*Journal of Internet Services and Applications, 2024, 15:1,*doi: 10.5753/jisa.2024.4044

MESFLA: Model Efficiency through Selective Federated Learning Algorithm

Alex Barros [**Federal University of Pará** | ***alexbarros@ufpa.br***]
Rafael Veiga [**Federal University of Pará** | ***rafael.teixeira.silva@icen.ufpa.br***]
Renan Morais [**Federal University of Pará** | ***renan.morais@itec.ufpa.br***]
Denis Rosário [**Federal University of Pará** | ***denis@ufpa.br***]
Eduardo Cerqueira [**Federal University of Pará** | ***cerqueira@ufpa.br***]

Computer Science Faculty, Federal University of Pará, Belém 66075-110, Brazil.

Received: 20 January 2024 • **Accepted:** 13 August 2024 • **Published:** 22 October 2024

Abstract
Integrating big data and deep learning across various applications has significantly enhanced intelligence and efficiency in our daily lives. However, it also requires extensive data sharing, raising significant communication and privacy concerns. In this context, Federated Learning (FL) emerges as a promising solution to enable collaborative model training while preserving the privacy and autonomy of participating clients. FL facilitates collaborative model training by enabling data to be trained locally on devices, eliminating the need for individual information sharing among clients. A client selection mechanism strategically chooses a subset of participating clients to contribute to the model training in each learning round. However, an efficient selection of clients to participate in the training process directly impacts model convergence/accuracy and the overall communication load on the network. In addition, FL faces challenges when dealing with non-Independent and Non-Identically Distributed (non-IID) data, where the diversity in data distribution often leads to reduced classification accuracy. Hence, designing an efficient client selection mechanism in a scenario with non-IID data is essential, but it is still an open issue. This article proposes a Model Efficiency through Selective Federated Learning Algorithm called MESFLA. The mechanism employs a Centered Kernel Alignment (CKA) algorithm to search for similar models based on data weight or similarity between models, *i.e.*, grouping participants with comparable data distributions or learning objectives. Afterward, MESFLA selects the clients with more relevance in each group based on data weight and entropy. Our comprehensive evaluation across multiple datasets, including MNIST, CIFAR-10, and CIFAR-100, demonstrates MESFLA's superior performance over traditional FL algorithms. Our results show an accuracy improvement and a minor loss in each client aggregation of the new global model sent to clients with a difference of 3 rounds using the Data Weight in comparison with the other selection methods.

Keywords: Federated Learning, Cluster, Aggregation Methods;

1 Introduction

Machine learning (ML) has experienced unprecedented growth since the world is more connected and eager to use Artificial Intelligence (AI) to improve people's lives for a wide range of applications, such as next-word prediction, healthcare and entertainment [Kusano *et al.*, 2023; Zhao *et al.*, 2023]. These applications mainly run on cell phones and tablets, the primary computing devices for many people [Smestad and Li, 2023]. According to the International Telecommunication Union (ITU) [, ITU], there were more than 8.58 billion mobile subscriptions worldwide in 2022. In this context, the world is generating unprecedented data through connected devices [Smestad and Li, 2023]. With a wealth of available data and the fact that ML models are voraciously hungry for data, AI has become ubiquitous and essential among critical stakeholders, making our lives more intelligent and efficient.

Among all these recent advances, the concerns and fears about the impacts of AI in our lives are increasing faster, especially related to communication and privacy with the proliferation of decision support systems being deployed [Gao *et al.*, 2023]. For instance, existing AI approaches are still based on cloud-centric architecture where data is stored and processed centrally. This, in turn, leads to communication problems, such as unacceptable latency and high communication costs. Furthermore, privacy and data security emerge as crucial concerns in future AI applications since without serious consideration of privacy, sensitive data is susceptible to disclosures, cyberattacks, and risks. In this context, it is believed that the future of AI and cloud computing will be distributed at the network edge [Zhang *et al.*, 2023], where many AI applications must avoid sending private data to a centralized server.

Federated Learning (FL) emerges as a crucial solution to provide a privacy-preserving property with reduced communication cost for future ML applications [McMahan *et al.*, 2017]. FL provides a decentralized ML paradigm, which leaves the training data distributed on mobile devices and learns a shared model by aggregating locally computed updates [Hu *et al.*, 2023]. In FL operation, the client selection mechanism deployed at the aggregation server selects a subset of participating devices, also called clients, to contribute to the model training in each communication round [Xiong *et al.*, 2023]. These clients receive the global model, conduct training based on their local data, and then share their model parameters instead of transmitting their raw sensing data, keeping the raw data on the client devices [Barros *et al.*, 2021]. On an aggregation server, only locally computed updates and analysis results are received, and aggregated for an improved global model that benefits from distributed learning, such as achieving improved accuracy or better generalization [Veiga *et al.*, 2023]. Hence, FL allows continuous learning by adapting the ML model without sharing raw data [Chen *et al.*, 2020].

The mechanism must choose a subset of clients eligible to participate in the upcoming learning rounds, where the mechanism must remove clients who do not add value to the training to improve the global model while reducing the waste of computation resources [Smestad and Li, 2023]. In this sense, selecting clients with valuable data samples is crucial, as it both saves computational resources and excludes data that no longer benefits the global model's improvement. [Xiong *et al.*, 2023]. Random client selection is commonly used in traditional FL algorithms, but client devices often have heterogeneous data distribution, and randomly selecting them might lead to some bias, low-performance metrics or some challenges to achieve ML model's convergence. It is also important to find techniques that reduce the number of communication rounds and the size of the transmitted messages [Li *et al.*, 2020]. The client selection mechanism is a critical component of the FL training process, as it ensures a diverse and representative data sample from various clients. Hence, client selection is not an easy task, and there are many issues to consider for defining the best client selection.

Furthermore, the client selection mechanism faces challenges when dealing with non-independent and non-identically distributed (non-IID) data scenarios, as datasets may have statistical heterogeneity. This diversity in data distribution often leads to reduced classification accuracy. One approach to mitigating the impact of non-IID data on model gradients involves assessing the similarity between trained model representations. In this way, clustering algorithms are an essential tool to group participants with comparable data distributions or learning objectives, creating clusters with similar models. Hence, it is possible to select clients with more relevance in each group, *i.e.*, clients that contribute to the model training in each learning round, improving the accuracy and convergence.

In this article, we propose a cluster-based client selection mechanism for FL applications that considers model and data size information, called MESFLA. The mechanism considers the clusterization and client selection steps. At the clusterization step, the mechanism considers a clusterization algorithm to group clients based on the weight data of each client. Afterward, at the client selection step, the mechanism selects the clients with more relevance in each group. In addition, the aggregation server evaluates the rate of change in accuracy to decide when to return the new aggregated Model to all clients, saving network resources while maintaining the global Model during many rounds. Evaluation results demonstrate that MESFLA achieves model efficiency by using significantly fewer communication rounds between clients and aggregation servers and exhibiting a faster recovery system than other approaches. The main research contributions of this article can be summarized as follows:

1. Cluster-based client selection mechanism for FL applications that considers Model and data size information.
2. A detailed performance evaluation where we demonstrate that our approach can maintain the fast convergence and also contribute to reducing the number of rounds between clients and the aggregation server.

The remainder of this article is structured as follows. Section 2 presents an overview of works that explore similar proposals related to client selection and clustering in an FL approach. Section 3 describes our methodology for extracting the client model, clustering, and selecting them for the training step. Section 4 explores the simulation model and the results of our method comparison. Section 5 analyzes the obtained results and explains them. Section 6 presents the concluding remarks and future works.

2 Related Works

Random client selection, the standard method, yields suboptimal results. Several client selection mechanisms have been proposed recently, each of them applied to some specific scenarios, architecture, or application. For instance, Qu *et al.* [2022] proposed the Context-aware Online Client Selection (COCS), which allows clients to use their computational information, bandwidth, and distance to select a subset of clients to maximize the training utilities. Qiao *et al.* [2022] also proposed a context-aware approach, where the central server evaluates the significance of clients based on both the content of their local updates and communication channel states.

Ami *et al.* [2023] introduced the Multi-Armed Bandit (MAB) for client selection, which formulates the trade-off between the training latency and the model's generalization. of the model. MAB reduces the training process time while increasing the model's generalization ability by avoiding overfitting. The authors proposed selecting clients based on a time-varying reward influenced by the history of previous selections. Sousa *et al.* [2023] introduced an entropy-based client selection for vehicular FL environments to address issues arising from non-IID data distributions. The proposed method is compared to a random selection mechanism in both IID and non-IID scenarios, and scenarios with random client drops.

Ouyang *et al.* [2021] introduced a cluster indicator matrix indicating the similarity of users called ClusterFL. In this work, the aggregation server drops stragglers which converge slower and clients which are less related to others in each cluster. This work aims to reduce the overall communication overhead while maintaining the overall accuracy performance. Sattler *et al.* [2020] presented a Clustered FL (CFL) that combines privacy constraints by applying an encryption mechanism with clustering. This work uses the geometric structures of a surface to calculate the positioning and grouping of some clients. This method helps to achieve

Table 1. Summary of related works and comparison with our MESFLA mechanism

Papers	**Cluster Approach**	**Selection Method**	**Communication Cost Mitigation**
[Qu *et al.*, 2022]	X	Computational and client-ES pairs transmission information	Hierarchical FL reduces communication to the cloud server, using nearby edge servers instead
[Qiao *et al.*, 2022]	X	Client Local Updates and Communication Channel	Selects the client subset based on both the wireless channel states and the content of model updates
[Ami *et al.*, 2023]	X	Time-varying reward influenced by the history of previous selections	Developed a time-varying reward function that captures the client latency
[Sousa *et al.*, 2023]	X	Entropy-based for Vehicular FL environments	Focused only in the challenges posed by non-IID data in vehicular networks
[Ouyang *et al.*, 2021]	✓	Similarity of users and drop stragglers	Reduces overall communication overhead through cluster-wise straggler dropout and correlation-based node selection
[Sattler *et al.*, 2020]	✓	Similarity with client weights after the convergence	Focused only on properties of the FL loss surface to group the client population into clusters
[Nguyen *et al.*, 2020]	✓	Re-weighting mechanism of updated parameter	Communication and heterogeneity is improved with the re-weighting mechanism of updated parameters
[Ghosh *et al.*, 2020]	✓	Similarity based on weight sharing technique	Focused only on convergence of Iterative Federated Clustering Algorithm (IFCA)
[Albaseer *et al.*, 2021]	✓	Cosine similarity between the weight-updates	Schedule the clients based on their round latency and exploits the bandwidth reuse
[Huang *et al.*, 2023]	✓	Least confidence, margin, entropy, vote entropy, and loss	Reduces communication overhead requiring less client participation in the learning process
[Li *et al.*, 2022]	✓	Soft clustering with overlapping clusters	Focused only on combining the strengths of soft clustering and IFCA
MESFLA	✓	Data weight and similarity(entropy) cluster with convergence rate monitor	Combines clusterization and metrics evaluation to reduce communication between aggregation server and clients

improved performance due to greater equality between the trained client weights and their modes.

Nguyen *et al.* [2020] proposed a fast convergent FL algorithm called FOLB, which performs intelligent sampling of devices in each round to improve the expected convergence speed. The clients are sampled based on the re-weighting mechanism of updated parameters received from participating devices in every round. FOLB is capable of handling the heterogeneity of device communication and computation by using their model to create groups with the ability to achieve faster convergence. This work also evaluates the heterogeneity of devices by only assigning random images from a fixed number of different digits to each device with MNIST Dataset. The authors compared the number of rounds necessary to reach a certain accuracy level to indicate its communication network reduction cost.

Ghosh *et al.* [2020] proposed an Iterative Federated Clustering Algorithm (IFCA), which considers user cluster identity estimation and gradient descent-based model parameter optimization. In this way, IFCA distributes and clusters users, where diverse groups pursue individual learning tasks collaboratively. The clusters may represent groups of users interested in different categories of news. The authors used data augmentation to simulate an environment where the data on different worker machines are generated from different probabilistic distributions.

Albaseer *et al.* [2021] introduced Clustered Federated Multitask Learning (CFL) as an efficient method for developing reliable specialized models in non-IID scenarios. The cluster is designed using the cosine similarity between the weight updates of diverse clients. All clients have an equal probability of participating in the training phase, even if they have wrong channels or low computational capabilities to obtain unbiased models. The client's expected latency is the parameter used to schedule the model uploading, which can be the bottleneck of this approach. Huang *et al.* [2023] propose an active client selection for clustered federated Learning (CFL) to address data heterogeneity by customizing models for different client groups. This work combines many CFL client selection strategies, including calculating the least confidence, margin, entropy, vote entropy, and loss to find the most informative and helpful clients.

Fraboni *et al.* [2021] introduced a clustered sampling for client selection, demonstrating better client representativity and reduced variance in aggregation weights. They introduced two clustering approaches based on sample size and model similarity. The approach seamlessly integrates into FL implementations without client-side operations, remaining compatible with privacy and communication reduction technologies. Li *et al.* [2022] proposed a soft clustering method with the clients being partitioned into overlapping clusters, and the information of each participating client is used by multiple clusters simultaneously during each round. Client selection contributes to improving the model's results and even indirectly contributes to reducing communication since only a portion of the clients participate in the process.

Table 1 summarizes the main characteristics of reviewed studies regarding the challenge of select clients and summarizes the existing works regarding the Clustering approach, Clients Selection Method, and Communication Cost Mitigation. Based on the state-of-the-art analysis, we argue that it is important to group users based on the statistical features of their datasets to discover the relationships within user datasets, mitigating the impact of non-IID data. In this context, clustering algorithms are an important tool for grouping participants with comparable data distributions or learning objectives, creating clusters with similar models. In addition, we could select significant or relevant users in each group, which could be achieved by considering data weight and similarity between trained model representations to mitigate the effects of non-IID data. This approach helps to select the most relevant clients for each training round, reducing the amount of data transmitted and improving overall communication efficiency. In addition, a large number of devices may participate in FL over unreliable networks, leading to significant communication costs and performance bottlenecks. In this way, it is important to only send the new model when accuracy reduces its improvement rate. To the best of our knowledge, only MESFLA considers every critical characteristic previously mentioned not provided by the existing client selection mechanism.

3 Model Efficiency through Selective Federated Learning Algorithm (MESFLA)

This section introduces a cluster-based client selection mechanism for FL applications that considers model and data size information called MESFLA. The proposed mechanism consists of two steps, namely, clusterization and client selection. At the clusterization step, the mechanism considers a clusterization algorithm to group clients based on the weight data of each client. Afterwards, at the client selection step, the mechanism selects the clients with more relevance in the each group. In the following, we introduce the system model and MESFLA operations.

3.1 Scenario Overview for Clustering

We considered the traditional FL architecture, where at each communication round, a set of Client k is selected to receive the global model, perform the training based on its Dataset Dn. The clients improve the Model Mn by training with its specific data. Afterward, the model updates, *i.e.*, learned parameters or gradients, are sent periodically to the central server [Song *et al.*, 2022]. The aggregation server applies a given aggregation policy, such as The Federated Learning Average (FedAVG). For instance, FedAVG computes an average of the shared local models at edge servers to produce an accurate global model. Finally, the updated global model is distributed to the selected clients [Lobato *et al.*, 2022].

The client selection mechanism must select a subset of clients to participate in the upcoming learning rounds. The mechanism must select a given client k with valuable samples to reduce the waste of computation resources, and it must remove the clients whose data are no longer critical for the model training. In addition, FL is subject to data that are not independent or have different statistical distributions, *i.e.*, non-IID data scenarios. This statistical heterogeneity of Dataset Dn results in lower classification accuracy, where non-IID data over model contributions can be addressed by measuring the similarity between trained model representations. In this context, clustering algorithms play a vital role in FL by grouping participants with similar data distributions or learning objectives, enabling to group similar models.

Figure 1 shows the MESFLA operation, where each device has its own collected data (D), *i.e.*, the usual preferences and data pattern recorded tipically in that device due to per-

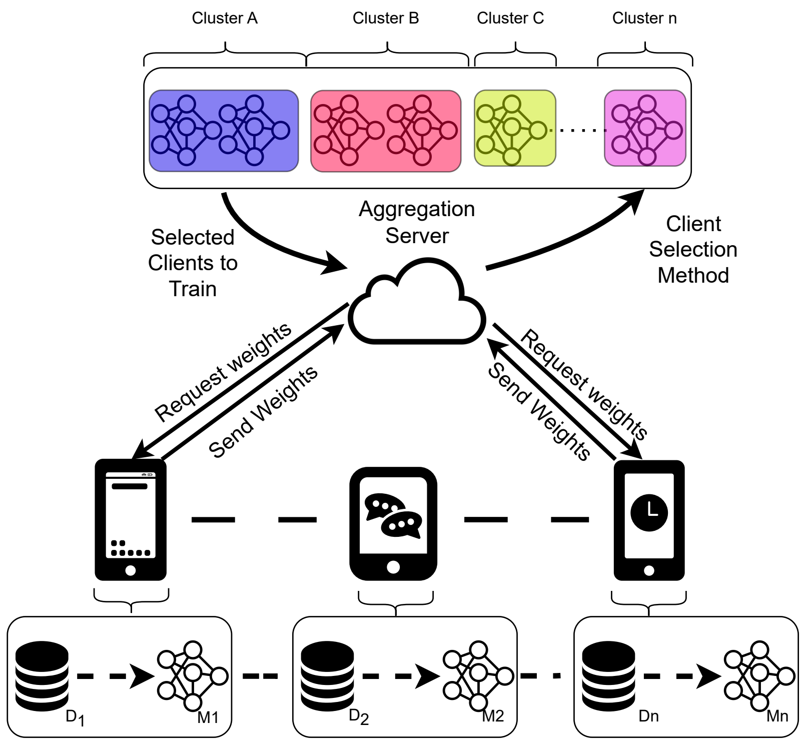

Figure 1. MESFLA operational overview

sonal usage patterns. Each device uses its own collected data *D* to train a local model *M* associated with that device. Then each device sends the trained model together with the dataset size to the central server. In this way, MESFLA collects all local model *M* from all the available devices, and the central server aggregates each model based on a given aggregation policy, such as FedAVG, providing a global model. In addition, the central server clusters that received model *M* were into similar groups/clusters. Afterward, MESFLA selects the clients with more relevance in each group, which is the client with the large dataset. Table 2 summarizes the list of main symbols used to introduce the MESFLA mechanism.

3.2 MESFLA Operation

MESFLA Operation is divided into **Clusterization** and **Client Selection** steps. In the Clusterization step, we consider Centered Kernel Alignment (CKA) implementation on the models to determine which group each user belongs to, *i.e.*, the set of clients *k* converges into groups based on the model similarity after their training [Kornblith *et al.*, 2019]. In its operation, a given client *k* sends its client weight *Wk* to the server, which clusters *C* these parameters with the aim of grouping the set of k clients based on model similarity. For instance, clients have a correlation based on the values of model gradients, which is used to search for the most similar internal representations between models (*i.e.*, clients) [Ouyang *et al.*, 2021]. In this way, CKA provides insights into how well models capture similar patterns of client weight *Wk* and structures in the data, even when trained independently or on different tasks. Algorithm 1 illustrates the Clusterization step performed by the MESFLA mechanism operations.

The CKA uses different methods to measure the similarity between the matrices. In this way, we use the Dot Product-based Similarity, which relies on the Eq. 1 of the relative dots products of the samples where H is the centering matrix. The Eq. 1 presents the empirical estimator for the Hilbert-Schmidt Independence Criterion (HSIC), a non-parametric measure of the dependency between two sets of variables, *i.e.*, X and Y, represented within kernel matrices K and L. The elements $K_{ij} = k(X_i, X_j)$ and $L_{ij} = l(Y_i, Y_j)$ are derived from kernel functions k and l. That means mapping pairs of data points into a high-dimensional feature space, capturing their similarities. The matrix H is a centering ma-

Table 2. List of Symnols

Acronyms	Description
Dn	Dataset
Gm	Global Model
Mn	Model
Wfl	FL model
Wk	Client weight within the cluster
Wi	Cluster weight
Wj	Client weight
C	Set of clients
k	Client
P(k)	Probability of k
Sn	The sum result in the total data from all samples.
Sk	Number of data from that sample
t	Threshold
w	Set of w rounds

trix that adjusts the kernels to have a zero mean. The trace operation, tr$(KHLH)$, in the numerator, captures the alignment of the transformed data points. Divided by the square of the number of observations minus one, $(n-1)^2$, this formula normalizes the value to account for sample size, providing a scaled estimate of dependence robust to the number of data points in the sample.

Quantifying the similarity between features involves calculating the sum of squared dot products across all pairs. Otherwise, CKA involves calculating the alignment between the centered representations of two models by comparing their kernel matrices. The kernel matrix represents the pairwise similarity between data points in the high-dimensional space.

$$HSIC(K,L) = \frac{1}{(n-1)^2} tr(KHLH), \tag{1}$$

The alignment is then normalized to provide a similarity score between 0 and 1 based on Eq. 2, where 1 indicates perfect alignment. The Eq. 2 uses the HSIC to compare the similarity of K and L matrices. That matrix typically represent features in ML models, by normalizing HSIC's measure of dependency between the two sets of variables. The CKA offers a scale-invariant index, ensuring that the similarity assessment is not affected by the magnitude of the data. This normalization, defined as $\sqrt{HSIC(K,K)HSIC(L,L)}$, allows for a direct and equitable comparison of feature representations across different models or data transformations that provices for CKA a valuable tool in the analysis and interpretation of complex machine learning systems.

$$CKA(K,L) = \frac{HSIC(K,L)}{\sqrt{HSIC(K,K)HSIC(L,L)}} \tag{2}$$

Algorithm 1: Clusterization step

```
1  foreach client do
2      Get user weights for the client;
3      $Wfl \leftarrow Wc.append, Client.name$;
   // Cluster all clients' weights
4  Call clusterUsers();
5  return Clustered clients weights;
6  Function clusterUsers():
       // Perform operations on user weights
          for a specific client if needed
7      foreach Wi in Wfl do
8          $listWi \leftarrow + \sum_{k=n}^{W} Wi$;
9          foreach Wk in Wfl do
10             $listWj \leftarrow + \sum_{k=n}^{W} Wj$;
11         if Wi $\geq$ listWi - listWk and Wi $\leq$ listWi +
            listWk then
12             $Clusterlist \leftarrow client.name$;
```

In the client selection step, the edge server considers the data size as the metric to select a given client *k* that is more relevant to each group. For this purpose, the server receives the model *Mn* for each client and assigns a client weight within the *C* weight *Wi*. Afterward, we use the probability of each client entering the training based on its greater relevance than others in the same *C* cluster. In this way, we can use the probability *P(k)* computed based on Eq. 3 we ensure that the best candidates are more likely to participate in the training to prevent overfitting. The *Sk* means the total number of times this *k* has priority, and Sn denotes the sum of times each k appears on this array sum to the *Sk*. This method helps to balance the influence of each client, ensuring that even those with smaller datasets have a fair chance of being selected. This mitigates the risk of overfitting the model to the characteristics of larger datasets alone by balancing the weights with this probability *P(k)*.

$$P(k) = \frac{S_k}{\sum_{n=0}^{N} S_n} \tag{3}$$

Once the model is trained, it is evaluated on a test dataset to determine its performance, where model aggregation increases the accuracy results on each round evaluation because the round behaves like epochs. MESFLA considers a window (*i.e.*, a set of w rounds) to evaluate the variation of the accuracy value, which can be compared with the performance of the model trained in the previous window [Sousa *et al.*, 2023]. Specifically, after w rounds, the aggregation server evaluates if the model accuracy changed less than a given threshold t. If so, the server sends a new model for the clients, reducing the amount of data required to transfer for model transmission. In this way, the MESFLA relies on a convergence point as a way to reduce the bandwidth consumption to transfer a new model at each round. It is important to mention that MESFLA requires a threshold t and a window w, where these values may need to be adjusted based on the specific characteristics of the dataset and the network environment.

4 Evaluation

This section describes the scenario, including the framework, database, and simulation details. We also discuss the obtained results in terms of computational effort, loss, and the global model's accuracy.

4.1 Simulation Description

We conducted simulations using the Personalized Federated Learning Platform, a framework available on GitHub of TsingZ0[1]. We chose this framework due to its flexibility and the various aggregation models that were implemented. In this way, we use such a platform to support our scenario running a server with the following specifications: 13th Gen Intel i9-13900K (32) @ 5.500GHz, eight cores, 128GB RAM. We considered the MNIST dataset for classification in FL, which is a widely used dataset to train and test for model validation, particularly for tasks related to image classification. Specifically, the MNIST dataset is a widely used collection of 28x28 pixel grayscale images of handwritten dig-

[1]TsingZ0/PFL-Non-IID: Personalized federated learning simulation platform (https://github.com/TsingZ0/PFL-Non-IID)

its (0 through 9). MNIST Comprises a training set of 60,000 images and a test set of 10,000 images, which serves as a benchmark for testing and validating ML algorithms. We consider a ML model with two convolutional layers with filter sizes of 5x5. A 2x2 max-pooling operation succeeds each convolutional layer. Table 3 summarizes the main evaluation parameters.

We consider that data exhibit non-IID characteristics due to data heterogeneity across various FL applications. In this sense, numerous distributed ML training approaches encounter substantial degradation of accuracy due to the data quantity and category distribution disparities within non-IID data. In the non-IID context, we use label distribution skew to characterize the local data distribution among clients, resulting in varying label proportions [Ma *et al.*, 2022]. Specifically, we sample a proportion $p_{k,i} \sim Dir(\beta)$ that represents the instances of class k to the client, where $Dir(\beta)$ is a Dirichlet distribution with a concentration parameter $\beta = 0.2$ [Li *et al.*, 2021].

In our evaluation, we analyze the impact of different aggregation policies by comparing their behavior of using a cluster approach with different client selection methods. Specifically, we consider FedAVG and Model-Contrastive Federated Learning (MOON) [Li *et al.*, 2021] as the aggregation policies. The FedAVG uses the client's averages to aggregate their parameters to the newest global model in the FL system, providing fair processing of the weights for all clients who do the training step. For this purpose, we compare the behavior of this strategy with our clustering method. On the other hand, MOON uses the local dataset, such as the weights of each client relevant in the aggregation step, providing efficiency when using the parameter in the update step. Then, they send extra clients' weights relevant to the system data. In other words, each client has different relevant data when their parameters are aggregated in the newest global model Gm. Therefore, the primary approach of the MOON is to declare a disparity of each client's contribution to the global Gm model for better use.

In addition, we evaluate the cluster approach based on CKA with different client selection methods, namely MESFLA Default, MESFLA Data Weight, and MESFLA Entropy Weight. Specifically, MESFLA Default means a random client selection for each group, which serves as a baseline method that simply selects a random subset of clients to participate in each round of training. On the other hand, MESFLA Entropy selects the set of clients based on the number of labels as input to compute the entropy for each user, such as introduced by Sousa *et al.* [2023]. Finally, MESFLA Data Weight means the client selection introduced in Section 3.2.

We compare these algorithms with commonly used metrics for FL classification, namely, processing level of the device, accuracy, and loss. The **Accuracy** metric is obtained by the number of hits (positive) divided by the total number of examples, which is used on data with the examples for each class and when they miss.

In the case of disproportionate classes, it gives a false impression of good performance, delivering a flawed result. The **Loss** metric compares the target and predicted output values, which helps see how the neural network models the training data. The computational effort metric is computed based on the average time of each round and the maximum TeraFlops that an RTX 4090 has.

Table 3. Simulation Parameters

Characteristics	**Values**
Cluster Method	CKA
Total Number of Clients	100
Selection methods	Random Selection, Entropy, and Data size
Number of selection clients	20% clients each round
Server strategies	FedAVG
Dataset	MNIST
Rounds	100 rounds
Local Training	1 epoch
Number of labels	10 labels/classes

4.2 Simulation Results

We start our evaluation by analyzing the behavior of different aggregation strategies, where the ideal result is searching for a clustering method that can work with high accuracy and low loss values, allowing us to understand how good it is in our estimated recovery time. Figure 2 demonstrates each computer processing time in terms of Teraflops for different evaluated categories. The main goal is to achieve the model efficiency performance of the clients by considering time and calculations. By analyzing the results, we can observe that FedAVG costs less in computation processing than MOON, regardless of the client selection approach. For instance, MOON shows more time to process and analyze the new model of FL in each round, which has higher processing. This result shows the best strategy for our approach, leading to a strategy that uses the power of clients' datasets as a relevant argument to evaluate them with more processing recurses. We can also do it on devices with low processing when we know these values are necessary to train them in a high-level cloud. In this way, we need to see how the strategies are forming in the accuracy of these strategies when choosing them.

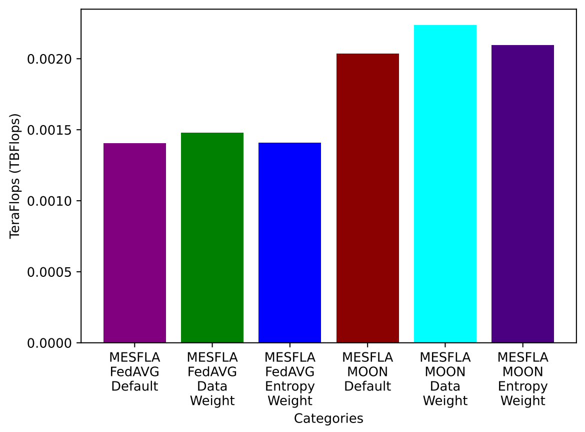

Figure 2. Processing results

It is essential to highlight that the aggregation server evaluates if the model accuracy changed less than a given threshold t. If so the server sends a new model for the clients, which might impact the amount of data transferede for model transmission. Figure 3 shows the accuracy evaluation of different client selection approaches with FedAVG as an aggregation policy in the simulation with a threshold t of 1% convergence in the last five rounds. This result shows how MESFLA Data Weight improves the results with a convergence variety.

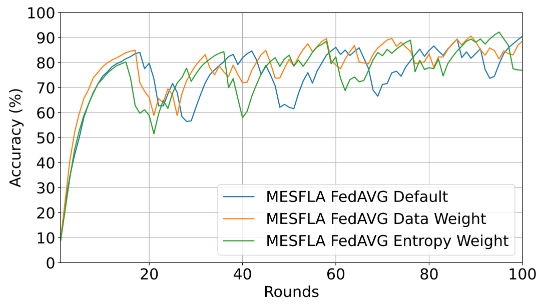

Figure 3. Accuracy results with FedAVG as aggregation policy with 1% of convergence

shows the evaluation of different client selection approaches with FedAVG as an aggregation policy and a threshold of 5% in the last five rounds.

Figure 4 shows the evaluation of different client selection approaches with FedAVG as an aggregation policy and a threshold t of 5% in the last five rounds. It is noticed that accuracy falls each time that the global model Gm is sent to devices for each one of our categories. Our approach evaluates the accuracy of every round and as soon as the convergence reaches five rounds without more than 5% of progression, it sends the global model Gm to the clients. By comparing the results of Figures 3 with 4, we could see that threshold t of 1% proves to have a longer time to convergence than using 5%, as well as the model updates are sent more frequently, as evidenced by the more oscillatory nature of the accuracy lines across rounds. This leads to a higher network cost due to increased transmissions required to achieve this stringent convergence level. On the other hand, Figure 4 considers a threshold t of 5%, leading to a smoother trajectory for accuracy improvements. Hence, this indicates that the model under a 5% threshold t requires fewer updates to maintain or improve accuracy, thereby reducing network expenditures.

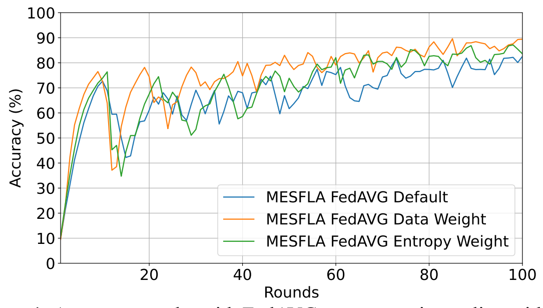

Figure 4. Accuracy results with FedAVG as aggregation policy with 5% of convergence

Figure 5 shows a Loss results with different client selection approaches with FedAVG as aggregation policy for a scenario with convergence threshold t of 5%. There is a round at the beginning of training with a loss value above 0.75 for data weight, followed by other approaches. However, the evaluation goes up again in sequence because the clients receive the global models Gm. In the following rounds, the impact of local model re-training is lower. Figure 5 demonstrates the selection with Data Weight having the best evaluation with less round of training to reach an efficient global model Gm. In addition, the entropy method was close to this value during a few more rounds in means of 3 or 4 rounds more to reach the same evaluation, an improvement of 20% of our FedAVG Default. That could easily rearrange itself after increasing values. In this way, we maintain the random selection (Default) in a random approach to calculate new parameters for the scenario, meaning our baseline to validate the selection improvement.

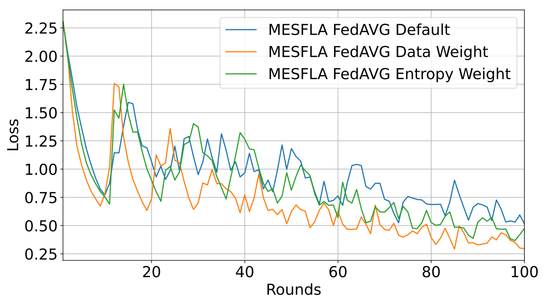

Figure 5. Loss results with FedAVG as aggregation policy with 1% of convergence

Figure 6 shows accuracy results with different client selection approaches with MOON as aggregation policy for a scenario with convergence threshold t of 5%. By analyzing the results, we can observe that the first threshold occurs around 78% in means in each code run, which means an almost 5% improvement compared to FedAVG approach. However, the selection shows another behavior of using this strategy with our approach of selection by Datat weights the data of most of the cases the global model Gm does not reach the threshold t in the middle rounds, becoming a more stable system with around 80 to 90 accuracy during rounds 50 and 80. Moreover, in another way, this strategy also sends the global model Gm to the clients a few times.

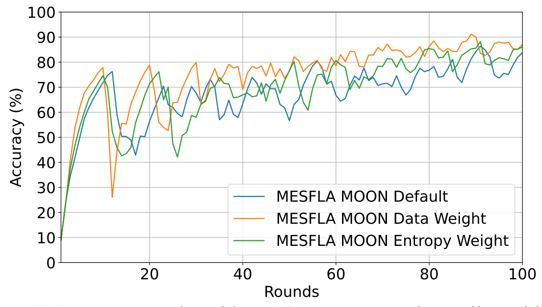

Figure 6. Accuracy results with MOON as aggregation policy with 5% of convergence

Figure 7 shows loss results with different client selection approaches with MOON as aggregation policy for a scenario with convergence threshold t of 5%. By analyzing the results, we can observe that the results of the default method are similar to those of others in the initial rounds. However, the MOON Data Weight gets more useful results after some rounds, having a peak of almost 2 of Loss after the first time they reach the threshold t. However, data weight and entropy evaluation demonstrate less time or a slight loss and, in a few rounds, a total recovery to improve these metrics, reaching the exact evaluation of the last threshold t in only around seven rounds. Further, the result shows an improvement to MESFLA using this strategy and an excellent approach when treating discalceate data in clients.

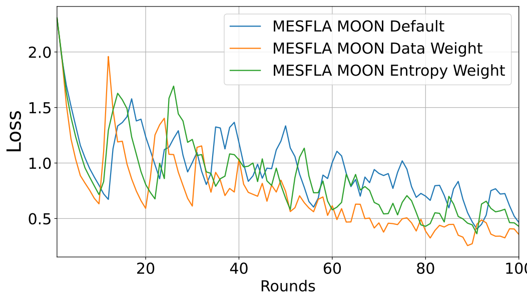

Figure 7. Loss results with MOON as aggregation policy with 5% of convergence

Table 4 summarizes the number of rounds required to converge the model with different client selection approaches and FedAVG and MOON as aggregation policy. By analyzing the results, we can see that the FedAVG requires more convergence rounds than MOON training. This is because the MOON update weight, which is not necessarily needed in all upgrades, transforms into a problem to reach our convergence of 5 rounds of improving by 5 %. It is important to mention that there is a variance in the results, as we can see in Figure 6, which compares the rounds around 40 and 60. We observe an occasional worsening in the evaluation, primarily when novel data for the global model Gm is not used. This complication arises when the best fit for training is not employed.

After these results, we must compare the primary purpose of MESFLA and the maintenance of the results using fewer sendings of the global model Gm to the client side. We search for a "good" evaluation as we have a low convergence time. . However, we can see that the FedAVG has more convergence steps during the MOON training. That was caused by his update weight, which is not necessarily needed in all upgrades, transforming into a problem to reach our convergence of 5 rounds of improving by 5 %.

Table 4. Simulation Results of Convergence sending.

	Category	Rounds to Convergence
FedAVG	MESFLA FedAVG	12.6
	MESFLA FedAVG Data Weight	14.6
	MESFLA FedAVG Entropy Weight	12.6
MOON	MESFLA MOON	8.3
	MESFLA MOON Data Weight	11.0
	MESFLA MOON Entropy Weight	9.4

5 Discussion

The threshold t of 1% as a criterion within the last five rounds can enhance model precision, as shown in Figures 4 and 3. However, this approach requires additional communication rounds, potentially leading to increased communication costs and processing consumption. In contrast, the threshold t of 5% achieves quicker model stability, but possibly at the cost of model accuracy. These findings have significant implications for FL deployment, where devices with varying capacities and network conditions can be highly dynamic.

Setting the convergence threshold to 1% typically causes the model to interact and refine its parameters with each round until it achieves a precise level of accuracy. This rigorous standard can enhance model precision by allowing the model to learn and adjust from a greater volume of aggregated updates. However, this meticulous process requires frequent client and server communication to exchange model updates. The Communication cost consideration is crucial in areas where network traffic may be congested, potentially leading to delays and increased operational costs.

The model achieves an acceptable accuracy level with fewer communication rounds with a threshold t of 5%, accelerating model stabilization. This is because the model requires fewer communication rounds to reach an acceptable level of accuracy, allowing for faster deployment of the updated global model. This experience is advantageous in dynamic settings where faster response times might be necessary. It may come at the expense of the model's overall accuracy. Such a trade-off might be acceptable in applications where speed is critical and absolute precision is less vital. In summary, a lower convergence threshold could strain the network and the devices, leading to processing use and potential network bottlenecks. Conversely, a higher threshold could save resources but compromise the quality of the service.

Furthermore, the loss patterns observed in Figures 5 and 7 highlight client selection methods' significant impact on both convergence speed and model generalization across diverse data distributions. Data Weight and Entropy-based selections achieve a reduction in training rounds and loss values, accelerating the stabilization of the model. The reduction in loss under the Data Weight strategy suggests that this method is particularly effective in scenarios where data distributions are heterogeneous. By assigning weights to clients based on the relevance of their data, the aggregation process becomes more representative of the overall data landscape, enhancing the model's generalization capabilities. Limiting the convergence threshold to 5% prevents the models from overfitting, as they do not continue to learn once an acceptable loss is reached.

The FedAVG approach demonstrates a more consistent decline in loss values when Data Weight is applied, indicating a smoother and potentially more stable path to convergence. This behavior has benefits scenarios with heterogeneously distributed data, where the learning process favors a gradual and steady approach. The more uniform decline in loss values across rounds suggests that FedAVG, combined with Data Weight optimization, may be more suitable for environments where predictable and reliable model performance is critical. Therefore, the choice between the two strategies should be based on the application's specific needs. That is when the priority is stability and reliability with FedAVG or faster adaptability with MOON.

The convergence data offers direct conclusions about the communication cost in FL algorithms. For FedAVG, applying Data Weight increases the communication rounds needed, potentially leading to higher network usage and potentially more significant costs. The Entropy Weight method, however, does not impact the frequency of communication compared to the baseline, suggesting it might be a more efficient choice within the FedAVG framework when communication overhead is a concern. In contrast, MOON's mean convergence figures are consistently lower, reflecting its communication efficiency. The MOON is a preferable algorithm for scenarios with limited bandwidth or where minimizing communication is crucial. The results invite a focused discussion on choosing FL strategies that align with specific network efficiency needs. The MOON highlights its Entropy Weight variant, potentially more suitable for communication-constrained environments.

6 Conclusion and Future Works

In this article, we introduced a Model Efficiency through Selective FL Algorithm called MESFLA. It aims to better select clients to improve the model, while reducing the number of rounds of communication between clients and the central server. The initial step creates clusters that consider the data weight or similarity between models. The client model is transmitted to the server, where the model is retained to prevent decay, facilitating faster convergence. This approach allows for a more accurate and efficient comparison of accuracy and loss metrics, as the global model sends only when it reaches a 5% convergence point, eliminating the need for frequent model updates in each round. Afterward, the mechanism selects the clients with more relevance in each group. The obtained results show that our method called MESFLA improved its values according to the rounds with a clustering method to select the better clients to train in each training round. However, using MESFLA to select and to retain the global model with the client users and with the server aggregation without sending his global model demonstrates a valid strategy for fast convergence.

In future works, we aim to improve the client selection by searching for aways to focus on new metrics of the default FL. For instance, the bias of the client selection shows an approach that can be improved to select clients with more data and a better entropy simultaneously. This means that entropy is not only directly to his value but also to his size. In this way, we must improve the selection by using a weighting scheme to calculate the most relevant client to provide better results. Furthermore, it is essential to train the MESFLA under a scenario with client failure or malicious clients to analyze the resiliency of the client selection mechanism. Finally, the local update is also one possible future work, as the FL needs to add on a training step. This needs to facilitate clients' faster training in a global model, as FEDALA shows his approach by using weights to determine how much training each client will initially do in the first round.

Acknowledgments

This research was partially funded by CNPq grant #405940/2022-0 and CAPES grant #88887.954253/2024-00. This study was also funded, in part, by the São Paulo Research Foundation (FAPESP), Brazil, under Process Number #2023/00673-7. This project was supported by the Brazilian Ministry of Science, Technology, and Innovations, with resources from Law nº 8,248, of October 23, 1991, within the scope of PPI-SOFTEX, coordinated by Softex, and published under Arquitetura Cognitiva (Phase 3), DOU 01245.003479/2024-10. This work used generative artificial intelligence technology only in the stages of grammar review, typos, and vocabulary in the introduction, related works sections and in the description of the functioning of the proposed algorithm. All content is our sole responsibility.

Authors' Contributions

A.B. and RV conceived and planned this study and experiments. A.B., R.V. and R.M. carried out the experiments, the simulations and computed the results. A.B., R.V, D.R., E.C. contributed to the interpretation of the results. A.B. took the lead in writing the manuscript with final review of D.R and E.C. All authors provided critical feedback and helped shape the research, analysis and manuscript.

Competing interests

The authors declare that they have no competing interests.

References

Albaseer, A., Abdallah, M., Al-Fuqaha, A., and Erbad, A. (2021). Client selection approach in support of clustered federated learning over wireless edge networks. In *2021 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE. DOI: 10.48550/arXiv.2108.08768.

Ami, D. B., Cohen, K., and Zhao, Q. (2023). Client selection for generalization in accelerated federated learning: A multi-armed bandit approach. DOI: 10.48550/arXiv.2303.10373.

Barros, A., Rosário, D., Cerqueira, E., and Fonseca, N. (2021). A strategy to the reduction of communication overhead and overfitting in federated learning. In *Proceedings of the 26th Workshop on Management and Operation of Networks and Service (WGRS)*, pages 1–13, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/wgrs.2021.17181.

Chen, Y., Sun, X., and Jin, Y. (2020). Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation. *IEEE Transactions on Neural Networks and Learning Systems*, 31(10):4229–4238. DOI: https://doi.org/10.1109/TNNLS.2019.2953131.

Fraboni, Y., Vidal, R., Kameni, L., and Lorenzi, M. (2021). Clustered sampling: Low-variance and improved representativity for clients selection in federated learning. In

International Conference on Machine Learning, pages 3407–3416. PMLR. DOI: 10.48550/arXiv.2105.05883.

Gao, H., Thai, M. T., and Wu, J. (2023). When decentralized optimization meets federated learning. *IEEE network*, 37(5):233–239. DOI: 10.1109/MNET.132.2200530.

Ghosh, A., Chung, J., Yin, D., and Ramchandran, K. (2020). An efficient framework for clustered federated learning. *Advances in Neural Information Processing Systems*, 33:19586–19597. DOI: 10.48550/arXiv.2006.04088.

Hu, X., Li, R., Ning, Y., Ota, K., and Wang, L. (2023). A Data Sharing Scheme Based on Federated Learning in IoV. *IEEE Transactions on Vehicular Technology*, pages 1–13. DOI: 10.1109/TVT.2023.3266100.

Huang, H., Shi, W., Feng, Y., Niu, C., Cheng, G., Huang, J., and Liu, Z. (2023). Active client selection for clustered federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–15. DOI: 10.1109/tnnls.2023.3294295.

(ITU), I. T. U. Itu statistics. Available at: `https://www.itu.int/en/ITU-D/Statistics/Pages/stat/default.aspx` accessed on 11/29/2023.

Kornblith, S., Norouzi, M., Lee, H., and Hinton, G. (2019). Similarity of neural network representations revisited. In *International conference on machine learning*, pages 3519–3529. PMLR. DOI: 10.48550/arXiv.1905.00414.

Kusano, K. D., Scanlon, J. M., Chen, Y.-H., McMurry, T. L., Chen, R., Gode, T., and Victor, T. (2023). Comparison of waymo rider-only crash data to human benchmarks at 7.1 million miles. DOI: 10.48550/arXiv.2312.12675.

Li, C., Li, G., and Varshney, P. K. (2022). Federated learning with soft clustering. *IEEE Internet of Things Journal*, 9(10):7773–7782. DOI: 10.1109/JIOT.2021.3113927.

Li, Q., He, B., and Song, D. (2021). Model-contrastive federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10713–10722. DOI: 10.48550/arXiv.2103.16257.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. (2020). Federated optimization in heterogeneous networks. DOI: 10.48550/arXiv.1812.06127.

Lobato, W., Costa, J. B. D. D., Souza, A. M. d., Rosário, D., Sommer, C., and Villas, L. A. (2022). FLEXE: Investigating Federated Learning in Connected Autonomous Vehicle Simulations. In *IEEE 96th Vehicular Technology Conference (VTC-Fall)*. IEEE. DOI: 10.1109/VTC2022-Fall57202.2022.10012905.

Ma, X., Zhu, J., Lin, Z., Chen, S., and Qin, Y. (2022). A state-of-the-art survey on solving non-iid data in federated learning. *Future Generation Computer Systems*, 135:244–258. DOI: 10.1016/j.future.2022.05.003.

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. DOI: 10.48550/arXiv.1602.05629.

Nguyen, H. T., Sehwag, V., Hosseinalipour, S., Brinton, C. G., Chiang, M., and Poor, H. V. (2020). Fast-convergent federated learning. *IEEE Journal on Selected Areas in Communications*, 39(1):201–218. DOI: 10.48550/arXiv.2007.13137.

Ouyang, X., Xie, Z., Zhou, J., Huang, J., and Xing, G. (2021). Clusterfl: a similarity-aware federated learning system for human activity recognition. In *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services*, pages 54–66. DOI: 10.1145/3458864.3467681.

Qiao, Z., Shen, Y., Yu, X., Zhang, J., Song, S., and Letaief, K. B. (2022). Content-aware client selection for federated learning in wireless networks. In *2022 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, pages 49–54. DOI: 10.1109/MeditCom55741.2022.9928665.

Qu, Z., Duan, R., Chen, L., Xu, J., Lu, Z., and Liu, Y. (2022). Context-aware online client selection for hierarchical federated learning. *IEEE Transactions on Parallel and Distributed Systems*, 33(12):4353–4367. DOI: 10.1109/TPDS.2022.3186960.

Sattler, F., Müller, K.-R., and Samek, W. (2020). Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE transactions on neural networks and learning systems*, 32(8):3710–3722. DOI: 10.48550/arXiv.1910.01991.

Smestad, C. and Li, J. (2023). A systematic literature review on client selection in federated learning. In *Proceedings of the 27th International Conference on Evaluation and Assessment in Software Engineering*, EASE '23, page 2–11, New York, NY, USA. Association for Computing Machinery. DOI: 10.1145/3593434.3593438.

Song, R., Zhou, L., Lakshminarasimhan, V., Festag, A., and Knoll, A. (2022). Federated Learning Framework Coping with Hierarchical Heterogeneity in Cooperative ITS. In *IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. DOI: 10.1109/ITSC55140.2022.9922064.

Sousa, J. L. R., Lobato, W., Rosário, D., Cerqueira, E., and Villas, L. A. (2023). Entropy-based client selection mechanism for vehicular federated environments. In *Proceedings of the 22nd Workshop on Performance of Computer and Communication Systems*, pages 37–48. SBC. DOI: 10.5753/wperformance.2023.230700.

Veiga, R., Both, C., Medeiros, I., Rosário, D., and Cerqueira, E. (2023). A federated learning approach for authentication and user identification based on behavioral biometrics. In *Proceedings of the 41st Brazilian Symposium on Computer Networks and Distributed Systems*, pages 15–28, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/sbrc.2023.536.

Xiong, Y., Wang, R., Cheng, M., Yu, F., and Hsieh, C.-J. (2023). Feddm: Iterative distribution matching for communication-efficient federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16323–16332. DOI: 10.48550/arXiv.2207.09653.

Zhang, X., Liu, J., Hu, T., Chang, Z., Zhang, Y., and Min, G. (2023). Federated learning-assisted vehicular edge computing: Architecture and research directions. *IEEE Vehicular Technology Magazine*, pages 2–11. DOI: 10.1109/MVT.2023.3297793.

Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., Liu, P., Nie, J.-Y., and Wen, J.-R. (2023). A survey of large language models. DOI: 10.48550/arXiv.2303.18223.