Decentralized Multimodal AI Agents for Resource-Constrained Inferencing or Decisioning

MID SEM REPORT

Submitted in partial fulfillment of the requirements for the degree of

MASTERS OF TECHNOLOGY

by

Mathew Kadambatt

Under the Supervision of

Prathyusha V

and Mentorship of

Ashwini Chandrashekharaiah

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

2025

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI SEMESTER 24-25
AIMLCZG628T DISSERTATION

Mid Semester Report

BITS ID No. 2023AA05887 **Name of Student:** Mathew Kadambatt

Topic of Dissertation: Decentralized Multimodal AI Agents for Resource-Constrained Inferencing or Decisioning

Name of Supervisor: Prathyusha V
Designation of Supervisor: Senior Software Engineer
Qualification and Experience: Senior Technology Engineer experienced in Networking
Official E-mail ID of Supervisor: prathyusha.vadde@walmart.com

Name of Mentor: Ashwini Chandrashekharaiah
Designation of Mentor: Senior Data Science Manager
Official E-mail ID of Mentor: ashwini.chandrashekharaiah@walmart.com

Signature of Student

Signature of Supervisor

Abstract

The emergence of intelligent edge ecosystems, comprising distributed, heterogeneous IoT devices, necessitates the design of decentralized AI agents that can collaboratively learn, infer, and make decisions using diverse data modalities under stringent resource constraints. This paper proposes a comprehensive framework for Decentralized Multimodal AI Agents, integrating advances in Federated Learning (FL), Multimodal Machine Learning (MML), and Multi-Agent Deep Reinforcement Learning (MADRL) to enable scalable, privacy-preserving intelligence across resource-constrained environments.

The paper establishes its approach in Multimodal Federated Learning on IoT or similar diverse simulated data, which demonstrates effective cross-device learning with heterogeneous and non-IID multimodal inputs. To enhance adaptability, it incorporates strategies from Multimodal Online Federated Learning with Modality Missing in IoT (1), allowing agents to perform robust inferencing and decisioning even when certain modalities are absent or dynamically changing. The decentralized coordination and policy optimization aspects are informed by Federated Multi-Agent Deep Reinforcement Learning for Intelligent IoT Wireless Communications, empowering agents to operate autonomously while maintaining collaborative objectives within dynamic wireless networks.

To meet the challenges of communication bottlenecks and edge deployment, it adopts the asynchronous and temporally weighted aggregation strategies from Communication-Efficient Federated Deep Learning, which reduce synchronization overhead and improve convergence in non- ideal network conditions. To further support resource-constrained execution, it incorporates principles from Efficient Deep Learning: A Survey, employing model compression techniques such as pruning, quantization, and knowledge distillation to ensure real-time performance on edge devices.

The work is further guided by A Survey of Multi-Agent Deep Reinforcement Learning with Communication, which offers insights into scalable agent cooperation with minimal inter-agent bandwidth, and Multimodal Machine Learning: A Survey and Taxonomy, which informs the design of modality fusion, alignment, and translation mechanisms.

Collectively, these foundations support a new class of Decentralized Multimodal AI Agents capable of performing adaptive, efficient, and resilient inferencing or decisioning in real-world IoT deployments. It's concluded by highlighting open challenges, including modality-aware aggregation, asynchronous agent collaboration, and efficient decision reasoning under missing data, and outline future research directions for enabling intelligent, decentralized edge-AI systems.

Key Words: Federated Learning; Decentralized AI; Multimodal Machine Learning; Multi-Agent Reinforcement Learning; Edge Intelligence; Resource-Constrained Inference; Asynchronous Model Aggregation; Modality Missing; IoT Systems; Efficient Deep Learning.

List of Symbols

1. *DFL* : Decentralized Federated Learning
2. *IoT* : Internet of Things
3. *MSE* : Mean Squared Error
4. *Sync* : Synchronous model synchronization
5. *MAE* : Mean Absolute Error
6. *MSE* : Mean Squared Error
7. *k8s* : Kubernetes
8. *GKE* : Google Kubernetes Engine
9. *CI/CD* : Continuous Integration and Continuous Deployment

List of Tables

List of Figures

Contents

Chapter 1

Introduction

The increasing ubiquity of intelligent edge ecosystems, powered by distributed and heterogeneous IoT devices, is driving a shift in how AI systems are designed and deployed. These systems must not only process data locally but also collaboratively learn and infer from diverse data modalities, all while operating under stringent computational, memory, and bandwidth constraints. Traditional centralized approaches to AI model training and deployment are insufficient in such dynamic and resource-limited settings.

This work proposes a novel framework for building Decentralized Multimodal AI Agents, leveraging principles from Federated Learning (FL), Multimodal Machine Learning (MML), and Multi-Agent Deep Reinforcement Learning (MADRL). These agents are designed to be capable of scalable, privacy-preserving, and adaptive learning across constrained environments where modality availability can change over time and network synchronization cannot be guaranteed.

Key to this approach is the integration of the following components:

Multimodal Federated Learning (MMFL): Enabling collaborative learning across non-IID, multimodal data distributed among IoT devices.

- Online and Modality-Missing Federated Learning: Allowing robust inferencing when some data modalities are unavailable. (2)
- Federated MADRL: Facilitating decentralized coordination and policy optimization for autonomous agent behavior. (3)
- Communication-efficient and asynchronous training: Reducing network bottlenecks through temporally weighted aggregation. (2)
- Efficient Deep Learning (EDL): Ensuring lightweight deployment using pruning, quantization, and knowledge distillation. (1)

This chapter outlines the research objectives derived from the abstract and details the milestones achieved in the first phase of the project.

1.1 Problem Statement

In distributed IoT environments, where nodes vary in data distribution, compute power, and availability, there is a pressing need for resilient, intelligent agents that can:

- Collaboratively learn without centralized coordination.

- Operate efficiently even with incomplete or missing modalities.
- Communicate minimally yet effectively with peer agents.
- Make decisions autonomously while optimizing collective performance.

Designing such a system involves addressing multiple research challenges related to multimodality, decentralization, synchronization efficiency, and lightweight inference.

1.2 Objectives

Based on the abstract, the major objectives of this work are:

1. Design and implement a framework for Multimodal Federated Learning across decentralized IoT pods with non-IID and heterogeneous inputs
2. Support modality-missing learning, enabling AI agents to train and infer when certain input modalities are absent or changing dynamically.
3. Employ federated multi-agent reinforcement learning principles for decentralized coordination and policy optimization under uncertain wireless environments.
4. Integrate communication-efficient FL strategies, such as asynchronous updates and temporally weighted aggregation, to reduce synchronization overhead.
5. Implement model compression techniques (e.g., pruning, quantization, knowledge distillation) to enable inference on resource-constrained IoT devices.
6. Incorporate monitoring and observability mechanisms using Prometheus and Grafana to collect, analyze, and visualize real-time performance metrics.
7. Evaluate and compare sync vs no-sync training performance in terms of convergence, stability, and generalization (MSE, loss, accuracy).

1.3 Observations Summarized

Initial experiments reveal several key insights:

1. Sync strategies reduce divergence among decentralized nodes and improve generalization as measured by eval MSE and final training loss.
2. Asynchronous nodes tend to overfit or drift due to the lack of parameter homogenization, though they offer higher independence and less communication overhead.
3. The monitoring stack (Prometheus + Grafana) enabled real-time visualization of pod metrics, significantly helping in debugging and experimentation

1.4 Unitil Mid-Semester Progress

Id	Task	Status	Summary
1	Research, Objectives/Scope of work analysis and Abstract submission	Done	Abstract and Scope submitted and evaluated
2	System Designing	Done	Designing the model to incorporate FL and Multimodal logic
3	Multimodal development and Redesign	Pending	Develop the model and later compress it based on G. Menghani's "Efficient deep learning" (4)
4	Infra setup and Deployment	Done	GKE(k8s) infra setup, MLOps configuring & deployment
5	Simulation using raw data	Done	Simulate the scenario.
6	Improvise the model networking and communication	Pending	Decentralized federated learning improvisations from network point of view
7	Implement real time scenario	Pending	Simulate IoT device image & communication in GKE and analyze or use real world IoT device.

Table 1.1: Objectives and Status

Chapter 2

System Architecture & Methodology

2.1 Architecture Overview

The system consists of three decentralized IoT nodes, each running in a Kubernetes pod.

- Set up the environment
  - Setup GKE cluster - auto pilot cluster and Container Registry
  - Setup CI/CD building the image for deploying the image aon GKE pods
  - Integrate github actions using service account secrets
  - Setup monitoring pipeline in Prometheus & Grafana
- Analyse the data and identify the subject wise split on pods
- Build a simple model and stream the data to pods and perform sync operations
- Periodically syncs weights with peers and exposes metrics for Prometheus.
- Build the Dashboards and Analyse the Loss, MSE and Accuracy

2.2 Components

Below describes the list of major components being used for this project atleast until the mid-sem

Component	Description
GKE Cluster	GCP project and GKE & Container registry APIs enabled
Docker	Lightweight image with base image as python
Model	A simple model for loading data, training, evaluating
Python Fast API	API interface
Communication	Handles inter-node sync over TCP using gossip protocol exposed over python Fast API
Stateful sets	k8s statefulsets to simulate realtime IoT environments
Prometheus	Scrape the data from metrics endpoint
Grafana	Dashbaords to Project the metrics

Table 2.1: Project Components and Their Descriptions

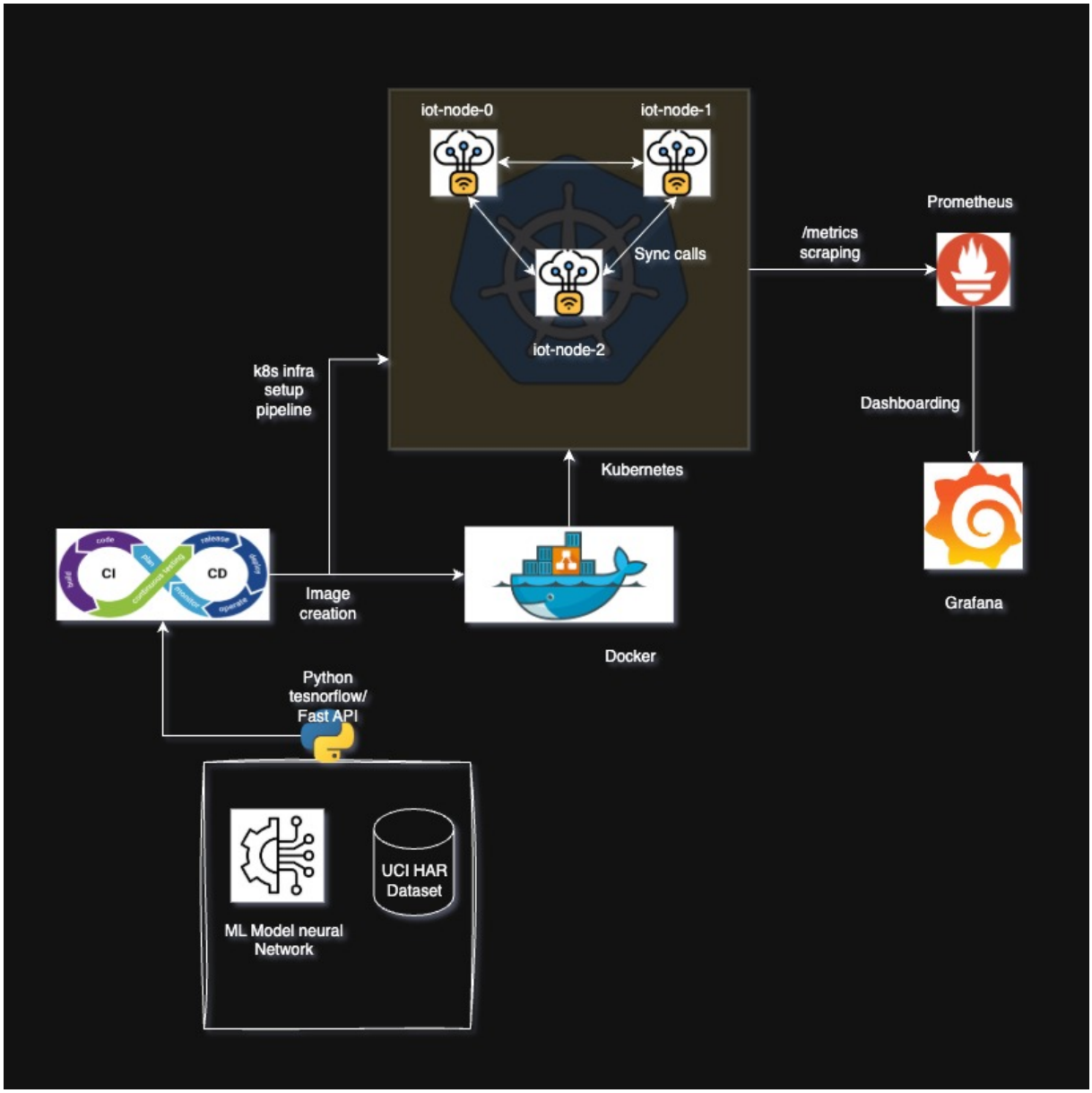

Figure 2.1: Architecture diagram

2.3 Datasets

Dataset: UCI HAR

- Activities: Walking, Standing, Sitting, etc.
- Subjects: 30 individuals, divided across 3 pods.
- Features: 561 features from 3-axial accelerometer & gyroscope.
- Label Classes: 6
- Each pod sees different subjects → non-IID setup.

2.4 Model

Below Model configurations and parameters are used:

- Model: Fully connected dense neural network in NumPy.
- Loss: Categorical cross-entropy.
- Optimizer: SGD.
- Batch Size: 32.
- Evaluation: MSE and accuracy on held-out validation.

2.5 Synchronization Strategies

No-Sync: Each pod trains independently; risk of overfitting or drift Sync: Periodic averaging of model weights; reduces divergence

2.6 Monitoring

Metrics exposed at /metrics endpoint of each pod:

- training_loss
- training_accuracy
- eval_mse

Prometheus scrapes every 15s. Grafana dashboards provide node-wise comparison.

Chapter 3

Infrastructure Setup, Deployment & Evaluation

3.1 Infrastructure setup & Deployment

The infra is setup on kubernetes platform of GCP, also known as GKE under a dedicated project. An auto-pilot cluster with 2 nodes is used here. The infrastructure contains the following parts:

- Statefulset of IoT pods
- Prometheus and Grafana deployments for monitoring.

Along with that a Container Registry is also maintained to push/pull the images after building and deploy them to kubernetes.

Statefulsets of IoT pods are maintained so that the pod names would not be affected by deployments, and also a volume mount can be used to store the model if needed. Currently the pods are scaled to 3 with all pods communicating with each other since its within the same namespace. The Prometheus and Grafana are single-pod deployments along with a ClusterIp service to communicate with the pods. Grafana also uses a volume mount to store dashboard configs to be reused. Port forwarding is performed to connect the Grafana from local.

A GCP service account with required IAM permission are created to connect and perform CI/CD operations. The code is deployed in a github repository where the project ID and the service account's secret key are added as github secrets. Using Github actions, along with the GCP plugins, the code is dockerized and pushed to GCP container registry. Then the Kubernetes deployment is triggered where the resource file with necessary configuration is applied, using the kubernetes plugins integrated in the github flow. In this manner, the complete CI/CD pipeline is managed from github actions with a rule to be triggered on a push/pull-request merge to the main branch of the github repository.

3.2 Evaluation

The Docker image is a light-weight Python image which uses a Fast API uvicorn server for running as a service exposing REST APIs. Following endpoints are configured in the service:

- /sync: sync a particular node
- /sync-all: sync all the peer nodes
- /train: to trigger training manually
- /eval: to trigger evaluation manually
- /metrics: exposes Prometheus metrics to be scraped by Prometheus pods deployed

The downloaded and extracted UCI HAR Data is embedded within the image, because the download of data during startup takes some time and results in pod startup probe failures. The train, eval and sync-all events are configured to run as scheduled jobs every 15 secs.

The data is divided into subjects, filtering the top 3 subjects with the most number of data points and split among pods based on the pod index for non-IID data distribution. The data pipeline is also configured in the code for each pod. The data for the respective subjects are filtered in each pod. The data is split into training (80%) and test (20%) data. The data is streamed for training at a batch of 10 to the models. The trained models are then evaluated with the test data. Once the training data stream reaches the end, the data is restarted from beginning and the process continues within each pod.

In order to evaluate and compare the metrics with & without sync, the train & eval is performed on two models with the same configuration at beginning. One model gets the streamed data without a sync of peers weights. While the other one gets trained & evaluated after the pod makes the sync-all call to all its peers, making sure that the weights are updated first and then training/evaluation is performed. The model metrics are segregated between the synced model & normal model model using a label named stage in the Prometheus metrics, passed after performing the train/eval operations.

The captured metrics are used to build dashboards on Grafana and compare between the sync & no-sync models, as shared in the next chapter.

Chapter 4

Results and Evaluation

4.1 Training Loss Analysis

As shown in the Grafana plot:

- No-Sync Loss: Fluctuations and inconsistency between pods.
- Sync Loss: Rapid convergence.

Conclusion: Sync leads to better convergence and reduced variation.

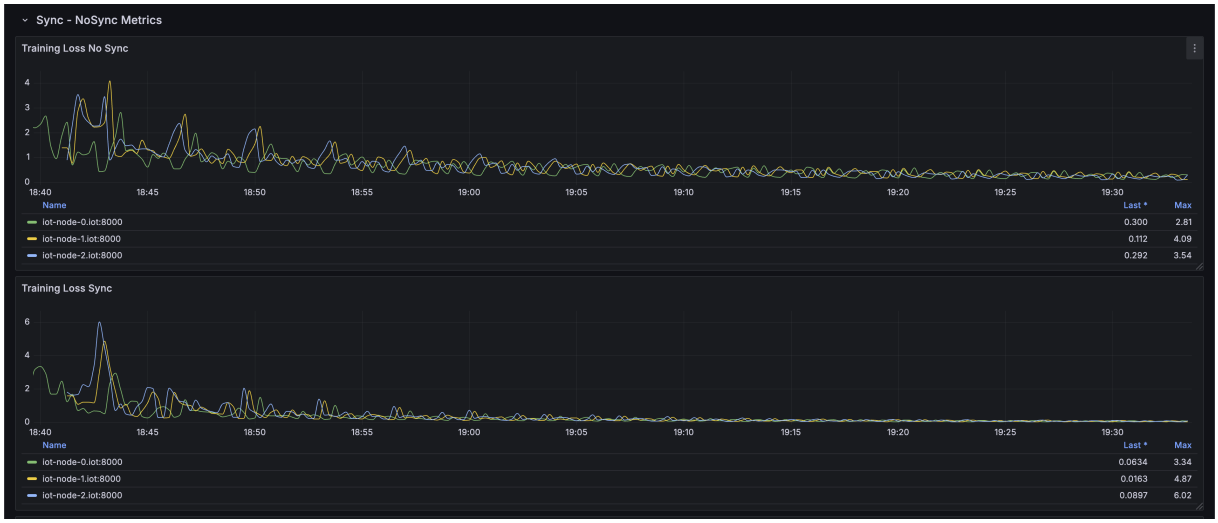

Figure 4.1: Training Loss Analysis

4.2 Evaluation MSE

More stability across pods with sync but takes time to reduce across pods compared to without sync

- Sync: Stable and lower MSE
- No-Sync: Varied MSE across pods, occasional divergence.

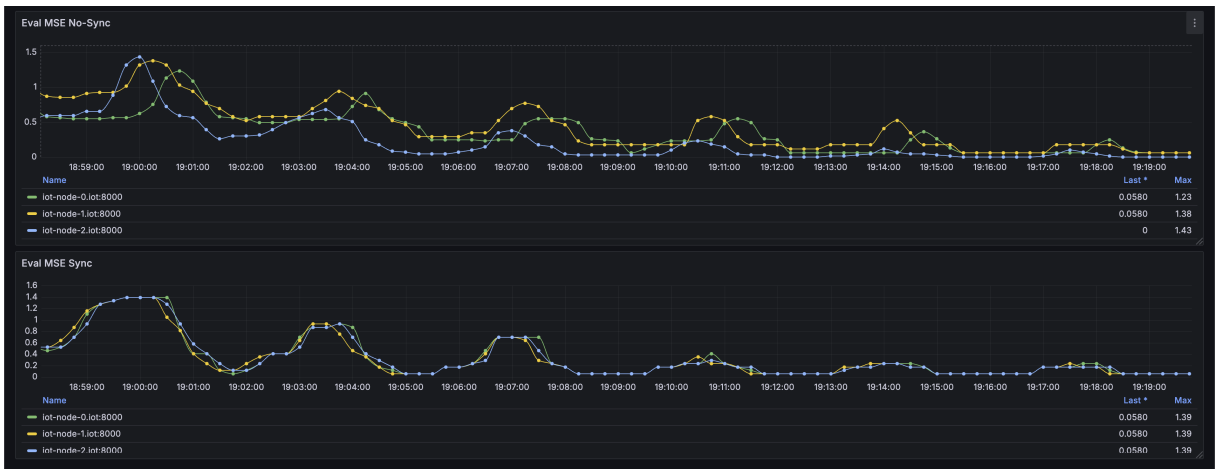

Figure 4.2: Evaluation MSE Analysis

4.3 Accuracy Trends

All nodes achieve good training accuracy by end of training.

- No-Sync: High fluctuation initially.
- Sync: Faster and smoother stabilization.

Figure 4.3: Accuracy Trends Analysis

Chapter 5

Future Work Directions

As stated in abstract timelines, post Mid semester (Week 10) will be focussing on the next set of action items. Since Phase III regarding mult-modal redesign as per simulation is pending that would also be taken up with that bandwidth already being utilized for raw Data Simulation(Phase V). Future works in short are below:

1. **Missing Modality Simulation:**
    - Drop or mask input modalities during training.
    - Evaluate performance drop and recovery mechanisms by distillation.
2. **Scale Beyond 3 Pods:**
    - Test gossip protocols at scale (10–20 nodes).
    - Validate using other subjects or variations in data
3. **Dynamic Sync Trigger:**
    - Replace periodic sync with loss-divergence-based sync-on-demand to simulate real time scenario.
    - Analyze the system(pod) performance with sync and no-sync.
4. **Network Communication Improvisation:**
    - Adding time weightages for the sync events processing as cited in (5)
    - Adding distance weightage by incorporating GPS distancing logic
    - Segregating parameters as deep and shallow to reduce the communication and updates as cited in (5)
5. **Multi-Agent RL Integration(Optional):**
    - Use policy networks and model rules for decisioning. (6)

Bibliography

[1] X. Chen, Y. Hu, J. He, and C. Xu, "Multimodal federated learning on iot data," *arXiv preprint arXiv:2109.04833*, 2021.

[2] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, "Multimodal machine learning: A survey and taxonomy," *arXiv preprint arXiv:1705.09406*, 2017.

[3] X. Li, Y. Liang, Y. Xu, and W. Guo, "Federated multiagent deep reinforcement learning for intelligent iot wireless communications: Overview and challenges," *IEEE Communications Magazine*, 2024.

[4] G. Menghani, "Efficient deep learning: A survey on making deep learning models smaller, faster, and better," *arXiv preprint arXiv:2106.08962*, 2021.

[5] J. Chen, Q. Sun, and Y. Jin, "Communication-efficient federated deep learning with asynchronous model update and temporally weighted aggregation," *arXiv preprint arXiv:1903.07424*, 2019.

[6] C. Zhu, M. Dastani, and S. Wang, "A survey of multi-agent deep reinforcement learning with communication," *arXiv preprint arXiv:2203.08975*, 2022.

[7] H. Wang, X. Liu, X. Zhong, L. Chen, F. Liu, and W. Zhang, "Multimodal online federated learning with modality missing in internet of things," *arXiv preprint arXiv:2505.16138*, 2025.