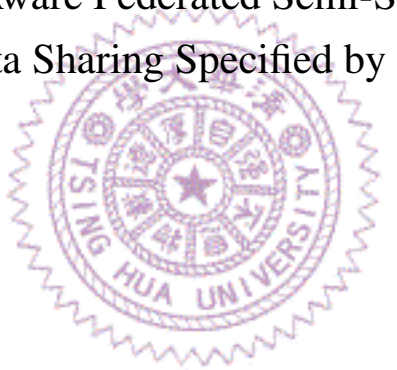
國立清華大學電機資訊學院資訊工程研究所
碩士論文
Department of Computer Science
College of Electrical Engineering and Computer Science
National Tsing Hua University
Master Thesis

基於個體客戶選擇性資料分享的模態感知聯邦半監督學習

MAFS: Modality-Aware Federated Semi-Supervised Learning with Selective Data Sharing Specified by Individual Clients



111062679
李翊辰
Yi-Chen Li

指導教授：徐正炘 博士
Advisor: Cheng-Hsin Hsu, Ph.D.

中華民國 113 年 8 月
August, 2024

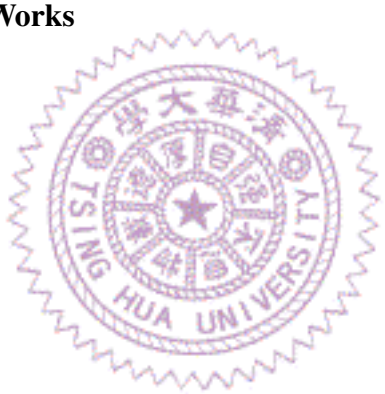
中文摘要

與單一模態資料相比，多模態感測資料能提升複雜任務的模型表現。聯邦學習（Federated Learning）進一步增強了這一點，既保護數據隱私又確保模型訓練效果。然而，現有的FL算法往往忽視了部分用戶分享特定資料模態的意願，並且因缺乏大規模公開資料集而難以獲取足夠的標記資料。我們提出了模態感知聯邦半監督學習（MAFS）架構，允許各個客戶端選擇他們認為不敏感且願意與FL伺服器分享的資料模態。MAFS從這些未標記的不敏感資料中提取有用訊息，以減輕標記資料匱乏的問題。我們在情感識別（Emotion Recognition）和人類活動識別（Human Activity Recognition）兩個任務上評估了MAFS，並將其與多種最先進的FL算法進行比較。實驗結果顯示，在情感識別任務中，當標記資料比例僅為30%時，MAFS將準確率提高了至少6.94%，F1-Score提高了至少9.49%，僅比完全標記資料的結果低0.63%和0.46%。在人類活動識別任務中，MAFS也有良好表現，例如，準確率提高了至少3.37%，且F1-Score沒有顯著下降。

Abstract

Compared to unimodal data, multimodal sensor data improves model performance for complex tasks. Federated Learning (FL) further enhances this by preserving data privacy while ensuring well-trained models. However, existing FL algorithms often overlook some users' willingness to share certain data modalities and struggle to acquire sufficient labeled data due to a scarcity of large-scale public datasets. We propose Modality-Aware Federated Semi-Supervised Learning (MAFS) paradigm, allowing individual clients to select which data modalities they consider insensitive and are willing to share with the FL server. MAFS then extracts useful information from those unlabeled insensitive data to mitigate labeled data scarcity. We evaluate MAFS on two tasks: Emotion Recognition (ER) and Human Activity Recognition (HAR), and compare it with several state-of-the-art FL algorithms. The experimental results show that, in the ER task, when the labeled data rate is only 30%, MAFS improves the accuracy by at least 6.94% and F1-score by at least 9.49%, which are merely 0.63% and 0.46% away from those from a fully labeled dataset. MAFS also performs well in the HAR task, e.g., it improves the accuracy by at least 3.37% with no significant drop in F1-score.

Contents



List of Figures

List of Tables

Chapter 1

Introduction

In today's digitally driven world, multimodal machine learning [4, 62] has become an integral part of our daily lives, significantly enhancing the way we interact with technologies and each other. Sensors collect a vast amount of data in various modalities, including visual, auditory, and tactile inputs [5, 48, 3]. While such technologies provide unprecedented convenience and insights, they raise critical data privacy and security concerns.

Federated Learning (FL) [45] emerges as a powerful solution in this context, enabling the collaborative training of machine learning models on decentralized devices. FL ensures that sensitive data collected by multimodal sensors never leaves the user's device, thereby protecting individual privacy while still benefiting from collective insights. Realizing FL, however, incurs multiple challenges, such as aggregation optimization [45, 47, 71, 60, 35] and heterogeneity [11, 17, 14, 29, 61, 69, 21, 52]. While several algorithms and research efforts have been put forward to address these challenges, a key problem in leveraging FL effectively is the scarcity of labeled data in the realm of multimodal sensor inputs. Most advanced machine learning models require large amounts of annotated data to learn accurately and quickly, but obtaining such labeled datasets is often expensive and time-consuming [2, 51].

Current solutions to this challenge include semi-supervised learning (SSL) and unsupervised learning (UL) techniques [72, 23, 9, 55] that attempt to learn from the labeled and unlabeled data owned by each client. The problem is that if the degree of non-i.i.d. of the client data is greater, the bias between the client models generated will also be greater. Although we can use FL algorithms (such as FedAvg) to aggregate these client models into a global model containing the knowledge of different clients and then use this global model to perform SSL or UL on unlabeled data to reduce model bias, the model performance and robustness of this method are worse than directly learning the global model from raw data, and the convergence speed is slower. This is because the process of aggregating client models can only obtain indirect information such as gradients or parameters.

Therefore, we want to improve the above problems by directly obtaining the raw data, which will involve privacy issues.

When users share raw data with the server, there is a risk of malicious attacks that could lead to privacy leakage issues. To address this, researchers have proposed algorithms like differential privacy, k-anonymity, l-diversity, and t-closeness [58, 41, 46] to encrypt the raw data. While these algorithms can quantify the degree of protection against malicious access to the raw data, they cannot quantify the extent of privacy leakage if the data has been maliciously obtained. In fact, quantifying the latter is difficult, as individuals have subjective judgments on data leakage. For example, when registering on a website, the gender field is often selectively filled since some people may not mind others knowing their gender, while others may not want anyone to know. In other words, different people have varying privacy concerns even for the same data modality. Using a user-centric metric to quantify this type of privacy is challenging. Therefore, our research has shifted towards exploring users' willingness to share different data modalities rather than establishing a privacy quantifying metric. Leaving the decision to share raw data with the user does not mean the privacy issue has been resolved. Rather, it allows users to share data that is relatively insensitive to them, so they feel comfortable even if it is leaked.

(a) (b) (c)

Figure 1.1: Sample data from: (a) an RGB camera, (b) a microphone, and (c) a mmWave radar.

The only relevant work [12] applied raw data directly in the FL scenario. This work proposes the concept of data sensitivity differentiation, which divides different modalities of data into sensitive and insensitive. Fig. 1.1 shows an example to illustrate the data sensitivity differentiation among different modality data, which uses an RGB camera to capture the driver's facial expressions, a microphone to record the tone of the driver's speech, and a mmWave radar to capture the driver's hand gestures. Notice that we can easily recognize the driver's identity through images from RGB cameras. Conversely, it's much harder to identify the driver using the sparse point clouds from mmWave radars.

The difference in the amount of information contained in various modality data can further influence the user's willingness to share data. Generally, users do not want to share their RGB images, but it may not matter for celebrities. Different users have subjective considerations regarding their willingness to share different modality data. However, this work is limited to the scenario where all training data needs to be labeled and cannot be applied to the problem of scarce labeled data.

1.1 Contributions

We proposed the Modality-Aware Federated Semi-Supervised Learning (MAFS) paradigm as shown in Fig. 1.2 for a broader scenario, which makes the following contributions:

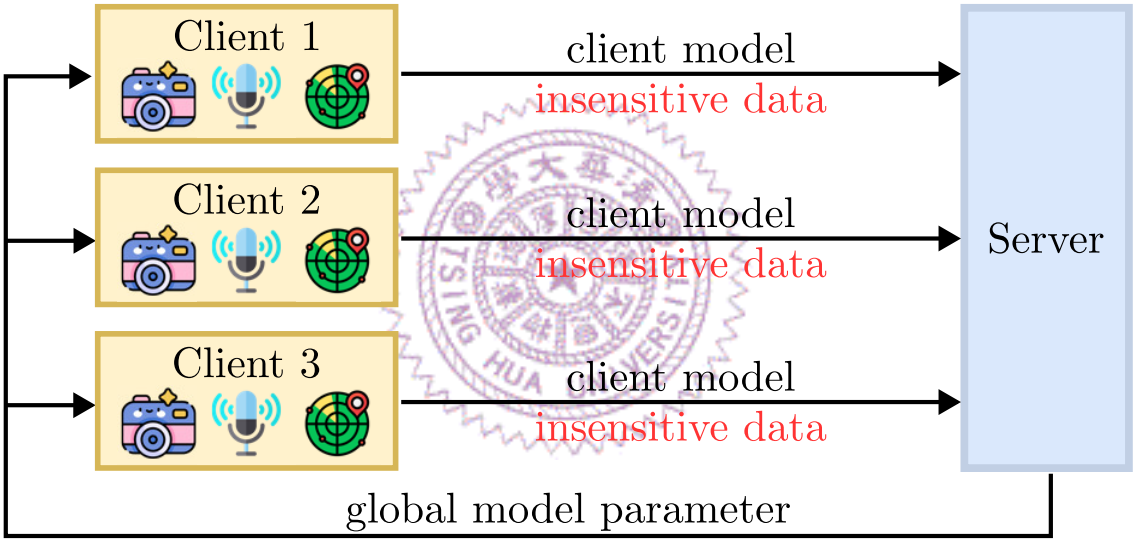


Figure 1.2: Illustration of multimodal sensor data sharing in MAFS.

- MAFS paradigm collects unlabeled insensitive data from clients and uses SSL pseudo-labeling to generate usable data for server training. This novel approach in FSSL reduces client model bias and increases convergence speed.

- MAFS paradigm comes with a modularized design on FL clients and servers, allowing developers to readily augment FL neural network structures into MAFS-ied version.

- MAFS paradigm has been applied to two sample classification problems on Emotion Recognition (ER) and Human Activity Recognition (HAR) to demonstrate its practicality and efficiency.

1.2 Limitations

While the MAFS paradigm shows significant promise, it also has certain limitations. One major limitation is the dependency on the pseudo-labeling technique used for unlabeled data, which may introduce noise and inaccuracies, potentially impacting the overall model performance. Additionally, the computational overhead required for training complex multimodal models on client devices could be a bottleneck, especially for clients with limited resources. Furthermore, while MAFS improves privacy by only sharing insensitive data, there remains a risk of indirect privacy leakage through model updates and pseudo-label data. Another limitation is the assumption that the data collected from clients includes complete and correct labels and modalities. Addressing these limitations requires ongoing research and development to enhance the robustness, efficiency, and privacy-preserving capabilities of federated learning systems.

1.3 Organizations

This thesis is organized into several sections to provide a comprehensive understanding of the MAFS paradigm and its implications. Ch. 1 outlines the motivations behind this research, the challenges faced in federated learning with multimodal data, and the contributions of the MAFS paradigm. Ch. 2 provides an overview of federated learning, data sharing, distillation, federated transfer learning, and multimodal representation learning. It also includes a convergence analysis of the proposed methods. Ch. 3 introduces our previous work called Heterogeneous Privacy Federated Learning (HPFL), which proposed the idea of data sensitivity differentiation. We apply this concept to our MAFS paradigm and promote the model performance in federated learning. Ch. 4, discusses the current state of research in multimodal federated learning and federated semi-supervised learning, highlighting the gaps that this thesis aims to address. Ch. 5 defines the specific problems that this research tackles, such as the challenges of unlabeled data usage and missing modality issues. Ch. 6 details the MAFS framework, including the client trainer, aggregator, server trainer, and merger. It explains the methodology used to address the identified challenges. Ch. 7 describes the datasets and neural network architectures used in the experiments, specifically for emotion recognition and human activity recognition. Ch. 8 presents the implementation details, hyperparameters, and the experimental results that demonstrate the effectiveness of the MAFS paradigm. Ch. 9 summarizes the findings of the research, the contributions of the MAFS paradigm, and potential future directions for further improving federated learning systems.

Chapter 2

Background

In this chapter, we introduce three important keywords: Federated Learning (FL), Semi-Supervised Learning (SSL), and Data Sharing in FL. Through FL, we can reduce the degree of privacy leakage during the process of training machine learning models. We can also utilize SSL to help us extract more useful information from large amounts of unlabeled data, thereby enhancing the performance of the machine learning model. Finally, through the mechanism of selective data sharing, we can effectively obtain clients' unlabeled data to assist in the training of the semi-supervised learning model while minimizing privacy leakage.

2.1 Federated Learning

Machine Learning (ML) aids in solving various tasks in daily life, such as Human Activity Recognition, Object Detection, and Emotion Recognition. Conventional ML involves collecting data and training models on a powerful machine, a method known as Centralized Learning (CL). However, CL faces several challenges:

Privacy Preservation. Training a CL model requires collecting data from all users, including photos and videos, potentially leading to privacy leakage issues.

Computation Overhead. Large volumes of collected user data can cause computational overload on the machine. While user data can grow infinitely, machine computation power has inherent bottlenecks.

Model Personalization. A CL model trained on data from all users may achieve high generalization but often fails to perform well on subtle differences between users, such as variations in country or habits. Conversely, training individual CL models for each user can result in poor performance when collected data significantly differs from training data, indicating insufficient robustness.

Federated Learning (FL) effectively addresses these issues. Fig. 2.1 illustrates the FL

training process:

Client trainer. Each user employs a client trainer to train their own client model locally and shares the parameters of this model with the server.

Aggregator. The aggregator implements various FL algorithms. The server receives client models from different clients and uses the aggregator to combine these into a single global model. This global model is then sent back to clients, completing one epoch.

At the start of each new epoch, clients fine-tune the received global model and generate a new client model. This process repeats continuously. During the FL model training process, the server does not collect any user data, effectively resolving the user privacy leakage problem. The server also doesn't perform any model training; all training occurs on the client side, significantly reducing the server's computation overhead. Furthermore, the global model incorporates knowledge from various clients, enhancing the generalization ability of client models. When combined with fine-tuning user data, this approach produces improved personalized models.

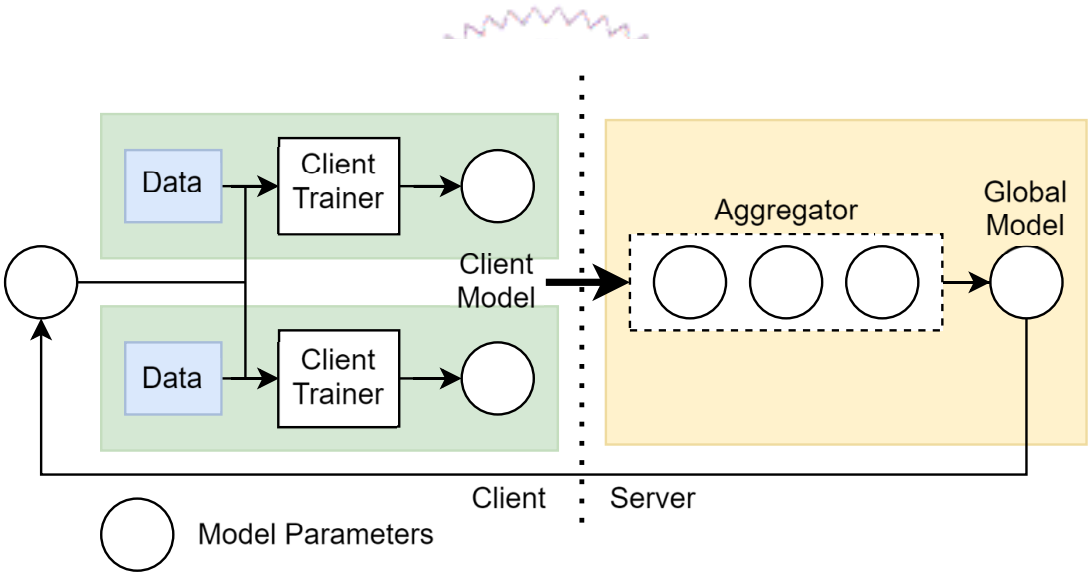


Figure 2.1: Illustration of Federated Learning Workflow.

2.2 Semi-supervised Learning

Machine learning has transformed numerous fields, yet it frequently encounters significant challenges:

Limited Labeled Data. Many ML algorithms, especially Supervised Learning (SL) methods, demand substantial amounts of labeled data for optimal performance. However, acquiring labeled data in various domains proves challenging. For example, in medical

imaging, the annotation of images by expert radiologists is both time-consuming and costly.

High Costs of Data Labeling. Data labeling often requires considerable human effort and expertise, making it prohibitively expensive, particularly for large datasets. In natural language processing, tasks such as sentiment analysis or named entity recognition demand skilled linguists and significant time investment for text annotation.

Impracticality of Full Data Labeling. Labeling all available data is often unfeasible in real-world scenarios. This is especially true in domains with continuous data generation, such as social media content or IoT sensor readings, where the volume and velocity of data production frequently exceed manual labeling capacity.

Semi-supervised learning (SSL) emerges as a potent solution to these challenges. SSL utilizes both labeled and unlabeled data for model training. SSL algorithms initially learn patterns and decision boundaries from a small amount of labeled data, then refine and improve these models using a larger quantity of unlabeled data. Common SSL techniques include pseudo labeling (self-training), co-training, and graph-based methods.

By learning from a limited set of labeled examples and extending this knowledge to a larger unlabeled dataset, SSL can significantly enhance model performance compared to using only the limited labeled data. This approach substantially reduces the cost and effort associated with data annotation by requiring labeling for only a fraction of the data. Consequently, SSL enables the development of effective models even when labeling resources are constrained.

2.3 Data Sharing in FL

FL faces the challenge of data incompleteness for individual clients, necessitating innovative solutions for data sharing without compromising privacy. Direct large-scale data sharing is restricted to maintain privacy, leading researchers to explore alternative approaches. These efforts can be categorized into two main strategies.

The first strategy involves the FL server providing additional data to clients before training begins, either from other clients or public datasets. For example, Huang et al.[28] suggested the server distribute a small, random subset of the data (up to 1%) to each client. Although this approach does not consider the local data distribution of each client, it yields a performance boost of around 1.5%. Jeong et al.[31] improved on this by having clients upload sample data and report their local data distributions. The server uses these samples to train a generative adversarial network (GAN) to augment the data, then redistributes data matching each client's distribution, enhancing performance by 6%.

Wang et al. [59] introduced the K-Nearest-Neighbors Synthetic Minority Over-Sampling

Technique (K-SMOTE) for the peer-to-peer (P2P) FL architecture. K-SMOTE generates new data from existing data, and during P2P communication, clients exchange these synthetic data points along with the model, increasing the volume of training data.

The second strategy allows the FL server to use collected data for further training after aggregating client models. Yoshida et al.[68] proposed using the server to create an i.i.d. dataset from client-collected data for additional training. Elbir et al.[16] suggested a hybrid training approach where some clients upload all their training data to the server. The server performs centralized machine learning on this data while clients conduct FL, and the models are then combined at the server. Hong et al. [27] also proposed a hybrid system focusing on minimizing communication and computation overhead while optimizing model performance.

In contrast to these approaches, Heterogeneous Privacy Federated Learning (HPFL) [12], which is our previous work, offers a novel solution by leveraging the diverse privacy sensitivity levels of various data types. In Ch.3, we will introduce this work in more detail.

Chapter 3

Heterogeneous Privacy Federated Learning (HPFL)

In this chapter, we introduce our previous work called Heterogeneous Privacy Federated Learning (HPFL) [12], which is the first work that considers the diverse privacy sensitivity levels of different modalities in FL. We summarize what kind of problem we try to solve, how we overcome the challenges, and analyze the model convergence issue.

3.1 Design Intuition

Ideally, we can roughly categorize sensor data based on their level of privacy risk. For example, an IR image is usually considered being privacy risky than RGB images. However, determining an exact value to evaluate data sensitivity is challenging. To the best of our knowledge, there is no clear definition of sensitivity levels for media data as it largely depends on an individual's subjective opinion. Therefore, there are no set rules to determine the accessibility and sharing of data. Hence, we leave it up to users to decide whether specific data can be shared. We also encourage users to share more data to improve model performance. We use three tools to solve the challenges we meet, which are Knowledge Distillation (KD), Federated Transfer Learning (FTL), and Multimodal Representation Learning (MRL).

3.2 Distillation in FL

Knowledge distillation, originally developed for neural network compression, transfers insights from complex "teacher" models to simpler "student" models [26, 22]. Jeong et al. [30] adapted this concept to FL, introducing Federated Distillation (FD) to reduce communication costs [49, 73, 24]. In FD, clients send model-specific outputs (logits)

to the server after local training. The server averages these logits and returns them to clients for use as soft targets in subsequent training rounds. While FD typically yields lower performance than FedAvg, it significantly reduces bandwidth usage to about 1% of FedAvg's requirements. Park et al. [49] proposed Federated Learning After Distillation (FLD), which combines FL and FD elements. FLD addresses limited client upload bandwidth by having clients upload logits and a small portion of their local dataset for server-side knowledge distillation (KD), while downloading the model. This approach halves communication overhead compared to FedAvg while maintaining similar performance. Further innovations in FD include Zhu et al.'s [73] server-side knowledge generator to enhance client model training, and Guha et al.'s [24] use of public datasets for model compression. FD has also been applied to personalize FL methods [34] and support heterogeneous client model structures [40]. HPFL distinguishes itself by focusing on improving server model performance without accessing privacy-sensitive data. Unlike traditional offline distillation, HPFL uses client-computed model-specific outputs to guide the training of the distillation model on the server. This novel approach utilizes in-domain privacy-insensitive data, reducing privacy risks compared to methods relying on client-uploaded or public datasets.

3.3 Federated Transfer Learning (FTL)

Transfer learning [57] involves applying domain knowledge from one area to a different but similar target domain. This approach is particularly useful when training complex models on small datasets. By pretraining a model on a large dataset and then fine-tuning it on a smaller, domain-specific dataset, we can achieve good performance on the target task. In deep neural networks, shallow layers learn general features, while deeper layers learn specific features. Typically, the shallow layers are trained on large public datasets to capture general features and are then frozen. The deeper layers are subsequently fine-tuned on the target task dataset. In FL, transfer learning is employed for personalization [33]. While a global model may not adapt well to each client due to varying data distributions, the server model generally possesses more knowledge than individual client models. Federated Transfer Learning (FTL) [13, 66] applies transfer learning principles to FL, enhancing knowledge transfer between server and client models. Similar techniques have been extended to more secure systems [53, 43]. Both FTL and HPFL freeze certain model parameters during training, albeit for different purposes. FTL freezes high-level general features to derive personalized client models, while HPFL freezes parameters related to privacy-sensitive data unavailable on the server. These approaches are complementary due to their orthogonal goals. The key difference between FTL and HPFL lies

in their methods and targets. FTL freezes layers learning general features and fine-tunes low-level layers to adapt to local data distributions. In contrast, HPFL freezes model components related to sensitive data, effectively fine-tuning specific parts of the model. HPFL conducts this fine-tuning on the server-side to improve global performance and reduce model bias, whereas FTL aims for client-side personalization.

3.4 Multimodal Representation Learning

A modality represents a distinct form of data or information, reflecting a unique way of perceiving or interacting with the world. Common modalities include visual (e.g., images, video), auditory (e.g., speech, music), textual (e.g., natural language, documents), numerical (e.g., structured data, time series), and tactile (e.g., touch, pressure) data. These diverse modalities often provide complementary information, which, when combined, can lead to a more comprehensive understanding of complex phenomena. For instance, in human communication, we simultaneously process visual cues (facial expressions, gestures) and auditory information (speech, tone) to fully comprehend the message.

Leveraging this advantage, multimodal approaches frequently outperform unimodal methods, particularly in complex real-world scenarios. This superiority is evident in tasks such as emotion recognition, where combining facial expressions, voice tone, and linguistic content yields more accurate results than using any single modality alone. Multimodal Representation Learning (MMRL) applies this concept to Machine Learning (ML), focusing on creating unified representations from multiple modalities. The objective is to learn a joint representation that captures cross-modal relationships and correlations, enabling a more robust and comprehensive understanding of complex data. This approach is particularly valuable in fields like human-computer interaction, medical diagnosis, and autonomous systems, where integrating diverse data types can lead to more powerful and versatile models.

Fig 3.1 illustrates the MMRL workflow. In this context, representation refers to the encoded or transformed format of data that algorithms can effectively process. It captures essential features or characteristics of the data, often in a lower-dimensional space. These representations can be learned through various methods, including deep neural networks, dimensionality reduction techniques, or embedding algorithms.

Common techniques for generating representations include using Convolutional Neural Networks (CNNs) for visual data processing, which excel at capturing spatial hierarchies in images. Recurrent Neural Networks (RNNs) or Transformers are often employed for textual data handling, effectively capturing sequential dependencies in language. Following representation generation, fusion techniques combine these representations from

different modalities. This step is crucial in MMRL as it determines how information from different sources is integrated. Popular fusion methods include concatenation-based, weighted sum, attention-based, and so on. Through this process, MMRL facilitates the extraction of more valuable information from diverse modality data, enhancing the overall performance and capabilities of machine learning models in multimodal tasks.

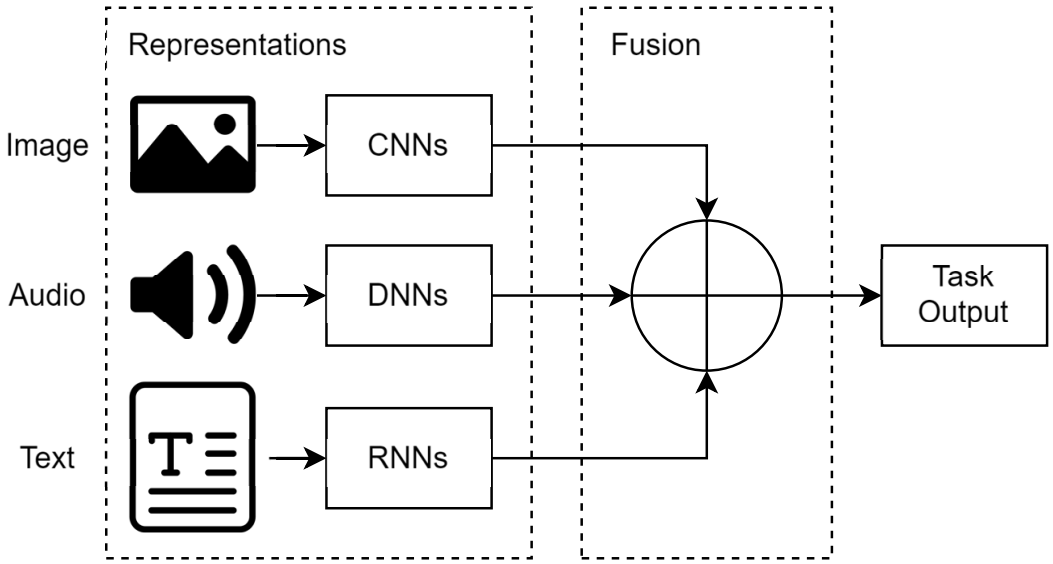


Figure 3.1: Illustration of Multimodal Representation and Fusion.

3.5 Convergence Analysis on HPFL.

We prove that the HPFL with KD architecture converges. The HPFL architecture combines the FL and KD features. On the client side, HPFL trains the model using a method similar to FedAvg. We thus prove the convergence of the client model using a method similar to FedAvg convergence analysis. On the server side, HPFL trains a distillation model. We prove the convergence of HPFL by observing the trend of the global model loss after combining the client model and the distillation model. Our proof process consists of three steps: (i) analyzing the convergence of the client model, (ii) analyzing the convergence of the server-side distillation model, and (iii) analyzing the convergence of the global model resulting from the merging of the client and distillation models. The following sections provide a detailed explanation of each of these steps.

3.5.1 Convergence Analysis on Client Models

We present a proof technique based on Li et al. [36] and make assumptions about the loss function and optimizer. The first two assumptions restrict the selection of loss function by requiring convexity and smoothness. Limiting the convexity and smoothness of loss functions is crucial for the convergence of deep learning models. Convex functions guarantee a single global minimum, ensuring optimization algorithms like gradient descent converge effectively. Smoothness relates to the behavior of function gradients; a smooth function provides consistent gradient directions, facilitating stable optimization. In the complex, high-dimensional landscapes of deep learning, ensuring convexity and smoothness can mitigate issues like local minima and saddle points, leading to faster convergence. The latter two constrain the selection of the optimizer by focusing on the gradients produced during each update. High gradient variance can cause unstable parameter updates, leading to oscillation or divergence. Bounding the squared norm ensures manageable update sizes, facilitating appropriate learning rate selection. Theoretical convergence guarantees for optimization methods like SGD often require gradient boundedness assumptions. Additionally, controlling gradient variance helps in preventing overfitting by reducing the model's susceptibility to training data noise and enhancing optimization robustness. In essence, to ensure stability, convergence, and generalization in deep learning, it is crucial to manage the magnitude and variability of gradient updates.

Assumption 1 *(Smoothness) For all* $k \in K$, L_k *are all L-smooth, i.e., for any two points* w *and* w' *within the convex set of the loss function,* $L_k(w) - L_k(w') \leq L\|w - w'\|$, *for some* $L > 0$, L_k *is the loss function of the k-th client.*

Assumption 2 *(Convexity) For all* $k \in K$, L_k *are all convex, i.e., for any two points* w *and* w' *within the convex set of the loss function,* $L_k(a \cdot w + (1-a) \cdot w') \leq a \cdot L_k(w) + (1-a) \cdot L_k(w')$, $a \in [0,\ 1]$.

We aim to discuss whether the chosen loss function conforms to the first two assumptions. Specifically, we employed the Categorical Cross Entropy (CE) as the loss function for the experimental design, including the semantic segmentation and emotion recognition tasks. Notably, CE itself is an L-smooth function, which satisfies Assumption 1. Furthermore, we incorporated the softmax function into our experiments, which renders the resulting function convex. As such, our experimental design satisfies Assumption 2.

Assumption 3 *(Bounded Variance of Stochastic Gradient)* $E\|\nabla L_k(M_{C,t}^k, S_t^k) - \nabla L_k(M_{C,t}^k)\|^2 \geq \sigma_k^2$, S_t^k *is the local data uniformly sampled at random from the k-th device at the t-th round,* σ_k *is the expected value of the squared L2 norm of the gradient,* $M_{C,t}^k$ *is the parameter of the k-th client model at the t-th round.*

Assumption 4 *(Bounded Squared Norm of Stochastic Gradient)* $E\|\nabla L_k(M_{C,t}^k, S_t^k)\|^2 \leq G^2, G = \frac{\sigma_k}{\eta}, \eta$ *is the step size.*

Next, we discuss the selection constraints for the optimizer. Our experiments used two optimizers: Stochastic Gradient Descent (SGD) and Adaptive Momentum Estimation (Adam). Although SGD cannot guarantee that stochastic gradient has bounded variance and squared norm, reducing momentum and decaying learning rate over time can limit the variance. Additionally, gradient clipping can limit the squared norm. Hence, by adjusting hyperparameters such as momentum and learning rate, SGD can meet Assumptions 3 and 4.

Similarly, Adam cannot guarantee bounded variance and squared norm for its stochastic gradient. However, adjusting the momentum and learning rate can limit the magnitude of the stochastic gradient, enabling Adam to meet Assumption 3 and Assumption 4. In Appendix 3.5.3, we provide a more detailed explanation of the impact of the size of momentum and learning rate on convergence. In our experiments, we selected the commonly used cross-entropy (CE) loss function for classification tasks and the two commonly used optimizers, SGD and Adam. We have shown that HPFL can train a client-side model that converges under most similar conditions.

3.5.2 Convergence Analysis of the Distillation Model

KD uses different sources of distilled knowledge and loss functions depending on the task, necessitating the establishment of a separate loss function for each task [22]. In HPFL, the distillation model is trained on knowledge sources from the intermediate layers of the client model, also known as the learning target. This method is called feature-based KD. To perform HPFL, we establish the following loss function: $L_S(\phi_t(f_t(x)), \phi_s(f_s(x)))$, where $f_t(x)$ and $f_s(x)$ are the feature maps of the intermediate layers for the teacher and student models, respectively. ϕ_t, ϕ_s are the transformation functions to ensure the output size from both models is the same. Lastly, $L_S(\cdot)$ is the similarity function to match teacher and student feature maps.

To prove the convergence of feature-based KD, we rely on three assumptions based on Phuong and Lampert [50]. Assumption 5 limits the number of training samples. Assumption 6 requires the loss function to be convex, like Assumption 2, whereas Assumption 7 guarantees that each update of the model parameters brings us closer to the optimal solution.

Assumption 5 $\mathbf{D_T} \leq \mathbf{d}$*, where* $\mathbf{D_T}$ *and* $\mathbf{d}$ *are the volumes of training data and whole dataset, respectively.*

Assumption 6 *$L_S(M_{D,t}) \to 0$ when $t \to \infty$, where $L_S(\cdot)$ is the loss function of the distillaion model, and $M_{D,t}$ is the distillation model parameter after t rounds.*

Assumption 7 *$L_S(M_{D,t}) \geq \frac{\mu}{2}\|M_{D,t} - M^*\|^2$ for some $\mu > 0$. where M^* is the optimal model parameter.*

Assumptions 5, 6 and 7 are important to the success of the distillation model. Assumption 5 states that with enough training samples, KD guarantees that the distillation model parameters will approach the teacher model. Even if the training samples are insufficient, the distillation model's parameters will still be a projection in the dataset's space, meaning the number of training samples does not directly affect the model's convergence. Assumption 6 requires that the loss function must be convex. Assumption 7 requires the model weight to continuously approach the optimal solution during training, which depends on the optimizer. Sec. 3.5.1 provides more detailed information on these assumptions. In summary, we satisfy the above three assumptions and complete the convergence proof of the server-side distillation model.

3.5.3 Convergence Analysis of the Global Model

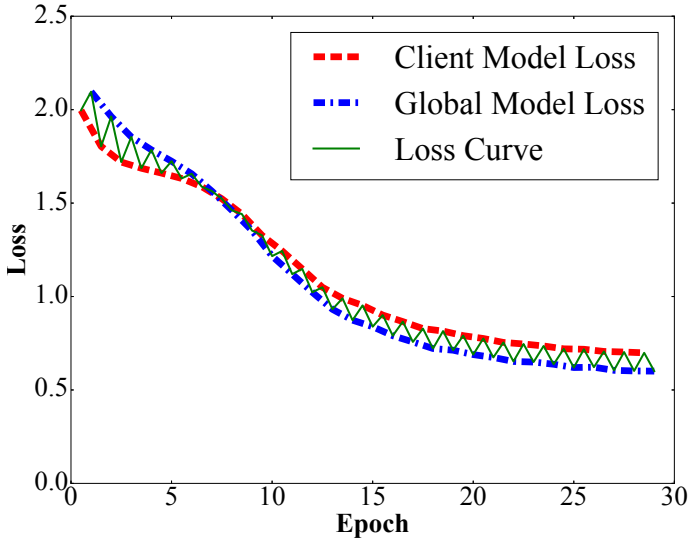
We investigate the impact of the weight allocation between the client model and the distillation model on the convergence of the global model since it is determined via the weight allocation between the client and distillation models. As training the distillation model lacks sensitive data information, it introduces uncertainty in the convergence of the global model. Hence, we compare the loss curve between the client, distillation, and global models to evaluate the convergence of the global model. Next, we individually discuss the two target problems in the experiments.

Semantic segmentation. Fig. 3.2 shows that the client model and global model exhibit decreasing loss as the number of epochs increases, and the loss oscillates within a fixed range. We also found that as the momentum decreases, the oscillation becomes more moderate. In addition, it can be observed from Fig. 3.3 that adjusting the decay rate of the learning rate helps reduce the oscillation of the loss curve without impacting the overall trend of the loss curve. However, a decay rate that is too large may decrease the model's performance. Therefore, we chose 0.95 as the optimal decay rate.

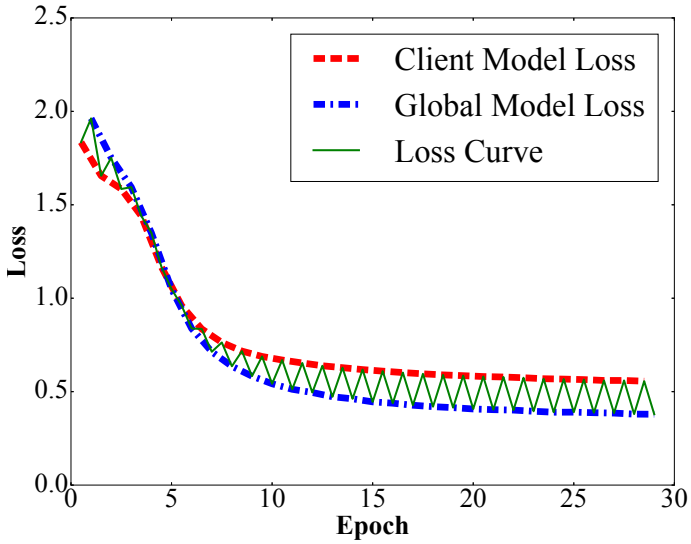
Emotion recognition. From Fig. 3.4, it can be observed that the loss curve oscillates between 0.6 and 0.7. As the momentum decreases, the oscillation not only becomes milder but also the upper bound of the oscillation can be reduced to lower than 0.65. We also examined the effect of different learning rate decay values on the loss. As shown in Fig. 3.5, the lower the learning rate decay value, the smaller the oscillation amplitude and the faster the convergence. After testing different learning rate decay values, we found

that a decay value of 0.92 provides the best balance to prevent performance degradation of the model.

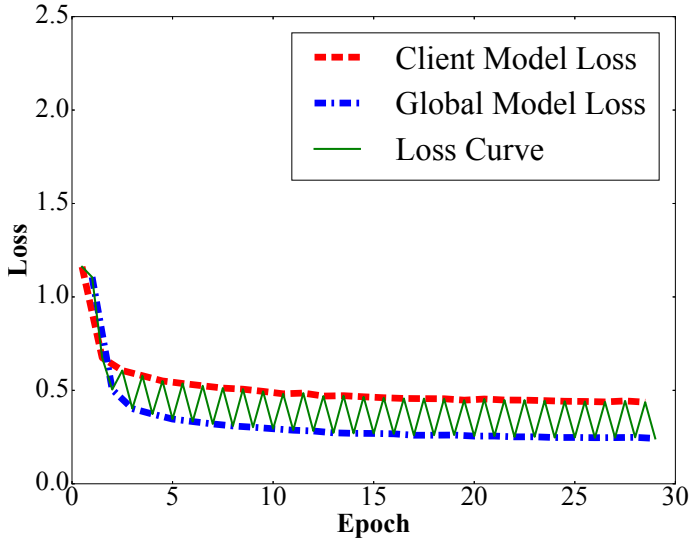
Overall, the HPFL architecture's convergence depends on the target task, loss functions, and optimizers. If the loss function and optimizer meet the aforementioned assumptions, the model trained by the HPFL framework will converge.



(a)

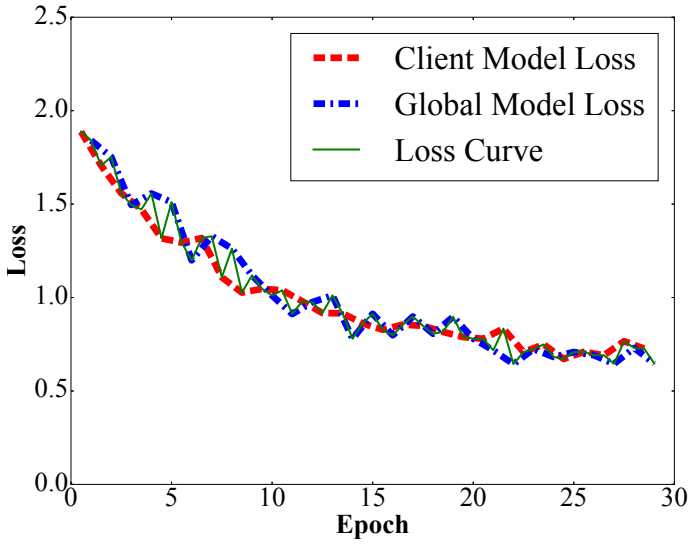


(b)

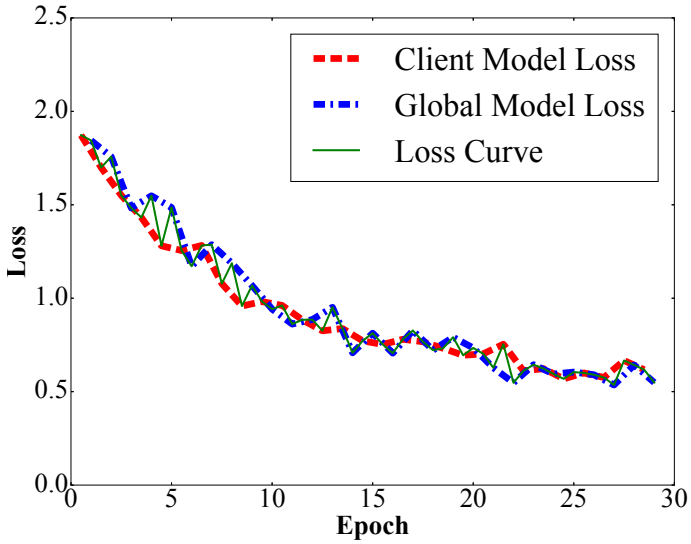


(c)

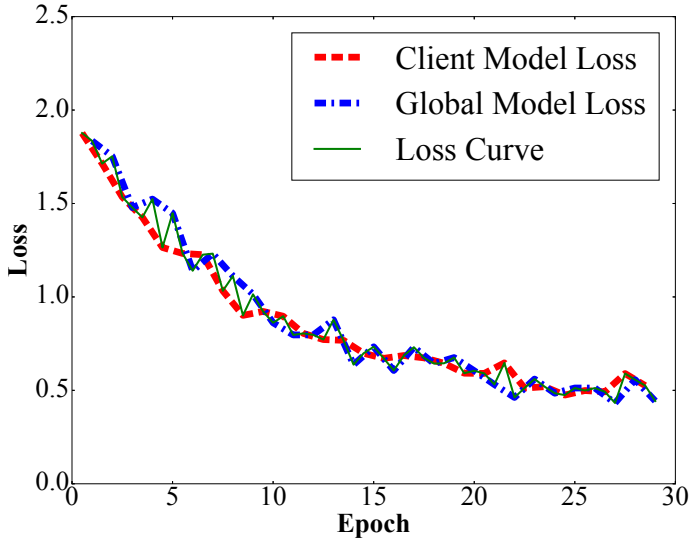
Figure 3.2: Semantic segmentation loss curves with different momentum: (a) 0, (b) 0.5, and (c) 0.9.



(a)

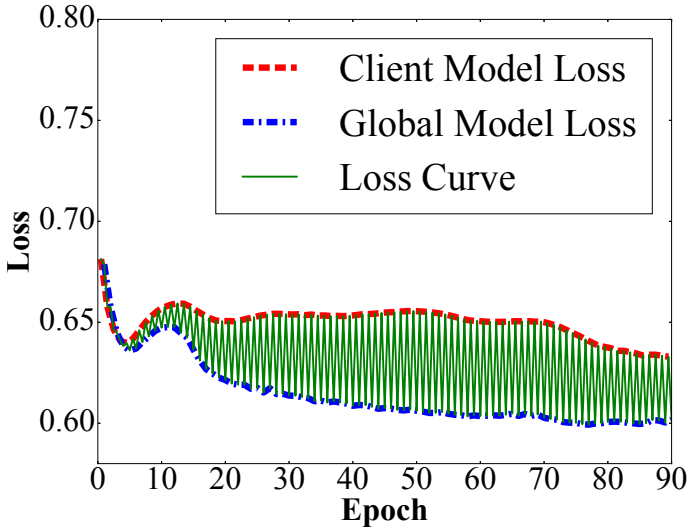


(b)

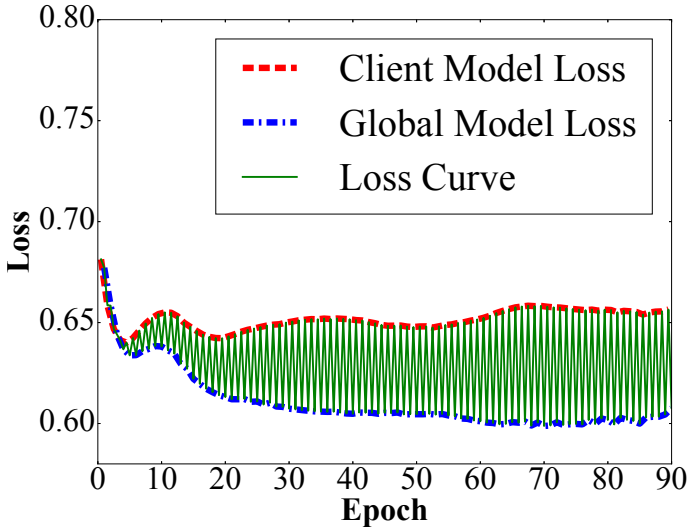


(c)

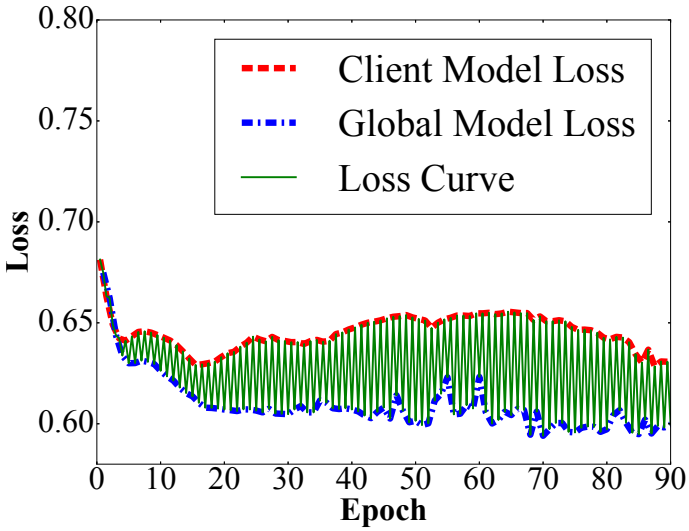
Figure 3.3: Emotion recognition loss curves with different learning rate decay: (a) 0.91, (b) 0.93, and (c) 0.95.



(a)

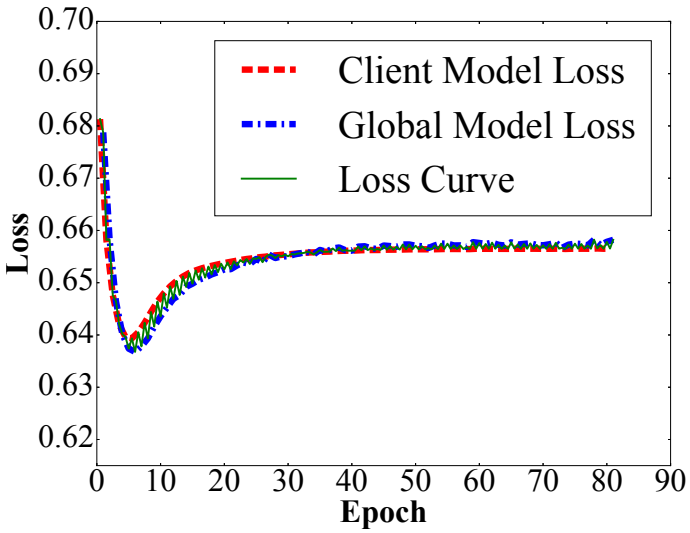


(b)

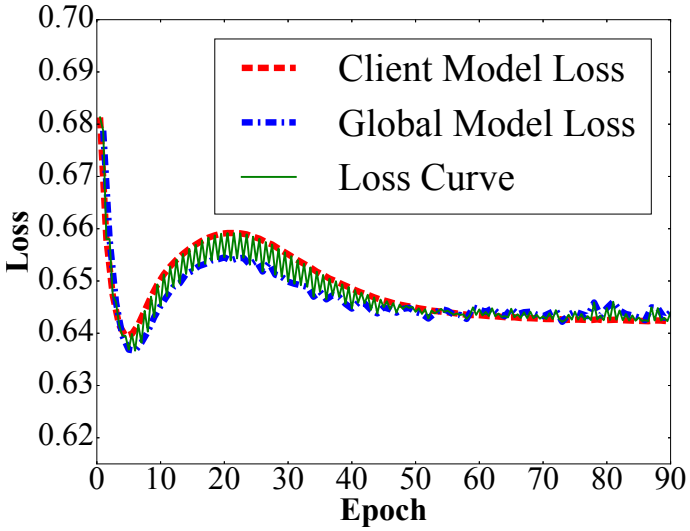


(c)

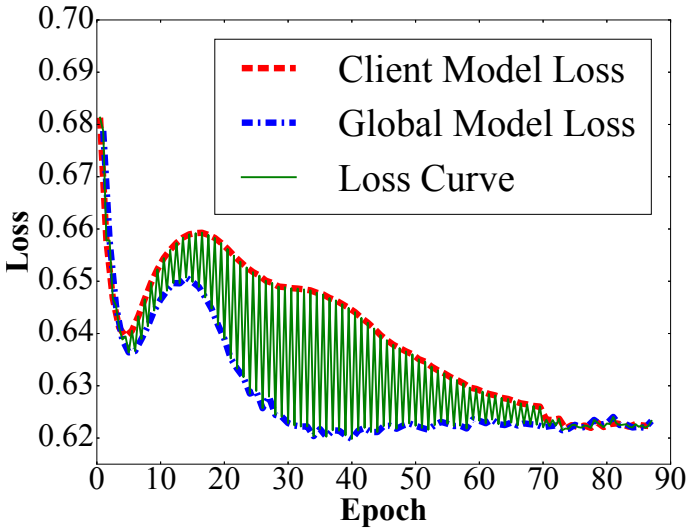
Figure 3.4: Emotion recognition loss curves with different momentum: (a) 0, (b) 0.5, and (c) 0.9.



(a)



(b)



(c)

Figure 3.5: Emotion recognition loss curves with different learning rate decay: (a) 0.9, (b) 0.92, and (c) 0.94.

Chapter 4

Related Work

In this chapter, we introduce research related to our work. We discuss three areas: Multimodal Federated Learning, Federated Semi-Supervised Learning, and Modality-Aware Selective Data Sharing.

4.1 Multimodal Federated Learning

Multimodal Federated Learning (MFL) allows the model to leverage a diverse set of data modalities, enhancing its ability to capture complex patterns and insights that might not be possible with unimodal data. The approaches to MFL can be naturally grouped into several related research directions, each tackling different facets of the field. The first cluster focuses on enhancing model performance with modality-specific features. Xiong et al. [64] proposed a unified framework for MFL that addresses the challenges of modality discrepancy and limited high-quality labeled data in traditional FL methods. The framework employs three key components: a co-attention mechanism for integrating complementary cross-modal information, an advanced FL algorithm for extracting global features across modalities, and a MAML-based personalization technique to tailor the final model to individual clients. This approach aims to leverage the benefits of multimodal data while preserving privacy and improving model performance in FL settings. Similarly, Yang et al. [67] introduced a feature-disentangled activity recognition network (FDARN) to address the new task of cross-modal federated human activity recognition (CMF-HAR). The network is structured around five essential components: two distinct encoders (altruistic and egocentric), dual activity classifiers (shared and private), and a modality discriminator. The FDARN aims to learn modality-agnostic features across clients while preserving modality-specific characteristics collaboratively. It utilizes decentralized optimization with a spherical modality discriminative loss to achieve good generalization across clients and capture client-specific discriminative features. This ap-

proach is designed to promote large-scale deployment of HAR models on local devices while addressing the challenges of cross-modal FL. Zhao et al. [70] enhanced the existing approaches by introducing a framework that combines multimodal and semi-supervised federated learning, utilizing autoencoders to derive shared or correlated representations from diverse local data modalities across client devices. This approach incorporates a novel multimodal FedAvg algorithm to consolidate locally trained autoencoders across various data modalities. By utilizing auxiliary labeled data on the server, the framework employs the global autoencoder for downstream classification tasks. Experiments conducted on a diverse dataset, including sensory data and both depth and RGB camera videos, reveal enhanced classification performance when integrating multiple modalities into federated learning. The framework demonstrates robust cross-modal generalization, achieving F1 scores of up to 60% when applying models trained on single-modality labeled data to test data from different modalities. This performance is particularly notable when leveraging inputs from both unimodal and multimodal clients.

Another significant research direction is the application of FL to specific fields. Bernecker et al. [7] addressed challenges in liver segmentation across multi-modal medical imaging using FedNorm and FedNorm+, two FL algorithms with modality-based normalization. These methods overcome privacy concerns and data heterogeneity issues, achieving Dice scores up to 0.961 across 428 patients from six databases. The approaches outperform local models and are comparable to centralized models. In parallel, Agbley et al. [1] applied MFL to melanoma disease analysis, fusing skin lesion images with clinical data while preserving patient privacy. The FL model's performance nearly matched CL, with only slight differences in F1-Score and Accuracy. FL demonstrated higher sensitivity and competitive results compared to the literature, effectively learning predictive models without sharing training data.

Advancing FL with cross-modal retrieval and grounding is another area of focus. Zong et al. [74] cross-modal retrieval by proposing federated cross-modal retrieval (FedCMR), which learns from decentralized multi-modal data. The method trains local models, aggregates a common subspace on a central server, and updates local models accordingly. This approach addresses privacy concerns and maintenance costs while leveraging FL benefits, demonstrating effectiveness across four benchmark datasets. Complementarily, Liu et al. [42] merged image representations from various vision-and-language grounding tasks using a federated learning framework called aimNet (Aligning, Integrating and Mapping Network). This approach combines features from different tasks to create more powerful representations. Tested on image captioning and VQA, aimNet demonstrated significant improvements over baselines, with up to 14% gains in task-specific metrics. Guo et al. [25] centered on the method called Contextual Optimization

(CoOp) to adapt pre-trained vision-language models in FL. However, current FL methods lack personalization. To address this, pFedPrompt leverages multimodality to personalize prompts, enhancing performance. Extensive experiments confirm pFedPrompt's superiority and robust performance across datasets. Liang et al. [38] proposed an FL algorithm that learns compact local representations on devices and a global model across all devices, reducing communicated parameters. This approach enhances communication efficiency, handles heterogeneous data, and maintains privacy, with theoretical and empirical validation demonstrating reduced variance and fair representation learning.

However, it's important to note that these studies often overlook the aspect of data sensitivity differentiation. This aspect is crucial in FL due to the diverse nature of data sources and the varying levels of privacy concerns associated with different types of data. By considering data sensitivity, MFL systems can ensure more robust privacy protection, build trust with data providers, and comply with varying regulatory requirements. This tailored approach not only enhances data security and user trust but also improves the overall efficacy and adaptability of the learning model to diverse data environments.

4.2 Federated Semi-Supervised Learning

In traditional FL, models are trained collaboratively across multiple clients without sharing their data, thus preserving privacy. However, FL typically assumes that sufficient labeled data are available for training, which isn't always the case. On the other hand, SSL can leverage a large amount of unlabeled data along with a small portion of labeled data to improve learning efficacy. Therefore, Federated Semi-Supervised Learning (FSSL) addresses a common real-world limitation: the scarcity of labeled data.

Xu et al. [65] introduced the Ada-FedSemi system, which combines on-device labeled data with cloud-based unlabeled data to enhance deep learning model performance in FL. Using a multi-armed bandit algorithm, it adaptively determines client participation and pseudo-label confidence, achieving higher accuracy and lower training costs, especially with heterogeneous clients. Diao et al. [15] tackled the issue of unlabeled data in FL by introducing SemiFL, which combines communication-efficient FL with Semi-Supervised Learning. In SemiFL, clients train with unlabeled data while the server fine-tunes the global model with labeled data, significantly improving performance and outpacing existing SSFL methods. Jeong et al. [32] addressed the challenge of unlabeled data in FL by proposing Federated Matching (FedMatch), which improves upon traditional methods with inter-client consistency loss and parameter decomposition for disjoint learning on labeled and unlabeled data. This approach outperforms both local semi-supervised learning and naive FL combinations. Fan et al. [18] focused on developing a federated semi-

supervised learning framework (FedSSL) to train models using scarce and unevenly distributed labeled data. By creating a unified data space and integrating differential privacy, FedSSL effectively leverages both labeled and unlabeled data, achieving a 5-20% performance boost on SSL tasks with minimal labeled data. Finally, Liang et al. [39] considered the challenge of FSSL in a Non-IID setting, introducing RSCFed, which addresses uneven model reliability among clients. By using random sub-sampling and distance-reweighted aggregation, RSCFed effectively distills sub-consensus models, achieving superior performance on various datasets compared to state-of-the-art methods.

4.3 Modality-Aware Selective Data Sharing

In the realms of MFL and FSSL, recent advancements showcase enhanced model performance through diverse data modalities and the innovative use of both labeled and unlabeled data. MFL studies have improved healthcare diagnostics and personalized learning, focusing on modality-specific features and cross-modal retrieval. FSSL addresses labeled data scarcity so as to optimize data use and enhance model efficacy. However, both fields often overlook data sensitivity differentiation. To our best knowledge, modality-aware selective data sharing has only been mentioned by Chen et al. [12], but their method is limited to ideal scenarios where data are all labeled for training. This is why MAFS is critical, as it generalizes to more realistic use cases.

Chapter 5

Problem Statement

To increase the model's training data as much as possible in the FL scenario with scarce labeled data, we considered that different clients have subjective opinions on sharing data of different modalities. As Fig. 5.1 shows, we refer to the data that the client is willing to share as insensitive data and the data that they are not willing to share as sensitive data. Based on the above scenario, we consider K clients, for all $i \in \{1, 2, \ldots, k\}$. Each client collects a dataset $\mathbf{D}_i = \mathbf{D}_i^{SL} \cup \mathbf{D}_i^{SU} \cup \mathbf{D}_i^{IL} \cup \mathbf{D}_i^{IU}$. We use $\mathbf{D}_i^{SL}, \mathbf{D}_i^{SU}, \mathbf{D}_i^{IL}, \mathbf{D}_i^{IU}$ to represent sensitive labeled data, sensitive unlabeled data, insensitive labeled data and insensitive unlabeled data of i-th client, respectively. Our goal is to solve the problem of declining model performance caused by labeled data scarcity by leveraging the insensitive data the client is willing to share.

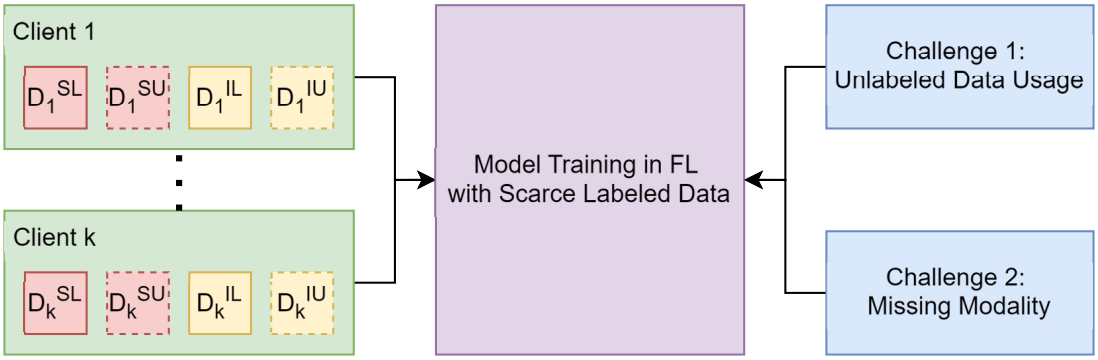


Figure 5.1: Problem Statement.

Challenge 1: Unlabeled data usage. The advantage of SSL over UL is that SSL applies the information in labeled data to unlabeled data, allowing the model to gain better generalization and robustness to noise [72, 37, 6]. Hence, we have decided to use SSL to handle a large amount of unlabeled data. SSL can be divided into two scenarios: client- and server-side SSL. Almost all prior FSSL methods adopted client-side SSL, where each client uses labeled and unlabeled data to train an SSL model and then sends this model

to the server for aggregation. Since clients can only use their own data for SSL, if the distribution of the unlabeled data is not similar to that of the labeled data, it will cause the SSL model to be biased toward the labeled data at the clients. For example, if most actions in client A's labeled data are drinking water, and the unlabeled data contains many phone-calling actions, client A's SSL model may not be able to learn the phone-calling actions effectively. Although FL algorithms can aggregate SSL models, while considering the fractions of labeled data to reduce model bias, the effect may not be obvious. We propose to address this challenge by leveraging server-side SSL combined with data sensitivity differentiation. Server-side SSL can learn from labeled data shared by different clients to benefit others with unlabeled data. Based on the data sensitivity differentiation, we allow users to subjectively choose which modalities to share. This method allows us to harness the power of SSL on the server while minimizing privacy risks, striking a balance between model performance and user privacy protection.

Challenge 2: Missing modality. Clients have varying degrees of willingness to share different data modalities, which can lead to the problem of missing modalities in the model input. For example, in a multimodal model that requires audio, video, and text inputs. If a client is only willing to share the text modality, the audio and video inputs would become missing modalities. Researchers proposed methods to handle missing modalities, including modality imputation [56, 48], and adaptive fusion [63]. However, these methods require modifications to the original model structure and training samples, or even a new model to generate full-modality data. These methods increase the time and engineering complexity. We propose to address this challenge by simply filling the missing modality input with zeros. This method is conservative, giving a lower bound of model performance while significantly reducing the complexity.

Chapter 6

Proposed Solutions

6.1 Overview

We propose the Modality-Aware Federated Semi-Supervised Learning (MAFS) paradigm to utilize the data these clients are willing to share, increasing the model's training data and mitigating the performance degradation brought by FL. Our more general focus is on obtaining more raw data by leveraging the characteristic that each client has a different willingness to share data of different modalities rather than focusing on designing a brand-new model architecture or new feature extraction methods to improve the model performance of FSSL.

Fig. 6.1 shows the training workflow of the MAFS. The client holds data from multiple modalities. Among these, not all data have labels. We refer to the data with labeled data, represented in the figure with solid-line boxes, while those without labels are referred to as unlabeled data, represented with dashed-line boxes. Moreover, based on data sharing considerations, each client categorizes these various modalities of data into two types: sensitive and insensitive data, represented in the figure with red and yellow boxes, respectively. Before training begins, the client sends the labeled and unlabeled insensitive data to the server, as well as all the yellow boxes in the figure.

During the training process on the client side, each client first utilizes a client trainer to train their respective labeled data and generate their own client model, denoted as M^C in the diagram. Upon completion of training, all clients send their respective client models to the server. The server first utilizes an aggregator to aggregate all collected client models into one aggregated model, denoted as M^A in the diagram, adopting FedAvg [45] approach after receiving client models from each client. During the SSL process, the server feeds insensitive unlabeled data from different clients into the aggregated model, retains data that meets certain criteria, and generates a pseudo-label dataset. Subsequently, the server employs a server trainer to train the insensitive labeled data shared by clients

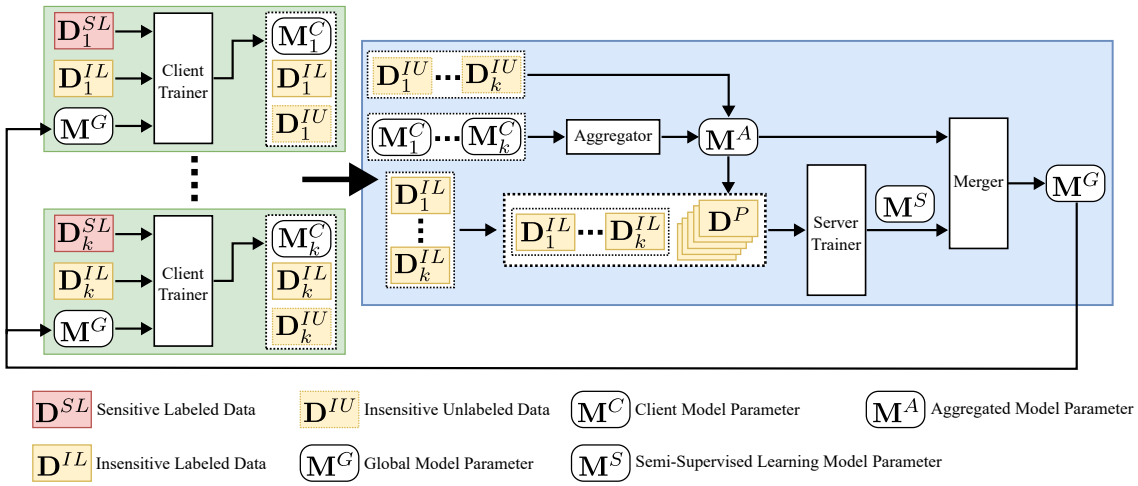


Figure 6.1: Training workflow of the Modality-Aware Federated Semi-Supervised Learning (MAFS).

and the newly generated pseudo-label dataset to produce an SSL model, denoted as $\mathbf{M}^S$ in the diagram. Then, the server uses a merger to merge the aggregated and SSL models to generate the final global model, denoted as $\mathbf{M}^G$ in the diagram. Finally, the server sends this global model back to all clients for the next round of training, enabling all clients to possess knowledge and information of both labeled and unlabeled data simultaneously.

6.2 Notations

We consider K clients, for all $i \in \{1, 2, \ldots, K\}$. Each client collects a dataset $\mathbf{D}_i = \mathbf{D}_i^{SL} \cup \mathbf{D}_i^{SU} \cup \mathbf{D}_i^{IL} \cup \mathbf{D}_i^{IU}$. We use $\mathbf{D}_i^{SL}, \mathbf{D}_i^{SU}, \mathbf{D}_i^{IL}, \mathbf{D}_i^{IU}$ to represent sensitive labeled data, sensitive unlabeled data, insensitive labeled data and insensitive unlabeled data of i-th client, respectively. Considering the whole training process with T rounds, we use $\mathbf{M}_{i,t}^C$ to represent the client model of i-th client in t-th round. We also use $\mathbf{D}_t^P, \mathbf{M}_t^A, \mathbf{M}_t^S, \mathbf{M}_t^G$ to represent pseudo-label dataset, aggregated model parameter, SSL model parameter and global model parameter in each round, respectively. During the training, we use τ to represent the pseudo-label threshold and use hyperparameter α to represent the weight that will be applied to merge the SSL model $\mathbf{M}^S$ and the aggregated model $\mathbf{M}^A$.

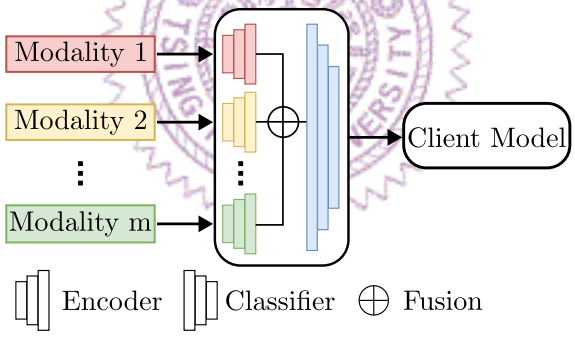
6.3 Client Trainer

The purpose of the client trainer is to allow each client model to learn its features and distribution from its own collected data. In each round t, client i would use its collected labeled data $\mathbf{D}_i^L = \mathbf{D}_i^{SL} \cup \mathbf{D}_i^{IL}$ with labels Y_i to train its client model $\mathbf{M}_{i,t}^C$ by its client

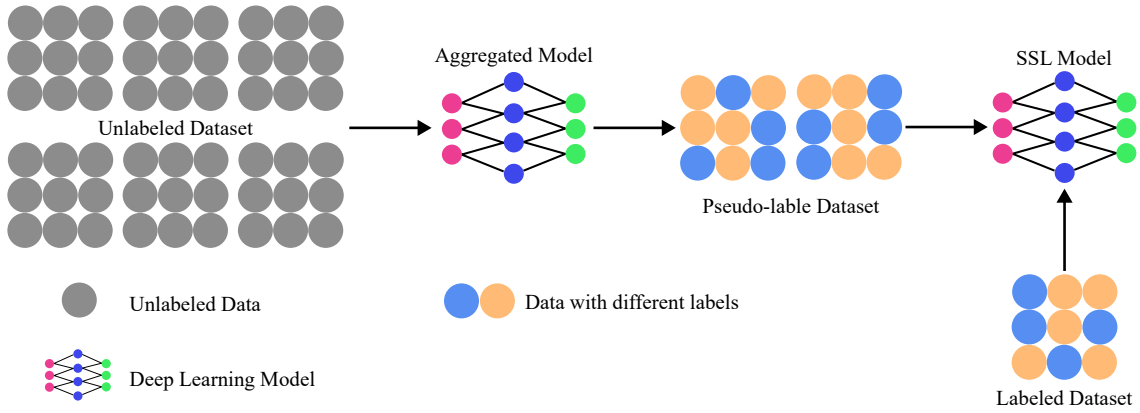
trainer.

$$\mathbf{M}_{i,t+1}^{C} = \underset{\mathbf{M}_{i,t}^{C}}{argmin}\, L_i(\mathbf{D}_i^L, Y_i \mid \mathbf{M}_{i,t}^{C}), \text{ where}$$
$$L_i(\mathbf{D}_i^L, Y_i \mid \mathbf{M}_{i,t}^{C}) = \frac{1}{|\mathbf{D}_i^L|} \sum CE(\mathbf{M}_{i,t}^{C}(\mathbf{D}_i^L), Y_i), \tag{6.1}$$

$L_i(\cdot)$ represents the client loss function. Developers can choose a suitable loss function according to different tasks or applications, such as Cross Entropy (CE) or Mean Square Error (MSE), among others. Similarly, the model architecture used for training client models can also be different depending on the developer's task, and the client trainer's training steps will also differ. Fig. 6.2(a) illustrates the operation of the model trainer under one model architecture. This client trainer comprises three steps: (i) using an encoder to convert raw data into feature vectors, (ii) fusing feature vectors of different modalities through mid-level fusion (e.g., concatenation, matrix multiplication, etc.), and (iii) feeding the fused vector into a classifier to obtain the output, eventually resulting in the client model, which is then sent to the server.



(a)



(b)

Figure 6.2: Training workflows of: (a) client model and (b) server trainer.

6.4 Aggregator

The most important core concept of MAFS is how to utilize the insensitive unlabeled data that clients are willing to share. We hope to use the pseudo-labeling method to select suitable unlabeled data to augment the training dataset. Like traditional FL algorithms, we use an aggregator to aggregate client models from different clients into an aggregated model $\mathbf{M}_t^A$. We believe that this aggregated model contains knowledge from different client data, and the pseudo-label data obtained through pseudo-labeling using this aggregated model has higher reliability. After collecting $\mathbf{D}_i^I$ and $\mathbf{M}_{i,t}^C$ from all the clients, the server aggregator aggregates the client models to generate the aggregated model $\mathbf{M}_t^A$. Here, the aggregator applies FedAvg as the aggregation algorithm, where

$$\mathbf{M}_t^A = \frac{1}{K}\sum_{i=1}^{K}\mathbf{M}_{i,t}^C. \tag{6.2}$$

6.5 Server Trainer

Training the SSL model with the server trainer involves two steps: (i) generating a pseudo-label dataset and (ii) training using both the labeled dataset and the pseudo-label dataset. Fig. 6.2(b) shows the server trainer's more detailed training process. The server trainer first uses the aggregated model $\mathbf{M}_t^A$ to pseudo-label all the insensitive unlabeled data $\mathbf{D}^{IU} = \mathbf{D}_1^{IU} \cup \mathbf{D}_2^{IU} \cup \cdots \cup \mathbf{D}_K^{IU}$ shared by different clients, employing the same architecture as the client model (as referenced in Fig. 6.2(a)). We believe that when using the aggregated model for pseudo-labeling, compared to each client using only its own client model to pseudo-label unlabeled data, the labels determined by the aggregated model through integrating knowledge from different client models and pseudo-labeling the unlabeled data are more accurate. Since not all modalities have data when clients share unlabeled data, our approach to addressing this missing modality issue is to replace the missing input modality data with zeros directly. Once the prediction probability of the unlabeled data is higher than the threshold τ, this unlabeled data would be grouped into the pseudo-label dataset $\mathbf{D}_t^P$.

$$\begin{aligned}
&\mathbf{D}_t^P = \{x \in \mathbf{D}^{IU} \mid f(x) > \tau\},\\
&Y_t^P = \{f(x) \mid x \in \mathbf{D}_t^{IU} \wedge f(x) > \tau\}, \text{ where}\\
&f(x) = argmax(\mathbf{M}_t^A(x)),
\end{aligned} \tag{6.3}$$

Then, The server uses both the insensitive labeled dataset $\mathbf{D}^{IL} = \mathbf{D}_1^{IL} \cup \mathbf{D}_2^{IL} \cup \cdots \cup \mathbf{D}_K^{IL}$ and the pseudo-label dataset $\mathbf{D}_t^P$ to train the SSL model $\mathbf{M}_t^S$. Again, the model architecture of the SSL model and the client models are the same, and we also replace the

missing modality data with zeros. Here, we use the aggregated model $\mathbf{M}_t^A$ as the initial model parameters instead of using random model parameters due to the faster convergence speed. The training process is similar to the client trainer, and the loss function of the SSL model is as follows:

$$\begin{aligned} \mathbf{M}_t^S &= \underset{\mathbf{M}_t^A}{argmin}\, L_S(\mathbf{D}^{IL}, \mathbf{D}_t^P, Y^{IL}, Y_t^P \mid \mathbf{M}_t^A), \text{ where} \\ L_S(\cdot) &= \frac{\sum CE(\mathbf{M}_t^A(\mathbf{D}^{IL}), Y^{IL}) + \sum CE(\mathbf{M}_t^A(\mathbf{D}_t^P), Y_t^P)}{|\mathbf{D}^{IL}| + |\mathbf{D}_t^P|}, \end{aligned} \tag{6.4}$$

Here, $L_S(\cdot)$ is the total loss of the insensitive labeled dataset $\mathbf{D}^{IL}$ and the pseudo-label dataset $\mathbf{D}_t^P$ with label set Y^{IL} and Y_t^P, respectively.

6.6 Merger

Unlike the aggregated model, which contains only knowledge from clients' labeled data, the SSL model incorporates knowledge from pseudo-label data. However, we cannot guarantee that the labels of these pseudo-labeled data are completely correct. In contrast, the training data seen by the aggregated model all have correct labels. Therefore, we use a merger to balance the high accuracy of the small amount of labeled data and the label uncertainty of the large amount of pseudo-labeled data. The server merger would merge the aggregated model $\mathbf{M}_t^A$ and the SSL model $\mathbf{M}_t^S$ to generate the final global model $\mathbf{M}_t^G$:

$$\mathbf{M}_t^G = \alpha \times \mathbf{M}_t^A + (1 - \alpha) \times \mathbf{M}_t^S, \tag{6.5}$$

balanced by a hyperparameter α. At the end of each round, the server would send the global model $\mathbf{M}_t^G$ back to all the clients for the next training round. If the global model $\mathbf{M}^G$ converges, the MAFS training procedure will be stopped.

Chapter 7

Multimodal Applications

7.1 Emotion Recognition Dataset

We use the IEMOCAP [8] dataset for the Emotion Recognition (ER) application experiment. This dataset captures the dialogues of two actors. The facial and tonal changes during the actors' conversations are recorded separately using a camera and a microphone to capture the emotional expressions. Additionally, the dialogue content is recorded in a purely textual format. The dataset includes 4453 triplets with three different modalities of data, represented by audio, video, and text, capturing the overall conversation processes of the two actors. We split the dataset into training and testing data at an 8:2 ratio, resulting in 3515 and 938 instances, respectively.

The 3515 training data instances are further allocated to $\{8, 16, 32\}$ clients, with 30% of the allocated training data for each client being treated as labeled data and the remaining 70% as unlabeled data to simulate a scenario with the scarcity of labeled data. Each client trainer uses labeled data in all modalities allocated to it to train its respective client model. When clients share their client models with the server, they selectively share their unlabeled data for one or two unspecified modalities. The server uses the 938 testing data instances for model performance evaluation to evaluate the global model.

7.2 IEMOCAP Neural Networks

We adopted the Low-rank-Multimodal-Fusion (LMF) [44] approach as our neural network architecture during training, with a more detailed model architecture shown in Fig. 7.1. The audio and video inputs are passed through an encoder with three fully connected layers, while the text input is passed through an encoder with one LSTM layer and one fully connected layer. The modality features are then fused through matrix multiplication to obtain the output.

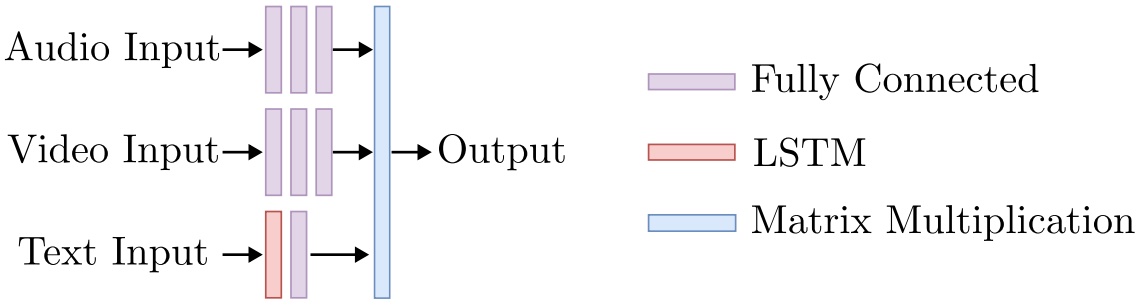


Figure 7.1: Neural network for LMF.

7.3 Human Activity Recognition Dataset

We use KU-HAR [54] as the dataset for our Human Activity Recognition (HAR) experiments. This dataset collected data from 90 users using wearable devices. The user data includes two modalities, accelerometer and gyroscope data, to recognize user actions, with 18 actions. Due to hardware computational resource limitations, we did not use the complete KU-HAR dataset. We followed the method from the FedMultimodal [19] to use a portion of the dataset. This paper is the first to propose a benchmark for MFL, covering various multimodal applications, multimodal datasets, and FL algorithms and comparing the performance of different algorithms. In the KU-HAR dataset, they selected only 65 users and included only 8 actions. We used their lightweight version of the KU-HAR dataset for our experiments.

Unlike IEMOCAP, KU-HAR treats each user as a client, eliminating the need to divide the dataset into different clients using a Dirichlet parameter. Among the 65 users, 63 are used for training data, 1 for validation, and 1 for testing. Each client will use 70% of their data as unlabeled data, simulating a scenario of labeled data scarcity. During the MAFS training process, each client will first train their individual client model using 30% of the labeled data, and then share insensitive labeled data and unlabeled data with the server. Finally, the server will perform tests on the test client to evaluate the performance of MAFS.

7.4 KU-HAR Neural Networks

We also referred to FedMultimodal's neural network architecture for training KU-HAR. Fig. 7.2 provides a more detailed explanation of the neural network we used. The accelerometer input and gyroscope input each pass through 3 layers of 1D convolution layers and 1 layer of Gated Recurrent Unit (GRU) to obtain their respective modality feature vectors. Then, concatenation-based fusion combines these two feature vectors into a single feature vector. Subsequently, this fused feature vector is fed into a classifier composed

of two fully connected layers to obtain the output.

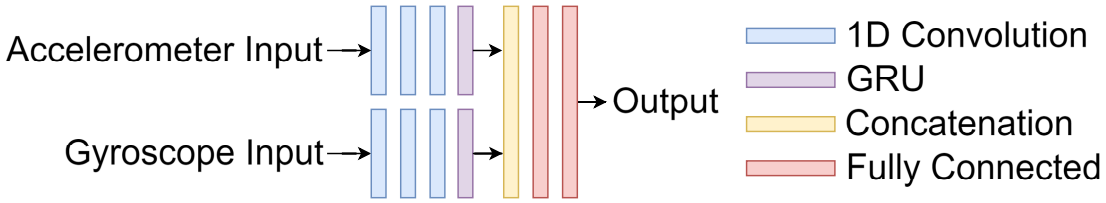


Figure 7.2: Neural network for LMF.

Chapter 8

Evaluations

8.1 Implementations

We referred to the FL algorithms selected by FedMultimodal and chose three of them: FedAvg, FedProx, and FedOpt, to compare with the MAFS paradigm. All experiments were implemented using PyTorch 1.7.1 and Python 3.8.5 with CUDA 10.1 acceleration, and were run on an Intel E5 server at 2.50 GHz with 4 NVIDIA GTX1080Ti GPUs.

8.2 Hyperparameters

For the ER experiment, we set the hyperparameters as follows: (i) 100 rounds, each round containing three client epochs and ten server epochs (ii) batch size of 16, (iii) Cross-Entropy (CE) as the loss function, (iv) learning rate $\eta_t = 0.003 \times 0.965^{t-1}$, (v) the client trainer using the Adam optimizer with a weight decay of 0.002, and (vi) the server trainer using the SGD optimizer with a momentum of 0.9 and a weight decay of 0.0005. For the HAR experiment, we set the hyperparameters as follows: (i) 200 rounds, each round containing one client epoch and ten server epochs (ii) batch size of 16, (iii) NLLLoss as the loss function, (iv) learning rate $\eta_t = 0.001$, (v) both the client trainer and the server trainer using the SGD optimizer with a momentum of 0.9 and a weight decay of 0.0005. Additionally, when the FL algorithm is FedProx, we set the regularization parameter as 0.001.

8.3 System Parameter Settings

We conducted experiments and discussions within the MAFS paradigm for the following parameters. For the ER task, the default values shown in bold: (i) $c \in \{\mathbf{8}, 16, 32\}$, (ii) $\tau \in \{0.5, \mathbf{0.6}, 0.7, 0.8, 0.9\}$, (iii) $\alpha \in \{\mathbf{0.1}, 0.3, 0.5, 0.7, 0.9\}$, (iv) Dirichlet Distribution

Parameter $\in \{\mathbf{0.1}, 1, 10\}$. For the HAR task, the default values shown in bold: (i) $\tau \in \{0.5, \mathbf{0.6}, 0.7, 0.8, 0.9\}$, (iii) $\alpha \in \{\mathbf{0.1}, 0.3, 0.5, 0.7, 0.9\}$.

8.4 Results

In this section, we discuss the impact of different system parameters on MAFS performance. Additionally, we examine how various modality sharing combinations among clients affect MAFS. In the ER experiments, we selected five different seeds to determine the division of training and testing samples, and averaged the experimental results. The standard deviation of accuracy and F1-score for each different set of experiments falls between 0.87% and 1.22%. In the HAR experiments, we similarly selected five different seeds to determine which one of the 65 clients would be the testing client, and averaged the experimental results. The standard deviation of accuracy and F1-score for each different set of experiments falls between 6.50% and 7.39%.

8.4.1 Impact of Pseudo-Labeling Threshold τ.

We conducted experiments for a classification task, using a threshold τ during the pseudo-labeling to determine whether unlabeled data should be added to the pseudo-label dataset for retraining. Specifically, if the model's predicted probability for an unlabeled data point in a particular class is more significant than τ, then that data point is added to the pseudo-label dataset.

We conducted tests with different values of τ on both the ER and HAR tasks. Looking at ER, since the clients share a variety of modalities, we will use the case where all clients are willing to share audio and video data as an example to analyze the impact of τ.

From Fig. 8.1, we can see that when the value of τ is 0.6, the improvement in the global model's accuracy and F1-Score is the largest, reaching 6.94% and 9.49%, respectively.

Fig. 8.2 shows the performance of MAFS on the HAR task with different τ values. From the figure, we can see that regardless of the τ value, its impact on MAFS is minimal. Therefore, in our subsequent experiments, we will use 0.6 as the default value for τ on both ER and HAR tasks.

8.4.2 Impact of Labeled Data Proportion

A labeled data ratio of 30% is commonly used in SSL [72]. For the ER experiment, our default setting also uses 1000 labeled samples out of the total 3515 training samples,

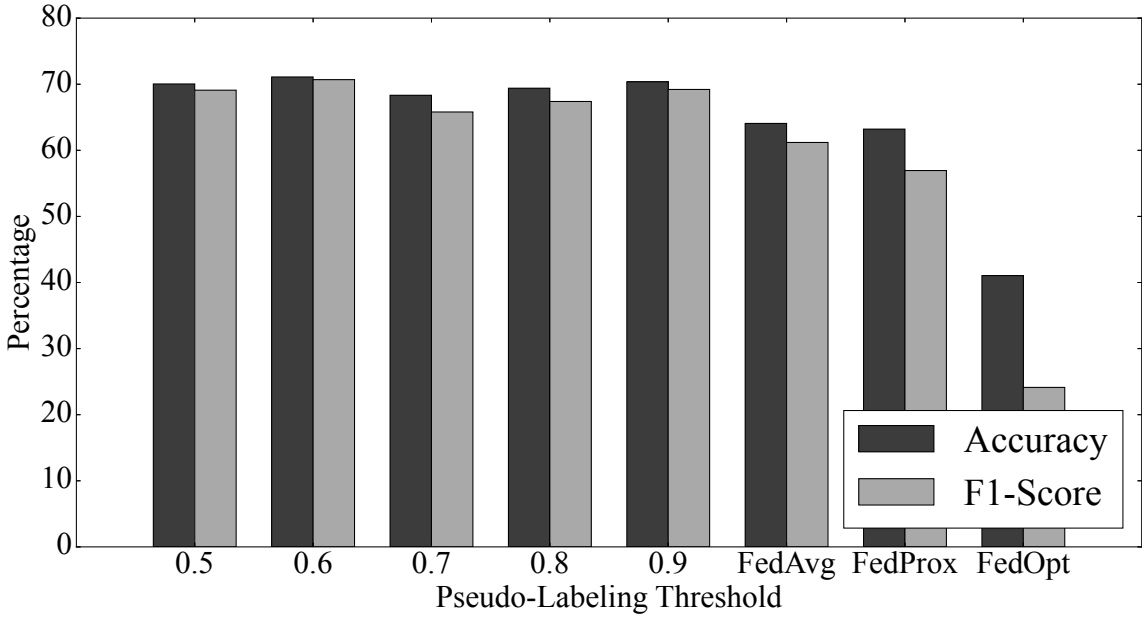


Figure 8.1: MAFS results for ER under different τ values.

which is slightly lower at around 29%. We also evaluated MAFS performance with training sample sizes of $1500, 2000, 2500, 3000$. Again, we will use the case where all clients are willing to share audio and video data as an example to observe the impact of different labeled data proportions on MAFS.

As observed from Table 8.1, training solely with labeled data leads to a decline in both accuracy and F1-score as the quantity of labeled data decreases. However, when employing MAFS for pseudo-labeling and training with unlabeled data, there is an improvement in both accuracy and F1-score, indicating that increasing the amount of training data through this method enhances model performance.

For HAR, in addition to using 30% labeled data rate to simulate a scenario of labeled data scarcity, we also used 40%, 50%, 60%, 70%, and 80% labeled data rates to observe how MAFS performs when there is sufficient labeled data. Furthermore, we used the example where all clients are willing to share accelerometer data. As seen in Table 8.2, MAFS performs the best among all compared methods, regardless of whether the labeled data rate is high or low. Moreover, the lower the labeled data rate, the greater the improvement in model performance by MAFS. For example, when the labeled data rate is 80%, MAFS improves accuracy and F1-score by 12.18% and 16.08% respectively compared to FedAvg. When the labeled data rate decreases to 30%, MAFS improves accuracy and F1-score by 35.43% and 28.57% respectively compared to FedAvg, significantly mitigating the problem of model performance degradation caused by labeled data scarcity. In the following experiments, we will use a labeled data rate of 30% to observe MAFS's performance in addressing the labeled data scarcity problem under different scenarios.

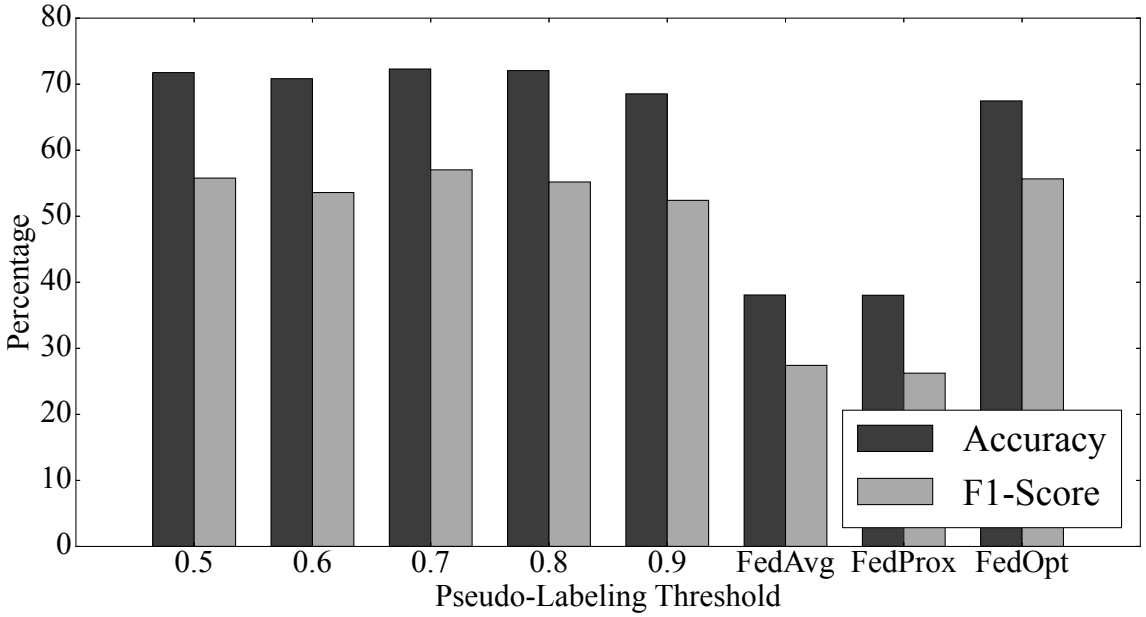


Figure 8.2: MAFS results for HAR under different τ values.

8.4.3 Impact of Merger Weight α

Since the SSL model refers to pseudo-label data during training, inaccurate pseudo-labels could decrease SSL model performance. In contrast, the aggregated model's training samples all have correct labels. Therefore, we use α to adjust the weights of the aggregated and SSL models, with a lower α value indicating a lower contribution from the aggregated model. Again, for the ER task, we will use the case where all clients are willing to share audio and video data as an example to observe the impact of different α values on MAFS. As shown in Fig. 8.3, a gradual decrease in the α value correlates with increases in both model accuracy and F1-score. This suggests that a higher proportion of the SSL model parameter leads to more significant improvements in model performance. Therefore, we recommend setting α to 0.1 in the LMF context.

For the HAR task, we used the example where all clients are willing to share accelerometer data. From Fig 8.4, we can see that MAFS performs best when the α value is 0.1. However, compared to the ER task, changes in the α value do not have a very significant impact on MAFS performance in the HAR task. Considering the results from both ER and HAR, we recommend setting the α value to 0.1. We have also set this as the default value and used it in subsequent experiments.

8.4.4 Impact of the Dirichlet Parameter

We adjusted the Dirichlet parameter for the ER experiment to split the 3515 training samples across eight clients, simulating different degrees of non-i.i.d. data distribution.

Table 8.1: ER model performance comparisons across different labeled data propotion under threshold = 0.6.

Compared to FedAvg					
# of labeled data	1000	1500	2000	2500	3000
Impr. of Accuracy	+3.09%	+2.78%	+0.96%	+0.85%	+0.22%
Impr. of F1-Score	+4.45%	+3.92%	+3.74%	+2.79%	+2.43%
Compared to FedProx					
# of labeled data	1000	1500	2000	2500	3000
Impr. of Accuracy	+3.73%	+3.30%	+2.45%	+2.34%	+1.60%
Impr. of F1-Score	+8.89%	+8.82%	+7.92%	+7.88%	+5.45%
Compared to FedOpt					
# of labeled data	1000	1500	2000	2500	3000
Impr. of Accurcy	+26.12%	+25.69%	+24.84%	+24.73%	+23.99%
Impr. of F1-Score	+41.48%	+41.41%	+40.51%	+40.47%	+38.04%

A smaller Dirichlet parameter indicates a higher degree of non-i.i.d. Again, we will use the case where all clients are willing to share audio and video data as an example to observe the impact of different Dirichlet parameters on MAFS. Fig. 8.5 illustrates the effect of MAFS on model performance under various degrees of non-i.i.d. conditions. From the figure, it can be observed that, regardless of the severity of non-i.i.d., ranging from 0.1, representing the most severe non-i.i.d. condition, to 10, representing the mildest non-i.i.d. condition, there is minimal fluctuation in the accuracy and F1-score of the model. Moreover, MAFS consistently outperforms the scenario without using MAFS by a significant margin. This indicates that even in situations with considerable non-i.i.d., using MAFS substantially increases the amount of data available for the SSL model during training, effectively addressing issues related to uneven data distribution.

8.4.5 Sharing One Modality vs. Two Modalities

Among the different modality-sharing scenarios, we first wanted to observe whether the model performance would be better when clients share two modalities than when sharing only one. Here, we take the ER task as an example and consider the case where all clients share the same types and quantities of modalities, while the case where some clients share one modality and others share two modalities will be discussed in the following subsection. Fig. 8.6 illustrates the impact of MAFS on model performance when different clients share one or two modality data. If clients are willing to share both audio and video modalities, the model performance is the best, with accuracy and F1-score reaching 71.01% and 70.68%, respectively. This difference is only 0.63% and 0.46% compared to

Table 8.2: HAR model performance comparisons across different labeled data propotion under threshold = 0.6.

Compared to FedAvg						
Labeled data rate	30%	40%	50%	60%	70%	80%
Impr. of Accuracy	+35.43%	+34.49%	+29.78%	+24.50%	+14.86%	+12.18%
Impr. of F1-Score	+28.57%	+29.02%	+27.88%	+25.26%	+17.21%	+16.08%
Compared to FedProx						
Labeled data rate	30%	40%	50%	60%	70%	80%
Impr. of Accuracy	+35.47%	+34.40%	+30.94%	+24.34%	+14.74%	+11.52%
Impr. of F1-Score	+29.75%	+29.02%	+28.27%	+24.23%	+17.58%	+15.87%
Compared to FedOpt						
Labeled data rate	30%	40%	50%	60%	70%	80%
Impr. of Accurcy	+6.04%	+5.37%	+4.94%	+4.33%	+3.10%	+3.55%
Impr. of F1-Score	+5.15%	+4.08%	+3.49%	+2.88%	+4.67%	+3.14%

the scenario with 100% labeled data. If clients are willing to share two modalities, such as audio and text, the model's performance is also quite good. If clients are only willing to share one modality, such as video, the performance gap increases to 3.73% and 7.33%, but it still shows a performance improvement of 3.84% and 2.62% compared to not using MAFS. Additionally, we can observe that, in general, sharing two modalities leads to a higher improvement in model performance compared to sharing only one modality.

8.4.6 Impact of Selective Modality Sharing

The above experimental analyses assume that all clients are willing to share the same types and numbers of modalities, which may not reflect real-world scenarios. We conducted additional experiments with different modality-sharing combinations to observe MAFS' performance. We categorized the experiments into three main groups, where clients are willing to share only two modalities: audio and video, audio and text, and video and text. Within each group, we varied the number of clients who are only willing to share one modality, with $2, 4, 6, 8$ clients in this setting. For the remaining clients, they share two modalities, with $6, 4, 2, 0$ clients. Since we want MAFS to perform well even in highly non-i.i.d. scenarios (Dirichlet parameter of 0.1), selecting which clients share only one modality becomes essential. Here, we used the number of training samples as the selection criterion, either starting with: (i) the clients with the least training samples or (ii) the clients with the most training samples.

When comparing sharing one or two modalities (reported above), we found the model

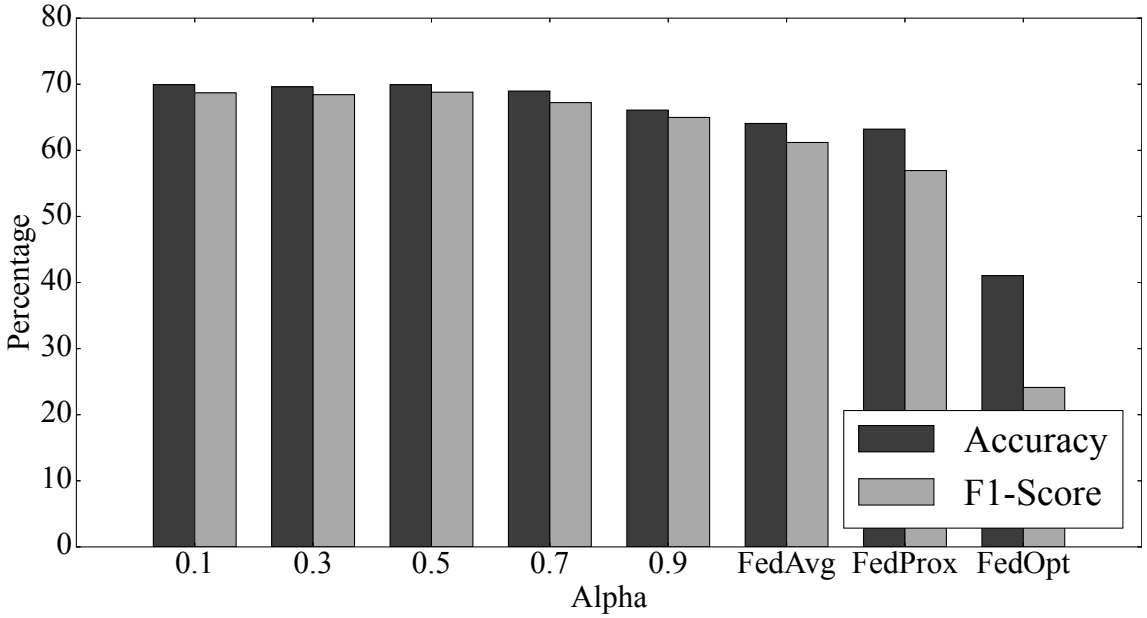


Figure 8.3: MAFS results for ER under different α values.

performance is always better when clients are willing to share two modalities. Among the cases of sharing only one modality, the performance is best for sharing audio, followed by video, and then text. Therefore, in this subsection, where we discuss the mixed modality sharing scenario, we will use the modality with the worst performance as the baseline. Taking Table 8.3 as an example, since the model performance with eight clients sharing only video is lower than eight clients sharing only audio, we use the case of 8 clients sharing only video as the baseline. We then gradually increased the number of clients sharing two modalities to observe the impact of the additional audio data shared by some clients on the model performance. The "number of clients" column indicates the number of clients sharing only one modality. The "client selection" column represents the method used to choose which clients will share two modalities—"least to most" means starting with the clients who have the least training samples, while "most to least" means starting with the clients with the most training samples. From Table8.3 to Table 8.5, we will use video, text, and text as the modality that clients are only willing to share.

From Table 8.3, we can observe that regardless of whether the clients with more training samples or those with fewer training samples are willing to share two modalities, the global model's accuracy and F1-score are improved compared to the case where all clients only share one modality. As the number of clients sharing two modalities increases, the degree of improvement in the global model performance also increases. The most significant performance improvement is seen when four clients are willing to share one modality and the other four are willing to share two modalities. In other words, as long as half of the clients are willing to share more modalities, the model performance can be maximized.

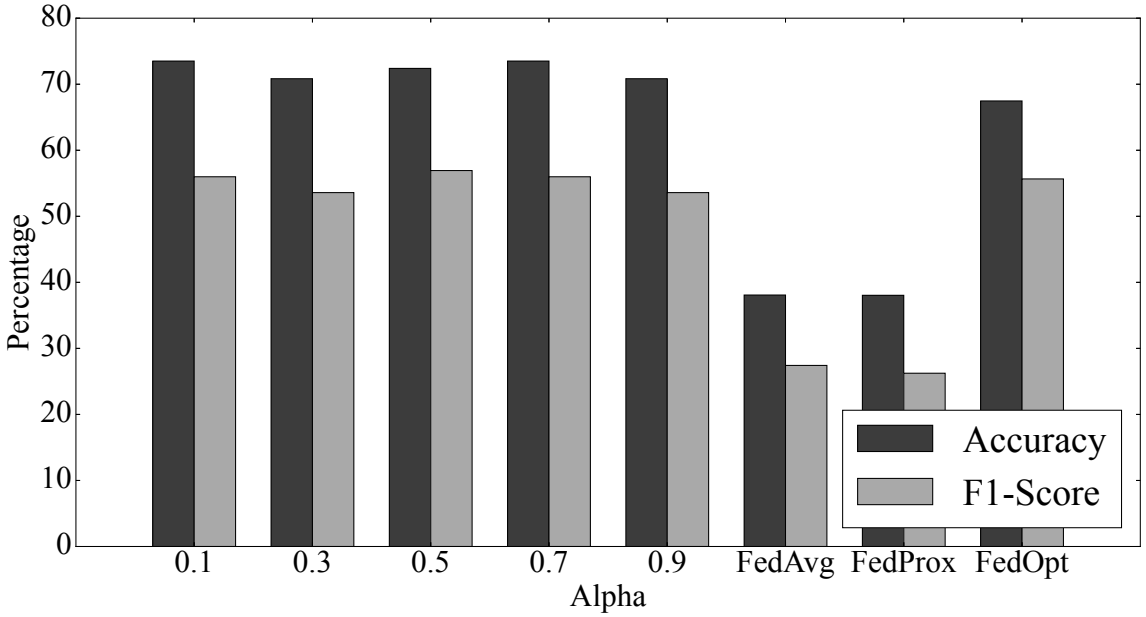


Figure 8.4: MAFS results for HAR under different α values.

Table 8.3: Global Model Performance with Different Numbers of Clients Sharing Audio and Video

Client selection (Based on data amount)	**Least to Most**				**Most to Least**			
# of Clients	8	6	4	2	8	6	4	2
Accuracy	68.12%	+2.02%	+2.77%	+2.66%	68.12%	+2.45%	+3.41%	+2.45%
F1-Score	65.98%	+3.32%	+4.25%	+3.1%	65.98%	+3.98%	+4.95%	+3.51%

This result can also be observed in Table 8.4 and Table 8.5.

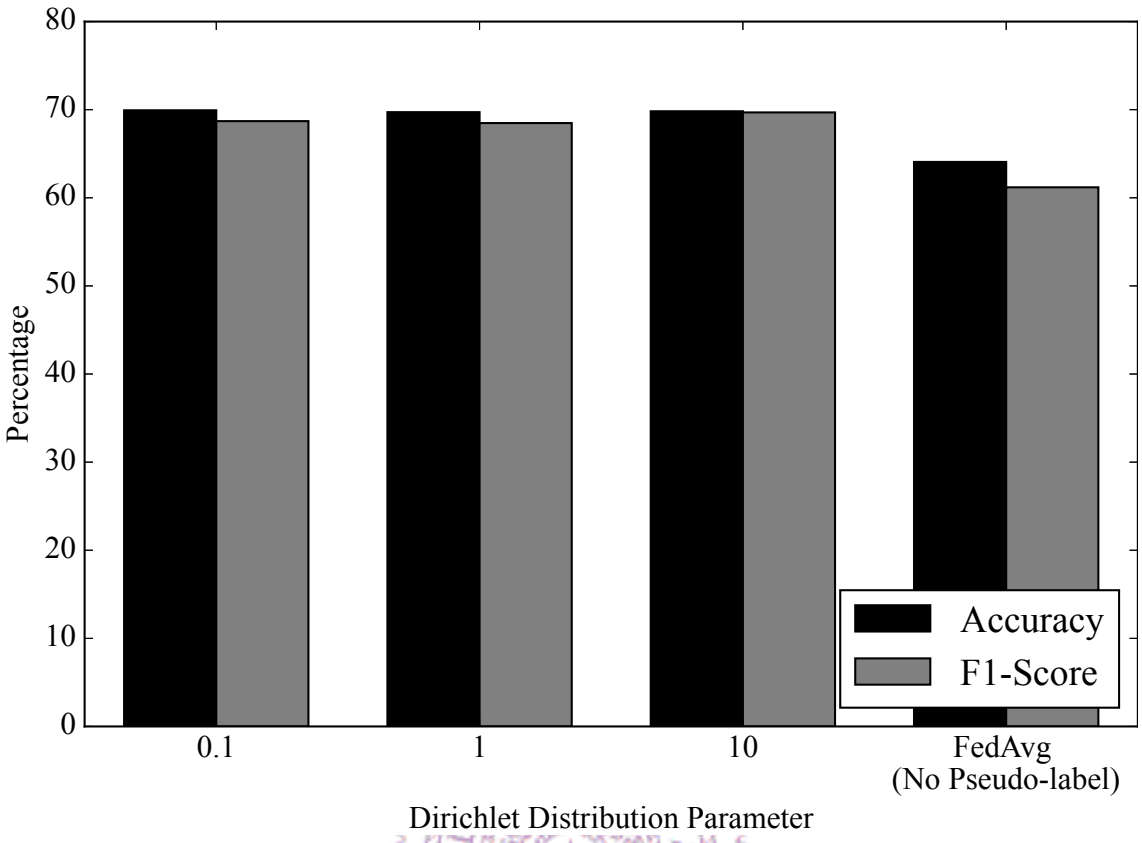


Figure 8.5: Model performance comparisons across different Dirichlet distribution parameters.

Table 8.4: Global Model Performance with Different Numbers of Clients Sharing Audio and Text

Client selection (Based on data amount)	**Least to Most**				**Most to Least**			
# of Clients	8	6	4	2	8	6	4	2
Accuracy	67.91%	+1.27%	+2.98%	+1.81%	67.910%	+1.59%	+2.34%	+2.87%
F1-Score	63.81%	+3.71%	+6.37%	+4.57%	63.81%	+3.92%	+5.51%	+5.99%

Table 8.5: Global Model Performance with Different Numbers of Clients Sharing Video and Text

Client selection (Based on data amount)	**Least to Most**				**Most to Least**			
# of Clients	8	6	4	2	8	6	4	2
Accuracy	67.91%	+1.5%	+1.82%	-2.02%	67.91%	+1.39%	+2.03%	+2.46%
F1-Score	63.81%	+3.5%	+4.02%	+0.21%	63.81%	+3.61%	+5.59%	+6.3%

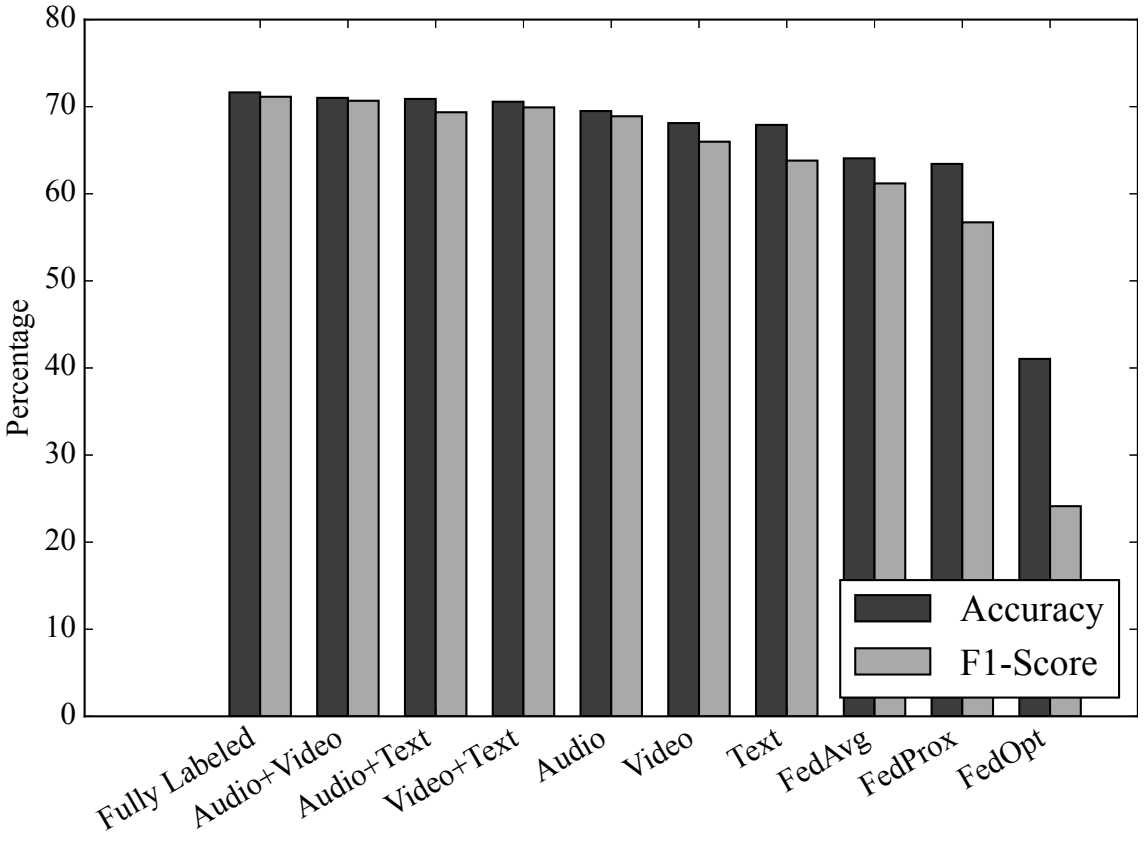


Figure 8.6: Model performance comparisons across different sharing modality types.

Chapter 9

Conclusions & Future Works

This paper proposes the MAFS paradigm to tackle data sensitivity differentiation and labeled data scarcity in FL setup. MAFS is the very first FSSL paradigm that allows individual clients to selectively share insensitive data modalities for augmenting training samples and mitigating labeled data scarcity. Our extensive experiments reveal that: (i) MAFS outperforms SOTAs under different labeled data rates and data distributions; (ii) MAFS allows and encourages clients to selectively share more data modalities, while more clients share more data modalities lead to better model performance; and (iii) MAFS works well in two sample tasks under our recommended hyperparameters, while the same hyperparameter search strategy can be readily applied to other tasks. The proposed MAFS can be extended in several directions, including:

- **Addressing pseudo-label dataset concerns.** The pseudo-label dataset may become biased if labeled and unlabeled data features differ significantly. Solutions like domain adaptation [20] or data augmentation [10] could be employed to ensure the pseudo-label dataset remains representative and sufficiently large.

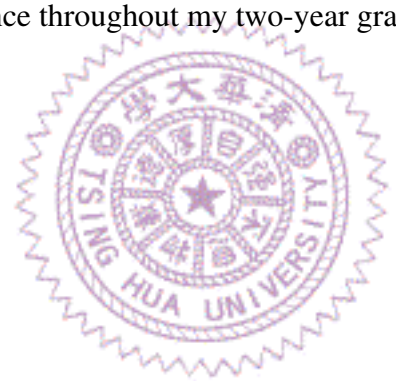
- **Managing dynamic unlabeled data and diverse user participation.** The quantity of unlabeled data may grow as new data are continuously collected, and more diverse users joining the FL setup could affect model performance due to variations in data distribution. Adaptive algorithms that dynamically adjust the model training process could be employed to cope with this issue. Additionally, user stratification or data filtering mechanisms could be integrated to maintain model stability.

- **Promoting client data sharing incentives.** Implementing a reward mechanism could motivate clients to share more data, improving model performance and robustness. These incentives stimulate a collaborative ecosystem where clients feel valued for their contributions to improving overall model accuracy and effectiveness.

- **More applications and baselines.** MAFS achieved good performance on both the IEMOCAP and KU-HAR datasets. However, these experiments cannot comprehensively demonstrate that MAFS is suitable for all multimodal datasets and applications. Other fields such as Social Media, Healthcare, and similar applications can be explored further in the future. Additionally, the current FSSL SOTAs are only applicable to unimodal datasets and cannot be applied to multimodal datasets. To compare with these SOTAs, it would be necessary to adapt them to versions that can handle multimodal datasets, which is also a potential direction for future research.

Acknowledgments

I would like to express my gratitude to Chih-Fan Hsu and to Chung-Chi Tsai and Jian-Kai Wang, representing the Qualcomm team, for their invaluable help and advice during the process of writing and publishing my conference and journal papers. I'm thankful to Hsin-Che Chiang for helping me debug late at night and for assisting in revising the grammar and sentences in my papers. Lastly, I would like to thank my advisor, Professor Cheng-Hsin Hsu, for his tireless assistance throughout my two-year graduate research journey.



Bibliography

[1] L. Agbley, J. Li, A. Haq, K. Bankas, S. Ahmad, O. Agyemang, D. Kulevome, D. Ndiaye, B. Cobbinah, and S. Latipova. Multimodal melanoma detection with federated learning. In *Proc. of 18th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, pages 238–244, Chengdu, China, December 2021.

[2] S. Amershi, M. Cakmak, B. Knox, and T. Kulesza. Power to the people: The role of humans in interactive machine learning. *AI magazine*, 35(4):105–120, 2014.

[3] P. Atrey, A. Hossain, A. El Saddik, and M. Kankanhalli. Multimodal fusion for multimedia analysis: a survey. *Multimedia systems*, 16:345–379, 2010.

[4] T. Baltrusaitis, C. Ahuja, and L.-P. Morency. Multimodal machine learning: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2), 2018.

[5] T. Baltrušaitis, C. Ahuja, and L.-P. Morency. Multimodal machine learning: A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence*, 41(2):423–443, 2018.

[6] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research*, 7(11), 2006.

[7] T. Bernecker, A. Peters, C. Schlett, F. Bamberg, F. Theis, D. Rueckert, J. Weiß, and S. Albarqouni. Fednorm: Modality-based normalization in federated learning for multi-modal liver segmentation. *arXiv preprint arXiv:2205.11096*, 2022.

[8] C. Busso, M. Bulut, C.-C. Lee, A. Kazemzadeh, E. Mower, S. Kim, J. Chang, S. Lee, and S. Narayanan. IEMOCAP: Interactive emotional dyadic motion capture database. *Language Resources and Evaluation*, 42(4), 2008.

[9] E. Celebi and K. Aydin. *Unsupervised learning algorithms*, volume 9. Springer, 2016.

[10] K. Chaitanya, N. Karani, C. Baumgartner, A. Becker, O. Donati, and E. Konukoglu. Semi-supervised and task-driven data augmentation. In *Proc. of 26th Information Processing in Medical Imaging International Conference*, pages 29–41, Hong Kong, China, 2019.

[11] F. Chen, M. Luo, Z. Dong, Z. Li, and X. He. Federated meta-learning with fast convergence and efficient communication. *arXiv preprint arXiv:1802.07876*, 2018.

[12] Y. Chen, C.-F. Hsu, C.-C. Tsai, and C.-H. Hsu. Hpfl: Federated learning by fusing multiple sensor modalities with heterogeneous privacy sensitivity levels. In *Proc. of the 1st International Workshop on Methodologies for Multimedia*, pages 5–14, Lisboa, Portugal, 2022. ACM.

[13] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao. Fedhealth: A federated transfer learning framework for wearable healthcare. *IEEE Intelligent Systems*, 35(4):83–93, 2020.

[14] L. Corinzia, A. Beuret, and J. Buhmann. Variational federated multi-task learning. *arXiv preprint arXiv:1906.06268*, 2019.

[15] E. Diao, J. Ding, and V. Tarokh. Semifl: Semi-supervised federated learning for unlabeled clients with alternate training. *Advances in Neural Information Processing Systems*, 35:17871–17884, 2022.

[16] A. M. Elbir, S. Coleri, and K. V. Mishra. Hybrid federated and centralized learning. In *European Signal Processing Conference (EUSIPCO)*, pages 1541–1545, Dublin, Ireland, 2021. IEEE.

[17] A. Fallah, A. Mokhtari, and A. Ozdaglar. Personalized federated learning: A meta-learning approach. *arXiv preprint arXiv:2002.07948*, 2020.

[18] C. Fan, J. Hu, and J. Huang. Private semi-supervised federated learning. In *Proc. of International Joint Conference on Artificial Intelligence*, pages 2009–2015, Vienna, Austria, July 2022.

[19] T. Feng, D. Bose, T. Zhang, R. Hebbar, A. Ramakrishna, R. Gupta, M. Zhang, S. Avestimehr, and S. Narayanan. Fedmultimodal: A benchmark for multimodal federated learning. In *Proc. of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4035–4045, 2023.

[20] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189, Lille, France, 2015.

[21] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran. An efficient framework for clustered federated learning. *Advances in Neural Information Processing Systems*, 33:19586–19597, 2020.

[22] J. Gou, B. Yu, S. Maybank, and D. Tao. Knowledge distillation: A survey. *Springer International Journal of Computer Vision*, 129(6):1789–1819, 2021.

[23] N. Grira, M. Crucianu, and N. Boujemaa. Unsupervised and semi-supervised clustering: a brief survey. *A review of machine learning techniques for processing multimedia content*, 1(2004):9–16, 2004.

[24] N. Guha, A. Talwalkar, and V. Smith. One-shot federated learning. *arXiv preprint arXiv:1902.11175*, 2019.

[25] T. Guo, S. Guo, and J. Wang. pfedprompt: Learning personalized prompt for vision-language models in federated learning. In *Proc. of the ACM Web Conference*, pages 1364–1374, Austin, TX, April 2023.

[26] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[27] W. Hong, X. Luo, Z. Zhao, M. Peng, and T. Quek. Optimal design of hybrid federated and centralized learning in the mobile edge computing systems. In *ICC*, pages 1–6. IEEE, 2021.

[28] L. Huang, Y. Yin, Z. Fu, S. Zhang, H. Deng, and D. Liu. LoAdaBoost: Loss-based AdaBoost federated machine learning with reduced computational complexity on iid and non-iid intensive care data. *Plos One*, 15(4), 2020.

[29] Y. Huang, L. Chu, Z. Zhou, L. Wang, J. Liu, J. Pei, and Y. Zhang. Personalized cross-silo federated learning on non-iid data. In *Proc. of the AAAI conference on artificial intelligence*, volume 35, pages 7865–7873, Virtual, 2021.

[30] E. Jeong, S. Oh, H. Kim, J. Park, M. Bennis, and S.-L. Kim. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *arXiv preprint arXiv:1811.11479*, 2018.

[31] E. Jeong, S. Oh, J. Park, H. Kim, M. Bennis, and S.-L. Kim. Hiding in the crowd: Federated data augmentation for on-device learning. *IEEE Intelligent Systems*, 36(5):80–87, 2020.

[32] W. Jeong, J. Yoon, E. Yang, and S. J. Hwang. Federated semi-supervised learning with inter-client consistency & disjoint learning. *arXiv preprint arXiv:2006.12097*, 2020.

[33] V. Kulkarni, M. Kulkarni, and A. Pant. Survey of personalization techniques for federated learning. In *WorldS4*, pages 794–797. IEEE, July 2020.

[34] D. Li and J. Wang. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*, 2019.

[35] T. Li, K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

[36] X. Li, W. Yang, Z. Zhang, K. Huang, and S. Wang. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019.

[37] Y.-F. Li and Z.-H. Zhou. Towards making unlabeled data never hurt. *IEEE transactions on pattern analysis and machine intelligence*, 37(1):175–188, 2014.

[38] P. Liang, T. Liu, L. Ziyin, N. Allen, R. Auerbach, D. Brent, R. Salakhutdinov, and L.-P. Morency. Think locally, act globally: Federated learning with local and global representations. *arXiv preprint arXiv:2001.01523*, 2020.

[39] X. Liang, Y. Lin, H. Fu, L. Zhu, and X. Li. Rscfed: Random sampling consensus federated semi-supervised learning. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10154–10163, New Orleans, Louisiana, June 2022.

[40] T. Lin, L. Kong, S. Stich, and M. Jaggi. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33:2351–2363, 2020.

[41] B. Liu, M. Ding, S. Shaham, W. Rahayu, F. Farokhi, and Z. Lin. When machine learning meets privacy: A survey and outlook. *ACM Computing Surveys (CSUR)*, 54(2):1–36, 2021.

[42] F. Liu, X. Wu, S. Ge, W. Fan, and Y. Zou. Federated learning for vision-and-language grounding problems. In *Proc. of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11572–11579, New York, NY, February 2020.

[43] Y. Liu, Y. Kang, C. Xing, T. Chen, and Q. Yang. A secure federated transfer learning framework. *IEEE Intelligent Systems*, 35(4):70–82, 2020.

[44] Z. Liu, Y. Shen, V. B. Lakshminarasimhan, P. Liang, A. Zadeh, and L.-P. Morency. Efficient low-rank multimodal fusion with modality-specific factors. *arXiv preprint arXiv:1806.00064*, 2018.

[45] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proc. of PMLR International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1273–1282, FL, USA, 2017.

[46] R. Mendes and J. P. Vilela. Privacy-preserving data mining: methods, metrics, and applications. *IEEE Access*, 5:10562–10582, 2017.

[47] L. Nagalapatti and R. Narayanam. Game of gradients: Mitigating irrelevant clients in federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9046–9054, Virtual, 2021.

[48] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng. Multimodal deep learning. In *Proc. of the 28th international conference on machine learning (ICML-11)*, pages 689–696, Bellevue, Washington, 2011.

[49] J. Park, S. Wang, A. Elgabli, S. Oh, E. Jeong, H. Cha, H. Kim, S.-L. Kim, and M. Bennis. Distilling on-device intelligence at the network edge. *arXiv preprint arXiv:1908.05895*, 2019.

[50] M. Phuong and C. Lampert. Towards understanding knowledge distillation. In *International Conference on Machine Learning*, pages 5142–5151, Long Beach, CA, 2019. PMLR.

[51] A. Ratner, S. Bach, H. Ehrenberg, and C. Ré. Snorkel: Fast training set generation for information extraction. In *Proc. of the 2017 ACM international conference on management of data*, pages 1683–1686, Hilton, Chicago, 2017.

[52] Y. Ruan and C. Joe-Wong. Fedsoft: Soft clustered federated learning with proximal local updating. In *Proc. of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8124–8131, Virtual, 2022.

[53] S. Sharma, C. Xing, Y. Liu, and Y. Kang. Secure and efficient federated transfer learning. In *IEEE International Conference on Big Data (Big Data)*, pages 2569–2576, Los Angeles, CA, 2019. IEEE.

[54] N. Sikder and A.-A. Nahid. Ku-har: An open dataset for heterogeneous human activity recognition. *Pattern Recognition Letters*, 146:46–54, 2021.

[55] K. Sindhu Meena and S. Suriya. A survey on supervised and unsupervised learning techniques. In *Pro. of international conference on artificial intelligence, smart grid and smart city applications: AISGSC 2019*, pages 627–644, Coimbatore, India, 2020.

[56] K. Sohn, W. Shang, and H. Lee. Improved multimodal deep learning with variation of information. *Advances in neural information processing systems*, 27, 2014.

[57] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu. A survey on deep transfer learning. In *International conference on artificial neural networks*, pages 270–279, Rhodes, Greece, 2018. Springer.

[58] I. Wagner and D. Eckhoff. Technical privacy metrics: a systematic survey. *ACM Computing Surveys (Csur)*, 51(3):1–38, 2018.

[59] H. Wang, L. Munoz-Gonzalez, D. Eklund, and S. Raza. Non-iid data re-balancing at IoT edge with peer-to-peer federated learning for anomaly detection. In *WiSec*, pages 153–163. ACM, 2021.

[60] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni. Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440*, 2020.

[61] K. Wang, R. Mathews, C. Kiddon, H. Eichner, F. Beaufays, and D. Ramage. Federated evaluation of on-device personalization. *arXiv preprint arXiv:1910.10252*, 2019.

[62] Y. Wang. Survey on deep multi-modal data analytics: collaboration, rivalry, and fusion. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 17(1s), 2021.

[63] Z. Wu, Y. Xiong, S. Yu, and D. Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proc. of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, Salt Lake City, UT, 2018.

[64] B. Xiong, X. Yang, F. Qi, and C. Xu. A unified framework for multi-modal federated learning. *Elsevier, Neurocomputing*, 480:110–118, 2022.

[65] Y. Xu, L. Wang, H. Xu, J. Liu, Z. Wang, and L. Huang. Enhancing federated learning with server-side unlabeled data by adaptive client and data selection. *IEEE Transactions on Mobile Computing*, 2023.

[66] H. Yang, H. He, W. Zhang, and X. Cao. Fedsteg: A federated transfer learning framework for secure image steganalysis. *IEEE Transactions on Network Science and Engineering*, 8(2):1084–1094, 2020.

[67] X. Yang, B. Xiong, Y. Huang, and C. Xu. Cross-modal federated human activity recognition via modality-agnostic and modality-specific representation learning. In *Proc. of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3063–3071, Virtual, February 2022.

[68] N. Yoshida, T. Nishio, M. Morikura, K. Yamamoto, and R. Yonetani. Hybrid-FL for wireless networks: Cooperative learning mechanism using non-iid data. In *ICC*, pages 1–7. IEEE, 2020.

[69] T. Yu, E. Bagdasaryan, and V. Shmatikov. Salvaging federated learning by local adaptation. *arXiv preprint arXiv:2002.04758*, 2020.

[70] Y. Zhao, P. Barnaghi, and H. Haddadi. Multimodal federated learning on iot data. In *Proc. of IEEE/ACM Seventh International Conference on Internet-of-Things Design and Implementation (IoTDI)*, pages 43–54, Milan, Italy, May 2022.

[71] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

[72] X. J. Zhu. Semi-supervised learning literature survey. 2005.

[73] Z. Zhu, J. Hong, and J. Zhou. Data-free knowledge distillation for heterogeneous federated learning. In *International Conference on Machine Learning*, pages 12878–12889, Virtual, 2021. PMLR.

[74] L. Zong, Q. Xie, J. Zhou, P. Wu, X. Zhang, and B. Xu. Fedcmr: Federated cross-modal retrieval. In *Proc. of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1672–1676, Virtual, July 2021.