Hierarchical Clustering of Nodes for Accuracy Increase in Federated Learning

Lucas Airam C. de Souza[1], Gustavo F. Camilo[1],
Miguel Elias M. Campista[1], Luís Henrique M. K. Costa[1]

Universidade Federal do Rio de Janeiro (UFRJ), Brazil[1]

***Abstract*—Federated learning performance depends on the data distribution, presenting lower performance in scenarios where clients hold heterogeneous data. We propose a hierarchical client clustering system to mitigate the convergence obstacles of federated learning in non-Independent and Identically Distributed (IID) scenarios. We identify in the server the existing clusters executing an unsupervised clustering algorithm on the bias vector of the last layer of the clients' neural network. We evaluate three clustering algorithms: K-Means, DBSCAN, and OPTICS. The DBSCAN algorithm demonstrated better clustering results, correctly identifying the clients' clusters in IID and non-IID data distributions. Finally, the results show an increase of model accuracy by up to 16% compared to traditional federated learning non-IID scenarios.**

I. Introduction

Federated learning (FL), proposed by Google [1], has become popular among researchers and industry due to its ability to create machine learning models while preserving users' data privacy [2]. 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.

The most widely used algorithm for creating federated models is FedAVG (Federated Averaging), introduced in Google's federated learning proposal. The algorithm uses a client-server architecture, in which the global server shares a model, and the clients return to the server only the trained model weights. This procedure prevents the clients from sharing their data and increases the system's privacy. Nevertheless, the non-Independent and Identically Distributed (non-IID) distribution of the clients' training data reduces the final model performance when trained with FedAVG [3].

This paper 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 is similar. The proposal uses the last-layer neural network weights of the clients as information to perform clustering to preserve the client's privacy. We use the last-layer neural network weights, as it maintains statistical relationships with the clients' private data without revealing them [4]. Each cluster trains a personalized model with independent hyperparameters and parameters, allowing high classification performance on specific tasks. Our proposal can easily be extended to identify network attacks and improve system privacy with a high accuracy level.

We analyze different clustering models and show the advantage of adopting the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [5] 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) [6] and K-Means [7] algorithms. Furthermore, the results show that the proposed approach performs at least equal 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 high classification performance.

II. Related Work

The time to convergence of the global model in federated learning is attributed to three main factors: communication latency, processing capacity, and data representation. The first two factors are related to the user device, while the third depends on the data distribution collected and stored by the client. The random client selection, however, shown in the original proposal of federated learning [1], ignores these three relevant characteristics. Therefore, several research groups present proposals to reduce the convergence time of federated learning and improve the final performance of the generated model. The main improvement approaches include client selection and personalized models. Usually, client selection relies on device characteristics or clients' data. The goal is to decide the best subset of clients for the current training epoch, increasing the average model performance. On the other hand, model personalization allows the creation of models specialized in clients' characteristics, consequently improving local model performance.

A. Client Selection for Efficient Training

Luo *et al.* [8] and Lai *et al.* [9] propose client selection schemes that optimize the time to convergence in federated learning environments. The authors argue that selection based

[1]https://github.com/GTA-UFRJ-team/Hierachical-Federated-Learning

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. Thus, the authors simultaneously consider the characteristics of the devices and the distribution of the collected data to reduce the convergence time of the global model.

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 [10]. 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 with a predefined fraction of each cluster to achieve faster model convergence.

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.

B. Personalized Models in Federated Learning

Dennis *et al.* [11] takes 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.

IFCA (Iterative Federated Clustering Algorithm) [12] is a proposal for clustering clients to personalize models. In the proposal, clients are responsible for choosing their clusters. In addition, the authors propose the employment 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. Delegating the process of identifying clusters to the clients, however, the environment can be susceptible to malicious behavior and requires more computational cost from the clients' devices. Another disadvantage of this proposal is to assume that the number of clusters is previously known, which may be unfeasible and can either overestimate or underestimate the number of existing clusters.

CFL [13] is a proposal that recursively partitions federated learning clients into more homogeneous clusters to mitigate 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.

Fraboni *et al.* propose a clustering sampling method for client selection [14]. 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 is not efficient.

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. The proposed approach aims at clustering clients with homogeneous data. Thus, each computed cluster holds IID data, benefiting the system participants individually. Furthermore, clustering clients can improve model convergence in the cluster and increase the final performance for specific learning tasks. 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.

III. Hierarchical Client Clustering System

This section presents the details of the proposal 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 possible 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 particular for a cluster of clients with IID data, improving the system's overall performance. Thus, instead of having one global model, the proposed system has n personalized models, called cluster models, for each one of the n existing clusters sharing IID data.

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.

A. Proposed System

The proposed system for training federated models is composed of an initialization followed by two phases. The division of clients aims to minimize the heterogeneity of the data used for training the global model. For this, during the system initialization, the server performs a global round 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, which is to uniquely allocate clients to 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 to the same homogeneous cluster. Note that cluster creation occurs without the need to access the client's samples, which is a requirement for keeping privacy. After cluster creation, 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 $k \geq 1$ specific models, where k is the number of clusters generated by the server. In addition, clients can generate on-demand combinations of cluster models to form generic models that are capable of classifying generated samples into distinct distributions. Unlike traditional federated learning, the proposal calls the federated learning server a selection and aggregation server, because, in addition to aggregating the results, the server is responsible for selecting and storing each client's cluster.

B. System initialization

The steps in Fig. 1 are performed to initialize the system. In the first step, the server shares the test model with an initial set of 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 clusters. 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. Adding new clients to a cluster only affects training if the model has not converged. After model convergence, new clients only receive the final model from the cluster. Furthermore, the proposal predicts a fixed number of clusters after initialization, assuming that clients have stationary data. If all clients in a cluster fail simultaneously, 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. The clustering algorithm is not run 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 algorithms before defining 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.

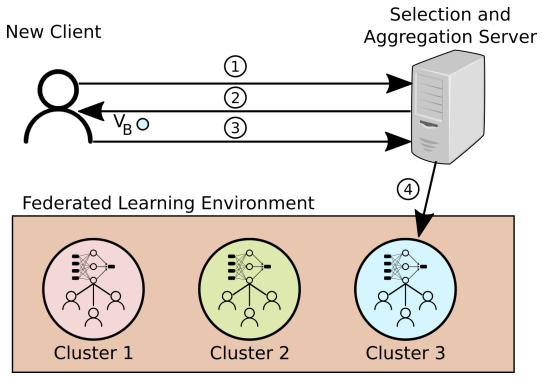

Figure 1. Initialization of the proposed system. (1) The initial set of clients receives the test model. (2) Clients adjust the neural network weights and return the *bias* vector from the last layer of the model to the server. (3) The selection and aggregation server runs the clustering algorithm to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

C. Dynamics of Clients in the System

The entry of new clients into the environment is shown in Fig. 2. The new client who wants to participate in federated training asks the selection and aggregation server for a new cluster. The server sends the client a test template, which is identical for all new clients. 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 a cluster. Finally, federated learning is traditionally performed independently in each cluster.

Clients are clustered according to the model update sent to the aggregation server. Nevertheless, it was experimentally verified that the final layers of the model retain more information about the data. Thus, one can reduce the communication cost of the system, by sending only part of the deep neural network to the server.

Fig. 2 details the steps developed by the proposal for the creation of system models. The client requests the server to be included in a cluster. The server sends the test model to the client to obtain statistical information. This step is fundamental 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

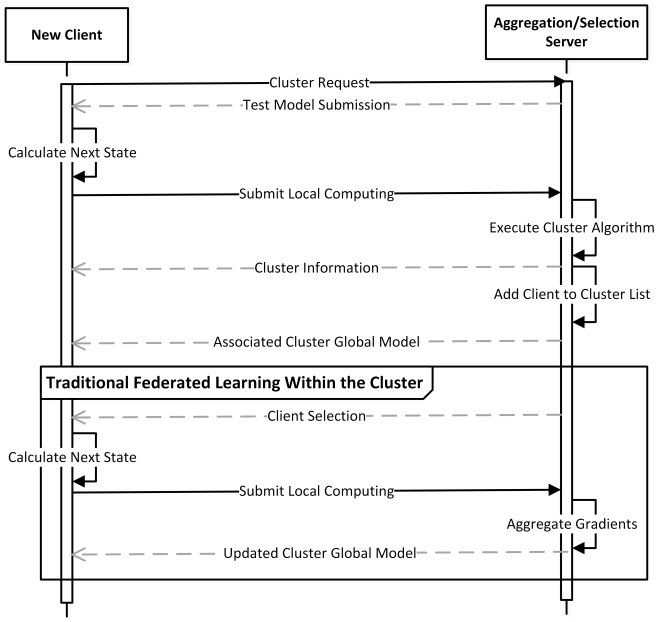

Figure 2. Proposed execution diagram. The first step is to allocate a new client to an existing cluster. After the client is allocated to a cluster, traditional federated learning is performed in the cluster.

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 m_i model of the cluster 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_i cluster in which the client was allocated. At each E_i epoch of the cluster, 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 p_i weights and sends them to the server for aggregation. Finally, the server sends the updated model m_i to the clients of the cluster g_i and a new epoch, $E_{i'}$, 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 $E_{i_{max}}$.

IV. Prototype Development and Evaluation

A prototype of the system was developed using Python v3.9.1 with flower v0.18.0 for the development of the federated learning environment and the scikit-learn library v1.0.1 to create the cluster models. The experiments were carried out on an Intel i7-8700 CPU 3.20 GHz server with 6 processing cores and 32 GB of RAM. The experimental results of the evaluation of the models present the average obtained among all the clients and a confidence interval of 95%. For training the models and evaluating the performance of the proposal, the dataset CIFAR-10 [15] is used. The dataset consists of 60,000 color images with a dimension of 32x32 belonging to 10 different classes. Each class has the same number of samples. In the experiments, the term traditional federated learning is equivalent to FedAVG without the clusters.

A. 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 [7], OPTICS [6] and DBSCAN [5].

K-Means is a widely adopted algorithm for clustering tasks. Its main hyperparameter is the number of k clusters existing in a given dataset. Nevertheless, defining this hyperparameter in an unknown dataset is an arduous task, as there is a prior need to know the number of existing clusters. Thus, there is a higher computational cost to use an iterative approach and estimate the best number of clusters. 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. Moreover, setting a minimum number of samples per large cluster can still generate heterogeneous clusters. Finally, an alternative to the previous algorithms is DBSCAN. Besides the ability 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 practically establish a direct relationship between the clients' replies to define if they have IID data.

The clustering results show that the algorithm that best divides clients when varying the scenarios was DBSCAN using a minimum distance of 0.0279 between client vectors. In the three scenarios, DBSCAN optimally clustered the clients, overcoming OPTICS in the IID scenario by identifying only one cluster. Furthermore, the three clustering models generated the same clusters in the non-IID scenarios. 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. The K-Means algorithm establishes an optimal 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.

B. 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 k of K-Means was initially set to 5 was set to 5. This hyperparameter is decremented until clusters with the minimum number of participants are established. So, through this approach, Experiment II uses $k = 3$. 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.

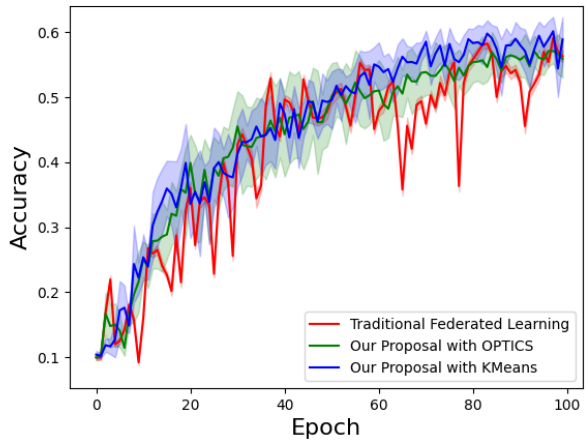

Figure 3. Accuracy evolution as a function of global epoch for customers with IID datasets. The red line represents the traditional approach of federated learning, while the blue and the green lines represent the proposal of this paper with clustering through DBSCAN and OPTICS, respectively.

The result of Fig. 3 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 3 demonstrates that the behavior in each cluster is similar to the average behavior shown in Fig. 3.

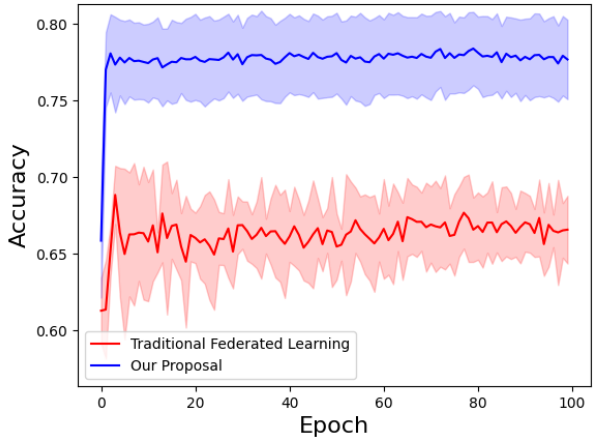

Figure 4. Evolution of the test accuracy as a function of global epoch for clients with datasets that contain only two classes. The red line represents the traditional approach, while the blue line represents the proposal of this paper.

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. To do this, the 6,000 samples of a class are split among five clients to create balanced datasets. Fig. 4 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. 5 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.

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

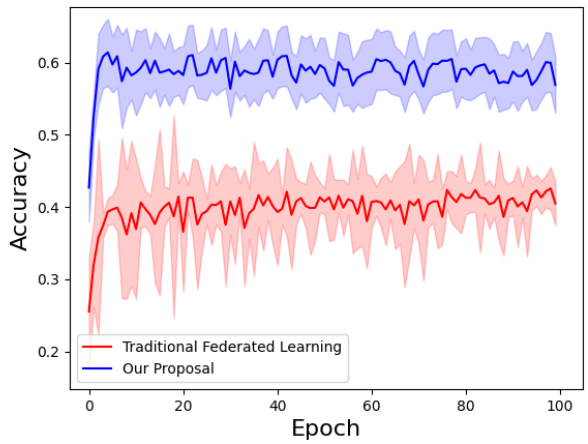

Figure 5. Test accuracy evolution as a function of overall epoch for clients with datasets containing only five classes. The red line represents the traditional approach, while the blue line represents the proposal in this paper.

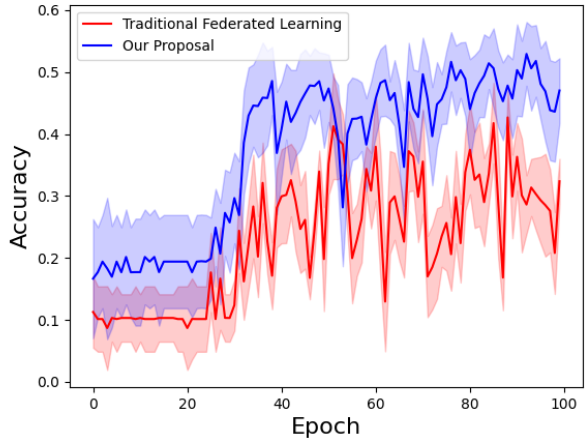

Figure 6. Evolution of the test accuracy as a function of global epoch for clients with datasets that contain only two classes. The clients' datasets are generated with the LDA using $\alpha - 0.5$. The red line represents the traditional approach, while the blue line represents our current proposal.

the Label-based Dirichlet Partition (LDA), which generates the data based on the number of clients and non-IID degree. Figure 6 shows the results of Experiment V. Our approach increases by more than 14% the accuracy compared with traditional federated learning.

V. Conclusion

We proposed a system to increase the performance of federated learning when clients have non-IID data distributions. The DBSCAN clustering algorithm, which requires only the minimum distance between samples as a hyperparameter, provided a high ability to detect the existing clusters even when clients have IID datasets. Besides, the proposed system outperformed traditional federated learning 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. The proposal will be extended to non-stationary scenarios in future work to deal with the concept drift of the data, where the aggregation server can change clusters on demand.

VI. Acknowledgment

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001. This paper was also funded by CNPq, FAPERJ, and Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), Grants 2015/24494-8, 2018/23292-0, 2015/24485-9, 2014/50937-1.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient Learning of Deep Networks from Decentralized Data," *Artificial Intelligence and Statistics*, pp. 1273–1282, 2017.

[2] L. A. C. de Souza *et al.*, "DFedForest: Decentralized Federated Forest," in *International Conference on Blockchain*. IEEE, 2020, pp. 90–97.

[3] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin *et al.*, "Federated Learning with Non-IID Data," *arXiv preprint arXiv:1806.00582*, 2018.

[4] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing Federated Learning on Non-IID Data with Reinforcement Learning," in *INFOCOM*, 2020, pp. 1698–1707.

[5] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in *KDD*, 1996, pp. 226–231.

[6] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "OPTICS: Ordering Points to Identify the Clustering Structure," *ACM Sigmod record*, pp. 49–60, 1999.

[7] J. MacQueen *et al.*, "Some Methods for Classification and Analysis of Multivariate Observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Oakland, CA, USA, 1967, pp. 281–297.

[8] B. Luo *et al.*, "Tackling system and statistical heterogeneity for federated learning with adaptive client sampling," in *INFOCOM*. IEEE, 2022, pp. 1739–1748.

[9] F. Lai, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "Oort: Efficient Federated Learning via Guided Participant Selection," in *OSDI*, 2021, pp. 19–35.

[10] S. Rai, A. Kumari, and D. K. Prasad, "Client Selection in Federated Learning under Imperfections in Environment," *AI*, vol. 3, no. 1, pp. 124–145, 2022.

[11] D. K. Dennis, T. Li, and V. Smith, "Heterogeneity for the Win: One-Shot Federated Clustering," *arXiv preprint arXiv:2103.00697*, 2021.

[12] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An Efficient Framework for Clustered Federated Learning," *arXiv preprint arXiv:2006.04088*, 2020.

[13] F. Sattler, K.-R. Müller, and W. Samek, "Clustered Federated Learning: Model-Agnostic Distributed Multitask Optimization under Privacy Constraints," *Transactions on Neural Networks and Learning Systems*, 2020.

[14] Y. Fraboni, R. Vidal, L. Kameni, and M. Lorenzi, "Clustered Sampling: Low-Variance and Improved Representativity for Clients Selection in Federated Learning," in *International Conference on Machine Learning*. PMLR, 2021, pp. 3407–3416.

[15] A. Krizhevsky, G. Hinton *et al.*, "Learning Multiple Layers of Features from Tiny Images," *Citeseer*, 2009.