2024-10-09

Federated Learning Model Aggregation in Heterogeneous Aerial and Space Networks

Dong, Fan

UNIVERSITY OF CALGARY

Federated Learning Model Aggregation in Heterogeneous Aerial and Space Networks

by

Fan Dong

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN ELECTRICAL ENGINEERING

CALGARY, ALBERTA

OCTOBER, 2024

Abstract

Federated learning offers a promising solution for overcoming the challenges of networking and data privacy in aerial and space networks by harnessing large-scale private edge data and computing resources from drones, balloons, and satellites. Although existing research has extensively explored optimizing the learning process, improving computing efficiency, and reducing communication overhead, statistical heterogeneity remains a substantial challenge for federated learning optimization. While state-of-the-art algorithms have made progress, they often overlook diversity heterogeneity and fail to significantly improve performance in high-degree label heterogeneity conditions. In this thesis, statistical heterogeneity is further dissected into two categories: diversity heterogeneity and label heterogeneity, allowing for a more nuanced analysis. It also emphasizes the importance of addressing both diversity heterogeneity and high-degree label heterogeneity in aerial and space network applications. A theoretical analysis is provided to guide optimization in these two challenging scenarios. To tackle diversity heterogeneity, the WeiAvgCS algorithm is introduced to accelerate federated learning convergence. This algorithm employs weighted aggregation and client selection based on an estimated diversity measure, termed *projection*, enabling WeiAvgCS to outperform other benchmarks without compromising privacy. For high-degree label heterogeneity, the FedBalance algorithm is proposed, utilizing the label distribution information of each client. A novel metric, termed *relative scarcity*, is introduced to determine the aggregation weights assigned to clients. During the training process, fully homomorphic encryption is employed to protect

clients' label distributions. Additionally, two communication protocols are designed to facilitate training across different scenarios. Extensive experiments were conducted, demonstrating the effectiveness of WeiAvgCS and FedBalance in addressing the research gaps in diversity heterogeneity and high-degree label heterogeneity.

Preface

This thesis is the original and independent work of the author, Fan Dong. Portions of this thesis have been published in IEEE Xplore and accepted by the IEEE World Forum on the Internet of Things. The remaining sections have been submitted to the Journal of Network and Computer Applications.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my supervisor, Dr. Steve Drew, and co-supervisor, Dr. Henry Leung, for their invaluable guidance, support, and encouragement throughout my research and the writing of this thesis. Their expertise and insightful feedback have been instrumental in shaping the direction and outcome of this work.

I extend special thanks to Dr. Jiayu Zhou and my colleagues in the ESE Department and DENOS Lab for their insightful suggestions, collaborative spirit, stimulating discussions, and unwavering support during this journey.

I also wish to thank the oral examination committee for their valuable suggestions and questions, which helped me refine this thesis further.

Lastly, I acknowledge the funding and resources provided by Dr. Steve Drew, Dr. Henry Leung, and the Department of Electrical and Software Engineering at the University of Calgary. Their support has been crucial, and this research would not have been possible without their contributions.

Thank you all for making this journey a remarkable and fulfilling experience.

Contents

Abstract ii

Preface iv

Acknowledgements v

Contents vi

List of Figures ix

List of Tables xiv

List of Symbols, Abbreviations, & Nomenclature xv

1 Introduction 1

1.1 Background on Aerial and Space Networks and Federated Learning 1

1.2 Heterogeneity Issues in Federated Learning 3

1.2.1 Common Simulation Schemes for Statistical Heterogeneity: Dirichlet and Shard-Based Distributions 3

1.2.2 Two Types of Statistical Heterogeneity: Label and Diversity Heterogeneity 5

1.2.3 Statistical Heterogeneity and Class Imbalance 6

1.3 Challenges of Applying Federated Learning in Aerial and Space Networks 7
1.3.1 How Diversity Varies Among Clients 7
1.3.2 How Battery Life Constraint Exacerbates the Label Heterogeneity 8
1.4 Research Overview and Contributions 10

2 Related Work **14**
2.1 Current Research 15
2.1.1 Schemes to Mitigate the Heterogeneity Effect 15
2.1.2 Class Imbalance Issue 16
2.1.3 Various Weighted Averaging Approaches 16
2.1.4 Client Selection Techniques 17
2.2 Research Gaps and Contributions 17

3 Theoretical Analysis **19**
3.1 Weighted Averaging and Client Selection Based on Diversity (WeiAvgCS) 21
3.2 Weighted Averaging Based on *Relative Scarcity* (FedBalance) 22

4 Methodologies **24**
4.1 The WeiAvgCS Algorithm 24
4.2 The FedBalance Algorithm 28
4.2.1 Core of FedBalance: Weighted Averaging 28
4.2.2 Computing ω with Fully Homomorphic Encryption 30
4.2.3 Further Improvement: FedBalanceFilter 32
4.2.4 Why FedBalance Should Work 34

5 Experiment Results **35**

5.1 The WeiAvgCS Algorithm . . . 36
5.1.1 WeiAvgCS with Different λ . . . 36
5.1.2 Comparing WeiAvgCS with Different Benchmarks. . . . 37
5.1.3 Identifying Limitations: When WeiAvgCS Falls Short . . . 39
5.2 The FedBalance Algorithm . . . 39
5.2.1 Under Globally Balanced Distributions . . . 40
5.2.2 Under Globally Imbalanced Distributions . . . 42
5.2.3 Comparison with the Ratio Loss algorithm . . . 44

6 Conclusion **47**

A Detailed Analysis of the Impacts of Battery Constraint to Label Heterogeneity **54**

A.1 Selecting Fewer Clients in Each Global Epoch . . . 55
A.2 Selecting Clients From a Smaller Pool . . . 55
A.3 FedAvg Performance Under Different Heterogeneity Degrees . . . 57
A.4 Performance of State-of-the-Art FL Algorithms Under Varying Degrees of Heterogeneity . 59

B More Experimental Results **61**

B.1 Experimental Results on the FashionMNIST Dataset . . . 61
B.2 Experimental Results on the CINIC10 Dataset . . . 64

List of Figures

1.1 Federated Learning Process: (a) Broadcasting. The central server selects a subset of M clients from the total N clients ($M \leq N$). It then broadcasts the current global model to these selected clients. (b) Uploading. Clients perform local training using their own data and the received global model. After training, they upload their updated model parameters to the central server for aggregation. 2

1.2 Common heterogeneity simulation schemes. (a) Dirichlet distribution, where smaller α values result in more heterogeneous distributions. (b) Shard-based distribution, where fewer shards per client lead to more heterogeneous distributions. 4

1.3 Two types of statistical heterogeneity. (a) Label heterogeneity, where clients have different label distributions but similar degrees of diversity. (b) Diversity heterogeneity, where clients have both different label distributions and varying degrees of diversity. 5

1.4 Two types of imbalance. (a) shows a scenario in which local devices have imbalanced data while the global data is balanced. (b) shows a scenario in which imbalance exists both locally and globally. 6

1.5 Different devices would possess varying degrees of data diversity concerning their locations and altitudes. 7

1.6 Grouped dataset imbalance degree with different number of clients selected. 9

1.7 Imbalance degree under different pool sizes. 9

1.8 The average distribution and individual distributions of clients A and B. Client A's distribution is more closely aligned with the average distribution, indicating that client B's distribution is scarcer. 11

4.1 The *projection* ($\boldsymbol{p}_i^t$) of client i's local update ($\mathbf{w}_i^t - \mathbf{w}^t$) at the t_{th} global epoch onto the global update ($\mathbf{w}^{t+1} - \mathbf{w}^t$). 27

4.2 Correlation between *projection* and actual diversity in WeiAvgCS with different λ values on FashionMNIST and CIFAR10. 27

4.3 Communication protocols: (a) with a third party, and (b) without a third party. 30

4.4 Time consumption for FHE. The FHE process can be completed within seconds on an Apple M1 Pro chip, significantly faster than local training. Additionally, FHE and local training can be conducted in parallel, resulting in no additional time cost. 31

4.5 Grouped data distribution under simple averaging (FedAvg) and weighted averaging (FedBalance). A more uniform distribution indicates a tighter upper bound for approximation error, as shown in Part 2 of Inequality (3.5). 34

5.1 WeiAvgCS (λ) using (a) actual diversity and (b) *projection*. Both experiments demonstrate the superiority of WeiAvgCS over FedAvg across a wide range of λ values. 37

5.2 Test accuracy of FedAvg and WeiAvgCS with three benchmarks: (a) FedProx, (b) MOON, (c) Scaffold, and their respective combinations on FashionMNIST ($\gamma = 1$). 37

5.3 Test accuracy of FedAvg and WeiAvgCS with three benchmarks: (a) FedProx, (b) MOON, (c) Scaffold, and their respective combinations on CIFAR10 (γ=1). 38

5.4 Correlation between *projection* and actual diversity across different local training epochs (1, 2, 5, and 10). Stronger correlations are observed with an increasing number of local epochs. 39

5.5 Convergence trends with different local training epochs. WeiAvgCS shows no performance improvement when the number of local epochs is too low. 40

5.6 Performance of algorithms on the CIFAR10 dataset under Dirichlet distribution with varying degrees of heterogeneity. Scaffold performs effectively under low heterogeneity ($\alpha \geq 0.1$) but fails to outperform FedAvg in highly heterogeneous settings. FedBalance and FedBalanceFilter bridge the gap in training highly heterogeneous FL scenarios. 41

5.7 Experiments on the CIFAR10 dataset with the same degree of heterogeneity but varying numbers of majority classes. Subfigure (a) displays results on a balanced global dataset for comparison, where $IR = 1$. IR represents the imbalance ratio between the number of samples in the majority class and the minority class, with a higher IR indicating a more imbalanced global dataset. MC denotes the number of majority classes, while the number of minority classes in the 10-class classification is $10 - MC$. 43

5.8 Experiments on the CIFAR10 dataset with the same degree of heterogeneity but varying numbers of majority classes. Subfigure (a) displays results on a balanced global dataset for comparison, where $IR = 1$. IR represents the imbalance ratio between the number of samples in the majority class and the minority class, with a higher IR indicating a more imbalanced global dataset. MC denotes the number of majority classes, while the number of minority classes in the 10-class classification is $10 - MC$. 43

5.9 Experimental results of the Ratio Loss algorithm, FedAvg, and FedBalance algorithms under varying learning rates and degrees of heterogeneity. 45

5.10 Label distribution of the EMNIST dataset used in the Ratio Loss algorithm. This distribution exhibits a low degree of heterogeneity based on prior experimental results shown in Fig. A.4. 46

A.1 Clients are queued according to their respective battery percentages. Only clients with sufficient battery levels are selected in each global epoch. 56

A.2 Imbalance degree under different window sizes with different pool sizes. 56

A.3 Imbalance degree under different step sizes with different window sizes. 57

A.4 Average distribution on each client with different settings. 58

A.5 Test accuracy lines of the FedAvg algorithm under different degrees of heterogeneity. . . . 59

A.6 Performance of algorithms on the CIFAR10 dataset under Dirichlet distribution with varying degrees of heterogeneity. Scaffold performs effectively under low heterogeneity ($\alpha \geq 0.1$) but fails to outperform FedAvg in highly heterogeneous settings. 60

B.1 Performance of algorithms on the FashionMNIST dataset under Dirichlet distribution with varying degrees of heterogeneity. Scaffold performs effectively under low heterogeneity ($\alpha \geq 0.1$) but fails to outperform FedAvg in highly heterogeneous settings. FedBalance and FedBalanceFilter outperform all other benchmarks in highly heterogeneous FL scenarios. 62

B.2 Experiments on the FashionMNIST dataset with the same degree of heterogeneity but varying numbers of majority classes. Subfigure (a) displays results on a balanced global dataset for comparison, where $IR = 1$. IR represents the imbalance ratio between the number of samples in the majority class and the minority class, with a higher IR indicating a more imbalanced global dataset. MC denotes the number of majority classes, while the number of minority classes in the 10-class classification is $10 - MC$. 62

B.3 Experiments on the FashionMNIST dataset with the same degree of heterogeneity but varying numbers of majority classes. Subfigure (a) displays results on a balanced global dataset for comparison, where $IR = 1$. IR represents the imbalance ratio between the number of samples in the majority class and the minority class, with a higher IR indicating a more imbalanced global dataset. MC denotes the number of majority classes, while the number of minority classes in the 10-class classification is $10 - MC$. 63

B.4 Performance of algorithms on the CINIC10 dataset under Dirichlet distribution with varying degrees of heterogeneity. Scaffold performs effectively under low heterogeneity ($\alpha \geq 0.1$) but fails to outperform FedAvg in highly heterogeneous settings. FedBalance and FedBalanceFilter outperform all other benchmarks in highly heterogeneous FL scenarios. . 64

B.5 Experiments on the CINIC10 dataset with the same degree of heterogeneity but varying numbers of majority classes. Subfigure (a) displays results on a balanced global dataset for comparison, where $IR = 1$. IR represents the imbalance ratio between the number of samples in the majority class and the minority class, with a higher IR indicating a more imbalanced global dataset. MC denotes the number of majority classes, while the number of minority classes in the 10-class classification is $10 - MC$. 65

B.6 Experiments on the CINIC10 dataset with the same degree of heterogeneity but varying numbers of majority classes. Subfigure (a) displays results on a balanced global dataset for comparison, where $IR = 1$. IR represents the imbalance ratio between the number of samples in the majority class and the minority class, with a higher IR indicating a more imbalanced global dataset. MC denotes the number of majority classes, while the number of minority classes in the 10-class classification is $10 - MC$. 65

List of Tables

4.1 Notations Used in the FedBalance Algorithm 28

5.1 Numerical experimental results for FashionMNIST and CIFAR10 Across Different Degrees of Diversity Heterogeneity 38

5.2 Numerical experimental results for the CIFAR10 dataset under Dirichlet distribution with varying α values. 41

5.3 Numerical experimental results for the CIFAR10 dataset with an imbalance ratio (IR) set to 5. 43

5.4 Numerical experimental results for the CIFAR10 dataset with an imbalance ratio (IR) set to 10. 43

A.1 Numerical experimental results for the CIFAR10 dataset under Dirichlet distribution with varying α values. 60

B.1 Numerical experimental results for the FashionMNIST dataset under Dirichlet distribution with varying α values. 61

B.2 Numerical experimental results for the FashionMNIST dataset with an imbalance ratio (IR) set to 5. 62

B.3 Numerical experimental results for the FashionMNIST dataset with an imbalance ratio (IR) set to 10. 63

B.4 Numerical experimental results for the CINIC10 dataset under Dirichlet distribution with varying α values. 64

B.5 Numerical experimental results for the CINIC10 dataset with an imbalance ratio (IR) set to 5. 64

B.6 Numerical experimental results for the CINIC10 dataset with an imbalance ratio (IR) set to 10. 65

List of Symbols, Abbreviations, & Nomenclature

ASNs	Aerial and space networks
FL	Federated learning
FHE	Fully homomorphic encryption
B	Number of classes. For FashionMNIST and CIFAR10, $B = 10$. For CIFAR100, $B = 100$
MC	Number of the majority classes, $0 \leq MC \leq B$
IR	Imbalance ratio, reflecting the degree of imbalance in the global dataset
$\mathcal{D}$	Label distribution of the grouped dataset, $\mathcal{D} = [\rho_1, \rho_2, \cdots, \rho_B]$
Δ	Imbalance degree of the grouped dataset, $\Delta = max(\mathcal{D}) - min(\mathcal{D})$
α	Hyperparameter controlling the heterogeneity degree in the Dirichlet distribution
$\mathcal{X}$	Dataset
γ	Hyperparameter controlling the diversity heterogeneity degree in the WeiAvgCS algorithm
d_i	Diversity degree of client i's label distribution
λ	Hyperparameter controlling the weight distribution
T	Total number of global epochs
t	Index of current global epoch

lr	Learning rate
K	Public key used for encryption in fully homomorphic encryption
K^{-1}	Private key used for decryption in fully homomorphic encryption
F	Global objective function
∇F	Gradient of of the global objective function F
$\mathbf{w}$	Global model
$\mathbf{w}^1$	Initial global model
$\mathbf{w}^t$	Global model at the t_{th} global epoch
N	Total number of clients
C	Client pool consisting of all N clients
M	Number of participating clients in each global epoch
A	Number of excess clients selected for the FedBalanceFilter algorithm
$\boldsymbol{c}^{t,M}$	Set of M selected clients in epoch t
R	Maximum retaining epochs for each client
r	Number of clients retained each epoch
$\boldsymbol{p}_i^t$	The *projection* of client i's local update at the t_{th} epoch onto the global update
Use_Actual_Diversity	Indicator variable of using actual diversity or estimated diversity
$\mathcal{D}_i$	Label distribution on client i. $\mathcal{D}_i = [\rho_{i,1}, \rho_{i,2}, \cdots, \rho_{i,j}, \cdots, \rho_{i,B}]$
$\rho_{i,j}$	Portion of class j on client i
$\bar{\mathcal{D}}$	Average distribution of M clients. $\bar{\mathcal{D}} = [\frac{1}{M}\sum_{i=1}^{M}\rho_{i,1}, \cdots, \frac{1}{M}\sum_{i=1}^{M}\rho_{i,B}]$
σ_i^2	Variance of the label distribution for a randomly selected client i
s_i	Relative scarcity of client i among M participating clients. $s_i = \frac{1}{\langle \mathcal{D}_i, \bar{\mathcal{D}} \rangle}$
ω_i	Weight assigned to client i

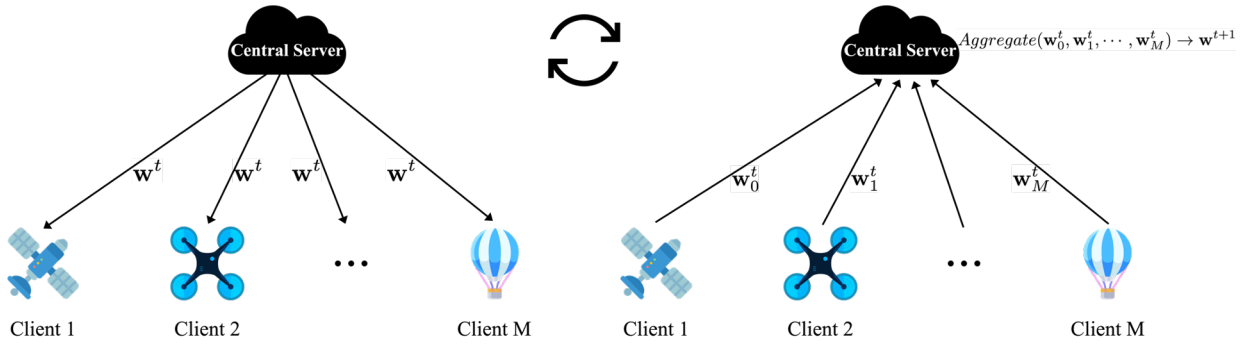
Chapter 1

Introduction

1.1 Background on Aerial and Space Networks and Federated Learning

Aerial and Space Networks (ASNs) [21] are new types of networks providing networked integration of aerial and space assets. These networks leverage drones, balloons, and satellites to gather and process a wide range of sensor data, characterized by diverse distributions. Furthermore, these devices in ASNs can provide edge computing capabilities [40] to perform complex machine learning tasks for localized data processing. The diverse nature of devices, constrained bandwidths, and varying ownership models within ASNs hinder centralized data processing and machine learning for predictive model training. Limited bandwidth makes transmitting large datasets impractical, while data privacy regulations and concerns often prohibit sharing sensitive information across ASNs. These challenges collectively impede the ability to train predictive models centrally in ASNs.

Federated learning (FL) [23] has emerged as a promising solution where distributed clients train their models locally and send only the model parameters to the central server. In a typical FL setting with a star topology [30], as illustrated in Figure 1.1, there are a central server and multiple clients. The central server manages model aggregation and client selection, while clients perform local training using their



(a) Model broadcasting

(b) Model uploading

Figure 1.1: Federated Learning Process: (a) Broadcasting. The central server selects a subset of M clients from the total N clients ($M \leq N$). It then broadcasts the current global model to these selected clients. (b) Uploading. Clients perform local training using their own data and the received global model. After training, they upload their updated model parameters to the central server for aggregation.

own data. The central server is typically a powerful computing center that has connection with all clients, while clients can be diverse devices, such as drones, balloons, and satellites in ASNs.

A typical global epoch in FL training involves the following steps:

1. Client Selection: The central server selects a subset of M clients from the total N clients ($M \leq N$). This sampling approach balances communication and computational overhead. While increasing M can accelerate model convergence, it also increases local computation costs and communication overhead between the central server and clients during each global epoch.

2. Model Broadcasting: The central server broadcasts the current global model to the selected clients, as shown in Fig. 1.1 (a). For the initial global epoch, the server initializes a global model. Subsequently, it sends the aggregated model from the previous epoch.

3. Local Training: Each client performs multiple local training epochs on their local dataset using the received global model.

4. Model Uploading: After local training, clients upload their trained model parameters to the central server, as shown in Fig. 1.1 (b).

5. Model Aggregation: The central server aggregates the received model parameters to create a new global model.

1.2 Heterogeneity Issues in Federated Learning

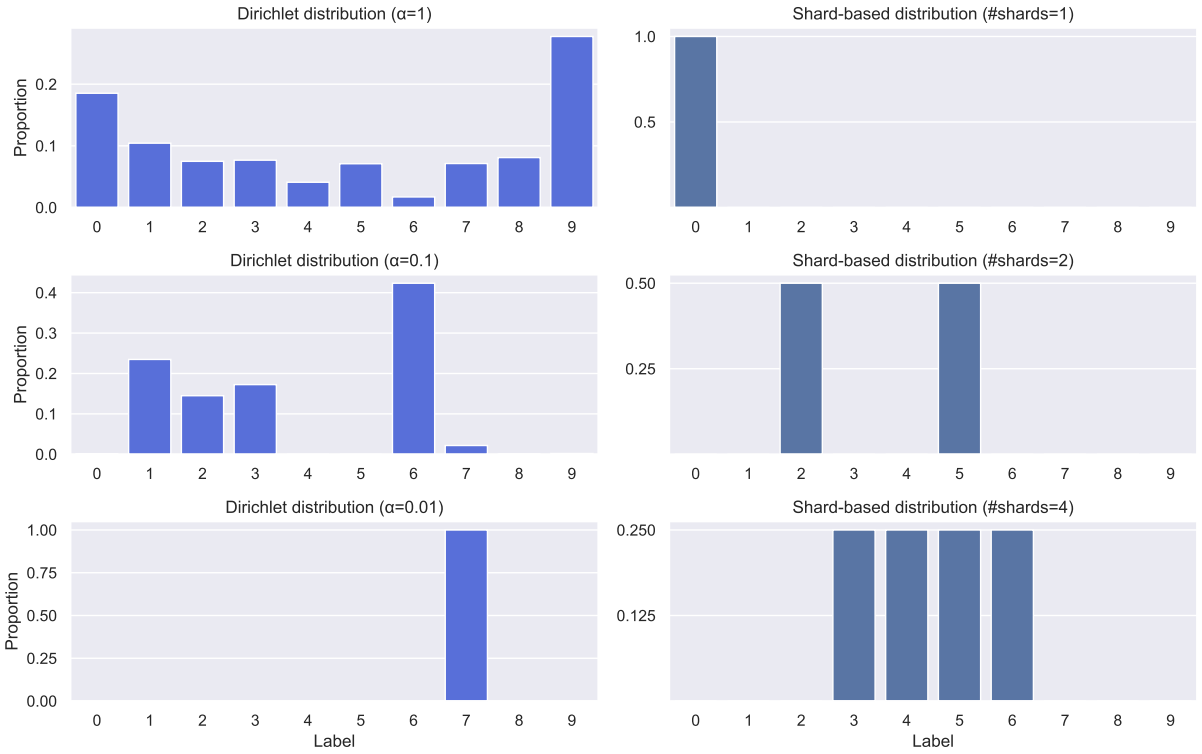
While FL offers advantages over traditional centralized training, it introduces new challenges, one of which is heterogeneity. Heterogeneity refers to the state of being diverse or different. In the context of FL, it often implies that the participating clients have varying characteristics, such as:

1. Data Distribution: Different clients may have varying data distributions, affecting the model's performance.
2. Computational Power: Clients may have varying computational resources, which can impact their ability to contribute to the training process.
3. Network Conditions: Clients may experience varying network conditions, affecting communication speed and reliability.

Heterogeneity can hinder the ability of FL to train models that perform well across all clients. Data distribution heterogeneity is also known as statistical heterogeneity, while heterogeneity in computational power and network conditions is referred to as systems heterogeneity [19]. This thesis focuses on statistical heterogeneity, as it is more challenging and offers greater potential for improving model training if addressed effectively.

1.2.1 Common Simulation Schemes for Statistical Heterogeneity: Dirichlet and Shard-Based Distributions

Current research on statistical heterogeneity often employs Dirichlet distribution [18, 36] or shard-based distribution [23, 19] to generate heterogeneous distributions among clients. For Dirichlet distribution, a

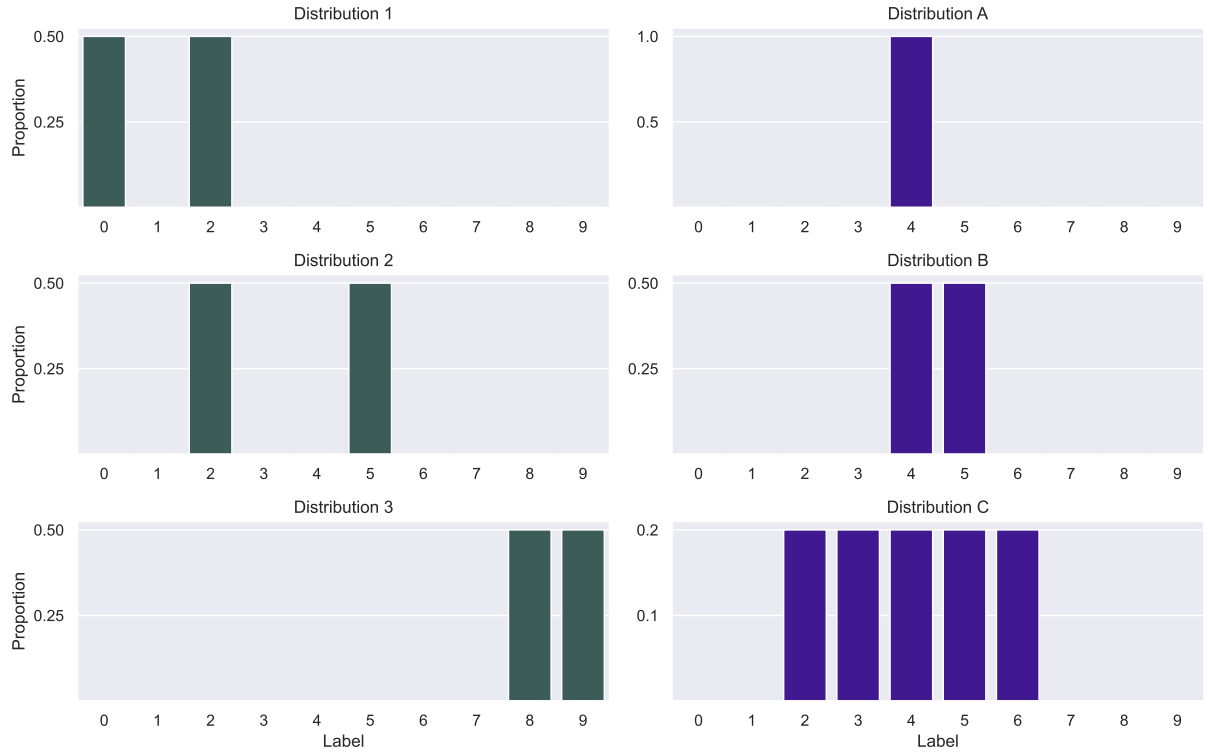


(a) Dirichlet distribution

(b) Shard-based distribution

Figure 1.2: Common heterogeneity simulation schemes. (a) Dirichlet distribution, where smaller α values result in more heterogeneous distributions. (b) Shard-based distribution, where fewer shards per client lead to more heterogeneous distributions.

hyperparameter α controls the degree of heterogeneity. Smaller α values result in more heterogeneous distributions. In shard-based distribution, samples are partitioned into different shards based on their labels. Each shard contains a specific number of samples from the same label. Subsequently, clients are assigned multiple shards. As shown in Fig. 1.2 (a), for a 10-class classification task, the Dirichlet distribution can simulate different levels of heterogeneity, with smaller values of α leading to increased heterogeneity. Fig. 1.2 (b) demonstrates that shard-based distribution also influences the degree of heterogeneity by varying the number of shards.



(a) Label heterogeneity (b) Diversity heterogeneity

Figure 1.3: Two types of statistical heterogeneity. (a) Label heterogeneity, where clients have different label distributions but similar degrees of diversity. (b) Diversity heterogeneity, where clients have both different label distributions and varying degrees of diversity.

1.2.2 Two Types of Statistical Heterogeneity: Label and Diversity Heterogeneity

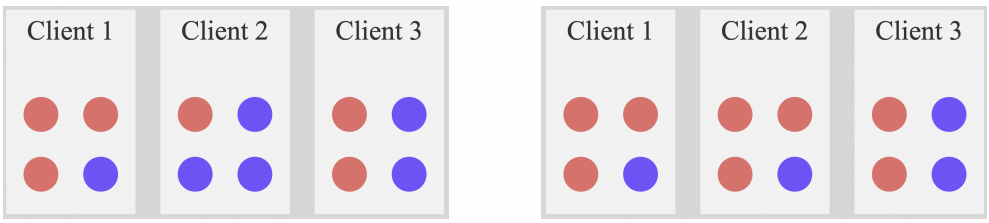
Current research often uses fixed heterogeneity levels across all clients in FL training, resulting in identical diversity degrees among clients. The diversity degree measures the diversity variability in label distribution. In this thesis, the negative value of the label distribution's variance is used as the metric for diversity degree. The details of this metric can be found in Chapter 4.

The distinction between diversity heterogeneity (a type of statistical heterogeneity often overlooked in previous research and identified in this thesis) and label heterogeneity (a term introduced in this thesis to represent statistical heterogeneity excluding diversity heterogeneity) is illustrated using shard-based distribution as follows: As illustrated in Fig. 1.3 (a), despite the differences in the distributions, their

diversity degrees are essentially the same. In contrast, Fig. 1.3 (b) demonstrates that both the distributions and diversity degrees are heterogeneous, with distribution C exhibiting a higher degree of diversity than distribution B, and distribution B showing a higher degree of diversity than distribution A.

1.2.3 Statistical Heterogeneity and Class Imbalance

As highlighted in the 2023 study [39], the class imbalance within grouped dataset is the cause of FL performance degradation under heterogeneous settings. A grouped dataset is formed by combining the local datasets of selected clients during each global epoch. A theoretical analysis inspired by this provided insights for optimizing FL training. Algorithms were then developed based on these insights and the analysis. The details of the theoretical analysis and the algorithms are provided in Chapter 3 and Chapter 4, respectively.



(a) Globally balanced (b) Globally imbalanced

Figure 1.4: Two types of imbalance. (a) shows a scenario in which local devices have imbalanced data while the global data is balanced. (b) shows a scenario in which imbalance exists both locally and globally.

This imbalance can arise from two primary factors: statistical heterogeneity among clients and an inherently imbalanced global dataset. When clients exhibit diverse data distributions, the grouped dataset tends to become more imbalanced. Similarly, a global dataset with uneven class distributions will likely lead to imbalanced subsets, even when clients are selected randomly. As shown in Fig. 1.4 (a), each circle denotes a sample in the class ”red” or the class ”blue”. Each client contains four samples. Even if the global dataset is balanced (#red: #blue = 6: 6), client 1 and client 2 still contain imbalanced distributions. In Fig. 1.4 (b), the global and local distributions are both imbalanced. The algorithms proposed in this

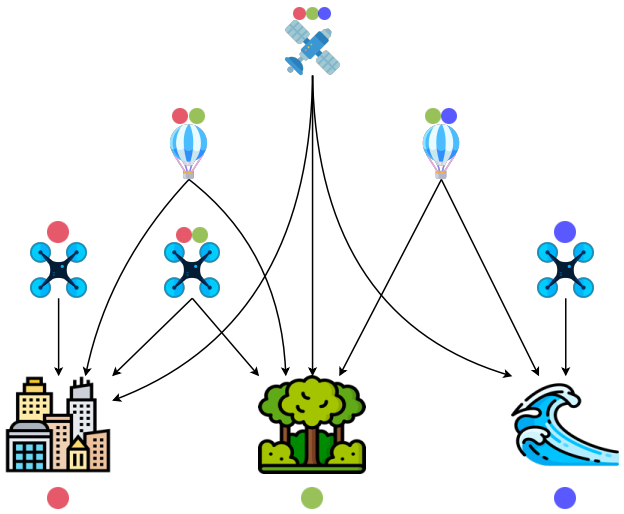


Figure 1.5: Different devices would possess varying degrees of data diversity concerning their locations and altitudes.

thesis will be tested against both types of imbalance.

1.3 Challenges of Applying Federated Learning in Aerial and Space Networks

1.3.1 How Diversity Varies Among Clients

The distributed nature of edge devices in ASNs leads to high statistical heterogeneity in their collected data, due to variations in location and altitude. When satellites in earth observation constellations capture images of different regions, the collected data exhibits heterogeneity in terms of labels, such as oceans, lands, deserts, plains, cities, and countryside.

As an intuitive example shown in Fig. 1.5, different devices in ASNs possess different data regarding their types, locations, and orbital heights. The arrows indicate that the devices in ASNs have collected data from their target sources. The red, blue, and green dots represent city, forest, and sea data, respectively. The colored dots above each device illustrate the data distribution within each ASNs device. As depicted in Figure 1.5, devices in ASNs at higher altitudes generally collect more diverse datasets. However, even

within the same altitude, devices in ASNs located in different regions can exhibit varying degrees of data diversity.

Existing research has explored various strategies to address statistical heterogeneity in federated learning, including proximal terms [19], reinforcement learning [26], and weighted averaging based on cosine similarity [28]. While these approaches have shown promise, they have not considered the existence of diversity heterogeneity. Consequently, there is a lack of algorithms specifically designed to leverage this aspect for optimization in FL.

1.3.2 How Battery Life Constraint Exacerbates the Label Heterogeneity

Unlike traditional FL applications, devices in ASNs are often constrained by their limited battery life. Given the current state of lithium battery technology [12], substantial improvements in energy density are unlikely in the near future. For example, a typical DJI drone can only fly for approximately 30 minutes.

To prioritize task completion and safety, these devices must conserve energy. This limitation significantly impacts client selection for FL training. On the one hand, it can reduce the number of clients participating in each global epoch of FL training. On the other hand, it shrinks the pool size of clients with sufficient battery levels, as clients with low battery must prioritize essential operations like returning to base.

A recent study [39] highlighted that class imbalance in the grouped dataset is the cause of FL performance degradation. Based on this insight, simulations were conducted to explore scenarios where a limited number of clients were selected to participate in FL training. As depicted in Fig. 1.6, the degree of class imbalance within grouped dataset was higher when fewer clients were selected for FL training. The imbalance degree, denoted by Δ, is defined as: $\Delta = max(\mathcal{D}) - min(\mathcal{D})$, where $\mathcal{D}$ represents the distribution of the grouped dataset. For a B-class classification task, the distribution is expressed as $\mathcal{D} = [\rho_1, \rho_2, \cdots, \rho_B]$.

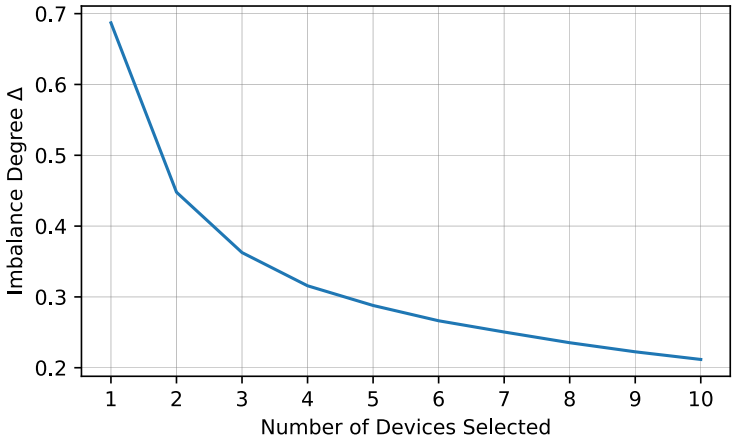


Figure 1.6: Grouped dataset imbalance degree with different number of clients selected.

To gain a more comprehensive understanding of the impacts of imbalance, a more extensive study was conducted by combining grouped datasets from multiple global epochs. A hyperparameter, ”window size,” was introduced to adjust the observation window. The grouped datasets from each ”window size” of consecutive global epochs were combined to calculate the imbalance degree.

As previously discussed, the battery constraint can limit the pool of available clients during FL training. Fig. 1.7 demonstrates that a smaller pool size can lead to a higher degree of imbalance within the grouped dataset, regardless of the observation window size. A detailed analysis of how the battery constraint exacerbates the heterogeneity issue is presented in Appendix A.

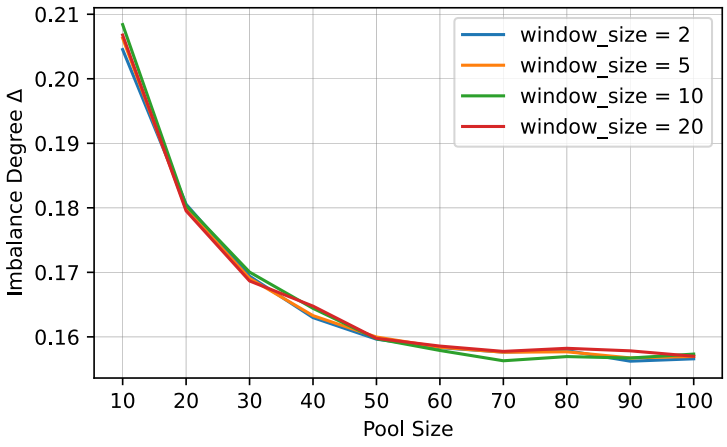


Figure 1.7: Imbalance degree under different pool sizes.

Given the challenges posed by the battery constraint for FL applications in ASNs, algorithms should be designed to accommodate higher degrees of heterogeneity to address the exacerbated class imbalance. While current research [18, 15, 19, 27] often neglects to evaluate algorithms or fails to significantly improve FL performance under high-heterogeneity conditions, this can lead to suboptimal performance in FL applications in ASNs.

1.4 Research Overview and Contributions

In this thesis, a more detailed analysis was conducted on the statistical heterogeneity issue in FL, where statistical heterogeneity was dissected into diversity heterogeneity and label heterogeneity. The diversity heterogeneity is often overlooked in other research, where statistical heterogeneity is frequently equated with label heterogeneity. Furthermore, challenges of applying FL to ASNs were identified. On the one hand, the distributed nature of ASNs necessitates the optimization of FL on diversity heterogeneity, where the various locations and altitudes lead to heterogeneity in diversity. On the other hand, the battery constraint exacerbates the class imbalance issue during the FL training course, which makes the optimization of FL on high-degree label heterogeneity distributions paramount. A theoretical analysis was then performed. Based on the target of minimizing the gradient estimation error, an inequality was derived, which indicates the upper bound of the estimation error. Based on this inequality, corresponding algorithms were devised for diversity heterogeneity and label heterogeneity.

For the diversity heterogeneity scenario, a *projection*-based weighted averaging and client selection algorithm (WeiAvgCS) was proposed by utilizing the diversity of label distributions. The diversity degree was quantified with the negative value of the label distribution's variance. However, the requirement of uploading diversity degrees to the central server may introduce privacy concerns. To avoid this privacy violation, a new metric termed *projection* was proposed, which strongly correlates with label diversity. The

projection was then used as an estimation of the actual label diversity to perform the weighted averaging and client selection, as shown in Fig. 4.1. The details of the calculation of *projection* are provided in Equation (4.2) in Chapter 4. By adopting this approach, the convergence speed of FL was significantly improved without compromising privacy. Furthermore, the WeiAvgCS algorithm can be combined with other state-of-the-art algorithms like FedProx [19], MOON [18], and Scaffold [15] to further boost the performance.

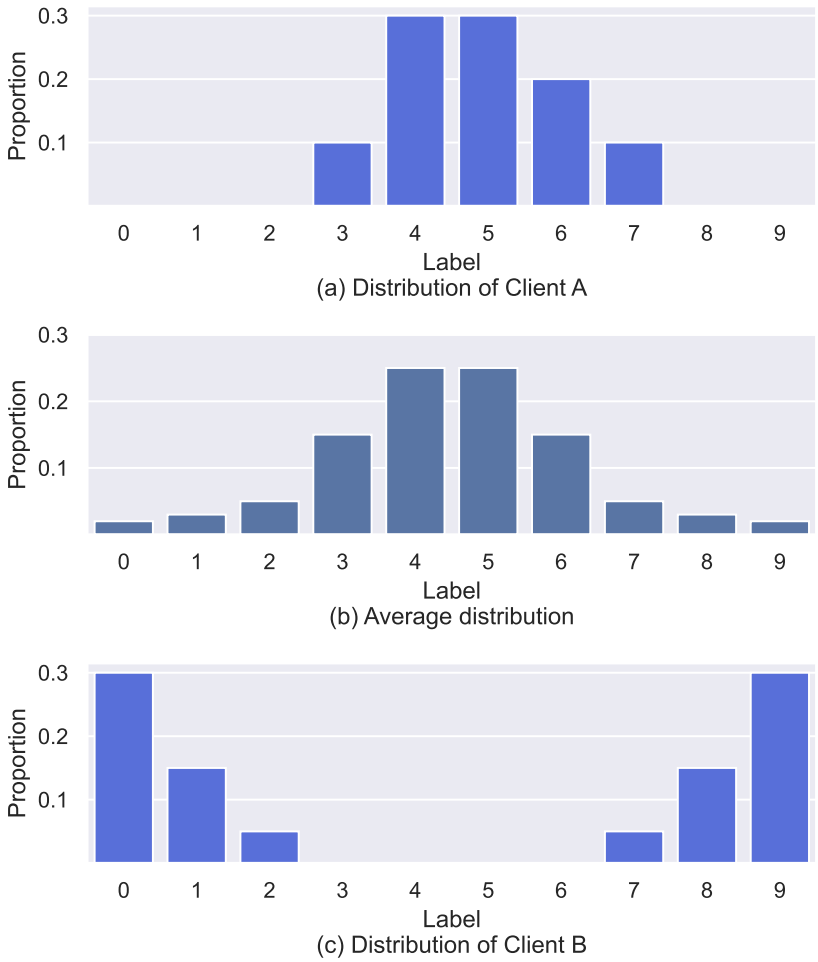


Figure 1.8: The average distribution and individual distributions of clients A and B. Client A's distribution is more closely aligned with the average distribution, indicating that client B's distribution is scarcer.

To address the high-degree label heterogeneity, a new algorithm called FedBalance was developed. It employs weighted aggregation, secured by Fully Homomorphic Encryption (FHE), to mitigate the adverse

effects of imbalance within the grouped dataset. Compared with other FL algorithms, the clients need to upload their label distribution information to the central server for weighted aggregation. However, client distribution information is not exposed to any other parties, including the central server and other clients. This is achieved through the use of FHE and the careful design of communication protocols. Figure 1.8 illustrates the average distribution of M clients in a global epoch, along with individual distributions from client A and client B. Client A's distribution is more closely aligned with the average distribution than that of client B. Conversely, client B's distribution is scarcer, indicating that its dataset can provide more valuable information for FL training in this global epoch. To emphasize the importance of client B, a higher weight will be assigned during the aggregation process. A novel metric, termed *relative scarcity*, was proposed in this thesis to quantify this distribution discrepancy. It is calculated as the inverse of the dot product between each client's distribution vector and the average distribution vector. Subsequently, M participating clients were assigned different weights based on their *relative scarcity*. Clients with scarcer distributions received higher weights during the aggregation process.

Extensive experimental results have proven the effectiveness of WeiAvgCS and FedBalance algorithms.

The key contributions of this thesis are summarized below:

- Statistical heterogeneity was dissected into diversity heterogeneity and label heterogeneity. An analysis of FL applications on ASNs revealed that previous research often overlooked diversity heterogeneity and frequently focused on less heterogeneous label heterogeneity.

- A theoretical analysis was conducted to guide the optimization of FL for both diversity heterogeneity and high-degree label heterogeneity.

- The WeiAvgCS and FedBalance algorithms were developed to address diversity heterogeneity and high-degree label heterogeneity, respectively.

- Extensive experiments were conducted to demonstrate the superiority of WeiAvgCS and FedBalance in various scenarios.

Chapter 2

Related Work

Originally proposed by Google in 2017, FL [23] offers a promising approach to training models requiring large amounts of data without collecting clients' data to a central server. By exchanging model parameters instead of the raw data with clients during the training course, FL avoids high communication costs and preserves privacy. Since then, the convergence time, communication cost, and privacy-preserving mechanisms [3, 30, 2, 22, 4, 38] remain critical research topics in FL.

While FL enables privacy-preserving distributed machine learning for massive devices, challenges like heterogeneity and class imbalance continue to hinder the performance of state-of-the-art methods. Class imbalance is closely linked to statistical heterogeneity in FL, and the battery constraint of FL applications in ASNs can further exacerbate this issue, as discussed in Section 1.3.2 and Appendix A.

The next section summarizes existing studies relevant to this research across four categories: previous schemes to mitigate the heterogeneity effect, class imbalance issue, various weighted averaging approaches, and client selection techniques. Following this, the thesis's contributions to addressing the identified research gaps are outlined.

2.1 Current Research

2.1.1 Schemes to Mitigate the Heterogeneity Effect

Heterogeneity can be categorized into two types: statistical heterogeneity and systems heterogeneity [33, 19]. Statistical heterogeneity is mostly caused by the fact that the distribution of the data varies among different clients. This leads to local models converging in different directions and the global model converging slowly. Systems heterogeneity mainly refers to the differences in hardware, network connectivity, and computational power among clients participating in FL. Research like [19, 25] focused on tackling the stragglers' issue in FL to improve the overall performance.

This thesis focuses on the statistical heterogeneity. The impacts of statistical heterogeneity were theoretically analyzed in [20]. Numerous studies have sought to mitigate the effects of statistical heterogeneity. Data distillation and transfer learning techniques were adopted in FedMD [17] to address the heterogeneity challenges. FedProx [19] introduced an additional proximal term to the local objective to prevent overfitting during local training. However, tuning the hyperparameter μ in FedProx can be a challenge, the introduced proximal term can also slow down the convergence speed. FedProx can also be used to tackle the stragglers caused by systems heterogeneity. Scaffold [15] maintained control variates to rectify the local training to mitigate the heterogeneity effect. However, as illustrated in the experimental results in Chapter 5, Scaffold failed to outperform FedAvg [23] under high-degree heterogeneity. FedMix [35] relaxed the limitation of accessing others' raw data and performed data augmentation with the assistance of other clients' data. By leveraging this strategy, FedMix can accommodate FL with different levels of privacy depending on the applications and achieve better performance. However, this algorithm may not be suitable for highly sensitive scenarios with stringent privacy requirements. MOON [18] used the similarity between model representations to correct for local training. On the contrary, relying on previous local models reduced its effectiveness when selecting a small portion of clients from a vast pool. Despite

these efforts, there is still significant room for improvement in addressing heterogeneity. While adding regularization terms [19, 18] to local objective functions requires longer computation time, algorithms like Scaffold [15] cannot even outperform FedAvg [23] under highly heterogeneous distributions.

2.1.2 Class Imbalance Issue

Class imbalance has long been an issue in the field of machine learning [14]. Resampling, reweighting, and cost modification methods [14, 8] have been proposed to mitigate its detriment. However, these techniques cannot be applied to FL directly. Recently, [39] pointed out that the class imbalance of the grouped dataset is the cause of performance degradation under heterogeneous distributions. Several existing studies in the field of FL have focused on addressing the heterogeneity problem from the perspective of class imbalance. Astraea [11] introduced a mediator to perform client selection based on the label distribution of clients. However, acquiring label distribution information directly from clients may pose a privacy violation. The method of [32] also performed client selection to reduce the class imbalance based on the estimation of clients' label distribution. However, the estimation may not be accurate since the estimation cannot be generalized to multi-class classification tasks [39]. Ratio Loss algorithm [27] proposed a monitor to detect the imbalance in an always-training setting. This monitor would alert the administrator to apply the proposed ratio Loss function to mitigate the imbalance issue. Nevertheless, this method can not work well in settings where training is not continuous. According to the experimental results in Chapter 5, the Ratio Loss algorithm's performance is sensitive to both the learning rate and the degree of heterogeneity, which can limit its generalizability.

2.1.3 Various Weighted Averaging Approaches

Weighted model aggregation has also been explored in FL studies. FedAT [5] leveraged a straggler-aware, weighted aggregation heuristic to steer and balance the training for further accuracy improvement. FedAdp

[28] employed a weighted averaging method to accelerate convergence, based on the angle between the local and global gradients. However, this approach is primarily suitable for cross-silo scenarios. In [1], the authors proposed using entropy as a weight for model aggregation based on clustering results. However, their approach requires all clients to participate and upload entropy information to the central server, which can raise privacy concerns.

2.1.4 Client Selection Techniques

A client selection method was adopted to counterbalance the bias introduced by heterogeneous data distributions [26]. The POWER-OF-CHOICE algorithm selects clients with higher local losses to improve convergence speed [7]. FedLE utilizes a clustering approach for client selection to optimize energy consumption in FL [29]. FedCS [24] focused on client selection under heterogeneous resources in the mobile edge, but it did not account for diversity differences among clients. Fed-CBS [39] aimed to select the optimal set of clients probabilistically to mitigate class imbalance. However, client selection may not be viable when only a limited number of clients are available within a given time slot.

2.2 Research Gaps and Contributions

While existing studies have made significant contributions to addressing statistical heterogeneity, they have primarily focused on label heterogeneity and overlooked the importance of diversity heterogeneity. Consequently, these studies have not been able to fully leverage this aspect for FL optimization. Additionally, state-of-the-art algorithms often struggle to perform well under distributions with high-degree label heterogeneity, which is particularly critical in FL applications within ASNs. This thesis deconstructs statistical heterogeneity into two categories: diversity heterogeneity and label heterogeneity, allowing for a more nuanced analysis. Based on theoretical insights and extensive experimental results, specialized algorithms, WeiAvgCS and FedBalance, were developed to address diversity heterogeneity and high-degree

label heterogeneity, respectively.

Chapter 3

Theoretical Analysis

For the training dataset $\mathcal{X}$, consisting of B classes, the dataset can be partitioned by class as $\mathcal{X} = \{\mathcal{X}_{C_1}, \mathcal{X}_{C_2}, \cdots, \mathcal{X}_{C_B}\}$. The label distribution on client i is given by $\mathcal{D}_i = [\rho_{i,1}, \rho_{i,2}, \cdots, \rho_{i,j}, \cdots, \rho_{i,B}]$, where $\rho_{i,j}$ indicates the proportion of samples in the j_{th} class for client i and $\sum_{j=1}^{B} \rho_{i,j} = 1$. Typically, the goal is to train a model that generalizes well and assigns equal importance to each class. Accordingly, the objective function for model training should be evaluated in the same manner.

Following this approach, the global objective function (F), typically the loss function given data samples ($\mathcal{X}$) and global model parameters ($\mathbf{w}$), can be decomposed into the average loss for each class, all weighted equally by $\frac{1}{B}$, as shown in Equation (3.1).

$$F(\mathcal{X}, \mathbf{w}) = \sum_{j=1}^{B} \frac{1}{B} F(\mathcal{X}_{C_j}, \mathbf{w}) \tag{3.1}$$

For neural network models, which are widely used and serve as the default model type in this thesis, the gradient of the global objective function F is computed following forward propagation by applying the chain rule. This theoretical global gradient, denoted as ∇F, is expressed as shown in Equation (3.2). Since $\mathbf{w}$ represents the global model parameters and remains unchanged for each client during a given

global epoch, it will be omitted in the following equations for simplicity.

$$\nabla F(\mathcal{X}, \mathbf{w}) = \sum_{j=1}^{B} \frac{1}{B} \nabla F(\mathcal{X}_{C_j}, \mathbf{w}) \tag{3.2}$$

In FL, the aggregation of local gradients is used as an approximation of the centralized global gradient. However, approximation errors may arise due to discrepancies between local and global objective functions, which are caused by distribution heterogeneity across local datasets. Based on the class distribution on client i, $\mathcal{D}_i$, the local gradient can then be expressed as shown in Equation (3.3).

$$\nabla F_i(\mathcal{X}_i) = \sum_{j=1}^{B} \rho_{i,j} \nabla F(\mathcal{X}_{i,C_j}) \tag{3.3}$$

At the end of each global epoch in FL, the central server aggregates the model parameters received from the participating clients. If the aggregation is parameterized with different weights ω_i for each client i, where $\sum_{i=1}^{M} \omega_i = 1$, the estimated global gradient can be expressed as shown in Equation (3.4). For simplification, it is assumed that each client contains an equal number of samples.

$$\begin{aligned} \widetilde{\nabla F(\mathcal{X})} &= \sum_{i=1}^{M} \omega_i \nabla F_i(\mathcal{X}_i) \\ &= \sum_{i=1}^{M} \omega_i \sum_{j=1}^{B} \rho_{i,j} \nabla F(\mathcal{X}_{i,C_j}) \\ &= \sum_{j=1}^{B} \sum_{i=1}^{M} \omega_i \rho_{i,j} \nabla F(\mathcal{X}_{i,C_j}) \end{aligned} \tag{3.4}$$

The gap between the theoretical global gradient ($\nabla F(\mathcal{X})$) and the estimated global gradient ($\widetilde{\nabla F(\mathcal{X})}$) is the approximation error, which is illustrated in Equation (3.5). Through triangle inequality, the approximation error, denoted by $\parallel \nabla F(\mathcal{X}) - \widetilde{\nabla F(\mathcal{X})} \parallel$, is bounded by $\sum_{j=1}^{B} \parallel \nabla F(\mathcal{X}_{i,C_j}) \parallel \left| \frac{1}{B} - \sum_{i=1}^{M} \omega_i \rho_{i,j} \right|$.

$$
\begin{aligned}
&\| \nabla F(\mathcal{X}) - \widetilde{\nabla F(\mathcal{X})} \| \\
&= \Big\| \sum_{j=1}^{B} \Big(\frac{1}{B} \nabla F(\mathcal{X}_{i,C_j}) - \sum_{i=1}^{M} \omega_i \rho_{i,j} \nabla F(\mathcal{X}_{i,C_j}) \Big) \Big\| \\
&\leq \sum_{j=1}^{B} \underbrace{\| \nabla F(\mathcal{X}_{i,C_j}) \|}_{Part\ 1} \underbrace{\Big| \frac{1}{B} - \sum_{i=1}^{M} \omega_i \rho_{i,j} \Big|}_{Part\ 2}
\end{aligned}
\tag{3.5}
$$

Part 1 of Inequality (3.5) remains unchanged during the aggregation step. Therefore, the optimization will focus on Part 2 of Inequality (3.5).

3.1 Weighted Averaging and Client Selection Based on Diversity (WeiAvgCS)

For $\rho_{i,j}$, since there are B classes in total and each class is treated equally, its expectation is $\frac{1}{B}$, represented as Equation (3.6).

$$
E(\rho_{i,j}) = \frac{1}{B} \tag{3.6}
$$

The variance of $\rho_{i,j}$ depends on the label distribution on client i, which is denoted as σ_i^2 as shown in Equation (3.7).

$$
Var(\rho_{i,j}) = \sigma_i^2 \tag{3.7}
$$

For all M participating clients, the expectation and variance of $\sum_{i=1}^{M} \omega_i \rho_{i,j}$ are $\frac{1}{B}$ and $\sum_{i=1}^{M} \omega_i^2 \sigma_i^2$, as shown in Equations (3.8) and (3.9), respectively.

$$
\begin{aligned}
E(\sum_{i=1}^{M} \omega_i \rho_{i,j}) &= \sum_{i=1}^{M} \omega_i E(\rho_{i,j}) \\
&= \sum_{i=1}^{M} \omega_i \frac{1}{B} \\
&= \frac{1}{B} \sum_{i=1}^{M} \omega_i \\
&= \frac{1}{B}
\end{aligned}
\tag{3.8}
$$

$$
\begin{aligned}
Var(\sum_{i=1}^{M} \omega_i \rho_{i,j}) &= \sum_{i=1}^{M} \omega_i^2 Var(\rho_{i,j}) \\
&= \sum_{i=1}^{M} \omega_i^2 \sigma_i^2
\end{aligned}
\tag{3.9}
$$

Thus, the term $\frac{1}{B} - \sum_{i=1}^{M} \omega_i \rho_{i,j}$ has an expectation of 0 and a variance of $\sum_{i=1}^{M} \omega_i^2 \sigma_i^2$. Because a smaller variance results in a tighter bound on the approximation error, two approaches can be used to reduce the error: (1) Distributing ω_i inversely proportional to σ_i^2 to minimize the overall variance. (2) Selecting clients with higher class diversity during the client selection step, which is achieved by retaining the high-diversity clients selected in the previous global epoch. Based on this strategy, an algorithm called WeiAvgCS is developed, with further details provided in Chapter 4.

3.2 Weighted Averaging Based on *Relative Scarcity* (FedBalance)

When heterogeneity arises solely from label distribution rather than diversity, the previously mentioned algorithm may not be effective. However, optimization can still be achieved by assigning weights according to the label distribution to reduce Part 2 in Inequality (3.5). The goal is to assign weights to the M clients such that, for each class, $\sum_{i=1}^{M} \omega_i \rho_{i,j}$ closely approximates the value $\frac{1}{B}$. This is accomplished by introducing a metric, termed *relative scarcity*, based on the distribution information of the M selected clients. As illustrated in Fig. 1.8, *relative scarcity* is defined as the inverse of the dot product between the average

distribution and individual client distributions. By assigning weights in proportion to the *relative scarcity*, the gap between $\sum_{i=1}^{M} \omega_i \rho_{i,j}$ and $\frac{1}{B}$ is reduced, thereby tightening the upper bound on the approximation error. As shown in Fig. 4.5, compared to simple averaging, this weighted aggregation method brings each $\sum_{i=1}^{M} \omega_i \rho_{i,j}$ closer to $\frac{1}{B}$. The details of the algorithm following this approach, named FedBalance, are provided in Chapter 4.

Chapter 4

Methodologies

4.1 The WeiAvgCS Algorithm

For a dataset $\mathcal{X}$, with B distinct classes, the index of a client in FL is denoted as $i \in [1, N]$, where N represents the total number of available clients. A higher variance in client i's distribution, $\mathcal{D}_i$, suggests a more skewed distribution, indicating lower sample diversity for that client. Therefore, the negative variance of $\mathcal{D}_i$ is used to quantify the dataset diversity for client i, as shown in Equation (4.1).

$$d_i = -\sigma^2(\mathcal{D}_i) \tag{4.1}$$

Building on the insights from Chapter 3, an algorithm named *WeiAvgCS* was developed, as presented in Algorithm 1.

Algorithm 1 WeiAvgCS

Require: Initial global model $\mathbf{w}^1$, number of participating clients in each global epoch M, number of global epochs T, maximum retaining epochs for each client R, number of clients retained each epoch r, client pool consisting of all N clients C.

for global epoch t in $[1, 2, 3, \cdots, T]$ **do**
  if t ==1 **then**
    $\boldsymbol{c}^{t,M} \leftarrow$ The central server randomly selects M clients from C.
  else
    The central server retains r clients $\boldsymbol{c}^{t,r}$ with highest value based on $\vec{\boldsymbol{d}}_{\cdot}^{t-1}$.
    The central server randomly selects $M - r$ clients $\boldsymbol{c}^{t,M-r}$ from C.
    If any clients in $\boldsymbol{c}^{t,r}$ and $\boldsymbol{c}^{t,M-r}$ are selected R times right before the t_{th} global epoch, replace them with other clients.
    $\boldsymbol{c}^{t,M} \leftarrow \boldsymbol{c}^{t,r}$ + $\boldsymbol{c}^{t,M-r}$
  end if
  The central server broadcasts the current global model $\mathbf{w}^t$ to $\boldsymbol{c}^{t,M}$. Clients in $\boldsymbol{c}^{t,M}$ perform local training and upload the trained local model parameters $\vec{\mathbf{w}}_{\cdot}^t = [\mathbf{w}_1^t, \mathbf{w}_2^t, \cdots, \mathbf{w}_i^t, \cdots, \mathbf{w}_M^t]$ to the central server.
  if Use_Actual_Diversity **then**
    Clients in $\boldsymbol{c}^{t,M}$ compute and upload the diversity $\vec{\boldsymbol{d}}_{\cdot}^t = [d_1, d_2, \cdots, d_M]$.
  else
    The central server simply averages $\vec{\mathbf{w}}_{\cdot}^t$ to obtain the temporal global model $\mathbf{w}^{t+1}$.
    The central server calculates each client's *projection* as shown in Equation (4.2), denoted as $\vec{\boldsymbol{p}}_{\cdot}^t = \{\boldsymbol{p}_1^t, \boldsymbol{p}_2^t, \cdots, \boldsymbol{p}_i^t, \cdots, \boldsymbol{p}_M^t\}$.
    Use $\vec{\boldsymbol{p}}_{\cdot}^t$ as the estimation of $\vec{\boldsymbol{d}}_{\cdot}^t$: $\vec{\boldsymbol{d}}_{\cdot}^t = \vec{\boldsymbol{p}}_{\cdot}^t$.
  end if
  $\vec{\boldsymbol{z}}_{\cdot}^t \leftarrow (\vec{\boldsymbol{d}}_{\cdot}^t - min(\vec{\boldsymbol{d}}_{\cdot}^t))/(max(\vec{\boldsymbol{d}}_{\cdot}^t) - min(\vec{\boldsymbol{d}}_{\cdot}^t))$.
  $\vec{\boldsymbol{z}}_{\cdot}^t \leftarrow (\vec{\boldsymbol{z}}_{\cdot}^t + 1)^{\lambda}$.
  The central server calculates the weight for each client in $\boldsymbol{c}^{t,M}$: $\vec{\omega}_{\cdot}^t = \vec{\boldsymbol{z}}_{\cdot}^t / \sum \vec{\boldsymbol{z}}_{\cdot}^t$, where $\vec{\omega}_{\cdot}^t = [\omega_1^t, \omega_2^t, \cdots, \omega_i^t, \cdots, \omega_M^t]$ and $\sum_{i=1}^M \omega_i^t = 1$.
  The central server conducts the aggregation to obtain the new global model: $\mathbf{w}^{t+1} \leftarrow \sum_{i=1}^M \omega_i^t \cdot \mathbf{w}_i^t$.
end for
return $\mathbf{w}^{T+1}$

The algorithm initializes the global model as $\mathbf{w}^1$. In each global epoch, M clients are selected for local training, and the process spans a total of T global epochs. Lines 2 to 9 in Algorithm 1 outline the client selection procedures. In the first global epoch, clients are selected randomly, as indicated in line 3. From the second global epoch onward, high-diversity clients from the previous epoch are retained to participate in the next epoch of training. The hyperparameters r and R control the number of clients retained and the number of consecutive epochs each client participates in, as detailed in lines 4 to 8. The r hyperparameter balances the retaining of high-diversity clients and variability of clients participating in the FL training, while the R hyperparameter prevents certain high-diversity clients from being selected for too long. After local training, clients upload their model updates, as shown in line 10 of Algorithm 1. In some scenarios, clients may also upload their label diversity to the central server, as described in lines 11 and 12, although this may not always be practical. When direct diversity measurement is not feasible, client diversity is estimated following the procedures in lines 13 to 16. The specifics of this diversity estimation are discussed in the following paragraph. Line 18 of Algorithm 1 performs min-max normalization on the (estimated) diversity vector. In line 19, a non-linear transformation is applied to adjust the weights assigned to clients based on their diversity. The hyperparameter λ controls the emphasis placed on model parameters from high-diversity clients. When λ equals 0, the algorithm degenerates to FedAvg with identical weights assigned to each client. For simplicity, an equal number of samples are distributed to each client. Finally, line 20 normalizes the weights to ensure the sum of weights assigned to the M clients equals 1. Line 21 then performs weighted aggregation to obtain the new global model parameters.

Although WeiAvgCS based on diversity can improve performance compared to FedAvg, uploading clients' actual diversity may raise privacy concerns. To address this issue, a metric termed *projection* is proposed to estimate the label distribution diversity. The calculation of *projection* was illustrated in Fig. 4.1, where $\mathbf{w}^t$ represents the global model parameters at the t_{th} global epoch before being sent to clients,

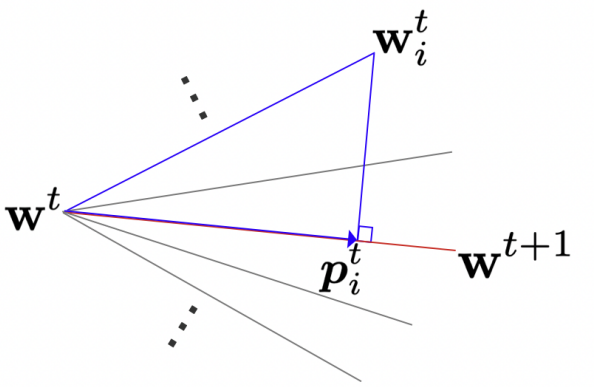


Figure 4.1: The *projection* ($\boldsymbol{p}_i^t$) of client i's local update ($\mathbf{w}_i^t - \mathbf{w}^t$) at the t_{th} global epoch onto the global update ($\mathbf{w}^{t+1} - \mathbf{w}^t$).

$\mathbf{w}_i^t$ represents the updated model parameters on client i, and $\mathbf{w}^{t+1}$ represents the simple averaged global model parameters after clients have sent their local model updates to the central server. Note that $\mathbf{w}^{t+1}$ is identical to the aggregated model in FedAvg. The *projection* norm, as shown in Fig. 4.1, can be calculated using Equation (4.2) after flattening the model parameters into a one-dimension vector.

$$\boldsymbol{p}_i^t = \frac{(\mathbf{w}_i^t - \mathbf{w}^t) \cdot (\mathbf{w}^{t+1} - \mathbf{w}^t)}{\|\mathbf{w}^{t+1} - \mathbf{w}^t\|} \tag{4.2}$$

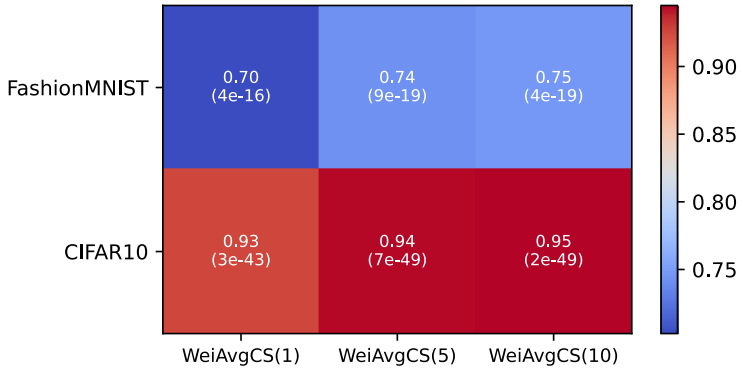


Figure 4.2: Correlation between *projection* and actual diversity in WeiAvgCS with different λ values on FashionMNIST and CIFAR10.

A correlation heatmap was also constructed, as illustrated in Fig. 4.2. For each grid, all *projection* values within an epoch are concatenated and averaged based on the client ID. The Pearson correlation

coefficient and p-value between the actual diversity and the *projection* were then calculated. These calculations were performed for each dataset, as shown in the rows of Fig. 4.2. Additionally, different values of λ in WeiAvgCS were applied within each dataset to further assess the correlation between actual diversity and the *projection*, as shown in the columns of Fig. 4.2. The numbers in parentheses on the horizontal axis represent the λ values, while the numbers in each grid display the correlation coefficient (top) and the corresponding p-value (bottom). The results demonstrate a strong positive correlation, indicating the potential of using *projection* values as an estimate of actual diversity for weighted averaging and client selection. This approach allows WeiAvgCS to eliminate the need for actual client diversity data, thereby reducing the risk of privacy violations in real applications.

4.2 The FedBalance Algorithm

4.2.1 Core of FedBalance: Weighted Averaging

Through the theoretical analysis in previous chapter, the optimization can also be achieved by assigning weights to the M clients such that, for each class, $\sum_{i=1}^{M} \omega_i \rho_{i,j}$ closely approximates the value $\frac{1}{B}$. This is accomplished by introducing a metric, termed *relative scarcity*, based on the distribution information of the M selected clients. Table. 4.1 shows the notations needed in the FedBalance algorithm. The pseudo-code of the FedBalance algorithm is shown in Algorithm 2.

Table 4.1: Notations Used in the FedBalance Algorithm

Notation	Meaning
B	Number of classes. For MNIST, FashionMNIST, CIFAR10, $B = 10$. For CIFAR100, $B = 100$.
N	Total number of clients
C	Client pool consisting of all N clients.
M	Number of selected clients for each global epoch.
$\mathcal{D}_i$	Label distribution on client i. $\mathcal{D}_i = [\rho_{i,1}, \rho_{i,2}, \cdots, \rho_{i,j}, \cdots, \rho_{i,B}]$.
$\bar{\mathcal{D}}$	Average distribution of M clients. $\bar{\mathcal{D}} = [\frac{1}{M}\sum_{i=1}^{M}\rho_{i,1}, \frac{1}{M}\sum_{i=1}^{M}\rho_{i,2}, \cdots, \frac{1}{M}\sum_{i=1}^{M}\rho_{i,B}]$
s_i	Scarcity of client i among M clients. $s_i = 1/\langle \mathcal{D}_i, \bar{\mathcal{D}} \rangle$.
ω_i	Weight assigned to client i.

Algorithm 2 FedBalance

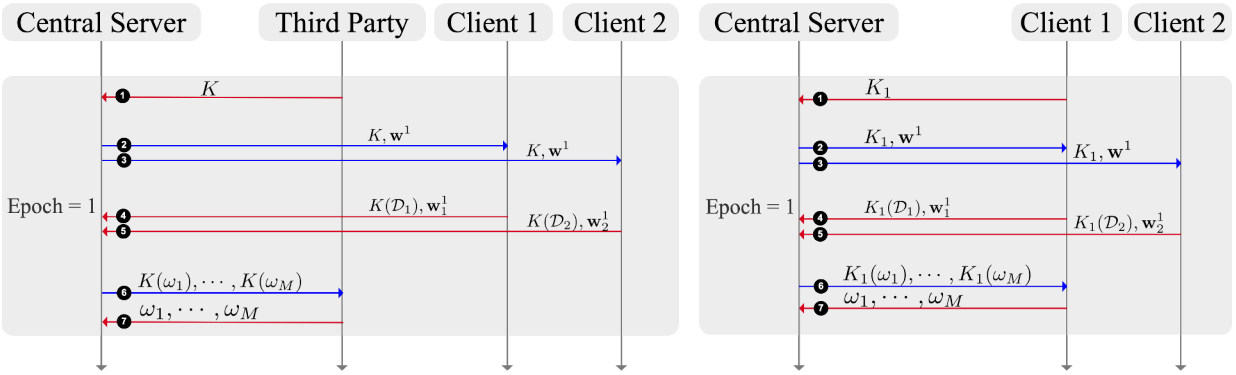
Require: Initial global model $\mathbf{w}^1$, number of participating clients in each global epoch M, number of global epochs T, client pool consisting of all N clients C. Public key K and private key K^{-1} of the third party.

1: The third party sends the public key K to the central server.
2: **for** t in $[1, 2, 3, \cdots, T]$ **do**
3: $c^{t,M} \leftarrow$ The central server randomly select M clients from C.
4: **for** each client $i \in c^{t,M}$ **in parallel do**
5: The central server broadcasts global model $\mathbf{w}^t$, public key K to client i.
6: Client i conducts the local training, resulting in a trained model $\mathbf{w}_i^t$, and generates the label distribution $\mathcal{D}_i$.
7: Client i sends trained model $\mathbf{w}_i^t$ and encrypted label distribution $K(\mathcal{D}_i)$ back to the central server.
8: **end for**
9: The central server calculates the average distribution $K(\bar{\mathcal{D}}) \leftarrow [\frac{\sum_{i=1}^{M} K(\rho_{i,1})}{M}, \cdots, \frac{\sum_{i=1}^{M} K(\rho_{i,B})}{M}]$.
10: The central server calculates the *relative scarcity* of each client: $K(s_i) = 1/\langle K(\mathcal{D}_i), K(\bar{\mathcal{D}})\rangle$.
11: The central server calculates the weight assigned to each client: $K(\omega_i) = K(s_i)/\sum_{l=1}^{M} K(s_l)$.
12: The central server sends encrypted weights $[K(\omega_1), K(\omega_2), \cdots, K(\omega_M)]$ to the third party.
13: The third party sends the decrypted weights $[\omega_1, \omega_2, \cdots, \omega_M]$ back to the central server.
14: The central server performs aggregation: $\mathbf{w}^{t+1} \leftarrow \sum_{i=1}^{M} \omega_i \cdot \mathbf{w}_i^t$
15: **end for**
16: **return** $\mathbf{w}^{T+1}$

At the start of the FedBalance process, a third party sends the public key K to the central server, as outlined in line 1 of Algorithm 2. For each global epoch, the central server broadcasts the global model $\mathbf{w}^t$ and the public key K to the selected clients. The clients then perform local training and upload the trained models along with encrypted label distributions to the central server, as described in lines 3 to 7 of Algorithm 2. The central server subsequently computes the *relative scarcity* of each client's data. Since FHE is used to secure the label distribution information, this calculation is conducted directly on the encrypted distributions. The *relative scarcity* s_i for client i is calculated as follows. Given each participating

client's encrypted label distribution $K(\mathcal{D}_i) = [K(\rho_{i,1}), K(\rho_{i,2}), \cdots, K(\rho_{i,j}), \cdots, K(\rho_{i,B})]$, the averaged distribution across the M participating clients is $K(\bar{\mathcal{D}}) = [\frac{1}{M}\sum_{i=1}^{M} K(\rho_{i,1}), \cdots, \frac{1}{M}\sum_{i=1}^{M} K(\rho_{i,B})]$. As illustrated in Fig. 1.8, the distribution of client A is more closely aligned with the average distribution compared to that of client B, indicating that client B's distribution is scarcer. Therefore, the inverse of the dot product between client i's distribution and the average distribution is used as the metric for *relative scarcity*. A higher s_i value indicates that client i's distribution is scarcer. The calculation of s_i is reflected in lines 9 to 10 of Algorithm 2. Based on the *relative scarcity*, different weights are assigned to clients, with more emphasis on those possessing scarcer data. Since the previous calculations are based on encrypted label distributions, the final weights remain encrypted. Consequently, the central server sends the encrypted weights $[K(\omega_1), K(\omega_2), \cdots, K(\omega_M)]$ to the third party for decryption. The process of calculating and decrypting the weights is detailed in lines 11 to 13 of Algorithm 2. Finally, the central server performs a weighted average to obtain the new global model, as depicted in line 14 of Algorithm 2.

4.2.2 Computing ω with Fully Homomorphic Encryption



(a) with a third party (b) without a third party

Figure 4.3: Communication protocols: (a) with a third party, and (b) without a third party.

To securely transmit label distribution information without disclosing it to other clients or the central server, the FHE technique is adopted to encrypt the communication. Several communication protocols

can be designed for this purpose. For instance, a third party may be introduced to provide the encryption and decryption keys. As shown in Fig. 4.3 (a), in step 1, a public key (K) is sent by a third party to the central server. The central server then broadcasts the public key (K) and the initial global model ($\mathbf{w}^1$) to all selected clients in steps 2 and 3. After local training, the clients upload the updated local model parameters ($\mathbf{w}_1^1$ and $\mathbf{w}_2^1$) and the encrypted label distributions ($K(\mathcal{D}_1)$ and $K(\mathcal{D}_2)$) to the central server in steps 4 and 5. The central server then performs calculations based on the encrypted label distributions to obtain the encrypted weights ($K(\omega_1), K(\omega_2), \cdots, K(\omega_M)$). These encrypted weights are sent to the third party in step 6. After the decryption procedure, the weights ($\omega_1, \omega_2, \cdots, \omega_M$) are sent back to the central server in step 7 for weighted aggregation. In scenarios where a third party is not available, a trusted client can be selected to act as the third party, responsible for generating the encryption and decryption keys. The procedures are similar to those involving a third party, as shown in Fig. 4.3 (b).

A test was conducted to assess the time consumption of FHE. As shown in Fig. 4.4, the entire process can be completed within seconds on an Apple M1 Pro chip. In contrast, the local training for each global epoch requires significantly more time depending on different settings, rendering the time consumption of FHE negligible. Moreover, the local training and FHE procedures can be executed in parallel: once the selected clients receive the global model ($\mathbf{w}^t$) and encryption key (K), they can upload the encrypted label distributions while simultaneously performing local training. In conclusion, FedBalance incurs minimal communication and computation overhead.

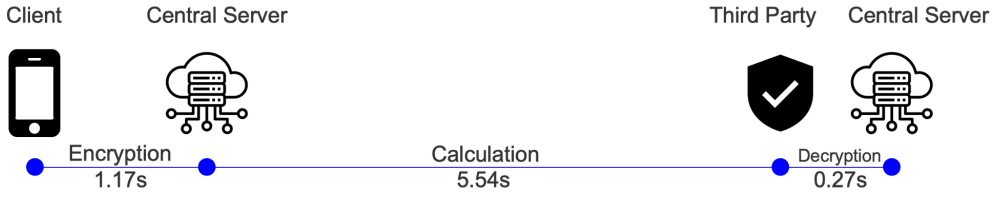


Figure 4.4: Time consumption for FHE. The FHE process can be completed within seconds on an Apple M1 Pro chip, significantly faster than local training. Additionally, FHE and local training can be conducted in parallel, resulting in no additional time cost.

4.2.3 Further Improvement: FedBalanceFilter

Since the calculation of *relative scarcity* s_i (and weight ω_i) is independent of the local training procedure, it is possible to asynchronously upload the encrypted label distributions and the trained model parameters to the central server. Specifically, more than M clients can be randomly selected and asked to upload their encrypted label distributions to the central server. After the relative scarcities and weights are calculated, clients with low weights (or low relative scarcities) can be filtered out to reduce training costs. These filtered-out clients contribute minimally to the final model in that epoch. This approach is referred to as the FedBalanceFilter algorithm. As shown in Algorithm 3, on line 2, more than M clients are selected to collect their encrypted label distributions. After calculating the respective weights, the excess clients with the lowest weights values are filtered out. The remaining procedures follow those of the FedBalance algorithm, with the exception that the label distributions have already been uploaded to the central server. Overall, a more balanced dataset can be achieved with minimal cost by filtering out a small portion of clients.

Algorithm 3 FedBalanceFilter

Require: Initial global model $\mathbf{w}^1$, number of participating clients in each global epoch M, number of global epochs T, client pool consisting of all N clients C. Public key K and private key K^{-1} of the third party.

for t in $[1, 2, 3, \cdots, T]$ **do**
  $c^{t,M+A} \leftarrow$ The central server randomly select $M + A$ clients from C.
  for each client $i \in c^{t,M+A}$ **in parallel do**
    The central server broadcasts public key K to client i.
    Each client i uploads $K(\mathcal{D}_i)$ to the central server.
  end for
  The central server calculates the average distribution $K(\bar{\mathcal{D}}) \leftarrow [\frac{\sum_{i=1}^{M+A} K(\rho_{i,1})}{M+A}, \cdots, \frac{\sum_{i=1}^{M+A} K(\rho_{i,B})}{M+A}]$.
  The central server calculates the *relative scarcity* of each client: $K(s_i) = 1/\langle K(\mathcal{D}_i), K(\bar{\mathcal{D}})\rangle$.
  The central server calculates the weight assigned to each client: $K(\omega_i) = K(s_i)/\sum_{l=1}^{M+A} K(s_l)$.
  The central server sends encrypted weights $[K(\omega_1), \cdots, K(\omega_{M+A})]$ to the third party.
  The third party sends the decrypted weights $[\omega_1, \cdots, \omega_{M+A}]$ back to the central server.
  $c^{t,M} \leftarrow$ Filter out A clients with the lowest ω_i from $c^{t,M+A}$.
  for each client $i \in c^{t,M}$ **in parallel do**
    The central server broadcasts global model $\mathbf{w}^t$ to client i.
    Client i conducts the local training, resulting a trained model $\mathbf{w}_i^t$.
    Client i sends the trained model $\mathbf{w}_i^t$ back to the central server.
  end for
  The central server calculates the average distribution of the M remaining clients $K(\bar{\mathcal{D}}) \leftarrow [\frac{\sum_{i=1}^{M} K(\rho_{i,1})}{M}, \cdots, \frac{\sum_{i=1}^{M} K(\rho_{i,B})}{M}]$.
  The central server calculates the *relative scarcity* of each client: $K(s_i) = 1/\langle K(\mathcal{D}_i), K(\bar{\mathcal{D}})\rangle$.
  The central server calculates the weight assigned to each client: $K(\omega_i) = K(s_i)/\sum_{l=1}^{M} K(s_l)$.
  The central server sends encrypted weights $[K(\omega_1), \cdots, K(\omega_M)]$ to the third party.
  The third party sends the decrypted weights $[\omega_1, \cdots, \omega_M]$ back to the central server.
  The central server performs aggregation: $\mathbf{w}^{t+1} \leftarrow \sum_{i=1}^{M} \omega_i \cdot \mathbf{w}_i^t$.
end for
return $\mathbf{w}^{T+1}$

4.2.4 Why FedBalance Should Work

The rationale behind why the weighted aggregation in FedBalance is effective is visualized in Fig. 4.5. In the FedAvg algorithm, model parameters are simply averaged to obtain the global model. Similarly, the label distributions of the selected clients are averaged to derive the overall distribution as shown in Fig. 4.5. To ensure robustness, this averaging process is repeated 1,000 times, with the resulting distribution sorted in descending order. A similar approach is used to derive the average distribution for the FedBalance algorithm. As depicted in Fig. 4.5, FedBalance achieves a more uniform distribution on average compared to FedAvg. As analyzed in Part 2 of Inequality (3.5) in Chapter 3, when $\sum_{i=1}^{M} \omega_i \rho_{i,j}$ is closer to $\frac{1}{B}$, the upper bound of the approximation error becomes tighter, resulting in improved FL performance.

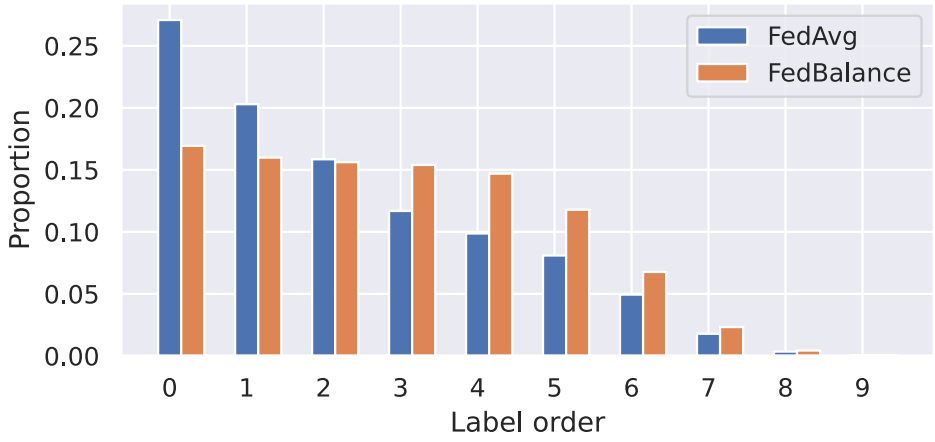


Figure 4.5: Grouped data distribution under simple averaging (FedAvg) and weighted averaging (FedBalance). A more uniform distribution indicates a tighter upper bound for approximation error, as shown in Part 2 of Inequality (3.5).

Chapter 5

Experiment Results

The experiments were implemented in PyTorch using three different datasets: FashionMNIST [31], CIFAR10 [16], and CINIC10 [9]. FashionMNIST serves as a drop-in replacement for the traditional MNIST dataset, featuring images of clothing items instead of handwritten digits. It consists of 60,000 training images and 10,000 testing images, each with a resolution of 28×28 pixels in grayscale, categorized into 10 classes. CIFAR10, another widely used dataset in machine learning, has the same number of training and testing samples as FashionMNIST. Each CIFAR10 image is a 32×32 RGB image, also divided into 10 categories. CINIC10 was created by combining CIFAR10 with images selected and downsampled from the ImageNet database [9], serving as a drop-in alternative to CIFAR10. CINIC10 bridges the benchmarking gap, especially when ImageNet is too large or complex for comparison with CIFAR10. For each dataset, the following neural networks were used: a 4-layer CNN for FashionMNIST, a 5-layer CNN for CIFAR10, and a 5-layer CNN for CINIC10. The learning rate was set to 0.01 by default, with SGD as the optimizer, using a momentum coefficient of 0.9. The L2 normalization coefficient was set to 1e-4. The local training batch size was set to 32 for FashionMNIST and 64 for CIFAR10 and CINIC10, with the number of local epochs set to 10 for all three datasets. For simplicity, each client was assigned 500 samples. To perform a thorough comparison with our algorithms, other state-of-the-art benchmarks in FL were also applied, in-

cluding FedProx [19], MOON [18], Scaffold [15], and Ratio Loss [27]. Since the Ratio Loss algorithm is sensitive to the learning rate and degree of heterogeneity, the comparison with it is presented in a separate section later. The experimental details for the WeiAvgCS algorithm and the FedBalance algorithm differ slightly, and these differences will be described in the following sections.

5.1 The WeiAvgCS Algorithm

Experiments for the WeiAvgCS algorithm were conducted on the FashionMNIST and CIFAR10 datasets. The number of global epochs was set to 50 for FashionMNIST and 100 for CIFAR10. To better control the diversity of the label distribution, a manual distribution scheme with a hyperparameter γ was devised to manage label diversity, rather than using the Dirichlet distribution commonly applied in other studies [7, 36, 37, 13, 6]. The effect of γ on the algorithm will be discussed in later sections. For each global epoch, $M = 10$ clients were selected out of a total of $N = 100$ clients for local training. Due to the stochastic nature of FL in the client selection step, hundreds of different random seeds were employed to obtain an average performance for comparison in the experiment. This approach was taken to ensure a more consistent evaluation and to allow conclusions to be drawn with statistical confidence.

5.1.1 WeiAvgCS with Different λ

As shown in Fig. 5.1 (a), WeiAvgCS was performed on FashionMNIST using the actual diversity with different values of λ. WeiAvgCS demonstrates superiority over vanilla FedAvg across a wide range of λ values, though different λ values eventually converged to similar target accuracy. Subsequently, the *projection* was used as an approximation of the actual diversity to conduct the same experiment, as depicted in Fig. 5.1 (b). The accuracy trend observed is similar to that in Fig. 5.1 (a), confirming that the WeiAvgCS algorithm is effective and can achieve better convergence performance than FedAvg across a wide range of λ values.

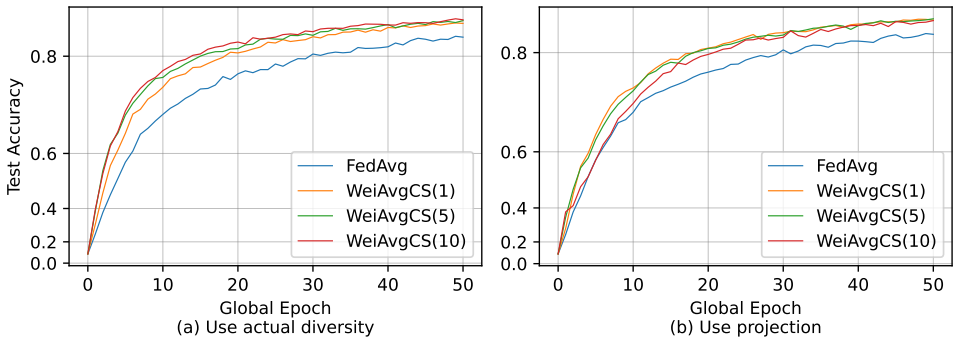


Figure 5.1: WeiAvgCS (λ) using (a) actual diversity and (b) *projection*. Both experiments demonstrate the superiority of WeiAvgCS over FedAvg across a wide range of λ values.

5.1.2 Comparing WeiAvgCS with Different Benchmarks.

To demonstrate the performance of WeiAvgCS, several state-of-the-art benchmarks were implemented, including FedProx [19], MOON [18], and Scaffold [15], which were designed to address the heterogeneity problem in FL. Since WeiAvgCS is orthogonal to these algorithms, it was also combined with each of them to evaluate whether further improvements could be achieved.

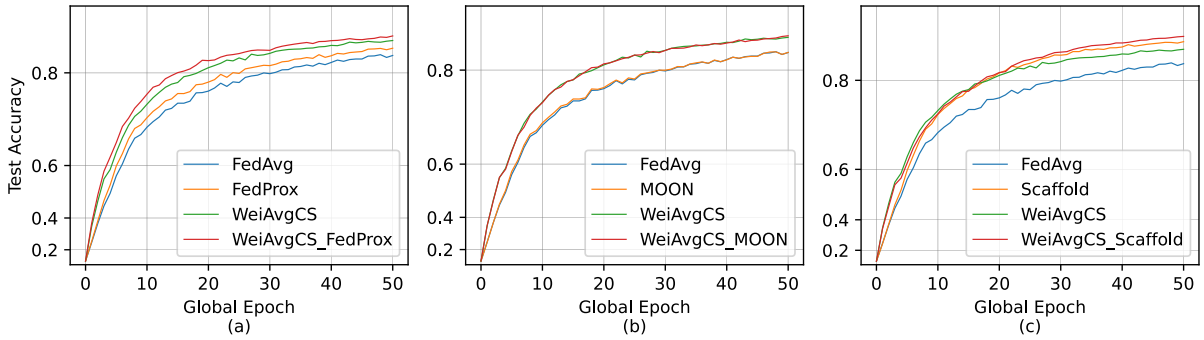


Figure 5.2: Test accuracy of FedAvg and WeiAvgCS with three benchmarks: (a) FedProx, (b) MOON, (c) Scaffold, and their respective combinations on FashionMNIST ($\gamma = 1$).

As shown in Fig. 5.2 and Fig. 5.3, the experiments on FashionMNIST and CIFAR10 demonstrate the superiority of WeiAvgCS. Compared to FedProx, WeiAvgCS achieves better performance improvement on both datasets. Furthermore, when combined with FedProx, the performance is enhanced even further. For MOON, WeiAvgCS also improves the performance, even though MOON alone shows limited effectiveness

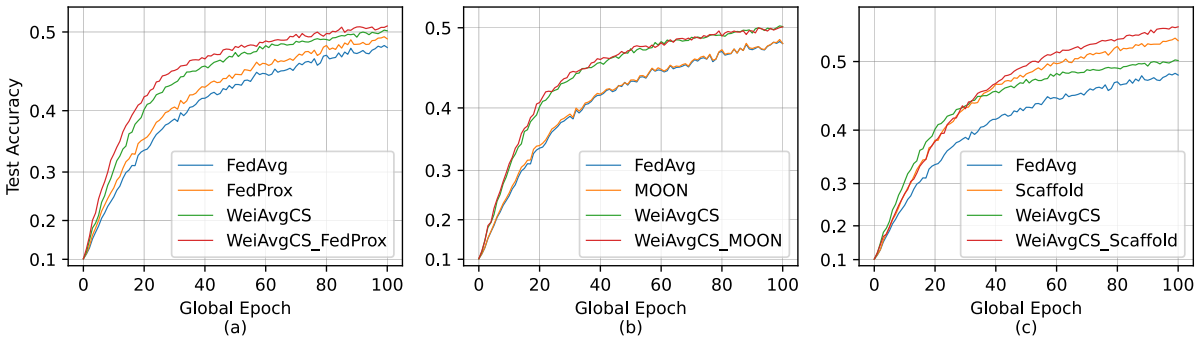


Figure 5.3: Test accuracy of FedAvg and WeiAvgCS with three benchmarks: (a) FedProx, (b) MOON, (c) Scaffold, and their respective combinations on CIFAR10 (γ=1).

in these scenarios. With Scaffold, WeiAvgCS provides better convergence performance only in the initial epochs. When combining WeiAvgCS with Scaffold, the combined approach significantly outperforms both Scaffold and WeiAvgCS alone.

Additional experiments were conducted under different distribution settings with γ set to 0. γ controls the degree of the diversity heterogeneity. When γ is small, the diversity heterogeneity among clients becomes lower. The results of these experiments are presented in Table 5.1. The column labeled "Epochs" indicates the number of global epochs each algorithm requires to reach the target accuracy, which is defined by the lowest accuracy achieved by any algorithm. The column labeled "TestAcc(std)" represents the final accuracy after a specified number of global epochs, along with the standard error of the final accuracy across different random seeds. The conclusions remain consistent with the above analysis, demonstrating the robustness of WeiAvgCS in addressing diversity heterogeneity.

Table 5.1: Numerical experimental results for FashionMNIST and CIFAR10 Across Different Degrees of Diversity Heterogeneity

	FashionMNIST				CIFAR10			
	γ=1		γ=0		γ=1		γ=0	
	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)
FedAvg	46	0.8291(0.0332)	50	0.8542(0.0207)	97	0.4822(0.0400)	90	0.5177(0.0294)
FedProx	38	0.8391(0.0282)	39	0.8598(0.0170)	79	0.4922(0.0354)	82	0.5215(0.0270)
MOON	46	0.8286(0.0528)	47	0.8551(0.0206)	88	0.4822(0.0395)	90	0.5161(0.0290)
Scaffold	25	0.8559(0.0637)	25	0.8756(0.0085)	50	0.5259(0.0294)	44	0.5627(0.0245)
WeiAvgCS	30	0.8500(0.0201)	31	0.8659(0.0127)	58	0.5013(0.0298)	67	0.5268(0.0179)
WeiAvgCS_FedProx	24	0.8563(0.0158)	28	0.8686(0.0129)	50	0.5067(0.0238)	62	0.5258(0.0195)
WeiAvgCS_MOON	26	0.8508(0.0226)	30	0.8686(0.0115)	60	0.5010(0.0290)	65	0.5257(0.0140)
WeiAvgCS_Scaffold	23	0.8674(0.0098)	24	0.8784(0.0052)	45	0.5421(0.0653)	41	0.5795(0.0131)

5.1.3 Identifying Limitations: When WeiAvgCS Falls Short

The effectiveness of the WeiAvgCS algorithm depends on a strong correlation between *projection* and actual diversity. However, this correlation may weaken if the number of local epochs is reduced or if the local models are significantly underfitted, potentially leading to the failure of the WeiAvgCS algorithm. Unlike global epochs T, local epochs refer specifically to the local training conducted by each client. Experimental results on FashionMNIST, as shown in Fig. 5.4, highlight the impact of underfitting. When the number of local epochs is reduced to 1 or 2, the correlation diminishes, resulting in poorer performance of WeiAvgCS compared to FedAvg, as depicted in Fig. 5.5. Conversely, increasing the number of local epochs to 5 or 10 strengthens the correlation, thereby enhancing the convergence performance of WeiAvgCS. In general, such low local epoch settings are uncommon, especially when each client has a small number of samples. Therefore, WeiAvgCS is expected to perform effectively in most scenarios.

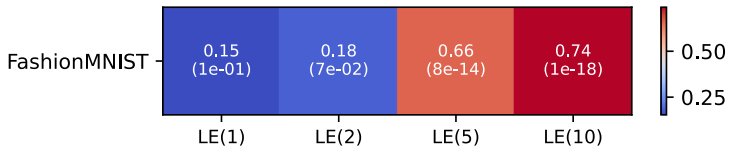


Figure 5.4: Correlation between *projection* and actual diversity across different local training epochs (1, 2, 5, and 10). Stronger correlations are observed with an increasing number of local epochs.

5.2 The FedBalance Algorithm

The FL setting was simulated by splitting each dataset into $N = 100$ clients. In each global epoch, $M = 10$ clients were selected for local training. For the FedBalanceFilter algorithm, $M + A = 15$ clients were initially selected, and $A = 5$ clients with the lowest weights were subsequently filtered out. Two common heterogeneity simulation schemes were adopted to conduct the experiments: Dirichlet distribution and shard-based distribution [7, 36, 37, 13, 6]. For the Dirichlet distribution, a hyperparameter α controls the degree of heterogeneity, with smaller α values resulting in more heterogeneous distributions. In the shard-

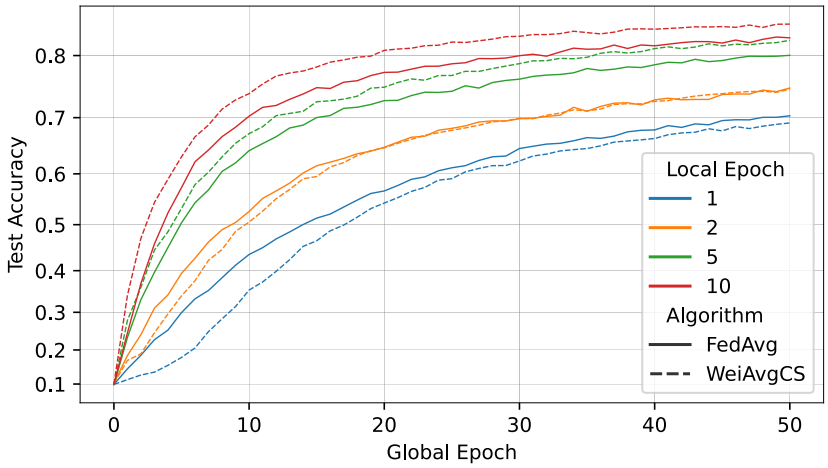


Figure 5.5: Convergence trends with different local training epochs. WeiAvgCS shows no performance improvement when the number of local epochs is too low.

based distribution, samples were partitioned into different shards based on their labels, with the number of shards assigned to each client indicating the degree of heterogeneity. The shard size in the experiment was set to 250, with each client containing 2 shards of different classes. This approach allows the global data distribution to be changed without affecting the degree of heterogeneity, which is not feasible with the Dirichlet distribution. Previous research [10] often omits highly heterogeneous distribution scenarios, which could be more common when conducting FL with ASNs due to the battery constraint [10]. Such scenarios were included in this thesis to demonstrate the algorithms' performance under high-degree label heterogeneity. As mentioned in Chapter 1, a recent study [39] revealed that class imbalance within grouped dataset is a major obstacle in FL training. Statistical heterogeneity and class imbalance are often closely related. The imbalance of the grouped dataset can arise from statistical heterogeneity itself or be combined with a globally imbalanced dataset. In the following sections, both scenarios were explored.

5.2.1 Under Globally Balanced Distributions

When the global dataset is balanced, the degree of imbalance depends on the degree of heterogeneity. Under varying degrees of heterogeneity settings, the test accuracy lines of different algorithms in the

CIFAR10 dataset are displayed in Fig. 5.6. Experimental results on FashionMNIST and CINIC10 datasets are shown in Appendix B.

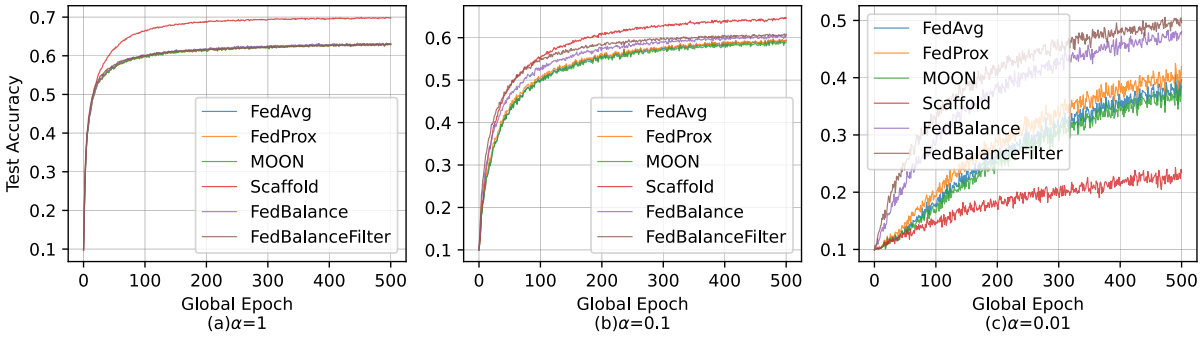


Figure 5.6: Performance of algorithms on the CIFAR10 dataset under Dirichlet distribution with varying degrees of heterogeneity. Scaffold performs effectively under low heterogeneity ($\alpha \geq 0.1$) but fails to outperform FedAvg in highly heterogeneous settings. FedBalance and FedBalanceFilter bridge the gap in training highly heterogeneous FL scenarios.

Table 5.2: Numerical experimental results for the CIFAR10 dataset under Dirichlet distribution with varying α values.

Algorithms	Dirichlet($\alpha = 1$)		Dirichlet($\alpha = 0.1$)		Dirichlet($\alpha = 0.01$)	
	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs
FedAvg	0.6315(0.0051)	380	0.5936(0.0248)	418	0.3966(0.0578)	490
FedProx	0.6310(0.0051)	450	0.5943(0.0218)	418	0.4193(0.0536)	417
MOON	0.6288(0.0073)	383	0.5896(0.0232)	-	0.3847(0.0710)	-
Scaffold	0.6977(0.0048)	59	0.6462(0.0153)	153	0.2346(0.0828)	-
FedBalance	0.6303(0.0083)	372	0.6044(0.0199)	309	0.4808(0.0454)	214
FedBalanceFilter	0.6289(0.0065)	434	0.6083(0.0137)	243	0.4962(0.0466)	158

When the distribution is close to homogeneity (Dirichlet distribution with $\alpha = 1$), the Scaffold performs significantly better than other benchmarks. Under low heterogeneity (Dirichlet distribution with $\alpha = 0.1$), Scaffold remains the best, while FedBalance and FedBalanceFilter outperform other benchmarks except for Scaffold. However, when the degree of heterogeneity is high (Dirichlet distribution with $\alpha = 0.01$), the Scaffold does not even surpass FedAvg. While FedProx consistently shows superiority over FedAvg, MOON, and Scaffold, the FedBalance and FedBalanceFilter algorithms achieve significantly better convergence levels and speeds. The numerical results are also presented in Table 5.2. Under a highly heterogeneous distribution, FedBalance achieves 8.42% and 6.15% higher test accuracy than FedAvg and

FedProx, respectively, while Scaffold and MOON fail to outperform FedAvg. The target accuracy was set as the last epoch of FedAvg test accuracy minus 0.001, to determine how many epochs each algorithm requires to reach that target. FedBalance converges much faster than the other benchmarks, with FedBalanceFilter performing even better than FedBalance. The performance degradation of Scaffold under high-heterogeneity distribution has been observed in other studies as well, such as [34], revealing a potential limitation for Scaffold: its effectiveness is restricted to low heterogeneity settings. Given that other benchmarks show only marginal improvement over FedAvg under high heterogeneity, FedBalance and FedBalanceFilter address the gap in training highly heterogeneous FL scenarios.

5.2.2 Under Globally Imbalanced Distributions

When the global dataset is imbalanced, combined with statistical heterogeneity, the grouped dataset distribution can become even more imbalanced. To simulate different degrees of imbalance in the global distribution without altering the heterogeneity degree, the shard-based distribution was adopted. In this thesis, each shard contains 250 samples from the same class, with each client containing two shards from different classes. A subset of B classes was randomly selected as majority classes (MC), while the remaining classes ($B - MC$) were designated as minority classes. The Imbalance Ratio (IR) denotes the ratio of the number of samples in each majority class to the number in each minority class in the global dataset. The IR values were set to 1, 5, and 10 to observe how the imbalance ratio impacts the performance of each algorithm. When IR equals 1, the global dataset is balanced. The algorithm was also tested by varying the number of MC to 2, 5, and 8.

As shown in Fig. 5.7 and Table 5.3, when the imbalance ratio is 1, indicating a balanced global dataset, Scaffold performs worse than FedAvg. However, the FedBalance and FedBalanceFilter algorithms demonstrate significantly better convergence performance compared to other benchmarks. When the imbalance ratio is set to 5, with the majority class comprising 20% of the classes ($MC = 2$), a similar performance

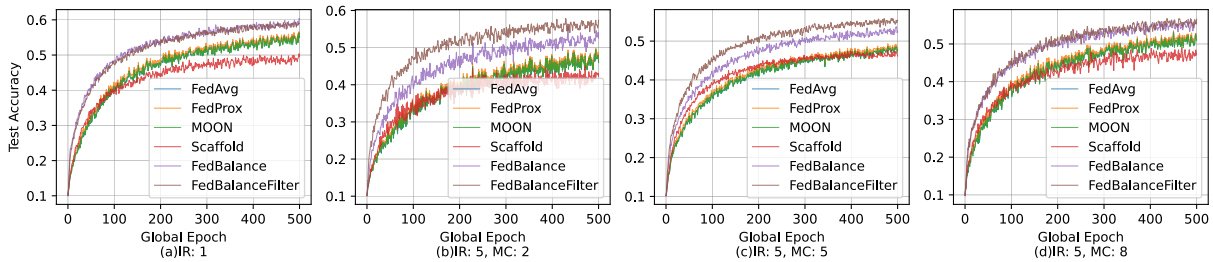


Figure 5.7: Experiments on the CIFAR10 dataset with the same degree of heterogeneity but varying numbers of majority classes. Subfigure (a) displays results on a balanced global dataset for comparison, where $IR = 1$. IR represents the imbalance ratio between the number of samples in the majority class and the minority class, with a higher IR indicating a more imbalanced global dataset. MC denotes the number of majority classes, while the number of minority classes in the 10-class classification is $10 - MC$.

Table 5.3: Numerical experimental results for the CIFAR10 dataset with an imbalance ratio (IR) set to 5.

Algorithms	$IR = 1$		$IR = 5$					
			$MC = 2$		$MC = 5$		$MC = 8$	
	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs
FedAvg	0.5497(0.0788)	431	0.4612(0.0776)	352	0.4785(0.0555)	434	0.5121(0.0564)	372
FedProx	0.5565(0.0757)	377	0.4681(0.0740)	296	0.4838(0.0522)	375	0.5156(0.0560)	372
MOON	0.5503(0.0774)	431	0.4597(0.0765)	342	0.4772(0.0554)	434	0.5081(0.0568)	372
Scaffold	0.4962(0.0709)	-	0.4228(0.0878)	-	0.4615(0.0496)	-	0.4710(0.0625)	-
FedBalance	0.6023(0.0459)	206	0.5255(0.0767)	167	0.5249(0.0511)	191	0.5563(0.0422)	209
FedBalanceFilter	0.5894(0.0494)	207	0.5742(0.0496)	91	0.5537(0.0459)	129	0.5658(0.0395)	175

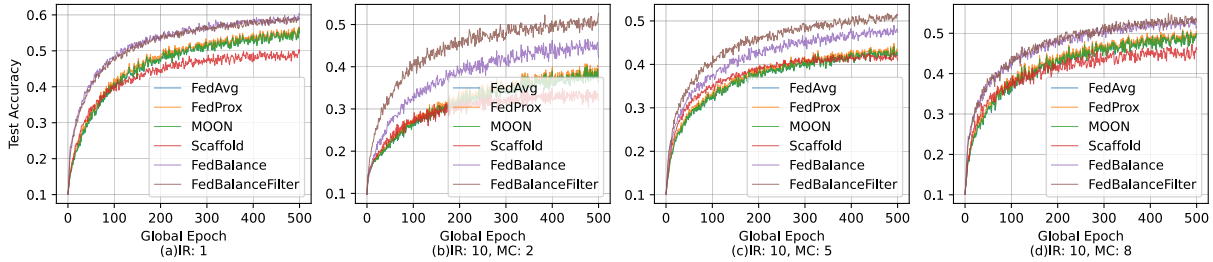


Figure 5.8: Experiments on the CIFAR10 dataset with the same degree of heterogeneity but varying numbers of majority classes. Subfigure (a) displays results on a balanced global dataset for comparison, where $IR = 1$. IR represents the imbalance ratio between the number of samples in the majority class and the minority class, with a higher IR indicating a more imbalanced global dataset. MC denotes the number of majority classes, while the number of minority classes in the 10-class classification is $10 - MC$.

Table 5.4: Numerical experimental results for the CIFAR10 dataset with an imbalance ratio (IR) set to 10.

Algorithms	$IR = 1$		$IR = 10$					
			$MC = 2$		$MC = 5$		$MC = 8$	
	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs
FedAvg	0.5497(0.0788)	431	0.3621(0.0788)	280	0.4224(0.0543)	328	0.4979(0.0484)	452
FedProx	0.5565(0.0757)	377	0.3687(0.0780)	269	0.4281(0.0517)	318	0.5049(0.0498)	393
MOON	0.5503(0.0774)	431	0.3644(0.0777)	280	0.4211(0.0505)	328	0.4971(0.0500)	470
Scaffold	0.4962(0.0709)	-	0.3203(0.0729)	-	0.4097(0.0455)	322	0.4604(0.0580)	-
FedBalance	0.6023(0.0459)	206	0.4211(0.0943)	131	0.4740(0.0525)	169	0.5165(0.0399)	240
FedBalanceFilter	0.5894(0.0494)	207	0.5262(0.0453)	63	0.5128(0.0428)	117	0.5296(0.0513)	217

pattern is observed across all algorithms, with FedBalanceFilter showing a notably better performance than FedBalance. Maintaining the imbalance ratio at 5 and increasing the number of majority classes to 5 does not change the overall performance pattern. However, when the majority class number is increased to 8, the performance gap between FedBalanceFilter and FedBalance narrows, though both still outperform the other benchmarks. Fig. 5.8 and Table 5.4 present consistent experimental results when the imbalance ratio is set to 10.

According to the results in Fig. A.5, the heterogeneity degree of the shard-based distribution with two distinct shards per client falls between Dirichlet distributions with $\alpha = 0.1$ and $\alpha = 0.01$, indicating a medium level of heterogeneity. Despite this, the Scaffold algorithm still fails to outperform FedAvg, underscoring the need for further FL optimization under medium and high heterogeneity. The FedBalance and FedBalanceFilter algorithms proposed in this thesis effectively bridge this gap, enhancing training performance in highly heterogeneous FL scenarios.

5.2.3 Comparison with the Ratio Loss algorithm

In addition to the benchmarks used in previous experiments, Ratio Loss [27] directly addresses the issue of class imbalance in FL. The Ratio Loss algorithm treats FL as an always-training training process, introducing a monitor to track the data composition. If a consistent imbalance pattern is detected, the Ratio Loss function replaces the original cross-entropy loss function to correct the training process.

However, it was observed that the Ratio Loss algorithm is highly sensitive to the learning rate. In all experiments, the learning rate was set to 0.01. When applying the same learning rate to the Ratio Loss algorithm with the same optimizer under the Dirichlet distribution with $\alpha = 0.1$, as shown in Fig. 5.9 (a), the algorithm fails to improve after several training epochs. After reducing the learning rate to 0.001, as used in [27], the Ratio Loss algorithm demonstrates superiority over other benchmarks, as shown in Fig. 5.9 (b). However, even with the learning rate maintained at 0.001, the Ratio Loss algorithm fails

when the heterogeneity level increases from low heterogeneity (Dirichlet distribution with $\alpha = 0.1$) to medium (shard-based distribution with two shards per client) or high heterogeneity (Dirichlet distribution with $\alpha = 0.01$), as illustrated in Fig. 5.9 (c) and Fig. 5.9 (d).

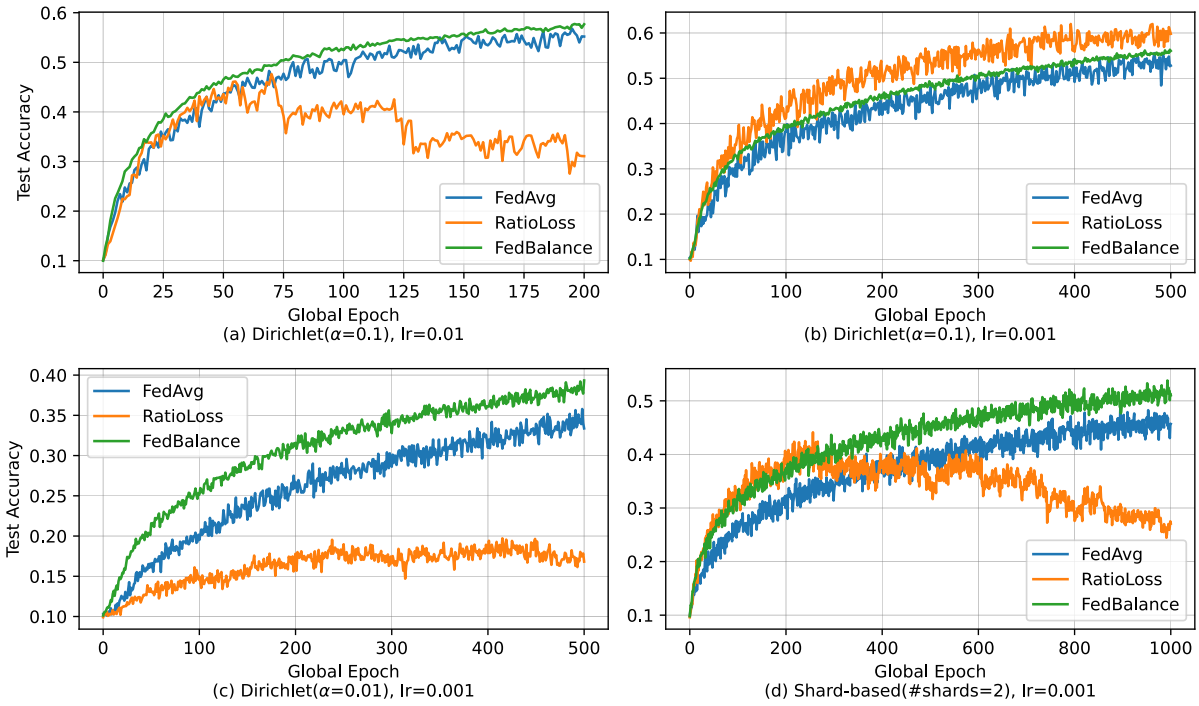


Figure 5.9: Experimental results of the Ratio Loss algorithm, FedAvg, and FedBalance algorithms under varying learning rates and degrees of heterogeneity.

Upon reviewing the average label distribution used in the original paper [27], it was found that the average distribution of the EMNIST dataset is as depicted in Fig. 5.10. Based on previous experimental results in Fig. A.4, it can be inferred that the heterogeneity degree of the distribution in [27] falls between a Dirichlet distribution with $\alpha = 1$ and homogeneity. As shown in the experimental results in Fig. A.5, FedAvg's performance under a Dirichlet distribution with $\alpha = 1$ is already close to its performance under a homogeneous distribution. This suggests that the Ratio Loss algorithm is not well-suited to address the imbalance caused by heterogeneity, particularly in cases of high-degree heterogeneity.

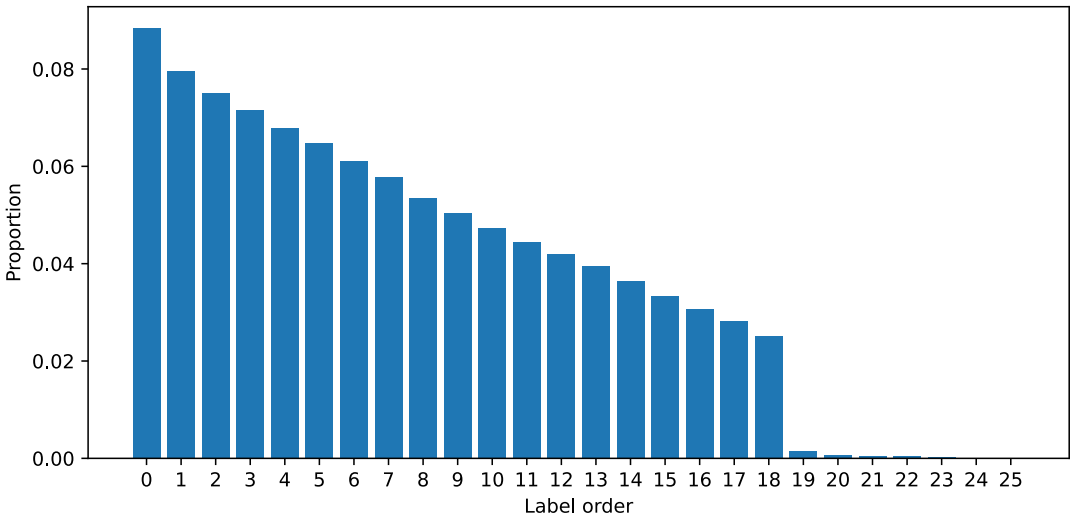


Figure 5.10: Label distribution of the EMNIST dataset used in the Ratio Loss algorithm. This distribution exhibits a low degree of heterogeneity based on prior experimental results shown in Fig. A.4.

Chapter 6

Conclusion

This thesis investigated FL applications on ASNs. Detailed analyses were conducted to demonstrate the existence of diversity heterogeneity in ASNs and to explain why the heterogeneity issue is exacerbated by the unique battery constraint.

A review of the literature revealed that current research has not adequately addressed these challenges. Diversity heterogeneity is often overlooked, and FL optimizations under high-degree heterogeneity have failed to achieve significant improvements.

A theoretical analysis was conducted to identify directions for FL optimization under diversity heterogeneity and high-degree label heterogeneity. Based on this analysis, corresponding algorithms were proposed.

To address diversity heterogeneity, the WeiAvgCS algorithm was introduced. It performs weighted aggregation and client selection based on a metric termed *projection*, which approximates the actual diversity of label distribution for each client. This approximation allows the WeiAvgCS algorithm to achieve optimization without obtaining additional information from clients, thus preserving privacy.

For high-degree label heterogeneity, the FedBalance algorithm was proposed. It conducts weighted aggregation using a metric termed *relative scarcity*, which reflects the scarcity of each client's distribution

relative to the average distribution in each global epoch. The calculation of *relative scarcity* is protected by the FHE technique, ensuring no party learns other clients' distributions.

Extensive experiments were conducted to demonstrate the superiority of WeiAvgCS and FedBalance over other state-of-the-art benchmarks.

Bibliography

[1] Brahim Aamer et al. “Entropy-Driven Stochastic Policy for Fast Federated Learning in Beyond 5G Edge-RAN”. In: *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE. 2021, pp. 1–6.

[2] Muhammad Asad, Ahmed Moustafa, and Takayuki Ito. “Fedopt: Towards communication efficiency and privacy preservation in federated learning”. In: *Applied Sciences* 10.8 (2020), p. 2864.

[3] Keith Bonawitz et al. “Towards federated learning at scale: System design”. In: *Proceedings of machine learning and systems* 1 (2019), pp. 374–388.

[4] Christopher Briggs, Zhong Fan, and Peter Andras. “Federated learning with hierarchical clustering of local updates to improve training on non-IID data”. In: *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2020, pp. 1–9.

[5] Zheng Chai et al. “Fedat: A communication-efficient federated learning method with asynchronous tiers under non-iid data”. In: *ArXivorg* (2020).

[6] Huancheng Chen, Haris Vikalo, et al. “The best of both worlds: Accurate global and personalized models through federated learning with data-free hyper-knowledge distillation”. In: *arXiv preprint arXiv:2301.08968* (2023).

[7] Yae Jee Cho, Jianyu Wang, and Gauri Joshi. “Client selection in federated learning: Convergence analysis and power-of-choice selection strategies”. In: *arXiv preprint arXiv:2010.01243* (2020).

[8] Yin Cui et al. “Class-balanced loss based on effective number of samples”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 9268–9277.

[9] Luke N Darlow et al. “Cinic-10 is not imagenet or cifar-10”. In: *arXiv preprint arXiv:1810.03505* (2018).

[10] Fan Dong, Henry Leung, and Steve Drew. “Navigating High-Degree Heterogeneity: Federated Learning in Aerial and Space Networks”. In: *arXiv preprint arXiv:2406.17951* (2024).

[11] Moming Duan et al. “Astraea: Self-balancing federated learning for improving classification accuracy of mobile deep learning applications”. In: *2019 IEEE 37th international conference on computer design (ICCD)*. IEEE. 2019, pp. 246–254.

[12] Haoyu Fang. “Challenges with the ultimate energy density with Li-ion batteries”. In: *IOP Conference Series: Earth and Environmental Science*. Vol. 781. 4. IOP Publishing. 2021, p. 042023.

[13] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. “Measuring the effects of non-identical data distribution for federated visual classification”. In: *arXiv preprint arXiv:1909.06335* (2019).

[14] Nathalie Japkowicz and Shaju Stephen. “The class imbalance problem: A systematic study”. In: *Intelligent data analysis* 6.5 (2002), pp. 429–449.

[15] Sai Praneeth Karimireddy et al. “Scaffold: Stochastic controlled averaging for federated learning”. In: *International conference on machine learning*. PMLR. 2020, pp. 5132–5143.

[16] Alex Krizhevsky, Geoffrey Hinton, et al. “Learning multiple layers of features from tiny images”. In: (2009).

[17] Daliang Li and Junpu Wang. “Fedmd: Heterogenous federated learning via model distillation”. In: *arXiv preprint arXiv:1910.03581* (2019).

[18] Qinbin Li, Bingsheng He, and Dawn Song. “Model-contrastive federated learning”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 10713–10722.

[19] Tian Li et al. “Federated optimization in heterogeneous networks”. In: *Proceedings of Machine learning and systems* 2 (2020), pp. 429–450.

[20] Xiang Li et al. “On the convergence of fedavg on non-iid data”. In: *arXiv preprint arXiv:1907.02189* (2019).

[21] Jiajia Liu et al. “Space-air-ground integrated network: A survey”. In: *IEEE Communications Surveys & Tutorials* 20.4 (2018), pp. 2714–2741.

[22] Xiaoyuan Liu et al. “Privacy-enhanced federated learning against poisoning adversaries”. In: *IEEE Transactions on Information Forensics and Security* 16 (2021), pp. 4574–4588.

[23] Brendan McMahan et al. “Communication-efficient learning of deep networks from decentralized data”. In: *Artificial intelligence and statistics*. PMLR. 2017, pp. 1273–1282.

[24] Takayuki Nishio and Ryo Yonetani. “Client selection for federated learning with heterogeneous resources in mobile edge”. In: *ICC 2019-2019 IEEE international conference on communications (ICC)*. IEEE. 2019, pp. 1–7.

[25] Amirhossein Reisizadeh et al. “Straggler-resilient federated learning: Leveraging the interplay between statistical accuracy and system heterogeneity”. In: *IEEE Journal on Selected Areas in Information Theory* 3.2 (2022), pp. 197–205.

[26] Hao Wang et al. “Optimizing federated learning on non-iid data with reinforcement learning”. In: *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE. 2020, pp. 1698–1707.

[27] Lixu Wang et al. “Addressing class imbalance in federated learning”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 11. 2021, pp. 10165–10173.

[28] Hongda Wu and Ping Wang. “Fast-convergent federated learning with adaptive weighting”. In: *IEEE Transactions on Cognitive Communications and Networking* 7.4 (2021), pp. 1078–1088.

[29] Jiajun Wu, Steve Drew, and Jiayu Zhou. “FedLE: Federated Learning Client Selection with Lifespan Extension for Edge IoT Networks”. In: *Proceedings of 2023 IEEE International Conference on Communications (ICC’23)*. 2023.

[30] Jiajun Wu et al. “Topology-aware federated learning in edge computing: A comprehensive survey”. In: *ACM Computing Surveys* 56.10 (2024), pp. 1–41.

[31] Han Xiao, Kashif Rasul, and Roland Vollgraf. “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms”. In: *arXiv preprint arXiv:1708.07747* (2017).

[32] Miao Yang et al. “Federated learning with class imbalance reduction”. In: *2021 29th European Signal Processing Conference (EUSIPCO)*. IEEE. 2021, pp. 2174–2178.

[33] Mang Ye et al. “Heterogeneous federated learning: State-of-the-art and research challenges”. In: *ACM Computing Surveys* 56.3 (2023), pp. 1–44.

[34] Rui Ye et al. “Fake It Till Make It: Federated Learning with Consensus-Oriented Generation”. In: *arXiv preprint arXiv:2312.05966* (2023).

[35] Tehrim Yoon et al. “Fedmix: Approximation of mixup under mean augmented federated learning”. In: *arXiv preprint arXiv:2107.00233* (2021).

[36] Liangkun Yu et al. “Latency-Aware Semi-Synchronous Client Selection and Model Aggregation for Wireless Federated Learning”. In: *Future Internet* 15.11 (2023), p. 352.

[37] Mikhail Yurochkin et al. “Bayesian nonparametric federated learning of neural networks”. In: *International conference on machine learning*. PMLR. 2019, pp. 7252–7261.

[38] Haobo Zhang et al. “A privacy-preserving hybrid federated learning framework for financial crime detection”. In: *arXiv preprint arXiv:2302.03654* (2023).

[39] Jianyi Zhang et al. “Fed-cbs: A heterogeneity-aware client sampling mechanism for federated learning via class-imbalance reduction”. In: *International Conference on Machine Learning*. PMLR. 2023, pp. 41354–41381.

[40] Yuru Zhang et al. “Aerial edge computing on orbit: A task offloading and allocation scheme”. In: *IEEE Transactions on Network Science and Engineering* 10.1 (2022), pp. 275–285.

Appendix A

Detailed Analysis of the Impacts of Battery Constraint to Label Heterogeneity

Heterogeneity, specifically statistical heterogeneity, is caused by the distribution discrepancy among participating clients. This discrepancy leads to an imbalance in the grouped dataset. It was noted in [39] that the imbalance of the grouped dataset in FL leads to the degradation of FL performance. A higher degree of heterogeneity leads to a higher imbalance degree within the grouped dataset. However, heterogeneity is not the only cause of the imbalance within the grouped dataset. The battery constraint can also exacerbate the class imbalance issue. The imbalance degree, denoted by Δ, is defined as: $\Delta = max(\mathcal{D}) - min(\mathcal{D})$, where $\mathcal{D}$ represents the distribution of the grouped dataset. For a B-class classification, the distribution is expressed as $\mathcal{D} = [\rho_1, \rho_2, \cdots, \rho_B]$.

Unlike traditional FL applications, devices in ASNs are often constrained by limited battery life. Given the current state of lithium battery technology [12], substantial improvements in energy density are unlikely in the near future. For example, a typical DJI drone can only fly for approximately 30 minutes. To prioritize task completion and safety, these devices must conserve energy. This limitation significantly

impacts client selection for FL training. On the one hand, it can reduce the number of clients participating in the FL training in each global epoch. On the other hand, it shrinks the pool size of clients with sufficient battery levels, as clients with low battery must prioritize essential operations like returning to base. In the standard FL setting used in this thesis, 10 clients are selected from a pool of 100. The following sections will explore the consequences of these limitations from both perspectives.

A.1 Selecting Fewer Clients in Each Global Epoch

In this section, a simulation was conducted to examine the effect of selecting different numbers of clients for participation in FL training on the imbalance degree of the grouped dataset. More than 10,000 experiments were performed and averaged to ensure robust results. As shown in Fig. 1.6, the imbalance degree increases as fewer clients are selected in each global epoch.

A.2 Selecting Clients From a Smaller Pool

Given that maintaining normal operations is the priority for aerial and space devices, clients with battery levels below a certain threshold should not be selected for FL training until they are recharged.

During the training process, a priority queue is maintained based on each client's battery percentage. Only clients with sufficient battery levels are selected to participate in FL training. Low-battery clients are recharged after some time and re-enter the queue, as shown in Fig. A.1. Under this setup, the selection pool for local training is temporarily limited to fewer clients, but once the low-battery clients are recharged, the pool is updated. To better understand the impact of imbalance, a more extensive study was conducted by combining grouped datasets across multiple global epochs. A hyperparameter, "window size," was introduced to adjust the observation window, combining datasets from consecutive epochs to calculate the imbalance degree. As illustrated in Fig. 1.7, a smaller pool size consistently leads to a higher degree of

Figure A.1: Clients are queued according to their respective battery percentages. Only clients with sufficient battery levels are selected in each global epoch.

imbalance within the grouped dataset, regardless of the observation window size.

The effect of varying window sizes on the imbalance degree, Δ, was further examined using pool sizes of 30, 50, and 70. As illustrated in Fig. A.2, a smaller device pool consistently increases the imbalance degree across different observation window sizes.

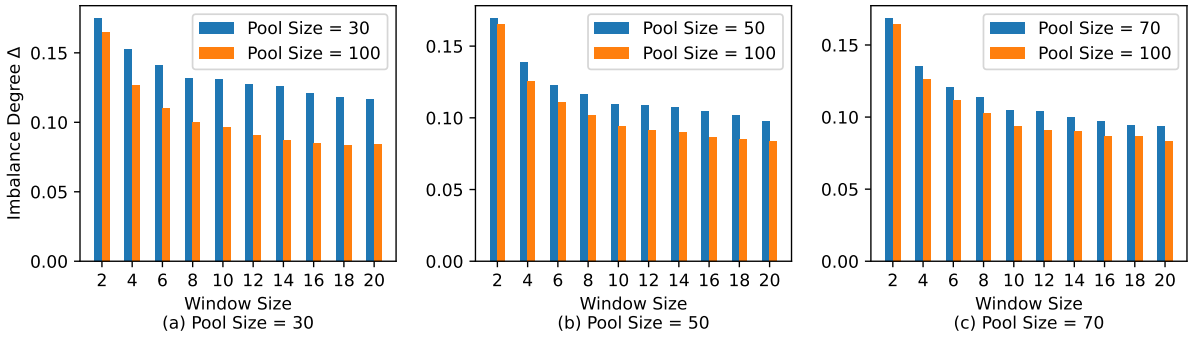


Figure A.2: Imbalance degree under different window sizes with different pool sizes.

The charging speed or the rate at which the client pool is updated also affects the imbalance degree. In

the previous analysis, a uniform update rate was used, where one client re-enters the pool in each global epoch. To assess the impact of different update rates, experiments were conducted with 0.2, 0.5, 1, 2, and 5 clients being updated per global epoch. This hyperparameter is referred to as ”step size.”. As shown in Fig. A.3, a smaller pool size consistently increases the imbalance degree of the grouped dataset across various observation window sizes and step sizes.

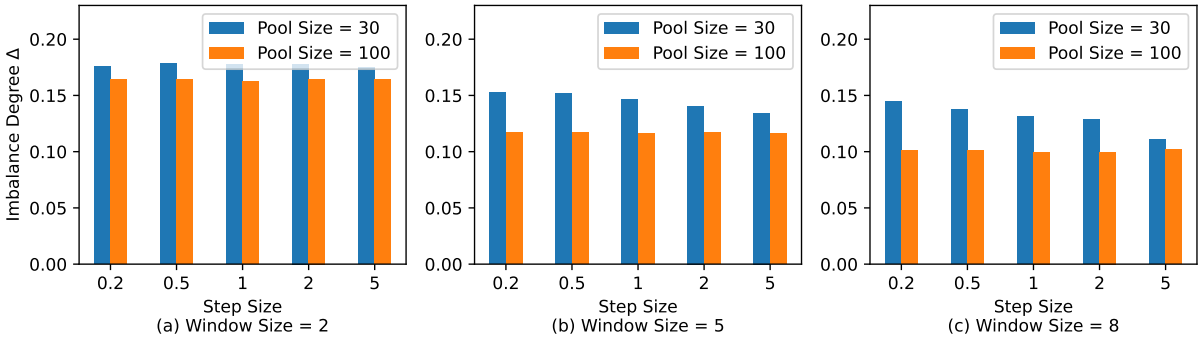


Figure A.3: Imbalance degree under different step sizes with different window sizes.

The analyses above confirm that the battery constraint of ASNs exacerbate the imbalance issue in FL, further intensifying the existing heterogeneity problem. While prior research has addressed the issue of label heterogeneity [15, 18, 7, 36], the distributions used are often not sufficiently heterogeneous. In the next section, the impacts of varying degrees of heterogeneity on FL were explored.

A.3 FedAvg Performance Under Different Heterogeneity Degrees

Since the battery constraint exacerbates class imbalance in FL applications for ASNs, a higher degree of heterogeneity should be considered in FL optimization to mitigate this effect. Fig. A.4 illustrates how the parameter α influences the level of heterogeneity. The visualization was generated through the following process: For each given α, 1,000 label distributions were randomly generated, sorted in descending order, and then averaged to produce a stable result. A shard-based distribution with two shards per client was also included for comparison. As shown in Fig. A.4, a smaller α results in a more heterogeneous distribution,

with the barplot becoming increasingly skewed, and the distribution becomes more imbalanced across all class labels. When α is set to 0.01, each client contains nearly only one class of samples. Note that the x-axis ticks represent the frequency order of labels rather than the specific labels.

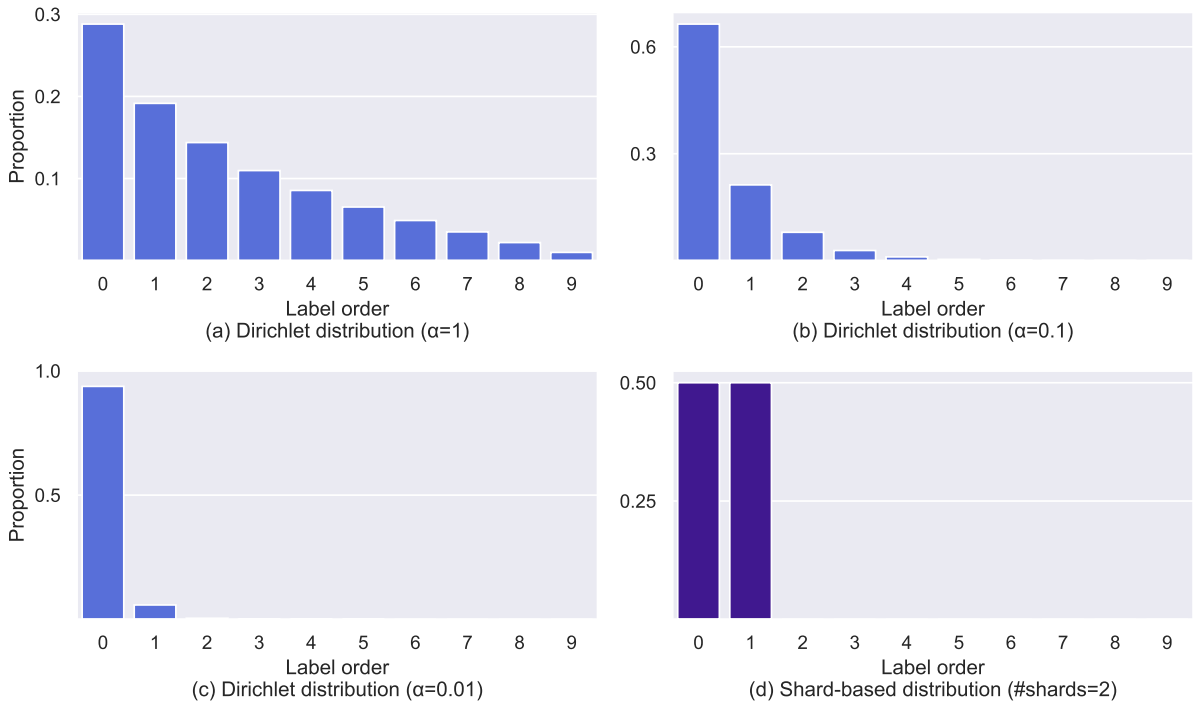


Figure A.4: Average distribution on each client with different settings.

Fig. A.5 shows the impact of different degrees of label heterogeneity on FL performance using the CIFAR-10 dataset. The Dirichlet distribution parameter α was set to 10^{15} to simulate a homogeneous distribution, where each class represents exactly 10% of the samples in a 10-class dataset. All experiments in Fig. A.5 were conducted using the FedAvg algorithm [23], with the only variable being the degree of label distribution heterogeneity. Multiple trials with different random seeds were averaged to produce smoother test accuracy curves. As shown in Fig. A.5, the test accuracy for the Dirichlet distribution with $\alpha = 1$ is very close to that of the homogeneous distribution. Although previous studies often used $\alpha = 0.1$ as an indicator of heterogeneity, this experiment demonstrates that $\alpha = 0.1$ is not heterogeneous enough. The shard-based distribution, with two shards per client, commonly used in other FL research [23], shows

that its heterogeneity degree falls between the Dirichlet distributions with $\alpha = 0.01$ and $\alpha = 0.1$. In this thesis, the Dirichlet distribution with $\alpha = 1$ is considered nearly homogeneous, $\alpha = 0.1$ represents low heterogeneity, and $\alpha = 0.01$ indicates high heterogeneity.

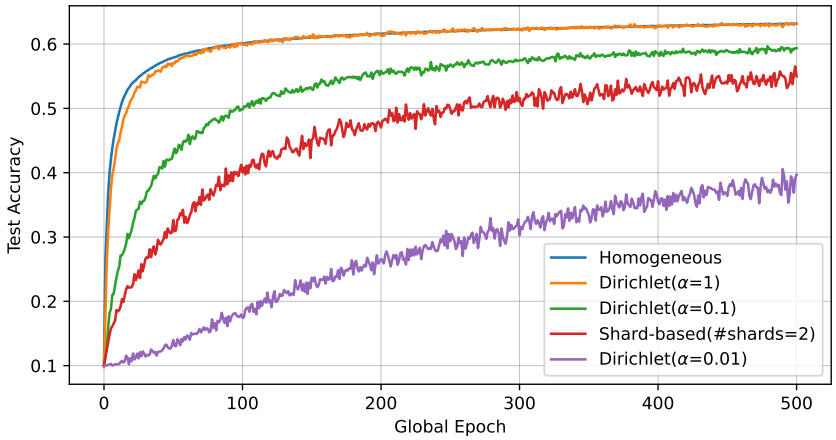


Figure A.5: Test accuracy lines of the FedAvg algorithm under different degrees of heterogeneity.

A.4 Performance of State-of-the-Art FL Algorithms Under Varying Degrees of Heterogeneity

Several state-of-the-art FL algorithms were evaluated across varying levels of data heterogeneity. As illustrated in Fig. A.6 and its corresponding numerical results in Table A.1, FedProx [19] demonstrated marginal performance gains compared to the baseline, FedAvg [23]. Moreover, MOON [18] was unable to outperform FedAvg, even in scenarios with low heterogeneity. While Scaffold [15] exhibited promising results in homogeneous and low-heterogeneity settings, it fails to outperform FedAvg under high heterogeneity.

Experimental results in Fig. A.6 and Table A.1 show that current state-of-the-art algorithms effectively enhance FL performance under low heterogeneity. However, the performance improvement under high-degree heterogeneity remains marginal. Some algorithms, such as Scaffold, fail to outperform Fe-

dAvg in highly heterogeneous distributions. This highlights the critical importance of optimizing FL for high-degree heterogeneity, as FL applications in ASNs are likely to encounter more severe heterogeneity challenges compared to other applications.

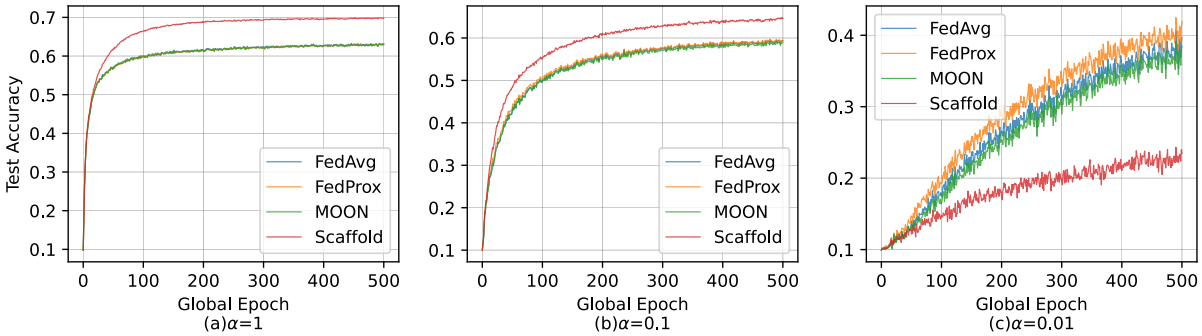


Figure A.6: Performance of algorithms on the CIFAR10 dataset under Dirichlet distribution with varying degrees of heterogeneity. Scaffold performs effectively under low heterogeneity ($\alpha \geq 0.1$) but fails to outperform FedAvg in highly heterogeneous settings.

Table A.1: Numerical experimental results for the CIFAR10 dataset under Dirichlet distribution with varying α values.

Algorithms	Dirichlet($\alpha = 1$)		Dirichlet($\alpha = 0.1$)		Dirichlet($\alpha = 0.01$)	
	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs
FedAvg	0.6315(0.0051)	380	0.5936(0.0248)	418	0.3966(0.0578)	490
FedProx	0.6310(0.0051)	450	0.5943(0.0218)	418	0.4193(0.0536)	417
MOON	0.6288(0.0073)	383	0.5896(0.0232)	-	0.3847(0.0710)	-
Scaffold	0.6977(0.0048)	59	0.6462(0.0153)	153	0.2346(0.0828)	-

Appendix B

More Experimental Results

Additional experimental results on the FashionMNIST dataset are presented in Fig. B.1 and Table B.1 under Dirichlet distribution, and with IR set to 5 and 10 in the shard-based distribution, as shown in Fig. B.2, Table B.2, Fig. B.3, and Table B.3. The corresponding results for the CINIC10 dataset are provided in Fig. B.4 and Table B.4, Fig. B.5 and Table B.5, and Fig. B.6 and Table B.6.

The conclusions are consistent with the experiments on the CIFAR-10 dataset: FedBalance and FedBalanceFilter perform well across all scenarios, particularly under high-degree heterogeneity, addressing the shortcomings of existing research in handling highly heterogeneous distributions.

B.1 Experimental Results on the FashionMNIST Dataset

Table B.1: Numerical experimental results for the FashionMNIST dataset under Dirichlet distribution with varying α values.

Algorithms	Dirichlet($\alpha = 1$)		Dirichlet($\alpha = 0.1$)		Dirichlet($\alpha = 0.01$)	
	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs
FedAvg	0.8851(0.0045)	88	0.8459(0.0239)	97	0.7174(0.0581)	97
FedProx	0.8851(0.0043)	84	0.8515(0.0205)	81	0.7276(0.0573)	79
MOON	0.8850(0.0045)	88	0.8458(0.0236)	97	0.6934(0.0624)	-
Scaffold	0.8922(0.0033)	46	0.8596(0.0162)	50	0.6665(0.1544)	-
FedBalance	0.8854(0.0039)	83	0.8524(0.0177)	70	0.7459(0.0509)	63
FedBalanceFilter	0.8867(0.0035)	69	0.8588(0.0149)	49	0.7649(0.0444)	50

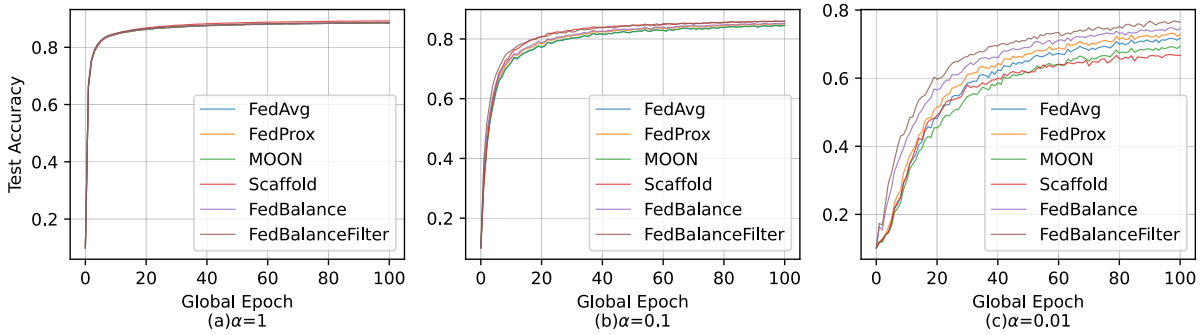


Figure B.1: Performance of algorithms on the FashionMNIST dataset under Dirichlet distribution with varying degrees of heterogeneity. Scaffold performs effectively under low heterogeneity ($\alpha \geq 0.1$) but fails to outperform FedAvg in highly heterogeneous settings. FedBalance and FedBalanceFilter outperform all other benchmarks in highly heterogeneous FL scenarios.

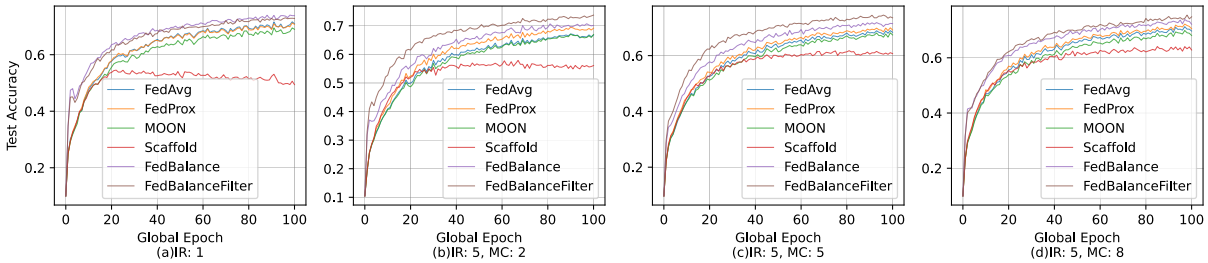


Figure B.2: Experiments on the FashionMNIST dataset with the same degree of heterogeneity but varying numbers of majority classes. Subfigure (a) displays results on a balanced global dataset for comparison, where $IR = 1$. IR represents the imbalance ratio between the number of samples in the majority class and the minority class, with a higher IR indicating a more imbalanced global dataset. MC denotes the number of majority classes, while the number of minority classes in the 10-class classification is $10-MC$.

Table B.2: Numerical experimental results for the FashionMNIST dataset with an imbalance ratio (IR) set to 5.

Algorithms	IR = 1		IR = 5					
			MC = 2		MC = 5		MC = 8	
	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs
FedAvg	0.7094(0.0703)	91	0.6681(0.1094)	88	0.6837(0.0945)	89	0.6968(0.0786)	85
FedProx	0.7073(0.1031)	91	0.6916(0.0661)	64	0.6919(0.0953)	77	0.7052(0.0758)	69
MOON	0.6898(0.0950)	-	0.6695(0.0720)	90	0.6741(0.0896)	-	0.6808(0.0827)	97
Scaffold	0.4940(0.2756)	-	0.5610(0.2426)	-	0.6071(0.2115)	-	0.6283(0.1957)	-
FedBalance	0.7382(0.0739)	53	0.7013(0.0812)	53	0.7147(0.0903)	55	0.7215(0.0741)	47
FedBalanceFilter	0.7283(0.0886)	68	0.7374(0.0876)	32	0.7354(0.0809)	37	0.7484(0.0550)	42

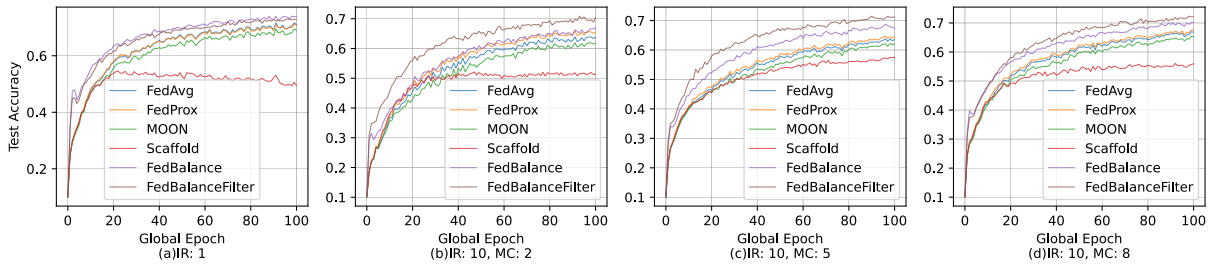


Figure B.3: Experiments on the FashionMNIST dataset with the same degree of heterogeneity but varying numbers of majority classes. Subfigure (a) displays results on a balanced global dataset for comparison, where $IR = 1$. IR represents the imbalance ratio between the number of samples in the majority class and the minority class, with a higher IR indicating a more imbalanced global dataset. MC denotes the number of majority classes, while the number of minority classes in the 10-class classification is $10 - MC$.

Table B.3: Numerical experimental results for the FashionMNIST dataset with an imbalance ratio (IR) set to 10.

Algorithms	IR = 1		IR = 10					
			MC = 2		MC = 5		MC = 8	
	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs
FedAvg	0.7094(0.0703)	91	0.6379(0.0943)	91	0.6354(0.0949)	96	0.6684(0.0791)	99
FedProx	0.7073(0.1031)	91	0.6558(0.0800)	69	0.6443(0.0998)	89	0.6774(0.0875)	90
MOON	0.6898(0.0950)	-	0.6178(0.0944)	-	0.6205(0.0889)	-	0.6524(0.0810)	-
Scaffold	0.4940(0.2756)	-	0.5137(0.2431)	-	0.5759(0.1842)	-	0.5593(0.2156)	-
FedBalance	0.7382(0.0739)	53	0.6710(0.1049)	67	0.6733(0.0715)	56	0.7026(0.0863)	60
FedBalanceFilter	0.7283(0.0886)	68	0.7030(0.1152)	38	0.7122(0.0645)	36	0.7209(0.0613)	45

B.2 Experimental Results on the CINIC10 Dataset

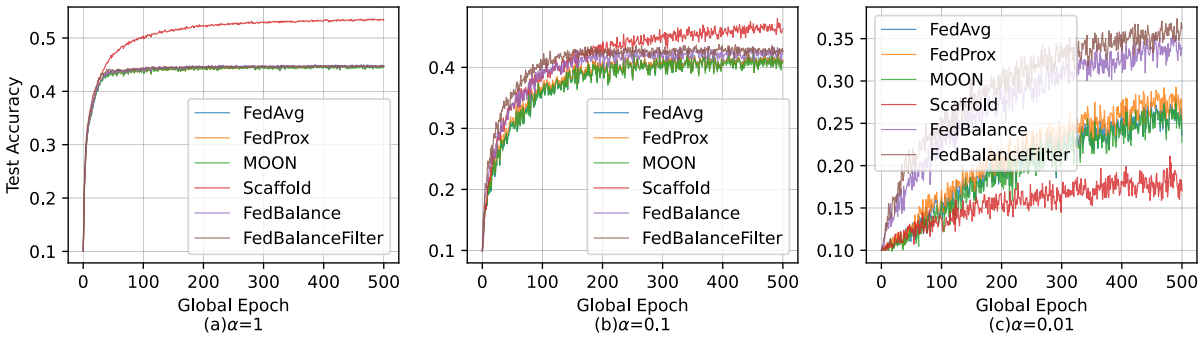


Figure B.4: Performance of algorithms on the CINIC10 dataset under Dirichlet distribution with varying degrees of heterogeneity. Scaffold performs effectively under low heterogeneity ($\alpha \geq 0.1$) but fails to outperform FedAvg in highly heterogeneous settings. FedBalance and FedBalanceFilter outperform all other benchmarks in highly heterogeneous FL scenarios.

Table B.4: Numerical experimental results for the CINIC10 dataset under Dirichlet distribution with varying α values.

Algorithms	Dirichlet($\alpha = 1$)		Dirichlet($\alpha = 0.1$)		Dirichlet($\alpha = 0.01$)	
	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs
FedAvg	0.4466(0.0053)	136	0.4108(0.0253)	176	0.2362(0.0593)	266
FedProx	0.4468(0.0071)	209	0.4112(0.0221)	159	0.2539(0.0585)	243
MOON	0.4447(0.0054)	209	0.4083(0.0238)	235	0.2277(0.0515)	313
Scaffold	0.5330(0.0056)	36	0.4626(0.0221)	123	0.1700(0.0467)	-
FedBalance	0.4485(0.0055)	116	0.4265(0.0172)	136	0.3387(0.0490)	84
FedBalanceFilter	0.4469(0.0045)	114	0.4262(0.0165)	97	0.3622(0.0334)	74

Table B.5: Numerical experimental results for the CINIC10 dataset with an imbalance ratio (IR) set to 5.

Algorithms	IR = 1		IR = 5					
			MC = 2		MC = 5		MC = 8	
	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs
FedAvg	0.3950(0.0390)	262	0.3611(0.0646)	479	0.3614(0.0579)	264	0.3772(0.0545)	247
FedProx	0.3973(0.0388)	262	0.3635(0.0651)	343	0.3670(0.0561)	223	0.3812(0.0559)	223
MOON	0.3944(0.0425)	262	0.3572(0.0635)	479	0.3613(0.0583)	288	0.3738(0.0506)	247
Scaffold	0.3520(0.0737)	-	0.3309(0.0814)	-	0.3516(0.0493)	297	0.3572(0.0497)	437
FedBalance	0.4303(0.0402)	126	0.4186(0.0364)	147	0.3917(0.0526)	130	0.4103(0.0504)	132
FedBalanceFilter	0.4287(0.0504)	113	0.4254(0.0423)	79	0.4050(0.0486)	97	0.4278(0.0306)	117

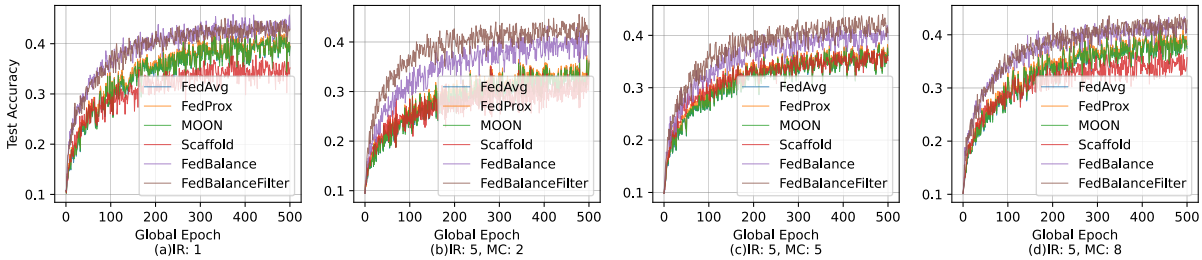


Figure B.5: Experiments on the CINIC10 dataset with the same degree of heterogeneity but varying numbers of majority classes. Subfigure (a) displays results on a balanced global dataset for comparison, where $IR = 1$. IR represents the imbalance ratio between the number of samples in the majority class and the minority class, with a higher IR indicating a more imbalanced global dataset. MC denotes the number of majority classes, while the number of minority classes in the 10-class classification is $10 - MC$.

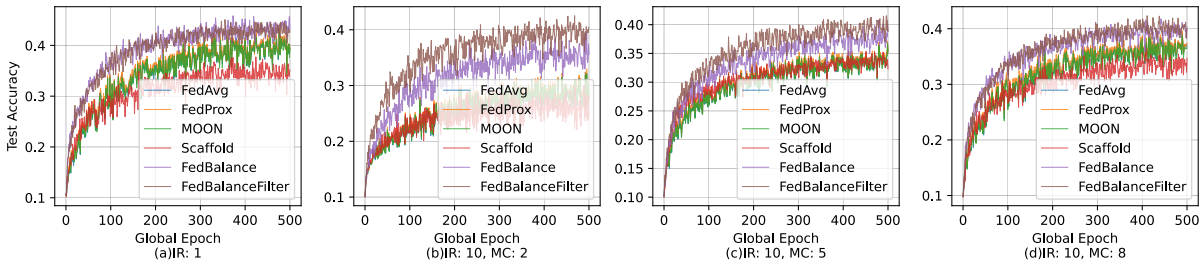


Figure B.6: Experiments on the CINIC10 dataset with the same degree of heterogeneity but varying numbers of majority classes. Subfigure (a) displays results on a balanced global dataset for comparison, where $IR = 1$. IR represents the imbalance ratio between the number of samples in the majority class and the minority class, with a higher IR indicating a more imbalanced global dataset. MC denotes the number of majority classes, while the number of minority classes in the 10-class classification is $10 - MC$.

Table B.6: Numerical experimental results for the CINIC10 dataset with an imbalance ratio (IR) set to 10.

Algorithms	IR = 1		IR = 10					
			MC = 2		MC = 5		MC = 8	
	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs	TestAcc(std)	Epochs
FedAvg	0.3950(0.0390)	262	0.2747(0.0491)	189	0.3532(0.0456)	412	0.3679(0.0357)	295
FedProx	0.3973(0.0388)	262	0.2813(0.0515)	123	0.3535(0.0443)	302	0.3739(0.0325)	231
MOON	0.3944(0.0425)	262	0.2739(0.0507)	123	0.3513(0.0442)	412	0.3648(0.0357)	307
Scaffold	0.3520(0.0737)	-	0.2429(0.0315)	210	0.3232(0.0456)	412	0.3468(0.0364)	-
FedBalance	0.4303(0.0402)	126	0.3556(0.0762)	76	0.3885(0.0393)	156	0.4037(0.0277)	141
FedBalanceFilter	0.4287(0.0504)	113	0.4042(0.0581)	37	0.3879(0.0405)	128	0.3987(0.0542)	136