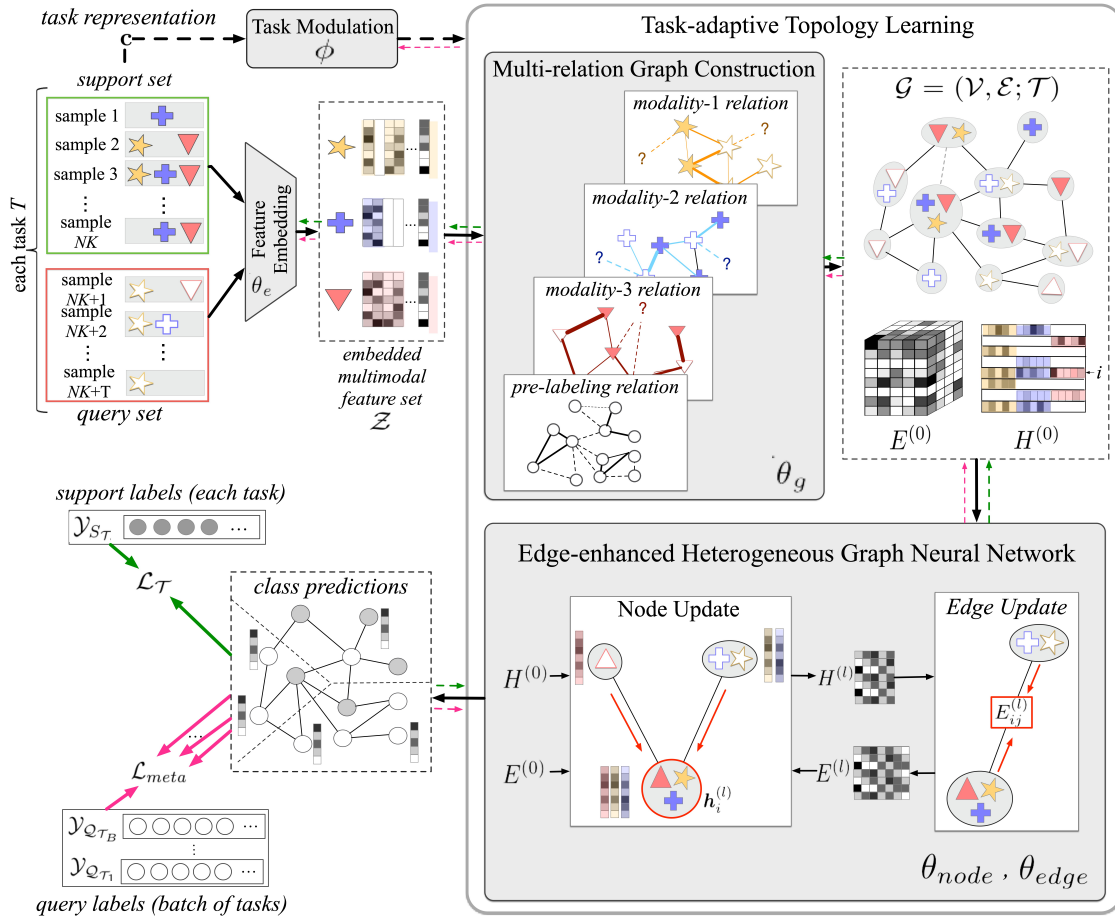


Figure 5.4.1: The proposed TopoNet framework.

as  $\widehat{\mathcal{D}}_{\text{colab}}^{\text{train}}$  contains the query data of each task to enable the training of meta-learner.

Intuitively, during the transductive task adaptation, we incorporate unlabeled samples and jointly learn all the samples in the task with heterogeneous spaces, so that the meta-learner can obtain extra information about the task-specific data distribution and the relationships between spaces to make better predictions. In this section, we will introduce the proposed the Task-adaptive Topological Transduction Network (TopoNet), whose overview is in Figure 5.4.1.

### 5.4.1 Feature Embedding

A feature embedding network  $f_{\text{enc}}(\cdot; \theta_{\text{enc}})$  is used to extract features of an input  $\tilde{\mathbf{x}}_j$ , where  $\theta_{\text{enc}}$  indicates its parameters. Suppose there are  $M$  modalities,  $f_{\text{enc}}$  contains  $M$  paralleled modality-specific subnetworks  $f_{\text{enc}}^{(1)}, f_{\text{enc}}^{(2)}, \dots, f_{\text{enc}}^{(M)}$ . Each *existing* (not

missing) modality  $\mathbf{x}_j^{(m)}$  of a sample  $\tilde{\mathbf{x}}_j$  is embedded independently through the subnetwork  $f_{\text{enc}}^{(m)}(\cdot; \boldsymbol{\theta}_{\text{enc}}^m)$ ,

$$\mathbf{z}_j^{(m)} = f_{\text{enc}}^{(m)}(\mathbf{x}_j^{(m)}; \boldsymbol{\theta}_{\text{enc}}^m) \in \mathbb{R}^F, \quad (5.6)$$

where  $F$  is the dimension of each modality’s embedding. Hence, each sample  $\tilde{\mathbf{x}}_j$  is embedded as a tuple containing  $|\mathcal{I}_j|$  modality-specific embeddings,  $\mathbf{z}_j = (\mathbf{z}_j^{(m)} | m \in \mathcal{I}_j)$ . With the transductive inference that jointly learns support and query data in task  $\mathcal{D}$ , we will obtain an *embedded feature set* for all support and query samples within the task, i.e.,  $\mathcal{Z} = \{\mathbf{z}_j | \forall \tilde{\mathbf{x}}_j \in \mathcal{S} \cup \mathcal{Q}\}$ . Note that for uniform multimodal FSL,  $f_{\text{enc}}$  will generate a feature set with fixed number of embeddings per sample so that  $\mathcal{Z} = \mathbf{Z} \in \mathbb{R}^{(UK+n) \times MF}$ .

## 5.4.2 Topological Transductive Learning

To overcome the hybrid-shot and data-scarcity dilemma of hFSL, our key idea is to build connections and unify all different types of samples in a task during model training. Therefore, we consider the transductive inference (as in Eq.(5.5)) that can jointly learn support and query samples from multiple input spaces. In this transductive framework, we focus on solving two subproblems: 1) how to explore the relationships between multiple input spaces so that samples can be aligned in a uniform semantic space; 2) how to discover the inter- and intra-class data relationships and then utilize them to improve the representativeness of the learned class distribution.

To facilitate the exploration of data and multi-space relationships within the transductive learning framework, we propose to explicitly model a learnable graph structure to connect all the samples in a task. We consider the input set of a task is believed to have some geometric structure, and the edges (topology) of a graph structure can naturally connect different input spaces, as well as leverage the potential inter- and intra-class data relations of the task. Therefore, given a task  $\mathcal{D}$ , our goal is to learn its *underlying topological graph*  $\mathcal{G} = (\mathcal{V}, \mathcal{E}; \mathcal{D})$ , which represents the relations among the support and query samples within the task.  $\mathcal{V} = \{v_j\}_{j=1}^{UK+n}$  denotes the vertex set combining support and query samples, and

$\mathcal{E} = \{e_{j,j'}\}_{i,j=1}^{UK+n}$  is the edge set which connects each pair of samples from different classes and different input spaces. Each node  $v_j$  is associated with a node feature  $\mathbf{h}_j$ , and each edge  $e_{j,j'}$  is also associated with an edge feature/weight  $\mathbf{e}_{j,j'}$  which is relevant to node relationships.

Solving an hFSL task can be viewed as learning the node and edge features of graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E}; \mathcal{D})$ . We formulate such graph learning task as a *semi-supervised node classification task*, supervised by the  $|\mathcal{S}|$  labeled nodes. In this section, we will first construct a multi-relation graph with its initialized node and edge features, and then, edge and node features are refined step-by-step via an edge-enhanced heterogeneous graph neural network.

#### 5.4.2.1 Graph Construction with Multi-space Nodes

From the multi-space feature set  $\mathcal{Z}$  produced by the feature embedding network, we can construct a graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E}; \mathcal{D})$  with initial node features  $\mathbf{H}^{(0)}$  and initial edge features  $\mathbf{E}^{(0)}$ .

The initial feature of each node  $v_j$  is the concatenation of available modalities of the sample, i.e.,  $\mathbf{h}_j^{(0)} = \|\|_{m \in \mathcal{I}_j} \mathbf{z}_j^{(m)}$ , where  $\|\|$  denotes concatenation. The initial node feature set  $\mathbf{H}^{(0)} = \{\mathbf{h}_j^{(0)} \in \mathbb{R}^{|\mathcal{I}_j|F}\}_{j=1}^{UK+n}$  is **heterogeneous** as different nodes (samples) have different combinations of modalities. Note that if two nodes  $\mathbf{h}_j^{(0)}$  and  $\mathbf{h}_k^{(0)}$  with  $\mathcal{I}_j = \{1, 2\}$  and  $\mathcal{I}_k = \{2, 3\}$ , although both are  $2F$ -dimensional (i.e.,  $|\mathcal{I}_j| = |\mathcal{I}_k| = 2$ ), they still belong to different feature spaces.

Edge features leverage data relationships. However, it is unfeasible to directly measure the similarity between a pair of heterogeneous nodes; also, some pairs of nodes may not contain common modalities, such as node  $\mathcal{I}_j = \{1\}$  and node  $\mathcal{I}_k = \{2, 3\}$ , but belong to the same class and should be connected. Considering these difficulties, we initialize an **multi-relation graph** where each edge measures multiple views of node relationships: 1) each modality- $m$  can provide a view of node relations by comparing the  $m$ th modality (if available); 2) the given labels of support samples can provide an additional view of class similarities. We obtain an edge-feature tensor  $\mathbf{E}^{(0)} \in \mathbb{R}^{(UK+n) \times (UK+n) \times (M+1)}$ . Each  $(i, j, m)$ -entry of  $\mathbf{E}^{(0)}$



is calculated as

$$\begin{aligned} \mathbf{E}_{i,j,m \leq M}^{(0)} &= \begin{cases} \sigma(g^m(\Delta_{j,j'}^m; \boldsymbol{\theta}_{\text{geo}}^m)) & \text{if } m \in \mathcal{I}_j \cap \mathcal{I}_{j'} \\ 0.5 & \text{if } m \notin \mathcal{I}_j \cap \mathcal{I}_{j'}, \end{cases} \\ \mathbf{E}_{i,j,M+1}^{(0)} &= \begin{cases} 1 & \text{if } y_j = y_{j'} \text{ and } v_j, v_{j'} \in \mathcal{S} \\ 0 & \text{if } y_j \neq y_{j'} \text{ and } v_j, v_{j'} \in \mathcal{S} \\ 0.5 & \text{if } v_j \in \mathcal{Q} \text{ or } v_{j'} \in \mathcal{Q}, \end{cases} \end{aligned} \quad (5.7)$$

where  $\Delta_{j,j'}^m = |\mathbf{z}_j^{(m)} - \mathbf{z}_{j'}^{(m)}|$ ;  $g^m(\cdot; \boldsymbol{\theta}_{\text{geo}}^m)$  is the metric function for modality- $m$ , a stacked Multilayer Perceptron network with parameter  $\boldsymbol{\theta}_{\text{geo}}^m$ ; and  $\sigma$  is sigmoid function. The edge feature (relationship) between a pair of nodes is an  $(M + 1)$ -dimensional vector, constructed by measuring each view’s similarity scores. Note that for some pairs of samples without common views but belong to similar classes, they should have high similarity scores in some missing views but the missing views’ similarity scores cannot be calculated; we use 0.5 to account for these uncertain views.

#### 5.4.2.2 Edge-enhanced Heterogeneous Graph Neural Network

An hFSL classification task has been converted into a *node-heterogeneous and multi-relation graph*  $\mathcal{G} = (\mathcal{V}, \mathcal{E}; \mathcal{D})$ . In the proposed topological transduction framework, solving an hFSL classification task can be formulated as the semi-supervised node classification task on  $\mathcal{G}$ , supervised by the training nodes  $\mathcal{S}$ . Yet the difficulty here is the complexity of learning a graph with several types of nodes and multi-view node connections. Therefore, we employ the power of Graph Neural Networks (GNNs) to facilitate transductive learning on  $\mathcal{G}$ .

Given the initial heterogeneous node features  $\mathbf{H}^{(0)}$  and multi-relation edge features  $\mathbf{E}^{(0)}$ , the edge and node features are updated iteratively layer by layer through a stacked edge-enhanced heterogeneous graph neural network (EHGNN):

$$\begin{aligned} \mathbf{H}^{(l)} &= g_{\text{node}}^l(\mathbf{H}^{(l-1)}, \mathbf{E}^{(l-1)}; \boldsymbol{\theta}_{\text{node}}^l) \\ \mathbf{E}^{(l)} &= g_{\text{edge}}^l(\mathbf{H}^{(l)}, \mathbf{E}^{(l-1)}; \boldsymbol{\theta}_{\text{edge}}^l), \end{aligned} \quad (5.8)$$

where  $\boldsymbol{\theta}_{\text{node}}^l$  and  $\boldsymbol{\theta}_{\text{edge}}^l$  are the node and edge updating parameters at layer  $l$ , respec-

tively. Basically, nodes from multiple spaces are aligned into a unified semantic space along the procedure; edge features are directly encoded in the node updating model so that multi-view similarity scores can be incorporated to improve node representativeness.

**Heterogeneous Space Alignment via Node Update.** At the first layer, we are give the initial heterogeneous node features  $\mathbf{H}^{(0)}$ . Each node feature  $\mathbf{h}_j^{(1)}$  is updated by aggregating its one-hop neighborhoods from each feature space

$$\mathbf{h}_j^{(1)} = \parallel_{r=1}^{M+1} \sigma \left[ \left( \sum_{u \in \mathcal{U}_{\mathcal{D}}} \sum_{j' \in \mathcal{N}(j,u)} \widehat{\mathbf{E}}_{j,j',r}^{(0)} \mathbf{W}_{r,u}^{(1)} \mathbf{h}_j^{(0)} \right) \right], \quad (5.9)$$

where  $\parallel$  is concatenation operation,  $\mathcal{U}_{\mathcal{D}} \subseteq [2^M + 1]$  denotes a set of input spaces in  $\mathcal{D}$ , and  $\mathcal{N}(i, u)$  denotes a set of neighboring nodes for  $v_j$  on the input-space  $u$ .  $\mathbf{W}^{(1)} = \{\mathbf{W}_{r,u}^{(1)} \in \mathbb{R}^{F_1 \times F_u} \mid r \leq M + 1, u \leq \mathcal{U}_{\mathcal{D}}\}$  are parameters of node encoders for nodes in each feature space and each view of relationship, where  $F_u$  is the dimension of feature space  $u$ , and  $F_1$  is the dimension of node encoders' outputs. Edge features are incorporated into the neighborhood aggregation, where each view of the multi-relation edge features generates a new node feature which is then concatenated with other views' new features. To avoid increasing the scale of output features by multiplication, we normalized edge features over the neighborhood of  $v_j$ , that is,  $\widehat{\mathbf{E}}_{ijr}^{(0)} = \frac{\mathbf{E}_{ijr}^{(0)}}{\sum_{k \in \mathcal{N}(i)} \mathbf{E}_{ikr}^{(0)}}$ . Then, at layers  $l > 1$ , we simplify the aggregation process for training efficiency as node features are early homogenized in an  $F_1(M + 1)$ -dimensional space. Given features obtained in the last layer  $\mathbf{H}^{(l-1)} \in \mathbb{R}^{(UK+n) \times F_{l-1}}$  and  $\mathbf{E}^{(l-1)} \in \mathbb{R}^{(UK+n) \times (UK+n)}$ ,

$$\mathbf{h}_j^{(l)} = \sigma \left[ \sum_{j' \in \mathcal{N}(j)} \widehat{\mathbf{E}}_{j,j'}^{(l-1)} \mathbf{W}^{(l)} \mathbf{h}_j^{(l-1)} \right], \quad (5.10)$$

where  $\widehat{\mathbf{E}}_{j,j'}^{(l-1)} = \frac{\mathbf{E}_{j,j'}^{(l-1)}}{\sum_{k \in \mathcal{N}(j)} \mathbf{E}_{jk}^{(l-1)}}$ , and  $\mathbf{W}^{(l)} \in \mathbb{R}^{F_l \times F_{l-1}}$  denotes the layer- $l$  node encoder shared by each sample.

**Edge Update.** Edge feature update is done by measuring the relationships of

current node features. The goal of edge update is to modify the previous representations for inter- and intra- class relationships, making the topological structure more relevant to the specific task. To simplify and reduce the parameter size, the dimensions of edge features after the first layer are reduced to 1. Therefore, at the first edge updating layer, the initial  $(M + 1)$ -view edge features are compressed into a single view

$$\mathbf{E}_{j,j'}^{(1)} = \frac{1}{M + 1} \sum_{r=1}^{M+1} \alpha_{j,j',r}^1 \mathbf{E}_{j,j',r}^{(0)}, \quad (5.11)$$

where  $\alpha_{j,j',r}^1 = g_{\text{edge},r}^l(\mathbf{h}_j^{(1)}, \mathbf{h}_{j'}^{(1)}; \boldsymbol{\theta}_{\text{edge},r}^1)$  is a scalar that measures the relationship between  $\mathbf{h}_j^{(1)}$  and  $\mathbf{h}_{j'}^{(1)}$ , which can be calculated using any metric or attention function (e.g., additive attention, dot-product, multiplicative attention) [18]. Then, at layers  $l > 1$ , to simplify the calculation, edge features are updated directly using the attention scores over current node features,

$$\mathbf{E}_{j,j'}^{(l)} = g_{\text{edge}}^l(\mathbf{h}_j^{(l)}, \mathbf{h}_{j'}^{(l)}; \boldsymbol{\theta}_{\text{edge}}^l). \quad (5.12)$$

To summarize, the information aggregation through edges takes into account the current edge features, thus automatically leveraging the current learned inter- and intra-class relationships and achieving. The information exchange among support and query samples jointly models different types spaces, where each space could incorporate extra information from other spaces. This process implicitly achieves multi-space alignment so that could alleviate the hybrid-shot and data-scarcity challenges.

### 5.4.2.3 Task Modulation

Meta learning explores the transferable knowledge across tasks. In TopoNet, we aim to generalize the underlying topological structure over different hFSL tasks, including the multi-space alignment parameters and the parameters used in modeling intra- and inter-class data relationships. Despite the globally shared structural knowledge, there is also specific knowledge about underlying topological structure for each task. For example, the importance of each modality may vary between different tasks. Therefore, following [52], we build a *task modulation network*  $g(\cdot; \phi)$

to condition the topological transductive learning module, which utilizes external task-level information to slightly adjust the prior knowledge for each task, may be better suited for finding correct underlying task-specific class distribution.

### 5.4.3 Optimization

**Task Objective.** In our framework, a hybrid  $U$ -way  $K$ -shot classification task is converted into a semi-supervised  $U$ -way  $K$ -shot node classification task with heterogeneous nodes. After obtaining node features  $\mathbf{H}^{(L)} \in \mathbb{R}^{(UK+n) \times F_L}$  at the last GNN layer  $L$ , we use a nonlinear classifier  $p(\cdot; \boldsymbol{\theta}_p)$  followed by a softmax layer to make class predictions for each node. The predictions are compared with ground-truth labels to calculate cross-entropy losses. The inner-loop optimization is supervised by the support labels, by minimizing the cross-entropy loss defined as follows:

$$\mathcal{L}_{\widehat{\mathcal{D}}(t)} = - \sum_{y_j \in \mathcal{Y}_{\mathcal{S}_{\widehat{\mathcal{D}}(t)}}} y_j \cdot \log(\text{softmax}(p(\mathbf{h}_j^{(L)}; \boldsymbol{\theta}_p))), \quad (5.13)$$

where  $\mathcal{Y}_{\mathcal{S}_{\widehat{\mathcal{D}}(t)}}$  denotes the  $UK$  labels in the support set  $\mathcal{S}_{\widehat{\mathcal{D}}(t)}$ . Note that the final-layer node representation  $\mathbf{h}_j^{(L)}$  of  $v_j$  has aggregated the data information from both  $\mathcal{S}_{\widehat{\mathcal{D}}(t)} \setminus \mathcal{Y}_{\mathcal{S}_{\widehat{\mathcal{D}}(t)}}$  and  $\mathcal{Q}_{\widehat{\mathcal{D}}(t)} \setminus \mathcal{Y}_{\mathcal{Q}_{\mathcal{D}_T}}$  through GNNs. With the supervision of the support labels, the topological structure learned by the topological learning network can be relevant to true class distribution of the specific task.

**Meta-objective.** We train TopoNet following the optimization-based meta-learning paradigm [39], which solves a *bilevel optimization* problem to find a *prior*  $\boldsymbol{\theta}$  as the meta-learner’s parameters. The parameters of our three-module network is  $\boldsymbol{\theta} = \{\boldsymbol{\theta}_{\text{enc}}, \boldsymbol{\psi}, \boldsymbol{\theta}_p\}$ , where  $\boldsymbol{\psi} = \{\boldsymbol{\theta}_{\text{geo}}, \boldsymbol{\theta}_{\text{node}}, \boldsymbol{\theta}_{\text{edge}}\}$  is the topological transduction module. The meta-objective is to obtain a set of meta-initialization parameters  $\boldsymbol{\theta}_0$ , an appropriate generalization of prior knowledge for all tasks, plus the parameters of the external task-modulation meta-network  $\boldsymbol{\phi}$  [52].

**Bilevel optimization.** Formally, let  $\boldsymbol{\theta}'_t$  signify  $\boldsymbol{\theta}$  for the agent  $\widehat{\mathcal{D}}(t)$  during the inner-loop optimization, and let the initial  $\boldsymbol{\theta}'_t = \boldsymbol{\theta}_0$ . In the inner-loop adaption,

---

**Algorithm 2:** Training Procedure of TopoNet
 

---

- 1: **Requires:** Distribution of hybrid few-shot tasks  $P(\mathcal{D})$
  - 2: **Requires:** Learning rates  $\alpha, \beta$ ; GNN layer number  $L$
  - 3: Randomly initialize task network  $\theta$  and meta-network  $\phi$ .
  - 4: **for** each round  $t = 1, 2, 3, \dots, T$  **do**
  - 5:   Sample batches of agents  $\widehat{\mathcal{D}}(t) \sim P(\mathcal{D})$
  - 6:   **for** each agent  $\mathcal{D}_i \in \widehat{\mathcal{D}}(t)$  **do**
  - 7:     Obtain data  $\{\mathcal{S}_i, \mathcal{Q}_i\}$  for each task  $\mathcal{D}_i$ .
  - 8:     Initialize task network  $\theta'_t = \theta_0$ , and replace  $\phi_0$  using  $\phi_{0,t}$ .
  - 9:     Calculate embedded multi-space feature set  $\mathcal{Z}_t$ .
  - 10:     Construct graph  $\mathcal{G}_t$  and initialize  $\mathbf{H}_t^{(0)}$  and  $\mathbf{E}_t^{(0)}$ .
  - 11:     Update node and edge features via EHGNN; obtain  $\mathbf{H}_t^{(L)}$ .
  - 12:     Obtain predictions  $\mathcal{Y}_{\mathcal{S}_i}$  for support set, compute adapted internal parameters with a fixed number of steps w.r.t. the  $UK$  examples from  $\mathcal{S}_i$  as in Eq.(5.14).
  - 13:     Evaluate  $\mathcal{L}_t(f(\tilde{\mathbf{x}}; \theta'_t, \phi), y^*; \mathcal{Q}_i)$  w.r.t.  $n$  testing samples of  $\mathcal{Q}_i$ .
  - 14:   **end for**
  - 15:   Update initialization of task network  $\theta_0$  as Eq.(5.15).
  - 16:   Update meta-network  $\phi$  as Eq.(5.16).
  - 17: **end for**
  - 18: **return:**  $\theta_0$  and  $\phi$
- 

during each gradient update, we compute

$$\theta'_t \leftarrow \theta'_t - \alpha \nabla_{\theta'_t} \mathcal{L}_{\widehat{\mathcal{D}}(t)}(f(\tilde{\mathbf{x}}; \theta'_t, \phi), y; \mathcal{S}_i), \quad (5.14)$$

where  $f(\cdot)$  is the forward function and  $\mathcal{L}_{\widehat{\mathcal{D}}(t)}(\cdot; \mathcal{S}_i)$  is the loss on the support set of  $\widehat{\mathcal{D}}(t)$  as in Eq.(5.13). Separately on each agent, after a fixed number of inner-loop fine tuning steps, we obtain the personal parameters  $\theta'_j(\theta_0)$  from the meta-initialization  $\theta_0$ . Then, the outer-loop optimization updates the  $\theta_0$  and  $\phi$  using the feedback from a batch of agents:

$$\theta_0 \leftarrow \theta_0 - \beta \nabla_{\theta_0} \sum_{\widehat{\mathcal{D}}(t) \sim P(\widehat{\mathcal{D}})} \mathcal{L}_{\widehat{\mathcal{D}}(t)}(f(\tilde{\mathbf{x}}^*; \theta'_j(\theta_0), \phi), y^*; \mathcal{Q}_i) \quad (5.15)$$

$$\phi \leftarrow \phi - \beta \nabla_{\phi} \sum_{\widehat{\mathcal{D}}(t) \sim P(\widehat{\mathcal{D}})} \mathcal{L}_{\widehat{\mathcal{D}}(t)}(f(\tilde{\mathbf{x}}^*; \theta'_j(\theta_0), \phi), y^*; \mathcal{Q}_i), \quad (5.16)$$

where  $\mathcal{L}_{\widehat{\mathcal{D}}(t)}(\cdot; \mathcal{Q}_i)$  is the loss on the query set of task  $\widehat{\mathcal{D}}(t)$ . Algorithm 2 shows the

overall training workflow of TopoNet.

## 5.5 Experiments

### 5.5.1 Dataset

Two source datasets are used to simulate the CoMML scenario studied in this chapter, including the **CUB-200 (image+text)** [209] originated from CUB-200, where each image is annotated with a 312-dimensional text (attribute) modality, and the 3D-object recognition dataset **miniModel40 (view1+view2)** constructed from the ModelNet40 [146], which contains 3D CAD objects covering 40 common categories (split as  $|\mathcal{C}_{train}| = 25$ ,  $|\mathcal{C}_{test}| = 9$ , and  $|\mathcal{C}_{val}| = 6$ ) and each object is marked by two views of feature representations as in [141].

From the two dataset, we construct the scenarios where concepts are scarcely labeled and data is heterogeneous, we constructed *two* hybrid few-shot classification datasets, as hFSL was never studied before and we cannot find existing datasets available. The two simulations are named **h-CUB-200** and **h-miniModel40**. Each contains a hybrid combination of modalities. Specifically, in order to simulate the irregular and frequent occurrence of missing modality in the real-world web applications, each uniform task in the source dataset was turned into the hybrid task by randomly deleting modalities from randomly picked samples. The deletion process is as follows. For each task, we first union the support and query set, and shuffle the instances. Then, we separate the combined set, which contains  $(UK + n)$  instances, into  $2^M - 1$  disjoint subsets (groups): given the hybrid ratio  $0 < \rho < 1$ , the first group has  $(1 - \rho)(UK + n)$  samples, and the other groups has  $\rho(UK + n)/(2^M - 2)$  samples. Each group except the first one is a proper subset of  $\{1, \dots, M\}$  indicating the modality availability, and for all the samples in the same group, we remove the absent modalities from the original multimodal data. Finally, in the first group, we picked  $\rho$  percentage of samples, and from each picked sample, we randomly deleted one of modalities.

## 5.5.2 Baseline Methods

We compared with three families of existing FSL approaches: 1) supervised learning approaches with inductive inference: **ProtoNet** [51], **RelationNet** [54], and **MAML** [198]; 2) semi-supervised learning approaches with transductive inference: **GNN** [198], **TPN** [210], **TransductiveTuning** [196] and **LaplacianShot** [190]; 3) while the previous two families are single-modal baselines, we also consider recent works on multimodal domain: **AM3** [58] and **MultiProtoNet** [191]. As for reproducibility of these single-modal baselines (e.g., MAML, ProtoNet, RelationNet, LaplacianShot, etc.), we imputed the missing modalities by zeros on the input before concatenating all the original/imputed modalities.

## 5.5.3 Setups

**Model Configurations:** (1) Feature embedding network contains  $M = 2$  modality-specific feature extractors. The feature extractor for CUB-200 images is ResNet-18 as in [190], followed by a fully-connected layers with output dimension  $F = 128$ ; the feature extractor for CUB-200 attributes consists of 3 fully-connected layers with hidden dimensions  $F = 128$ ; the feature extractor for *miniModel40* view1 and view2 consists of 3 fully-connected layers with hidden dimensions  $F = 128$ . (2) For graph construction, the metric function for constructing modality-wise node connections consists of two fully-connected layers with weight shapes  $F \times F/2$  and  $F/2 \times 1$ . (3) GNN layer numbers are fixed to  $L = 3$  in all experiments. Graph nodes are heterogeneous, with three node feature spaces  $F_{u1} = 128$ ,  $F_{u2} = 128$ ,  $F_{u3} = 256$ . At the first layer, nodes use different node encoders are  $64 \times 128$ ,  $64 \times 128$ , and  $64 \times 256$ . Edges are 3-dimensional. During node update, we aggregated the top- $k$  ( $k = 8$ ) neighbors instead of all the neighbors. The output dimensions of GNN node encoders are  $F_1 = 64$ ,  $F_2 = 64$ , and  $F_3 = 32$ . For edge updates, the parameter size of the attention mechanism at each layer is  $2F_l \times 1$ .

**Hyperparameters.** The number of inner-loop gradient updates are fixed to 10 steps in all experiments. The number of participants for updating the meta-learner at each round was fixed to 4 training agents.

Table 5.5.1: Average accuracy (%) of few-shot classification on testing agents.

Method	<b>h-CUB-200</b>		<b>h-miniModel40</b>	
	5-way 1-shot	5-way 5-shot	5-way 1-shot	10-way 1-shot
MAML	69.45	74.26	77.83	67.54
ProtoNet	62.44	68.5	69.30	57.10
RelationNet	73.90	78.72	80.45	67.4
GNN	67.41	72.34	73.45	62.58
TPN	71.17	76.38	79.83	66.05
TransductiveTuning	69.73	68.62	76.15	68.10
LaplacianShot	78.06	82.37	84.63	74.43
AM3-ProtoNet++	72.46	76.55	78.68	67.18
AM3-TADAM	73.15	77.28	79.54	68.72
MultiProtoNet	71.34	77.44	79.71	69.44
<b>TopoNet (Ours)</b>	<b>80.23</b>	<b>83.11</b>	<b>86.46</b>	<b>77.15</b>

### 5.5.4 Results and Analysis

**Main results.** Table 5.5.1 reports the results on the two simulations with the modality-gap level of  $\rho = 0.5$ . These results compare our method, which directly learned with the original heterogeneous data, against the baselines (designed for uniform tasks), which used zeros to impute missing modalities so that hybrid tasks were converted into uniform tasks. The effectiveness of our methods demonstrated our heterogeneous neighborhood aggregation can comprehensively utilize other samples’ information to alleviate the impact of missing information.

**Impact of Modality-gap Levels.** In Table 5.5.2 (columns 2, 3 and 4), we increase the hybrid ratio of tasks over the dataset. The larger  $\rho$  implies that more missing modalities and a larger number of agents or samples having a smaller subset of modalities. The last column shows the results when, on each agent, the value of  $\rho$  was not given but randomly sampled from a normal distribution with a mean of 0.5 and a std of 0.3, thus different tasks have different hybrid levels. As the hybrid ratio increases, the less change on our method’s performance rather than baselines demonstrates the effectiveness of our method to handle higher heterogeneity.



Table 5.5.2: Hyperparameter and ablation studies on 5-way 1-shot h-CUB-200.

Method	$\rho = 0.3$	$\rho = 0.5$	$\rho = 0.7$	dynamic $\rho$
TPN	73.29	71.33	66.23	67.42
AM3-TADAM	75.54	73.15	67.91	66.73
MultiProtoNet	73.67	71.34	64.78	69.72
LaplacianShot	80.31	78.06	72.01	75.13
<b>TopoNet<sup>‡</sup></b>	71.20	69.18	63.89	69.93
<b>TopoNet<sup>†</sup></b>	80.16	68.50	69.82	77.34
<b>TopoNet</b>	81.67	80.23	75.13	74.96

**Ablation Study.** In Table 5.5.2 (rows 6, 7, and 8), we evaluate the influence of each component in our model. TopoNet<sup>†</sup> replaces graph construction with a non-parameter metric kernel (i.e., dot-product similarity) and removes missing-view connections. TopoNet<sup>‡</sup> deletes the GNN-based node and edge updating mechanism, and replaces it with the non-parameterised Label Propagation [210] strategy. As the heterogeneity level increased, the performance of TopoNet<sup>†</sup> and TopoNet<sup>‡</sup> dropped more dramatically than TopoNet. These proved the ability of heterogeneous GNN in multi-space alignment, and the ability of the topology learning module to generalize reliable inter- and intra-class data relationships across tasks.

## 5.6 Conclusion

In this chapter, we focused on CoMML where individual MML agents are permitted to have both modality preferences and concept preferences. We delved into this by formulating a novel hybrid few-shot learning (hFSL) task and employing meta-learning to enable collaborative learning with multiple hFSL agents. We proposed a task-adaptive topological transduction network (TopoNet) to solve hFSL, which trained a heterogeneous graph-based transductive meta-learner to handle the special few-shot tasks with multiple input spaces. Our experimental results demonstrated that TopoNet successfully generalized the meta-knowledge about data and multi-space relationships over tasks, and could fast adapt to real tasks with different levels of hybrid settings.

## Part II

# Collaboration with Explicit Knowledge Transfer

# Chapter6

## Splittable and Adaptive Transfer for Fast Collaboration

### 6.1 Introduction

In this chapter, we focus on the collaboration between agents who are permitted to simultaneously possess their **modality preference**, **concept preferences**, and **domain preferences**, assuming agents are solving the same downstream task type (e.g., object classification). In such CoMML cases, the existence of **modality gap**, **concept shifts**, and **domain drifts**, jointly define the agent heterogeneity. Meanwhile, we switch our focus to a new CoMML learning paradigm. Previous chapters employing the global centralised CoMML paradigm (see Eq.(2.1)) tend to be slow and lack robustness in handling the complex personalization patterns that will be studied in the following chapters, as well as have other drawbacks such as risk of data privacy. Since the complexity of heterogeneity will increase starting from this chapter, we will switch to using the **global-local decentralized CoMML paradigm** (see Eq.(2.2)), which simultaneously learns *personal* models that frequently share information among each other. Real-world examples of such a CoMML setting can be found in *Multimodal Federated Learning* [211, 40, 41]. Federated Learning (FL) currently stands as the dominant framework for distributed training of machine learning models under communication and privacy constraints [24, 36, 26, 27, 28]. A FL setting typically involves multiple *agents* collecting

data and jointly train models without sharing their local data, inherently possessing the natural property of statistical heterogeneity among agents (i.e., agents), which includes domain drift and concept shift. Multimodal FL [211, 40, 41], as an extension of FL, additionally addresses how different modalities of data distributed over multiple agents. Multimodal FL is naturally a setting that possess both statistical heterogeneity and modality incongruity across agents. Therefore, this chapter chooses to adopt the **Multimodal Federated Learning (MFL)** setting to conduct the research on CoMML with simultaneous modality preferences, domain preferences, domain concept preferences.

In MFL systems, agents can have personal definitions of concepts and interested concepts, personal environments for collecting data, and personal setups of sensors. Therefore, they possess not only the *statistical heterogeneity* but also the *modality gap* problems. The escalated heterogeneity among agents results in more divergence between their personalized model parameters or between their gradient directions during training, thus posing challenges to collaboration and knowledge sharing across the agents. There are two technical **challenges** in the MFL. (1) The *first* challenge lies in determining how much personal knowledge can be utilized for appropriate collaboration. The greater the heterogeneity among agents, the more difficult it becomes for each of them to find shareable knowledge to communicate with peer agents. (2) The *second* challenge pertains to training stability and convergence speed. Since only a subset of agents participate in the communicative collaboration each round [24], greater heterogeneity leads to imbalanced knowledge communication and collaboration bias. Knowledge that is more prevalent among agents tends to be communicated more frequently than minority knowledge. As a result, achieving generalization for minority knowledge (e.g., less-occurring modalities or features) might be slow. Existing FMTL approaches either perform random agent selection or just select nearly all agents (tasks) to participate in each round, which is not efficient with modality gap settings. For faster convergence, it is of importance to select different types of agents in a balanced manner at each round.

Prior FL and MFL approaches mainly focus on the solutions for statistical heterogeneity [24, 26, 27, 28], but there are fewer efforts for addressing the challenges mentioned above. To tackle the aforementioned challenges, we formulate a new

fundamental structure that learn to adaptively transfer knowledge amongst different types of multimodal agent models for fast collaboration. We propose the **FedMSplit** framework. The key idea of FedMSplit is to employ a *dynamic and multi-view graph structure*, where each vertex corresponds to an agent solving the subproblem (local objective) on its local dataset, to handle the multi-view relationships among agents and hence to achieve adaptive and fast inter-agent weight sharing. In particular, we propose to split agent models into smaller blocks—some blocks are shared by all agents while some are shared amongst a subset of agents, and allow each type of blocks to provide a specific view on agent relationships. Then, given the graph representation of multimodal agents, the underlying statistical correlations between agents can be captured as the edge features in the multi-view graph, and then be used to promote local model relations through the neighborhood message passing in graph. This chapter is summarized as follows:

- In the eras of CoMML and FL, we study the less-explored field of Multimodal Federated Learning (MFL), which has broad real-world applications, paving the way for future research in this direction.
- We focus on the learning paradigm of *explicit* knowledge transfer in MFL, where collaboration occurs on the model’s parameter space or subspaces. While extensive approaches within the explicit knowledge transfer paradigm has been proposed, we focus on the less-addressed challenges on both transfer accuracy and transfer efficiency due to the modality gap among agents.
- A technical novelty in the proposed FedMSplit is that we leverage the *splittability* of model architectures to construct a *multi-view dynamic graph*, which is utilized to facilitate *adaptive and fast* knowledge transfer among clients. The graph stores real-time relationships among heterogeneous agents, enabling fast collaboration.
- Another technical novelty in the proposed FedMSplit is that we utilize the multi-armed bandit algorithm over the graph to perform efficient agent selection among different agent architectures, enabling fast collaboration.
- The empirical results show the FedMSplit’s effectiveness.

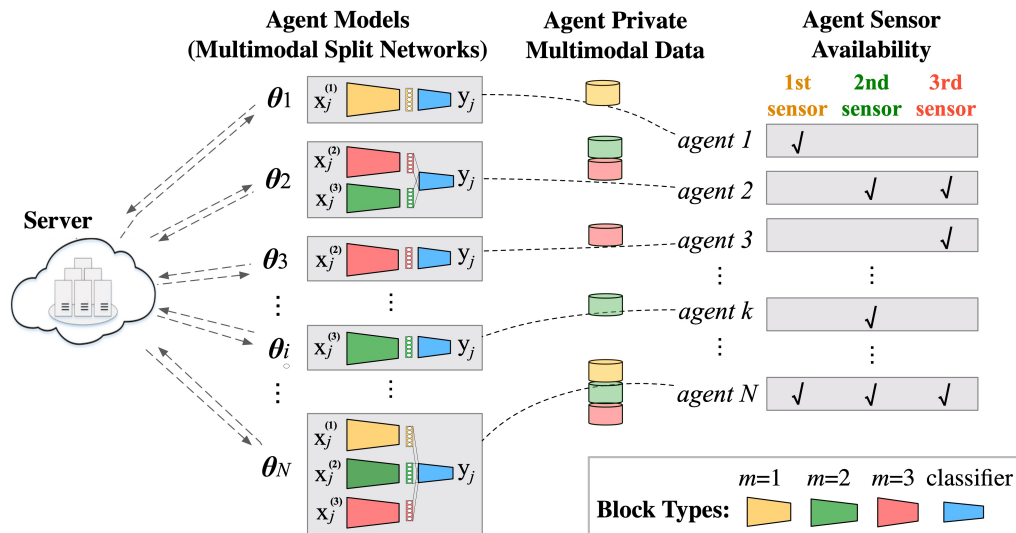


Figure 6.2.1: Multimodal federated learning (MFL) setup, where a total of  $M = 3$  types of sensors are involved. The models shown here intend for multimodal fusion tasks, yet can be replaced by other models. Any type of agent models can be split into at most four different blocks.

## 6.2 Problem Formulation

Assuming  $M$  modalities across  $N$  agents, each agent  $i \in [N]$  local datasets is  $\mathcal{D}_i = \{(\tilde{\mathbf{x}}_{ij}, y_{ij})\}_{j=1}^{n_i}$ , where  $\tilde{\mathbf{x}}_{ij} = (\mathbf{x}_{ij}^{(m)} | \forall m \in \mathcal{I}_i)$  is the input modalities of each sample  $i$  and  $\mathcal{I}_i \subseteq [M]$  is the set of active sensors at agent  $i$ . Similar to Chapter 3 and Chapter 5, there can be at most  $(2^M - 1)$  types of agents in the network. Different from them, in this chapter we will focus on the global-local decentralized paradigm and consider domain drift in conjunction with modality gap and concept shifts, by formulating the system in Personalized Federated Learning. In Figure 6.2.1, we show a trimodal federated dataset, where the cylinders in different colors (yellow, green, and red) illustrate the data collected from different types of sensors and there can be at most seven types of agents. For example, healthcare centers in remote areas usually lack advanced medical equipment so that their models relies on data collected by other available sensors; in dynamic systems or online learning applications, sensor availability may be not stable over time, thus several modalities can be missing frequently.

**Multimodal Split Networks.** Given the modality gap amongst agents, agents having different sensor setups do not share the same model architecture  $\theta_i \in \mathbb{R}^{d_i}$  where  $d_1 \neq d_2 \neq \dots \neq d_N$ , as shown in the middle of Figure 6.2.1. This will lead to difficulties in server-agent and cross-agent communication as different agent models cannot be copied or aggregated directly. One may consider a heuristic strategy unifying all agent models into the largest one by inserting missing blocks on the input layers, or deleting some blocks for under-predominant modalities. Yet this will introduce bias to model aggregation as well as not efficient. One may also argue that the agents having different sensors should be totally separated from each other during modal aggregation; however, this is not true as the different modalities across agents may still have common knowledge to learn—for example, the sound data and visual appearance of the same object can exist in different but statistically closer agents. Therefore, instead of totally unifying or separating agents, we aim to directly learn the original agent models. Inspired by the idea of split learning [212, 213] we *split* each agent models into *blocks* such that there are two types of model blocks among all agent models: 1) blocks shared globally by all agents and can be aggregated amongst the entire network; 2) blocks shared locally by the agents having the same corresponding sensors and can be aggregated across partial agents. For example, as for multimodal integration tasks [138, 20, 21, 195], which learn predictive models that integrate the information of given modalities to make decisions, we can split any agent model into one or more *modality-specific feature extractors* and a *classifier* that takes the cross-modal aligned features as input. Figure 6.2.1 illustrates the multimodal split networks, i.e., the diverse model architectures of different types of integration tasks. Formally, for parameter vector  $\theta_i \in \mathbb{R}^{d_i}$  at agent  $i$  whose sensor set is  $\mathcal{I}_i$ , we can split as

$$\theta_i = \{\theta_{i,m} | \forall m \in \mathcal{I}_i\} \cup \{\bar{\theta}_i\}, \quad (6.1)$$

where  $\theta_{i,m} \in \mathbb{R}^{d'_m}$  is the weight vector of the feature extractor for sensor- $m$  and  $\bar{\theta}_i \in \mathbb{R}^{d'}$  is the weight vector of the classifier. That is, we break the weight vector of dimension as  $d_i = d' + \sum_{m \in \mathcal{I}_i} d'_m$ . Then, the loss function for the multimodal

sample  $(\tilde{\mathbf{x}}_{ij}, y_{ij})$  can be computed as

$$l_{ij}(\boldsymbol{\theta}_i; \tilde{\mathbf{x}}_{ij}, y_{ij}, \mathcal{I}_i) := l_i \left( f_{\text{inter}} \left( \bigoplus_{m \in \mathcal{I}_i} \mathbf{h}_i^{(m)}; \bar{\boldsymbol{\theta}}_i \right), y_{ij} \right), \quad (6.2)$$

where the modality-specific hidden feature  $\mathbf{h}_{ij}^{(m)} = f_{\text{enc}}^{(m)}(\mathbf{x}_{ij}^{(m)}; \boldsymbol{\theta}_{i,m})$  is obtained through the sensor  $m$ 's specific feature encoder  $f_{\text{enc}}^{(m)}$ . Then,  $\bigoplus$  denotes the sum of the hidden representations of individual modalities, which combines their complementary information, and  $l_i(\cdot)$  is the loss function for each sample at agent  $i$ .

**Federated Multi-task Learning amongst Multimodal Split Networks.** Regarding the modality incongruity between agents together with the statistical and systematic challenges of FL problem, we formulate our problem based on the FMTL framework. The idea is that FMTL naturally explores the agent relationships which can also help to find relationships between the split blocks of agent models and is suitable for multimodal FL. Formally, we aim to learn a set of implicitly correlated models  $\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_N$  having different parameter spaces by minimizing the objective:

$$\min_{\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_N, \boldsymbol{\Omega}_1, \dots, \boldsymbol{\Omega}_M, \bar{\boldsymbol{\Omega}}} \left\{ \sum_{i=1}^N \sum_{j=1}^{n_i} l_{ij}(\boldsymbol{\theta}_i; \tilde{\mathbf{x}}_{ij}, y_{ij}, \mathcal{I}_i) + \mathcal{R}(\oplus, \boldsymbol{\Lambda}) \right\} \quad (6.3)$$

where  $\oplus = \{\boldsymbol{\theta}_i \in \mathbb{R}^{d_i}\}_{i=1}^N$  represents a collection of the multimodal models and  $\boldsymbol{\Lambda} = [\boldsymbol{\Omega}_1, \dots, \boldsymbol{\Omega}_M, \bar{\boldsymbol{\Omega}}] \in \mathbb{R}^{N \times N \times (M+1)}$  is a tensor representing multi-view relationships amongst agent models. Each view of the relationships  $\boldsymbol{\Lambda}_{\cdot, \cdot, m} \in \mathbb{R}^{N \times N}$  is a matrix corresponding to the relationships amongst certain type of blocks of all agent models. If agent  $i$  and agent  $i'$  do not have the common block- $m$ ,  $\boldsymbol{\Lambda}_{i, i', m} = 0$ . Basically, the first term of Eq.(6.3) allows agents to learn on its own local data, while the second term encourages them to take advantages of related models from other agents'. It is noticeable that, for any pair of agent- $i$  and agent- $i'$ : (1) *Observation 1*: their models  $\boldsymbol{\theta}_i, \boldsymbol{\theta}_{i'} \in \oplus$  may be not comparable as they may belong to different parameter spaces, i.e.,  $d_i \neq d_{i'}$ ; (2) *Observation 2*: the relationship between  $w_i$  and  $\boldsymbol{\theta}_{i'}$  is measured by a non-scalar but multi-dimensional vector  $\boldsymbol{\Lambda}_{i, i', \cdot} \in \mathbb{R}^{M+1}$ . As a result, directly optimizing  $\mathcal{R}(\oplus, \boldsymbol{\Lambda})$  to enforce nuanced model relations can be intractable.



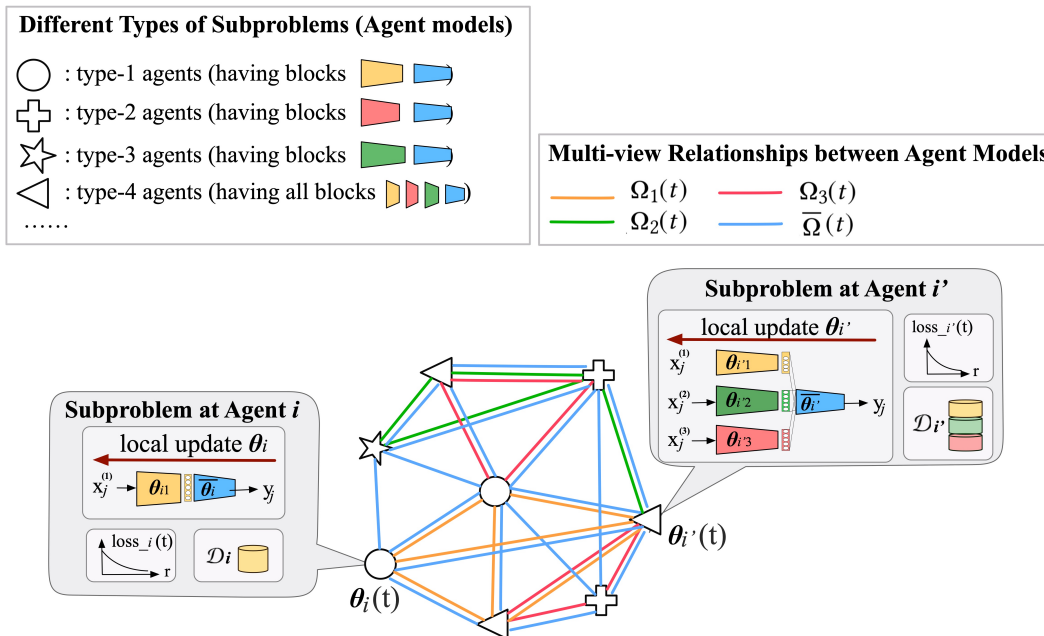


Figure 6.2.2: A graph view for training separate but related agents having different concepts, domains, and input modalities, with predefined splittable model architectures. Each vertex solves separate but related subproblems. Node features are model parameters, which changes over time based on local data and related agents.

### 6.3 Proposed FedMSplit

In this section, we will introduce a federated training algorithm to solve the objective Eq.(6.3). We will focus on the following issues during the federated training with modality incongruity. (1) **Adaptive model correlation with local dynamics.** The correlation tensor  $\mathbf{\Lambda}$  is a measurement of statistical similarity between local datasets. With the privacy requirements in FL, we cannot calculate it in advance. Further, while we can arbitrarily provide  $\mathbf{\Lambda}$  as priori [65, 64], in real-world applications, the relationships of agents may not be fixed all the time. For example, the statistics of local data can change through time such as given time-series data or continual learning tasks. Therefore, adaptively learning  $\mathbf{\Lambda}$  with models is necessary. However, it is difficult to *simultaneously optimize* the multi-space parameters  $\oplus = \{\theta_i \in \mathbb{R}^{d + \sum_{m \in \mathcal{I}_i} d_m}\}_{i=1}^N$  and the tensor  $\mathbf{\Lambda}$ . We adopt the *alternative optimization approach* following [28]. At each round  $t$ , while fixing the structure  $\mathbf{\Lambda}(t)$ , we optimize local model parameters  $\oplus(t)$  based on local

datasets  $\mathcal{D}_i, i = 1, \dots, N$  and the penalized term given  $\mathbf{\Lambda}(t)$ ; then, while fixing  $\oplus(t)$ , we optimize the model correlation tensor  $\mathbf{\Lambda}(t + 1)$  based on current local models  $\oplus(t)$ . The model correlation is dynamically updated along with the convergence of local models. (2) **Correlated model update constrained by multi-view relationships**: The individual models of different agents may be not comparable if they belong to different parameter spaces. In this sense, how to measure model relationships, and how to leverage relational local model training according to multiple views of relations, remain unexplored. (3) **Agent selection among multiple types of agents**: Vanilla FTML relies on the complete adjacency for all the agents and update all the agent models at each round. Although a complete relationship structure could benefit the correctness of correlated local updates, in practical scenarios where massive agents participate in the training, the computation time and cost of storage at each round would huge. Regarding the systematic challenge and communication limitation [211], instead of computing all agents at each round, we sample a subset of agents to participate into training. However, given the multimodal discrepancy, the difficulty is that how to select agents at each round such that each model blocks are optimized in a balanced manner and we efficiently find the optimal for all agents.

We propose the **FedMSplit** framework to allow federated training over multimodal agents without assuming the congruity of sensor types over agents. Details of the framework are as follows.

### 6.3.1 Correlation-adaptive Model Update

According to the *alternative optimization* process, at each round  $t$ , while fixing the structure  $\mathbf{\Lambda}(t)$ , we optimize local model parameters  $\oplus(t)$  based on local datasets  $\{\mathcal{D}_i\}_{i=1}^N$  and the penalized term given  $\mathbf{\Lambda}(t)$ . In other words, the local model  $\theta_i(t)$  of each agent  $i$  updates depends on not only the local dataset  $\mathcal{D}_i$  but also its related agents' datasets  $\mathcal{D}_{i'}$ , which is not seen but can be reflected by  $\theta_{i'}(t)$ . Then, any pair of  $\mathbf{\Lambda}_{i,i'}(t)$  can update based on  $\theta_i(t)$  and  $\theta_{i'}(t)$ . Through this process,  $\mathbf{\Lambda}(t), \oplus(t)$  are dynamically updated until convergence.

Given the heterogeneity of parameter space over agents, “updating  $\oplus(t)$  fixing  $\mathbf{\Lambda}(t)$ ” would be difficult as the calculation of  $\nabla_{\oplus(t)} \mathcal{R}(\oplus(t), \mathbf{\Lambda}(t))$  relies on the  $\oplus(t)$

having multiple parameter spaces, and the multi-view relationships  $\mathbf{\Lambda}(t)$ . The idea is that we can allow the two sets of parameters  $\mathbf{\Lambda}(t), \oplus(t)$  to be embedded in a dynamic graph and then solve them as a node-edge alternative updating problem.

### 6.3.1.1 Dynamic Multi-view Graph of Subproblems

We define a dynamic multi-view graph structure  $\mathcal{G}(t) = (\mathcal{V}, \Phi(t), \mathcal{E}, \mathbf{\Lambda}(t), q)$  which consists of the following components and properties:

- $\mathcal{V} = \{v_i\}_{i=1}^N$  is the vertex set, where each vertex  $v_i$  is associated with a agent  $i$  containing a local multimodal dataset  $\mathcal{D}_i = \{(\tilde{\mathbf{x}}_{ij}, y_{ij})\}_{j=1}^{n_i}$ . Each vertex represents a *subproblem*: the agent  $i$  aims to fit a model  $\theta_i \in \mathbb{R}^{d_i}$  to its local data  $\mathcal{D}_i$ .
- $\Phi(t) = \{\theta_i(t) \in \mathbb{R}^{d_i}\}_{i=1}^N$  is the content of vertices, representing the *model parameters* of each agents at round  $t$ . In this way, the model parameters of agents can be treated as *agent embeddings* in the graph. agent model updating implies the changing of agent embeddings through time, so we say the graph is *dynamic*. Recall that the agent models has different parameter spaces  $\theta_i(t) = \{\theta_{i,m}(t) | \forall m \in \mathcal{I}_i\} \cup \{\bar{\theta}_i(t)\}$  so that the vertices are multimodal and message cannot be directly transferred across vertices.
- $\mathcal{E} = \{e_{i,i'}\}_{i,i'=1}^N$  is the edge set. Edges are undirected and fully connected, and each edge refers to the similarity between a pair of agents.
- $\mathbf{\Lambda}(t) = [\mathbf{\Omega}_1(t), \dots, \mathbf{\Omega}_M(t), \bar{\mathbf{\Omega}}(t)] \in \mathbb{R}^{N \times N \times (M+1)}$  represents the edge features. An edge feature  $\mathbf{\Lambda}(t)_{i,i'}$ , indicates the model weight similarities between two agents and consists of *multiple dimensions (multi-view)* of Euclidean distances; each dimension is corresponding to a type of blocks in agent models. If agent  $i$  and agent  $i'$  do not have the common block- $j$ ,  $\mathbf{\Lambda}_{i,i',m}(t) = 0$ . The edge features of a fully connected graph with massive agents can be huge. Fortunately, in practice, the server does not need to calculate or store them until the end of each round, and only the edges between participated agents at each round will be calculated (see Section 4.2.2.).

- $q : \mathbb{R}^{M+1} \rightarrow \mathbb{R}$  is a function to measure the similarity between any of the two agents  $(i, i')$  based on the multi-view correlation vector  $\mathbf{\Lambda}_{i,i',\cdot} \in \mathbb{R}^{M+1}$ .

Figure 6.2.2 illustrates the defined dynamic graph structure, where we use different shapes to indicate individual agents having different types of parameter spaces. The embeddings of agents (model parameters) change over time. For simplicity, we only show eight agents, three modalities, and four types of agents here. Note that in real-world applications we would have more agent types in a large-scale graph containing massive number of agents.

Giving the graph of local problems, then the intractable term  $\mathcal{R}(\oplus, \mathbf{\Lambda})$  can be rewritten as the linear combination of multiple views of regularization terms (each view is corresponding to a type of blocks):

$$\mathcal{R}(\oplus, \mathbf{\Lambda}) = \lambda \overline{\mathcal{R}}(\overline{\mathbf{W}}, \mathbf{\Lambda}, \overline{\mathbf{\Omega}}) + \sum_{j=1}^M \lambda_m \mathcal{R}_m(\mathbf{W}_m, \mathbf{\Lambda}, \mathbf{\Omega}_m) \quad (6.4)$$

where  $\overline{\mathbf{W}} = [\overline{\boldsymbol{\theta}}_1, \dots, \overline{\boldsymbol{\theta}}_N] \in \mathbb{R}^{d' \times N}$  and  $\mathbf{W}_m = [\boldsymbol{\theta}_{i,m}; \forall i \text{ if } m \in \mathcal{I}_i] \in \mathbb{R}^{d'_m \times N_m}$  are matrices. Each matrix is a collection of a specific block in all agents, over one parameter space.

### 6.3.1.2 Correlation-adaptive Model Optimization

Updating correlation  $\mathbf{\Lambda}(t)$  fixing  $\oplus(t)$  is treated as updating edge features based on current node features (agent embeddings). Formally, for each pair of agents  $(i, i')$  and each split block  $j$ , their relationship can be measured as  $\boldsymbol{\Omega}_{j,kl} = \text{Att}(\boldsymbol{\theta}_{i,m}, \boldsymbol{\theta}_{i',m})$  using any metric or attention function  $\text{Att}(\cdot, \cdot)$ , such as additive attention, dot product, multiplicative attention [18]. In the experiments, we use dot product for all simulations.

Then, updating models  $\oplus(t)$  fixing the structure  $\mathbf{\Lambda}(t)$  is viewed as updating node features (agent models) based on local datasets as well as current edge features (agent-agent relationships). The idea is to take into consideration the current overall agent-to-agent relationships  $\mathbf{\Lambda}(t)$  to the local training of each agent  $\boldsymbol{\theta}_i$ . More specifically, we take two steps to incorporate such heterogeneous and multi-space complex relationships. *First*, for each agent, we approximate its complex neighborhood information, by aggregating other agent models through a multi-

view, attentive, and graph-based message passing process.

$$\begin{cases} \bar{\boldsymbol{\theta}}_i^{agg} \leftarrow \sum_{l=1}^N \frac{q(\boldsymbol{\Lambda}_{i,i',\cdot}) \bar{\boldsymbol{\Omega}}_{i,i'}}{\sum_{p=1}^N q(\boldsymbol{\Lambda}_{kp,\cdot}) \bar{\boldsymbol{\Omega}}_{kp}} \bar{\boldsymbol{\theta}}_{i'}, \\ \boldsymbol{\theta}_{i,m}^{agg} \leftarrow \sum_{l=1}^N \frac{q(\boldsymbol{\Lambda}_{i,i',\cdot}) \boldsymbol{\Omega}_{j,kl}}{\sum_{p=1}^N q(\boldsymbol{\Lambda}_{kp,\cdot}) \boldsymbol{\Omega}_{j,kp}} \boldsymbol{\theta}_{i',m} \text{ for } \forall m \in \mathcal{I}_i \cap \mathcal{I}_{i'}. \end{cases} \quad (6.5)$$

After that, each agent  $i$  independently performs local SGD on  $\mathcal{D}_i$  by  $\tau$  steps, meanwhile, it considers extra relational information:

$$\boldsymbol{\theta}_i \leftarrow \boldsymbol{\theta}_i - \alpha \frac{1}{n_i} \sum_{j=1}^{n_i} (\nabla_{\boldsymbol{\theta}_i} l_{ij}(\boldsymbol{\theta}_i, \tilde{\boldsymbol{x}}_{ij}, y_{ij}; \mathcal{I}_i) + \nabla_{\boldsymbol{\theta}_i} \lambda R_i(\boldsymbol{\theta}_i)), \quad (6.6)$$

where the multi-view relational information is incorporated by minimizing the Mean Squared Error (MSE) loss between the model and the approximate neighborhood information  $R_i(\boldsymbol{\theta}_i) = \|\bar{\boldsymbol{\theta}}_i - \bar{\boldsymbol{\theta}}_i^{agg}\|_2^2 + \sum_m \|\boldsymbol{\theta}_{i,m} - \boldsymbol{\theta}_{i,m}^{agg}\|_2^2$ . And  $\lambda$  is a hyperparameter to balance the local personalization and global correlation.

In this way, the model parameters  $\boldsymbol{\theta}_i$  is updated based on  $\mathcal{D}_i$  (Eq.(6.6)) as well as other models parameters  $\boldsymbol{\theta}_{i'}$  which is related to  $\boldsymbol{\theta}_i$  (Eq.(6.5)). The dynamic graph becomes stable once all the agent models converge and correlated.

### 6.3.2 Federated Training

We present the training workflow of FedMSplit. We show that the federated multitask learning over the multimodal split networks of agents can be done in a way like learning the agent embeddings in  $\mathcal{G}(t) = (\mathcal{V}, \boldsymbol{\Phi}(t), \mathcal{E}, \boldsymbol{\Lambda}(t), q)$  through multiple rounds ( $t = 1, 2, \dots, T$ ) of agent-server interactions until convergence.

#### 6.3.2.1 Alternative Optimization on Subgraphs

The convergence of  $\boldsymbol{\Phi}$  is achieved by multiple rounds of alternative optimization. However, due to communication cost and the systematic challenge [211], it is consuming as well as impossible to calculate the complete correlation tensor  $\boldsymbol{\Lambda}(t)$  for all the agents and update all the agent models  $\boldsymbol{\Phi}(t)$ —in each round we would have  $O(N^2)$  time and space complexity. Therefore, instead of computing all agents, we sample a subset of agents  $\mathcal{S}(t) \subset \mathcal{V}$  to participate at each round  $t$ . In other words, at each round  $t$ , we select a subgraph  $\mathcal{G}_s(t)$  of  $\mathcal{G}(t)$  to per-

---

**Algorithm 3:** Federated Training Algorithm of FedMSplit
 

---

- 1: **Input:** agents  $i \in [N]$ , modalities  $m \in [M]$ , multimodal datasets  $\mathcal{D}_1, \dots, \mathcal{D}_N$  and active sensor sets  $\mathcal{I}_1, \dots, \mathcal{I}_N$  of agents
  - 2: **Hyper-parameters:**  $\alpha, T, \tau, C, \gamma$
  - 3: For each client  $k$  in parallel, initialize  $\theta_i$  and split the model into  $|\mathcal{I}_i| + 1$  blocks as Eq.(6.1).
  - 4: Initialize  $I_i(0) = P_i(0) = L_i(0) = 0$
  - 5: Initialize  $\theta_i^{agg} = \theta_i$
  - 6: **for** each round  $t = 1, \dots, T$  **do**
  - 7:   Sampling a subset of  $C$  agents  $\mathcal{S}(t) \subset [N]$  using Eq.(6.8)
  - 8:   // local SGD independently
  - 9:   **for** each participant client  $k \in \mathcal{S}(t)$  **do**
  - 10:     **for** each local update step **do**
  - 11:       update  $\theta_i$  as in Eq.(6.6)
  - 12:     **end for**
  - 13:   **end for**
  - 14:   Send  $\{\theta_i\}_{i \in \mathcal{S}(t)}$  to the server
  - 15:   // adapt model correlation via attentive aggregation
  - 16:   **for** each participant client  $k \in \mathcal{S}(t)$  **do**
  - 17:     Split  $\theta_i$  into blocks
  - 18:     **for** each related client  $l \in \mathcal{S}(t)$  **do**
  - 19:       Calculate attention weight for each pairs of blocks  $\theta_i$  and  $\theta_{i'}$ .
  - 20:     **end for**
  - 21:     Obtain aggregated model  $\theta_i^{agg}$  using Eq.(6.7)
  - 22:   **end for**
  - 23:   Server sends  $\{\theta_i^{agg}\}_{i \in \mathcal{S}(t)}$  to agents.
  - 24:   Update  $I_i(t), P_i(t), L_i(t)$  using the selected agents in  $\mathcal{S}(t)$  and counting each type of blocks in  $\mathcal{S}(t)$ .
  - 25: **end for**
  - 26: **return:** Each client will store its final model  $\theta_i$ .
-

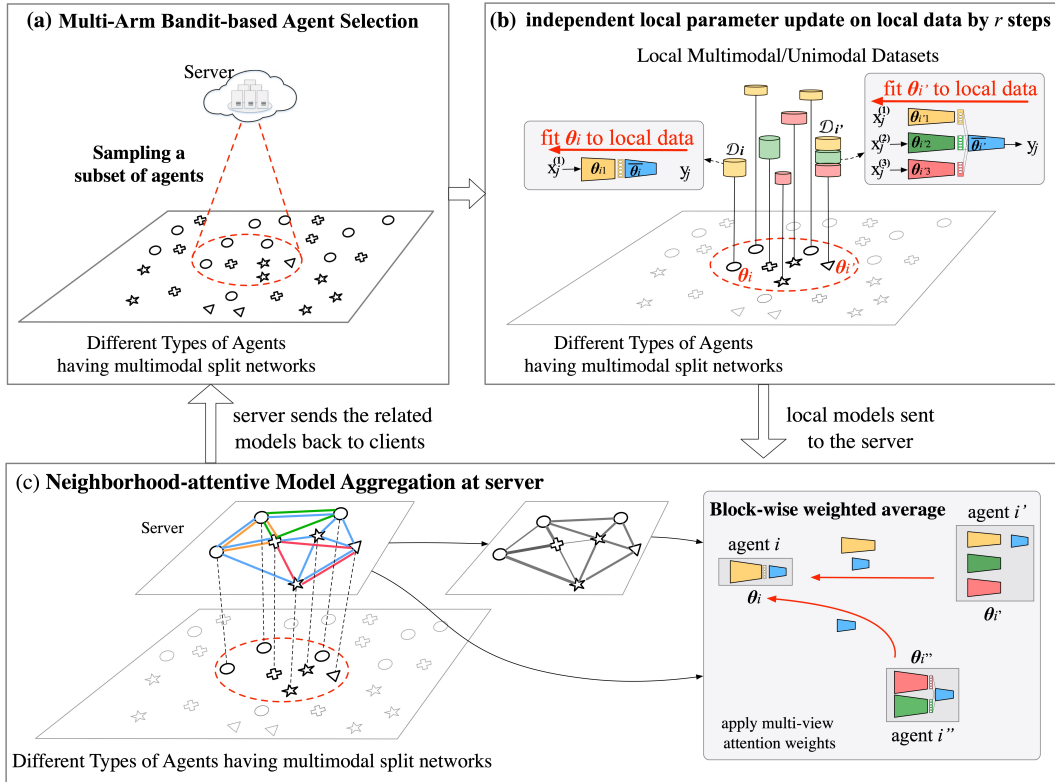


Figure 6.3.1: FedMSplit workflow at each server-agent communication round.

form alternative optimization. In particular, we consider a subgraph of agents and their relationships  $\mathcal{G}_s(t) = (\mathcal{S}(t), \Phi_s(t), \mathcal{E}_s, \Lambda_s(t), q)$  where  $\Phi_s(t) = \{\theta_i(t)\}_{i \in \mathcal{S}(t)}$  and  $\Lambda_s(t) \in \mathbb{R}^{|\mathcal{S}(t)| \times |\mathcal{S}(t)| \times (M+1)}$ . Instead of optimizing on the entire graph at each round, our method is more practical in real operation—complexity is  $O(C^2)$  where  $C = |\mathcal{S}(t)| \ll N$ , but note that we somewhat tradeoff the convergence speed because only partial agents and their relationships are considered while neglecting other related agents.

At each round, we perform one-step alternative optimization on subgraph  $\mathcal{G}_s(t)$  through the following agent-server communication. (1) On the server, we update  $\Lambda_s(t)$  fixing  $\Phi_s(t)$  and then propagate among the separate agent models over subgraph  $\mathcal{S}(t)$  as in Eq.(6.5). Note that in practice, the two steps can be replaced by applying a multi-head attention mechanism to node propagation among a subgraph—treating  $\Lambda_s(t)$  as the attention coefficients (see the next section for details). (2) The server then sends the aggregated relational information to each agent. (3) On each agent, we improve  $\Phi_s(t)$  on local datasets while considering

potential agent relationship  $\Lambda_s(t)$ . Each agent  $i \in \mathcal{S}(t)$  perform local update by local SGD and multi-relational regularization (Eq.(6.6)), and, finally, all participants send their new models  $\Phi_s(t+1)$  back to the server. Figure 6.3.1 shows the overview of FedMSplit training at each round, and the pseudocode of FedMSplit is summarized in Algorithm 3.

### 6.3.2.2 Neighborhood-attentive Model Aggregation

Once the server receives new agent models  $\Phi_s(t+1) = \{\theta_i\}_{i \in \mathcal{S}(t)}$  (the models after performing  $\tau$  steps of local updates), it adapts current model correlation to be able to promote more relational local models in future rounds. Since we use the normalized relationships among all the neighbors of agent  $i$  in Eq.(6.5), this model aggregation can be treated as 1-hop attentive message passing [214, 215, 195] among a subgraph. That is, for  $\forall i \in \mathcal{S}(t)$ ,

$$\begin{cases} \bar{\theta}_i^{agg} \leftarrow \sum_{l \in \mathcal{S}(t)} \frac{q(\Lambda_{i,i'}) \text{Att}(\bar{\theta}_i, \bar{\theta}_m)}{\sum_{p \in \mathcal{S}(t)} q(\Lambda_{kp}) \text{Att}(\bar{\theta}_i, \bar{\theta}_{i'})} \bar{\theta}_{i'}, \\ \theta_{i,m}^{agg} \leftarrow \sum_{l \in \mathcal{S}(t)} \frac{q(\Lambda_{i,i'}) \text{Att}(\theta_{i,m}, \theta_{i',m})}{\sum_{p \in \mathcal{S}(t)} q(\Lambda_{kp}) \text{Att}(\theta_{i,m}, \theta_{pj})} \theta_{i',m} \text{ for } \forall m \in \mathcal{I}_i \cap \mathcal{I}_{i'}, \end{cases} \quad (6.7)$$

where agents having similar statistics will become more related through the weighted model aggregation.

### 6.3.2.3 Agent Sampling via Multi-armed Bandit

To reduce communication cost of FMTL, we operate on a subset of agent at each round. Yet the random sampling strategy (i.e., unbiased agent selection) of typical FL frameworks may significantly suffer from non-IID local distributions as well as the multimodal discrepancy of MFL. The problem is that the agents selected at each round (i.e, a subset of vertices in the large-scale graph and perform message passing) may not contain balanced numbers of each type blocks, thus we may not efficiently find the accurate correlations between different types of agents and modalities. In order to achieve faster convergence, we aim to select agents having larger local loss (i.e., exploitation) [216, 217] as well as having blocks that were less frequently seen before (i.e., exploration). Following [217], to balance the exploration-exploitation trade-off in the multimodal agent selection



problem. we employ Multi-Armed Bandit (MAB) algorithms [218] for the problem of agent selection in Multimodal FL. Regarding the local loss of individual agents are non-stationary during training, we make use of the discounted MAB algorithms as in [217]. The agents are viewed as *arms* in the MAB problem. The discounted cumulative local loss of each agent is  $L_i(t) = \sum_{t'=1}^t \gamma^{t-t'} F_i(t')$ ; the discounted number of times each agent has been selected over the previous rounds is  $I_i(t) = \sum_{t'=1}^t \gamma^{t-t'} 1_{i \in \mathcal{S}(t')}$ ; and, the discounted number of times each type of block  $j$  has been sampled over previous rounds is,  $P_m(t) = \sum_{t'=1}^t \gamma^{t-t'} 1_{m \in \mathcal{I}_i \forall i \in \mathcal{S}(t')}$ . Here,  $0 \leq \gamma \leq 1$  is the discount rate.

Then, we define the estimated UCB reward of agent  $i$  up to round  $t$  as

$$A_i(t) = L_i(t)/I_i(t) + U_i(t) \quad (6.8)$$

where  $U_i(t) = \sqrt{\sum_{r=1}^t \gamma^{t-r} / (I_i(t) + \sum_{m \in \mathcal{I}_i} P_m(t))}$  is the exploration term for agent  $i$ . At communication round  $t$ , we select the top  $C$  agents with largest discounted UCB rewards. The first term of Eq.(6.8) enforces selecting agents with estimated larger local loss (exploitation) [217]. However, if certain agent has not been selected recently, or any type of model block of the agent has not been selected recently,  $U_i(t)$  will get larger. This forces the server to select them regardless of their local loss values (exploration).

## 6.4 Experiments

### 6.4.1 Simulations

**Datasets:** We choose three multimodal integration datasets to create our simulation environments. (1) **Vehicle Sensor** [219] for classifying vehicles driving by a segment of road. It contains 23 instances. Each instance is a separate agent described by 50 acoustic and 50 seismic features and we predict between AAV-type and DW-type vehicles. (2) **ModelNet40** [146] dataset for *multi-view 3D object recognition* tasks. It contains 12,311 3D shapes covering 40 common categories, including airplane, bathtub, bed, bookshelf, chair, cone, cup, and so on. Each 3D CAD object has  $M = 2$  modalities as two views of its shapes [141]. (3) **IEMOCAP**

Table 6.4.1: Statistics of Multimodal Federated Simulations.

Dataset	#Agents	#Modalities	Feature Sizes ( $\{\text{modality } m : F_m\}$ )	$ \mathcal{I}_i $ range	$n_i$ (mean, std)	#Concept
Vehicle Sensor	23	2	{Acoustic: (50) , Seismic: (50)}	[1,2]	$\mathcal{N}(255, 50)$	2
ModelNet40	12	2	{View 1: (4096), View 2: (2048)}	[1,2]	$\mathcal{N}(1026, 200)$	5
IEMOCAP	15	3	{Acoustic: (74) , Text: (300) , Visual: (35)}	[1,3]	$\mathcal{N}(297, 80)$	2

[148] for *emotion recognition tasks*. It consists of a collection of 4,453 video segments of recorded dialogues. Each segment is annotated for the presence of 9 emotions (happy, angry, excited, fear, etc.), from which we use only the “happy” tag for binary classification. We adopted the same feature extraction scheme [20] for language, visual and acoustic modalities. The feature sizes of the modalities are summarized in Table 6.4.1.

**Simulation of Statistical Heterogeneity:** The size of training samples at each agent is sampled from a Gaussian distribution whose mean and standard deviation is pre-defined as in Table 6.4.1. The domain shifts are created by randomly adding noises to the original videos, audios, and images; randomly rotating the visual objects; replacing the image backgrounds; or, replacing low-frequency signals in raw audios with other sounds. The concepts shifts are created by permuting the local label spaces and limiting the number of labels of each agent. The maximum number of classes per agent is shown in Table 6.4.1.

**Simulation of Modality Incongruity:** We impose no restrictions on the modality or combinations of modalities used in the local agents. We simulate this real-world scenario as follows. First, we assume the availability of each sensor  $j$  follows a Bernoulli distribution  $\text{Bernoulli}(\rho_j)$  and different sensors are independent. Here, we use a missing rate  $\rho_j$  to indicate the probability a agent does not have the modality- $j$ . We set equal missing rates for each modalities  $\rho_1 = \dots = \rho_M = \rho$  in all experiments. After that, we shuffle the agents and for each possible sensor set  $\mathcal{I}' \subset \{1 \dots M\}$  we separately pick  $N(\mathcal{I}', \rho)$  agents and assign the sensor set  $\mathcal{I}'$  to each of them. For example, for IEMOCAP dataset ( $M = 3$ ), there will be 7 types of agents focusing on different tasks: audio-only, text-only, video-only, audio-text, audio-visual, text-video, and audio-text-visual tasks.

### 6.4.2 Baselines

We compare FedMSplit with three categories of baselines: (1) *Fully local* training on *multimodal* federated datasets: **Local** separately trains personal models that have different building blocks, without considering their potential relationships. (2) *Fully global* and *multimodal* FL frameworks: **FedAvg**, **Multi-FedAvg**, and **Multi-FedProx**, where we apply the vanilla FedAvg [33] and FedProx [24] to our multimodal federated datasets. (3) *Local but globally related multimodal* FL methods: **MOCHA** and **Multi-MOCHA**.

**Reproducibility and Hyperparameters:** The local objectives use cross entropy losses. In all experiments, we fix  $\gamma = 0.9$ ,  $\tau = 4$ , and  $\eta = 0.005$  for Vehicle Sensor and ModelNet40; and fix  $\tau = 1$ ,  $\eta = 0.00002$  for IEMOCAP. Model configurations are as follows. The encoded modality’s hidden dimension is  $P = 32$  for vehicle sensor dataset,  $P = 128$  for ModelNet,  $P = 64$  for IEMOCAP. **FedAvg** and **MOCHA**’s global model consists of 2 layers,  $\{\sum_{j \in M} \text{in\_dim}_j \times P, P \times \text{num\_class}\}$ . For local model training, we replace the missing sensor data by zeros and then directly concatenate the input modalities into one feature. **Multi-FedAvg**, **Multi-FedProx**, and **Multi-MOCHA** uses individual feature extractors for each modality (i.e.,  $\text{in\_dim}_j \times P$ ) followed by a classifier (a fully connected layer of size  $P \times \text{num\_class}$  followed by Softmax). The output of feature extractors (modality-specific hidden representations) are combined using sum operation. **Local** uses the partial architecture of Multi-FedAvg or Multi-FedProx, including the feature extractors for only available modalities (i.e.,  $\text{in\_dim}_j \times P$ ) followed by a classifier (a fully connected layer of size  $P \times \text{num\_class}$  followed by Softmax). Note that there is no feature extractor block in the agents having missing modalities. FedMSplit’s model architectures are the same as the **Local**, which are partial architectures of the complete model architecture of other methods.

### 6.4.3 Main Results

Table 6.4.2 reports the average local testing accuracy of FedMSplit compared with baselines, under different levels of modality incongruity and non-IID scenarios. We

Table 6.4.2: Average Testing Accuracy (%) on Agent Local Testing Data at global round  $T=10$  (Non-IID,  $C = 0.3N$ ).

Method	Vehicle Sensor			ModelNet40			IEMOCAP		
	$\rho=0.5$	$\rho=0.7$	$\rho=0.8$	$\rho=0.3$	$\rho=0.5$	$\rho=0.7$	$\rho=0.5$	$\rho=0.7$	$\rho=0.9$
FedAvg [33]	75.26	74.42	68.48	87.48	86.79	83.48	80.48	79.71	-
Multi-FedAvg [33]	76.98	73.61	75.59	85.75	75.46	74.60	70.86	-	60.48
Multi-FedProx [24]	76.92	74.69	72.52	93.48	74.57	-	79.62	79.33	79.14
Local	74.84	73.56	69.29	91.56	90.20	83.18	73.24	73.71	54.48
MOCHA [28]	80.28	76.65	73.28	90.03	95.88	88.54	82.38	82.10	82.86
Multi-MOCHA [28]	78.35	76.61	75.73	<b>98.25</b>	96.06	90.70	80.95	80.19	80.86
<b>FedMSplit</b>	<b>81.92</b>	<b>78.85</b>	<b>77.68</b>	<b>98.34</b>	<b>98.54</b>	<b>98.38</b>	<b>85.24</b>	<b>84.16</b>	<b>84.95</b>

report the performance of the global model (FedAvg, Multi-FedAvg, and Multi-FedProx) or the globally stored separate models (Local, MOCHA, Multi-MOCHA, and FedMSplit) on all the agents’ local testing data. From Table 6.4.2, we can observe that, in general, the increasing modality incongruity between agents results in performance drops of all methods. It is because as  $\rho$  increases, models receive less information used for fitting parameters. Overall, FedMSplit gained more advantages over baselines as more agents have missing modalities and more local models have inactive neurons. On ModelNet40, FedMSplit maintains its performance as  $\rho$  increases. It is because FedMSplit does not train inactive neurons as well as did not aggregate parameters as FedAves, FedProxs and FedMTLs. In FedMSplit, inactive neurons or blocks are not uploaded to the server and do not influence future models of other agents. Moreover, FedMSplit outperforms Local as well, even though Local train the same local architectures as ours. It is because in comparison to Local, the agent models in FedMSplit can obtain knowledge about the task from other agents’ data.

#### 6.4.4 Ablation Study

In Table 6.4.3, we evaluate the influence of each component in our model.

**Adaptive Correlation v.s. Non-adaptive Correlation.** First, we test a variant of FedMSplit, namely FedMSplit-nAC, such that we do not learn an adaptive correlation tensor between agents; instead, we assume that the relationships be-

Table 6.4.3: Ablation study of FedMSplit at global round  $t=10$ .

Method	Vehicle Sensor		ModelNet40	
	$\rho=0.5$	$\rho=0.7$	$\rho=0.5$	$\rho=0.7$
<b>FedMSplit-nAC</b>	80.12	75.32	98.43	97.63
<b>FedMSplit-rightAC</b>	79.88	76.82	96.20	97.67
<b>FedMSplit-leftAC</b>	78.32	77.48	97.79	96.37
<b>FedMSplit-<math>\pi_{\text{rand}}</math></b> (C=0.3N)	80.47	77.36	96.62	94.33
<b>FedMSplit-<math>\pi_{\text{UCB}}</math></b> (C=0.3N)	<b>81.92</b>	<b>78.85</b>	<b>98.54</b>	<b>98.38</b>

tween agents is given (i.e., identity matrix). That is, at each round, all the participants having the same block contribute equally to each other. We can observe that FedMSplit-nAC still outperform baselines since we split each model into blocks and avoid transferring inactive neurons that are corresponding to missing sensors. In addition, arbitrarily fixing the relationship (FedMSplit-nAC) leads to slightly performance drop rather than adapting the relationships.

**Impact of Multi-view Relationship Measurement.** Second, since in our model the agents relationships are measured as the linear combination of each block’s relationships, we are interested in whether each block in the model contributes equally to such measurement or not. In general, we use the measurement function  $q(\mathbf{\Lambda}_{i,i',.}) = \|\mathbf{\Lambda}_{i,i',.}\|_1 / (1 + |\mathcal{I}_i \cap \mathcal{I}_{i'}|)$ , where the classifier and modality-specific feature extractors have similar importance. We then tested other types of measurements: FedMSplit-rightAC, which measures two models based only on their classifier weights  $q(\mathbf{\Lambda}_{i,i',.}) = \mathbf{\Lambda}_{i,i',(M+1)}$ , and FedMSplit-leftAC, which measures two models based only on their common feature extractors  $q(\mathbf{\Lambda}_{i,i',.}) = \sum_{m \in \mathcal{I}_i \cap \mathcal{I}_{i'}} \mathbf{\Omega}_{m,i,i'} / |\mathcal{I}_i \cap \mathcal{I}_{i'}|$ . It can be observed that the equally weighed measurement achieved the best performance, as the local data may have not only different classes but also low-level appearance nuances.

**Impact of Agent Selection Strategy.** We finally tested our bandit-based agent sampling strategy that encourages the server to explore agents who have blocks that are less selected. We propose a variant FedMSplit- $\pi_{\text{rand}}$ , which replaces MAB with the random agent selection strategy, and observed that random selection converged

slower than the bandit-based counterpart, especially with a high level of modality incongruity.

## 6.5 Conclusion

This chapter introduced the FedMSplit framework for addressing the efficient and effective multi-agent collaboration with the co-existence of modality gaps, domain drifts, and concept shifts among the agents. Focusing on explicit parameter-level knowledge transfer, FedMSplit employs a dynamic graph structure to capture the adaptive correlations amongst multimodal agent models that have been split into smaller shareable blocks. The underlying statistical correlations between the different types of agents are captured as multi-view features and then are used to promote model relations. Our empirical results demonstrated the effectiveness of our method.

# Chapter 7

## Disentangled and Gated Transfer for Asymmetrical Collaboration

### 7.1 Introduction

In this chapter, we further extend the setting of Chapter 6 by incorporating task-type preferences. That is, we study cases where CoMML agents possess the freedom to have their own **modalities**, **concepts**, **task categories**, and **domains**. In such CoMML cases, the agent heterogeneity pattern consists of the simultaneous existence of **modality gap**, **concept shifts**, **task differences**, and **domain drifts**. We continue to use the global-local decentralized CoMML paradigm (Eq.(2.2)) as in Chapter 6. However, unlike previous chapters where corresponding problem settings can be found in some AI/ML fields, we have found no previous work that strictly formulates the aforementioned CoMML setting for us to use. We therefore propose a novel **Modality-task Agnostic Federated Learning (AFL)** setting to conduct research on CoMML with modality, concept, domain, and task preferences. We formally define AFL as a setup where each agent independently trains a personalized model on its own modalities and tasks, while periodically transfer trained knowledge (e.g., weights or gradients) with each other through a central server housing a global foundation model. AFL has real-world applications; one such application is the ambiguous privacy-preserving pre-training or fine-tuning frameworks of AGI that aim to learn a global multimodal foundation model as well

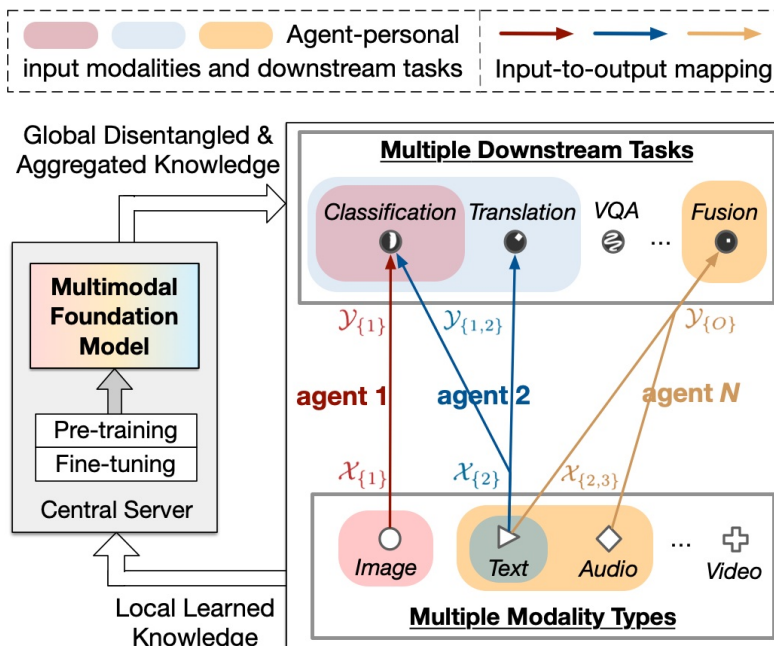


Figure 7.1.1: Modality-task Agnostic Federated Learning (AFL). Agents learn personal models for their specific modalities and tasks using local data.

as smaller personal local models for each agent, as illustrated in Figure 7.1.1.

Compared to existing settings in the FL community, the AFL problem introduces highly **Asymmetrical Knowledge Relationships (AKR)** among agents, implying that the types of mutual shareable knowledge between each pair of agents vary dramatically. The reason for AKR is attributed to the MTDC heterogeneity of AFL—simultaneous Modality gaps (M), Task gaps (T), Domain shifts (D), and Concept drifts (C) among agents. The high degree of AKR in AFL raises a crucial **challenge** in learning an optimal inter-agent information sharing scheme (i.e. maximizing positive transfer and minimizing negative transfer)—it would be difficult to efficiently and automatically identify correct transferable knowledge for each pair of agents through agent-server interactions. Existing FL works [63, 73] mainly address the symmetrical knowledge transfer between agents, which struggle to perform sufficient positive transfer and cannot fully avoid negative transfer during the inter-agent collaboration under an AKR situation.

To overcome the abovementioned challenge in AFL and achieve an optimal inter-agent information sharing scheme that maximizes positive transfer and mini-



mizes negative transfer, we propose a novel knowledge disentanglement-based federated learning framework, namely **DisentAFL**. The key idea of DisentAFL is to explicitly disentangle the original *asymmetrical* inter-agent information sharing scheme into several independent *symmetrical* inter-agent information sharing schemes, each of which corresponds to certain semantic knowledge type learned from the local tasks. DisentAFL empowers the server-agent communication to be aware of the true pairwise mutual knowledge type(s) through a **Knowledge Disentanglement (KD)** and **Gated Collaboration (GC)** mechanism. KD has two stages: the stage one leverages *coarse-grained group-wise disentanglement* to reduce the original asymmetrical problem into several *intermediate* asymmetrical subproblems, and the stage two leverages *fine-grained knowledge-type disentanglement* that further decomposes each of the asymmetrical subproblems into several independent symmetric information sharing schemes. The unique contributions of this chapter compared with the previous ones are as follows:

- We propose **DisentAFL** to address the complex asymmetrical inter-agent knowledge relationships of AFL. Technically, DisentAFL is one of first FL methods that explicitly leverage the fine-grained disentanglement of inter-agent relationships to achieve sufficient positive knowledge while excluding negative knowledge.
- We evaluate DisentAFL on three AFL simulations with 4 modalities and 4 downstream tasks. Our model is based on encoder-only Transformers, the testing modalities range from image and text to audio, video, and 3D point clouds, and the downstream tasks include both generative and classification tasks. Empirical results demonstrate the effectiveness of our method on a variety of simulation scenarios.

## 7.2 Related Works

In addition to the literature reviewed in Section 2.4, our proposed techniques and models are related to and inspired by two lines of work: 1) Disentanglement for Knowledge Transfer, and 2) Unified Multimodal Large Foundation Models.

**Disentanglement for Knowledge Transfer.** Disentanglement, initially studied in deep generative models [220, 221], has been recently utilized in multimodal representation decoupling [222], cross-modal and cross-domain transfer learning [223], and multimodal knowledge distillation [224] to enhance knowledge transfer effectiveness. In Federated Learning community, recent works [225, 226, 63, 227] show disentanglement helps to achieve better interpretability and privacy protection, as well as perform better the global-local knowledge tradeoff. Different from them, our work employs *finer*-grained disentanglement to *purify* the positive knowledge transfer among agents.

**Unified Multimodal Large Foundation Models.** The Artificial General Intelligence (AGI) aims to attain Foundation Models that emulate human-like intelligence on a variety of cognitive *tasks* across diverse *modalities* [1]. Multimodal Large Language Models (MLLMs) *pretrained* on large-scale multimodal data have emerged as a pivotal paradigm for AGI [228, 229, 230]. Pretrained MLLMs could quickly *adapt* to various multimodal downstream tasks through few-shot fine-tuning or zero-shot inference, catering to both deterministic tasks (e.g. multimodal fusion) [195, 231, 232, 233, 234] and generative tasks (e.g. cross-modal video generation) [235, 236]. To enhance the success of AGI, many Multimodal Interaction Modeling techniques have been incorporated into MLLMs and have played important roles [229, 237], including *model design* (e.g., inter-modal interaction architecture), *training* algorithms (e.g., co-training of different modalities), and *task adaptation* mechanisms (e.g., hypernetworks, soft prompting, and the prompt design of input structures that combines multiple modalities).

## 7.3 Problem Formulation

### 7.3.1 Modality-task Agnostic Federated Learning (AFL)

Assuming a total of  $M$  types of modalities and  $O$  types of downstream tasks over the  $N$  agents. Each agent  $i$  it has its own input modality types  $\mathcal{I}_i \subseteq [M]$  and target task types  $\mathcal{O}_i \subseteq [O]$ , and it aims to learn a personal input-to-output mapping

function parameterized by trainable weights  $\boldsymbol{\theta}_i \in \mathbb{R}^{d_i}$

$$f(\cdot; \boldsymbol{\theta}_i) : \mathcal{X}_{\mathcal{I}_i} \rightarrow \mathcal{Y}_{\mathcal{O}_i}. \quad (7.1)$$

$\mathcal{X}_{\mathcal{I}_i} := \text{Join}(\mathcal{X}^{(m)} | \forall m \in \mathcal{I}_i)$  is the a *agent-specific* structured/joint input space, where  $\mathcal{X}^{(m)}$  denote the raw input space associated to the  $m$ -th modality type.  $\mathcal{Y}_{\mathcal{O}_i} := \{\mathcal{Y}^{(o)} | \forall o \in \mathcal{O}_i\}$  is the *agent-specific* label space consisting of multiple subspaces for each type of local tasks, where  $\mathcal{Y}^{(o)}$  the label space for the  $o$ -th task. Figure 7.1.1 shows an illustration of the mapping function differences over agents.

Each agent  $i$  has its agent-specific input distribution  $\mathcal{P}_i(\tilde{\boldsymbol{x}})$  over the combinatorial input space  $\mathcal{X}_{\mathcal{I}_i}$  and the conditional output distribution  $\mathcal{Q}_i(\tilde{\boldsymbol{y}} | \tilde{\boldsymbol{x}})$  over the space  $\mathcal{Y}_{\mathcal{O}_i}$ . The local dataset  $\mathcal{D}_i = \{(\tilde{\boldsymbol{x}}_{ij}, \tilde{\boldsymbol{y}}_{ij})\}_{j=1}^{n_i}$  is sampled from  $\tilde{\boldsymbol{x}}_{ij} \sim \mathcal{P}_i(\tilde{\boldsymbol{x}})$  and  $\tilde{\boldsymbol{y}}_{ij} = \{\boldsymbol{y}_{ij}^{(o)}\}_{o \in \mathcal{O}_i} \sim \mathcal{Q}_i(\tilde{\boldsymbol{y}} | \tilde{\boldsymbol{x}}_{ij})$ . The **local objective** at agent  $i$  jointly minimizes losses for multimodal local downstream tasks

$$\min_{\boldsymbol{\theta}_i} \tilde{\mathcal{L}}_i(\boldsymbol{\theta}_i) := \mathbb{E}_{(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}) \sim \mathcal{D}_i} \frac{1}{|\mathcal{O}_i|} \sum_{o \in \mathcal{O}_i} \mathcal{L}^{(o)}(\boldsymbol{y}^{(o)}, f(\tilde{\boldsymbol{x}}; \boldsymbol{\theta}_i)_o) \quad (7.2)$$

where  $\mathcal{L}^{(o)}$  is the loss function for the type- $o$  task. Then, following PFL [73, 61], the **global objective** of AFL is formulated as

$$\min_{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \dots, \boldsymbol{\theta}_N} \left[ \frac{1}{N} \sum_{i=1}^N \tilde{\mathcal{L}}_i(\boldsymbol{\theta}_i) \right] + \mathcal{R}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \dots, \boldsymbol{\theta}_N), \quad (7.3)$$

where the regularizer  $\mathcal{R}(\cdot)$  indicates the information sharing scheme (i.e. knowledge transfer) among agents, which is encouraged to transfer beneficial knowledge among agents to boost each local model's performance.

### 7.3.2 Four Heterogeneity Patterns in AFL

Since agents in AFL do not necessarily have the same input modalities or downstream tasks, there could be simultaneous 4 heterogeneity patterns between agents: **Modality gap**, **Task gap**, **Domain shift**, and **Concept drift (MTDC)**. (1) **M (modality gap)**: the agents vary in their *input spaces* due to their input modality divergence, that is,  $\mathcal{X}_{\mathcal{I}_i} \neq \mathcal{X}_{\mathcal{I}_i'}$  when  $\mathcal{I}_i \neq \mathcal{I}_i'$ . For example, a vehicle may use its

onboard camera to capture videos to predict traffics, while another vehicle may use both video and RADAR signals to predict traffics. (2) **T (task gap)**: agents vary in their *output spaces*  $\mathcal{Y}_{\mathcal{O}_i} \neq \mathcal{Y}_{\mathcal{O}_{i'}}$ , since they target at different downstream tasks  $\mathcal{O}_i \neq \mathcal{O}_{i'}$ . For example, while a agent may focus on image classification, the other agent may focus on image segmentation. (3) **D (domain shift)**: slightly different from traditional FL’s definition on domain shift, AFL considers the joint distribution shift, meaning that the multimodal interaction behaviors can vary between agents. (4) **C (concept shift)**: agents vary in their conditional *output distribution*, or label space.

## 7.4 Asymmetrical Knowledge Transfer

We begin with discussing the key challenges in solving the AFL’s global objective (Eq.(7.3)) due to MTDC heterogeneity.

**Definition 7.1 (Positive & Negative Knowledge Transfer).** Positive Transfer (**PT**) is defined as the information sharing behavior between a *pair* of agents that will lead to the improvement of each other models. Negative Transfer (**NT**), on the other hand, is a phenomenon when sharing parameters between two local models results in poorer results than solving individual tasks (or, *unlearning*).

**Rethinking Information Sharing in Federated Learning.** The information sharing scheme  $\mathcal{R}(\theta_{1:N})$  in FL is essentially to find an inter-agent Pairwise Knowledge Transfer (**PKT**) mechanism that can lead to the improvement of each agent model. For any pair of agents, there exists both mutual common knowledge and conflicting knowledge between them—if  $\nabla_{\psi} f_i(\psi) \nabla_{\psi} f_{i'}(\psi) > 0$ , we say the knowledge representation  $\psi$  at agent  $i$  and agent  $i'$  aligns/matches with each other; on the other hand, if  $\nabla_{\psi} f_i(\psi) \nabla_{\psi} f_{i'}(\psi) < 0$ , the knowledge  $\psi$  at agent  $i$  and agent  $i'$  conflicts. As in [238], the transfer behavior of conflicting knowledge will result in Negative Transfer; and, the un-transfer of common knowledge will result in *insufficient* Positive Transfer. Both need to be avoided for better performance. Therefore, the optimal  $\mathcal{R}(\theta_{1:N})$  relies on a PKT mechanism that can *maximize positive transfer and minimize negative transfer* between each pair of agents—that

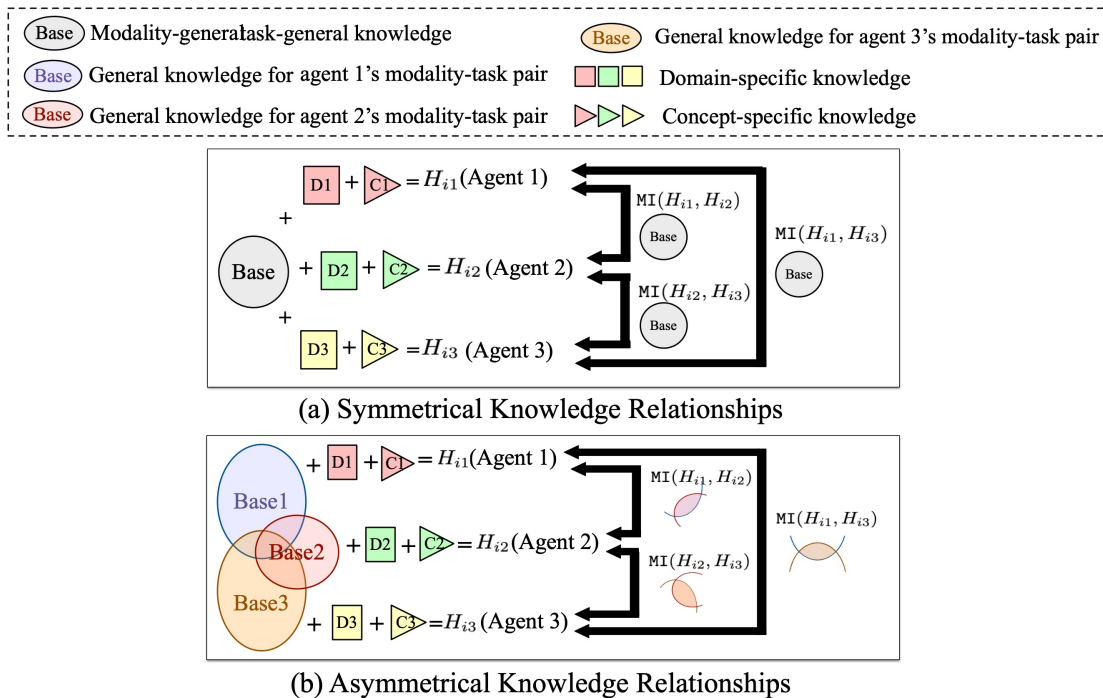


Figure 7.4.1: Comparison between symmetrical and asymmetrical inter-agent knowledge relationships. In (a), the three example agents share the same modality-task pair. In (b), the three example agents have different modality-task pairs.

is, all the true aligned knowledge is encouraged to be transferred and all the true conflicting knowledge should be excluded during transfer.

**Definition 7.2 (Symmetrical & Asymmetrical Knowledge Relationships).**

Suppose  $H_i$  denotes the knowledge learned by the agent  $i$  and  $\text{MI}(H_i, H_{i'})$  denotes the *true* mutual/common knowledge between by a pair of agents  $(i, i')$ . We say the knowledge relationships over  $N$  agents is **symmetrical** if the mutual information (common knowledge) between each pair of agents are the same  $\text{MI}(H_{i_1}, H_{i_2}) = \text{MI}(H_{i_2}, H_{i_3}) = \text{MI}(H_{i_1}, H_{i_3}), \forall i_1, i_2, i_3 \in [N]$ . On the other hand, we say the knowledge relationships over  $N$  agents is **asymmetrical** if  $\text{MI}(H_{i_1}, H_{i_2}) \neq \text{MI}(H_{i_2}, H_{i_3}) \neq \text{MI}(H_{i_1}, H_{i_3}), \exists i_1, i_2, i_3 \in [N]$ . Figure 7.4.1 shows an comparison between the two scenarios.

**Challenge of Optimizing Information Sharing in AFL.** Existing FL algorithms mainly address the **symmetrical** knowledge relationships. For example, Non-IID PFL [63, 29] with a universal domain shift and concept shift can be a

*symmetrical* case (Figure 7.4.1(a)) since there exists global common knowledge  $\text{MI}_g = \text{MI}(H_i, H_{i'})$  shared by all pairs of agents  $i, i' \in [N]$  and all the other learned knowledge is considered as personalized knowledge. However, due to the complexity of MTDC heterogeneity, **AFL** has more complex *asymmetrical* knowledge relationships among agents, as illustrated in Figure 7.4.1(b). Using the 4 agents in Figure 7.5.2 as an example, the common knowledge between agent 1 and agent 3 includes the modality-1's encoding function, which is however not shareable between agent 1 and agent 2 since the modality 1 is not learned at agent 2. Unfortunately, the asymmetrical knowledge relationships in AFL brings difficulties in optimizing the information sharing scheme  $\mathcal{R}$ —it is hard to efficiently and adaptively identify transferable knowledge for each pair of agents through agent-server interactions. Existing FL methods may result in negative transfer or insufficient positive transfer under the asymmetry of AFL.

Given such *complex* and *unknown* user-to-user knowledge sharing in AFL, it is desirable to explicitly maximize positive transfer and minimize negative transfer for the optimization of  $\mathcal{R}$ . Ideally, for any pair of agents  $(i, i')$ , an optimal PKT mechanism should perform the transfer to approximate the true mutual knowledge  $\text{MI}(H_i, H_{i'})$ .

## 7.5 Proposed DisentAFL

In order to achieve an efficient and optimal PKT mechanism that maximizes positive transfer and minimizes negative transfer with the *asymmetrical* knowledge relationships of AFL, we propose **DisentAFL**, whose overview is shown in Figure 7.5.1. The key idea is to *disentangle* the asymmetrical information sharing scheme on the original knowledge space into  $K$  independent *symmetrical* information sharing schemes on each of the disentangled knowledge subspaces

$$\mathcal{R}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \dots, \boldsymbol{\theta}_N) = \sum_{k=1}^K \mathcal{R}_k(\{\boldsymbol{\theta}_i^{(k)} | \forall i \in C_k\}) \quad (7.4)$$

such that each  $\mathcal{R}_k(\cdot)$  is a *symmetric* information sharing scheme among a subset of agents  $C_k \subseteq [N]$ , where  $\boldsymbol{\theta}_i^{(k)}$  is the disentangled knowledge type  $k$  from  $\boldsymbol{\theta}_i$ .

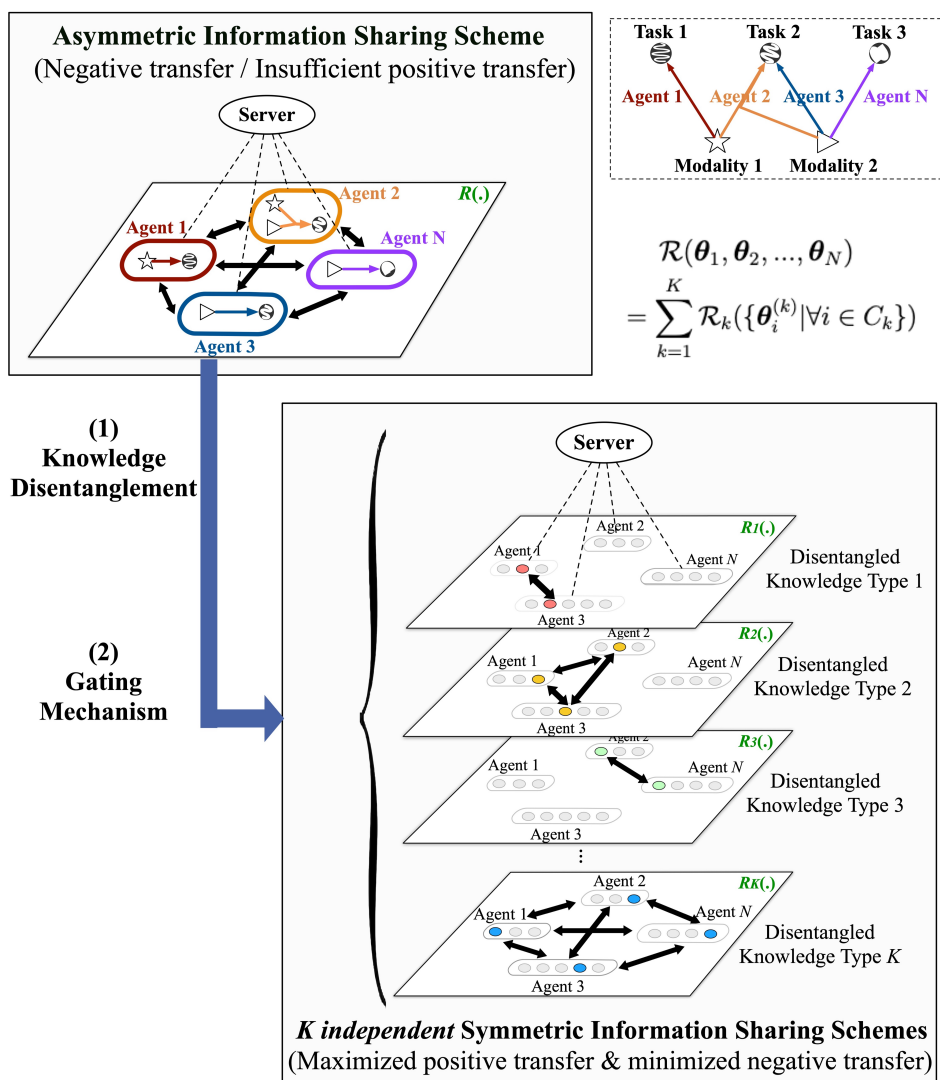


Figure 7.5.1: Overview of the proposed DisentAFL.

To find an optimal inter-agent communication solution for Eq.(7.4), DisentAFL consists of two components: Knowledge Disentanglement (KD) and Gated Collaboration (GC) mechanism. KD includes two stages: **coarse-grained** group-wise disentanglement and **fine-grained** knowledge-type disentanglement. The two-stage KD is shown in Figure 7.5.2. Then, the GC mechanism is used to automatically **route** the disentangled knowledge to improve beneficial knowledge transfer during inter-agent collaboration.

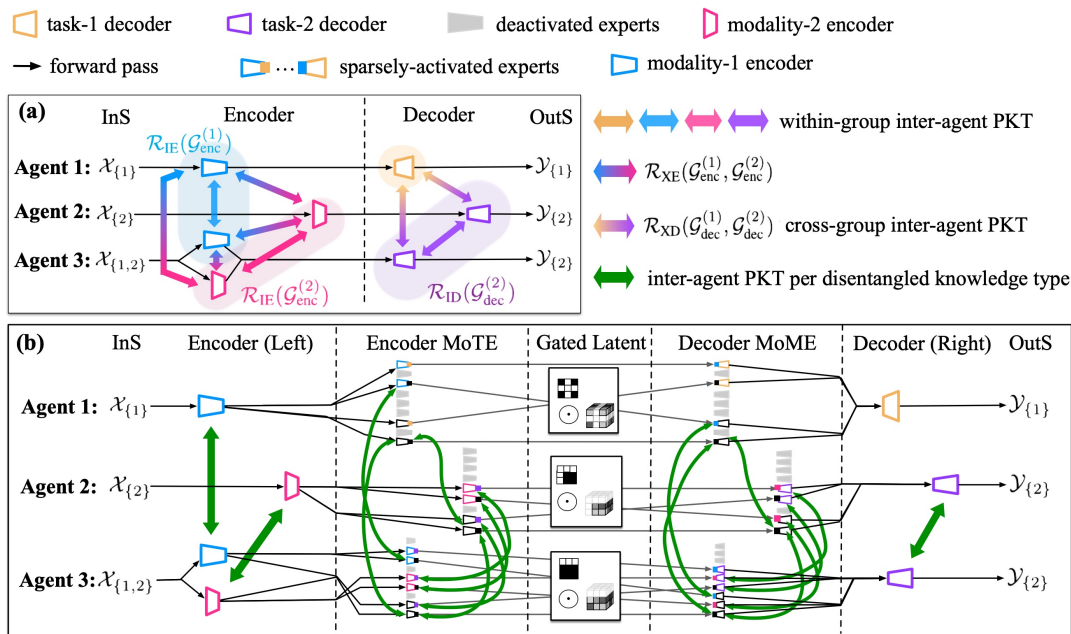


Figure 7.5.2: The two-stage Knowledge Disentanglement with Gated Collaboration mechanism in DisentAFL. (a) The intermediate asymmetrical information sharing schemes after coarse-grained disentanglement; (b) The final symmetrical information sharing schemes after fine-grained disentanglement.

### 7.5.1 Stage One: Coarse-grained Disentanglement

Group-wise disentanglement reduces the *original asymmetrical problem* with complex MTDC heterogeneity into several *intermediate asymmetrical subproblems* with less complex agent diversity. **First**, we separate the encoding and decoding related knowledge such that the agents sharing the same modality or downstream task could share the corresponding encoder or decoder parameters/representations. For example, an agent aiming at image classification task using the ViT [239] encoder and a MLP classification head, might share the image encoder with an image-text classification agent that uses a Multimodal Transformer backbone [240]. We rewrite the local parameters of  $i$ -th agent as  $\theta_i = \{\theta_{enc,i}^{(m)}\}_{m \in \mathcal{I}_i} \cup \{\theta_{dec,i}^{(o)}\}_{o \in \mathcal{O}_i}$ , where  $\theta_{enc,i}^{(m)}$  denotes the modality  $m$ 's encoder and  $\theta_{dec,i}^{(o)}$  denotes the decoder for the type- $o$  downstream task. We define two types of knowledge groups: (1) encoding-knowledge groups  $\mathcal{G}_{enc}^{(m)} = \{\theta_{enc,i}^{(m)} | \forall i \in [N] \text{ if } m \in \mathcal{I}_i\}$ , where each group is a collection of encoders from those agents having the modality  $m$  within their inputs; and (2) decoding-knowledge groups  $\mathcal{G}_{dec}^{(o)} = \{\theta_{dec,i}^{(o)} | \forall i \in [N] \text{ if } o \in \mathcal{O}_i\}$ , where each group is



a collection of decoders from those agents having the target downstream task type  $o$ . **Then**, we can rewrite the asymmetrical information sharing scheme  $\mathcal{R}(\boldsymbol{\theta}_{1:N})$  as

$$\begin{aligned} \mathcal{R}(\boldsymbol{\theta}_{1:N}) &= \sum_{m=1}^M \mathcal{R}_{\text{IE}}(\mathcal{G}_{\text{enc}}^{(m)}) + \sum_{o=1}^O \mathcal{R}_{\text{ID}}(\mathcal{G}_{\text{dec}}^{(o)}) \\ &+ \sum_{m,m'=1}^M \mathcal{R}_{\text{XE}}(\mathcal{G}_{\text{enc}}^{(m)}, \mathcal{G}_{\text{enc}}^{(m')}) + \sum_{o,o'=1}^O \mathcal{R}_{\text{XD}}(\mathcal{G}_{\text{dec}}^{(o)}, \mathcal{G}_{\text{dec}}^{(o')}). \end{aligned} \quad (7.5)$$

where the  $\mathcal{R}(\boldsymbol{\theta}_{1:N})$  with MTDC agent heterogeneity is split into four sub-problems:

- $\mathcal{R}_{\text{IE}}(\cdot)$  indicates the information sharing scheme within each modality-specific group  $\mathcal{G}_{\text{enc}}^{(m)}$ , which is an *asymmetrical* but *single-modal task-agnostic* problem with **TD** heterogeneity (no modality shift and concept shift).
- $\mathcal{R}_{\text{ID}}(\cdot)$  indicates that within each task-specific group  $\mathcal{G}_{\text{dec}}^{(o)}$ , which is an *asymmetrical* but *modality-agnostic single-task* problem with **MC** heterogeneity (no task shift and domain shift).
- $\mathcal{R}_{\text{XE}}(\cdot, \cdot)$  indicates the potential encoding-information sharing between agents having different modalities, which is an *asymmetrical* but *cross-modal task-agnostic* problem with **MT** heterogeneity (no domain shift and concept shift).
- $\mathcal{R}_{\text{XD}}(\cdot, \cdot)$  indicates the decoding-information sharing scheme between the agents that have diversified downstream tasks, which is an *asymmetrical* but *cross-task modality-agnostic* problem with **MT** heterogeneity (no domain shift and concept shift).

### 7.5.2 Stage Two: Fine-grained Disentanglement

We further disentangle each of the above four asymmetrical sub-problems into several independent symmetric problems.

To achieve this, we first need to find the largest knowledge components that can sufficiently describe the global asymmetric PKT problem as the combination of several symmetric PKT problems. Specifically, we assume a total of  $K = M(D + 1) + O(N + 1) + (M + 1)(O + 1)$  fine-grained knowledge types **globally existing**

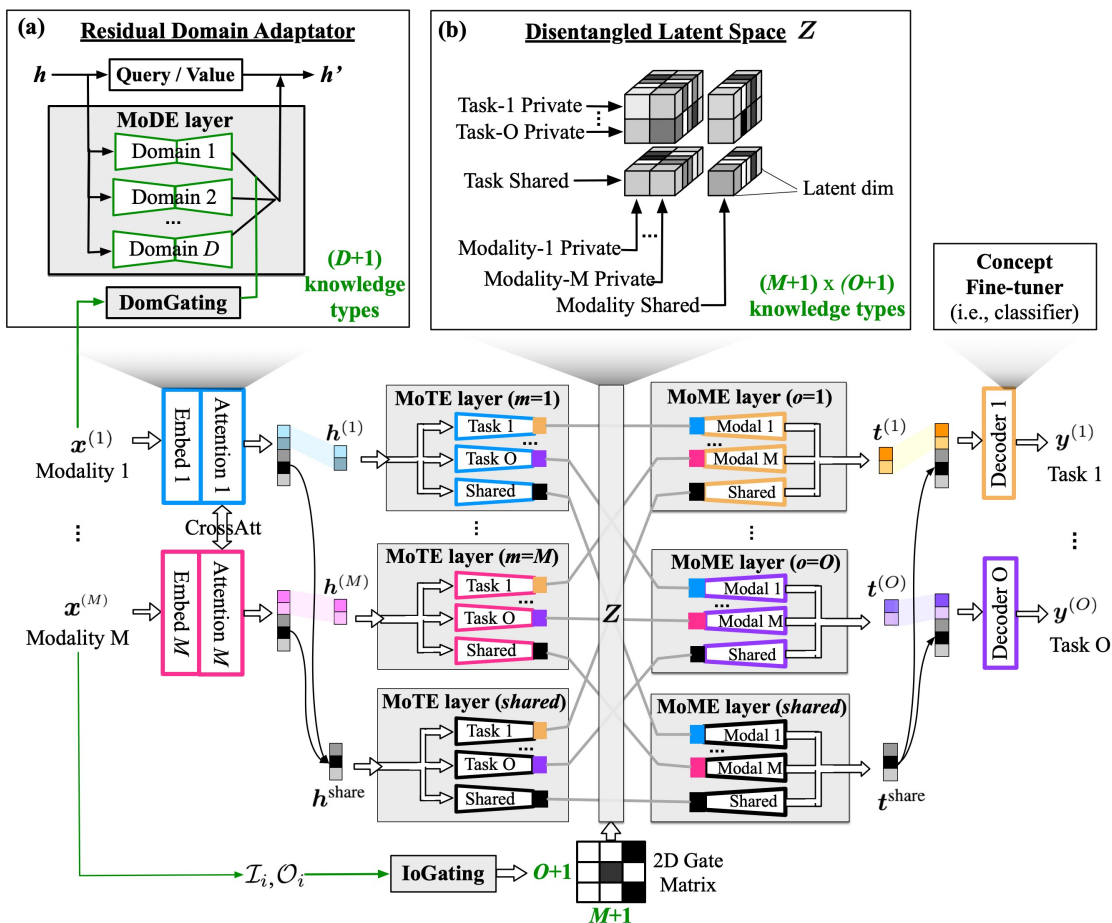


Figure 7.5.3: The supernet  $\theta^{\text{sup}}$  at the server.

over the  $N$  agents: **(1)**  $M(D + 1)$  knowledge types related to domain shift of each modality; each domain  $d \in [D]$  consists of a domain-specific and a domain-agnostic knowledge. **(2)** At most  $O(N + 1)$  knowledge types related to concept drift regarding individual fine-tuning on the decoder. **(3)**  $(M + 1)(O + 1)$  knowledge types related to modality and task gaps, including the task-specific and task-shared knowledge per modality; the modality-specific and modality-shared knowledge per task type; and the knowledge shared by all tasks and all modalities, such as the commonsense cognition.

### 7.5.2.1 Mixture-of-Knowledge-Expert Supernet at Server

In order to learn the  $K$  global fine-grained knowledge types mentioned above, we design a *wide and deep* supernet  $\theta^{\text{sup}}$  stored at the central server, whose

overview is shown in Figure 7.5.3. The mission of  $\theta^{\text{sup}}$  is to accommodate  $K$  knowledge types. To this end, we design  $\theta^{\text{sup}}$  as follows. Basically,  $\theta^{\text{sup}}$  is a Multi-input Multi-head Transformer-like architecture, consisting of  $M$  input channels for all seen modalities over agents and  $O$  output channels for all seen tasks over agents, where the interaction between modalities can be captured by cross-attention mechanism. On the basis of that, we introduce several modules and operations to learn fine-grained knowledge types from the model to further disentangling the asymmetric  $\mathcal{R}_{\text{IE}}(\cdot)$ ,  $\mathcal{R}_{\text{ID}}(\cdot)$ ,  $\mathcal{R}_{\text{XE}}(\cdot, \cdot)$  and  $\mathcal{R}_{\text{XD}}(\cdot, \cdot)$ . Specifically, we adopt the idea of Mixture of Experts and replace several layers in the Transformer with Mixture of Domain Experts (**MoDE**) layers, Mixture of Task Experts (**MoTE**) layers, or Mixture of Modality Experts (**MoME**) layers. The main ideas are as follows.

- We propose the **MoDE** layer (denoted as  $\theta_{\text{mode}}$ ) to capture  $D$  **domain-specific** knowledge types. It consists of  $D$  parallel expert models, where each expert model (denoted as  $\text{MoDE}_d(\cdot; \theta_{\text{mode}}^d)$ ) in MoDE stands for the knowledge type for a specific domain  $d$ . MoDE acts as a residual connection attached to an original model block, which we treat as the **domain-agnostic** knowledge (as show in Figure 7.5.3(a)). In practice, we apply MoDE layers to the query and value linear layers.
- Suppose  $\theta_{\text{left}}^{(m)}$  denotes a combination of modality- $m$ 's embedding layers, attention layers, and MoDE layers (as shown as the colored-contour rectangles on the left of Figure 7.5.3). Let  $\mathcal{H}^{(m)}$  denote the feature space learned by  $\theta_{\text{left}}^{(m)}$  (after necessary cross-attention multimodal interaction).  $\mathcal{H}^{(m)}$  contains the modality- $m$  specific information. It is straightforward different modalities' output space  $\mathcal{H}^{(m)} \neq \mathcal{H}^{(m')}$  contains complementary and heterogeneous information that describe different aspects/views of any object. However, in order to maximize positive transfer between different modalities that may contain common knowledge, especially in the asymmetric  $\mathcal{R}_{\text{XE}}(\cdot)$  (cross-modal multi-task) problem, the modality-shared information should be learned as well. Therefore, we follow [241] and split the modality- $m$ 's output feature space as  $\mathcal{H}^{(m)}$  into two types of information:  $[\mathbf{h}^{\text{share}} || \mathbf{h}^{(m)}] = \mathbf{h}'^{(m)} \in \mathcal{H}^{(m)}$ , where  $||$  denotes concatenation operation,  $\mathbf{h}^{(m)} \in \mathbb{R}^{d_m^{\text{modal}}}$  represents the **modality-**

**private knowledge** that is not shared with the other modalities and  $\mathbf{h}^{\text{share}} \in \mathbb{R}^{d_{\text{share}}^{\text{modal}}}$  represents the **modality-shared knowledge** type.

- Let  $\mathcal{T}^{(o)}$  denote the input feature space of the task  $o$ 's decoder network (denoted as  $\boldsymbol{\theta}_{\text{right}}^{(o)}$ ).  $\mathcal{T}^{(o)}$  contains the information that specifically used for prediction or decision making for the downstream task  $o$ . It is straightforward different tasks' input spaces  $\mathcal{T}^{(o)} \neq \mathcal{T}^{(o')}$  contains diversified information; for example, the information for classification should be different from that used for segmentation. However, in order to maximize positive transfer between different tasks that may contain common knowledge, especially in the asymmetric  $\mathcal{R}_{\text{XD}}(\cdot)$  (multi-modal cross-task) problem, it is beneficial to leverage any task-shared information as well. Therefore, we assume  $\mathcal{T}^{(o)}$  is a fused space by combining a task-private and a task-shared feature space,  $[\mathbf{t}^{\text{share}} \parallel \mathbf{t}^{(o)}] = \mathbf{t}'^{(o)} \in \mathcal{T}^{(o)}$ . where  $\mathbf{t}^{(o)} \in \mathbb{R}^{d_o^{\text{task}}}$  represents the **task-private knowledge** that is not shared with the other tasks and  $\mathbf{t}^{\text{share}} \in \mathbb{R}^{d_{\text{share}}^{\text{task}}}$  represents the **task-shared knowledge**.
- The asymmetrical problems—the single-modality *multi-task*  $\mathcal{R}_{\text{IE}}$  and the *cross-modality multi-task*  $\mathcal{R}_{\text{XE}}$ , both seek how one modality-level knowledge type (i.e., modality-private or modality-shared) can serve diverse tasks. The reason that may cause asymmetrical knowledge relationships in  $\mathcal{R}_{\text{IE}}$  or  $\mathcal{R}_{\text{XE}}$  is the downstream task identity—those agents that share more downstream tasks should transfer more knowledge than other pairs with less/no common downstream task. To this end, we propose the **MoTE** layer (denoted as  $\boldsymbol{\theta}_{\text{mote}}$ ) to capture  $(O+1)$  **task-related knowledge type for each types of modality-private/shared knowledge**. In the server's supernet, there are a total  $(M+1)$  MoTE layers:  $\boldsymbol{\theta}_{\text{mote}}^{(1)}, \boldsymbol{\theta}_{\text{mote}}^{(2)}, \dots, \boldsymbol{\theta}_{\text{mote}}^{(M)}, \boldsymbol{\theta}_{\text{mote}}^{(\text{share})}$ , where  $\boldsymbol{\theta}_{\text{mote}}^{(m)}$  denote the MoTE layer for modality- $m$ 's private knowledge and  $\boldsymbol{\theta}_{\text{mote}}^{(\text{share})}$  denote the MoTE layer for modality-shared knowledge. The MoTE layer of each modality  $m$  contains  $(O+1)$  expert models  $\boldsymbol{\theta}_{\text{mote}}^{(m)} = \{\boldsymbol{\theta}_{\text{mote}}^{(m)1}, \dots, \boldsymbol{\theta}_{\text{mote}}^{(m)O}, \boldsymbol{\theta}_{\text{mote}}^{(m)\text{share}}\}$ . As shown in Figure 7.5.3(b), the output of all MoTE layers of the server network can be represented as a tensor  $\mathbf{Z} \in \mathbb{R}^{(M+1) \times (O+1) \times d^{\text{latent}}}$  consists of  $(M+1)(O+1)$  features from each of the expert models, where  $d^{\text{latent}}$  is the

feature dimensional of each knowledge type.

- The asymmetrical problems—the *multi-modality* single-task  $\mathcal{R}_{\text{ID}}$  and the *multi-modality cross-task*  $\mathcal{R}_{\text{XD}}$ , both seek how one task-level knowledge type (i.e., task-private or task-shared) can be learned from diverse tasks. The reason that may cause asymmetrical knowledge relationships in  $\mathcal{R}_{\text{ID}}$  or  $\mathcal{R}_{\text{XD}}$  is the incoming modality identity—those agents that share more modality types should transfer more knowledge than other pairs with less/no common modalities types. To this end, We propose the **MoME** layer (denoted as  $\boldsymbol{\theta}_{\text{mome}}$ ) to capture  $(M+1)$  **modality-related knowledge types for each type of task-private/shared knowledge**. Similar to MoTE, we denote the MoME layer for task- $o$ 's private knowledge as  $\boldsymbol{\theta}_{\text{mome}}^{(o)} = \{\boldsymbol{\theta}_{\text{mome}}^{(o)1}, \boldsymbol{\theta}_{\text{mome}}^{(o)2}, \dots, \boldsymbol{\theta}_{\text{mome}}^{(o)M}, \boldsymbol{\theta}_{\text{mome}}^{(o)\text{share}}\}$ . The order of inputs feed to all the MoME layers of the server network is the transposed  $\mathbf{Z}^\top \in \mathbb{R}^{(O+1) \times (M+1) \times d^{\text{latent}}}$ .

### 7.5.2.2 Disentanglement Losses

Disentanglement of  $\mathbf{Z}$  is important for purifying and separating the semantics of knowledge transfer. To encourage this, we introduce auxiliary losses to the local objective. Many advanced disentanglement techniques can be applied here [241]. (1) First, we incorporate an auxiliary loss added to local objective—the *alignment regularization loss* between the shared feature learned by each modality,  $f_i^{\text{align}}(\boldsymbol{\theta}_{\text{left},i}^{(m)}, m \in \mathcal{I}_i) := \sum_{m,m' \in \mathcal{I}_i} \|\mathbf{h}_{:d_{\text{share}}^{\text{modal}}}^{(m)} - \mathbf{h}_{:d_{\text{share}}^{\text{modal}}}^{(m')}\|_2^2$ . (2) Second, in order to explicitly enforce the latent space to be as disentangled as designed above, we propose the *orthogonal regularization loss* between each knowledge types

$$f_i^{\text{orth}}(\boldsymbol{\theta}_{\text{left},i}^{(m)}, \boldsymbol{\theta}_{\text{mote},i}^{(m)o} | m \in \mathcal{I}_i, o \in \mathcal{O}_i) := \sum_{(m,o),(m',o') \in \mathcal{I}_i \times \mathcal{O}_i} \mathbf{Z}_{o,m}^\top \cdot \mathbf{Z}_{o',m'} \quad (7.6)$$

Without an accurate disentangled latent space, there would be false positive knowledge transfer and false negative knowledge transfer. Only an accurate disentangled space together with the designed parameter sharing strategy will encourage positive transfer and avoid negative transfer.

### 7.5.2.3 Proof of Symmetrical PKT After Disentanglement

We prove that the proposed two-stage disentanglement can successfully decompose the original asymmetric agent relationships  $\mathcal{R}(\boldsymbol{\theta}_{1:N})$  into  $K = M(D + 1) + O(N + 1) + (M + 1)(O + 1)$  independent symmetric agent relationships. Detailed proof is provided in Section 7.6.

### 7.5.3 Sparsely-gated Collaboration via Network Routing

Each agent’s local network  $\boldsymbol{\theta}_i$  is a sparsely-activated version of  $\boldsymbol{\theta}^{\text{sup}}$ . Formally, each agent has two auxiliary **gating functions**: **(1) IoGate**( $\cdot$ ) takes the input samples or the modality-task indicators  $\mathcal{I}_i, \mathcal{O}_i$  and outputs a binary gate matrix  $\mathbf{S}_i \in \{0, 1\}^{(M+1) \times (O+1)}$ , where each entry  $\mathbf{S}_{i,m,o} = 1$  if  $(m \in \mathcal{I}_i \wedge o \in \mathcal{O}_i) \vee (m \in \mathcal{I}_i \wedge o = O + 1) \vee (m = M + 1 \wedge o \in \mathcal{O}_i) \vee (m = M + 1 \wedge o = O + 1)$ ; otherwise,  $\mathbf{S}_{i,m,o} = 0$ . The  $\mathbf{S}_{i,M+1,O+1}$  always equals to one because any agent with any modality-task pair learn the task-shared and modality shared knowledge, which bridge the gap between a pair of agents with  $\mathcal{I}_i \cap \mathcal{I}_{i'} = \emptyset \wedge \mathcal{O}_{i'} \cap \mathcal{O}_i = \emptyset$ . **(2) DomGate**( $\cdot; \phi$ ) takes the input samples and produces a  $D$ -dimensional binary vector  $\mathbf{g}_i \in \{0, 1\}^D$ , where  $D \leq N$  denotes the pre-defined number of domains over agents and  $\phi$  is the parameters of the function.

The binary outputs of the two gating functions  $\mathbf{S}_i, \mathbf{g}_i$  are used to **route** each agent’s network through the supernet  $\boldsymbol{\theta}_i = \text{ROUTE}(\mathbf{S}_i, \mathbf{g}_i; \boldsymbol{\theta}^{\text{sup}})$ . The details of module routing are as follows.

- MoDE layers are gated by the one-hot vector  $\mathbf{g}_i$ . Suppose the input feature is  $\mathbf{h}$ . The output of any MoDE layer is  $\Delta \mathbf{h} = \sum_{d=1}^D \mathbf{g}_{i,d} \text{MoDE}_d(\mathbf{h}; \boldsymbol{\theta}_{\text{mode},i}^d)$ , which will be added to the original model’s output. Since  $\mathbf{g}_i$  is one hot, only one expert is activated and receives gradients from backward propagation. That is, only one **domain’s specific knowledge** is learned at each agent. The learned  $\mathbf{g}_i$  is treated as a guess of the domain identity of agent  $i$ .
- Modality encoders and task-specific decoders are gated by  $\mathcal{I}_i$  and  $\mathcal{O}_i$ , respectively. The agent  $i$ ’s network contains encoders  $\{\boldsymbol{\theta}_{\text{left}}^{(m)}\}_{m \in \mathcal{I}_i}$  and decoders  $\{\boldsymbol{\theta}_{\text{right}}^{(o)}\}_{o \in \mathcal{O}_i}$ . Accordingly, the agent learns  $|\mathcal{I}_i|$  **modality-private** knowledge

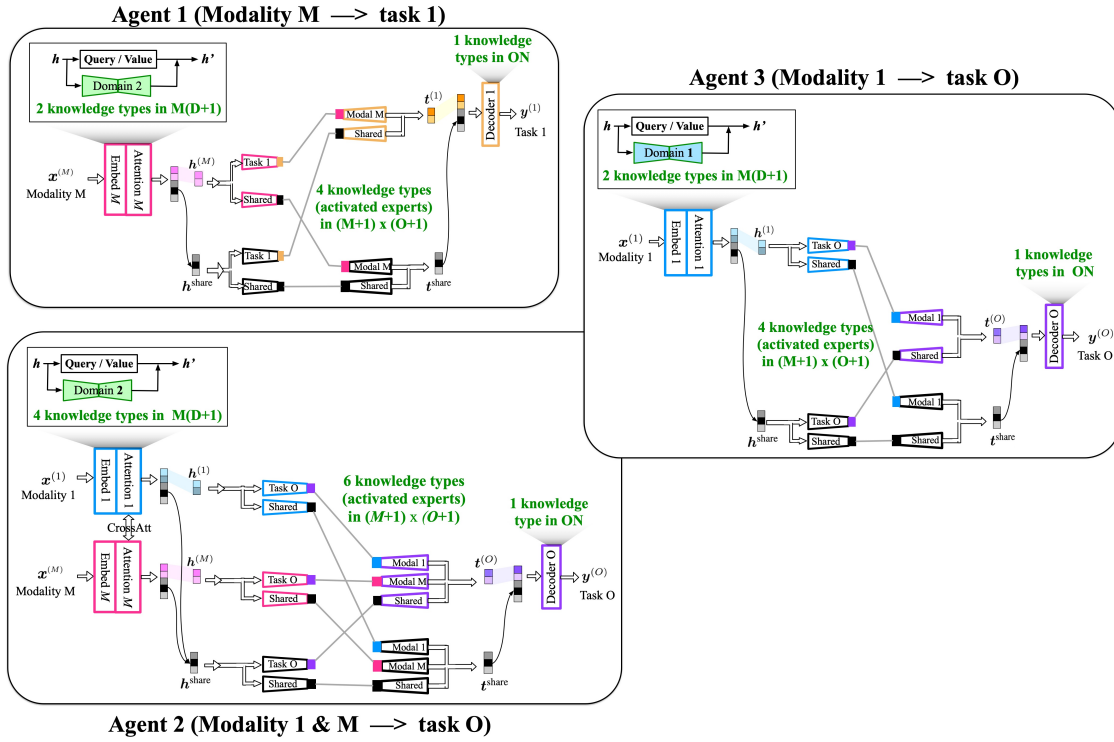


Figure 7.5.4: The routed network structures at three example agents.

types, one **modality-shared** knowledge type,  $|\mathcal{O}_i|$  **task-private** knowledge types, and one **task-shared** knowledge type.

- MoTE and MoME layers are routed by the gate matrix  $\mathbf{S}_i$ . Given a sample, the agent model learns  $(|\mathcal{I}_i| + 1) \times (|\mathcal{O}_i| + 1)$  **modality-task interactive** knowledge types, which are represented in a sparse tensor  $\mathbf{Z} \in \mathbb{R}^{(M+1) \times (O+1) \times d^{\text{latent}}}$ , where only  $(|\mathcal{I}_i| + 1) \times (|\mathcal{O}_i| + 1) \times d^{\text{latent}}$  values in this tensor is activated and requires back-propagation. Examples are shown in Figure 7.5.2(c). For example, given a modality-private feature  $\mathbf{h}^{(m)}$ , the outputs of an MoTE layer is  $|\mathcal{O}_i| + 1$  features; each expert  $o \in \mathcal{O}_i$  outputs  $\mathbf{Z}_{o,m,\cdot} = \text{MoTE}_m^o(\mathbf{h}^{(m)}; \boldsymbol{\theta}_{\text{mote},i}^{(m)o})$ . After  $\mathbf{Z}$  is learned, MoME layers reconstruct the task-private/shared knowledge from  $\mathbf{Z}$ . Specifically, the task  $o$ 's private and task-shared knowledge are decoded as  $\mathbf{t}^{(o)} = \sum_{m=1}^{M+1} \mathbf{S}_{i,o,m} \text{MoME}_o^m(\mathbf{Z}_{o,m,\cdot}; \boldsymbol{\theta}_{\text{mome},i}^{(o)m})$  and

$$\mathbf{t}^{\text{share}} = \sum_{m=1}^{M+1} \mathbf{S}_{i,o,m} \text{MoME}_{\text{share}}^m(\mathbf{Z}_{o,m,\cdot}; \boldsymbol{\theta}_{\text{mome},i}^{(\text{share})m}). \quad (7.7)$$

Through the sparse routing process, the number of knowledge types that can be disentangled from agent  $i$ 's model is  $K_i = 2|\mathcal{I}_i| + 2|\mathcal{O}_i| + (|\mathcal{I}_i| + 1)(|\mathcal{O}_i| + 1)$ . The collection of agent-specific knowledge types is a subset of globally shareable knowledge types. Figure 7.5.4 shows three example agents and their agent-specific knowledge types. In practice,  $\mathbf{S}_i, \mathbf{g}_i$  is very sparse, i.e.,  $K_i \ll K$ . Hence the agent network  $\theta_i$  is much thinner than  $\theta^{\text{sup}}$  and does not exceed the local memory constraint. Also, the number of activated MoTE and MoME experts depends on  $\mathcal{I}_i, \mathcal{O}_i$ , thus a larger  $M$  and  $O$  over the large-scale agents have no influence on the size of the local model  $\theta_i$ .

## 7.6 Details and Proofs

### 7.6.1 Global Knowledge Type Design

We aim to find the largest knowledge components that can sufficiently describe the global asymmetric PKT (Pairwise Knowledge Transfer) problem as the combination of several symmetric PKT problems.

The four sub-problems after the 1st-stage group-wise disentanglement give us the following inspirations: (1) First, to reduce the asymmetrical knowledge relationships of the subproblem-1 (TD heterogeneity), we investigate how a single modality serves different downstream tasks. Then, the problem is that, how to disentangle the modality-specific knowledge into several knowledge types, each of which serves a different task? And, how to disentangle the domain-specific knowledge from domain-agnostic knowledge? (2) To reduce the asymmetrical knowledge relationships of the subproblem-2 (MC heterogeneity), we investigate how multiple modalities serve the same task. We need to disentangle the task-specific knowledge into several modality types, each of which receive information from a different modality, as well as to disentangle the concept-specific knowledge between agents? (3) To reduce the asymmetrical knowledge relationships of the subproblem-3 (MT heterogeneity), it is necessary to *first learn the common knowledge shared by different modalities* and then *disentangle the modality-shared knowledge into several knowledge types, each of which serves a different task*. (4) To reduce the asym-



metrical knowledge relationships of the subproblem-4 (MT heterogeneity), it is necessary to *first learn the common knowledge shared by different task* and then *disentangle the task-shared knowledge into several knowledge types, each of which decodes a different modality*.

Inspired by them, we assume  $K = M(D + 1) + O(N + 1) + (M + 1)(O + 1)$  fine-grained knowledge types over  $N$  agents can sufficiently describe the global asymmetric PKT problem as the combination of several symmetric PKT problems. For **concept shift** (C), we simply use fine-tuning for the last layer of the decoder. The last layer each decoder is considered as one knowledge type. We assume the early layers except the last layer of each decoder  $o \in [O]$  is globally shareable by those agents that have task  $o$ . The last layer of each decoder is unique for each agent. Therefore, a total of  $O(N + 1)$  concept knowledge over  $N$  agents will be needed. For **domain shift** (D), each domain should contain a domain-specific knowledge type and, in addition, there is a single domain-agnostic based knowledge type, which results in  $M(D + 1)$  knowledge types related to domain shift of each modality. **Modality and Task shift** (MT) are correlated and more complex. for disentangling modality and task-related knowledge relationship asymmetric, we consider three levels of knowledge types as follows. First, for each modality, it can serve different tasks in different agents, and therefore, disentangling agents implies disentangling the **task-wise modality-specific knowledge**. Likewise, for each task, it can receive the information from different modalities in different agents, and therefore, disentangling agents implies disentangling the task-specific knowledge into **modality-wise task-specific knowledge**. More importantly, sub-problem-3/4 seeks any common knowledge shared by modalities and task to bridge the gap between agents that have neither similar modality nor similar tasks. Without shareable knowledge or without private knowledge types, the positive transfer is not maximized or the negative transfer can happen (see Appendix Figure 7.7.1). Therefore, it is straightforward to learn an additional **modality-shared knowledge type** and **task-shared knowledge type**. Over all the  $N$  agents with a total of  $O$  tasks, there would be  $O + 1$  independent types of knowledge: the **task-shared knowledge** and **task-private knowledge** per downstream task type. In summary, we need  $(M + 1)(O + 1)$  knowledge types for disentangling MT

heterogeneity.

## 7.6.2 Proofs

We here prove that our fine-grained knowledge disentanglement can therefore decompose the *original asymmetric agent relationships* into  $K = M(D + 1) + O(N + 1) + (M + 1)(O + 1)$  independent *symmetric agent relationships*, that is

$$\begin{aligned}
& \mathcal{R}(\boldsymbol{\theta}_i | i \in [N]) \\
&= \sum_{m \in [M]} \mathcal{R}_{\text{IE}}^m(\mathcal{G}_{\text{enc}}^{(m)}) + \sum_{o \in [O]} \mathcal{R}_{\text{ID}}^o(\mathcal{G}_{\text{dec}}^{(o)}) \\
&\quad + \sum_{m, m' \in [M], m \neq m'} \mathcal{R}_{\text{XE}}^{m, m'}(\mathcal{G}_{\text{enc}}^{(m)}, \mathcal{G}_{\text{enc}}^{(m')}) \\
&\quad + \sum_{o, o' \in [O], o \neq o'} \mathcal{R}_{\text{XD}}^{o, o'}(\mathcal{G}_{\text{dec}}^{(o)}, \mathcal{G}_{\text{dec}}^{(o')}) \\
&= \sum_{k=1}^K \mathcal{R}_k(\boldsymbol{\theta}_i^{(k)} | i \in C_k \subseteq [N]).
\end{aligned} \tag{7.8}$$

where  $\mathcal{G}_{\text{enc}}^{(m)} = \{\boldsymbol{\theta}_{\text{enc}, i}^{(m)} | i \in [N], m \in \mathcal{I}_i\}$  and  $\mathcal{G}_{\text{dec}}^{(o)} = \{\boldsymbol{\theta}_{\text{dec}, i}^{(o)} | i \in [N], o \in \mathcal{O}_i\}$ . The notation  $\boldsymbol{\theta}_{\text{enc}, i}^{(m)} = \{\boldsymbol{\theta}_{\text{BE}, i}^{(m)}, \boldsymbol{\theta}_{\text{mode}, i}^{(m)}, \boldsymbol{\theta}_{\text{mote}, i}^{(m)}, \boldsymbol{\theta}_{\text{mote}, i}^{(\text{share})}\}$  consists of the modality  $m$ 's encoder learned at agent  $i$  consisting of embedding, attention layers, MoDE layers, and the MoTE layers for modality- $m$ , as well as the MoTE layers for modality-shared information. The notation  $\boldsymbol{\theta}_{\text{dec}, i}^{(o)} = \{\boldsymbol{\theta}_{\text{mome}, i}^{(o)}, \boldsymbol{\theta}_{\text{mome}, i}^{(\text{share})}, \boldsymbol{\theta}_{\text{BD}, i}^{(o)}, \boldsymbol{\theta}_{\text{final}, i}^{(o)}\}$  consists of the task- $o$  MoME layers, the MoME layers for task-shared information, and the task  $o$ 's decoder consisting of early layers and the final concept layers, which are learned at agent  $i$ . The mixed knowledge captured in  $\boldsymbol{\theta}_i = \{\boldsymbol{\theta}_{\text{enc}, i}^{(m)} | m \in \mathcal{I}_i\} \cup \{\boldsymbol{\theta}_{\text{dec}, i}^{(o)} | o \in \mathcal{O}_i\}$ .

### 7.6.2.1 Knowledge Decomposition

We define  $(M + O + MO)$  subsets of agents: (1) For each modality type  $m$ , we define a subset of agents that contain the modality  $m$ , that is,  $C_{\text{modal}}^m = \{i | i \in [N], m \in \mathcal{I}_i\}$ ; (2) For each downstream task type  $o$ , we define a subset of agents that contain the task  $o$ , that is,  $C_{\text{task}}^o = \{i | i \in [N], o \in \mathcal{O}_i\}$ ; and, (3) For each modality-task pair  $(m, o)$ , we define a subset of agents that share both the modality  $m$  and the task  $o$ , that is,  $C_{\text{pair}}^{m, o} = \{i | i \in [N], m \in \mathcal{I}_i \wedge o \in \mathcal{O}_i\}$ .

Each asymmetric term  $\mathcal{R}_{\text{IE}}^m(\mathcal{G}_{\text{enc}}^{(m)})$  can be split into  $K_{\text{IE}}^m = (D + 1) + 2(O + 1)$

symmetric terms

$$\begin{aligned}
& \mathcal{R}_{\text{IE}}^m(\mathcal{G}_{\text{enc}}^{(m)}) \\
= & \mathcal{R}_{\text{IE}}^m\left(\boldsymbol{\theta}_{\text{enc},i}^{(m)} \mid i \in [N], m \in \mathcal{I}_i\right) \\
= & \mathcal{R}_{\text{BE}}^m\left(\boldsymbol{\theta}_{\text{BE},i}^{(m)} \mid i \in [N], m \in \mathcal{I}_i\right) \\
& + \sum_{d=1}^D \mathcal{R}_{\text{MoDE}}^{m,d}\left(\boldsymbol{\theta}_{\text{mode},i}^{(m)d} \mid i \in [N], g_{i,d} = 1\right) \\
& + \mathcal{R}_{\text{MoTE}}^{m,\text{share}}\left(\boldsymbol{\theta}_{\text{mote},i}^{(m)\text{share}} \mid i \in [N], m \in \mathcal{I}_i\right) \\
& + \sum_{o=1}^O \mathcal{R}_{\text{MoTE}}^{m,o}\left(\boldsymbol{\theta}_{\text{mote},i}^{(m)o} \mid i \in [N], m \in \mathcal{I}_i \wedge o \in \mathcal{O}_i\right) \\
& + \mathcal{R}_{\text{MoTE}}^{\text{share,share}}\left(\boldsymbol{\theta}_{\text{mote},i}^{(\text{share})\text{share}} \mid i \in [N], m \in \mathcal{I}_i\right) \\
& + \sum_{o=1}^O \mathcal{R}_{\text{MoTE}}^{\text{share},o}\left(\boldsymbol{\theta}_{\text{mote},i}^{(\text{share})o} \mid i \in [N], m \in \mathcal{I}_i \wedge o \in \mathcal{O}_i\right) \\
= & \mathcal{R}_{\text{BE}}^m\left(\boldsymbol{\theta}_{\text{BE},i}^{(m)} \mid i \in C_{\text{modal}}^m\right) \\
& + \sum_{d=1}^D \mathcal{R}_{\text{MoDE}}^{m,d}\left(\boldsymbol{\theta}_{\text{mode},i}^{(m)d} \mid i \in C_{\text{MoDE}}^d\right) \\
& + \mathcal{R}_{\text{MoTE}}^{m,\text{share}}\left(\boldsymbol{\theta}_{\text{mote},i}^{(m)\text{share}} \mid i \in C_{\text{modal}}^m\right) \\
& + \sum_{o=1}^O \mathcal{R}_{\text{MoTE}}^{m,o}\left(\boldsymbol{\theta}_{\text{mote},i}^{(m)o} \mid i \in C_{\text{pair}}^{m,o}\right) \\
& + \mathcal{R}_{\text{MoTE}}^{\text{share,share}}\left(\boldsymbol{\theta}_{\text{mote},i}^{(\text{share})\text{share}} \mid i \in C_{\text{modal}}^m\right) \\
& + \sum_{o=1}^O \mathcal{R}_{\text{MoTE}}^{\text{share},o}\left(\boldsymbol{\theta}_{\text{mote},i}^{(\text{share})o} \mid i \in C_{\text{pair}}^{m,o}\right),
\end{aligned} \tag{7.9}$$

where  $\mathcal{R}_{\text{BE}}^m(\cdot)$  denotes the information sharing over the **modality- $m$  domain-agnostic** knowledge type of a subset of agents that contain the modality  $m$ , that is,  $C_{\text{modal}}^m$ .  $\mathcal{R}_{\text{MoDE}}^d(\cdot)$  denotes the information sharing over the **domain- $d$ -specific** knowledge across a subset of agents that belong to domain  $d$ , that is,  $C_{\text{MoDE}}^d = \{i \mid i \in [N], g_{i,d} = 1\}$ .  $\mathcal{R}_{\text{MoTE}}^{m,\text{share}}(\cdot)$  denotes the information sharing over the **modality- $m$ -private and task-shared** knowledge across a subset of agents that contain the modality  $m$ , that is,  $C_{\text{modal}}^m$ .  $\mathcal{R}_{\text{MoTE}}^{m,o}(\cdot)$  denotes the information sharing over the **modality- $m$ -private and task- $o$ -private** knowledge across a subset of agents that share both the modality  $m$  and task  $o$ , that is,  $C_{\text{pair}}^{m,o}$ .  $\mathcal{R}_{\text{MoTE}}^{\text{share,share}}(\cdot)$  denotes the information sharing over the **modality-shared and task-shared** knowledge across the group  $C_{\text{modal}}^m$ .  $\mathcal{R}_{\text{MoTE}}^{\text{share},o}(\cdot)$  denotes the information sharing over the **modality-shared and task- $o$ -private** knowledge across  $C_{\text{pair}}^{m,o}$ .

Each asymmetric term  $\mathcal{R}_{\text{ID}}^o(\mathcal{G}_{\text{dec}}^{(o)})$  can be split into  $K_{\text{ID}}^o = (N + 1) + 2(M + 1)$

symmetric terms:

$$\begin{aligned}
& \mathcal{R}_{\text{ID}}^o(\mathcal{G}_{\text{dec}}^{(o)}) \\
&= \mathcal{R}_{\text{ID}}^o\left(\boldsymbol{\theta}_{\text{dec},i}^{(o)} \mid i \in [N], o \in \mathcal{O}_i\right) \\
&= \mathcal{R}_{\text{BD}}^o\left(\boldsymbol{\theta}_{\text{BD},i}^{(o)} \mid i \in [N], o \in \mathcal{O}_i\right) \\
&\quad + \sum_{i=1}^N \mathcal{R}_{\text{concept}}^{o,i}\left(\boldsymbol{\theta}_{\text{final},i}^{(o)} \mid \{i\} \text{ if } o \in \mathcal{O}_i \text{ else } \emptyset\right) \\
&\quad + \mathcal{R}_{\text{MoME}}^{\text{share},o}\left(\boldsymbol{\theta}_{\text{mome},i}^{(o)\text{share}} \mid i \in [N], o \in \mathcal{O}_i\right) \\
&\quad + \sum_{m=1}^M \mathcal{R}_{\text{MoME}}^{m,o}\left(\boldsymbol{\theta}_{\text{mome},i}^{(o)m} \mid i \in [N], m \in \mathcal{I}_i \wedge o \in \mathcal{O}_i\right) \\
&\quad + \mathcal{R}_{\text{MoME}}^{\text{share,share}}\left(\boldsymbol{\theta}_{\text{mome},i}^{(\text{share})\text{share}} \mid i \in [N], o \in \mathcal{O}_i\right) \\
&\quad + \sum_{m=1}^M \mathcal{R}_{\text{MoME}}^{m,\text{share}}\left(\boldsymbol{\theta}_{\text{mome},i}^{(\text{share})m} \mid i \in [N], m \in \mathcal{I}_i \wedge o \in \mathcal{O}_i\right) \\
&= \mathcal{R}_{\text{BD}}^o\left(\boldsymbol{\theta}_{\text{BD},i}^{(o)} \mid i \in C_{\text{task}}^o\right) \\
&\quad + \sum_{i=1}^N \mathcal{R}_{\text{concept}}^{o,i}\left(\boldsymbol{\theta}_{\text{final},i}^{(o)} \mid \{i\} \text{ if } o \in \mathcal{O}_i \text{ else } \emptyset\right) \\
&\quad + \mathcal{R}_{\text{MoME}}^{\text{share},o}\left(\boldsymbol{\theta}_{\text{mome},i}^{(o)\text{share}} \mid i \in C_{\text{task}}^o\right) \\
&\quad + \sum_{m=1}^M \mathcal{R}_{\text{MoME}}^{o,m}\left(\boldsymbol{\theta}_{\text{mome},i}^{(o)m} \mid i \in C_{\text{pair}}^{m,o}\right) \\
&\quad + \mathcal{R}_{\text{MoME}}^{\text{share,share}}\left(\boldsymbol{\theta}_{\text{mome},i}^{(\text{share})\text{share}} \mid i \in C_{\text{task}}^o\right) \\
&\quad + \sum_{m=1}^M \mathcal{R}_{\text{MoME}}^{\text{share},m}\left(\boldsymbol{\theta}_{\text{mome},i}^{(\text{share})m} \mid i \in C_{\text{pair}}^{m,o}\right),
\end{aligned} \tag{7.10}$$

where  $\mathcal{R}_{\text{BD}}^o(\cdot)$  denotes the information sharing over the **task- $o$  label-agnostic** knowledge type of a subset of agents that contain the task  $o$ , that is,  $C_{\text{task}}^o$ .  $\mathcal{R}_{\text{concept}}^{o,i}(\cdot)$  denotes the **task- $o$ 's concept that is specific on the agent  $i$** .  $\mathcal{R}_{\text{MoME}}^{\text{share},o}(\cdot)$  denotes the information sharing over the **task- $o$ -private and modality-shared** knowledge across a subset of agents that contain the task  $o$ , that is,  $C_{\text{task}}^o$ .  $\mathcal{R}_{\text{MoME}}^{o,m}(\cdot)$  denotes the information sharing over the **task- $o$ -private and modality- $m$ -private** knowledge across a subset of agents that share both the modality  $m$  and task  $o$ , that is,  $C_{\text{pair}}^{m,o}$ .  $\mathcal{R}_{\text{MoME}}^{\text{share,share}}(\cdot)$  denotes the information sharing over the **task-shared and modality-shared** knowledge across the group.  $\mathcal{R}_{\text{MoME}}^{\text{share},m}(\cdot)$  denotes the information sharing over the **task-shared and modality- $m$ -private** knowledge across a subset of agents that share both the modality  $m$  and task  $o$ , that is,  $C_{\text{pair}}^{m,o}$ .

Each cross-group asymmetric term  $\mathcal{R}_{\text{XE}}^{m,m'}(\mathcal{G}_{\text{enc}}^{(m)}, \mathcal{G}_{\text{enc}}^{(m')})$  with  $m \neq m'$  can be split

into  $K_{\text{XE}}^{m,m'} = O + 1$  symmetric terms

$$\begin{aligned}
& \mathcal{R}_{\text{XE}}^{m,m'}(\mathcal{G}_{\text{enc}}^{(m)}, \mathcal{G}_{\text{enc}}^{(m')}) \\
&= \mathcal{R}_{\text{XE}}^{m,m'}\left(\{\boldsymbol{\theta}_{\text{enc},i}^{(m)} | i \in [N], m \in \mathcal{I}_i\}, \{\boldsymbol{\theta}_{\text{enc},i}^{(m')} | i \in [N], m' \in \mathcal{I}_i\}\right) \\
&= \mathcal{R}_{\text{MoTE}}^{\text{share,share}}\left(\boldsymbol{\theta}_{\text{mote},i}^{(\text{share})\text{share}}, \boldsymbol{\theta}_{\text{mote},i'}^{(\text{share})\text{share}} | \forall(i, i') \in C_{\text{modal}}^m \times C_{\text{modal}}^{m'}\right) \\
&\quad + \sum_{o=1}^O \mathcal{R}_{\text{MoTE}}^{\text{share},o}\left(\boldsymbol{\theta}_{\text{mote},i}^{(\text{share})o}, \boldsymbol{\theta}_{\text{mote},i'}^{(\text{share})o} | \forall(i, i') \in C_{\text{pair}}^{m,o} \times C_{\text{pair}}^{m',o}\right)
\end{aligned} \tag{7.11}$$

where  $\mathcal{R}_{\text{MoTE}}^{\text{share,share}}(\cdot)$  denotes the information sharing over the **task-shared and modality-shared** knowledge across **all** pairs of agents from separate modality groups, and  $\mathcal{R}_{\text{MoTE}}^{\text{share},o}(\cdot)$  denotes the information sharing over the **modality-shared and task-o-private** knowledge across **all** pairs of agents from separate modality groups as well as sharing the task  $o$ .

Each cross-group asymmetric term  $\mathcal{R}_{\text{XD}}^{o,o'}(\mathcal{G}_{\text{dec}}^{(o)}, \mathcal{G}_{\text{dec}}^{(o')})$  with  $o \neq o'$  can be split into  $K_{\text{XD}}^{o,o'} = M + 1$  symmetric schemes

$$\begin{aligned}
& \mathcal{R}_{\text{XD}}^{o,o'}(\mathcal{G}_{\text{dec}}^{(o)}, \mathcal{G}_{\text{dec}}^{(o')}) \\
&= \mathcal{R}_{\text{XD}}^{o,o'}\left(\{\boldsymbol{\theta}_{\text{dec},i}^{(o)} | i \in [N], o \in \mathcal{O}_i\}, \{\boldsymbol{\theta}_{\text{dec},i}^{(o')} | i \in [N], o' \in \mathcal{O}_i\}\right) \\
&= \mathcal{R}_{\text{MoME}}^{\text{share,share}}\left(\boldsymbol{\theta}_{\text{mome},i}^{(\text{share})\text{share}}, \boldsymbol{\theta}_{\text{mome},i'}^{(\text{share})\text{share}} | \forall(i, i') \in C_{\text{task}}^o \times C_{\text{task}}^{o'}\right) \\
&\quad + \sum_{m=1}^M \mathcal{R}_{\text{MoME}}^{\text{share},m}\left(\boldsymbol{\theta}_{\text{mome},i}^{(\text{share})m}, \boldsymbol{\theta}_{\text{mome},i'}^{(\text{share})m} | \forall(i, i') \in C_{\text{pair}}^{m,o} \times C_{\text{pair}}^{m,o'}\right)
\end{aligned} \tag{7.12}$$

where  $\mathcal{R}_{\text{MoME}}^{\text{share,share}}(\cdot)$  denotes the information sharing over the **task-shared and modality-shared** knowledge across **all** pairs of agents from separate task groups, and  $\mathcal{R}_{\text{MoME}}^{\text{share},m}(\cdot)$  denotes the information sharing over the **task-shared and modality-m-private** knowledge across **all** pairs of agents from separate task groups as well as sharing the same input modality  $m$ .

### 7.6.2.2 Knowledge Independence

Considering Eq.(7.9) for each of  $M$  modalities, the terms  $\mathcal{R}_{\text{MoME}}^{\text{share},1}, \mathcal{R}_{\text{MoME}}^{\text{share},2}, \dots, \mathcal{R}_{\text{MoME}}^{\text{share},O}, \mathcal{R}_{\text{MoME}}^{m,1}, \mathcal{R}_{\text{MoME}}^{m,2}, \dots, \mathcal{R}_{\text{MoME}}^{m,O}, \mathcal{R}_{\text{MoME}}^{\text{share,share}}$ , and  $\mathcal{R}_{\text{MoME}}^{m,\text{share}}$  are encouraged to be independent with each other through the auxiliary orthogonal loss over the latent space  $\mathcal{Z}$ . When adding up Eq.(7.9) over all modalities, the modality-specific terms in different modalities  $\mathcal{R}_{\text{MoME}}^{1,o}, \mathcal{R}_{\text{MoME}}^{2,o}, \dots, \mathcal{R}_{\text{MoME}}^{M,o}$  are independent due to the disentanglement of the latent space; however, the  $(O + 1)$  modality-shared terms in different modalities,  $\mathcal{R}_{\text{MoME}}^{\text{share},o}$  and  $\mathcal{R}_{\text{MoME}}^{\text{share,share}}$  are not independent and can be combined. The independence between domain-agnostic and domain-specific knowledge

types can be done as first-order meta learning, where domain-agnostic parameters are treated as the meta-parameters and domain-specific parameters are treated as the inner-loop tunable parameters. Overall, by adding up Eq.(7.9) over all modalities, we end up with  $K_{\text{IE}}$  independent symmetric information sharing terms  $K_{\text{IE}} = (O + 1) + \sum_{m=1}^M [K_{\text{IE}}^m - (O + 1)] = M(D + 1) + (M + 1)(O + 1)$ . Likewise, by adding up Eq.(7.10) over all task types, we end up with  $K_{\text{ID}}$  independent symmetric information sharing terms  $K_{\text{ID}} = (M + 1) + \sum_{o=1}^O [K_{\text{ID}}^o - (M + 1)] = O(N + 1) + (O + 1)(M + 1)$ .

In Eq.(7.11), the terms  $\mathcal{R}_{\text{MoME}}^{\text{share,share}}$  and  $\mathcal{R}_{\text{MoME}}^{\text{share},o}$ ,  $o = 1, 2, \dots, O$  are shared over the  $M(M - 1)/2$  pairs of modalities. By adding up all pairs, we end up with  $K_{\text{XE}}$  independent symmetric information sharing terms  $K_{\text{XE}} = \sum_{m,m' \in [M], m \neq m'} K_{\text{XE}}^{m,m'} / (M(M - 1)/2) = O + 1$ . Likewise, by adding up Eq.(7.12) over all task types, we end up with  $K_{\text{XD}}$  independent symmetric information sharing terms  $K_{\text{XD}} = \sum_{o,o' \in [O], o \neq o'} K_{\text{XD}}^{o,o'} / (O(O - 1)/2) = M + 1$ .

Moreover, Eq.(7.9-7.12) have the following correlations: the term  $\mathcal{R}_{\text{MoTE}}^{\text{share,share}}$  is related to  $\mathcal{R}_{\text{MoME}}^{\text{share,share}}$ , the term  $\mathcal{R}_{\text{MoTE}}^{\text{share},o}$  is related to  $\mathcal{R}_{\text{MoME}}^{\text{share},o}$ , the term  $\mathcal{R}_{\text{MoTE}}^{m,\text{share}}$  is related to  $\mathcal{R}_{\text{MoME}}^{m,\text{share}}$ , and the term  $\mathcal{R}_{\text{MoTE}}^{m,o}$  is related to  $\mathcal{R}_{\text{MoME}}^{m,o}$ .

Finally, we end up with the following independent terms

$$\begin{aligned} K &= K_{\text{IE}} + K_{\text{ID}} + K_{\text{XE}} + K_{\text{XD}} - 3(M + 1)(O + 1) \\ &= M(D + 1) + O(N + 1) + (M + 1)(O + 1) \end{aligned} \quad (7.13)$$

$$\begin{aligned} &\mathcal{R}(\boldsymbol{\theta}_i | i \in [N]) \\ &= \sum_{m \in [M]} \mathcal{R}_{\text{BE}}^m \left( \boldsymbol{\theta}_{\text{BE},i}^{(m)} | i \in C_{\text{modal}}^m \right) \\ &\quad + \sum_{o \in [O]} \mathcal{R}_{\text{BD}}^o \left( \boldsymbol{\theta}_{\text{BD},i}^{(o)} | i \in C_{\text{task}}^o \right) \\ &\quad + \sum_{m \in [M]} \sum_{d=1}^D \mathcal{R}_{\text{MoDE}}^{m,d} \left( \boldsymbol{\theta}_{\text{mode},i}^{(m)d} | i \in C_{\text{MoDE}}^d \right) \\ &\quad + \sum_{o \in [O]} \sum_{i=1}^N \mathcal{R}_{\text{concept}}^{o,i} \left( \boldsymbol{\theta}_{\text{final},i}^{(o)} | \{i\} \right) \\ &\quad + \mathcal{R}_{\text{MoTE} \& \text{MoTE}}^{\text{share,share}} \left( \{ \boldsymbol{\theta}_{\text{mote},i}^{(\text{share})\text{share}}, \boldsymbol{\theta}_{\text{mome},i}^{(\text{share})\text{share}} \} | i \in [N] \right) \\ &\quad + \sum_{m \in [M]} \left[ \mathcal{R}_{\text{MoTE} \& \text{MoTE}}^{m,\text{share}} \left( \{ \boldsymbol{\theta}_{\text{mote},i}^{(m)\text{share}}, \boldsymbol{\theta}_{\text{mote},i}^{(\text{share})m} \} | i \in C_{\text{modal}}^m \right) \right] \\ &\quad + \sum_{m \in [M]} \sum_{o \in [O]} \left[ \mathcal{R}_{\text{MoTE} \& \text{MoTE}}^{m,o} \left( \{ \boldsymbol{\theta}_{\text{mote},i}^{(m)o}, \boldsymbol{\theta}_{\text{mome},i}^{(o)m} \} | i \in C_{\text{pair}}^{m,o} \right) \right] \\ &\quad + \sum_{o \in [O]} \left[ \mathcal{R}_{\text{MoTE} \& \text{MoTE}}^{m,\text{share}} \left( \{ \boldsymbol{\theta}_{\text{mote},i}^{(\text{share})o}, \boldsymbol{\theta}_{\text{mome},i}^{(o)\text{share}} \} | i \in C_{\text{task}}^o \right) \right] \end{aligned} \quad (7.14)$$

**Case Study** ( $M = O = 2$ ). For simplicity, we have assumed a single shared-

knowledge space can be shared by all pairs of modalities. If there are only two modalities and two tasks, the single-shared knowledge assumption is true, and we have  $K = M \cdot K_{\text{IE}} + O \cdot K_{\text{ID}} + M(M-1)K_{\text{XE}}/2 + O(O-1)K_{\text{XD}}/2 = M \cdot K_{\text{IE}} + O \cdot K_{\text{ID}} + K_{\text{XE}} + K_{\text{XD}}$ .

### 7.6.3 Pseudocode and Workflow

The training workflow and pseudo-code of DisentAFL is provided in Algorithm 4.

---

#### Algorithm 4: DisentAFL Workflow

---

- 1: **Input:** Total number of agents  $N$ ; total number of modalities  $M$ ; total number of tasks  $O$ ; total communication rounds  $T$ ; each agent's dataset  $\mathcal{D}_i$ , sensor sets  $\mathcal{I}_i$ , and task types  $\mathcal{O}_i$ ,  $\forall i \in [N]$ .
  - 2: **Initialization:** Randomly initialize supernet at server  $\theta_0^{\text{sup}}$  and gating function  $\phi_0$ ; compute and fix each agent's MoME/MoTE gate matrix  $\mathbf{S}_i = \text{IoGate}(\mathcal{I}_i, \mathcal{O}_i)$ ; each agent's memory buffer storing validation accuracy during policy search  $\mathcal{B}_i = \{\}$ .
  - 3: **for** round  $t = 1$  to  $T$  **do**
  - 4:   Sampling a subset of agents  $V_t \subset [N]$  with balanced load of MoME/MoTE experts.
  - 5:   // local SGD independently
  - 6:   **for** agent  $i \in V_t$  in parallel **do**
  - 7:     // Expert activation
  - 8:     Download the current policy  $\phi_{i,t} \leftarrow \phi_t$ .
  - 9:     Sample an MoDE expert with  $\epsilon$ -greedy:  $\tilde{\mathbf{g}}_{i,t} = \text{DomGate}(\mathcal{D}_i; \phi_{i,t}) + \epsilon$ .
  - 10:     Query the current server model  $\theta_t^{\text{sup}}$  using  $\tilde{\mathbf{g}}_{i,t}$  and  $\mathbf{S}_{i,t}$ .
  - 11:     Receive the sparsely-gated model from server  $\theta_{i,t} \leftarrow \text{ROUTE}(\mathbf{S}_{i,t}, \tilde{\mathbf{g}}_{i,t}; \theta_t^{\text{sup}})$
  - 12:     // Local Training
  - 13:      $\tilde{\theta}_{i,t} = \theta_{i,t}$
  - 14:     **for** each local update step **do**
  - 15:       Sample a batch of data  $\mathcal{D}_i^b$  from local training dataset  $\mathcal{D}_i$
  - 16:       Local update  $\tilde{\theta}_{i,t} \leftarrow \tilde{\theta}_{i,t} - \alpha \nabla_{\tilde{\theta}_{i,t}} f_i(\tilde{\theta}_{i,t}; \mathcal{D}_i^b)$ .
  - 17:     **end for**
  - 18:      $\theta_{i,t+\frac{1}{2}} = \tilde{\theta}_{i,t}$
  - 19:     Evaluate the current sampled expert using local validation dataset and  $\theta_{i,t+\frac{1}{2}}$
  - 20:     Add  $(\tilde{\mathbf{g}}_{i,t}, \text{ACC}_i^d)$  to memory buffer  $\mathcal{B}_i$
  - 21:     Update the policy  $\phi_{t+\frac{1}{2}} = \phi_t - \gamma \nabla_{\phi_t} Q$  using  $\mathcal{B}_i$  such that the best expert is produced.
  - 22:     Compute the best MoDE expert  $\mathbf{g}_{i,t+\frac{1}{2}} = \text{DomGate}(\mathcal{D}_i; \phi_{i,t+\frac{1}{2}})$ .
  - 23:     Upload  $\theta_{i,t+\frac{1}{2}}, \phi_{t+\frac{1}{2}}, \mathbf{g}_{i,t+\frac{1}{2}}$  to server.
  - 24:   **end for**
  - 25:   // Knowledge aggregation
  - 26:   Update  $\phi_{t+1} = \frac{1}{|V_t|} \sum_{i \in V_t} \phi_{t+\frac{1}{2}}$
  - 27:   **for** knowledge type  $k = 1$  to  $K$  in parallel **do**
  - 28:     Aggregate knowledge type  $k$ 's corresponding parameters,  $\theta_{t+1}^{\text{sup}(k)}$ , from  $\theta_{i,t+\frac{1}{2}}^{(k)}, i \in V_t$ .
  - 29:   **end for**
  - 30: **end for**
  - 31: **Output:** Super-network at server  $\theta_T^{\text{sup}}$  and gating network  $\phi_T$ ; each agent's personalized model  $\theta_{i,T} = \text{ROUTE}(\mathbf{S}_i, \mathbf{g}_{i,T}; \theta_T^{\text{sup}})$ , where MoDE gate  $\mathbf{g}_{i,T} = \text{DomGate}(\mathcal{D}_i; \phi_T)$ .
-

## 7.7 Experiments

### 7.7.1 Simulations

We select *five* multimodal or multitask datasets as the source to create *three* AFL simulations. The five **source datasets** are listed in Table 7.7.1, including a 3D object recognition dataset [146] containing four modalities—3d point clouds, voxels, and two rendered views of each object; a three-modal multimedia human emotion recognition dataset [242]; a bimodal human action recognition dataset [243] that identifies human actions from RGB videos and 3D skeletal data; a single-modal two-task dataset [242]; and, a bimodal audio-image digit classification dataset [242].

We then create *three* AFL simulations based on these datasets, whose details are summarized in Table 7.7.2. **Object4M2T** simulates a single-domain 4-modal 2-downstream-task AFL scenario with MTC heterogeneity, wherein there are discrepant modalities types (M), different tasks with different output spaces (T), and concept shifts (C) across 50 agents. The goal is to utilize agent collaboration to comprehensively understand real-world objects from various perspectives, including different angles observing 3D objects, their spatial point clouds, and voxel representations. **Human4M3T** simulates a multi-domain 4-modal 3-downstream-task AFL scenario, with MTDC heterogeneity patterns across 50 agents, where agents vary in their input spaces (M), output spaces (T), input distributions (D), as well as concept shifts (C). It is created by combining CMU-MOSE, a human *emotion* recognition dataset [242] containing video, audio, and text modalities, and NTU-RGBD, a human *action* recognition [243] containing videos and 3D skeletal points. The goal of this simulation is centered around human understanding—jointly learning the human facial emotions and the human physical motions. **Mnist2M4T** simulates a multi-domain 2-modal 4-downstream-task AFL scenario having 4 patterns of heterogeneity (MTDC) across 50 agents. The four downstream tasks include classifying the item on the top-left, on the bottom-right, generating the digit images, and generating spoken English digits.

For classification tasks within these simulations, the sizes of local label spaces are limited to at most 5. In Human4M3T and Mnist2M4T, the domain shifts across



Table 7.7.1: Statistics of 5 Source Datasets.

Dataset	# Samples	Modalities	Tasks
ModelNet40 [146]	12,300	{3dPointCloud, 3dVoxel, 3dView1, 3dView2}	{ <i>Classification</i> (40 object classes), <i>Generation</i> (point clouds)}
CMU-MOSEI [242]	22,777	{Audio, Text, Video}	{ <i>Classification</i> (6 human emotions), <i>Regression</i> (sentiment scores)}
NTU-RGBD [243]	56,880	{Video, 3dSkeleton}	{ <i>Classification</i> (60 human actions)}
Multi-FMNIST [242]	70,000	{Image}	{ <i>Classification Task 1</i> (10 digits), <i>Classification Task 2</i> (10 objects)}
AV-MNIST [242]	70,000	{Image, Acoustic}	{ <i>Generation Task 1</i> (image), <i>Generation Task 2</i> (audio), <i>Classification</i> (10 digits)}

Table 7.7.2: Statistics of the three AFL Simulations for evaluating the complex heterogeneity of Modality Gap, Task Difference, Concept Shift, with Domain Drifts.

Simulations	Source Datasets	Agents ( $N$ )	Modalities ( $M$ )	Tasks ( $O$ )	Modality-task Pairs ( $\#Agents \times (\mathcal{X}_{\mathcal{I}_i} \rightarrow \mathcal{Y}_{\mathcal{O}_i})$ )
Object4M2T	ModelNet40	50	4	2	$5 \times (\mathcal{X}_{\{\text{view1}\}} \rightarrow \mathcal{Y}_{\{\text{cls.objects}\}})$
					$5 \times (\mathcal{X}_{\{\text{view2}\}} \rightarrow \mathcal{Y}_{\{\text{cls.objects}\}})$
					$5 \times (\mathcal{X}_{\{\text{view1, voxel}\}} \rightarrow \mathcal{Y}_{\{\text{cls.objects}\}})$
					$5 \times (\mathcal{X}_{\{\text{voxel, view2}\}} \rightarrow \mathcal{Y}_{\{\text{cls.objects, point.cloud}\}})$
					$5 \times (\mathcal{X}_{\{\text{view1, view2}\}} \rightarrow \mathcal{Y}_{\{\text{cls.objects, point.cloud}\}})$
					$5 \times (\mathcal{X}_{\{\text{view1, view2}\}} \rightarrow \mathcal{Y}_{\{\text{point.cloud}\}})$
					$5 \times (\mathcal{X}_{\{\text{point.cloud, voxel}\}} \rightarrow \mathcal{Y}_{\{\text{cls.objects}\}})$
					$5 \times (\mathcal{X}_{\{\text{point.cloud}\}} \rightarrow \mathcal{Y}_{\{\text{cls.objects}\}})$
					$5 \times (\mathcal{X}_{\{\text{view1, point.cloud}\}} \rightarrow \mathcal{Y}_{\{\text{cls.objects}\}})$
$5 \times (\mathcal{X}_{\{\text{view1, view2, point.cloud}\}} \rightarrow \mathcal{Y}_{\{\text{cls.objects}\}})$					
Human4M3T	CMU-MOSEI + NTU-RGBD	50	4	3	$5 \times (\mathcal{X}_{\{\text{video}\}} \rightarrow \mathcal{Y}_{\{\text{emotions}\}})$
					$5 \times (\mathcal{X}_{\{\text{audio}\}} \rightarrow \mathcal{Y}_{\{\text{sentiment}\}})$
					$5 \times (\mathcal{X}_{\{\text{text}\}} \rightarrow \mathcal{Y}_{\{\text{sentiment, emotions}\}})$
					$5 \times (\mathcal{X}_{\{\text{video, audio}\}} \rightarrow \mathcal{Y}_{\{\text{emotions}\}})$
					$5 \times (\mathcal{X}_{\{\text{audio, text}\}} \rightarrow \mathcal{Y}_{\{\text{sentiment}\}})$
					$5 \times (\mathcal{X}_{\{\text{video, text}\}} \rightarrow \mathcal{Y}_{\{\text{sentiment, emotions}\}})$
					$5 \times (\mathcal{X}_{\{\text{video, audio, text}\}} \rightarrow \mathcal{Y}_{\{\text{emotions}\}})$
					$5 \times (\mathcal{X}_{\{\text{skeleton}\}} \rightarrow \mathcal{Y}_{\{\text{cls.action}\}})$
					$5 \times (\mathcal{X}_{\{\text{video, skeleton}\}} \rightarrow \mathcal{Y}_{\{\text{cls.action}\}})$
$5 \times (\mathcal{X}_{\{\text{video}\}} \rightarrow \mathcal{Y}_{\{\text{cls.action}\}})$					
Mnist2M4T	Multi-FMNIST + AV-MNIST	50	2	4	$5 \times (\mathcal{X}_{\{\text{image}\}} \rightarrow \mathcal{Y}_{\{\text{cls.digits}\}})$
					$5 \times (\mathcal{X}_{\{\text{audio}\}} \rightarrow \mathcal{Y}_{\{\text{cls.digits}\}})$
					$5 \times (\mathcal{X}_{\{\text{image, audio}\}} \rightarrow \mathcal{Y}_{\{\text{cls.digits}\}})$
					$5 \times (\mathcal{X}_{\{\text{audio}\}} \rightarrow \mathcal{Y}_{\{\text{cls.digits, gen.image}\}})$
					$5 \times (\mathcal{X}_{\{\text{image}\}} \rightarrow \mathcal{Y}_{\{\text{gen.audio}\}})$
					$6 \times (\mathcal{X}_{\{\text{image, audio}\}} \rightarrow \mathcal{Y}_{\{\text{gen.image, gen.audio}\}})$
					$6 \times (\mathcal{X}_{\{\text{image}\}} \rightarrow \mathcal{Y}_{\{\text{cls.digits, cls.objects}\}})$
$6 \times (\mathcal{X}_{\{\text{image}\}} \rightarrow \mathcal{Y}_{\{\text{cls.objects, gen.audio}\}})$					
$7 \times (\mathcal{X}_{\{\text{image}\}} \rightarrow \mathcal{Y}_{\{\text{cls.digits, cls.objects, gen.image}\}})$					

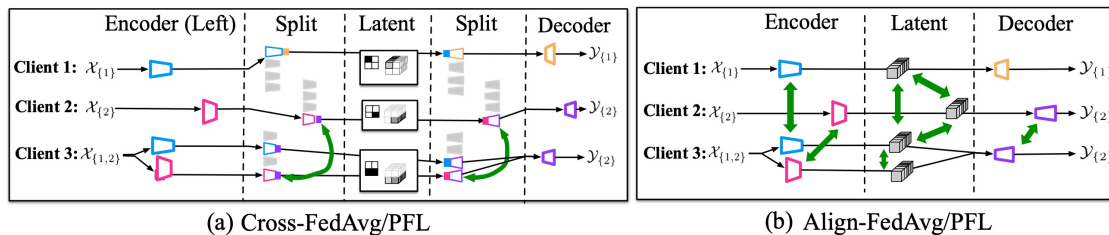


Figure 7.7.1: Illustration of alignment-based and no-alignment baselines.

agents are created by randomly rotating the objects in images, adding different noises in audio and images, replacing the image backgrounds, or, replacing low-frequency signals in raw audios with other sounds. All samples on each agent use the same random seed to generate noises. The  $N$  agents are divided into 5 groups and each group use a unique random seed when generating noises or selecting background. The number of samples and data split on each agent are determined in the same ways as Chapter 6.

## 7.7.2 Baseline Methods

We compared DisentAFL with five baselines. **Local** lets agents separately train their models without any collaboration—neither positive transfer nor negative transfer  $\mathcal{R}(\cdot)=0$ . To apply **FedAvg** [33] in our settings, we split agents into several disjoint groups such that each group share the same modality-task pair. The collaboration is within the same group of agents using FedAvg. Any information sharing between different groups is prohibited. **Cross-FedAvg**, in addition to FedAvg, encourages the sharing of certain modality-to-task transmitter between different groups that have overlapping on both modalities and tasks, as illustrated in Figure 7.7.1(a). There is no modality-shared or task-shared representations in this baseline. **Align-FedAvg**, in addition to FedAvg, encourages the sharing of certain encoders/decoders between different groups that have either overlapping modalities or overlapping tasks. The after-encoding and before-decoding representations of all modalities and task are aligned onto the same latent space. **Cross-PFL** is similar to Cross-FedAvg, except that using the personalized FL method [28] to every modality-task pair group of agents.

### 7.7.3 Setups

**Hyperparameters:** We implemented DisentAFL and baselines using Pytorch3 and ran all experiments on a single A100 GPU device. The hyperparameters are briefly as follows. (1) *Model configuration:*  $d^{\text{modal}} = 768$ ;  $d_m^{\text{modal}} = 384$ ;  $d_{\text{share}}^{\text{modal}} = 384$ ;  $d^{\text{latent}} = 128$ ;  $d_o^{\text{task}} = 384$ ; the number of experts  $M$ ,  $O$  are dynamically determined by the server based on the modality types and task types observed in the previously received agent models; and, the number of domain experts  $D = 5$  and the MoDE layers are attached to the early 1/2 attention layers. (2) *Local training:* epoch numbers fixed to 10; batch size fixed to 64; learning rates 0.015, 0.05, 0.001 for the three simulations, repetitively; for local optimizers we use Adam; the weights of auxiliary loss are in average 0.08. (3) *Global training:* total round ( $T = 30$ ); agents sampling ratio (0.25% per round);  $\epsilon$ -greedy sampling ( $\epsilon = 0.3$ ).

**Evaluation Metrics.** We have observed that the knowledge transfer quality influences not only the *convergence speed* (i.e., the number of rounds needed for approaches to reach certain overall performance) as well as the *overall performance after convergence*. In this chapter, since we mainly focus on the quality of knowledge transfer (i.e., whether our approach help to avoid negative transfer to accelerate the convergence and whether our approach can improve the performance after positive transfer), we pay attention to the convergence progress evaluated on each personal model—the *average local performance that the approach can converge to on a certain given round ID*. While other metrics such as the time cost each round and the memory cost could also provide perspectives for evaluating FL methods, they are not one of the motivations of this chapter.

### 7.7.4 Results

**Overall Performance.** Table 7.7.3 shows a comparison of the globally-averaged local performance achieved by different collaborative learning approaches at the round  $T = 30$ . In general, our method outperforms baselines with a large margin in all multi-task simulations. This demonstrates that our approach achieved better convergence speed, indicating that the past knowledge transfer might have more

Table 7.7.3: Average testing accuracy over all agents’ classification tasks at  $T$ .

Methods	Object4M2T	Human4M3T	Mnist2M2T	Mnist2M4T
<b>Local</b>	88.23 $\pm$ 0.72	70.23 $\pm$ 0.79	93.01 $\pm$ 0.30	92.38 $\pm$ 0.21
<b>FedAvg</b> [33]	84.63 $\pm$ 0.02	74.12 $\pm$ 0.93	92.79 $\pm$ 0.12	83.22 $\pm$ 0.35
<b>Cross-FedAvg</b> [33]	88.35 $\pm$ 0.20	72.41 $\pm$ 0.72	91.65 $\pm$ 0.32	88.65 $\pm$ 0.30
<b>Align-FedAvg</b> [33]	89.73 $\pm$ 0.68	69.65 $\pm$ 0.73	91.18 $\pm$ 0.38	85.18 $\pm$ 0.53
<b>Cross-PFL</b> [28]	90.11 $\pm$ 0.63	<u>75.37</u> $\pm$ 0.26	94.20 $\pm$ 0.94	92.20 $\pm$ 0.25
<b>DisentAFL</b> <sup>†</sup>	95.91 $\pm$ 0.89	73.39 $\pm$ 0.32	95.37 $\pm$ 0.14	92.97 $\pm$ 0.93
<b>DisentAFL</b>	<b>96.38</b> $\pm$ 0.41	<b>75.68</b> $\pm$ 0.74	<b>97.44</b> $\pm$ 0.36	<b>95.95</b> $\pm$ 0.16

efficiently captured and conveyed the positive information among agents.

**Impact of Downstream Task Difference.** Table 7.7.3 (column 4) shows the result of a two-task version of Mnist2M4T, where the two generative tasks are removed from the original setting. Comparing column 5 to column 4, we observe that the baselines exhibit a larger performance drop than our method when there are more downstream tasks during collaboration. This indicates that our method can better exclude conflicting knowledge that should be shared across agents.

**Ablation Study.** The last two rows in Table 7.7.3 compare the results with and without auxiliary losses for knowledge disentanglement in the latent space. DisentAFL<sup>†</sup> is a variant of DisentAFL that removes the disentanglement loss from local objective functions. While the performance increase is not significant for Object4M2T and Human4M3T, on Mnist2M4T, where the number of downstream tasks is larger, it demonstrates the positive impacts of disentangling knowledge on improving convergence speed. This underscores the importance of disentangling knowledge for purifying knowledge transfer in the context of federated learning, suggesting that the latent spaces of local models in AFL contain conflicting knowledge. Therefore, utilizing disentanglement loss can help prevent negative transfer. In the future, we will provide more experiments to examine the role of disentanglement loss in our framework.

## 7.8 Conclusions

This chapter studied the Modality-task Agnostic Federated Learning (AFL) problem to address a specific CoMML situation where there are simultaneous four personalization patterns—modality, concept, domain, and task type personalization. We discussed a unique challenge in AFL rather than traditional FL due to the asymmetrical inter-client knowledge relationships. Then, we introduced a new DisentAFL approach that can address this challenge via a two-stage Knowledge Disentanglement and Gating mechanism, whose main idea is to decompose the asymmetrical inter-client information sharing scheme into several independent symmetrical inter-client information sharing schemes. Experiments show that the proposed DisentAFL framework outperforms baseline approaches in simulated cross-modal cross-task object recognition and human understanding scenarios, which demonstrates that our approach effectively addressed the challenge of MTDC heterogeneity.

## Part III

# Collaboration with Implicit Knowledge Transfer

# Chapter 8

## Latent Transfer for Architecture-free Collaboration

### 8.1 Introduction

In this chapter, we explore scenarios in which CoMML agents have the flexibility to define their own **modalities, concepts, and domains**, as well as the freedom to design their own **network architectures**, while, for simplicity, we assume that all agents are targeting the same downstream task. Such a setting can be likened to a social scenario where individuals who think in different ways engage in efficient communication with each other; a person thinking in a unique way can be analogous to a unique architectural design that has a distinct computational flow for processing the inputs. In addition, we continue to explore in the global-local decentralized CoMML paradigm (Eq.(2.2)) as in Chapter 6 since it learns personal models and thus is well-suited for the CoMML setting with personalized model architectures. However, we found no previous work that strictly formulates the aforementioned CoMML setting for us to use. Although there are architecture-heterogeneous FL approaches proposed by [244, 93, 91, 92], their problem settings differ from ours because they still assume agents' model architectures have similar computational flows or share common architecture families. To fill this gap, we propose a novel **Architecture-personalized Multimodal Federated Learning (AMFL)** setting to delve into the research on CoMML with simultaneous modality,

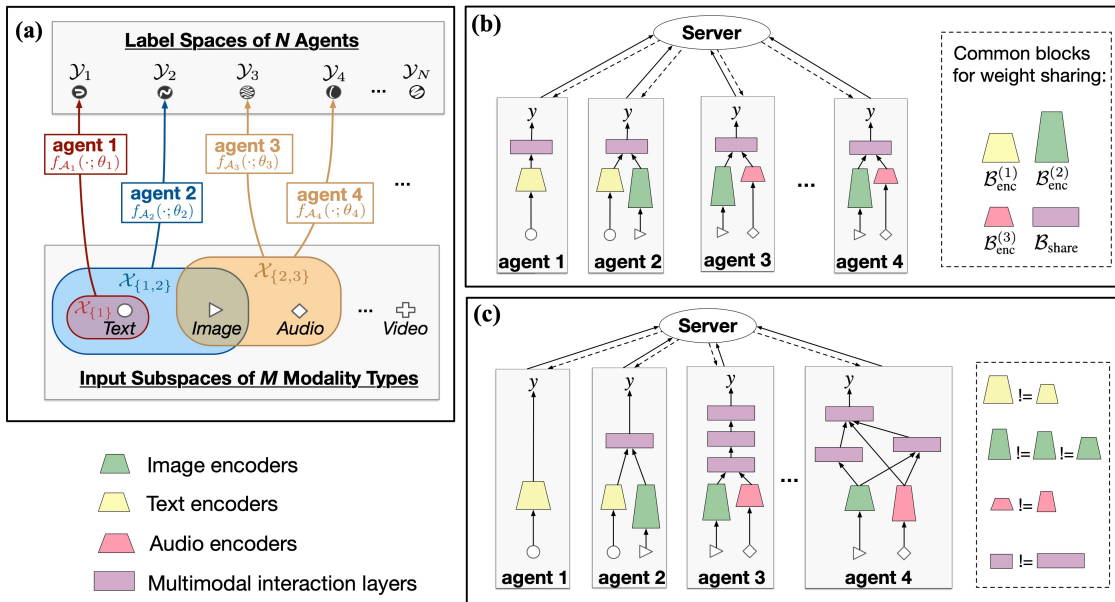


Figure 8.1.1: (a) Local mapping functions per agent in Multimodal Federated Learning (MFL). (b) Problem setting of Part II (previous chapters) that requires restrictive compositional neural architectures. (c) Problem setting of **Architecture-personalized MFL (AMFL)**, without a restriction on local model architectures. In this AMFL example, “agent 1” and “agent 2” show a layer-*width* difference, “agent 2” and “agent 3” show a *depth* difference, and “agent 3” and “agent 4” show a *topology* difference, at multimodal interaction modules.

concept, domain, and model architecture preferences.

An illustration of AMFL is shown in Figure 8.1.1. Different from previous chapters (Chapter 6 and 7) and different from traditional MFL [40, 73, 87] that leveraged a compositional neural architecture design (i.e., neural architectures are different but are made of common smaller blocks), in AMFL we assume no prior knowledge on how the heterogeneous architectures of different agents are correlated with each other. That is, in AMFL, the neural networks of different agents may adopt **structurally distinct computational flows** and thus are **non-splittable** into global common blocks. Example real-world applications of AMFL include: (1) *Task complexity difference*. First, due to personalization, agents may vary greatly in their *complexity* of modeling the inter-modal interactions, with some local tasks being more complex than others. In this situation, it is essential for different agents to employ diverse neural network sizes or even utilize varied network families that



align with their personal data distributions. **(2) Multimodal pattern difference.** Second, studies in Multimodal Fusion [245, 11] have demonstrated that the *patterns* of inter-modal interactions can also shift across agents, necessitating using different mechanisms (i.e. concatenation, element-wise product, cross-attention fusion, tensor interaction, and so on) to learn the specific inter-modality interaction style at each agent [245, 11]. For example, some *image-text* agents may benefit from just a straightforward concatenation of different modalities’ features; some image-audio agents may need an element-wise alignment; and some image-text-audio agents may need a complex intra- and inter-modal interaction mechanism. **(3) Resource budget difference.** Third, in real-world systems, agent devices can be mobile phones, tablets, and personal computers, thus vary greatly in *computation resource budgets* (e.g., computation capacity, memory, storage, and network bandwidth). A resource-poor agent cannot afford a large model architecture, such as a pretrained large language model, while a resource-rich agent can benefit from it.

The unique **challenge** of AMFL is how to encourage beneficial knowledge sharing between *statistically heterogeneous* agents whose model architectures might have *different computational flows*. In tackling this challenge, one may consider several straightforward approaches. (1) First, one might consider using a globally-shared large language model that unify all different local tasks. However, the communication would be forbiddingly expansive and many local devices may not afford such an expansive model. (2) Second, recent FL approaches have explored the settings with diversified neural architectures, including feature-sharing methods [244, 93, 91, 92] and explicit weight-sharing methods [95, 72, 96, 97], can be used. However, these methods have shown *inefficient* in AMFL scenarios, since the diverse model topologies in AMFL either impose a costly workload on the server or require long time for uploading and downloading a large supernetwork at each communication round. Also, these methods struggle to maximize the beneficial knowledge sharing among agents, especially with significant topology differences among models.

Different from existing works, we propose a novel **implicit** knowledge transfer framework to solve the AMFL challenge, namely **FedMBridge**. The main idea is that, instead of sharing original weights across diverse weight spaces, we introduce

a global *bridge function* that learns to perform knowledge sharing on a globally-shareable *latent* space. Intuitively, the bridge function is held on the server to act as a “bridge”, which *balances and digests* the two disentangled heterogeneity patterns (i.e. statistical and architecture heterogeneity) and then *generates* local weights in the raw statistical-architectural entangled heterogeneity pattern. To achieve such a bridge function, we introduce a **Topology-Aware HyperNetwork (TAHN)**, which is formulated as a two-stage process: the first stage encodes the implicit roles of each layer using graph neural networks, and the second stage aims to combine the layer-role information with task information to reconstruct local weights. This chapter’s contributions are threefold:

- We study an under-explored AMFL problem. To the best of our knowledge, this is one of the first works that tackle the collaboration of diversified multimodal fusion strategies for general-purpose federated AI systems.
- We propose the FedMBridge framework to solve AMFL, where we introduce a brand-new Topology-aware HyperNetwork to automatically balance and digest architecture gap and statistical heterogeneity in an efficient manner.
- We evaluate our approach on four AMFL simulations, which demonstrates the effectiveness of our approach for addressing AMFL.

## 8.2 Related Works

The majority of existing Federated Learning (FL) assumes a *uniform* neural architecture shared across agents, including fine-tuning methods (e.g., FedPer [60], pFedMe [61], Per-FedAvg [27]), meta-learning methods [39], mixture methods (e.g., APFL [62], PFL-MoE [29], Factorized-FL [63]), multi-task learning methods (e.g., MOCHA [28, 73]), and HyperNetwork based methods (e.g., HyperPFL [30]).

Given the recent real-world application requirements for collaboration across diverse computation platforms with varying resource budgets, the knowledge transfer across proprietary model architectures has attracted significant research attention in collaborative learning. We will review two *orthogonal* lines of approaches in FL that handle architecture gaps during collaboration.

**Feature Sharing Approaches.** Federated Mutual Learning utilizes Knowledge Distillation to share predictions or intermediate features of local models among agents, making it a natural choice for facilitating knowledge transfer across diverse architectures. For example, FedDistill [90], FML [49], PFML [91], and FedGKD [92], adopt collaborative knowledge distillation to align predictions among diverse client models on the server. Without requiring a public dataset at server, Data-free Federated Distillation learns an additional data generator to simulate local data on the server [93, 94]. However, these methods have shown inefficiency in AMFL scenarios, as Distillation typically requires an additional prediction inference process and relies on an extra public dataset or multimodal data generator to compute supervision signals.

**Weight Sharing Approaches.** Another line of works in Federated Learning (FL) has explored direct weight-sharing approaches to handle diversified network architectures. HeteroFL [95] allows participants to share parameters among prototypes. Another line of works, such as Split-Mix [72], DepthFL [96], and DisPFL [97], leverages a large supernet, where each agent is aligned to partial parameters in this supernet indicated by a supermask learned by pruning techniques, and/or adopts progressive distillation to handle depth difference. However, these methods have shown inefficient in AMFL scenarios as well as suboptimal in transferring the knowledge among agents—due to the significant topology differences across architectures, weight sharing in supernet becomes quite expensive and fail to effectively align local model weights, thereby reducing knowledge-sharing opportunities. This chapter introduces an efficient and effective weight-sharing scheme to overcome the two limitations of prior works.

In this chapter, we will follow and improve the weight-sharing approaches, as they require no extra public dataset or data generator on the server. In practice, since the two lines of work (feature sharing and weight sharing) are orthogonal, we can use them together. In our experiments, we will illustrate how our method can be combined with the feature-sharing baselines.

## 8.3 Problem Formulation

We deal with the setup of  $N$  agents with  $M$  modality types (e.g. image, video, text, audio, tabular) working together to improve their local personal models  $\theta_1, \theta_2, \dots, \theta_N$ . We introduce architecture heterogeneity; however, for simplicity, in this chapter, we assume all agents solve the same downstream task, i.e.,  $O = 1$ .

### 8.3.1 Local Task Setting on Each Agent

Each agent  $i \in \{1, 2, \dots, N\}$  focuses on learning a subset of modality types  $\mathcal{I}_i \subseteq \{1, 2, \dots, M\}$  and has a combinatorial *input space*  $\mathcal{X}_{\mathcal{I}_i} := (\mathcal{X}^{(m)} | \forall m \in \mathcal{I}_i)$ , where  $\mathcal{X}^{(m)}$  is the subspace associated with the modality type  $m$ . For example, as illustrated in Figure 8.1.1(a), “agent 2” focuses on an image-text bimodal task; “agent 3” focuses on an audio-visual bimodal task; “agent 1” learns a text-only unimodal task. Each agent  $i$  also has a personalized label space  $\mathcal{Y}_i$ . Each agent  $i$  has access *only* to its **local dataset**  $\mathcal{D}_i = \{(\tilde{\mathbf{x}}_{ij}, y_{ij})\}_{j=1}^{n_i}$ , sampled from  $\tilde{\mathbf{x}}_{ij} \sim \mathcal{P}_i(\tilde{\mathbf{x}})$  and  $y_{ij} \sim \mathcal{Q}_i(y | \tilde{\mathbf{x}}_{ij})$ , where  $\mathcal{P}_i$  is the agent-specific input distribution over the combinatorial input space  $\mathcal{X}_{\mathcal{I}_i}$ , and  $\mathcal{Q}_i$  is the conditional output distribution over the space  $\mathcal{Y}_i$ . Each sample’s input consists of the modalities  $\tilde{\mathbf{x}}_{ij} = (\mathbf{x}_{ij}^{(m)})_{m \in \mathcal{I}_i}$  present as in  $\mathcal{I}_i$ , where  $\mathbf{x}_{ij}^{(m)}$  denotes the modality  $m$  in  $\tilde{\mathbf{x}}_{ij}$ .

Each agent  $i$  aims to obtain a **local mapping function**

$$f_{\mathcal{A}_i}(\cdot; \theta_i) : \mathcal{X}_{\mathcal{I}_i} \rightarrow \mathcal{Y}_i \quad (8.1)$$

characterized by a **agent-specific model architecture**  $\mathcal{A}_i$  and parameterized by trainable weights  $\theta_i \in \mathbb{R}^{d_i}$ , where  $d_i$  indicates the structure of the weight space associated with  $\mathcal{A}_i$ .

### 8.3.2 Multi-agent Collaboration with Architecture Gap

We follow the weight-sharing paradigm for Multimodal Federated Learning (MFL), as it requires no extra public dataset or data generator on the server. MFL inherently faces the challenge of network architecture heterogeneity among agents. To distinguish this chapter from previous ones, we start by clarifying our task setting.

**Definition 8.1 (MFL with Splittable Architecture Gap):** Given that local models  $f_{\mathcal{A}_i}(\cdot; \boldsymbol{\theta}_i)$ ,  $i = 1, 2, \dots, N$ , have either one of modalities or multiple modalities as inputs, when addressing the knowledge sharing among heterogeneous multimodal model architectures, traditional MFL systems [61], as well as previous Chapter 6 and Chapter 7, have leveraged a design of *compositional/splittable* neural architectures  $\mathcal{A}_i := \{\mathcal{B}_{\text{enc}}^{(m)} | \forall m \in \mathcal{I}_i\} \cup \{\mathcal{B}_{\text{share}}\} \cup \{\mathcal{B}_{\text{dec},i}\}$ , such that heterogeneous model architectures are *manually* split into smaller homogeneous *blocks*, allowing any pair of agent to share some common blocks, as illustrated in Figure 8.1.1(b). The global objective of these weight-sharing approaches is formulated as

$$\min_{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \dots, \boldsymbol{\theta}_N} \left[ \frac{1}{N} \sum_{i=1}^N \mathcal{L}_i(\boldsymbol{\theta}_i) \right] + \mathcal{R}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \dots, \boldsymbol{\theta}_N) \quad (8.2)$$

s.t.

$$\mathcal{A}_i \cap \mathcal{A}_{i'} = \{\mathcal{B}_{\text{enc}}^{(m)} | \forall m \in \mathcal{I}_i \cap \mathcal{I}_{i'}\} \cup \{\mathcal{B}_{\text{share}}\}, \quad \forall i, i' \in [N] \quad (8.3)$$

which aims to **(1)** jointly optimize the local objectives of all agents  $\min_{\boldsymbol{\theta}_i} \mathcal{L}_i(\boldsymbol{\theta}_i) := \mathbb{E}_{(\tilde{\mathbf{x}}, y) \sim \mathcal{D}_i} l(y, f_{\mathcal{A}_i}(\tilde{\mathbf{x}}; \boldsymbol{\theta}_i))$ , and meanwhile, **(2)** leverage a central server to encourage a privacy-preserving *knowledge sharing* scheme among agents  $\mathcal{R}(\cdot)$  in order to boost each agent’s local model performance. The constraint Eq.(8.3) allows  $\mathcal{R}(\cdot)$  to be achieved through decomposed explicit weight sharing schemes.

**Definition 8.2 (AMFL with Computational Architecture Gap):** In this chapter, we focus on the Architecture-personalized MFL (**AMFL**), which relaxes the constraints of Eq.(8.3). AMFL addresses a *more general* MFL scenarios without setting any restriction on the architecture design. Local models  $f_{\mathcal{A}_i}(\cdot; \boldsymbol{\theta}_i)$ ,  $i = 1, 2, \dots, N$ , can take on any neural architectures specified by local users, each with potentially different *computational flows*. The global objective is

$$\min_{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \dots, \boldsymbol{\theta}_N} \left[ \frac{1}{N} \sum_{i=1}^N \mathcal{L}_i(\boldsymbol{\theta}_i) \right] + \mathcal{R}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \dots, \boldsymbol{\theta}_N) \quad (8.4)$$

which aims to **(1)** jointly optimize the local objectives of all agents  $\min_{\boldsymbol{\theta}_i} \mathcal{L}_i(\boldsymbol{\theta}_i) := \mathbb{E}_{(\tilde{\mathbf{x}}, y) \sim \mathcal{D}_i} l(y, f_{\mathcal{A}_i}(\tilde{\mathbf{x}}; \boldsymbol{\theta}_i))$ , where  $l(\cdot, \cdot)$  is the loss function, and meanwhile, **(2)** leverage a central server to encourage a privacy-preserving *knowledge sharing* scheme

among agents  $\mathcal{R}(\cdot)$  in order to boost each agent’s local model performance. The server has no prior knowledge about the inter-agent weight-space sharing scheme. It is possible that any pair of agents’ architectures  $\mathcal{A}_i$  and  $\mathcal{A}_{i'}$  have structurally-different computational flows. That is, for  $\forall i, i' \in [N]$ , it is possible

$$\mathcal{A}_i \cap \mathcal{A}_{i'} = \emptyset. \quad (8.5)$$

Given this relaxed design, there will be three particular **cases of architecture gap** that are not permitted in traditional MFL: **(1) Topology Difference** is a most common situation in multimodal FL systems. Two agents might use different model types (e.g. one agent is based on Transformer while the other is based on ResNet) or use different multimodal fusion strategies for different input modality types (e.g. one agent uses alignment while the other uses concatenation). **(2) Depth Difference** refers that two agents having the same topology (e.g. both are based on ResNet) but their numbers of layers/modules are different. **(3) Width Difference** describes a situation where two agents having the same topology and same depth, but their numbers of neurons at each layer are different. Examples of the three cases are illustrated in Figure 8.1.1(c).

## 8.4 Proposed FedMBridge

In the AMFL settings (Definition 8.2), any pair of agent models might have completely different computational flows. In order to solve Eq.(8.4) under such settings, there are two challenges we are going to deal with. **(1) Challenge 1:** how to design and maximize the benefits of the inter-agent knowledge sharing scheme  $\mathcal{R}(\cdot)$ , wherein there are simultaneous *architecture heterogeneity* ( $\mathcal{A}_1 \neq \mathcal{A}_2 \neq \dots \neq \mathcal{A}_N$ ) as well as *statistical heterogeneity* (Non-IIDness) among agents? **(2) Challenge 2:** Existing Multimodal FL approaches cannot address the computational architecture gap. Therefore, it is desirable to particularly deal with how to *automatically bridge the computational architecture gap*, for efficient and effective knowledge sharing among agents with heterogeneous neural architectures and distributions.

We propose a new multimodal FL framework, **FedMBridge**, which *automatically bridges the architecture gap* among statistically heterogeneous agents. We

will first introduce the main idea of FedMBSide and then present its three components: (1) the topological graph representation of local multimodal architectures; (2) the hypernetwork that generates personal weights conditioned on the topological graphs of architectures; (3) the federated training workflow.

### 8.4.1 Main Idea

**Rethinking Implicit and Explicit Weight Sharing:** Explicit weight sharing, or the simple weight aggregation within a globally-shared weight space, is seen in standard FL with homogeneous architectures [38, 30] or some Pruning-based FL methods with only width or depth differences [246, 247, 248]. However, in AMFL, an *explicit* weight sharing is *not available* since a globally shared weight space does not even exist, especially if agent models vary significantly in their topologies or depths. Alternatively, we explore an *implicit* weight sharing mechanism for AMFL: instead of sharing *original* weights across diverse weight spaces, we aim to perform knowledge sharing among agents within globally-shared *latent* space(s).

**Definition 8.3 (Bridge Function):** We propose an implicit weight sharing mechanism for AMFL by introducing a global “*bridge*” function  $h(\cdot, \cdot; \phi)$ , where  $\phi \in \mathbb{R}^D$  is the trainable weights of the bridge function. The original locally-trained weights of  $N$  agents from diverse weight spaces  $\theta_i \in \mathbb{R}^{d_i}, d_1 \neq d_2 \neq \dots \neq d_N$  are re-parameterized as the output of the bridge function conditioned on two agent-specific generative factors

$$\theta_i := h(\mathcal{A}_i, \mathbf{c}_i; \phi), \quad \forall i \in [N] \quad (8.6)$$

where the first generative factor  $\mathcal{A}_i \in \mathcal{G}$  is the local neural architecture from a globally-shared *latent topology space*  $\mathcal{G}$  and the second generative factor  $\mathbf{c}_i \in \mathcal{T}$  represents the local task from a globally-shared *latent task space*  $\mathcal{T}$ . While  $\mathcal{G}$  manages only architectural heterogeneity,  $\mathcal{T}$  manages only statistical heterogeneity. Intuitively, the bridge function  $h$  can be treated as a generative meta-learner that can digest two *disentangled* heterogeneity patterns to solve the raw *statistical-architectural entangled* heterogeneity pattern.

**Definition 8.4 (AMFL based on Implicit Weight Sharing):** Given the defined bridge function, we can rewrite our global objective in Eq.(8.7) as

$$\min_{\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_N, \phi} \left[ \frac{1}{N} \sum_{i=1}^N \mathcal{L}_i(h(\mathcal{A}_i, \mathbf{c}_i; \phi)) \right] + \mathcal{R}(\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_N; \phi). \quad (8.7)$$

where  $\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_N$  are trainable agent embeddings and  $\phi$  is the trainable weights of the generative bridge function.

## 8.4.2 Multimodal Neural Architectures as Graphs

### 8.4.2.1 Local Neural Architectures in AMFL

Following standard configurations [20], we let the local neural architecture  $\mathcal{A}_i$  of every agent  $i \in [N]$  in AMFL follows a many-to-one *multimodal fusion* pipeline that basically consists of three parts: **(1) Unimodal Encoders:** At the first step, we employ unimodal encoders to extract modality-specific features. For each modality type in AMFL, different agents may utilize various architecture families for the unimodal encoder of that modality. For instance, if the image modality appears in multiple agents, they may employ ResNets, CNNs, MLPs, RNNs, or Small Transformers as their image encoders. **(2) Multi-modality Interaction Module:** Then, we model the complex intra- and inter-modality interactions to effectively fuse the complementary information from multiple modalities. Such interaction modes can be diverse across agent tasks. Therefore, in AMFL, we allow that a variety of multimodal fusion strategies implemented in different local models. For example, some agents utilize concatenation operation to fuse their input modalities, while others might employ element-wise alignment operation. Additionally, some agents may utilize higher-order modeling techniques such as tensor fusion [20], low-rank fusion [111], or cross-attention fusion mechanisms. **(3) Personal Final Layer:** Finally, a fully-connected classifier is employed to project the fused features to predictions. Given that each agent has its own label space (i.e., personal concepts), the final layers are unique for each agent.



### 8.4.2.2 Graph Representation of Multimodal Neural Architecture

The neural architectures of local models, each with its own computational flow  $f_{\mathcal{A}_i}$ , can be represented as graph structures. Formally, we represent the multimodal neural architecture  $\mathcal{A}_i$  of each agent  $i$  as a **directed acyclic graph** structure:

$$\mathcal{A}_i := (\mathcal{V}_i, \mathcal{E}_i, \mathbf{Z}_i^{(0)}). \quad (8.8)$$

Each node  $v \in \mathcal{V}_i$  stands for a *computational operator*  $f_v$  in the neural architecture.  $f_v$  can be either non-parametric (e.g. a concatenation operator) or parametric (e.g. a linear layer with weights of size  $16 \times 64$ ). Edges  $\mathcal{E}_i$  represents the *computational flow* of the neural architecture, where each edge  $e_{v' \rightarrow v} \in \mathcal{E}_i$  indicates that the output of the operator  $f_{v'}$  is the input of the operator  $f_v$ . Every node  $v$  is equipped with  $K$  types of *configuration or prior information* for the operator  $f_v$ , including layer types, layer levels, layer shapes, modality types or fusion stage, etc. The node feature matrix  $\mathbf{Z}_i^{(0)} \in \mathbb{R}^{|\mathcal{V}_i| \times K}$  holds such  $K$  configuration/prior information types for all operators in the graph.

Figure 8.4.1 (left) shows three example architecture graphs. Agent-1 focuses on image-text classification, using a CNN for image encoding and a Transformer for text encoding, with feature summation for fusion. Agent-2 tackles audio-visual classification, utilizing an MLP for audio encoding and outer-product interactions followed by an MLP for fusion. Agent-3 also addresses audio-visual classification but employs the cross-attention to fuse information from image and audio encoders. The white nodes denote non-parametric computational operators or the input or output nodes, and the colored nodes denote parametric computational operators.

### 8.4.2.3 Graph Construction

**Collection of Edges and Nodes:** First, we adopt DARTS (Differentiable Architecture Search) [249, 250] and [251] to gather nodes  $\mathcal{V}_i$  and edges  $\mathcal{E}_i$  by tracing the backward gradients of variables. We utilize the auto-differentiation tracer of compilers to compute a chain of backward gradients, facilitating the construction of the graph as follows. *Initially*, a model object is instantiated, and a dummy multimodal input sample is defined in advance. We feed the dummy input to the model,

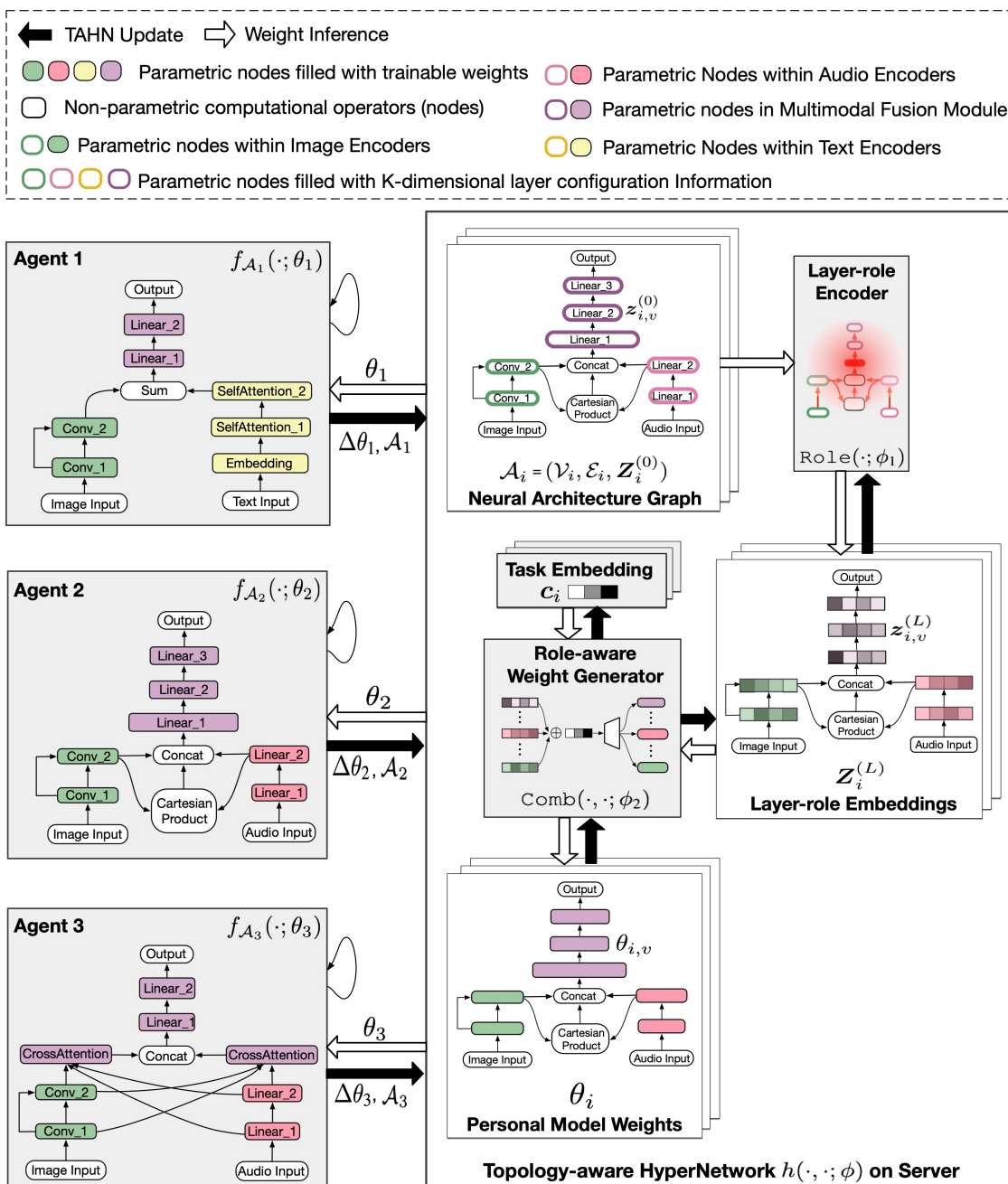


Figure 8.4.1: The proposed FedMBridge framework. The three example agents shown here use different multimodal fusion strategies. The server holds the Topology-aware HyperNetwork (TAHN) as a trainable bridge function for implicit weight sharing, which simultaneously overcomes statistical heterogeneity and architecture heterogeneity.

which undergoes the forward function execution of the model. *Subsequently*, we begin from the output variable’s gradient function, iteratively traversing the chain of gradient functions in reverse order along the computational graph. During this traversal, Breath First Search (BFS) is employed to collect nodes and edges. In particular, some gradient functions associated with parametric modules (such as the weights or biases of a Linear layer, the weights of a Conv layer, etc.) are gathered as ***parametric nodes***, while other gradient functions, without associated modules and named as ConcatBackward, AddBackward, ReluBackward, BmmBackward, or ViewBackward, etc., are collected as ***non-parametric nodes***. Since the traversal order is inverse, the directions of ***edges*** are reversed after gathering. *After the automatic traversal*,  $|\mathcal{I}_i|$  ***input nodes*** are additionally attached to the those operators with zero in-degrees, respectively. Each input node corresponds to a modality type in  $\mathcal{I}_i$ , determined by matching parameter names of the operators. Finally, there should be  $|\mathcal{I}_i|$  zero-indegree nodes and one zero-outdegree node.

**Construction of Node Features:** Second, we construct node features  $\mathbf{Z}_i^{(0)} = \{\mathbf{z}_{i,v}^{(0)}\}_{v \in \mathcal{V}_i}$  such that they provide sufficient context for learning the functionality role of each layer. For each node  $v \in \mathcal{V}_i$ . i.e. the computational operator  $f_v$  within the neural architecture  $\mathcal{A}_i$ , its node feature  $\mathbf{z}_{i,v}^{(0)} = [z_{i,v,1}^{(0)}, z_{i,v,2}^{(0)}, \dots, z_{i,v,k}^{(0)}, \dots, z_{i,v,K}^{(0)}]$  consist of  $K$  configuration/information types. In particular, we utilize  $K = 7$  information types and each of them is categorical. **(1) Branch Types ( $k = 1$ ).** Based on where the operator  $f_v$  is located within the computational flow and its nearest node’s parameter name, we assign each node  $v$  to one of the unimodal branches or to the fusion branch. Specifically, “0” indicates that the operator is located at the multimodal fusion module, “1” indicates that the operator is in the modality-1’s unimodal feature extractor, ..., and “ $M$ ” indicates that the operator belongs to the modality- $M$ ’s unimodal feature extractor. **(2) Operator Types ( $k = 2$ ).** Each node  $v$  has its own function that transforms the input message it receives from previous nodes, which can influence the roles of layers around it. Therefore, it is assigned information about the operator type or layer type associated with  $f_v$ . Since the nodes have been categorized into three types: parametric, non-parametric, and input nodes, we define operator types into three families ac-

cordingly. The input nodes, regardless of their modality, are assigned the index "0". Parametric operator types include "Linear/Conv weights" (indexed as "1"), "bias" (indexed as "2"), "layer normalization" (indexed as "3"), "batch normalization" (indexed as "4"), and so on. There are several non-parametric operator types, including "sum", "concatenation", "element-wise dot product", "inner product", "multiplication", "Cartesian product". The ReLU operators usually appears together with parametric layer and thus are removed from the graph. **(3) Layer Levels ( $k = 3$ ).** The early layers typically learn fine-grained features, while deeper layers tend to learn higher-level features or concepts. This difference in feature extraction also influences the functionality of layers. Therefore, we enable each node to be aware of which level it should be within the computation hierarchy. To achieve this, we compute the relative layer level of each operator within its branch and use this information as the categorical index. We set a maximum of 6 levels for this dimension. **(4) Parameter Shapes ( $k = 4, 5, 6, 7$ ).** Different parameter sizes for the same layer necessitate varying densities of message during weight generation. This is akin to accommodating different sentence lengths in text generation. To control the weight generation density, it is necessary to feed the parameter shape information to the layer-role learning pipeline. Therefore, we let the last 4 dimensions contain the parameter shape information associated with parametric layers. Specifically, to simplify, we assume that all types of parametric layers have a 4D weight tensor—even layers with 2D weights, such as Dense layers, are considered as 4D by expanding the last two dimensions by one. The original size spaces of parameter shapes are linear—each size in  $[d1, d2, w1, w2]$  is taken from a range of  $[1, \text{max\_size}]$ , which lead to sparsity during learning. Therefore, we design a non-linear mapping table, which maps the original sizes into a scale space. We rank the exact layer size into several scales. Taking the linear layer as an example, if the weight tensor size falls within 32 to 512, 32 is mapped to the scale "0", 64 is mapped to "1", 128 is mapped to "2", 256 is mapped to "3", and 512 is mapped to "4". For those non-parametric operators and input nodes, we use a special token " $\langle \text{NPM} \rangle$ " which is treated as a held-out scale value of the scale space (e.g., indexed as "5" in the previous example).

### 8.4.3 Topology-aware HyperNetwork

As in Eq.(8.6), we propose to learn a bridge function that can jointly digest the two heterogeneity patterns (i.e. statistics and architecture heterogeneity). A challenge underlying this goal is that *how to balance and combine the two separate heterogeneity patterns*, such as which pattern is more crucial and whether there is any inter-pattern interactions.

We propose a **Topology-Aware HyperNetwork (TAHN)** to build such a bridge function. The key idea of TAHN is to encourage  $h(\cdot, \cdot; \phi)$  to capture the **implicit roles of each layer** within the neural architecture, which are then combined with layer-invariant agent-specific task information. This is inspired by an intuition that for a pair of layers from two different agents, if they act as similar roles within their models, would tend to have similar operations and weights.

Specifically, we formulate TAHN as a **two-stage** process

$$h(\mathcal{A}_i, \mathbf{c}_i; \phi) = \text{Comb}(\mathbf{c}_i, \text{Role}(\mathcal{A}_i; \phi_1); \phi_2) \quad (8.9)$$

where the first stage  $\text{Role}(\cdot; \phi_1)$  parameterized by  $\phi_1$  learns the implicit roles of layers such that layers across agents share a unified *layer-role embedding* space, and the second stage  $\text{Comb}(\cdot, \cdot; \phi_2)$  parameterized by  $\phi_2$  aims to combine the two heterogeneity patterns and directly generates the weights. We represent  $\phi = \{\phi_1, \phi_2\}$ .

#### 8.4.3.1 Stage One: Layer-role Encoder

In order to encode the implicit roles of layers, we consider two types of information. First, each layer’s configuration information are important to determine the layer role. For example, if two layers from different architecture both are the Conv layer and both are in the early level in the entire network, they tend to have similar filter and role during the computational flow. Such information is specified in  $\mathbf{Z}_i^{(0)}$ . Second, the position and contexts of each layer within the graphical computational flow is also important. For example, if two layers from different architectures are located in the same position in the same computational flow, they tend to have the same role. Such information is specified as the graphical structure  $\mathcal{V}_i, \mathcal{E}_i$  of the computational flow.

Incorporating the two types of information can be achieved by applying a Graph Neural Network (GNN) on the neural architecture graph  $\mathcal{A}_i = (\mathcal{V}_i, \mathcal{E}_i, \mathbf{Z}_i^{(0)})$  [252]. We formulate the layer-role encoder as a  $L$ -layer GNN

$$\begin{aligned} \mathbf{Z}_i^{(L)} &= \text{Role}(\mathcal{A}_i; \phi_1) \\ &:= g_L \circ g_{L-1} \circ \dots \circ g_1(\mathbf{Z}_i^{(0)}; \mathcal{V}_i, \mathcal{E}_i), \end{aligned} \quad (8.10)$$

where  $\mathbf{Z}_i^{(l)} = g_l(\mathbf{Z}_i^{(l-1)}; \mathcal{V}_i, \mathcal{E}_i, \psi_l)$  is the  $l$ -th GNN layer with trainable weights  $\psi_l$ . Every computational operator  $z_{i,v}^{(l)} \in \mathbf{Z}_i^{(l)}$  is encoded through message passing as

$$\begin{aligned} z_{i,v}^{(l)} &= \sigma(\mathbf{W}_{\text{self}}^{(l)} z_{i,v}^{(l-1)} + \mathbf{W}_{\text{in}}^{(l)} \sum_{(v',v) \in \mathcal{E}_i} z_{i,v'}^{(l-1)} \\ &\quad + \mathbf{W}_{\text{out}}^{(l)} \sum_{(v,v') \in \mathcal{E}_i} z_{i,v'}^{(l-1)} + \mathbf{b}^{(l)}), \end{aligned} \quad (8.11)$$

where  $\psi_l = \{\mathbf{W}_{\text{self}}^{(l)}, \mathbf{W}_{\text{in}}^{(l)}, \mathbf{W}_{\text{out}}^{(l)}, \mathbf{b}^{(l)}\}$  are trainable parameters.  $\phi_1 = \{\psi_1, \psi_2, \dots, \psi_L\}$ . The output of the final GNN layer  $\mathbf{Z}_i^{(L)} = \{z_{i,v}^{(L)} \in \mathbb{R}^S\}_{v \in \mathcal{V}_i}$  is a collection of **layer-role embeddings** for all parametric computational operators in  $\mathcal{A}_i$ , where  $S$  is the size of the layer-role embedding space.

#### 8.4.3.2 Stage Two: Role-aware Weight Generator

The layer-role information obtained from the first stage  $\mathbf{Z}_i^{(L)}$  is combined with agent-specific task information  $\mathbf{c}_i$  and then is used to generate the agent weights in a node-wise manner:  $\theta_i = \text{Comb}(\mathbf{c}_i, \mathbf{Z}_i^{(L)}; \phi_2)$ . We represent the agent model weights as a collection of weights for all computational operators  $\theta_i = \{\theta_{i,v} | v \in \mathcal{V}_i\}$ . Specifically,  $\theta_i$  is obtained using a HyperNetwork-based **node decoder**  $g_{\text{nodec}}$  applied to each node in the neural architecture graph. Let  $\theta_{i,v}$  denote the weights associated with the parametric computational operator  $v$  of agent  $i$ . Every  $\theta_{i,v}$  is computed

$$\theta_{i,v} := g_{\text{nodec}}(\mathbf{c}_i \oplus z_{i,v}^{(L)}; \phi_2), \quad \forall v \in \mathcal{V}_i \quad (8.12)$$

where  $\oplus$  denotes an operation (e.g. concatenation or summation) combining two embedding vectors: **layer-specific role** embedding  $z_{i,v}^{(L)}$  and a trainable agent-specific **layer-invariant task** embedding  $\mathbf{c}_i \in \mathbb{R}^F$ , where  $F$  is the size of task

embedding space.  $g_{\text{nodec}}$  is an MLP-based neural network in all experiments.

#### 8.4.4 Workflow

We let agents hold only their local personal models but the server holds the TAHN model that acts as a bridge for knowledge sharing. During training, agents perform their local model updates, and meanwhile, they communicate frequently with the server to help to optimize the TAHN.

The training workflow of FedMBSide is as follows. Each communication round  $r$  contains the following steps: **(1) Download.** The server predicts the weights  $\{\theta_i\}_{i \in \mathcal{N}_t} = \{h(\mathcal{A}_i, \mathbf{c}_i; \phi)\}_{i \in \mathcal{N}_t}$  or a subset of agents  $\mathcal{N}_t \subset [N]$ , using the current TAHN parameters and the current task embedding  $\mathbf{c}_i$  and conditioned on agent architecture graphs  $\mathcal{A}_i$ . Note that the graphs  $\mathcal{A}_i$  can be auto-recognized and constructed on the server based on the uploaded agent models before the first round starts, and therefore, they do not raise significant privacy issue. Each participant agent’s graph representation  $\mathcal{A}_i$  should be up-to-date—whenever it is changed during the collaboration, the agent re-conducts the graph construction process to refresh the  $\mathcal{A}_i$  representation. **(2) Local Updates:** Each selected agent  $i \in \mathcal{N}_t$  begins from the downloaded  $\theta_i$ , performs several local optimization steps based on its local data  $\mathcal{D}_i$ , and finally obtain new weights  $\tilde{\theta}_i$ . **(3) Upload.** Each agent send its update direction  $\Delta\theta_i = \tilde{\theta}_i - \theta_i$  to the server. **(4) Global Update and Knowledge Sharing.** The server computes the updates for TAHN inspired by the chain rule:

$$\begin{aligned} \Delta\mathbf{c}_i &= \nabla_{\mathbf{c}_i} \mathcal{L}_i(\theta_i) = \Delta\theta_i \cdot \nabla_{\mathbf{c}_i} \theta_i \\ \Delta\phi_2 &= \frac{1}{|\mathcal{N}_t|} \sum_{i \in \mathcal{N}_t} (\Delta\theta_i \cdot \nabla_{\phi_2} \theta_i) \\ \Delta\phi_1 &= \frac{1}{|\mathcal{N}_t|} \sum_{i \in \mathcal{N}_t} \left( \Delta\theta_i \cdot \nabla_{\mathbf{z}_i^{(L)}} \theta_i \cdot \nabla_{\phi_1} \mathbf{z}_i^{(L)} \right), \end{aligned} \tag{8.13}$$

where the multi-step local update direction  $\Delta\theta_i$  has replaced the original single-step local gradients  $\nabla_{\theta_i} \mathcal{L}_i(\theta_i)$  that are not efficient in FL. We perform an average of TAHN updates over agents  $\mathcal{N}_t$  for implicit knowledge sharing. Figure 8.4.1 shows an illustration of the workflow.

## 8.5 Experiments

### 8.5.1 Simulations

We evaluated our approach in *four* AMFL simulation scenarios. **SceneAMF** is constructed from the bimodal NYU-v2 dataset [253] that recognizes scenes from pairs of aligned RGB and depth images for these scenes. We create 80 agents covering 2 modality types, 3 types of input signals (RGB-only, depth-only, and RGB-depth bimodal inputs), and 40 types of neural architectures for local models. Each bimodal agent adopts one of the two traditional multimodal fusion strategies: *concatenation* and *element-wise product*. Each agent has its personal label space of size 50 sampled from a pool of 464 scenes. **ObjectAMF** is constructed from the bimodal ModelNet40 dataset [146] whose task is 3D object recognition from two views of 3D models. We create 112 agents in this simulation covering 56 types of neural architectures. For each bimodal agent, we employ one of three multimodal fusion strategies: *concatenation*, *average alignment*, and *tensor fusion* [20]. **EmotionAMF** is created from the CMU-MOSEI dataset [242] that focuses on emotion recognition task from real-world online videos consisting of 3 modalities (video, language script, and audio). Each video is annotated for the presence of 9 discrete emotions (angry, excited, fear, sad, surprised, frustrated, happy, disappointed, and neutral). The local tasks can be unimodal, bimodal, or trimodal. We employ three multimodal fusion strategies across agents: *average alignment*, *tensor fusion* [20], and *MultiEMO* [254]. **MnistAMF** is made from AVMnist [242] and MultiMnist [255] datasets, covering three modalities (image of style one, image of style two, and the audio for digit). MnistAMF uses 4 multimodal fusion strategies: *average alignment*, *tensor fusion*, *MultiEMO*, and *cross-attention fusion* [256].

The process of how we simulated the architectural heterogeneity are as follows. We will use how we created ObjectAMF as an example. We use the ModelNet40 dataset [146] as the source to create ObjectAMF. Derived from ModelNet40, ObjectAMF consists of  $N = 112$  agents with  $M = 2$  modality types and 56 different local neural architectures. (1) *Feature Extractors for the Video Modality*: We use MLP models extracting features for view1 or view2 images. For each view point,



Table 8.5.1: Summary of the 4 simulations of AMFL. **FS**: number of different multimodal fusion strategies. **W**: number of different widths for each unimodal encoder. **D**: number of different depths for each unimodal encoder. **T**: number of topology types for visual modality’s encoder. Acronyms for some modality types: **V** (Video), **L** (Language), **A** (Audio), **I1** (Style-one image), **I2** (Style-two image).


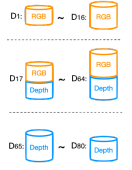


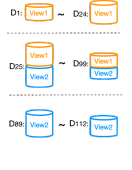








Simulation	#Agents	Input modalities per agent	#Target classes per agent	#Architectures (W, D, T, FS)
SceneAMF	80	{RGB}, {Depth}, or {RGB, Depth}	random 50 in 464 scenes	40 (1, 2, 2, 2)
ObjectAMF	112	{3D View1}, {3D View2}, or {3D View1, 3D View2}	random 5 in 40 objects	56 (2, 2, 1, 3)
EmotionAMF	90	{V}, {L}, {A}, {V, L}, {V, A}, {L, A}, or {V, L, A}	9 emotions	66 (1, 2, 1, 3)
MnistAMF	86	{I1}, {I2}, {A}, {I1,I2}, {I1,A}, {I2,A}, or {I1,I2,A}	random 4 digits in 0~9	86 (2, 1, 1, 4)

there are 4 different MLP configurations across agents, including *two different network widths* and *two different network depths*: 2-layer MLPs with a width of (64, 32) or (128, 64), and 3-layer MLPs with a width of (128, 64, 32) or (256, 128, 64). (2) *Multimodal Fusion Modules*. For the 48 unimodal agents, their feature extractors are followed by the same 2-layer MLP with a width of (64, 32), where there is no need for multimodal fusion. Then, for each of the 64 bimodal agents, we use one of the three multimodal fusion strategies: *Concatenation*, *Tensor Fusion* [20], and *Average Alignment*. As illustrated in Table 8.5.2 (row 2), those using alignment leverage a 2-layer MLP with (64, 32) to process the aligned features; those using concatenation fusion leverage a 3-layer MLP with (64, 64, 32) to process the concatenated features; and, those using the tensor fusion leverage a 3-layer MLP with a wider configuration of (128, 64, 32). (3) *Final Personal Layer*. The final layer on each agent is a dense layer with the output size of 5.

## 8.5.2 Baseline Methods

We compare FedMBSide with three families of baselines: (1) No Knowledge Sharing, i.e.,  $\mathcal{R}(\cdot)=0$ . **Local** separately trains local models that have different neural architectures, without any knowledge sharing among agents. (2) Feature-sharing FL approaches, such as **FedDistill** [33] and **FedGKD** [92], employ a public dataset at server that facilitate mutual knowledge distillation across heterogeneous architectures. Since the public dataset should not break much privacy, we let only 5% agents submitting only 5% of their samples to the server before the iteration starts. We do not use data-free distillation methods as they require the additional gen-

Table 8.5.2: Illustrations of Statistical and Architectural Heterogeneity in the Four Simulations of Architecture-agnostic Multimodal Federated Learning (AMFL).

Simulations	Modalities	Agent Heterogeneity Illustration	
		Statistical Heterogeneity	Architectural Heterogeneity
SceneAMF		<p>Local Datasets:</p> $\mathcal{D}_i \neq \mathcal{D}_j \neq \dots \neq \mathcal{D}_{90}$ $ \mathcal{D}_i  = n_i \sim \mathcal{N}(70, 10)$ 	<p><b>Feature Extractors</b></p> <p><b>For RGB Image</b></p> <ul style="list-style-type: none"> <li>CNN x 3</li> <li>CNN x 5</li> <li>ResNet x 3</li> <li>ResNet x 5</li> </ul> <p><b>For Depth Map</b></p> <ul style="list-style-type: none"> <li>VIT x 2</li> <li>VIT x 4</li> <li>CNN x 2</li> <li>CNN x 4</li> </ul>
			<p><b>Multimodal Fusion Modules</b></p> <ul style="list-style-type: none"> <li>Identity MLP [128,64,32] Client 1-4</li> <li>Identity MLP [64,32] Client 5-8</li> <li>Identity MLP [128,64,32] Client 9-12</li> <li>Identity MLP [64,32] Client 13-16</li> <li>Concatenation MLP [128,64,32] Client 17-18</li> <li>Element-wise Product MLP [64,32] Client 19</li> <li>Concatenation MLP [128,64,32] Client 20</li> <li>Element-wise Product MLP [64,32] Client 21-22</li> <li>Concatenation MLP [128,64,32] Client 23-24</li> <li>Element-wise Product MLP [64,32] Client 25</li> <li>Concatenation MLP [128,64,32] Client 26</li> <li>Element-wise Product MLP [64,32] Client 27-28</li> <li>...</li> <li>Concatenation MLP [128,64,32] Client 57-60</li> <li>Element-wise Product MLP [64,32] Client 61</li> <li>Concatenation MLP [128,64,32] Client 62</li> <li>Element-wise Product MLP [64,32] Client 63-64</li> <li>Identity MLP [128,64,32] Client 65-68</li> <li>Identity MLP [64,32] Client 69-72</li> <li>Identity MLP [128,64,32] Client 73-76</li> <li>Identity MLP [64,32] Client 77-80</li> </ul>
ObjectAMF	<p>1st View Point of 3D Shape</p>  <p>2nd View Point of 3D Shape</p> 	<p><math>\mathcal{D}_i \neq \mathcal{D}_j \neq \dots \neq \mathcal{D}_{112}</math></p> $ \mathcal{D}_i  = n_i \sim \mathcal{N}(300, 100)$ 	<p><b>For View1 Image</b></p> <ul style="list-style-type: none"> <li>MLP [64, 32]</li> <li>MLP [128, 64]</li> <li>MLP [128, 64, 32]</li> <li>MLP [256, 128, 64]</li> </ul> <p><b>For View2 Image</b></p> <ul style="list-style-type: none"> <li>MLP [64, 32]</li> <li>MLP [128, 64]</li> <li>MLP [128, 64, 32]</li> <li>MLP [256, 128, 64]</li> </ul>
			<p><b>Multimodal Fusion Modules</b></p> <ul style="list-style-type: none"> <li>Identity MLP [64,32] Client 1-6</li> <li>Identity MLP [64,32] Client 7-12</li> <li>Identity MLP [64,32] Client 13-18</li> <li>Identity MLP [64,32] Client 19-24</li> <li>Concatenation MLP [64,64,32] Client 25-26</li> <li>Average Alignment MLP [64,32] Client 27</li> <li>Tensor Fusion MLP [128,64,32] Client 28</li> <li>Concatenation MLP [64,64,32] Client 29</li> <li>Average Alignment MLP [64,32] Client 30-31</li> <li>Tensor Fusion MLP [128,64,32] Client 32</li> <li>...</li> <li>Concatenation MLP [64,64,32] Client 85</li> <li>Average Alignment MLP [64,32] Client 86</li> <li>Tensor Fusion MLP [128,64,32] Client 87-88</li> <li>Identity MLP [64,32] Client 89-94</li> <li>Identity MLP [64,32] Client 95-100</li> <li>Identity MLP [64,32] Client 101-106</li> <li>Identity MLP [64,32] Client 107-112</li> </ul>
MnistAMF	<p>Style1 of Digit</p>  <p>Style2 of Digit</p>  <p>Sound of Digits (Spectrogram)</p> 	<p><math>\mathcal{D}_i \neq \mathcal{D}_j \neq \dots \neq \mathcal{D}_{60}</math></p> $ \mathcal{D}_i  = n_i \sim \mathcal{N}(1800, 600)$ 	<p><b>For Style1 Image</b></p> <ul style="list-style-type: none"> <li>CNN [32, 64]</li> <li>CNN [16, 32]</li> </ul> <p><b>For Style2 Image</b></p> <ul style="list-style-type: none"> <li>CNN [32, 64]</li> <li>CNN [16, 32]</li> </ul> <p><b>For Sound</b></p> <ul style="list-style-type: none"> <li>CNN [32, 64]</li> <li>CNN [16, 32]</li> </ul>
			<p><b>Multimodal Fusion Modules</b></p> <ul style="list-style-type: none"> <li>Identity MLP [64,32] Client 1</li> <li>Identity MLP [64,32] Client 2,3</li> <li>Identity MLP [64,32] Client 4</li> <li>Identity MLP [64,32] Client 5,6</li> <li>MultiEMO Attention MLP [64,32] Client 7</li> <li>Average Alignment MLP [64,32] Client 8</li> <li>Tensor Fusion MLP [64,32] Client 9</li> <li>Cross-attention Fusion MLP [64,32] Client 10</li> <li>...</li> <li>MultiEMO Attention MLP [64,32] Client 23</li> <li>Average Alignment MLP [64,32] Client 24</li> <li>Tensor Fusion MLP [64,32] Client 25</li> <li>Cross-attention Fusion MLP [64,32] Client 26</li> <li>...</li> <li>MultiEMO Attention MLP [64,32] Client 39</li> <li>Average Alignment MLP [64,32] Client 40</li> <li>Tensor Fusion MLP [64,32] Client 41</li> <li>Cross-attention Fusion MLP [64,32] Client 42</li> <li>...</li> <li>MultiEMO Attention MLP [64,32] Client 55</li> <li>Average Alignment MLP [64,32] Client 56</li> <li>Tensor Fusion MLP [64,32] Client 57</li> <li>Cross-attention Fusion MLP [64,32] Client 58</li> </ul>
EmotionAMF	<p>Video</p>  <p>Audio</p>  <p>Language Script</p> 	<p><math>\mathcal{D}_i \neq \mathcal{D}_j \neq \dots \neq \mathcal{D}_{60}</math></p> $ \mathcal{D}_i  = n_i \sim \mathcal{N}(215, 40)$ 	<p><b>For Video</b></p> <ul style="list-style-type: none"> <li>VidT (fine-tune 1)</li> <li>VidT (fine-tune 3)</li> </ul> <p><b>For Audio</b></p> <ul style="list-style-type: none"> <li>SelfAttention x 1</li> <li>SelfAttention x 3</li> </ul> <p><b>For Language</b></p> <ul style="list-style-type: none"> <li>MLP [64, 32]</li> <li>MLP [128, 64, 32]</li> </ul>
			<p><b>Multimodal Fusion Modules</b></p> <ul style="list-style-type: none"> <li>Identity MLP [64,32] Client 1</li> <li>Identity MLP [64,32] Client 2</li> <li>Identity MLP [64,32] Client 3</li> <li>Identity MLP [64,32] Client 4</li> <li>Identity MLP [64,32] Client 5</li> <li>Identity MLP [64,32] Client 6</li> <li>Identity MLP [64,32] Client 7</li> <li>MultiEMO Attention MLP [64,32] Client 8</li> <li>Average Alignment MLP [64,32] Client 9</li> <li>...</li> <li>MultiEMO Attention MLP [64,32] Client 19</li> <li>Average Alignment MLP [64,32] Client 20</li> <li>Tensor Fusion MLP [64,32] Client 21</li> <li>...</li> <li>MultiEMO Attention MLP [64,32] Client 31</li> <li>Average Alignment MLP [64,32] Client 32</li> <li>Tensor Fusion MLP [64,32] Client 33</li> <li>...</li> <li>MultiEMO Attention MLP [64,32] Client 43,44</li> <li>Average Alignment MLP [64,32] Client 45,46</li> <li>Tensor Fusion MLP [64,32] Client 47,48</li> </ul>

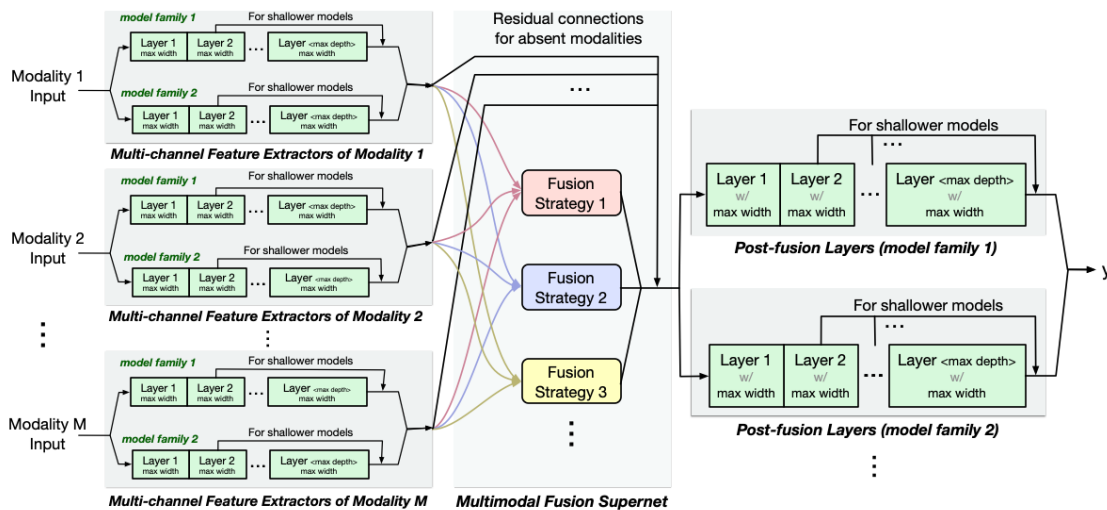


Figure 8.5.1: Baselines' supernet architecture on the server.

erators for multimodal data on the server. **(3)** Parameter-sharing FL approaches, such as **HeteroFL** [257] and extensions of the personalized FL or MFL methods (**APFL** [258], **HyperPFL** [30]) combined with Pruning techniques [259].

**Reproducibility:** **Local** allow agents to separately train their own models. There is no communication and knowledge transfer between agents. Hence a server is not needed. As agents may converge at varying speeds, to ensure a fair comparison, we adopt the practice of allowing each agent to showcase its optimal performance post-convergence. Subsequently, we calculate the average accuracy over the best performances of all agents. In **FedDistill** [33], the server utilizes an unlabeled public dataset to enable feature sharing among uploaded agent models through Knowledge Distillation (KD). When applying FedDistill to the AMFL scenarios, we make the following adjustments. *First*, the uploaded model of each agent is considered as a student. *Second*, before the start of federated training, a *public dataset* is collected from the agents such that every pattern of modality combinations has public samples available for agent-wise KD. Specifically, we randomly sample  $\rho\%$  agents from each group of agents having the same input structure (the same combination of modality types). Considering the potential privacy leakage of such data collection way, we mitigate this risk by deliberately keeping the hyperparameter  $\rho$  at a small value, specifically  $\rho = 5$ . In **FedGKD**, the server

employs a memory buffer to store multiple historical global models from recent rounds, which are used as teachers to guide the local model training via Knowledge Distillation (KD). Students are the local models trained on the data owners. Different from FedDistill, a public dataset is not needed in FedGKD since KD is performed locally using the local dataset. When we apply FedGKD to the AMFL scenarios, considering the limited storage capacity at local agents, the *number of downloaded teachers* is determined according to local memory budgets. In our implementations, each agent sends requests to the server to obtain  $s' \in [4, 8]$  teachers, and then, the server randomly sample  $s'$  teachers from its the buffer to give it back. **HeteroFL** [257] utilizes a global supernet on the server such that every local model is a sub-network of this supernet. Each local model is associated with a supermask indicating which parameters in the supernet are activated in the local model. Although HeteroFL is designed for single modality and assumes the same model family and the same network depths, we can adapt it to our AMFL scenarios as follows. We need to manually construct a supernet to cover all local networks. To obtain the supernet architecture, we merge all local neural networks following three rules: (1) If there are different network widths using the same model class (i.e. network topology) and the same depth, we can merge them into a single network having the same model family and depth, but using the *maximum width per layer* for the supernet. (2) If there are not only different widths but also different network depths using the same model class, we can also merge them into a single network, with the *maximum depths* for the supernet and add *residual connections* beginning from the layers corresponding to each local model’s end layer to the final layer. The merge of widths is similar to the first rule. (3) If two agents use different network topologies (i.e. model families) at the same module—for example, agent-A uses a CNN-based network for the video feature extractor and agent-B uses a ViT-based network for the video extractor, we cannot merge this part of their networks. Instead, we construct a two-channel network, where *each channel refers to a model family*, and the outputs of the two channels are added together. An overview of the implemented supernet is shown in Figure 8.5.1. *Second*, HeteroFL dynamically detects the complexity levels of agents during training, which are used to determine the how many parameters are needed at local agents and then the su-

permasks. Yet in AMFL the local models are given. Also, since HeteroFL does not sort the trained weights, the supermasks are fixed during training. The supermasks can be obtained according to the process of network merge. **APFL + Pruning** [258, 259] makes the following two adjustments. *First*, we extend APFL to the architecture-heterogeneous setting by straightforwardly combining AFPL with the supernet idea of HeteroFL. We let both global and local models of APFL use the supernet architecture instead. The supernet architecture is constructed using the same way as in HeteroFL. Since local datasets may not contain all modality types to feed to the supernet, those missing input modalities are imputed by zeros. *Second*, model aggregation on the server is guided by the supermasks of each agent. In contrast to HeteroFL, supermasks are dynamically updated at each round using structured magnitude Pruning, with the target compression ratio determined by the specific local network size. **HyperPFL + Pruning** solves the similar setup as APFL + Pruning, except using the HyperNetwork on the server to generate the local weights.

### 8.5.3 Setups

**Model Architecture Setup of Server’s TAHN.** FedMBSide leverages a global GNN network  $\phi_1$  to capture the roles of each parametric layer during the inference process along with each local neural architecture. Before applying GNN, we utilize seven embedding layers to embed each dimension of the categorical node features. For each node, the seven embeddings are summed together before being feed to GNN. Since the graph is directed, each GNN layer consists of a forward aggregation layer and then a backward aggregation layer. In each forward aggregation layer, every node/operator incorporates the computational flow structure from the operators it receives messages from; in each backward aggregation layer, every node/operator assimilates the computational flow structure from the operators that will receive messages from it. Both of them utilize a GRU cell to update the node features when aggregating neighborhood features. Then, as for the node decoders  $\phi_2$  for weight generation, we design a HyperNetwork using a mixture of experts. Different layer types (e.g., CNN, Dense, Embedding) may require distinct reconstruction methods to convert the role embedding space into the weight space.

Therefore, we automatically route different layer types to different HyperNetwork experts. For example, although a GNN may recognize that a Conv layer and a Dense layer play similar roles, we may use different reconstruction methods to decode their explicit weight space. In this way, the weights having different shapes are generated using different experts. This above technique can be considered as a Graph Neural Network with heterogeneous types of nodes, where nodes may use different node encoders. *Second*, we employ chunked HyperNetwork techniques together with upsampling and downsampling. When the target weight width does not match the generated size, we can downsample the generated weights if the target size is smaller than the generated size; otherwise, we can downsample the generated weights or iteratively generate multiple weight chunks of the layer.

**Evaluation Metrics.** We use two evaluation metrics. **ACC** refers to the average accuracy on the testing datasets over all agents. **CT** equals to the average time needed by each round of server-agent communication. CT is computed as the sum of the time costs of the three stages in each round: Stage1 is the average time for local training with 15 epochs; Stage2 is the communication between agent and server, including the time for agents uploading its local models and the time for agent downloading information from the server, such as the generated model weights, supernetwork weights, and supermask; and, Stage3 is the time for updating TAHN. We will run each experiment by 5 trials using different random seeds.

#### 8.5.4 Main Results

**Performance Comparison.** From Table 8.5.3, in general, the FL methods that utilize either features or weights for knowledge sharing outperform the non-knowledge-sharing Local, showing that the knowledge learned at local agents were successfully exchanged among agents and improved local performance. *However*, we observe that the feature-sharing baselines (rows 4-5) were sensitive to the modality gap and statistical heterogeneity in AMFL since feature distillation is not sufficiently robust to distribution shift; in addition to robustness issue, these methods relied on a public dataset with complete modalities, which raises privacy risk.

Table 8.5.3: Average performance comparison between different methods on all AMFL simulations. “s”: seconds. “\*”: privacy leakage risk.

Method	SceneAMF		ObjectAMF		EmotionAMF		MnistAMF	
	ACC $\uparrow$	CT $\downarrow$	ACC $\uparrow$	CT $\downarrow$	ACC $\uparrow$	CT $\downarrow$	ACC $\uparrow$	CT $\downarrow$
<b>Local</b>	76.56 $\pm$ 0.99	0	91.41 $\pm$ 0.90	0	67.83 $\pm$ 0.93	0	91.83 $\pm$ 1.32	0
<b>FedDistill*</b>	78.20 $\pm$ 0.84	59.2s	91.49 $\pm$ 0.33	32.4s	71.47 $\pm$ 0.56	52.9s	92.33 $\pm$ 1.21	42.4s
<b>FedGKD*</b>	81.32 $\pm$ 0.83	65.8s	93.82 $\pm$ 0.98	35.7s	73.97 $\pm$ 0.86	56.4s	93.91 $\pm$ 0.63	44.9s
<b>HeteroFL</b>	77.90 $\pm$ 0.85	86.4s	91.98 $\pm$ 0.83	57.3s	72.65 $\pm$ 1.33	65.7s	92.82 $\pm$ 0.89	68.2s
<b>APFL w/ Prune</b>	77.36 $\pm$ 0.83	98.2s	92.65 $\pm$ 0.92	59.4s	71.61 $\pm$ 0.32	73.5s	91.95 $\pm$ 0.80	89.3s
<b>HyperPFL w/ Prune</b>	78.92 $\pm$ 0.91	128.4s	92.39 $\pm$ 0.33	72.7s	72.85 $\pm$ 0.91	98.6s	92.67 $\pm$ 0.43	87.2s
<b>FedMBridge</b>	<b>83.92 <math>\pm</math> 0.95</b>	51.8s	<b>94.64 <math>\pm</math> 0.94</b>	35.3s	<b>75.96 <math>\pm</math> 0.83</b>	47.1s	<b>95.78 <math>\pm</math> 0.61</b>	38.2s

Also, we observe that the Parameter-sharing baselines (rows 6-9) significantly suffered from the architecture heterogeneity in AMFL: the more heterogeneous the local neural architectures, the less shareable weights between agents, and therefore, trained weights or gradients might be not sufficiently transferred among local models. *In contrast*, FedMBridge outperformed both feature- and Parameter-sharing baselines. This is because FedMBridge does not rely on public data; does not rely on knowledge transfer losses that is difficult to balance task shifts; and leverages TAHN to implicitly maximize sufficient weight sharing instead of explicit aggregation of unaligned weight spaces.

**Complexity Analysis.** Local has the best time efficiency as it requires no inter-agent knowledge sharing and thus no communication. Among all knowledge-sharing FL methods that require extra communication time, we have the following two observations. *First*, FedMBridge and feature-sharing baselines used the same time for uploading local models, but FedMBridge required less time to perform knowledge aggregation. This is because feature-sharing baselines need to compute again the gradients for each local models for feature distillation, but FedMBridge need only to compute the gradient for TAHN, which has less number of parameters. *Second*, we observe that FedMBridge and Parameter-sharing baselines used nearly the same time for knowledge aggregation, but FedMBridge requires less time for model uploading. This is because while FedMBridge upload the original local models, Parameter-sharing baselines upload a large-size supermask that indicates which parameters are shareable among agents. *Third*, while it is true that directly

Table 8.5.4: Ablation Study for FedMBridge using MnistAMF dataset.

FedMBridge Hyperparameters				ACC $\uparrow$	CT $\downarrow$
$L$	$\oplus$	$ \mathcal{N}_t /N$	w/wo FedGKD		
4	concat	0.25	$\beta = 0$	95.78	38.2s
<b>0</b>	concat	0.25	$\beta = 0$	87.75	<b>36.5s</b>
4	<b>sum</b>	0.25	$\beta = 0$	95.51	37.2s
4	concat	<b>0.40</b>	$\beta = 0$	<b>96.15</b>	44.6s
4	concat	0.25	$\beta = \mathbf{0.01}$	95.92	62.6s
4	concat	0.25	$\beta = \mathbf{0.05}$	<b>96.53</b>	62.6s

block-wise averaging model parameters (Stage 3) is more efficient than updating TAHN (Stage 3), the block-wise methods require defining extra large supermasks and supernetworks to be usable in Architecture-personalized MFL settings. Consequently, the time cost of Stage 2 in block-wise methods is longer than that of our method’s Stage 2. Therefore, the total TC of block-wise methods is higher than ours. While the time and memory costs per communication round of our method are lower than those of the baselines, we have observed that a drawback of our approach is its sensitivity to the number of agents selected at each round, affecting the total number of rounds required for convergence. This highlights the need for further improvements in the future.

**Ablation Study.** Table 8.5.4 reports the ablation study for our FedMBridge framework. We investigated the impacts of *four* components or factors in FedMBridge. **(1) Impact of TAHN Stage One.** The first stage of TAHN leverages a GNN-based network to learn layerwise role embeddings. Table 8.5.4 (row 4) removes the stage one by setting  $L = 0$ . The performance drop after this removal demonstrates the importance of this module. **(2) Impact of Role-Task Fusion Operator ( $\oplus$ ) in TAHN Stage Two.** The second stage of TAHN combines the layer-role embeddings with the task embedding. Table 8.5.4 (row 5) replaces the default concatenation with sum operation. The performance remains almost unchanged. **(3) Impact of Agent Selection ( $|\mathcal{N}_t|/N$ ).** Table 8.5.4 (row 6) slightly improves the performance by selecting more agents at each communication round.



Yet the efficiency drops along with the performance increase. This main reason of such communication time increase is that the number of input instances feed to the TAHN network increases. **(4) Combining with Feature-sharing FL.** Table 8.5.4 (rows 7-8) show that FedMBridge can be combined with the feature-sharing FL methods and can achieve better performance, where  $\beta$  denotes the importance of the distillation losses during the feature sharing on the server.

## 8.6 Conclusion

In this chapter, we focus on the novel Architecture-personalized MFL (AMFL) problem, which allows for free local multimodal neural architecture design with diversified multimodal fusion strategies. To attain an communication-efficient solution and improve beneficial parameter sharing in AMFL, we propose FedMBridge, which leverages a topology-aware hypernetwork as a bridge function to balance and digest the architecture heterogeneity and statistical heterogeneity. We conduct comprehensive experiments on several AMFL simulations and the result demonstrates the efficiency and effectiveness of FedMBridge over baselines.

# Chapter 9

## Conclusion and Future Directions

### 9.1 Conclusion

This thesis has targeted AI solutions that emulate the peer-to-peer learning dynamics observed in human societies—i.e., by collaborating and sharing experiences, individuals achieve a comprehensive understanding of the multimodal world, surpassing what any single agent could achieve alone. To accomplish this goal, we have systematically addressed several research topics focused on Multi-agent Collaborative Multimodal Machine Learning (CoMML) systems, with a particular emphasis on personalization for individual agents.

Specifically, we have investigated various personalization patterns found in the real world, such as modality, concept, domain, task, and architecture preferences. These patterns, individually or in combination, lead to specific heterogeneity among collaborative agents. We have discussed various critical challenges that have been underexplored or unexplored in prior work, proposing novel approaches to tackle them. These approaches are presented across six chapters, collectively advancing us toward our ultimate objective: Personalization-aware CoMML solutions. In Part I, we addressed the modality gap and concept shift problems and proposed graph-based, contrastive, and meta-learning approaches. In Part II, we addressed two additional personalization patterns—domain preferences and task type preferences, and explored advanced explicit inter-agent knowledge transfer approaches to enhance information sharing efficiency. In Part III, we investigated the novel

computational architecture gap problem and explored implicit information sharing approaches. Comprehensive experiments have been conducted in each chapter to validate the effectiveness of the proposed approaches, wherein we also created new datasets and simulations to provide a platform for future research in this direction.

## 9.2 Future Directions

**Scalability of CoMML.** With the increasing number of connected devices, scalability becomes a critical aspect. The scalability of CoMML should be addressed across various dimensions, including the growing number of agents participating in collaboration, the expanding diversity of domains and environmental backgrounds where agents operate (e.g., from multimedia to medical), and the heightened discrepancy in concept definitions across agents. The importance of these scalability aspects is that it helps in the system to *work gracefully* without any undue delay and unproductive resource consumption and makes a good use of the available resources. Additionally, to achieve a *unified* system, there is a desire to scale CoMML to accommodate a broader variety of modality types and downstream tasks. Presently, our evaluations have primarily focused combining classification and cross-modal generation tasks, while other popular multimodal tasks like Visual Question Answering and text-visual grounding can also be integrated into collaboration. Furthermore, scaling CoMML to accommodate a wider variety of modality types will yield more benefits and foster *deeper cognition* about the physical world. For instance, integrating 3D scene modalities into the existing audio-video understanding CoMML framework can enhance our system’s capability for spatial understanding in dynamic virtual reality applications.

**Lifelong CoMML.** Agents’ behaviors can evolve over time, resulting in dynamic changes in their local data distributions, modality types, task types, and concept spaces of interest. Hence the dynamics of collaboration among agents, including which agents provide higher levels of assistance to each other, can also change over time. A key objective of lifelong CoMML is to develop systems capable of continuous learning, enhancing performance steadily over time and adapting to *dynamic*

*inter-agent collaboration* environments without forgetting previous learning.

**Explainable and Logic-aware CoMML.** With the recent surge in explainable AI and the need to understand human-like logic rules and reasoning processes in AI models, it's worth exploring whether multi-AI-agent collaboration can enhance the symbolic reasoning ability of individual AI systems. Intuitively, humans can learn from each other to acquire commonsense-aligned logic rules, facilitating improved communication, even if our individual thinking processes differ slightly. However, CoMML's multi-agent interactive training framework usually makes it challenging for explaining the decision-making rationale behind the resulting models. Moreover, different agents may capture highly distinct concepts and develop their personal reasoning processes, akin to the diverse thinking processes observed among individual humans. That is, agent models might learn *different rule sets*. To address these challenges, in the future, we might leverage CoMML with symbolic neural networks or concept learning to acquire personal yet globally aligned logic rules and reasoning processes.

### 9.3 Broader Impact

**Learning at Home.** CoMML offers the convenience of learning at home, allowing users to participate in knowledge transfer anytime and anywhere, without risking the *privacy* of data collected at home. Therefore, CoMML can have significant practical applications in modern life, such as in-home health monitoring for the aging population worldwide. With the assistance of CoMML, users at home, equipped with diverse modalities and tasks, can explore collaboration opportunities to expand their knowledge.

**Artificial Internet of Things (AIoT).** Internet of Things have widely penetrated in different aspects of modern life and many intelligent IoT services and applications are emerging. AIoT applications often deploy different types of smart sensors or devices that generate data from different modalities (e.g., sensory, visual, and audio). For example, in one smart home, activities of a person can be

recorded by body sensors in a smartwatch worn by the person, and also by a video camera in the room at the same time. Meanwhile, for smart homes with different device setups, some of them may have multimodal local data (i.e., multimodal agents) while the others may have unimodal local data (i.e., unimodal agents). With huge amounts of smart devices connected together in IoT, we are able to get access to massive user data to yield insights, train task-specified machine learning models and ultimately provide high-quality smart services and products. Despite of rare existing work, CoMML will make substantial advances in all AIoT applications in our modern life, including Internet of Medical Things (IoMT), Internet of Augmented Reality Things (IoART), intelligent transportation infrastructure, etc.

**Applications on Multimodal Large Foundation Models.** CoMML can be used to help to develop Multimodal Large Foundation Models (MLFM). Here, we provide three example use cases to show our thoughts on the connections between CoMML and MLFM. **(1) Case1:** CoMML can be used to pre-train MLFL. MLFL need extensive and diverse data during pretraining. However, certain data types, such as medical, life, and document data, might be sensitive and private, thereby preventing MLFL from accessing them. By using CoMML, MLFL pretraining can be conducted in a multi-party collaboration manner, thereby indirectly benefiting from private data. Also, by being learned in CoMML systems, MLFL can reap the advantages of scaling up the number of agents, even if each individual agent has a limited number of samples. **(2) Case2:** Given a pretrained MLFL, CoMML can be used to maintain the generalization ability of MLFL in a lifelong manner. New concepts emerge day by day, thus the abilities of MLFL should be continuously developed in a lifelong manner. We believe that a pretrained MLFL would still need to evolve over time to incorporate and generalize more up-to-date knowledge. Such up-to-date knowledge is typically generated by humans, and therefore, can be challenging to obtain due to privacy concerns. By employing CoMML on post-training, MLFL can evolve over time. **(3) Case3:** When a pretrained MLFL is utilized to serve each user through fine-tuning with local user-specific data, a challenge arises when the performance of local MLFL fine-tuning is constrained by the privacy and limited sample size of local data. CoMML provides a way to fine-

tune MLFL using more data across distributed agents without compromising data privacy, referred to as Collaborative Fine-tuning for MLFL. Consider a scenario where there is a pretrained MLFL model held by agent-A, who needs the fine-tuned model, while agent-B and agent-C lack sufficient memory budgets and can only afford a traditional multimodal model or a scaled-down version of the full MLFL. There is architecture gap between agent-A, B, and C. During Federated Fine-tuning of agent-A's MLFL model, our proposed approach can facilitate knowledge transfer among models of agent-A, B, and C. Thus the fine-tuning process on agent-A can benefit from the data from agent-B and agent-C. These three cases suggest that, if pretrained MLFL is available, it is still necessary to consider CoMML, since CoMML can play an important role to let the MLFL benefit from being more private and thus achieve better results rather than without CoMML.

# Bibliography

- [1] Sébastien Bubeck et al. “Sparks of artificial general intelligence: Early experiments with gpt-4”. In: *arXiv preprint arXiv:2303.12712* (2023).
- [2] Nanyi Fei et al. “Towards artificial general intelligence via a multimodal foundation model”. In: *Nature Communications* 13.1 (2022), p. 3094.
- [3] Mingkun Xu et al. “A Unified Structured Framework for AGI: Bridging Cognition and Neuromorphic Computing”. In: *International Conference on Artificial General Intelligence*. Springer, 2023, pp. 345–356.
- [4] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. “Multimodal machine learning: A survey and taxonomy”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41.2 (2018), pp. 423–443.
- [5] Charlotte Højholt and Dorte Kousholt. “Developing knowledge through participation and collaboration: Research as mutual learning processes”. In: *Annual Review of Critical Psychology (Online)* 16.special issue (2019), pp. 575–604.
- [6] Yaochen Hu et al. “FDML: A collaborative machine learning framework for distributed features”. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2019, pp. 2232–2240.
- [7] Tao Shen et al. “Federated mutual learning: a collaborative machine learning method for heterogeneous data, models, and objectives”. In: *Frontiers of Information Technology & Electronic Engineering* 24.10 (2023), pp. 1390–1402.
- [8] Yashar Talebirad and Amirhossein Nadiri. “Multi-agent collaboration: Harnessing the power of intelligent llm agents”. In: *arXiv preprint arXiv:2306.03314* (2023).

- [9] Guile Wu and Shaogang Gong. “Peer collaborative learning for online knowledge distillation”. In: *Proceedings of the AAAI Conference on artificial intelligence*. Vol. 35. 12. 2021, pp. 10302–10310.
- [10] Nuzulla Mamat and Norazah Yusof. “Learning style in a personalized collaborative learning framework”. In: *Procedia-Social and Behavioral Sciences* 103 (2013), pp. 586–594.
- [11] Jing Gao et al. “A survey on deep learning for multimodal data fusion”. In: *Neural Computation* 32.5 (2020), pp. 829–864.
- [12] Junqing Le et al. “Federated continuous learning with broad network architecture”. In: *IEEE Transactions on Cybernetics* 51.8 (2021), pp. 3874–3888.
- [13] Hangyu Zhu, Haoyu Zhang, and Yaochu Jin. “From federated learning to federated neural architecture search: a survey”. In: *Complex & Intelligent Systems* 7.2 (2021), pp. 639–657.
- [14] Jin Shin and Sang Joon Kim. “A Mathematical Theory of Communication”. In: 2006.
- [15] Dana Lahat, Tülay Adalı, and Christian Jutten. “Multimodal Data Fusion: An Overview of Methods, Challenges, and Prospects”. In: *Proceedings of the IEEE* 103.9 (2015), pp. 1449–1477. DOI: 10.1109/JPROC.2015.2460697. URL: <https://doi.org/10.1109/JPROC.2015.2460697>.
- [16] Yiyuan Zhang et al. “Meta-transformer: A unified framework for multimodal learning”. In: *arXiv preprint arXiv:2307.10802* (2023).
- [17] Peng Xu, Xiatian Zhu, and David A. Clifton. “Multimodal Learning with Transformers: A Survey”. In: *ArXiv abs/2206.06488* (2022).
- [18] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.
- [19] Yang Liu et al. “A Survey of Visual Transformers”. In: *ArXiv abs/2111.06091* (2021).
- [20] Amir Zadeh et al. “Tensor Fusion Network for Multimodal Sentiment Analysis”. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2017, pp. 1103–1114.



- [21] Zhun Liu et al. “Efficient Low-rank Multimodal Fusion With Modality-Specific Factors”. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2018, pp. 2247–2256.
- [22] Hao Chen and Feihong Shen. “Hierarchical Cross-modal Transformer for RGB-D Salient Object Detection”. In: *ArXiv abs/2302.08052* (2023).
- [23] Brendan McMahan et al. “Communication-efficient learning of deep networks from decentralized data”. In: *Artificial intelligence and statistics*. PMLR. 2017, pp. 1273–1282.
- [24] Tian Li et al. “Federated optimization in heterogeneous networks”. In: *Proceedings of Machine Learning and Systems 2* (2020), pp. 429–450.
- [25] Alysa Ziyang Tan et al. “Towards personalized federated learning”. In: *arXiv preprint arXiv:2103.00710* (2021).
- [26] Canh T Dinh, Nguyen H Tran, and Tuan Dung Nguyen. “Personalized federated learning with moreau envelopes”. In: *arXiv preprint arXiv:2006.08848* (2020).
- [27] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. “Personalized federated learning: A meta-learning approach”. In: *arXiv preprint arXiv:2002.07948* (2020).
- [28] Virginia Smith et al. “Federated multi-task learning”. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 2017, pp. 4427–4437.
- [29] Binbin Guo et al. “PFL-MoE: Personalized Federated Learning Based on Mixture of Experts”. In: *Web and Big Data: 5th International Joint Conference, APWeb-WAIM 2021, Guangzhou, China, August 23–25, 2021, Proceedings, Part I*. 2021, pp. 480–486.
- [30] Aviv Shamsian et al. “Personalized federated learning using hypernetworks”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 9489–9502.
- [31] Naichen Shi et al. “Fed-ensemble: Improving generalization through model ensembling in federated learning”. In: *arXiv preprint arXiv:2107.10663* (2021).

- [32] Dinh C Nguyen et al. “Federated learning for internet of things: A comprehensive survey”. In: *IEEE Communications Surveys & Tutorials* 23.3 (2021), pp. 1622–1658.
- [33] H Brendan McMahan et al. “Learning Differentially Private Recurrent Language Models”. In: *International Conference on Learning Representations*. 2018.
- [34] Eduard Gorbunov, Filip Hanzely, and Peter Richtárik. “Local sgd: Unified theory and new efficient methods”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2021, pp. 3556–3564.
- [35] Sebastian Urban Stich. “Local SGD Converges Fast and Communicates Little”. In: *ICLR 2019-International Conference on Learning Representations*. CONF. 2019.
- [36] Farzin Haddadpour and Mehrdad Mahdavi. “On the convergence of local descent methods in federated learning”. In: *arXiv preprint arXiv:1910.14425* (2019).
- [37] Ahmed Khaled, Konstantin Mishchenko, and Peter Richtárik. “Tighter theory for local SGD on identical and heterogeneous data”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2020, pp. 4519–4529.
- [38] Xiang Li et al. “On the convergence of fedavg on non-iid data”. In: *arXiv preprint arXiv:1907.02189* (2019).
- [39] Chelsea Finn, Pieter Abbeel, and Sergey Levine. “Model-agnostic meta-learning for fast adaptation of deep networks”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 1126–1135.
- [40] Yuchen Zhao, Payam Barnaghi, and Hamed Haddadi. “Multimodal federated learning”. In: *arXiv preprint arXiv:2109.04833* (2021).
- [41] Baochen Xiong et al. “A Unified Framework for Multi-modal Federated Learning”. In: *Neurocomputing* (2022).
- [42] Zhiyuan Fang et al. “SEED: SELF-SUPERVISED DISTILLATION FOR VISUAL REPRESENTATION”. In: *9th International Conference on Learning Representations, ICLR 2021*. 2021.

- [43] Guocong Song and Wei Chai. “Collaborative learning for deep neural networks”. In: *Advances in neural information processing systems* 31 (2018).
- [44] Qiushan Guo et al. “Online knowledge distillation via collaborative learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 11020–11029.
- [45] Chuanguang Yang, Zhulin An, and Yongjun Xu. “Multi-view contrastive learning for online knowledge distillation”. In: *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2021, pp. 3750–3754.
- [46] Chuanguang Yang et al. “Mutual contrastive learning for visual representation learning”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 3. 2022, pp. 3045–3053.
- [47] Chao Peng et al. “Fedgm: Heterogeneous federated learning via generative learning and mutual distillation”. In: *European Conference on Parallel Processing*. Springer. 2023, pp. 339–351.
- [48] Linjun Luo and Xinglin Zhang. “Federated Split Learning Via Mutual Knowledge Distillation”. In: *IEEE Transactions on Network Science and Engineering* (2024).
- [49] Tao Shen et al. “Federated mutual learning”. In: *arXiv preprint arXiv:2006.16765* (2020).
- [50] Chuhan Wu et al. “Communication-efficient federated learning via knowledge distillation”. In: *Nature communications* 13.1 (2022), p. 2032.
- [51] Jake Snell, Kevin Swersky, and Richard S Zemel. “Prototypical networks for few-shot learning”. In: *arXiv preprint arXiv:1703.05175* (2017).
- [52] Boris N Oreshkin, Pau Rodriguez, and Alexandre Lacoste. “Tadam: Task dependent adaptive metric for improved few-shot learning”. In: *arXiv preprint arXiv:1805.10123* (2018).
- [53] Oriol Vinyals et al. “Matching networks for one shot learning”. In: *arXiv preprint arXiv:1606.04080* (2016).
- [54] Flood Sung et al. “Learning to compare: Relation network for few-shot learning”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 1199–1208.

- [55] Zhenguo Li et al. “Meta-SGD: Learning to Learn Quickly for Few Shot Learning”. In: *CoRR* abs/1707.09835 (2017). arXiv: 1707.09835. URL: <http://arxiv.org/abs/1707.09835>.
- [56] Jaesik Yoon et al. “Bayesian model-agnostic meta-learning”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 7332–7342.
- [57] Alex Nichol, Joshua Achiam, and John Schulman. “On first-order meta-learning algorithms”. In: *arXiv preprint arXiv:1803.02999* (2018).
- [58] Chen Xing et al. “Adaptive cross-modal few-shot learning”. In: *arXiv preprint arXiv:1902.07104* (2019).
- [59] Jiayi Chen and Aidong Zhang. “HetMAML: Task-Heterogeneous Model-Agnostic Meta-Learning for Few-Shot Learning Across Modalities”. In: *arXiv preprint arXiv:2105.07889* (2021).
- [60] Manoj Ghuhana Arivazhagan et al. *Federated Learning with Personalization Layers*. 2019. arXiv: 1912.00818 [cs.LG].
- [61] Canh T Dinh, Nguyen Tran, and Josh Nguyen. “Personalized federated learning with moreau envelopes”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 21394–21405.
- [62] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. “Adaptive personalized federated learning”. In: *arXiv preprint arXiv:2003.13461* (2020).
- [63] Wonyong Jeong and Sung Ju Hwang. *Factorized-FL: Agnostic Personalized Federated Learning with Kernel Factorization & Similarity Matching*. 2022. arXiv: 2202.00270 [cs.LG].
- [64] Yasmin SarcheshmehPour et al. “Networked Federated Multi-Task Learning”. In: *arXiv preprint arXiv:2105.12769* (2021).
- [65] Canh T Dinh et al. “A New Look and Convergence Rate of Federated Multi-Task Learning with Laplacian Regularization”. In: *arXiv preprint arXiv:2102.07148* (2021).
- [66] Vinod Kumar Chauhan et al. “A brief review of hypernetworks in deep learning”. In: *arXiv preprint arXiv:2306.06955* (2023).

- [67] Risto Vuorio et al. “Multimodal Model-Agnostic Meta-Learning via Task-Aware Modulation”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 1–12.
- [68] Huaxiu Yao et al. “Automated relational meta-learning”. In: *arXiv preprint arXiv:2001.00745* (2020).
- [69] Haozhao Wang et al. “DaFKD: Domain-Aware Federated Knowledge Distillation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 20412–20421.
- [70] Huan Gao et al. “Cross-Domain Correlation Distillation for Unsupervised Domain Adaptation in Nighttime Semantic Segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 9913–9923.
- [71] Othmane Marfoq et al. “Federated multi-task learning under a mixture of distributions”. In: *Advances in Neural Information Processing Systems 34* (2021).
- [72] Junyuan Hong et al. “Efficient Split-Mix Federated Learning for On-Demand and In-Situ Customization”. In: *International Conference on Learning Representations (ICLR 2022)*. 2022.
- [73] Jiayi Chen and Aidong Zhang. “FedMSplit: Correlation-adaptive federated multi-task learning across multimodal split networks”. In: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2022, pp. 87–96. URL: <https://dl.acm.org/doi/abs/10.1145/3534678.3539384>.
- [74] Sheng Shen et al. *Scaling Vision-Language Models with Sparse Mixture of Experts*. 2023. arXiv: 2303.07226 [cs.CV].
- [75] Tianyu Chen et al. “Task-Specific Expert Pruning for Sparse Mixture-of-Experts”. In: *arXiv preprint arXiv:2206.00277* (2022).
- [76] Carlos Riquelme et al. “Scaling vision with sparse mixture of experts”. In: *Advances in Neural Information Processing Systems 34* (2021), pp. 8583–8595.
- [77] Qifei Wang et al. “Multi-path neural networks for on-device multi-domain visual classification”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2021, pp. 3019–3028.

- [78] Liping Yi et al. “FedMoE: Data-Level Personalization with Mixture of Experts for Model-Heterogeneous Personalized Federated Learning”. In: *arXiv preprint arXiv:2402.01350* (2024).
- [79] Hu Wang et al. “Learnable cross-modal knowledge distillation for multi-modal learning with missing modality”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2023, pp. 216–226.
- [80] Sujin Jang et al. “STXD: Structural and Temporal Cross-Modal Distillation for Multi-View 3D Object Detection”. In: *Advances in Neural Information Processing Systems* 36 (2024).
- [81] Mengxi Chen et al. “Enhanced multimodal representation learning with cross-modal kd”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 11766–11775.
- [82] Zimian Wei et al. “Cross-modal knowledge distillation in multi-modal fake news detection”. In: *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2022, pp. 4733–4737.
- [83] Alex Andonian, Shixing Chen, and Raffay Hamid. “Robust cross-modal representation learning with progressive self-distillation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 16430–16441.
- [84] Saurabh Gupta, Judy Hoffman, and Jitendra Malik. “Cross modal distillation for supervision transfer”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2827–2836.
- [85] Peijun Bao et al. “Local-Global Multi-Modal Distillation for Weakly-Supervised Temporal Video Grounding”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 2. 2024, pp. 738–746.
- [86] Anil Rahate et al. “Multimodal co-learning: Challenges, applications with datasets, recent advances and future directions”. In: *Information Fusion* 81 (2022), pp. 203–239.
- [87] Qiying Yu et al. “Multimodal Federated Learning via Contrastive Representation Ensemble”. In: *arXiv preprint arXiv:2302.08888* (2023).

- [88] Yao Ma et al. “Multimodality in meta-learning: A comprehensive survey”. In: *Knowledge-Based Systems* 250 (2022), p. 108976.
- [89] Mengmeng Ma et al. “Smil: Multimodal learning with severely missing modality”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 3. 2021, pp. 2302–2310.
- [90] Donglin Jiang, Chen Shan, and Zhihui Zhang. “Federated learning algorithm based on knowledge distillation”. In: *2020 International Conference on Artificial Intelligence and Computer Engineering (ICAICE)*. IEEE. 2020, pp. 163–167.
- [91] Ruihong Yang, Junchao Tian, and Yu Zhang. “Regularized Mutual Learning for Personalized Federated Learning”. In: *Asian Conference on Machine Learning*. PMLR. 2021, pp. 1521–1536.
- [92] Dezhong Yao et al. “Fed GKD: Towards Heterogeneous Federated Learning via Global Knowledge Distillation”. In: *IEEE Transactions on Computers* (2023).
- [93] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. “Data-free knowledge distillation for heterogeneous federated learning”. In: *International conference on machine learning*. PMLR. 2021, pp. 12878–12889.
- [94] Lin Zhang et al. “Fine-tuning global model via data-free knowledge distillation for non-iid federated learning”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10174–10183.
- [95] Enmao Diao, Jie Ding, and Vahid Tarokh. “HeteroFL: Computation and Communication Efficient Federated Learning for Heterogeneous Clients”. In: *International Conference on Learning Representations*. 2021.
- [96] Minjae Kim et al. “DepthFL: Depthwise Federated Learning for Heterogeneous Clients”. In: *The Eleventh International Conference on Learning Representations*. 2023.
- [97] Rong Dai et al. “DisPFL: Towards Communication-Efficient Personalized Federated Learning via Decentralized Sparse Training”. In: *International Conference on Machine Learning*. PMLR. 2022, pp. 4587–4604.

- [98] Rui Liu et al. “Federated Graph Neural Networks: Overview, Techniques, and Challenges”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2024).
- [99] Yuanyishu Tian et al. “FedBERT: When Federated Learning Meets Pre-training”. In: *ACM Trans. Intell. Syst. Technol.* 13.4 (2022). ISSN: 2157-6904. DOI: 10.1145/3510033. URL: <https://doi.org/10.1145/3510033>.
- [100] Huan Li Xin’ao Wang, Ke Chen, and Lidan Shou. “FEDBFPT: An efficient federated learning framework for BERT further pre-training”. In: *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*. 2023, pp. 4344–4352.
- [101] Agrin Hilmkil et al. “Scaling federated learning for fine-tuning of large language models”. In: *International Conference on Applications of Natural Language to Information Systems*. Springer. 2021, pp. 15–23.
- [102] Zhuo Zhang et al. “FedPETuning: When federated learning meets the parameter-efficient tuning methods of pre-trained language models”. In: *Annual Meeting of the Association of Computational Linguistics 2023*. Association for Computational Linguistics (ACL). 2023, pp. 9963–9977.
- [103] Liam H Fowl et al. “Decepticons: Corrupted Transformers Breach Privacy in Federated Learning for Language Models”. In: *NeurIPS ML Safety Workshop*. 2022.
- [104] Zheng Xu et al. “Federated Learning of Gboard Language Models with Differential Privacy”. In: *arXiv preprint arXiv:2305.18465* (2023).
- [105] Mingbin Xu et al. “Training Large-Vocabulary Neural Language Models by Private Federated Learning for Resource-Constrained Devices”. In: *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2023, pp. 1–5.
- [106] Addi Ait-Mlouk et al. “FedBot: Enhancing Privacy in Chatbots with Federated Learning”. In: *arXiv preprint arXiv:2304.03228* (2023).
- [107] Jianyi Zhang et al. “Towards building the federatedGPT: Federated instruction tuning”. In: *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2024, pp. 6915–6919.



- [108] Rui Ye et al. *OpenFedLLM: Training Large Language Models on Decentralized Private Data via Federated Learning*. 2024. arXiv: 2402.06954 [cs.LG].
- [109] Huimin Zeng et al. *Federated Recommendation via Hybrid Retrieval Augmented Generation*. 2024. arXiv: 2403.04256 [cs.IR].
- [110] Zhengming Ding, Handong Zhao, and Yun Fu. *Learning representation for multi-view data analysis: models and applications*. Springer, 2018.
- [111] Amir Zadeh et al. “Memory fusion network for multi-view sequential learning”. In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.
- [112] Amir Zadeh et al. “Multi-attention recurrent network for human communication comprehension”. In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018. URL: <https://github.com/A2Zadeh/CMU-MultimodalSDK>.
- [113] Amir Zadeh et al. “Mosi: multimodal corpus of sentiment intensity and subjectivity analysis in online opinion videos”. In: *arXiv preprint arXiv:1606.06259* (2016).
- [114] Lei Cai et al. “Deep adversarial learning for multi-modality missing data completion”. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM. 2018.
- [115] Luan Tran et al. “Missing modalities imputation via cascaded residual autoencoder”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 1405–1414.
- [116] Lei Zhang et al. “Multi-view missing data completion”. In: *IEEE Transactions on Knowledge and Data Engineering* 30.7 (2018), pp. 1296–1309.
- [117] Qianqian Wang et al. “Partial Multi-view Clustering via Consistent GAN”. In: *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2018.
- [118] Qiuling Suo et al. “Metric learning on healthcare data with incomplete modalities”. In: *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press. 2019, pp. 3534–3540.
- [119] Yao-Hung Hubert Tsai et al. “Learning factorized multimodal representations”. In: *arXiv preprint arXiv:1806.06176* (2018).

- [120] Lei Yuan et al. “Multi-source feature learning for joint analysis of incomplete multiple heterogeneous neuroimaging data”. In: *NeuroImage* 61.3 (2012), pp. 622–632.
- [121] Shuo Xiang et al. “Multi-source learning with block-wise missing data for Alzheimer’s disease prediction”. In: *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2013, pp. 185–193.
- [122] Yan Li et al. “Multi-Task Learning based Survival Analysis for Predicting Alzheimer’s Disease Progression with Multi-Source Block-wise Missing Data”. In: *SDM*. 2018, pp. 288–296.
- [123] Mingxia Liu et al. “Multi-Hypergraph Learning for Incomplete Multimodality Data”. In: *IEEE journal of biomedical and health informatics* 22.4 (2017), pp. 1197–1208.
- [124] Hang Su et al. “Multi-view convolutional neural networks for 3d shape recognition”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 945–953.
- [125] Yifan Feng et al. “GVCNN: Group-view convolutional neural networks for 3D shape recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 264–272.
- [126] Ziyuan Zhao et al. “An image-text consistency driven multimodal sentiment analysis approach for social media”. In: *Information Processing & Management* 56.6 (2019), p. 102097.
- [127] Dhanesh Ramachandram and Graham W Taylor. “Deep multimodal learning: A survey on recent advances and trends”. In: *IEEE Signal Processing Magazine* 34.6 (2017), pp. 96–108.
- [128] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. “Challenges and applications in multimodal machine learning”. In: *The Handbook of Multimodal-Multisensor Interfaces*. Association for Computing Machinery and Morgan & Claypool, 2018, pp. 17–48.
- [129] Bruce Thompson. “Canonical correlation analysis”. In: *Encyclopedia of statistics in behavioral science* (2005).

- [130] Pei Ling Lai and Colin Fyfe. “Kernel and nonlinear canonical correlation analysis”. In: *International Journal of Neural Systems* 10.05 (2000), pp. 365–377.
- [131] Galen Andrew et al. “Deep canonical correlation analysis”. In: *International conference on machine learning*. 2013, pp. 1247–1255.
- [132] John Morris et al. “Cascaded multi-view canonical correlation (CaMCCo) for early diagnosis of Alzheimer’s disease via fusion of clinical, imaging and omic Features”. In: *Scientific reports* (2017).
- [133] Quinten McNamara, Alejandro De La Vega, and Tal Yarkoni. “Developing a comprehensive framework for multimodal feature extraction”. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2017, pp. 1567–1574.
- [134] Zhihong Zeng et al. “A survey of affect recognition methods: Audio, visual, and spontaneous expressions”. In: *IEEE transactions on pattern analysis and machine intelligence* 31.1 (2008), pp. 39–58.
- [135] Jiquan Ngiam et al. “Multimodal deep learning”. In: *Proceedings of the 28th international conference on machine learning (ICML-11)*. 2011, pp. 689–696.
- [136] Sidney K D’mello and Jacqueline Kory. “A review and meta-analysis of multimodal affect detection systems”. In: *ACM Computing Surveys (CSUR)* 47.3 (2015), p. 43.
- [137] Mohammad Soleymani, Maja Pantic, and Thierry Pun. “Multimodal emotion recognition in response to videos”. In: *IEEE transactions on affective computing* 3.2 (2011), pp. 211–223.
- [138] Verónica Pérez-Rosas, Rada Mihalcea, and Louis-Philippe Morency. “Utterance-level multimodal sentiment analysis”. In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2013, pp. 973–982.
- [139] Soujanya Poria et al. “Convolutional MKL based multimodal emotion recognition and sentiment analysis”. In: *2016 IEEE 16th international conference on data mining (ICDM)*. IEEE. 2016, pp. 439–448.

- [140] Haohan Wang et al. “Select-additive learning: Improving generalization in multimodal sentiment analysis”. In: *2017 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE. 2017, pp. 949–954.
- [141] Yifan Feng et al. “Hypergraph neural networks”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 2019, pp. 3558–3565.
- [142] Petar Veličković et al. “Graph attention networks”. In: *Proceedings of the 7th International Conference on Learning Representations*. 2018.
- [143] Xiao Wang et al. “Heterogeneous Graph Attention Network”. In: *The World Wide Web Conference*. ACM. 2019, pp. 2022–2032.
- [144] Chuxu Zhang et al. “Heterogeneous Graph Neural Network”. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM. 2019, pp. 793–803.
- [145] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. “Learning with hypergraphs: Clustering, classification, and embedding”. In: *Advances in neural information processing systems*. 2007, pp. 1601–1608.
- [146] Zhirong Wu et al. “3d shapenets: A deep representation for volumetric shapes”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1912–1920.
- [147] Ding-Yun Chen et al. “On visual similarity based 3D model retrieval”. In: *Computer graphics forum*. Vol. 22. 3. Wiley Online Library. 2003, pp. 223–232.
- [148] Carlos Busso et al. “IEMOCAP: Interactive emotional dyadic motion capture database”. In: *Language resources and evaluation* 42.4 (2008), p. 335.
- [149] Jeffrey Pennington, Richard Socher, and Christopher Manning. “Glove: Global vectors for word representation”. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543.
- [150] iMotions. *Facial expression analysis*.
- [151] Gilles Degottex et al. “COVAREP—A collaborative voice analysis repository for speech technologies”. In: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2014, pp. 960–964.

- [152] Thomas N Kipf and Max Welling. “Semi-supervised classification with graph convolutional networks”. In: *arXiv preprint arXiv:1609.02907* (2016).
- [153] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).
- [154] Yang Xu et al. “LayoutLMv2: Multi-modal Pre-training for Visually-rich Document Understanding”. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, Aug. 2021, pp. 2579–2591. DOI: 10.18653/v1/2021.acl-long.201. URL: <https://aclanthology.org/2021.acl-long.201>.
- [155] Lukasz Garncarek et al. “LAMBERT: layout-aware language modeling for information extraction”. In: *International Conference on Document Analysis and Recognition*. Springer. 2021, pp. 532–547. URL: [https://link.springer.com/chapter/10.1007/978-3-030-86549-8\\_34](https://link.springer.com/chapter/10.1007/978-3-030-86549-8_34).
- [156] Chen-Yu Lee et al. “FormNet: Structural Encoding beyond Sequential Modeling in Form Document Information Extraction”. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 3735–3754. DOI: 10.18653/v1/2022.acl-long.260. URL: <https://aclanthology.org/2022.acl-long.260>.
- [157] Zilong Wang and Jingbo Shang. “Towards Few-shot Entity Recognition in Document Images: A Label-aware Sequence-to-Sequence Framework”. In: *Findings of the Association for Computational Linguistics: ACL 2022*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 4174–4186. DOI: 10.18653/v1/2022.findings-acl.329. URL: <https://aclanthology.org/2022.findings-acl.329>.
- [158] Zifeng Wang et al. “QueryForm: A Simple Zero-shot Form Entity Query Framework”. In: *Findings of the Association for Computational Linguistics: ACL 2023*. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 4146–4159. DOI: 10.18653/v1/2023.findings-acl.255. URL: <https://aclanthology.org/2023.findings-acl.255>.
- [159] Yinbo Chen et al. “Meta-baseline: Exploring simple meta-learning for few-shot learning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 9062–9071. URL: <https://openaccess>.

thecvf.com/content/ICCV2021/html/Chen\_Meta-Baseline\_Exploring\_Simple\_Meta-Learning\_for\_Few-Shot\_Learning\_ICCV\_2021\_paper.html.

- [160] Yiheng Xu et al. “LayoutLM: Pre-training of Text and Layout for Document Image Understanding”. In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020, pp. 1192–1200. URL: <https://api.semanticscholar.org/CorpusID:209515395>.
- [161] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of NAACL-HLT*. 2019, pp. 4171–4186. URL: <https://arxiv.org/abs/1810.04805>.
- [162] Yupan Huang et al. “Layoutlmv3: Pre-training for document ai with unified text and image masking”. In: *Proceedings of the 30th ACM International Conference on Multimedia*. 2022, pp. 4083–4091. URL: <https://dl.acm.org/doi/abs/10.1145/3503161.3548112>.
- [163] Dongsheng Wang et al. “DocGraphLM: Documental Graph Language Model for Information Extraction”. In: *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (2023)*. URL: <https://api.semanticscholar.org/CorpusID:259949760>.
- [164] Peng Zhang et al. “TRIE: end-to-end text reading and information extraction for document understanding”. In: *Proceedings of the 28th ACM International Conference on Multimedia*. 2020, pp. 1413–1422. URL: <https://api.semanticscholar.org/CorpusID:218900797>.
- [165] Dengliang Shi et al. “LayoutGCN: A Lightweight Architecture for Visually Rich Document Understanding”. In: *Document Analysis and Recognition - ICDAR 2023*. Ed. by Gernot A. Fink et al. Cham: Springer Nature Switzerland, 2023, pp. 149–165. ISBN: 978-3-031-41682-8. URL: <https://api.semanticscholar.org/CorpusID:261101837>.
- [166] Teakgyu Hong et al. “Bros: A pre-trained language model focusing on text and layout for better key information extraction from documents”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 10. 2022, pp. 10767–10775. URL: <https://doi.org/10.1609/aaai.v36i10.21322>.
- [167] Zilong Wang et al. “LayoutReader: Pre-training of Text and Layout for Reading Order Detection”. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 2021, pp. 4735–4744. URL: <https://api.semanticscholar.org/CorpusID:237304067>.

- [168] Yisheng Song et al. “A Comprehensive Survey of Few-Shot Learning: Evolution, Applications, Challenges, and Opportunities”. In: *ACM Comput. Surv.* 55.13s (2023). ISSN: 0360-0300. DOI: 10.1145/3582688. URL: <https://doi.org/10.1145/3582688>.
- [169] Mona Köhler, Markus Eisenbach, and Horst-Michael Gross. “Few-Shot Object Detection: A Comprehensive Survey”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2023), pp. 1–21. DOI: 10.1109/TNNLS.2023.3265051.
- [170] Simone Antonelli et al. “Few-shot object detection: A survey”. In: *ACM Computing Surveys (CSUR)* 54.11s (2022), pp. 1–37. DOI: 10.1145/3519022.
- [171] Jing Li et al. “Few-Shot Named Entity Recognition via Meta-Learning”. In: *IEEE Transactions on Knowledge and Data Engineering* 34.9 (2022), pp. 4245–4256. DOI: 10.1109/TKDE.2020.3038670.
- [172] Jiaxin Huang et al. “Few-Shot Named Entity Recognition: An Empirical Baseline Study”. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 10408–10423. DOI: 10.18653/v1/2021.emnlp-main.813. URL: <https://aclanthology.org/2021.emnlp-main.813>.
- [173] Jiayi Chen and Aidong Zhang. “HetMAML: Task-Heterogeneous Model-Agnostic Meta-Learning for Few-Shot Learning Across Modalities”. In: *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. CIKM ’21. Virtual Event, Queensland, Australia: Association for Computing Machinery, 2021, 191–200. ISBN: 9781450384469. DOI: 10.1145/3459637.3482262. URL: <https://doi.org/10.1145/3459637.3482262>.
- [174] Zhiqiu Lin et al. “Multimodality helps unimodality: Cross-modal few-shot learning with multimodal models”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 19325–19337. URL: [https://openaccess.thecvf.com/content/CVPR2023/html/Lin\\_Multimodality\\_Helps\\_Unimodality\\_Cross-Modal\\_Few-Shot\\_Learning\\_With\\_Multimodal\\_Models\\_CVPR\\_2023\\_paper.html](https://openaccess.thecvf.com/content/CVPR2023/html/Lin_Multimodality_Helps_Unimodality_Cross-Modal_Few-Shot_Learning_With_Multimodal_Models_CVPR_2023_paper.html).
- [175] Duong Le et al. “POODLE: Improving Few-shot Learning via Penalizing Out-of-Distribution Samples”. In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021,

- pp. 23942–23955. URL: [https://proceedings.neurips.cc/paper\\_files/paper/2021/file/c91591a8d461c2869b9f535ded3e213e-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2021/file/c91591a8d461c2869b9f535ded3e213e-Paper.pdf).
- [176] Arindam Chaudhuri et al. “Optical character recognition systems”. In: *Optical Character Recognition Systems for Different Languages with Soft Computing*. Springer, 2017, pp. 9–41. URL: [https://link.springer.com/chapter/10.1007/978-3-319-50252-6\\_2](https://link.springer.com/chapter/10.1007/978-3-319-50252-6_2).
- [177] Adam W. Harley, Alex Ufkes, and Konstantinos G. Derpanis. “Evaluation of deep convolutional nets for document image classification and retrieval”. In: *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*. 2015, pp. 991–995. DOI: 10.1109/ICDAR.2015.7333910.
- [178] Prannay Khosla et al. “Supervised Contrastive Learning”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 18661–18673. URL: [https://proceedings.neurips.cc/paper\\_files/paper/2020/file/d89a66c7c80a29b1bdbab0f2a1a94af8-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2020/file/d89a66c7c80a29b1bdbab0f2a1a94af8-Paper.pdf).
- [179] Jie Ren et al. “A simple fix to mahalanobis distance for improving nearood detection”. In: *arXiv preprint arXiv:2106.09022* (2021). URL: <https://api.semanticscholar.org/CorpusID:235458597>.
- [180] Aniruddh Raghu et al. “Rapid learning or feature reuse? towards understanding the effectiveness of maml”. In: *arXiv preprint arXiv:1909.09157* (2019).
- [181] Yuanyishu Tian et al. “FedBERT: When federated learning meets pre-training”. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 13.4 (2022), pp. 1–26. URL: <https://api.semanticscholar.org/CorpusID:246531797>.
- [182] Seunghyun Park et al. “CORD: a consolidated receipt dataset for post-OCR parsing”. In: *Workshop on Document Intelligence at NeurIPS 2019*. 2019. URL: <https://openreview.net/forum?id=SJ13z659UH>.
- [183] Martin Zinkevich et al. “Parallelized Stochastic Gradient Descent”. In: *Advances in Neural Information Processing Systems*. Ed. by J. Lafferty et al. Vol. 23. Curran Associates, Inc., 2010. URL: [https://proceedings.neurips.cc/paper\\_files/paper/2010/file/abea47ba24142ed16b7d8fbf2c740e0d-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2010/file/abea47ba24142ed16b7d8fbf2c740e0d-Paper.pdf).



- [184] Krishna Pillutla, Sham M. Kakade, and Zaid Harchaoui. “Robust Aggregation for Federated Learning”. In: *IEEE Transactions on Signal Processing* 70 (2022), pp. 1142–1154. DOI: 10.1109/TSP.2022.3153135.
- [185] Zhisheng Xiao, Qing Yan, and Yali Amit. “Likelihood regret: An out-of-distribution detection score for variational auto-encoder”. In: *Advances in neural information processing systems* 33 (2020), pp. 20685–20696. URL: <https://api.semanticscholar.org/CorpusID:212628513>.
- [186] Laurens van der Maaten and Geoffrey E. Hinton. “Visualizing Data using t-SNE”. In: *Journal of Machine Learning Research* 9 (2008), pp. 2579–2605. URL: <https://api.semanticscholar.org/CorpusID:5855042>.
- [187] Bang Liu et al. “Matching long text documents via graph convolutional networks”. In: *arXiv preprint arXiv:1802.07459* (2018).
- [188] Yanbin Liu et al. “Learning to Propagate Labels: Transductive Propagation Network for Few-Shot Learning”. In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL: <https://openreview.net/forum?id=SyVuRiC5K7>.
- [189] Limeng Qiao et al. “Transductive episodic-wise adaptive metric for few-shot learning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 3603–3612.
- [190] Imtiaz Ziko et al. “Laplacian regularized few-shot learning”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 11660–11670.
- [191] Frederik Pahde et al. “Multimodal Prototypical Networks for Few-shot Learning”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2021, pp. 2644–2653.
- [192] Vladimir N Vapnik. “An overview of statistical learning theory”. In: *IEEE transactions on neural networks* 10.5 (1999), pp. 988–999.
- [193] Fei Wang and Changshui Zhang. “Label propagation through linear neighborhoods”. In: *IEEE Transactions on Knowledge and Data Engineering* 20.1 (2007), pp. 55–67.
- [194] Felix Wu et al. “Simplifying graph convolutional networks”. In: *International conference on machine learning*. PMLR. 2019, pp. 6861–6871.

- [195] Jiayi Chen and Aidong Zhang. “HGMF: Heterogeneous Graph-based Fusion for Multimodal Data with Incompleteness”. In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020, pp. 1295–1305.
- [196] Guneet S Dhillon et al. “A baseline for few-shot image classification”. In: *arXiv preprint arXiv:1909.02729* (2019).
- [197] Malik Boudiaf et al. “Information Maximization for Few-Shot Learning”. In: *Advances in Neural Information Processing Systems* 33 (2020).
- [198] Victor Garcia and Joan Bruna. “Few-shot learning with graph neural networks”. In: *arXiv preprint arXiv:1711.04043* (2017).
- [199] Jongmin Kim et al. “Edge-labeling graph neural network for few-shot learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 11–20.
- [200] Mengye Ren et al. “Meta-learning for semi-supervised few-shot classification”. In: *arXiv preprint arXiv:1803.00676* (2018).
- [201] Ruibing Hou et al. “Cross attention network for few-shot classification”. In: *arXiv preprint arXiv:1910.07677* (2019).
- [202] Zonghan Wu et al. “A comprehensive survey on graph neural networks”. In: *IEEE transactions on neural networks and learning systems* (2020).
- [203] Luca Franceschi et al. “Learning discrete structures for graph neural networks”. In: *International conference on machine learning*. PMLR. 2019, pp. 1972–1982.
- [204] Daniel Zügner and Stephan Günnemann. “Adversarial attacks on graph neural networks via meta learning”. In: *arXiv preprint arXiv:1902.08412* (2019).
- [205] Wenhan Xiong et al. “One-shot relational learning for knowledge graphs”. In: *arXiv preprint arXiv:1808.09040* (2018).
- [206] Mingyang Chen et al. “Meta relational learning for few-shot link prediction in knowledge graphs”. In: *arXiv preprint arXiv:1909.01515* (2019).
- [207] Fan Zhou et al. “Meta-GNN: On Few-shot Node Classification in Graph Meta-learning”. In: *Proceedings of the 28th ACM International Conference*

- on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*. Ed. by Wenwu Zhu et al. ACM, 2019, pp. 2357–2360. DOI: 10.1145/3357384.3358106. URL: <https://doi.org/10.1145/3357384.3358106>.
- [208] Huaxiu Yao et al. “Graph few-shot learning via knowledge transfer”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 04. 2020, pp. 6656–6663.
- [209] C. Wah et al. *The Caltech-UCSD Birds-200-2011 Dataset*. Tech. rep. CNS-TR-2011-001. California Institute of Technology, 2011.
- [210] Lu Liu et al. “Learning to Propagate for Graph Meta-Learning”. In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach et al. 2019, pp. 1037–1048. URL: <https://proceedings.neurips.cc/paper/2019/hash/00ac8ed3b4327bdd4ebbecb2ba10a00-Abstract.html>.
- [211] Fenglin Liu et al. “Federated learning for vision-and-language grounding problems”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 07. 2020, pp. 11572–11579.
- [212] Abhishek Singh et al. “Detailed comparison of communication efficiency of split learning and federated learning”. In: *arXiv preprint arXiv:1909.09145* (2019).
- [213] Chandra Thapa et al. “Splitfed: When federated learning meets split learning”. In: *arXiv preprint arXiv:2004.12088* (2020).
- [214] Li Zhang et al. “Dynamic graph message passing networks”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 3726–3735.
- [215] Giannis Nikolentzos, Antoine Tixier, and Michalis Vazirgiannis. “Message passing attention networks for document understanding”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 05. 2020, pp. 8544–8551.
- [216] Wenchao Xia et al. “Multi-armed bandit-based client scheduling for federated learning”. In: *IEEE Transactions on Wireless Communications* 19.11 (2020), pp. 7108–7123.

- [217] Yae Jee Cho et al. “Bandit-based communication-efficient client selection strategies for federated learning”. In: *2020 54th Asilomar Conference on Signals, Systems, and Computers*. IEEE. 2020, pp. 1066–1069.
- [218] Volodymyr Kuleshov and Doina Precup. “Algorithms for multi-armed bandit problems”. In: *arXiv preprint arXiv:1402.6028* (2014).
- [219] Marco F Duarte and Yu Hen Hu. “Vehicle classification in distributed sensor networks”. In: *Journal of Parallel and Distributed Computing* 64.7 (2004), pp. 826–838.
- [220] Emile Mathieu et al. “Disentangling disentanglement in variational autoencoders”. In: *International conference on machine learning*. PMLR. 2019, pp. 4402–4412.
- [221] Luan Tran, Xi Yin, and Xiaoming Liu. “Disentangled representation learning gan for pose-invariant face recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1415–1424.
- [222] Devamanyu Hazarika, Roger Zimmermann, and Soujanya Poria. “Misa: Modality-invariant and-specific representations for multimodal sentiment analysis”. In: *Proceedings of the 28th ACM international conference on multimedia*. 2020, pp. 1122–1131.
- [223] Abel Gonzalez-Garcia, Joost Van De Weijer, and Yoshua Bengio. “Image-to-image translation for cross-domain disentanglement”. In: *Advances in neural information processing systems* 31 (2018).
- [224] Yong Li, Yuanzhi Wang, and Zhen Cui. “Decoupled Multimodal Distilling for Emotion Recognition”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 6631–6640.
- [225] Chen Yang et al. “FedPD: Federated Open Set Recognition with Parameter Disentanglement”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 4882–4891.
- [226] Cosmin I Bercea et al. “Federated disentangled representation learning for unsupervised brain anomaly detection”. In: *Nature Machine Intelligence* 4.8 (2022), pp. 685–695.
- [227] Tiandi Ye et al. “Robust Clustered Federated Learning”. In: *International Conference on Database Systems for Advanced Applications*. Springer. 2023, pp. 677–692.

- [228] Katharine Sanderson. “GPT-4 is here: what scientists think”. In: *Nature* 615.7954 (2023), p. 773.
- [229] Jiayang Wu et al. “Multimodal large language models: A survey”. In: *arXiv preprint arXiv:2311.13165* (2023).
- [230] Lijun Yu et al. “DocumentNet: Bridging the Data Gap in Document Pre-training”. In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track*. Ed. by Mingxuan Wang and Imed Zitouni. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 707–722. DOI: 10.18653/v1/2023.emnlp-industry.66. URL: <https://aclanthology.org/2023.emnlp-industry.66>.
- [231] Jiayi Chen and Aidong Zhang. “Topological Transduction for Hybrid Few-Shot Learning”. In: *Proceedings of the ACM Web Conference 2022*. WWW ’22. Virtual Event, Lyon, France: Association for Computing Machinery, 2022, 3134–3142. ISBN: 9781450390965. DOI: 10.1145/3485447.3512033. URL: <https://doi.org/10.1145/3485447.3512033>.
- [232] Jiayi Chen and Aidong Zhang. “HetMAML: Task-heterogeneous model-agnostic meta-learning for few-shot learning across modalities”. In: *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2021, pp. 191–200.
- [233] Jiayi Chen et al. “On Task-personalized Multimodal Few-shot Learning for Visually-rich Document Entity Retrieval”. In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 9006–9025. DOI: 10.18653/v1/2023.findings-emnlp.604. URL: <https://aclanthology.org/2023.findings-emnlp.604>.
- [234] Yikai Wang et al. “Multimodal token fusion for vision transformers”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 12186–12195.
- [235] Jiayi Chen and Aidong Zhang. “On Hierarchical Disentanglement of Interactive Behaviors for Multimodal Spatiotemporal Data with Incompleteness”. In: *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2023, pp. 213–225.
- [236] Paul Hongsuck Seo et al. “End-to-end generative pretraining for multimodal video captioning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 17959–17968.

- [237] Yaowei Li et al. “Efficient Multimodal Fusion via Interactive Prompting”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 2604–2613.
- [238] Sen Wu, Hongyang R Zhang, and Christopher Ré. “Understanding and improving information transfer in multi-task learning”. In: *arXiv preprint arXiv:2005.00944* (2020).
- [239] Yang Liu et al. “A survey of visual transformers”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2023).
- [240] Peng Xu, Xiatian Zhu, and David A Clifton. “Multimodal learning with transformers: A survey”. In: *arXiv preprint arXiv:2206.06488* (2022).
- [241] Mihee Lee and Vladimir Pavlovic. “Private-shared disentangled multimodal vae for learning of latent representations”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 1692–1700.
- [242] Paul Pu Liang et al. “Multibench: Multiscale benchmarks for multimodal representation learning”. In: *arXiv preprint arXiv:2107.07502* (2021).
- [243] Amir Shahroudy et al. “NTU RGB+D: A large scale dataset for 3D human activity analysis”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 1010–1019.
- [244] Sabtain Ahmad and Atakan Aral. “FedCD: Personalized federated learning via collaborative distillation”. In: *2022 IEEE/ACM 15th International Conference on Utility and Cloud Computing (UCC)*. IEEE. 2022, pp. 189–194.
- [245] Pradeep K Atrey et al. “Multimodal fusion for multimedia analysis: a survey”. In: *Multimedia systems* 16 (2010), pp. 345–379.
- [246] Yuang Jiang et al. “Model pruning enables efficient federated learning on edge devices”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2022).
- [247] Saeed Vahidian, Mahdi Morafah, and Bill Lin. “Personalized federated learning by structured and unstructured pruning under data heterogeneity”. In: *2021 IEEE 41st international conference on distributed computing systems workshops (ICDCSW)*. IEEE. 2021, pp. 27–34.

- [248] Zhida Jiang et al. “Computation and Communication Efficient Federated Learning With Adaptive Model Pruning”. In: *IEEE Transactions on Mobile Computing* (2023).
- [249] Hanxiao Liu, Karen Simonyan, and Yiming Yang. “DARTS: Differentiable Architecture Search”. In: *International Conference on Learning Representations*. 2018.
- [250] Yushiro Funoki and Satoshi Ono. “DMNAS: Differentiable Multi-modal Neural Architecture Search”. In: *International Workshop on Advanced Imaging Technology (IWAIT) 2021*. Vol. 11766. SPIE. 2021, pp. 465–470.
- [251] Chris Zhang, Mengye Ren, and Raquel Urtasun. “Graph hypernetworks for neural architecture search”. In: *arXiv preprint arXiv:1810.05749* (2018).
- [252] Yijian Qin et al. “Graph Differentiable Architecture Search with Structure Learning”. In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 16860–16872. URL: [https://proceedings.neurips.cc/paper\\_files/paper/2021/file/8c9f32e03aeb2e3000825c8c875c4edd-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2021/file/8c9f32e03aeb2e3000825c8c875c4edd-Paper.pdf).
- [253] Pushmeet Kohli Nathan Silberman Derek Hoiem and Rob Fergus. “Indoor Segmentation and Support Inference from RGBD Images”. In: *ECCV*. 2012.
- [254] Tao Shi and Shao-Lun Huang. “MultiEMO: An Attention-Based Correlation-Aware Multimodal Fusion Framework for Emotion Recognition in Conversations”. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2023, pp. 14752–14766.
- [255] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. “Dynamic routing between capsules”. In: *Advances in neural information processing systems* 30 (2017).
- [256] R Gnana Praveen et al. “A joint cross-attention model for audio-visual fusion in dimensional emotion recognition”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 2486–2495.
- [257] Enmao Diao, Jie Ding, and Vahid Tarokh. “HeteroFL: Computation and Communication Efficient Federated Learning for Heterogeneous Clients”. In: *International Conference on Learning Representations*. 2020.
- [258] Alysa Ziyang Tan et al. “Towards personalized federated learning”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2022).

- [259] Gaurav Kumar and Durga Toshniwal. “Neuron Specific Pruning for Communication Efficient Federated Learning”. In: *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022, pp. 4148–4152.