FEDERATED LEARNING WITH ACCURATE MODEL TRAINING AND LOW COMMUNICATION COST IN HETEROGENEOUS SCENARIOS

Lucas Airam Castro de Souza

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientadores: Miguel Elias Mitre Campista
Luís Henrique Maciel Kosmalski Costa

Rio de Janeiro
Setembro de 2023

FEDERATED LEARNING WITH ACCURATE MODEL TRAINING AND LOW COMMUNICATION COST IN HETEROGENEOUS SCENARIOS

Lucas Airam Castro de Souza

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Orientadores: Miguel Elias Mitre Campista
Luís Henrique Maciel Kosmalski Costa

Aprovada por: Prof. Marcelo Gonçalves Rubinstein
Prof. Diogo Menezes Ferrazani Mattos
Prof. Miguel Elias M. Campista
Prof. Luís Henrique M. K. Costa

RIO DE JANEIRO, RJ – BRASIL
SETEMBRO DE 2023

de Souza, Lucas Airam Castro

Federated Learning with Accurate Model Training and Low Communication Cost in Heterogeneous Scenarios/Lucas Airam Castro de Souza. – Rio de Janeiro: UFRJ/COPPE, 2023.

XVI, 51 p.: il.; 29, 7cm.

Orientadores: Miguel Elias Mitre Campista

Luís Henrique Maciel Kosmalski Costa

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2023.

Referências Bibliográficas: p. 44 – 48.

1. federated learning. 2. non-IID data. 3. privacy. I. Campista, Miguel Elias Mitre *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

Aos meus pais e amigos.

Agradecimentos

Primeiramente agradeço a Deus, por todas as oportunidades que tive e colocar ao meu lado todas as pessoas que fizeram parte deste trabalho de forma direta ou indireta. Agradeço aos meus pais, Altair Tadeu e Maria Aparecida, por sempre incentivarem e se sacrificarem pela minha formação acadêmica. Sem eles, nada disto seria possível. Ao meu tio João Bosco e minha amiga Patrícia D'Icarahy, por me incentivarem e estarem presentes nas minhas conquistas acadêmicas. À minha companheira Isamara Cristina que sempre esteve ao meu lado e me apoiou em todos os momentos.

Aos professores Miguel Elias Mitre Campista e Luís Henrique Maciel Kosmalski Costa por orientarem essa dissertação de mestrado e o trabalho desenvolvido ao longo dos últimos 2 anos. Sem eles este trabalho não seria possível. Ao professor Otto Duarte, que me orientou por aproximadamente 4 anos e despertou meu interesse para a pesquisa científica. Sua orientação para o meu crescimento pessoal e profissional estarão presentes em mim.

A todos os meus companheiros do Grupo de Teleinformática e Automação (GTA), que me proporcionaram o maior crescimento pessoal e profissional que já tive através de seus conselhos. Estes são os grandes responsáveis pela minha formação profissional e pela qualidade desta dissertação. Em especial ao Gustavo Franco Camilo e Gabriel Antonio Fontes Rebello, os quais participaram comigo em diversas publicações e projetos de pesquisa.

Agradeço aos professores Diogo Menezes Ferrazani Mattos e Marcelo Gonçalves Rubinstein por aceitarem participar da banca de avaliação desta dissertação. Ao CNPq, CAPES, FAPERJ, FAPESP e RNP por viabilizarem este trabalho e fomentar continuamente a pesquisa de alto nível no Brasil.

Este trabalho é uma maneira de retribuir ao povo brasileiro o investimento na educação pública de qualidade e a confiança em mim depositada.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

APRENDIZADO FEDERADO COM TREINAMENTO DE MODELOS ACURADOS E BAIXO CUSTO DE COMUNICAÇÃO EM CENÁRIOS HETEROGÊNEOS.

Lucas Airam Castro de Souza

Setembro/2023

Orientadores: Miguel Elias Mitre Campista
Luís Henrique Maciel Kosmalski Costa

Programa: Engenharia Elétrica

O aprendizado federado (*Federated Learning* - FL) é uma abordagem descentralizada para treinar modelos de aprendizado de máquina sem divulgar dados privados dos clientes participantes para um servidor. No entanto, o desempenho do FL depende da distribuição dos dados, e o treinamento possui dificuldades para convergir quando os clientes têm distribuições de dados distintas, aumentando o tempo geral de treinamento e o erro de previsão do modelo final. Este trabalho propõe duas estratégias para reduzir o impacto da heterogeneidade de dados em cenários de FL. Primeiramente, é proposto um sistema hierárquico de agrupamento de clientes para contornar os obstáculos de convergência em cenários com dados não independentes e identicamente distribuídos (IID). A proposta identifica os grupos usando apenas um modelo de teste e o vetor de rede neural de pesos da última camada.Os resultados mostram que o sistema tem um desempenho de classificação melhor que o FedAVG, aumentando a precisão em aproximadamente 16% em cenários não-IID. Além disso, a primeira proposta é estendida ao implementar o ATHENA-FL, um sistema de aprendizado federado que compartilha conhecimento entre diferentes grupos. O sistema usa o modelo um-contra-todos para treinar um detector binário para cada classe no grupo. Assim, os clientes podem compor modelos complexos combinando múltiplos detectores. Os resultados mostram que o ATHENA-FL identifica corretamente as amostras, alcançando uma precisão até 10,9% maior do que o treinamento tradicional. Por fim, o ATHENA-FL atinge custos de comunicação de treinamento mais baixos do que a arquitetura MobileNet, reduzindo o número de bytes transmitidos entre 25% e 97% nos cenários avaliados.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

FEDERATED LEARNING WITH ACCURATE MODEL TRAINING AND LOW COMMUNICATION COST IN HETEROGENEOUS SCENARIOS

Lucas Airam Castro de Souza

September/2023

Advisors: Miguel Elias Mitre Campista
Luís Henrique Maciel Kosmalski Costa

Department: Electrical Engineering

Federated learning (FL) is a distributed approach to train machine learning models without disclosing private data from participating clients to a central server. Nevertheless, FL performance depends on the data distribution, and the training struggles to converge when clients have distinct data distributions, increasing overall training time and the final model prediction error. This work proposes two strategies to reduce the impact of data heterogeneity in FL scenarios. Firstly, we propose a hierarchical client clustering system to mitigate the convergence obstacles of federated learning in non-Independent and Identically Distributed (IID) scenarios. Our proposal identifies the clusters using only a test model and the clients' last layer weights neural network vector for a few training epochs for this model. The results show that our system has a better classification performance than FedAVG, increasing its accuracy by approximately 16% on non-IID scenarios. Furthermore, we improve our first proposal by implementing ATHENA-FL, a federated learning system that shares knowledge among different clusters. The proposed system also uses the one-*versus*-all model to train one binary detector for each class in the cluster. Thus, clients can compose complex models combining multiple detectors. Consequently, ATHENA-FL can also reduce communication costs, providing an additional positive aspect for resource-constrained scenarios. ATHENA-FL mitigates data heterogeneity by maintaining the clustering step before training to mitigate data heterogeneity. Our results show that ATHENA-FL correctly identifies samples, achieving up to 10.9% higher accuracy than traditional training. Finally, ATHENA-FL achieves lower training communication costs than MobileNet architecture, reducing the number of transmitted bytes between 25% and 97% across evaluated scenarios.

ACRONYMS

ATHENA-FL - Avoiding sTatistical HEterogeiNety with one-*versus*-All in Federated Learning

CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior

CCPA - California Consumer Privacy Act

CEFL - Communication-Efficient Federated Learning

CFL - Clustered Federated Learning

CIFAR - Canadian Institute For Advanced Research

CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico

COPPE - Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa em Engenharia

DBSCAN - Density-Based Spatial Clustering of Applications with Noise

DCS - Decomposed Cosine Similarity

FAPERJ - Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro

FAPESP - Fundação de Amparo à Pesquisa do Estado de São Paulo

FedAVG - Federated Averaging

FedOVA - Federated OvA

FedTP - Federated learning by Transformer Personalization

FL - Federated Learning

FLEE - Federated Learning Early Exit of inference

FlexCFL - Flexible Clustered Federated Learning

FMNIST - Fashion MNIST

GDPR - General Data Protection Regulations

GTA - Grupo de Teleinformática e Automação

Hier-SFL - Hierarchical Split Federated Learning

IFCA - Iterative Federated Clustering Algorithm

IID - Independent and Identically Distributed

LDA - Label-based Dirichlet Partition

MNIST - Modified National Institute of Standards and Technology

OPTICS - Ordering Points to Identify the Clustering Structure

OvA - One-*versus*-all

PEE - Programa de Engenharia Elétrica

RAM - Random Access Memory

RNP - Rede Nacional de Ensino e Pesquisa

StoCFL - Stochastic Clustered Federated Learning

TDT - Total Data Transmitted

UFRJ - Universidade Federal do Rio de Janeiro

MATHEMATICAL NOTATION

$B(k)$ - binarization function

B_{mcc} - amount of bytes transmitted to train the multiclass classifier

B_{ova} - amount of bytes transmitted to train OvA detectors

c - number of clusters in the K-Means algorithm

C_{ova} - One-*versus*-All model

e_{dec} - number of epochs for detector's convergence

e_{mcc} - number of epochs for multiclass classification model convergence

E_q - cluster's current epoch

F_{k_n} - local objective for the n-th client to the k-th detector

$F_n(w)$ - local objective function for the n-th client

$f_n(w)$ - loss function of the model for the n-th client

g_j - j-th system's cluster

G - total number of clusters

i - sample's index

j - cluster's index

k - detector's index

$l(x_i, y_i; w)$ - loss function for the prediction (x_i, y_i) with model parameter's w

n - client's index

p_{k_n} - selection probability of the n-th client during the k-th detector's training

p_n - client's selection probability

r_k - OvA detector

R - total number of detectors

s_n - size of the n-th client dataset

s - number of samples among all clients

T_{cmc} - multiclass classifier's size in bytes

T_{dec} - detector's size in bytes

tol - tolerance to estimate model convergence

V^a - model's parameter search space

w_k - detector's model parameter

w - multiclass classifier model parameter

x_i - features of the i-th sample

y_i - true class of the i-th sample

$\hat{y}_i$ - estimated class of the i-th sample

z_t - cluster's multiclass classification model

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Machine learning enables task automation by creating models that identify patterns in datasets to predict or classify future data. In traditional machine learning systems, model training requires client-data collection, which usually reveals private or sensitive information from the user or collection point [1]. Also, the high data volume generated by devices imposes a challenge to this practice of centralizing information into a single point in the network. Therefore, Federated Learning (FL) has emerged as a proposal for training machine learning models that preserve the privacy of the user without sharing local data. Federated learning (FL), proposed by Google [2], has become popular among researchers and industry due to the possibility of training machine learning models while preserving users' data privacy [3–6]. After the change in data processing regulations in several countries, for instance, the California Consumer Privacy Act (CCPA) in the USA and the General Data Protection Regulations (GDPR) in Europe, the importance of federated learning research and adoption increased.

Training in federated learning replaces data sharing with model parameter sharing. In Federated Averaging (FedAVG), the most widely used algorithm for model parameters' aggregation in FL, clients train the model locally for a few epochs and send the results to the aggregation server, which combines the individual trained models into a single global model. Then, the aggregation server broadcasts the global model back to the clients, which further improve it. The aggregation server and clients repeat this process until the global model converges or the training reaches the maximum global epochs, sending only the models' parameters. Thus, clients' training samples remain stored locally, preserving data and user privacy. Nevertheless, another problem persists when deploying FL on a large scale: clients may have heterogeneous data generated from non-independent and identically distributed (non-IID) distributions, leading to convergence difficulty and suboptimal performance [7, 8]. Therefore, a proposal that reduces the effects of data heterogeneity during model training and allows the generalization of the classifier to samples

originating from other data distributions becomes necessary. It can be achieved by clustering clients comparing data similarity, which allows models to be trained on IID data and quickly converge with high final classification performance [9].

1.1 Work Contributions

This work proposes a hierarchical client clustering system[1] to increase the efficiency of federated learning in scenarios where clients have non-IID datasets. Clients, also called nodes, are divided into clusters where the data are similar. The proposal uses clients' last-layer neural network weights as input to perform clustering. The last-layer neural network weights maintain statistical relationships with the clients' private data without revealing them [10]. Thus, we maintain the same privacy model as FedAVG. Firstly, each cluster trains a personalized model with independent hyperparameters and parameters, allowing high classification performance on specific tasks. We also investigate how to tune the clustering algorithms' hyperparameters to best fit our scenario. Then, we extend the first proposal by providing a mechanism to combine models created on different clusters, which we call ATHENA-FL (Avoiding sTatistical HEterogeiNety with one-*versus*-All in Federated Learning)[2], a system that creates models with the one-*versus*-all (OvA) technique to enable model sharing between groups. The one-*versus*-all model uses independently trained binary classifiers, which estimate the probability that a sample belongs to the class identified by a detector. After training, the detectors are combined for sample classification. Each detector estimates the probability that the sample belongs to its class, and the classifier labels the sample from the detector that generates the highest probability. Thus, the OvA method is used for efficient model sharing between groups to create a generic model for classifying data from different groups.

We analyze different clustering models and show the advantage of adopting the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [11] algorithm for client clustering. DBSCAN can identify homogeneous and heterogeneous data distributions from the client neural network weights and outperforms the Ordering Points to Identify the Clustering Structure (OPTICS) [12] and K-Means [13] algorithms. Furthermore, in the worst case, our results show that the proposed approach performs equally to traditional federated learning. For non-IID datasets, the FedAVG algorithm converges with low accuracy after 100 global epochs. Nonetheless, the current proposal achieves better results across all clusters in 10 global epochs, providing higher classification performance. Then, we evaluate the ATHENA-FL proposal according to the accuracy evolution of the detectors.

[1] Available at https://github.com/GTA-UFRJ/Hierachical-Federated-Learning
[2] Available at `https://github.com/GTA-UFRJ/ATHENA-FL`

Also, the evaluation compares the classification performance of the global model and communication requirements for training the models. The accuracy of the models depends on the diversity of the samples used for training the detectors, with the accuracy of the OvA model up to 10.9% higher than the MobileNet architecture [14]. At the same time, the amount of bytes transmitted over all training epochs is reduced by up to 97.37% using the one-*versus*-all model instead of MobileNet. Thus, the system provides an effective way to train models that maintain the privacy of client data in scenarios with heterogeneous distributions.

We summarize our contributions as follows:

- **High classification performance**: We propose a system that achieves high accuracy results under non-IID data distributions. Our proposal obtains these results by assigning clients to a cluster where the data is more homogeneous.
- **Low-communication requirement**: The proposal uses clients' last-layer neural network weights as the input to perform client clustering. Thus, we avoid the communication overhead of recursive clustering proposals. Also, we reduce the amount of data used to cluster clients compared to a proposal that sends a whole model.
- **Privacy-preserving**: Our proposal holds the same privacy model and assumptions as FedAVG, given that the system shares only the model weights.
- **Robust-classifier**: The OvA model is used to combine the classifiers trained among different clusters, which offers a classifier that is able to identify samples generated in different clusters.
- **Prototype**: We implement a proof-of-concept of our proposed system. We also present a practical evaluation of FedAVG under non-IID data distributions and compare it with our proposal.

This master thesis is a set of the following publications:

- de Souza, L. A. C., Camilo, G. F., Sammarco, M., Campista, M. E. M., Costa, L. H. M. - "Aprendizado Federado com Agrupamento Hierárquico de Clientes para Aumento da Acurácia". In Anais do XL Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (pp. 545-558). (2022, May). SBC.
- de Souza, L. A. C., Camilo, G. F., Rebello, G. A. F., Sammarco, M., Campista, M. E. M., Costa, L. H. M. - "ATHENA-FL: Evitando a Heterogeneidade Estatística através do Um-contra-Todos no Aprendizado Federado". In Anais do VII Workshop de Computação Urbana (pp. 40-53). (2023, May). SBC. **Honarable Mention**.

We also have submitted two papers for an international conference and a journal that are currently under review. Other publications produced during the master's research are listed in Appendix A.

1.2 Thesis Outline

This work is organized as follows. Chapter 2 reviews the state-of-the-art proposals to increase the classification performance of federated learning systems and reduce the impacts caused by data heterogeneity. Chapter 3 presents basic concepts about the federated learning scenario, which are useful for understanding the proposal. Chapter 4 describes our clustering proposal by formulating the problem and giving details about the system's architecture and protocols while Chapter 5 discusses the use of one-*versus*-all to combine models and share information among clusters. Chapter 6 presents the development of a prototype, evaluation, and analysis of the obtained results. Chapter 7 concludes this work and discusses future research directions. Besides, Appendix A shows the list of publications produced during the master course.

Chapter 2

Related Work

Currently, one of the most relevant challenges regarding federated learning is increasing the final model performance while reducing the total training time. The time to convergence of the global model in federated learning is attributed to three main factors: communication latency, processing capacity, and data representativeness. The first two factors are related to the user device, while the third depends on the data distribution collected and stored by the client. Devices with more computational power and better connectivity usually execute local epochs faster, whilst users with more data that truly represent the problem can reduce the time to global model convergence. Nonetheless, the initial federated learning proposal ignores these three relevant characteristics and considers a uniform distribution for the probability that a client participates in training [2]. Therefore, several research groups present proposals to reduce the convergence time of federated learning and improve the final performance of the generated model. Although hyperparameter optimization is a promising alternative to FL scenarios [15], client selection [16] or local model personalization [17] could also achieve faster convergence and higher accuracy. The client selection modifies the likelihood function to reflect the client's ability to significantly contribute to model training in a global epoch, relying on devices' characteristics or clients' data estimation. The goal is to decide the best subset of clients for the current training epoch, increasing the average model performance. Besides, model personalization allows the training of models specialized in clients' characteristics, consequently improving local model performance. Finally, another strategy of model personalization is clustered FL, where the idea is to divide clients into clusters to train models with similar data and achieve higher accuracy.

2.1 Client Selection for Efficient Training

Luo *et al.* [18] and Lai *et al.* [19] propose client selection schemes that optimize the time to convergence in federated learning environments. The authors argue that se-

lection based only on the representativeness of the data decreases the total number of epochs for model convergence. However, when clients with higher computational latency participate, the delay between epochs increases significantly. Nevertheless, a selection based only on processing capacity may incur more epochs for convergence if selected clients have irrelevant data. On the one hand, increasing the time between epochs implies a higher overall delay, as federated learning environments are generally synchronous and wait for all clients to respond or time out. On the other hand, selecting clients with higher computational power to reduce the time between epochs may incur more epochs for convergence if the selected clients have statistically insignificant data for the learning task. There is a trade-off between the number of epochs for model training and the total time for each epoch. Thus, the authors simultaneously consider the devices' characteristics and their data distribution relevance to reduce the convergence time of the global model. While Luo *et al.* propose a non-convex function regarding the expected number of epochs based on clients' previous updates, Lai *et al.* avoids the linear programming, creating a selection function that evaluates clients' training losses.

Wang *et al.* propose to apply reinforcement learning for client selection [10]. The reinforcement learning agent is previously trained to select and return the best clients for each global training epoch given the current global model state and clients' information. The system estimates clients' statistical relevance through the loss gradient vector sent to the aggregation server. This procedure maintains the privacy of the client's data. Fu *et al.* present a survey of client selection systems and frameworks in federated learning [16]. Their paper presents the state-of-the-art in client selection and compares their main differences, demonstrating that client selection has great potential to produce more accurate models in federated learning with less training time.

FedProx is a framework to reduce the effects of statistical and system heterogeneity [20]. The difference between FedProx and FedAVG is the flexible amount of work the clients perform. While FedAVG distributes an equal amount of work to all clients, FedProx considers individual hardware constraints to assign work to clients proportionally. Also, the authors propose inserting a regularization term on the local objective function to minimize the impact of non-IID data in the federated network.

Nishio and Yonetani propose a protocol for client selection in federated learning [21] in which clients with higher processing capabilities and lower communication latency are prioritized in the presented selection scheme. Liu *et al.* propose a hierarchical architecture for federated learning [22], which performs local aggregations on clients, partial aggregation at the network edges, and global convergence in the cloud. Processing data at the edge decreases communication latency, but the re-

stricted number of devices can make it difficult for the global model to converge. Cloud processing, however, can access more devices and capture more data variance at the cost of higher communication latency. Thus, the authors apply federated training on levels: client-edge and edge-cloud. Clients maintain low latency for communication with the edge, and the cloud can access the models generated at the edges. Hierarchical Split Federated Learning (Hier-SFL) [23] is an FL framework that divides the neural network into three different levels, client-edge-cloud, to train the model. Thus, each one of the neural network parts has an associated loss function, which is adjusted separately. Hier-SFL objective is to reduce the communication overhead during the training. However, the distribution of data generated by clients is not taken into account, which can affect the performance of models in scenarios with non-IID distributions.

Rai *et al.* propose the irrelevance sampling metric for client selection to improve the final accuracy of federated learning models in IID and non-IID scenarios [24]. The objective is to select clients considering the quality and quantity of their samples. Each client computes its irrelevance sampling metric and sends it to the server. The server then clusters the clients according to the informed value in the following three clusters: positive, negative, and zero. Finally, the proposed methodology randomly selects the clients of each cluster to achieve faster model convergence.

Fraboni *et al.* propose a clustering sampling method for client selection [25]. The authors provide two approaches for clustering clients: client sample size and model similarity. Nevertheless, the first approach highly depends on the clients' informing their exact sample size to the aggregation server for cluster definition. Therefore, this approach is susceptible to clients' malicious behaviors. On the other hand, the method uses all model weights in the model similarity proposal, which introduces communication overhead.

The selection of clients decreases the time to convergence and increases the final accuracy of the model. The client selection, however, is insufficient for practical purposes in environments with heterogeneous data. Thus, an alternative is model personalization, which fine-tunes clients' models according to their local data and improves their final performance.

2.2 Personalized Models in Federated Learning

Federated learning by Transformer Personalization (FedTP) is a framework for reducing data heterogeneity by transforming datasets and personalizing models [26]. FedTP objective is to define the basis of a transformation. In this transformation, the client data is similar, and they can personalize their models in some layers, overcoming the convergence problems. However, the proposal generates significant

communication overhead, requiring a long time to adjust the projection parameters and for the model to converge.

Federated Learning Early Exit of inference (FLEE) is a hierarchical-federated learning framework that splits the model into three different geolocations [27]. The model's division between the cloud, edge, and end device allows using the method of early exit inference from neural networks. Furthermore, FLEE's hierarchical division of the training reduces the impact caused by non-IID data distributions on the model convergence, as the authors assume that the data have higher similarity according to the geographic distance of the clients.

Ditto [28] is a framework for personalized federated learning. The authors separate the optimization problem into two parts: global optimization and local optimization. Global optimization considers the contribution of all clients to the model, while local optimization adjusts the client's model according to the local data. Also, the authors provide two definitions: robustness and fairness in federated learning environments. Robustness expresses the model's capacity to achieve high accuracy even in heterogeneous data or data poisoning attacks. Fairness represents how different clients' performance is when using local test data.

Dennis *et al.* [29] take advantage of data heterogeneity to produce an unsupervised federated learning algorithm. The clients train a clustering model with their local data and send the vectors that indicate the position of the clusters found in the sample space to an aggregation server. The server then runs a second clustering model based on clients' responses to identify the clusters in the environment. The greater the distance between client distributions, the more effective the detection of different clusters. The proposal has an application for both the selection of clients and the personalization of models.

Zhu *et al.* propose using one-*versus*-all classification scheme to mitigate the impact of heterogeneous data in training federated learning models [30]. Their Federated OvA (FedOVA) algorithm uses models for binary classification and selects clients having samples of the target class to perform the training. Nevertheless, their proposal lacks a study on the impact of the model creation and an adequate protocol for detector training. ATHENA-FL clusters clients according to data distributions, before training the OvA model, thus reducing detector training time.

2.3 Clustered Federated Learning

Clustered federated learning is a subfield of model personalization in federated learning research. This subfield focuses on creating strategies to arrange FL clients into groups, reducing data heterogeneity. The proposals are mostly differentiated by clustering strategy, metrics, and training.

Communication-Efficient Federated Learning (CEFL) is an FL-based framework for medical-data model training [31]. The authors determine the similarity between clients by calculating the Euclidean distance of the clients' neural network weights. Based on similarity, the system clusters clients using the Louvain method [32]. In each group, the client with the highest sum of similarity is the leader. The leader performs federated training of the model's first layers with leaders from other groups. The model's final layers are trained individually in each group.

Stochastic Clustered Federated Learning (StoCFL) [33] is a federated learning framework that clusters clients according to the cosine similarity. The proposal introduces two models: the global model and the cluster model. The global model maintains information from all clusters, while clients only participate in model training inside the cluster they are involved. Meanwhile, there is a high cost to the system's clients, who must simultaneously train the two models.

Iterative Federated Clustering Algorithm (IFCA) [34] is a proposal for clustering clients to personalize their models. In the proposal, clients are responsible for choosing their clusters. In addition, the authors propose the use of multi-task learning, which consists of sharing some neural network weights for clients who have data distributions with intersections but are in different clusters. Nevertheless, the downside of delegating the process of identifying groups to the clients is that the environment becomes susceptible to malicious behavior and requires more computational costs from the clients' devices. Another disadvantage of this proposal is to assume that the number of groups is known a priori, which may be unfeasible and can either overestimate or underestimate the number of existing clusters.

Clustered Federated Learning (CFL) [36] recursively partitions federated learning clients into more homogeneous groups. This procedure mitigates the problems generated by non-IID distributions. Partitioning occurs whenever the loss gradient vector exceeds a pre-established distance threshold. Nonetheless, clients' recursive partition leads to computational overhead on the aggregation server because it runs the procedure at each global training epoch. Ouyang *et al.* [9] present ClusterFL, a framework for creating homogeneous client groups in federated learning environments. ClusterFL periodically verifies groups to detect inefficient clients, which results in a fast convergence time of the group model. Nonetheless, their proposal introduces unnecessary computational overhead due to periodic model retraining. Furthermore, both proposals do not apply cross-cluster model sharing for clients with generic tasks.

Flexible Clustered Federated Learning (FlexCFL) [35] is a framework that considers clients' data distribution time shifts. The grouping strategy is static, which avoids rescheduling clients for each epoch, as [36] and [34] do. Also, the framework uses only a subset of clients to determine the number of clusters in the system and

Table 2.1: Comparison of related works

Work	**Strategy to Reduce Heterogeneity**	**Strategy Algorithm**	**Strategy Input**	**Iterative**	**Hierarchy Level**
[9]	Client Clustering	Periodic clustering	All models' weights	YES	1
[10]	Client Selection	Reinforcement learning	All Weights	NO	0
[18]	Client Selection	Optimization	Users' resources	NO	0
[19]	Client Selection	Client's utility measurement	Users' resources	NO	0
[20]	Client Selection	Model regularization	Proximal term value	NO	0
[21]	Client Selection	Optimization	Users' resources	YES	0
[22]	Hierarchical Training	Geographical training	Users' Location	NO	2
[23]	Hierarchical Training	Hierarchical early-exit	Transformed Data	YES	2
[24]	Client Selection	Irrelevance measurement	Irrelevance Metric	NO	0
[25]	Client Selection	Clustered sampling	Sample size or similarity	NO	0
[26]	Model Personalization	Learn-to-personalize	Local data	NO	0
[27]	Hierarchical Training	Hierarchical early-exit	Users' Location	NO	2
[28]	Model Personalization	Two-steps optimization	All Weights	YES	2
[29]	Client Clustering	Distributed clustering	Users' centroid	NO	1
[30]	Model Personalization	One-versus-All	Users' response	NO	0
[31]	Client Clustering	Louvain method	Euclidean distance	NO	1
[33]	Client Clustering	Cosine similarity	Users' loss function	NO	1
[34]	Client Clustering	Iterative clustering	Clients' decision	YES	Multiple
[35]	Client Clustering	DCS	All Weights	YES	1
[36]	Client Clustering	Recursive clustering	Loss gradient vector	YES	Multiple
Our	Client Clustering	Generic clustering	Last Layer Weights	NO	1

calculate the Decomposed Cosine Similarity (DCS). The remaining clients are clustered after the decision about the number of clusters. Before each round, the authors execute a strategy to detect if the clients remain with the same distribution or if it has changed. When the distance exceeds a predefined threshold, the client needs to perform the clustering step again. Nevertheless, the proposal requires higher computational resources than our proposal to detect the data shift.

2.4 Contributions

Our contribution is divided into two parts. First, we propose a model personalization system through client clustering, unlike previous client selection proposals that only consider optimizing a generic model across the entire federated learning system. Our system aims at clustering clients with homogeneous data. Like other CFL proposals, the clustering process increases the accuracy and reduces the training time of the models of each cluster, as it makes the training data more homogeneous. Furthermore, clustering clients can improve model convergence in the cluster and increase the final performance for specific learning tasks. Nevertheless, our proposal avoids interactive steps or sending the whole model to the clustering algorithm, therefore it reduces the overhead of defining the clusters. We define the clusters with an unsupervised clustering algorithm that receives clients' neural network weights as an input vector. Therefore, the proposal maintains the privacy requirements of traditional federated learning, since there is no private data sharing. In addition, the proposal is agnostic to the clustering algorithm, eliminating the need for prior knowledge of the number of existing clusters. In this part of the proposal, clients only retain local cluster knowledge and are limited to specific tasks, like classifying data similar to the cluster training data.

Also, we extend our system to implement ATHENA-FL, a federated learning system based on clustering clients by data distribution similarity and training neural network models through the one-*versus*-all (OvA) approach. This second part provides a way to combine the models generated in different clusters and to share knowledge among clusters. Initially, it maintains the clients' clustering scheme. Unlike previous proposals, each cluster trains multiple models, which are detectors of the classes present in the clients' datasets. At the end of training the models of each group, it is possible to ensemble them to detect the class of samples outside the cluster by using the OvA model. The performance of ATHENA-FL is compared to FedAVG using the MobileNet [14] and MobileNetV2 [37] architecture, lightweight deep neural networks for image classification. Furthermore, the experiments compare the communication cost of the two approaches, calculated through the number of bytes transmitted per client during training. We show that there is a trade-off

between the number of existing classes and the accuracy of the one-*versus*-all model when the clusters have few classes and there are structural similarities in data outside the cluster, e.g., cats and dogs have similar shapes. More classes in the system imply more detectors, which must be trained with more diverse data for a better distinction between classes and to learn the most important data characteristics to better classify the new samples. Thus, detectors trained with few classes present a lower performance on out-of-group data, reducing the accuracy of the OvA model.

Table 2.1 summarizes the characteristics and comparisons of all related works. We classify the works due to the strategy to reduce heterogeneity, the algorithm used, the algorithm input, whether the approach is iterative, and how many hierarchical levels there are. The strategies to reduce training heterogeneity are client selection, hierarchical training, model personalization, or client clustering. Besides, there are multiple algorithms used, which depend on different inputs. The algorithm also determines if the strategy is iterative. Regarding the clustering level, 0 means that the proposal uses a single global model, and 1 the proposal has clusters where each one has a cluster global model. Multiple or 2 clustering levels mean that the cluster can be divided into one or more sub-clusters.

Chapter 3

Federated Learning Concepts

Figure 3.1 exhibits the execution diagram of the proposal presented by Google. First, the aggregation server randomly selects a set of clients that will participate in epoch n. In the example in Figure 3.1 clients 1 and 4 are selected. In this step, the clients calculate the new model state for epoch $n-1$ using local data. The computed models are transferred to the aggregation server, which aggregates into the epoch n global model. Among the existing aggregation algorithms, the most popular is the Federated Average (*Federated Average* - FedAVG), which calculates the weights' average in each model layer to perform the model update. Federated learning average is formulated as an optimization problem in which we search for an optimal solution that minimizes the loss function by adjusting the w model's parameter over the parameter space V^a as formulated in Equation 3.1. After aggregation, the global model for epoch n is generated and the aggregation server shares it with all clients. The process runs until one of the stopping conditions is met, such as the maximum number of epochs, classification performance, or convergence of classification metrics:

$$\min_{w \in V^a} \left(f(w) \stackrel{\text{def}}{=} \sum_{n=1}^{N} p_n F_n(w) \right). \tag{3.1}$$

In the formulation above, N is the number of clients in the system and p_n is the neural network weights of the n-th client, a.k.a model's parameter, $p_n \leq 1$ and $\sum_n p_n = 1$. In FedAVG, $p_n = s_n/s$, where s_n is the size of the n-th client dataset and s is the number of samples among all clients. $F_n(w)$ is the local objective function for the client n. $F_n(w) = \frac{1}{s_n} \sum_{k \in s_n} f_i(w)$. The function $f_i(w)$ is related to $l(x_i, y_i; w)$, the loss function for the prediction (x_i, y_i) with model parameter's w. FL algorithms' convergence conditions depend on feature distribution among clients and data distribution. Hence, we define the classification of federated learning according to the feature distribution and discuss how the data can be distributed in the system.

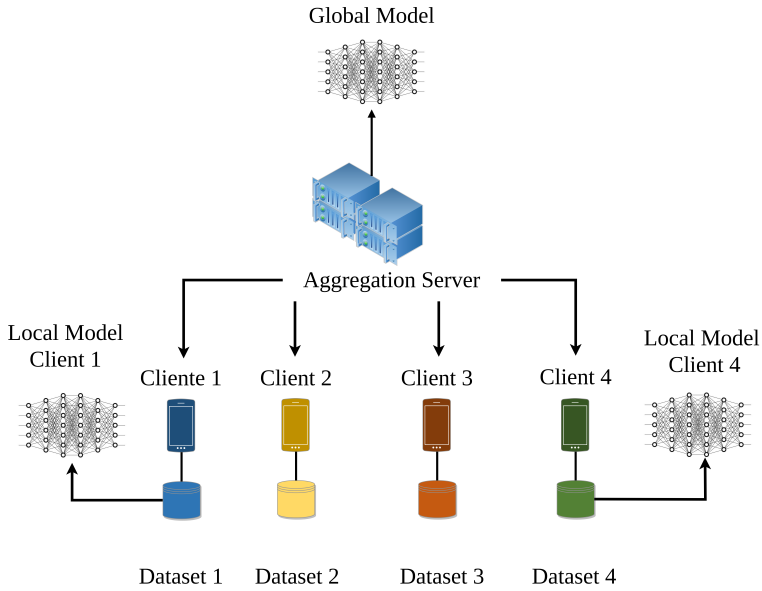

Figure 3.1: Federated learning architecture with four clients. Round n starts with the aggregation server randomly selecting some clients. Selected clients use their datasets to enhance the global model for epoch $n-1$ and build the local model for epoch n. Each client communicates with the aggregation server independently, and their datasets remain on their devices throughout the training process. The aggregation server receives local updates from clients and aggregates the responses to the global model, generating the global model for round n. Finally, the global model for round n is sent to clients. The process is repeated until a stopping condition is reached, such as model convergence or the maximum number of global epochs.

3.1 Feature Distribution

Federated learning has three classifications according to the characteristics that each client has [38]: Horizontal, vertical, or federated transfer learning.

Horizontal federated learning: In horizontal federated learning, clients have the same feature space and learning task, however, clients' samples are different. Horizontal learning has difficulty establishing previously which features the clients should collect and use in model training. However, the training and aggregation process is simple, as all clients use identical input sizes.

Vertical federated learning: Another form of federated learning is vertical, which assumes a scenario in which clients have non-intersecting feature distributions. This is the most challenging type of learning, as clients collect data that can vary in dimension, making the global model creation process more complex. However, there is no need to communicate which features are extracted, so clients have a more robust privacy model.

Transfer federated learning: This classification assumes that the participants have some similarities in the learning task. Thus, it is possible to train a model a few epochs in a dataset and afterward, execute a fine-tuning in neural network weights

with the final learning task to improve the model performance.

3.2 Data Distribution

In this section, we discuss sources of non-IID data distributions. Ma *et al.* identify five scenarios where data are heterogeneous among clients: Feature distribution skew, label distribution skew, vertical federated learning, label inconsistency, and quantity skew [7], which we consider in this work. Below are the details and characteristics of each one:

Feature distribution skew: When a client collects data, it can be specific rather than general. Therefore, the features might significantly differ from one user to another. The data can be related to geographical locations, such as temperature data, or even to user personality, such as vocabulary and handwriting traces.

Label distribution skew: Datasets that have imbalance classes have this type of heterogeneity. For instance, client 1 has 80% samples of class 1 while the other 20% are samples of the remaining classes, and client 2 has the opposite data behavior, 20% samples are from class 1 and the 80% are distributed among the remaining classes. Our experiments consider specifically this type of heterogeneity.

Vertical federated learning: This learning scenario is inherently Non-IID since clients have different feature space distributions.

Label inconsistency: Clients may disagree with sample labels. This can occur because the parties disagree on which class a sample belongs to, or the labeling processing lacks a well-defined protocol to assign classes to samples. For example, categorical features such as "good", "regular", and "bad" might subjectively change according to the professional responsible for defining a sample class.

Quantity skew: In this scenario, clients significantly differ in the dataset size. For instance, client 1 has $|s_1| = 2,000$ while client 2 has $|s_2| = 500,000$. The difference between quantity skew and label distribution skew is that in quantity skew sample might originate from the same distribution but with different sample sizes among the clients.

3.3 Problem Formulation

The developed work focuses on federated learning with horizontal data distributions and assumes that the aggregation server communicates to the clients the input size they will collect before starting the training process. We also assume that all data pre-processing is well-established and executed by clients before the training. Also, our non-IID tests are based on the label distribution skew scenarios. Thus, for the first part of this work, we cluster the clients to avoid statistical heterogeneity in

the training set. We assume that the clusters are homogeneously created, therefore, clients have similar samples of the same classes.

The second part comprises the sharing of intra-cluster models using the OvA method, in which each cluster trains detectors for all existing classes in the client datasets. Besides, the server together with the clients uniquely identifies existing classification tasks and labels. In this way, new classes can be incorporated into the system without loss of generality concerning existing models. Furthermore, there are no equal labels for samples belonging to different classes in the system before clustering. Our system, in this part, uses the weights of the last layer as input for a clustering algorithm, which assigns clients to one of the G clusters. The second part of the thesis, ATHENA-FL, uses one detector per class. Thus, instead of minimizing a single objective function as FedAVG, our system minimizes the classification error for each detector r_k by adjusting the parameter w_k:

$$\underset{w_k \in V^a}{min} \left(r_k(w_k) \stackrel{\text{def}}{=} \sum_{n=1}^{N} p_{k_n} F_{k_n}(w_k) \right), \tag{3.2}$$

where $r_k(w_k)$ is the detector of the k-th class, p_{k_n} is the selection probability of the n-th client during the k-th detector's training. Finally, $F_{k_n}(w_k)$ is the local objective for the n-th client to the k-th detector.

The following chapters present our two proposals in detail, explaining the system execution and how we avoid data heterogeneity.

Chapter 4

Clustering Clients to Reduce Training Data Heterogeneity

This chapter presents the details of the first part of our proposal, which consists of a method to create accurate models with hierarchical clustering of clients. Firstly, we approach the client's clustering for the creation of cluster models, and later we discuss the composition of more generic models using cluster models. Finally, we discuss possible attacks on the system and countermeasures introduced by our proposal. The system is applied in a horizontal federated learning environment, where clients sample data with the same set of features.

The model generated is specialized for a cluster of clients with IID data, improving the system's overall performance. Thus, instead of one global model, the proposed system has z personalized models, called cluster models, for each of the g existing clusters sharing IID data.

We assume that message exchanges between the clients and the server are encrypted, and only the server knows the clients. This prevents malicious clients from getting information about other clients and using it to degrade the model, assuming the server has not been compromised. Furthermore, it is assumed that the server is honest, running the FedAVG algorithm for aggregating the gradients correctly and combining the models on demand. Clients are assumed to have stationary data so that the distance between the individual loss gradient vectors of the selected clients and the average vector of the cluster for a specific epoch remains the same.

4.1 Clients' Clustering Procedure

The proposed system for training federated models is composed of an initialization followed by two phases. The division of clients aims to minimize the data heterogeneity during the global model training. Hence, during the system initialization,

the server performs a global epoch of federated learning, selecting all the clients in the network. Clients locally calculate the neural network weights with their data and return the result to the server. After receiving the response from the clients, the server executes the first phase after initialization to allocate clients into clusters. The server uses a clustering algorithm to determine the number of clusters and the clients of each cluster. As a result, clients with similar neural network weights are allocated into the same cluster. Note that cluster creation occurs without the need to access the client's samples, which is a requirement for keeping privacy. Also, we use only the last-layer weights, avoiding the transmission of the whole model. Besides, we need only to communicate the model and retrieve the weights one time, avoiding the iterative process overhead present in other proposals. After cluster definition, the second phase starts, consisting of traditional federated learning training among clients in the same cluster. Therefore, in the proposed system, there are at least $G \geq 1$ specific models, where G is the number of clusters generated by the server. Unlike traditional federated learning, the proposal calls the federated learning server a selection and aggregation server, because, besides aggregating the results, the server is responsible for selecting and keeping track of each client's cluster.

We define two system's stages: initialization and maintenance. The initialization comprise the steps to create the system's clusters with the initial clients, while the maintenance consists of the steps to cluster new clients once the cluster were defined. Thus, Section 4.2 discusses our premises for the system initialization and how we execute the clustering. Then, Section 4.3 exhibits the procedure for one or more clients that arrive after the system initialization.

4.2 System initialization

The steps in Fig. 4.1 are performed to initialize the system. In the first step, the server shares the test model with all known clients. The clients adjust the test model parameters with their local data and return only the bias vector from the last layer to the selection and aggregation server. The server runs the clustering algorithm and saves the existing clusters. Once clusters are established, new clients will not be able to generate new clusters, only allocated to existing ones. The initial model of each cluster may differ from the test model, as the server can adjust the model hyperparameters according to the data from each cluster. Nevertheless, cluster model optimization is out of the scope of this work. Adding new clients leads to different procedures, depending on whether the model training is finished or not. New clients can participate in the model training if they ingress before model convergence, or they only receive the model if they ingress after model convergence. Furthermore,

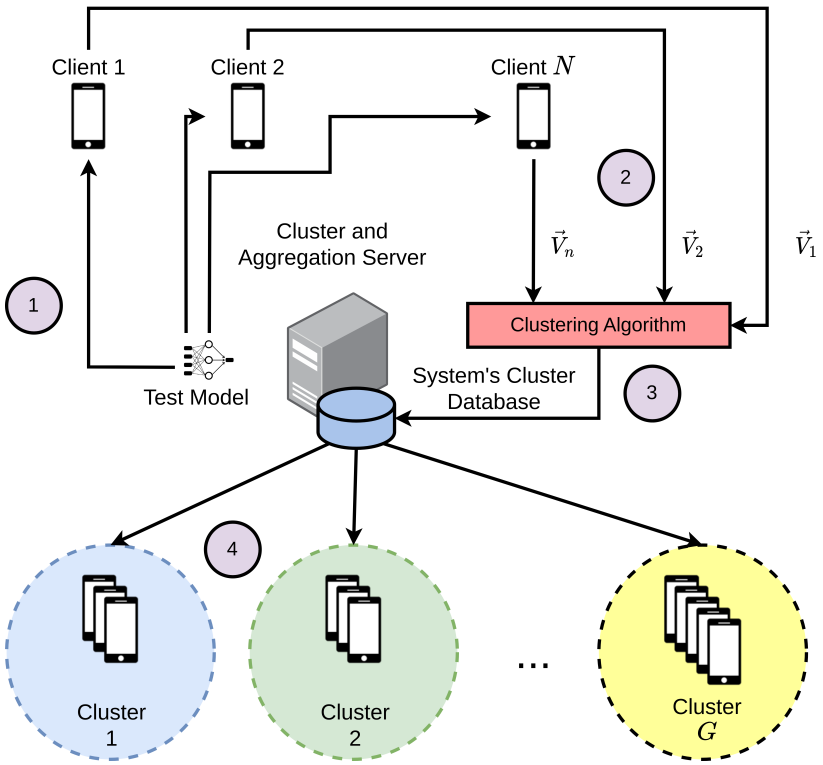

Figure 4.1: Initialization of the proposed system. (1) The initial set of clients receives the test model. (2) Clients adjust the neural network weights and return the *bias* vector from the last layer of the model to the server. (3) The selection and aggregation server runs the clustering algorithm to get the clusters from the system. (4) The server associates new clients with clusters.

the proposal predicts a fixed number of clusters after initialization, assuming that clients have stationary data. If all clients in a cluster fail, the aggregation server stores information about the cluster, consisting of the training state and current model. This is important as the failed clients can be recovered, and new clients can be eventually allocated to the cluster. After initialization, the clustering algorithm is no longer used to identify new clusters. It is used to identify malicious actions and assign new clients to existing clusters instead. The system is agnostic to the clustering model used, allowing the administrator to select the clustering algorithm that best suits the requirements. The proposal assumes that the administrator has relevant information to adjust the clustering model hyperparameters before computing the clusters, e.g., existing classes and a small dataset. Another hypothesis is that the clients' learning tasks are the same, e.g., image classification.

4.3 System Maintenance

The entry of new clients into the environment is shown in Figure 4.2. The new client that wants to participate in federated training demands the selection and

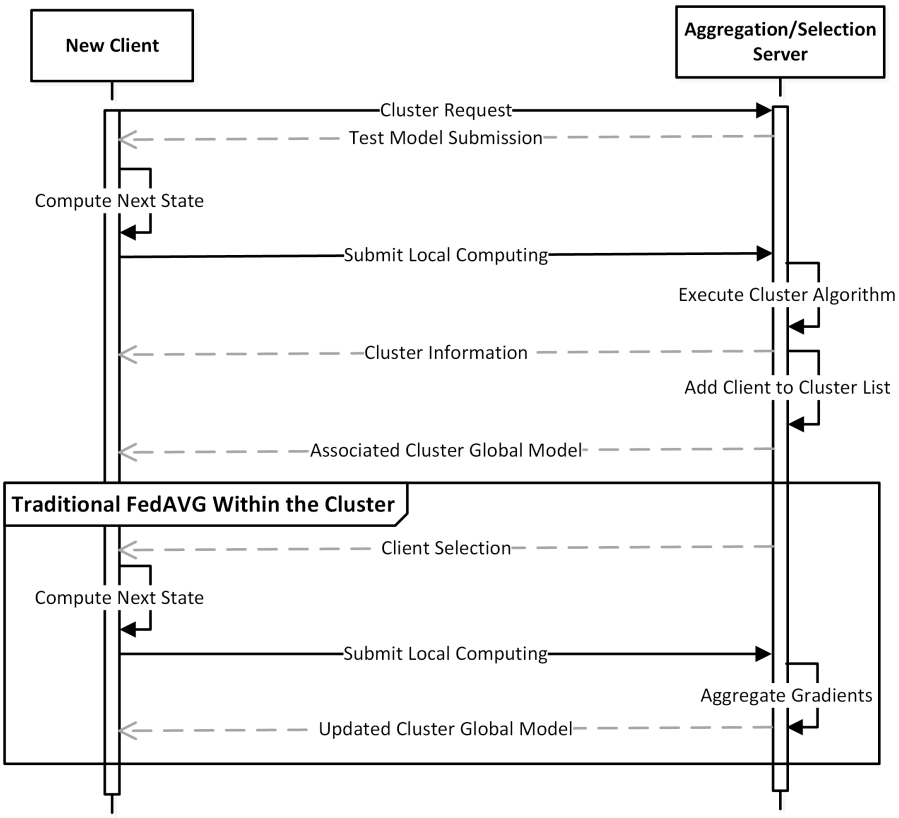

Figure 4.2: Proposed execution diagram. The first step is to allocate a new client to an existing cluster. After the client is allocated to a cluster, traditional federated learning is performed in the cluster.

aggregation server of a new cluster identifier. The server sends the client a test model, which is identical for all new clients and is also used to generate the system's clusters. After this step, the client calculates the update of the test model and sends the result to the server. With the responses from the clients, the server runs a clustering algorithm and associates the client with an existing cluster. Finally, the FedAVG algorithm is performed independently in each cluster.

Fig. 4.2 details the steps developed by the proposal for the creation of system models. Initially, the client requests the server to be included in a cluster and execute similar steps as the discussed in the system initialization phase. The server sends the test model to the client to obtain statistical information. This step is paramount in the proposal, as it clusters clients without revealing data privacy. The client updates the test model with its private data and sends part of the result to the selection and aggregation server. After receiving the response from the client, the server runs the clustering algorithm adjusted at system startup to determine which cluster the new client belongs to. The cluster information is sent to the client and their identity is added to the cluster list. Finally, the server sends the current z_t cluster's model to the client, allowing it to participate in the federated training.

From this stage, federated training takes place in the traditional way in the g_j cluster in which the client was allocated. At each E_q cluster's epoch, the server randomly selects a fraction of clients to adjust the parameters of the cluster model. If the client is selected, it calculates the new w' weights and sends them to the server for aggregation. Finally, the server sends the updated model z_{t+1} to the clients of the cluster g_j, and a new epoch, E_{q+1}, for the cluster, begins. The process is repeated until a stopping condition is reached, such as a model performance convergence criterion, e.g., accuracy within a low variation threshold in consecutive epochs, or a maximum number of global epochs executed $E_{q_{max}}$.

Our proposal can be used in scenarios when clients have distinct data distributions, like an image-processing scenario where clients collect different classes and try to learn how to classify them. We test our proposal in a scenario where clients have label distribution skew, to simulate the non-IID scenario within image datasets. These results are shown in Section 6.1 and 6.2.

Chapter 5

Combining Models from Different Clusters

This section presents the ATHENA-FL system, an improvement of the previous clustering proposal. ATHENA-FL introduces the intra-clustering information sharing, through the OvA model. Thus, we present the OvA model and the implemented changes in the training phase.

ATHENA-FL is a federated learning system for training models, ensuring data privacy. Our system relies on a set of clients clustered according to data similarity. Client clustering allows the training of machine learning models more efficiently than the initial federated learning proposal [2] under non-IID data distributions, increasing the final accuracy and reducing the total epochs for model convergence. Nevertheless, clustering creates specific models, which usually can only classify samples generated from the same cluster. Thus, we propose the adoption of the OvA model to share models trained on different clusters and to create generic models. The OvA learning model consists of training binary detectors for each data label. Each detector gives the probability that a sample belongs to its class. Since they execute a binary classification, their training process is easier and converges faster than a deep neural network for multi-class classification. We combine the detectors and take the one that provides the greatest probability to assign a sample predicted label. Figure 5.1 displays the steps proposed for executing the system.

5.1 Data Similarity-based Clients Clustering

Initially, the system arranges clients into clusters according to the data similarity. This step produces clusters in which the training data are independent and identically distributed. IID datasets facilitate model convergence and increase final classification performance [39]. Nevertheless, to maintain the privacy assumptions

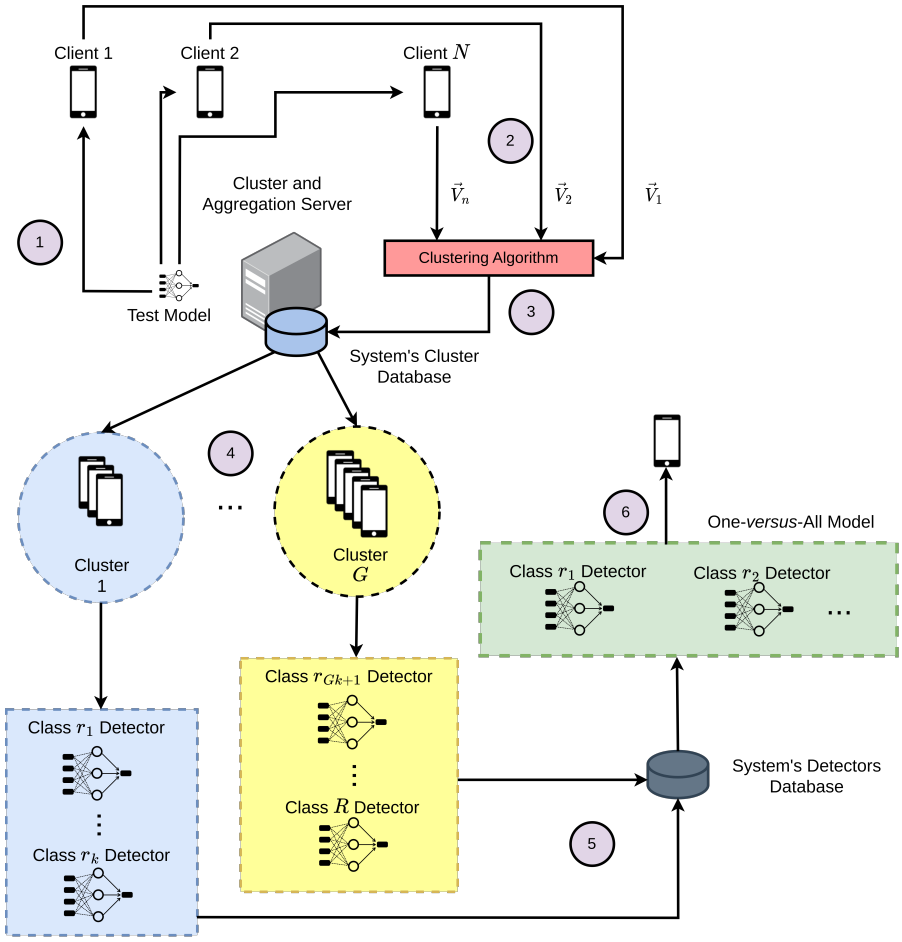

Figure 5.1: The ATHENA-FL execution scheme consists of 6 steps. Steps 1 to 4 are discussed in the previous chapter to mitigate the system's statistical heterogeneity by clustering clients according to the similarity of the data. Step 5 shares detectors among other clusters. Finally, in Step 6, a client can combine models from different clusters to create a generic OvA model.

of Federated Learning, ATHENA-FL indirectly obtains from clients' data distribution for client clustering. Thus, we use a test model, in which clients execute a few training epochs with their local data. The aggregation server uses these model weights as input for the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) clustering algorithm to identify the different data distributions and assign clients to their clusters. Figure 5.1 shows the steps executed by ATHENA-FL. Firstly, Steps 1 to 4 group clients according to data similarity as presented in Chapter 4.

Unlike detectors, the model used for testing is a deep model for multiclass classification, as it is necessary to compare data distributions across all clients at this step. The clustering process ends with Step 4, which trains the detectors. New clients, absent in the group creation stage, must request the server to be allocated into a cluster by executing the previous steps.

5.2 Detector Training

Detector models are trained independently, both concerning models from different clusters and detectors trained within each cluster. Thus, this step can be parallelized to reduce the models' training time. Each group has a total of k detectors, which can vary according to the group, and the system has R detectors in total.

Thus, in Step 4 of Figure 5.1 the system trains detectors for each class. Training starts with the selection and aggregation server determining which detector will be trained federated into a group. After defining the class of interest, the server sends this information so that the clients can pre-process the labels. The pre-processing consists of transforming the original labels of all client samples into binary labels. Thus, the detector's training needs an auxiliary function to binarize the labels according to the current detector. The binarization function is defined as:

$$B(k) = \sum_{n \in G} \mathbb{1}_k(s_n), \tag{5.1}$$

if the i-th label belongs to the k-th class, the function returns 1 and zero otherwise.

The convergence time for each detector is reduced due to the clients' clustering that has approximately IID datasets. However, the models generated in the cluster are specific and detect only the existing classes that belong to the cluster. Thus, after its convergence, each detector is made available in the detector's database of the system in Step 5 of Figure 5.1. This allows clients from other groups to build one-*versus*-all models with detectors generated in the whole system.

We highlight two advantages of using this approach. First, detectors are smaller, hence the time between training epochs is reduced compared to deeper neural networks. Secondly, the OvA model can identify the classes present on other clusters easily, since we merely combine the detectors from the system's detectors database.

5.3 One-*versus*-All Model

Step 6 allows the creation of a generic model, which uses detectors trained in other clusters. After sharing models in the ATHENA-FL's detectors database, clients select the detectors of interest to create the one-*versus*-all model, C_{ova}. Equation 5.2 shows the classification process of a sample x. C_{ova} returns $\hat{y}$, the estimate class for sample x.

$$C_{ova}(x) = \underset{i \in [1,R]}{\operatorname{argmax}}\ r_k(x). \tag{5.2}$$

The model comprises several detectors r_k, which classify the sample x and return the probability that the sample belongs to the class k. The classifier C_{ova} assigns to

the sample the label that has the highest probability among all detectors. ATHENA-FL allows a parallel classification process, given that, as detectors are independent of each other, each model can simultaneously generate the sample's probability of belonging to the detector's class.

Chapter 6

Development of the Prototype and Experimental Results

We develop a system prototype in Python v3.9.1 with Flower v0.18.0 [40] for the development of the federated learning environment, the scikit-learn library v1.0.1 [41] to create the cluster models, and the Keras v2.6.0 library to create the deep learning models. The experiments were executed on an Intel Xeon CPU E5-2650 2.00 GHz server with 32 processing cores and 504 GB of RAM. We show experimental results of the models' evaluation, with the average accuracy obtained among all the clients and a 95% confidence interval. We compare our proposal with FedAVG because other approaches use different models or datasets. Thus, FedAVG establishes a baseline comparison with the state-of-the-art algorithms.

For our first proposal, the evaluated scenario uses the configuration values displayed in Table 6.1. Not all clients are selected for training in a global epoch, thus the percentage of selected clients indicates how many of them execute a global epoch model update simultaneously. There is an exception during the system initialization when we select all clients to create the system's clusters. In addition, test accuracy and loss results are constructed from the selection of all clients at each global epoch. After allocating the clients into groups, the hyperparameters are adjusted according to the group data, e.g., reducing the number of neurons in the final layer due to the absence of all classes in the group. Therefore, the server is responsible for performing the hyperparameter tuning process on each group for the best performance of the models.

ATHENA-FL scenario has 50 clients that train the model during 5 local epochs for 200 global epochs, with a selection probability of 20%. The selection probability is the percentage of clients selected within each cluster for training global epoch. We use a batch size of 32 samples, and the accuracy results are obtained from the selection of all clients at the end of each global epoch.

We train and evaluate the performance of models on three image datasets:

Table 6.1: Parameters applied to the federated learning environment.

Parameter	**Settings**
Number of clients	10
Percentage of selected clients	50%
Number of global epochs	100
Number of local epochs	5
Size of local batches	32

CIFAR-10 [42], MNIST [43], and Fashion-MNIST (FMNIST) [44]. The CIFAR-10 dataset has 60,000 samples and a total of 10 classes representing objects or animals. The images are colored with 3 matrix, which represents RGB channels and have 32x32 pixels each. The second dataset, MNIST, has 70,000 samples divided into 10 classes, which represent handwritten decimal digits. FMNIST also has 10 classes and 70,000 samples representing different fashion clothes, divided into 60,000 for training and 10,000 for testing. MNIST and FMNIST images are provided in grayscale, originally 28x28 pixels. We process the datasets to extend to 32x32 pixels to use the same neural network architecture in all data. The datasets are equally balanced among existing classes.

The neural network architecture used in the OvA model was adapted from a dog and cat image identification problem, establishing a simple model for binary classification [45]. Then, this neural network architecture was adapted to be the basis of the detectors used in the one-*versus*-all model due to its small size and its high capacity to correctly identify samples.

The deep architectures used in classification tasks with multiple classes are MobileNet and MobileNetV2. MobileNetV2 was chosen because it is the architecture used in the classification example of the CIFAR-10 dataset in the Flower framework, while MobileNet was selected due to the shorter inference time and is shallower than MobileNetV2.

We conduct five experiments: (i) identify the best clustering algorithm and its parameters; (ii) test the performance of our clustering proposal and compare it with FedAVG; (iii) verify the detectors' accuracy and compare it with a complex model trained with FedAVG; (iv) define the number of epochs necessary for both models to converge, and finally, (v) estimate the total amount of bytes transmitted during the training process for the two alternatives. Evaluations (i) and (ii), discussed in Section 6.1 and 6.2 respectively, are related to the first part of our work while the remaining are related to ATHENA-FL. Experiments (iii) and (iv) are discussed in Section 6.3, while the last one is presented in Section 6.4.

6.1 Evaluation of Cluster Models

The first step of the system clusters the clients from the update vector sent to the server. Thus, Experiment I evaluates ways of clustering clients based on the information provided. In this experiment, the hyperparameters of three clustering algorithms are analyzed: K-Means [13], OPTICS [12] and DBSCAN [11].

K-Means is a widely adopted algorithm for clustering tasks. Its main hyperparameter is the number of c clusters existing in a given dataset. Nevertheless, defining this hyperparameter in an unknown dataset is an arduous task, as it is required to estimate the number of existing clusters in advance. To circumvent that, one may want to use an iterative approach and estimate the most suitable number of clusters, which is a computationally expensive task. In this way, OPTICS appears as a more practical alternative, as it requires the minimum number of samples to form a cluster as a hyperparameter. Nevertheless, setting a minimum number of samples is not enough for large clusters, as the same problem of data heterogeneity would recursively appear. Hence, an alternative to the previous algorithms is DBSCAN. Besides the possibility to configure a minimum number of samples to create a cluster, DBSCAN uses the maximum distance between two samples as a hyperparameter. Therefore, it is possible to find a practical approach to separate clients into clusters using test model replies. Each cluster contains clients with IID data.

The three clustering models correctly generated the same clusters in the non-IID scenarios. Nevertheless, in the IID scenario, DBSCAN optimally clustered the clients by identifying only one cluster, overcoming OPTICS. Hence, the clustering results show that the most consistent algorithm is DBSCAN, configured with a minimum distance of 0.0279 between client vectors, which we further explain how we reach this value. The next experiments evaluate the models generated after clustering the clients.

The three algorithms were evaluated with respect to the clustering of clients for different initialization hyperparameters. Table 6.2 displays the number of generated groups and hyperparameter values for the DBSCAN and OPTICS algorithms. Values for K-Means are omitted from the table, as the algorithm's hyperparameter directly indicates the number of groups. The K-Means algorithm establishes an op-

Table 6.2: Analysis of the number of groups generated by the clustering algorithms in different data distributions. The DBSCAN algorithm is set to a minimum distance of 0.0279, while the minimum number of samples of the OPTICS algorithm is set to 2.

Scenario	**IID**		**Non-IID 2 Classes**		**Non-IID 5 Classes**	
Algorithm	DBSCAN	OPTICS	DBSCAN	OPTICS	DBSCAN	OPTICS
Clusters (#)	1	5	5	5	2	2

Table 6.3: Evaluation of the range of distances to identify 5 homogeneous groups with DBSCAN algorithm as a function of the number of local epochs executed by the clients.

Local Epochs	**Minimum Distance**		**Maximum Distance**	
	2 classes	5 classes	2 classes	5 classes
5	0.0023	0.0508	0.0488	0.4496
10	0.0023	0.0083	0.0475	0.0904
20	0.0024	0.0067	0.0568	0.0939
50	0.0048	0.0063	0.0722	0.1184

timal threshold for the performance evaluation experiments of the proposal because as the non-IID scenarios are created in a controlled way, the optimal number of clusters is known in advance. For the IID case, K-Means is configured to create clusters with at least two clients in order to guarantee the consistency of federated learning. The purpose of this configuration is to verify the impact on accuracy when the number of clusters is overestimated, dividing clients unnecessarily. So three is the number of clusters that maintain a minimum of two clients per cluster for K-Means.

A relevant evaluation is to determine the distance that allows DBSCAN to identify the clusters whose data are IID in the experimental scenarios. Thus, it is necessary to vary the distance hyperparameter and check how many clusters the model returns. The distance between the neural network weights of the clients strongly depends on the number of local epochs of the clients. Increasing the number of local epochs implies specializing the neural networks on the training data and, thus, producing vectors that are more distant from each other. On the other hand, clients that have datasets with similar distributions, when specializing their weights, produce vectors that remain close. Table 6.3 displays the values found for the minimum and maximum distances that can be assigned to DBSCAN so that the model determines the correct number of existing groups. The result indicates that using 10 local epochs, with a distance between [0.0083, 0.0475], allows us to correctly identify the existing groups. Thus, the implementation adopts the value for the distance hyperparameter of the DBSCAN algorithm as 0.0279, which is the average value of the range.

6.2 Performance Evaluation of Specific Models

The second experiment evaluates the behavior of the proposal when all clients' datasets are IID. The goal is to evaluate the performance of the proposal when using different clustering algorithms and verify the impact on accuracy when there are more clusters than needed. The hyperparameter c of K-Means was initially set to 5. This hyperparameter is decremented until clusters with the minimum number

of participants are established. So, through this approach, Experiment II uses $c = 3$. OPTICS was configured with a minimum number of clients equal to 2, and it also generated clusters unnecessarily. On the other hand, DBSCAN, when analyzing the vectors of clients, correctly identified only one cluster, once the dataset is IID.

The result of Fig. 6.1 indicates the average behavior of accuracy in all 10 clients using the traditional approach and the current proposal. The proposal, even using K-Means and OPTICS for clustering, has a final accuracy value close to the traditional case, despite dividing clients into clusters unnecessarily in this experiment. Thus, it can be concluded that there is no significant loss of performance even when clients have IID datasets and are divided into clusters. Furthermore, the experiment demonstrates the ability of the DBSCAN algorithm to identify that creating clusters is unnecessary in this case. Table 6.4 demonstrates that the behavior in each cluster is similar to the average behavior shown in Fig. 6.1.

In the third experiment, the data distributions on the clients are non-IID, with only two classes present in their datasets. The goal of this experiment is to compare the performance of the model generated through traditional federated learning with the one created by the current proposal, in which the data distribution is non-IID. Thus, the 6,000 samples of a class are split among five clients to create balanced datasets. Fig. 6.2 displays the experimental mean result among all clusters, illustrated in the figure as the blue line, and the traditional result, represented by the red line. Traditional federated learning is affected by the level of data heterogeneity, while the current proposal allows for high classification performance.

The experiment also analyzes the individual behavior of the clusters' models. The results show that the first cluster has an accuracy of $(56 \pm 3)\%$, while the second has an accuracy of $(63 \pm 3)\%$. Although there is a cluster with higher performance, even the low performance of the first cluster is higher than that obtained when using the traditional proposal. Thus, it can be concluded that the proposal successfully mitigated the effects of heterogeneity in the scenario with five classes per client.

The fourth experiment again evaluates the performance of the proposal for non-IID datasets. The difference from the third experiment is the reduction in the number of classes each client has to two classes. The results shown in Fig. 6.3

Table 6.4: Evaluation of the group models in the scenario where clients have samples of only 2 classes.

Group Identifier	**Number of Clients in the Group**	**Test Accuracy (%)**
1	2	81.05 ± 0.03
2	2	71.0 ± 0.4
3	2	75.3 ± 0.4
4	2	81.5 ± 0.1
5	2	80.4 ± 0.3

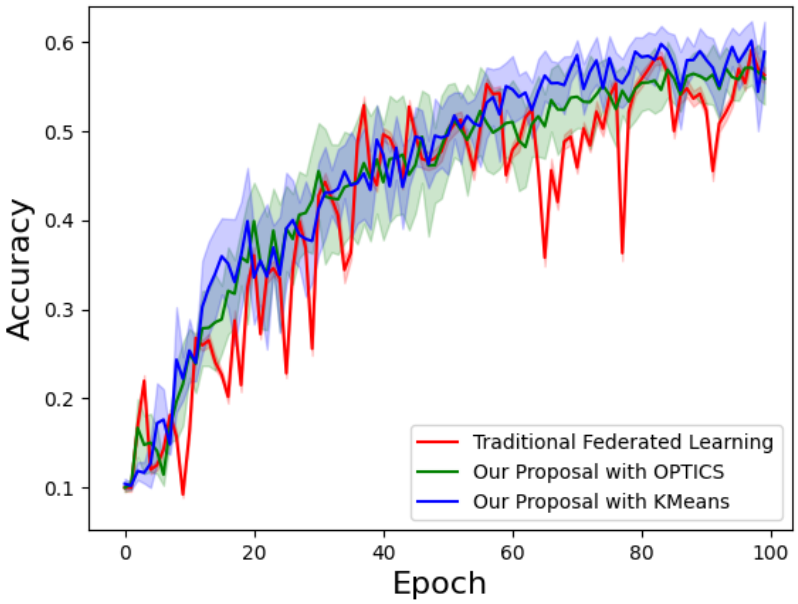

Figure 6.1: Accuracy evolution as a function of global epoch for customers with IID datasets. The red line represents the traditional approach of federated learning, while the blue and the green lines represent the proposal of this work with clustering through DBSCAN and OPTICS, respectively.

show similar behavior to the third experiment for both cases, but with a better final performance. This fact can be explained by the difficulty of the problem, simplified to a binary classification. Table 6.4 shows that although some groups have a slightly lower final performance, the result is higher than using traditional federated learning. Finally, it is possible to conclude that for groups 1 and 4 the performance improvement over the traditional proposal is close to 16%, indicating the relevance of the current proposal.

The fifth experiment evaluates the clustering approach in a non-IID scenario where the clients' datasets are not limited to samples of a subset of classes, thus creating a more realistic scenario. We build the datasets of this experiment through the Label-based Dirichlet Partition (LDA), which generates the data based on the number of clients and non-IID degree. Figure 6.4 shows the results of Experiment V. Our approach increases by more than 14% the accuracy compared with traditional federated learning.

6.3 ATHENA-FL Accuracy Evaluation

We evaluate ATHENA-FL on IID and non-IID data distributions. The non-IID data distributions are considered in two scenarios, in the first one each client has samples of only two classes of the dataset, and the second considers clients with samples of

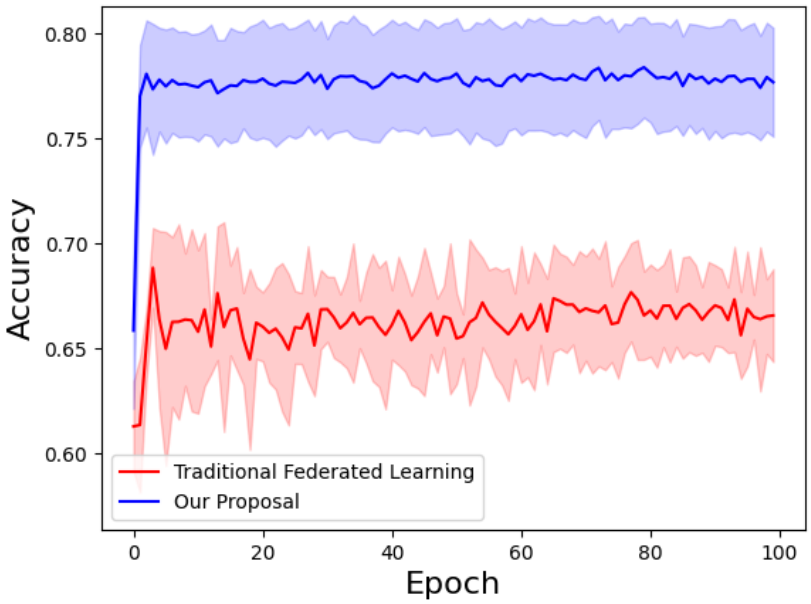

Figure 6.2: Evolution of the test accuracy as a function of global epoch for clients with datasets that contain only two classes. The red line represents the traditional approach, while the blue line represents the proposal of this work.

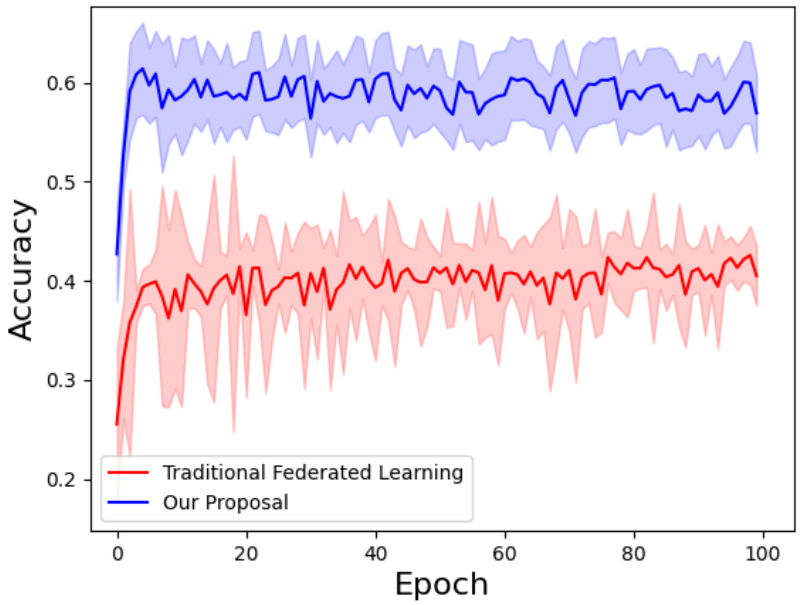

Figure 6.3: Test accuracy evolution as a function of overall epoch for clients with datasets containing only five classes. The red line represents the traditional approach, while the blue line represents the proposal in this work.

five different classes.

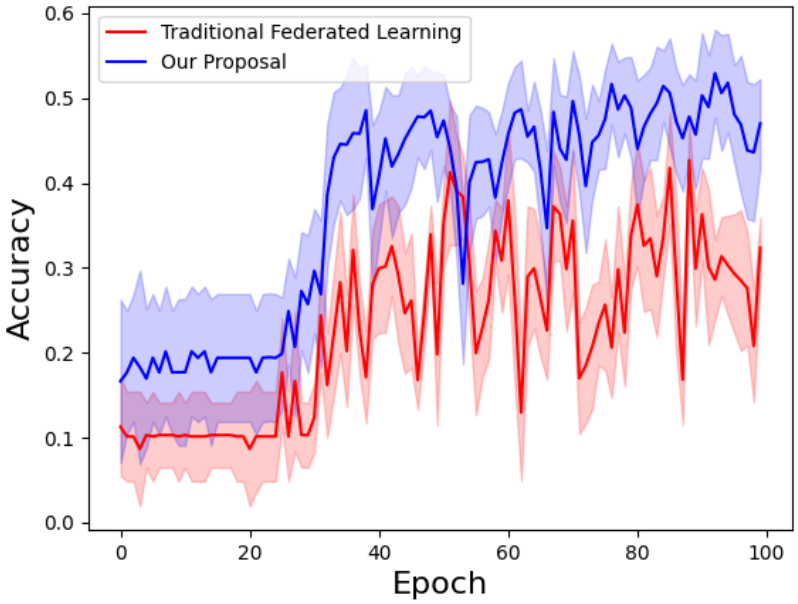

Figure 6.4: Evolution of the test accuracy as a function of global epoch for clients with datasets that contain only two classes. The clients' datasets are generated with the LDA using $\alpha - 0.5$. The red line represents the traditional approach, while the blue line represents our current proposal.

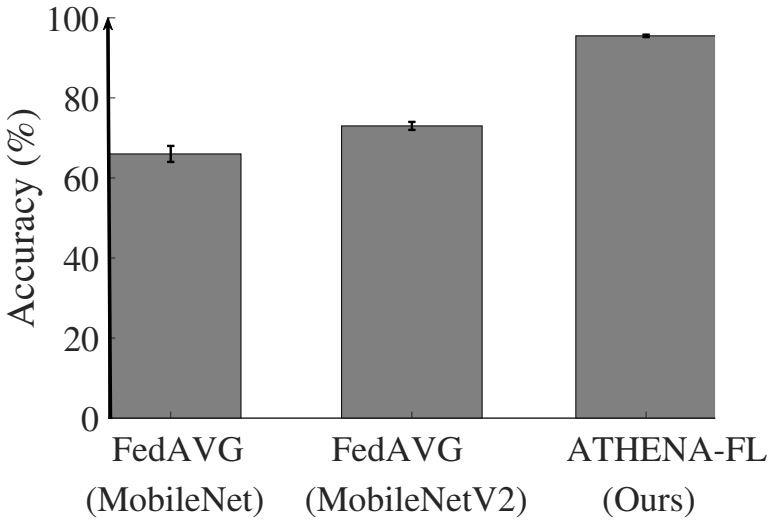

Figure 6.5: Final models' accuracy on the CIFAR-10 dataset with IID distribution.

6.3.1 IID Data Scenario

The detectors quickly converge to a high accuracy value, as shown in Figure 6.8, and the final performance of the one-*versus*-all model for the CIFAR-10 dataset is $(95.5 \pm 0.3)\%$, as shown in Figure 6.5. Meanwhile, the architecture MobileNetV2 has a final accuracy of $(73 \pm 1)\%$ and the MobileNet has (66 ± 2). In the MNIST dataset, the performance is better, with $(99.3 \pm 0.1)\%$ and $(99.8 \pm 0.1)\%$ for MobileNet and

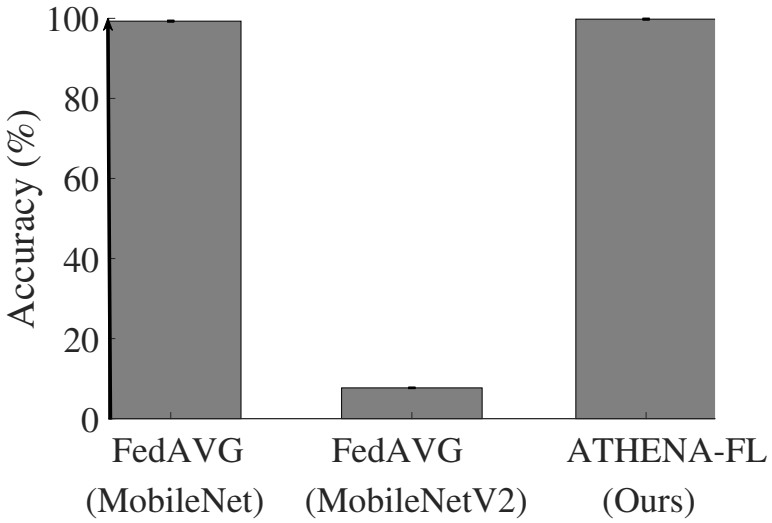

Figure 6.6: Final models' accuracy on the MNIST dataset with IID distribution.

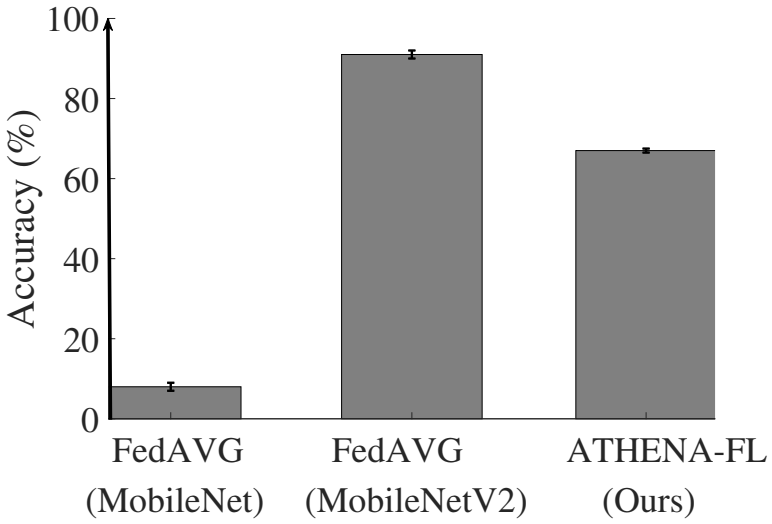

Figure 6.7: Final models' accuracy on the FMNIST dataset with IID distribution.

ATHENA-FL, as shown in Figure 6.6. However, MobileNetV2 is too complex for those data and has an overfitting problem, leading to only (7.69 ± 0.01) accuracy. Finally, the FMNIST shows that MobileNetV2 has the best performance in the IID setting, with (91 ± 1) against only (8 ± 1) for the MobileNet and (67.0 ± 0.5) for ATHENA-FL, as exhibited in Figure 6.7.

Thus, the experiment shows that under IID settings, ATHENA-FL has competitive results, but in some scenarios, the detectors might need to be well-adjusted to have a better performance. The variation in performance between the datasets is due to the difficulty of the problem presented by each one. MNIST has simpler grayscale images, which are simpler for classification. Therefore, the detectors have higher accuracy and less variance in this dataset than in CIFAR-10, which has color images with more elements. Finally, the shapes of clothes in the FMNIST dataset

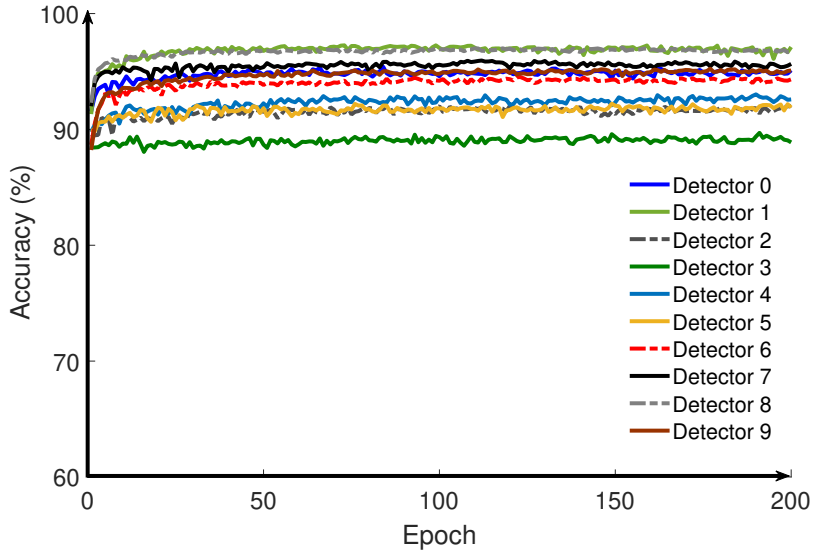

Figure 6.8: Evolution of detector's test accuracy over training epochs using the CIFAR-10 dataset.

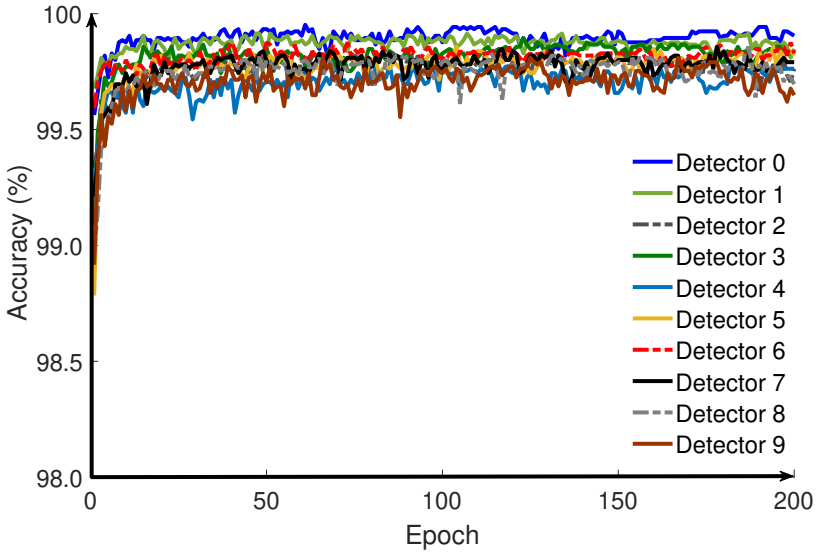

Figure 6.9: Accuracy of detectors over training epochs using the MNIST dataset.

are difficult to distinguish. Thus, we need more complex models to differentiate them. This behavior is also observed in the other evaluated scenarios.

6.3.2 Scenarios with Non-IID Data

Scenarios with non-IID data consider distributions of data where clients own only a subset of the classes of the dataset. In the first non-IID case, clients have samples from five distinct classes. Figure 6.10, 6.11, and 6.12 show the final performance of the one-*versus*-all model, MobileNet and MobileNetV2. ATHENA-FL provides the best accuracy results for all datasets in this setting, having with the CIFAR-10

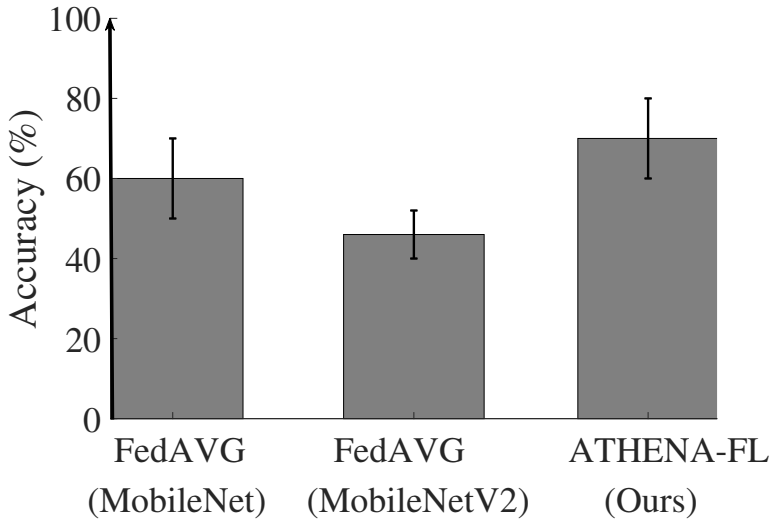

Figure 6.10: Final models' accuracy on the CIFAR-10 dataset with Non-IID distribution of 2 classes per client.

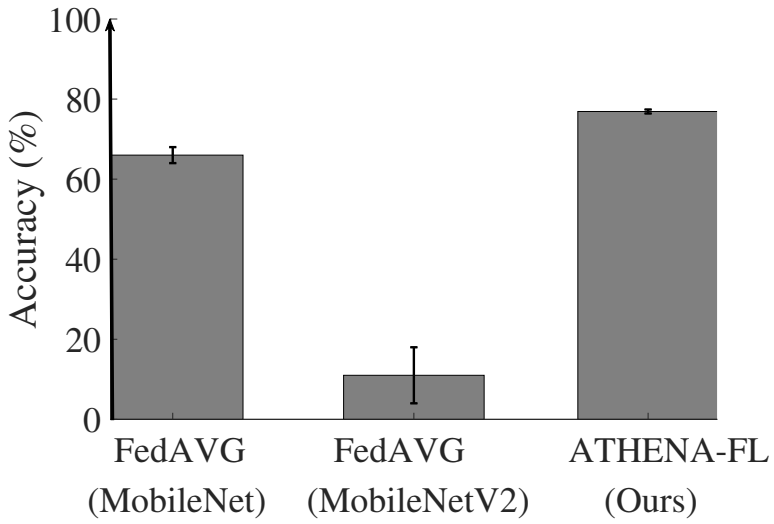

Figure 6.11: Final models' accuracy on the MNIST dataset with Non-IID distribution of 5 classes per client.

dataset 10% higher accuracy compared to the MobileNet, which is the second-best model, and 10.9% higher accuracy in the MNIST dataset. Thus, the experiment demonstrates that ATHENA-FL has the potential to increase the classification accuracy up to 10.9% under the Non-IID setting compared to the MobileNet model trained purely with FedAVG.

The last scenario considers a Non-IID data distribution with two classes per client. For the CIFAR-10 dataset, the observed accuracy is $(30 \pm 3)\%$ combining the detectors, while the MobileNet and MobileNetV2 architectures have a final accuracy of $(20 \pm 10)\%$ in this configuration, shown in Figure 6.13. The accuracy of ATHENA-FL was $(47 \pm 1)\%$ for the MNIST dataset, while the deep models achieved an accuracy

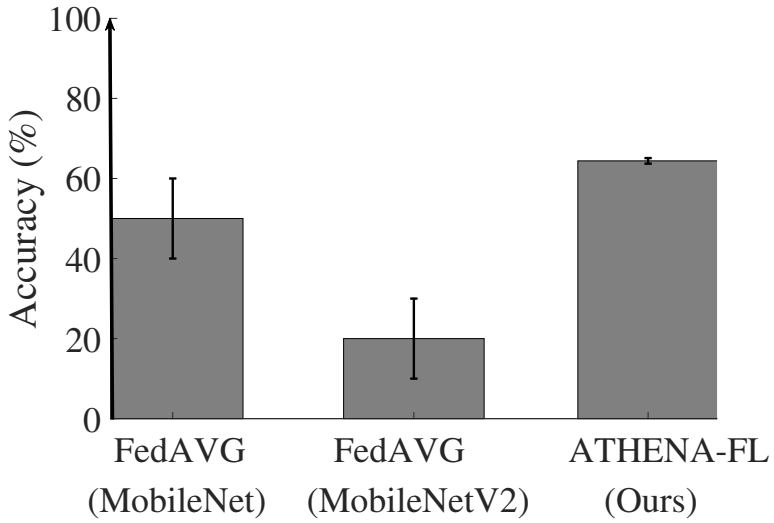

Figure 6.12: Final models' accuracy on the FMNIST dataset with Non-IID distribution of 5 classes per client.

of $(40 \pm 10)\%$ and $(10 \pm 5)\%$ for MobileNet and MobileNetV2 respectively, exhibited in Figure 6.14. Lastly, MobileNet has $(22 \pm 2)\%$, MobileNetV2 has $(60 \pm 20)\%$, and ATHENA-FL has $(50.0 \pm 0.7)\%$ accuracy in the FMNIST dataset, in Figure 6.15.

The results show that when the detectors are trained on datasets with more classes, the final classification performance is better since they can learn more patterns of the whole data distribution. This behavior can be explained by the higher variance of data from classes that are not of the detector's class. The greater variance of data in other classes allows the detector to identify more relevant features in the data of interest, instead of just differentiating specific image features that are not representative of the problem. For instance, the detectors trained in the MNIST dataset with only classes 2 and 3 have trouble differentiating 5 and 8 which have similar shapes. Nonetheless, we see that in MNIST and CIFAR-10, ATHENA-FL was able to reach up to 7% and 10% accuracy compared to the best deeper model.

6.4 ATHENA-FL Communication Evaluation

The objective of this experiment is to evaluate the cost of communication between the clients and the aggregation server while training the one-*versus*-all models and a deep neural network to classify multiple classes. The communication evaluation considers the total number of bytes transmitted on average to perform model training. Let T_{dec} be the size in bytes of the detector, e_{dec} be the probability density function that indicates the number of epochs necessary for the detector to converge, $\mathbb{E}[e_{dec}]$ the expected values of e_{dec}, and R the number of existing detectors in the problem of classification, the average amount of bytes transmitted per client $\overline{B_{ova}}$

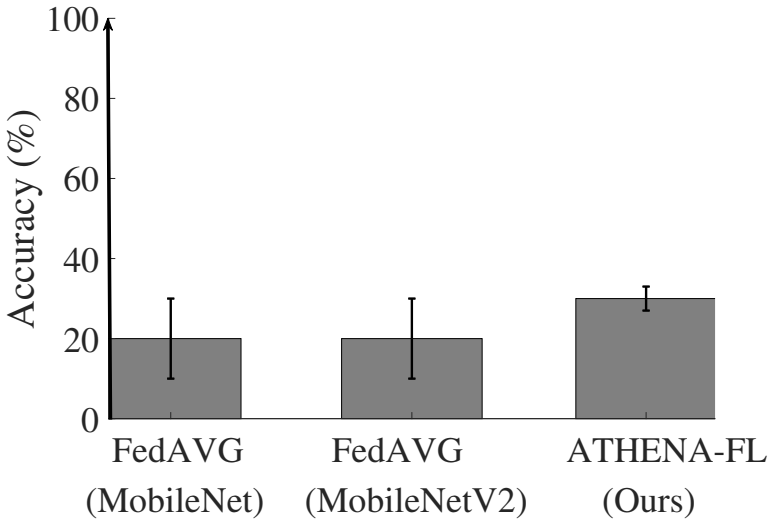

Figure 6.13: Final models' accuracy on the CIFAR-10 dataset with Non-IID distribution of 2 classes per client.

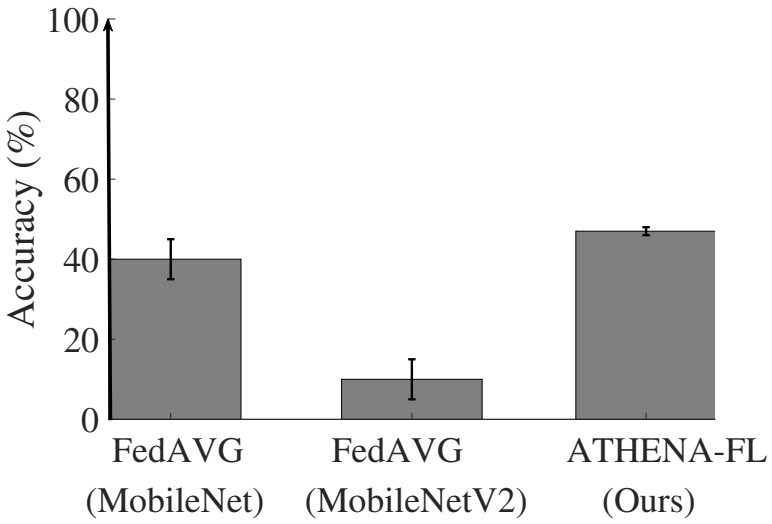

Figure 6.14: Final models' accuracy on the MNIST dataset with Non-IID distribution of 2 classes per client.

can be expressed by the following relation, for the one-versus-all learning model:

$$\overline{B_{ova}} = T_{dec} \times R \times \mathbb{E}[e_{dec}]. \tag{6.1}$$

On the other hand, the communication cost of neural network architectures for multi-class classification B_{mcc} is given by:

$$\overline{B_{mcc}} = T_{mcc} \times \mathbb{E}[e_{mcc}], \tag{6.2}$$

where T_{mcc} is the size and e_{mcc} is the probability density function that indicates the number of epochs required for the convergence of the multiclass classification

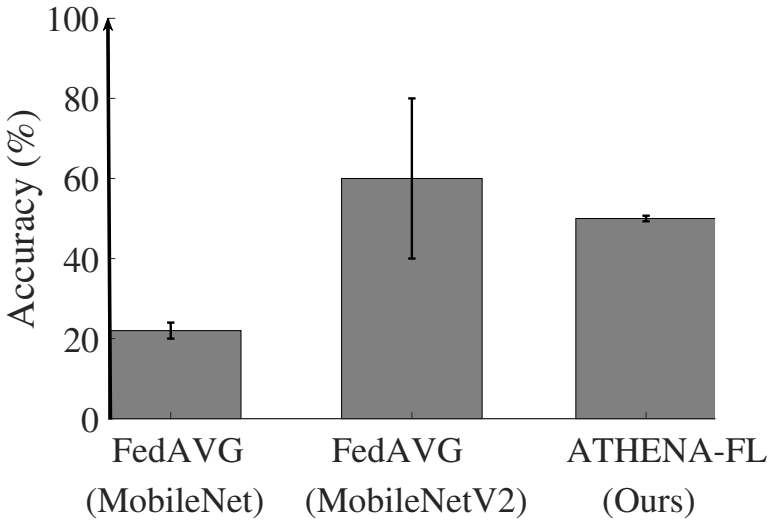

Figure 6.15: Final models' accuracy on the FMNIST dataset with Non-IID distribution of 2 classes per client.

model, while $\mathbb{E}[e_{mcc}]$ is the expected value of e_{mcc}. The values of T_{dec} and T_{mcc} are deterministic and depend on the neural network architecture used. Figure 6.16 shows the comparison between the size in bytes of the different neural network architectures evaluated in this work. Using the Keras library to create the models, each detector has $T_{dec} = 551$k bytes. Nonetheless, the MobileNetV2 architecture has 9.2MB, while MobileNet and Xception have 13MB and 81MB, respectively. The number of detectors is also a deterministic value that depends only on the dataset used for evaluation, which in the tested cases has the value $R = 10$. Furthermore, the expected values of e_{dec} and e_{mcc}, $\mathbb{E}[e_{dec}]$ and $\mathbb{E}[e_{mcc}]$, are obtained experimentally, through training performance along the epochs. When the model does not show a significant improvement concerning previous epochs, the training is considered finished. Analyzing the costs presented, so that the communication cost of the one-*versus*-all model is lower or equivalent to the multiclass neural network model, it is necessary that the total amount of bytes transmitted for training the models are related as follows: $\overline{B_{ova}} \leq \overline{B_{mcc}}$. To verify the validity of the relationship and thus the communication efficiency of the OvA model, it is necessary to estimate the values of $\mathbb{E}[e_{dec}]$ and $\mathbb{E}[e_{mcc}]$.

The criterion to determine the epoch at which the model converged consists of comparing the accuracy in the current epoch with the accuracy in the previous epoch. If the epoch is in the limit of $tol = 0.001\%$ the previous epoch, we consider that the model has converged. The adopted value of tol allows considering cases in which the model is stable only over a short interval. Increasing the value of tol implies reducing the number of epochs required for convergence, but it decreases the final classification performance. The opposite occurs when we reduce the parameter.

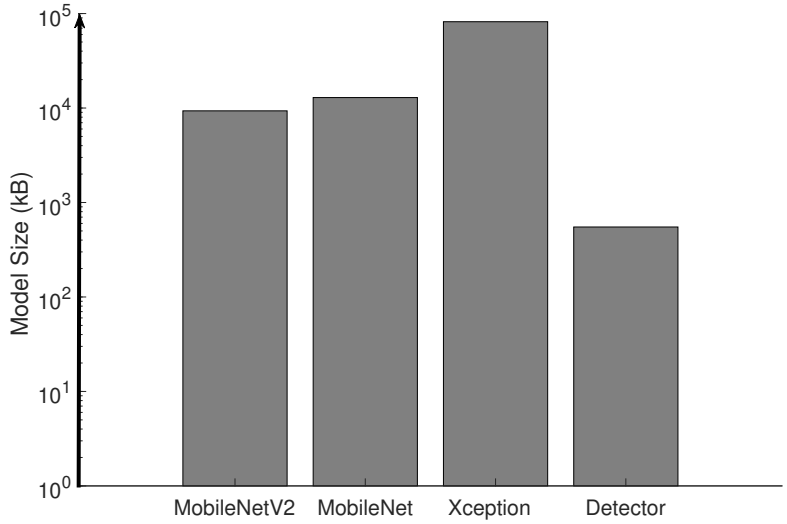

Figure 6.16: Comparison between the size in bytes of different neural network architectures. The model's size directly depends on the number of parameters used by the architecture.

The accuracy results from analysis over the training epochs presented in Section 6.3.1 according to the adopted convergence criterion, allows the estimation of the value $\mathbb{E}[e_{dec}]$ and $\mathbb{E}[e_{mcc}]$ for the datasets in each experimental setup. We show the results of ATHENA-FL compared with MobileNet in Table 6.5. The MobileNetV2 model needs hundreds of epochs to converge. For conciseness, we only show the results of MobileNet due to its faster convergence in tens of epochs in most scenarios.

After estimating the number of epochs necessary for the models to converge, it is possible to compare them in its scenario using properly the Equations 6.1 and 6.2. The results show how many bytes each neural network model needs to transmit during the training, and finally, we compare the economy of both approaches computing $1 - \overline{B_{ova}}/\overline{B_{mcc}}$.

In the IID setting, the detectors converge in a few epochs, reducing over 75% of bytes transmitted. A second highlight is the data used to train and test the models. ATHENA-FL converges faster than MobileNet in MNIST and FMNIST, which have grayscale images and, therefore, use a single channel to represent the pixels.

We observe that in all scenarios, ATHENA-FL reduces the total amount of bytes transmitted during the training execution. In the worst-case scenario, our approach saves at least 25% of the bytes transmitted. For the best case scenario, we can reduce by approximately 97% the communication requirements for training the model.

Table 6.5: Communication requirements, Total Data Transmitted (TDT), to train the models until convergence in different datasets and sample distribution settings.

		IID		**Non-IID 5**		**Non-IID 2**	
Dataset	**Model**	$\mathbb{E}[e]$ (#)	**TDT (MB)**	$\mathbb{E}[e]$ (#)	**TDT (MB)**	$\mathbb{E}[e]$ (#)	**TDT (MB)**
CIFAR-10	ATHENA-FL	7	37.635	12	64.517	14	75.270
	FedAVG	30	378.106	23	289.881	8	100.828
	Economy (%)	-	**90.05**	-	**77.74**	-	**25.35**
MNIST	ATHENA-FL	4	21.506	24	129.034	4	21.506
	FedAVG	7	88.225	58	731.005	65	819.230
	Economy (%)	-	**75.62**	-	**82.35**	-	**97.37**
FMNIST	ATHENA-FL	5	26.88	81	435.49	21	112.91
	FedAVG	44	554.56	151	1903.10	80	1008.30
	Economy (%)	-	**95.15**	-	**77.12**	-	**88.8**

Chapter 7

Conclusion and Future Work

We have proposed a system to increase the performance of federated learning when clients have non-IID data distributions. Our contribution is divided into two parts. Firstly, we propose a clustering strategy that increases client's accuracy by clustering them to train the FL model over IID datasets. Secondly, we extended the first scenario for intra-cluster information sharing. Therefore, in the first part, clients have only specific models while the second part allows them to have more generic models and classify data outside their cluster.

Both proposals maintains the privacy of the client's data given that it uses the weight values of the final layer of the client's neural networks for clustering. As the final layers of the neural network positively relate to the existing classes in the dataset, it is possible to send only a part of the neural network to perform the clustering of the clients. The data size is significantly lower than if we use the whole weight matrix, and this is an important feature because it could save energy, network resources and reduce the latency for data transfer.

The DBSCAN clustering algorithm, which requires only the minimum distance between samples as a hyperparameter, provides high ability to detect the existing clusters even when clients have IID datasets. Besides, the proposed system outperformed FedAVG when the clients' datasets are non-IID in all the evaluated scenarios. The generated models exhibited high classification performance and few epochs to converge, approximately ten global epochs. The traditional model, however, achieved low classification performance even after 100 global epochs for non-IID data. The accuracy showed an improvement of approximately 16% in the best case.

Furthermore, the system applies the one-*versus*-all model to create a generic classifier, which shares knowledge among clusters. The results show that communication during the training epochs is efficient, reducing between 25.35% and 97.37% total transmitted bytes compared to the FedAVG approach with the MobileNet neural network. The accuracy of the OvA model depends on the data distribution scenario used during the training step, being up to 10.9% higher in the best case

and presenting a better performance than the model trained with FedAVG in most of the evaluated scenarios.

In future works, we will implement the proposal's security guarantees, providing a protocol tolerant to malicious participants' updates. Also, the proposal will be extended to non-stationary scenarios in future work to deal with the data concept drift, where the aggregation server can change clusters on demand.

References

[1] LIU, B., DING, M., SHAHAM, S., et al. "When Machine Learning Meets Privacy: A Survey and Outlook", *ACM Computing Surveys (CSUR)*, v. 54, n. 2, pp. 1–36, 2021.

[2] MCMAHAN, B., MOORE, E., RAMAGE, D., et al. "Communication-efficient Learning of Deep Networks from Decentralized Data", *Artificial Intelligence and Statistics*, pp. 1273–1282, 2017.

[3] DE SOUZA, L. A. C., REBELLO, G. A. F., CAMILO, G. F., et al. "DFedForest: Decentralized Federated Forest". In: *International Conference on Blockchain*, pp. 90–97. IEEE, 2020. doi: 10.1109/Blockchain50366.2020.00019.

[4] DJENOURI, Y., MICHALAK, T. P., LIN, J. C.-W. "Federated Deep Learning for Smart City Edge-based Applications", *Future Generation Computer Systems*, v. 147, pp. 350–359, 2023.

[5] LI, D., LAI, J., WANG, R., et al. "Ubiquitous Intelligent Federated Learning Privacy-preserving Scheme under Edge Computing", *Future Generation Computer Systems*, v. 144, pp. 205–218, 2023.

[6] SINGH, S., RATHORE, S., ALFARRAJ, O., et al. "A Framework for Privacy-preservation of IoT Healthcare Data using Federated Learning and Blockchain Technology", *Future Generation Computer Systems*, v. 129, pp. 380–388, 2022.

[7] MA, X., ZHU, J., LIN, Z., et al. "A State-of-the-Art Survey on Solving Non-IID Data in Federated Learning", *Future Generation Computer Systems*, v. 135, pp. 244–258, 2022.

[8] ZHAO, Y., LI, M., LAI, L., et al. "Federated Learning with Non-IID Data", *arXiv preprint arXiv:1806.00582*, 2018.

[9] OUYANG, X., XIE, Z., ZHOU, J., et al. "ClusterFL: a Similarity-Aware Federated Learning System for Human Activity Recognition". In: *Proceedings*

of the International Conference on Mobile Systems, Applications, and Services, pp. 54–66, 2021.

[10] WANG, H., KAPLAN, Z., NIU, D., et al. "Optimizing Federated Learning on Non-IID Data with Reinforcement Learning". In: *IEEE INFOCOM*, pp. 1698–1707, 2020.

[11] ESTER, M., KRIEGEL, H.-P., SANDER, J., et al. "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise". In: *KDD*, pp. 226–231, 1996.

[12] ANKERST, M., BREUNIG, M. M., KRIEGEL, H.-P., et al. "OPTICS: Ordering Points to Identify the Clustering Structure", *ACM Sigmod record*, pp. 49–60, 1999.

[13] MACQUEEN, J. "Some Methods for Classification and Analysis of Multivariate Observations". In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297, 1967.

[14] HOWARD, A. G., ZHU, M., CHEN, B., et al. "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications", *arXiv preprint arXiv:1704.04861*, 2017.

[15] NETO, H. N. C., DUSPARIC, I., MATTOS, D. M., et al. "FedSA: Accelerating Intrusion Detection in Collaborative Environments with Federated Simulated Annealing". In: *International Conference on Network Softwarization (NetSoft)*, pp. 420–428. IEEE, 2022.

[16] FU, L., ZHANG, H., GAO, G., et al. "Client Selection in Federated Learning: Principles, Challenges, and Opportunities", *arXiv preprint arXiv:2211.01549*, pp. 1–8, 2022.

[17] TAN, A. Z., YU, H., CUI, L., et al. "Towards Personalized Federated Learning", *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–17, 2022.

[18] LUO, B., XIAO, W., WANG, S., et al. "Tackling System and Statistical Heterogeneity for Federated Learning with Adaptive Client Sampling". In: *IEEE INFOCOM*, pp. 1739–1748, 2022.

[19] LAI, F., ZHU, X., MADHYASTHA, H. V., et al. "Oort: Efficient Federated Learning via Guided Participant Selection". In: *USENIX OSDI*, pp. 19–35, 2021.

[20] LI, T., SAHU, A. K., ZAHEER, M., et al. "Federated Optimization in Heterogeneous Networks", *Proceedings of Machine Learning and Systems*, v. 2, pp. 429–450, 2020.

[21] NISHIO, T., YONETANI, R. "Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge". In: *International Conference on Communications*, pp. 1–7, 2019.

[22] LIU, L., ZHANG, J., SONG, S., et al. "Client-Edge-Cloud Hierarchical Federated Learning". In: *International Conference on Communications*, pp. 1–6, 2020.

[23] QIN, T., CHENG, G., WEI, Y., et al. "Hier-SFL: Client-Edge-Cloud Collaborative Traffic Classification Framework based on Hierarchical Federated Split Learning", *Future Generation Computer Systems*, 2023.

[24] RAI, S., KUMARI, A., PRASAD, D. K. "Client Selection in Federated Learning under Imperfections in Environment", *AI*, v. 3, n. 1, pp. 124–145, 2022.

[25] FRABONI, Y., VIDAL, R., KAMENI, L., et al. "Clustered Sampling: Low-Variance and Improved Representativity for Clients Selection in Federated Learning". In: *International Conference on Machine Learning*, pp. 3407–3416. PMLR, 2021.

[26] LI, H., CAI, Z., WANG, J., et al. "FedTP: Federated Learning by Transformer Personalization", *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[27] ZHONG, Z., OTHERS. "FLEE: A Hierarchical Federated Learning Framework for Distributed Deep Neural Network over Cloud, Edge and End Device", *ACM TIST*, pp. 1–24, 2022. Disponível em: <`https://doi.org/10.1145/3514501`>.

[28] LI, T., HU, S., BEIRAMI, A., et al. "Ditto: Fair and Robust Federated Learning through Personalization". In: *International Conference on Machine Learning*, pp. 6357–6368. PMLR, 2021.

[29] DENNIS, D. K., LI, T., SMITH, V. "Heterogeneity for the Win: One-Shot Federated Clustering", *arXiv preprint arXiv:2103.00697*, 2021.

[30] ZHU, Y., MARKOS, C., ZHAO, R., et al. "FedOVA: One-vs-All Training Method for Federated Learning with Non-IID Data". In: *IEEE IJCNN*, pp. 1–7, 2021.

[31] CHU, D., JAAFAR, W., YANIKOMEROGLU, H. "On the Design of Communication-Efficient Federated Learning for Health Monitoring", *IEEE GLOBECOM*, pp. 1–6, 2022.

[32] BLONDEL, V. D., OTHERS. "Fast Unfolding of Communities in Large Networks", *Journal of Statistical Mechanics: Theory and Experiment*, pp. 1–12, 2008.

[33] ZENG, D., HU, X., LIU, S., et al. "Stochastic Clustered Federated Learning", *arXiv preprint arXiv:2303.00897*, 2023.

[34] GHOSH, A., CHUNG, J., YIN, D., et al. "An Efficient Framework for Clustered Federated Learning", *arXiv preprint arXiv:2006.04088*, 2020.

[35] DUAN, M., LIU, D., JI, X., et al. "Flexible Clustered Federated Learning for Client-Level Data Distribution Shift", *IEEE Transactions on Parallel and Distributed Systems*, v. 33, n. 11, pp. 2661–2674, 2022.

[36] SATTLER, F., MÜLLER, K.-R., SAMEK, W. "Clustered Federated Learning: Model-Agnostic Distributed Multitask Optimization under Privacy Constraints", *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[37] SANDLER, M., HOWARD, A., ZHU, M., et al. "MobileNetV2: Inverted Residuals and Linear Bottlenecks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510–4520, 2018.

[38] YANG, Q., LIU, Y., CHEN, T., et al. "Federated Machine Learning: Concept and Applications", *Transactions on Intelligent Systems and Technology (TIST)*, v. 10, n. 2, pp. 1–19, 2019.

[39] WANG, J., LIU, Q., LIANG, H., et al. "Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization", *NeurIPS*, v. 33, pp. 7611–7623, 2020.

[40] BEUTEL, D. J., TOPAL, T., MATHUR, A., et al. "Flower: A Friendly Federated Learning Research Framework", *arXiv preprint arXiv:2007.14390*, 2020.

[41] PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., et al. "Scikit-learn: Machine Learning in Python", *Journal of Machine Learning Research*, v. 12, pp. 2825–2830, 2011.

[42] KRIZHEVSKY, A., NAIR, V., HINTON, G. "The CIFAR-10 Dataset", *online: http://www. cs. toronto. edu/kriz/cifar. html*, v. 55, n. 5, 2014.

[43] LECUN, Y., CORTES, C., BURGES, C. J. "MNIST Handwritten Digit Database". http://yann.lecun.com/exdb/mnist/, 2010. Disponível em: <http://yann.lecun.com/exdb/mnist/>.

[44] XIAO, H., RASUL, K., VOLLGRAF, R. "Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms", *arXiv preprint arXiv:1708.07747*, 2017.

[45] SANGUINETI, M. "Cats VS Dogs Convolutional Classifier". Towards Data Science, 2021. Acessado em 25 de agosto de 2023https://towardsdatascience.com/cats-vs-dogs-convolutional-classifier-44ec04c8eb7a.

Appendix A

List of Publications

The following works were published during the elaboration of this master thesis:

- Couto, R. S., Mattos D. M. F., Moraes, I. M., Caminha, P. H. C., Medeiros, D. S. V., de Souza, L. A. C., Táparo, F. G., Campista, M. E. M., and Costa L. H. M. K. - "Gerenciamento e Orquestração de Serviços em O-RAN: Inteligência, Tendências e Desafios", in Minicursos do XLI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2023), Brasília, Brazil, May 2023.

- **de Souza, L. A. C.,** Camilo, G. F., Rebello, G. A. F., Sammarco, M., Campista, M. E. M., Costa, L. H. M. K. - "ATHENA-FL: Evitando a Heterogeneidade Estatística através do Um-contra-Todos no Aprendizado Federado". In Anais do VII Workshop de Computação Urbana (pp. 40-53). (2023, May). SBC. **Honorable Mention**.

- **de Souza, L. A. C.**, Rebello, G. A. F., Camilo, G. F., Campista, M. E. M., Costa, L. H. M. K. - "GITI-CB: Gestao de Identidade com Troca de Informaçoes entre Correntes de Blocos". In Anais do VI Workshop em Blockchain: Teoria, Tecnologias e Aplicações (pp. 43-56). (2023, May). SBC.

- Camilo, G. F., Rebello, G. A. F., **de Souza, L. A. C.**, Campista, M. E. M., Costa, L. H. M. K. - "Posicionamento Lucrativo de Nós e Criaçao de Rotas de Baixo Custo na Rede Relâmpago". In Anais do XLI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (pp. 57-70). (2023, May). SBC. **Honorable Mention**.

- **de Souza, L. A. C.**, Camilo, G. F., Campista, M. E. M., Costa, L. H. M., Duarte, O. C. M. - "Enhancing Automatic Attack Detection through Spectral Decomposition of Network Flows". In Global Communications Conference (GLOBECOM) (pp. 2074-2079). (2022, December). IEEE.

- Camilo, G. F., Rebello, G. A. F., **de Souza, L. A. C.**, Potop-Butucaru, M., Amorim, M. D., Campista, M. E. M., Costa, L. H. M. K. - "Topological Evolution Analysis of Payment Channels in the Lightning Network". In Latin-American Conference on Communications (LATINCOM) (pp. 1-6). (2022, November). IEEE.

- Rebello, G. A. F., Camilo, G. F., Guimarães, L. C., **de Souza, L. A. C.**, Duarte, O. C. M. - "Security and Performance Analysis of Quorum-based Blockchain Consensus Protocols". In 6th Cyber Security in Networking Conference (CSNet) (pp. 1-7). (2022, October). IEEE.

- **de Souza, L. A. C.**, Camilo, G. F., Rebello, G. A. F., Campista, M. E. M., Costa, L. H. M. K. - "Gestão Segura e Escalável de Identidades através de Múltiplas Corrente de Blocos". In Anais Estendidos do XXII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (pp. 167-170). (2022, September). SBC.

- Camilo, G. F., Rebello, G. A. F., **de Souza, L. A. C.**, Potop-Butucaru, M., Amorim, M. D., Campista, M. E. M., Costa, L. H. M. K. - "Análise da Evolução Topológica da Rede Lightning de Canais de Pagamento". In Anais do XXII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (pp. 71-84). (2022, September). SBC.

- Camilo, G. F., Rebello, G. A. F., **de Souza, L. A. C.**, Thomaz, G. A., Potop-Butucaru, M., Amorim, M. D., Costa, L. H. M. K. - "Redes de Canais de Pagamento: Provendo Escalabilidade para Pagamentos em Criptomoedas". (2022). SBC.

- Camilo, G. F., Rebello, G. A. F., de Souza, L. A. C., Thomaz, G. A., Potop-Butucaru, M., Amorim, M. D., Campista, M. E. M., and Costa L. H. M. K. - "Redes de Canais de Pagamento: Provendo Escalabilidade para Pagamentos em Criptomoedas", in Minicursos do XL Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2022), Fortaleza, Brazil, May 2022.

- **de Souza, L. A. C.**, Camilo, G. F., Sammarco, M., Campista, M. E. M., Costa, L. H. M. - "Aprendizado Federado com Agrupamento Hierárquico de Clientes para Aumento da Acurácia". In Anais do XL Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (pp. 545-558). (May 2022). SBC.

- Camilo, G. F., **de Souza, L. A. C.**, Campista, M. E. M., Costa, L. H. M., Duarte, O. C. M. - "A Blockchain-based System for Secure and Distributed

Virtual Network Functions Orchestration". In International Conference on Communications (ICC) (pp. 347-352). (2022, May). IEEE.

- Thomaz, G. A., Camilo, G. F., **de Souza, L. A. C.**, Duarte, O. C. M. - "Architecture and Performance Comparison of Permissioned Blockchains Platforms for Smart Contracts". In Global Communications Conference (GLOBECOM) (pp. 1-6). (2021, December). IEEE.

- Alvarenga, I. D., Camilo, G. F., **de Souza, L. A. C.**, Duarte, O. C. M. - "DAGSec: A Hybrid Distributed Ledger Architecture for the Secure Management of the Internet of Things". In International Conference on Blockchain (Blockchain) (pp. 266-271). (2021, December). IEEE.

- Rebello, G. A. F., Camilo, G. F., Guimaraes, L. C., **de Souza, L. A. C.**, Thomaz, G. A., Duarte, O. C. M. - "A Security and Performance Analysis of Proof-based Consensus Protocols". Annals of Telecommunications. (2021). 1-21.

- Thomaz, G. A., Camilo, G. F., **de Souza, L. A. C.**, Duarte, O. C. M. - "Uma Análise Comparativa da Arquitetura e Desempenho de Plataformas de Corrente de Blocos Permissionadas para Contratos Inteligentes". In Anais do IV Workshop em Blockchain: Teoria, Tecnologias e Aplicaçoes (pp. 114-127). (2021, August). SBC.