FedSCCS: Hierarchical Clustering with Multiple Models for Federated Learning

Gabriel U. Talasso[1], Allan M. de Souza[1], Luiz F. Bittencourt[1]
Eduardo Cerqueira[2] Antonio A. F. Loureiro[3], Leandro A. Villas[1]
[1]Institute of Computing - University of Campinas
[2]Federal University of Minas Gerais
[3]Federal University of Para
E-mail: {allanms, bit, lvillas}@unicamp.br, cerqueira@ufpa.br, loureiro@dcc.ufmg.br

***Abstract*—The growth of mobile devices and the increased concern about model privacy have brought several challenges to distributed artificial intelligence scenarios. One of these challenges is related to the heterogeneity of the devices, which generates problems in the generalization of their models and in the management of their resources. Federated learning (FL) is a collaborative approach for training machine learning models by sharing only their locally computed parameters with an aggregation server, instead of the whole dataset. However, FL faces several challenges related to the heterogeneity of data, networks, and devices, contending to converge and optimize the models, and to the communication overhead, creating bottlenecks for the efficiency and performance of the solutions. This paper proposes FedSCCS, an FL-based framework for heterogeneous environments. It clusters devices according to the similarity of their models, thus enabling a more efficient model aggregation and resource efficiency, avoiding convergence and communication overhead issues. Results demonstrate that FedSCCS achieves higher accuracy, outperforming literature solutions and paving the way for more tailored FL solutions.**

I. Introduction

The advent of Artificial Intelligence (AI) has enabled the development of intelligent solutions to enhance the daily lives of millions of people worldwide, transforming how we live, work, and entertain [1, 2]. These solutions are built upon data collection from devices (e.g., smartphones, IoT devices, and vehicles) using physical and virtual sensors. The gathered data is often aggregated and processed by a central entity, where machine learning and AI algorithms are employed, generating knowledge and insights to create intelligent solutions across various fields.

Collecting and sharing data raises concerns about security and privacy since shared data may contain sensitive user information that malicious individuals or other entities could access [3]. This is where Federated Learning (FL) [4] comes into play as a distributed machine learning method that allows collaborative training across multiple devices with a shared model, ensuring that data remains on the user's device and is not transferred to any other entity [2]. This mechanism provides user privacy and allows broader learning through different data distributions.

FL faces challenges related to systems (i.e., devices with different processing and communication technologies) and statistical heterogeneity (i.e., devices with different data distributions) [5]. The uneven data distribution across contrasting devices can lead to a lack of representativity, resulting in a less accurate and equitable model among participants [5]. Moreover, the decentralized nature of FL imposes an additional challenge to the data quality assurance and the detection of potential issues. Consequently, selecting appropriate algorithms and models is essential to ensure thorough data processing and accurate outcomes.

In the literature, studies that propose solutions for heterogeneous environments are based on various approaches to aggregating these users (or clients, devices), which aim to create a unified model from local models. Some solutions categorize these clients based on specific similarity metrics [6, 7, 8, 9], subsequently aggregating cluster models and assigning weights to clusters to construct a global model [10]. Conversely, other solutions explore client selection [11, 12, 13, 14], which also acts as a form of weighting and prioritization for certain groups to train a more generalized model. However, this approach encounters particular challenges: using a single global model can sometimes lead to convergence issues. Moreover, it may also be unable to manage widely divergent clients, potentially introducing deviations in the global model. These techniques are uniformly applied across all models and clusters, thus not allowing for flexibility and tailored adjustments to suit individual client groups.

In that direction, we propose FedSCCS, an FL-based framework designed to cluster clients hierarchically based on their model similarity. FedSCCS enables multi-model management for FL solutions with different degrees of customization, ranging from generic to more specialized scenarios. The results show that FedSCCS improves the performance of the models in a heterogeneous scenario (i.e., with system or statistical heterogeneity), enables a more efficient client selection for FL solution, and reduces the communication cost between the server and the clients.

The remainder of this paper is organized as follows. Section II presents the related work in client selection and clustering solutions for FL, highlighting the solutions' methods. Section III details how FedSCCS works, including its model similarity, hierarchical clustering, and client selection strategies. Section IV analyses the performance of FedSCCS compared with the literature and shows our proposed method's

improvements in performance and efficiency. Finally, Section V concludes this study and presents some future work.

II. Related Work

This section describes a set of related work considering the heterogeneity of both data and devices in FL. Client selection is a crucial technique in FL to reduce communication costs and overhead, improve accuracy, and enhance fairness between the clients [11]. Several studies focused on developing new methods to address the issue. One primary approach is to select random clients, but other methods consider the client availability to train [13] or look at the local data quality [12]. Additionally, Cho et al. [14] proposed Power of Choice (POC), a biased client selection approach that focuses on choosing the worst clients, in terms of performance, to train the model in the next round, showing that it can improve the convergence of the global model. The POC solution has several variants, but the most straightforward method can accelerate the convergence and, at the same time, improve the efficiency of communication in several scenarios.

On the other hand, Clustered Federated Learning (CFL) [6] is a cluster-based solution that uses the similarity between the client's model to split them into clusters to improve performance in heterogeneous scenarios. CFL uses the cosine similarity in the gradients of the local models to create specialized models for a set of clients and proves mathematically that the gradients are a great metric for finding similar data distributions, so this approach is valid and proves the convergence of these scenarios. This method is viewed and applied after the complete training of the models, so it is a post-processing method specializing in the models based on their similarity. However, once in heterogeneous scenarios, the global model model may not converge and predicate the specialization of all clients.

Sener and Savarese [7] introduced FAVOR, a client selection approach that implements a reinforcement learning (RL) algorithm for selecting the best clients per cluster. The RL agent is trained to rank the clients concerning the expected gain in accuracy in the next round and then selects the top-K best clients. In the same work, the authors proposed using the KCenter [15] clustering algorithm in the weights of the client's models and creating 10 clusters, so they perform random client selection for each cluster, demonstrating that this method can reduce the communication cost. However, the KCenter algorithm struggles to perform better than FedAvg [16], especially in scenarios with extreme statistical heterogeneity.

Likewise, Palihawadana et al. [10] proposed FedSim, which uses the similarity between reduced representations of the gradients via Principal Component Analysis (PCA) as a distance metric in clustering algorithms, such as K-Means, to create representative clusters. FedSim selects random clients in each round to form these clusters, randomly and without replacement. Additionally, each cluster has a single model, and a global model aggregates the cluster models. Still, the experiments demonstrate that with high statistical heterogeneity, the model suffers and performs worse than the baselines.

Fraboni et al. [8] presented different methods to sample clients based on clusters. That work proposed two strategies to select the clients: the first is clustering the clients based on the size of the local dataset, and the second is based on the similarity of the models. The algorithm is compared to the standard solution that samples clients with a *Multinomial Distribution* and demonstrates that the proposed approach can improve the model convergence and training performance variance. The second way to use cluster similarity is to create clusters with the hierarchical clustering method and use a metric named *representative gradients* to measure this similarity. That metric calculates the difference between the client's updated and global models and constructs a similarity matrix used in a hierarchical clustering algorithm. In this case, the clusters are used only to select clients for a global model without the possibility of customization by clusters.

Finally, Morafah et al. [9] proposed Federated Learning by Inference Similarity (FLIS), which employs a small global dataset to collect the inferences of the models and calculate, via the Hadamard product, a similarity metric between two models used to create a matrix. This matrix is part of a hierarchical clustering algorithm that separates clients into groups representing local data distribution differences. That approach performs better than several aggregation algorithms, mainly in statistical heterogeneous scenarios. However, in FLIS, the clustering occurs every round when clients must choose which cluster they will use, which increases client-side processing costs. Furthermore, this solution does not select clients to improve communication efficiency.

As mentioned earlier, the solutions incorporate two fundamental design principles: (*i*) clustering clients based on their model similarity, and (*ii*) selecting clients from each cluster to enhance the model performance. However, most of these solutions still operate with a single global model, leading to overall performance degradation, as the clusters' characteristics might vary significantly. With this in mind, this paper introduces FedSCCS. Our framework clusters clients hierarchically based on their similarity and provides multi-model management for FL solutions instead of focusing on creating a single model for all parts. A comprehensive description of FedSCCS is presented in the next section.

III. Proposed Solution

This section introduces FedSCCS, a cluster-based framework for FL that offers multi-model management and supports different approaches among clusters, such as a specialized model creation for each group of clients. The main objective of FedSCCS is to improve the performance of FL in heterogeneous scenarios.

A. FedSCCS Overview

FedSCCS uses the Centered Kernel Alignment (CKA) method [17] to achieve better results in a heterogeneous context, which aims to identify the similarity between clients only looking at the models without access to private data. Then, a hierarchical clustering algorithm splits clients into

clusters, allowing FedSCCS to manage each cluster independently, i.e., allowing different treatments for each group depending on their needs. This provides different levels of customization, which can be chosen based on client data distributions, ranging from a generic management approach, joining clients with various distributions and maintaining some heterogeneity within clusters, to a highly specialized solution, making clusters homogeneous.

Additionally, FedSCCS can use various strategies to select clients to improve the model's overall performance. These techniques keep cluster models learning while training with fewer clients, reducing communication overhead and improving the method's efficiency.

B. FedSCCS Workflow

Federated Averaging (FedAvg) [16] is a standard FL method that aggregates client models by performing a weighted average of all model weights. Let $|C|$ be the number of clients in a scenario where C represents the set of clients, and T is the number of communication rounds. In each round $t \in 1, 2, 3, \ldots, T$, a randomly selected subset $\mathcal{S} \subseteq C$ of clients takes part in training the model. Each client $i \in \mathcal{S}$ has its local dataset D_i, $n_i = |D_i|$, and $N = |\bigcup_{i \in C} D_i|$. Thus, each client receives the current model parameters (i.e., weights) $w(t)$ from the server and updates its local model to train on its data. Subsequently, each client i sends the newly trained model parameters back to the server after training $w^i(t+1)$. The server then aggregates the received models by performing a weighted average, which produces the updated model parameters for the next communication round. At the end of round t, the updated global $w(t+1)$ is computed by averaging the trained model $w^i(t+1)$ of each client i, according to Equation 1.

$$w(t+1) = \sum_{i \in C} \frac{n_i}{N} w^i(t+1). \tag{1}$$

Hence, FedAvg's objective is shown in Equation 2, where w is the global model, $\mathcal{L}$ represents the model's loss function, and x_i and y_i are the input and label datasets, respectively.

$$\min f(w) \text{ where } f(w) = \sum_{i=1}^{|C|} \frac{n_i}{N} \mathcal{L}(x_i, y_i; w). \tag{2}$$

This generic approach of FedAvg (i.e., one global model) allows us to learn from all clients maintaining privacy, but it presents difficulties in heterogeneous scenarios. This occurs because when clients have a very different data distribution, the generic model may not converge or even deteriorate with the various models it receives. To overcome this limitation, FedSCCS proposes a hierarchical cluster-based management that handles each cluster independently, thus allowing not only generic models like FedAvg (i.e., one global model for all clients) but also highly specialized ones (i.e., multiple global models according to each cluster set up). Therefore, FedSCCS improves the overall performance of FL solutions.

FedSCCS organizes the clusters according to the model similarity. It employs a clustering function $\mathcal{K}$ that receives the set of models $\mathcal{W} = \{w^1, w^2, w^3, \ldots, w^{|C|}\}$ of the clients and returns a set of clusters $\Phi = \{C_1, C_2, \ldots C_K\}$, such that $\Phi = \mathcal{K}(\mathcal{W})$ and $\Phi \subset C$, where K is the number of clusters. Thus, each cluster can be managed independently, i.e., the aggregation of the models is specific and specialized to each cluster, we name these models as "specialized models". The weights of the model of cluster k, represented as w_k, are computed as shown in Equation 3.

$$w_k(t+1) = \sum_{i \in C_k} \frac{n_i}{N_k} w^i(t), \tag{3}$$

where C_k represents the set of clients addressed to cluster k (a client only can be in one cluster), and N_k is the total number of samples in k. The same occurs in the loss function, which is specific to each cluster.

Algorithm 1 Algorithm of the solution

Central Server do:

1: $w(0) \leftarrow$ *InitializeWeights*() ▷ *initialize the model*
2: **for** Each communication round $1, 2, \ldots, p-1$ **do**
3: **for** client $i \in \mathcal{C}$ **do** ▷ *in parallel*
4: $\mathcal{W} \leftarrow$ *clientUpdate*(w)
5: **end for**
6: **for** $i \in S$ **do**
7: $w(t+1) \leftarrow \sum \frac{n_i}{N} w^i(t)$
8: **end for**
9: **end for**
10: $\Phi \leftarrow \mathcal{K}(\mathcal{W})$ ▷ *clustering clients*
11: **for** Each communication round $p, p+1, \ldots, T$ **do**
12: **for** client $i \in \mathcal{C}$ **do** ▷ *in parallel*
13: $w_k \leftarrow$ *getModelbyCluster*(i)
14: $\mathcal{W} \leftarrow$ *clientUpdate*(w_k)
15: **end for**
16: **for** $C_k \in \Phi \mid k = \{1, 2, \ldots, K\}$ **do**
17: $c_k \leftarrow$ *clientSelection*(C_k)
18: $w_k(t+1) \leftarrow \sum_{i \in c_k} \frac{n_i}{N_k} w^i(t)$
19: **end for**
20: **end for**

21: **clientUpdate(w):** ▷ *Upon receiving a model update*
22: $\mathcal{B} \leftarrow$ *trainLoader*(D_i) ▷ *split local dataset in batches*
23: **for** each epoch $\in 1, 2, \ldots$ **do**
24: **for** batch $(x_i, y_i) \in \mathcal{B}$ **do**
25: $w^i \leftarrow \mathcal{L}(x_i, y_i, w)$ ▷ *fit the model with local data*
26: **end for**
27: **end for**
28: **Return** w^i to the server

FedSCCS is divided into rounds of traditional FedAvg and other specialized clusters. The full description of FedSCCS is described in Algorithm 1. Firstly, FedSCCS initializes the global model and sends it to the clients, each of which perform one local epoch in their private dataset (Line 1 and Lines 21-28). The second step is to run the FedAvg standard approach for $p-1$ rounds (Lines 2-9), i.e., all clients are aggregated at the beginning. The goal is to learn the clients' models for some epochs and then cluster them. In round p, the clustering algorithm creates the set clusters (Line 10), and the server sends specialized models for clients. In round p until the end, the server selects a c_k subset of clients from the C_k cluster for each cluster k (Line 17) and initiates the aggregation for each cluster (Line 18). So, for all other rounds, FedSCCS repeats Lines 11-20 until the convergence of the clusters' models,

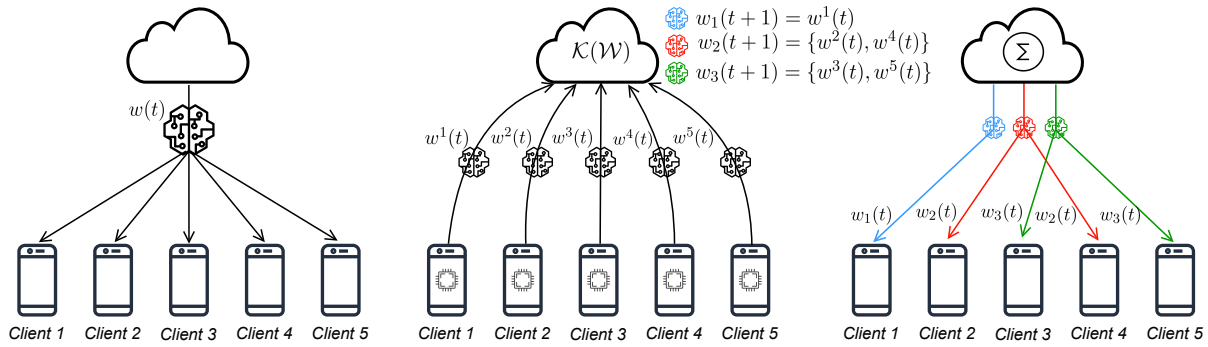


Figure 1. Illustration of how the global model is transformed into cluster models. At round p, the server sends the global aggregated model to clients (left), so they return the updated model, in which the server applies the $\mathcal{K}(\mathcal{W})$ clustering function, creating the clusters (middle). Finally, the server re-sends the specialized models to clients (right).

with occurs when all models don't have any more improves in accuracies for several rounds.

These last steps are depicted in Figure 1, illustrating the clustering creation processes. Firstly, FedSCCS sends a global model to all clients; next, each client performs local training and reports the new model to the server. Hence, the server employs the clustering function $\mathcal{K}$ to the set of clusters and organizes them into three different groups, described as w_1, w_2, and w_3 in this example. Eventually, each specialized model is sent back to each client according to the clustering function, illustrated as the last step in Figure 1. It is important to notice that this approach has no additional computation on the client side compared with FedAvg [16]. The models they receive in each round after clustering are specialized in their data distribution and trained with similar ones, improving the performance.

C. Model's Similarity

FedSCCS computes the similarity between models via Centered Kernel Alignment (CKA) [17] to create the clusters, which uses kernel transformations in matrices to measure a number between [0,1], representing the correlation of these matrices. In this way, FedSCCS calculates a_l^i, the activation of layer l of client i, given by:

$$a_l^i = \sigma_l^i(w_l^i a_{l-1}^i + b_l^i),$$

where σ_l^i, w_l^i, b_l^i are the activation function, the weights, and the bias of the l-th layer for client i, respectively. Thus, for the first layer:

$$a_1^i = \sigma_1^i(w_1^i D + b_1^i),$$

where D is a data sample in the server. So, a_l^i has two dimensions, one with the length of the server dataset and the other with the length of layer l.

Therefore, the kernel matrices K_1 and K_2 are calculated passing a_l^i and a_l^j to the kernel functions $K_1 = k_1(a_l^i, a_l^i)$ and $K_2 = k_2(a_l^j, a_l^j)$, which can be any operation in matrices. In our case, k_1 and k_2 were used as linear kernels, so $K_1 = k_1(\mathbf{x}, \mathbf{x}) = \mathbf{x}^T\mathbf{x}$ and $K_2 = k_2(\mathbf{y}, \mathbf{y}) = \mathbf{y}^T\mathbf{y}$. The CKA is calculated as:

$$\mathrm{CKA}(K_1, K_2) = \frac{\mathrm{HSIC}(K_1, K_2)}{\sqrt{\mathrm{HSIC}(K_1, K_1)\,\mathrm{HSIC}(K_2, K_2)}}, \quad (4)$$

$$\mathrm{HSIC}(K_1, K_2) = \frac{1}{(n-1)^2}\,\mathrm{tr}(K_1 H K_2 H), \quad (5)$$

where HSIC is the Hilbert-Schmidt Independence Criterion, "tr" is the trace of the matrix and H is a centralized matrix $H_n = I_n - \frac{1}{n}\underset{\sim}{\mathbf{1}} \times \underset{\sim}{\mathbf{1}}^{\mathrm{T}}$, with $\underset{\sim}{\mathbf{1}}^{\mathrm{T}} = \begin{pmatrix}1 & 1 & \dots & 1\end{pmatrix}$, a vector of ones, with length n and I_n the identity with order n where $n = |D|$, the size of server sample of data.

Finally, to use this similarity in the following steps, we will note the similarity of clients i and j as $\mathrm{CKA}_{j,i}^l(K_1, K_2) = \mathrm{CKA}_{i,j}^l(K_1, K_2)$ where $K_1 = k_1(a_l^i, a_l^i)$ and $K_2 = k_2(a_l^j, a_l^j)$ with respect to the layer l. With this metric, we can construct the matrix S, named similarity matrix, with dimensions $|C| \times |C|$, inputting in each cell $s_{i,j}$ the $\mathrm{CKA}_{i,j}^l$.

D. CKA-based Hierarchical Clustering

FedSCCS uses Ward's method [18], a hierarchical clustering based on the distance between the data points. In FedSCCS, the distances are calculated with the similarity matrix S generated by the CKA method and use the Euclidean pairwise distance in the columns $s_{.,i}$ and $s_{.,j}$ of the S matrix (Equation 6). As a result, the hierarchical clustering method is applied to the distance matrix D composed of elements $d_{i,j}$.

$$d_{i,j} = \sqrt{\sum_{q=1}^{|C|}(s_{q,i} - s_{q,j})^2}. \quad (6)$$

It is important to note that the hierarchical approach balances generalization and specialization, as depicted in Figure 2. The best number of clusters depends on the problem at hand and the number of similar data distributions of the clients. The diagram shows the structure established by this method, allowing us to select the cutting height, which determines the number of clusters. In this example, the first separation creates a cluster with the model w^1, while the remaining models form another cluster. Further partitions divide the largest cluster into

two smaller clusters, with these steps repeating as necessary. This approach provides flexibility, with higher cutting heights resulting in more general solutions like FedAvg, while lower heights lead to more specialized models.

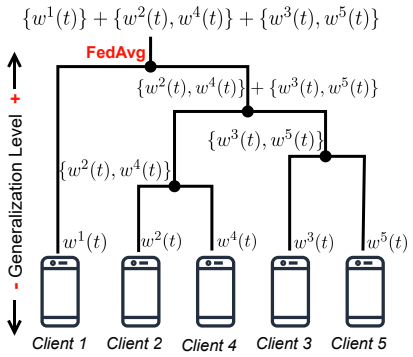


Figure 2. Illustration of the generalization capability of Ward's clustering method. Lower heights lead to more specialized models.

E. Client Selection Strategies

FedSCCS can use various client selection strategies within each cluster to enhance its performance. This selection process is crucial for improving the model convergence and optimizing downlink and uplink communications with the server [14].

We considered three methods to show the flexibility of FedSCCS in employing different client selection strategies based on three approaches: (*i*) selection of all clients, (*ii*) random selection, and (*iii*) biased selection. In the first strategy, every client of each cluster will be selected to train the model in that round. However, this approach has a downside as communication costs increase since all clients will send their local updates to the server.

The second strategy randomly selects clients from each cluster to train the model. This ensures that only one client is selected from each cluster, maintaining the specialization of each cluster while decreasing communication costs by reducing the number of selected clients (i.e., the number of chosen clients is equal to the number of clusters).

The third strategy selects clients based on their previous worst performance. The idea is to improve performance by boosting accuracy and reducing the loss of clients that have previously performed poorly [14]. This approach also reduces communication costs by selecting fewer clients from each cluster.

It is important to highlight that other methods could also be implemented in FedSCCS. For instance, FL approaches that that improve fairness and different context-aware strategies to optimize resource utilization (e.g., energy consumption and communication efficiency).

IV. Performance Evaluation

This section describes the performance evaluation of the FedSCCS in contrast to the literature solutions that uses other methods to clustering regarding the model's performance in heterogeneous scenarios.

A. Experiments Settings

FedSCCS is implemented by using both Flower [19] and TensorFlow. With Flower, each client is responsible for training the model with its local data and communicating with the central server using gRPC communication. On the other hand, the server is responsible for managing the clustering approach and selecting clients according to the strategies. The source code is available on the GitHub[1].

An essential use of FL is related to Human Activity Recognition (HAR) [20], especially in mobile device applications and with privacy concerns to this data. In this specific challenge, the dataset used to evaluate FedSCCS was MotionSense [21], which contains HAR data collected by a series of controlled experiments with 24 participants that vary in gender, age, height, and weight. This dataset was collected via an accelerometer and gyroscope (i.e., altitude, gravity, acceleration, and rotation rate) of an iPhone 6s. All participants did the same six activities: going downstairs and upstairs, walking, running, sitting, and standing, so this scenario is a Multiclass Classification problem with a mixture of distributions in the features since each participant has their own behavior, characterizing a heterogeneous scenario.

The model used in this evaluation is a Multi-Layer Perceptron (MLP) with three layers and 128, 64, and 32 neurons, respectively. The model has a ReLU activation function in each layer and a softmax in the end to calculate the probabilities of the classes. The network uses a Sparse Categorical Crossentropy as a loss function and an SGD optimizer to train.

The data sample in the server was made by collecting 100 random rows of the local dataset of each client. It is minimal compared to the size of the whole dataset, which has approximately 40000 samples each.

The similarity is calculated with the network's last layer because it was the most significant to differentiate the networks in our exploratory experimentation phase.

B. Clustering Analysis

Figure 3 presents the behavior of FedSCCS (left) varying the number of clusters, i.e. varying the generalization and specialization stated in Section III-D, and also of a baseline named Random Cluster (RC) solution (right), both creating the clusters in round 10. In this case, FedAvg represents that there is only one cluster created. The RC solution assigns each client to a random cluster without consideration of any metric, using a discrete uniform distribution. This comparison illustrates whether the clustering approach utilized by FedSCCS leads to an enhancement in overall performance.

This analysis demonstrates that FedSCCS provides better results than RC in all evaluations, which means that the clusters are meaningful and help the model's performance. The high accuracy reached in RC with 20 clusters demonstrates that this number leads to overfitting of the models, which is a consequence of their excessive specialization. The most significant disparity between FedSCCS and RC becomes

[1] https://github.com/GabrielTalasso/FedSCCS

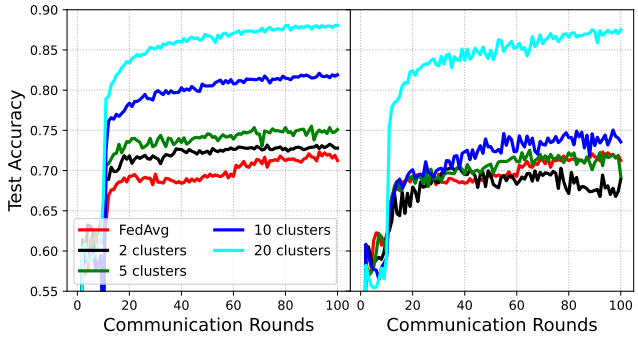
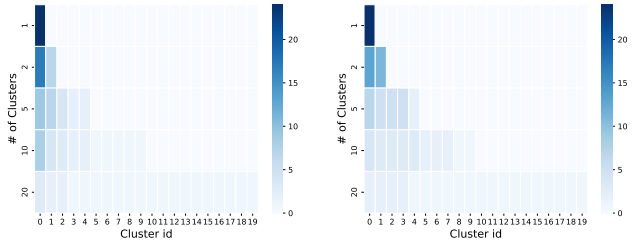


Figure 3. Test accuracy of FedSCCS (left) and RC (right) varying the number of clusters created. FedAvg represents that there is only one cluster.

evident with 10 clusters, exhibiting an improvement attributed to the clustering founded on similarity rather than solely on the models' heightened specialization. Another contributing factor is that the data from the 24 participants may exhibit nearly ten distinct data distributions or patterns, which accounts for this performance discrepancy. Subsequent experiments employ a cluster count of 10 due to these considerations. This is depicted in Figure 4, which displays the client distribution across clusters for both solutions. We can notice the greater concentration in our solution, meaning that similar clients with average behavior are clustered together.



(a) FedSCCS (b) Weights Clusters

Figure 4. Spread of clients in clusters for FedSCCS and RC.

C. Comparison with Literature Solutions

The following experiments compare the performance of FedSCCS with literature solutions. The clustering approach proposed in [15] uses the model weights and the KCenter clustering method. In our experiments, we named this solution as KCenter. The proposed clustering methods in [6, 10, 8] use the gradients of the models to compute the similarity, but as in [6], the gradients can be approximated with the models' weights. Wang et al. [15] demonstrated that, for models that train with the same initial model, the weights of updated models represent the discrepancy of data distributions. To describe these solutions, WC (Weights Clustering) uses the clustering with the cosine similarity of the weights as a similarity function and Ward's method as the clustering method, like in FedSCCS.

Figure 5 compares FedSCCS and these literature solutions. The better performance of FedSCCS in this experiment results from the clustering based on the similarity of the models that better represent and capture the clients' data distribution. All solutions of clusters reach higher performance than the classical FedAvg approach and the Random Clusters solution (RC), which means that they are valid methods in a heterogeneous scenario.

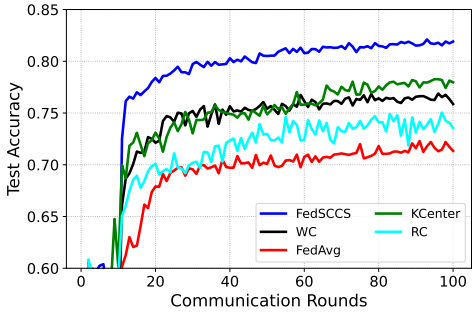


Figure 5. Test accuracy of different literature solutions.

As proposed in Section III-E, FedSCCS also selects clients in each round to reduce communication and improve the efficiency of FL processes. Figure 6 shows the result of choosing one random client of each cluster in each round compared to the FedAvg selecting all clients. The higher

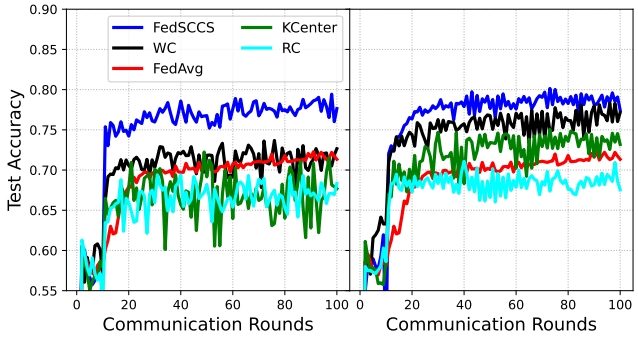


Figure 6. Left image shows the performance of Random Selection in different solutions. Right image shows the same for POC based selection.

accuracy of FedSCCS occurs because the clusters represent better similar clients, making the FL process learn with fewer clients. The same happens with the Worst Client Selection, described as the POC solution [14], in the case that 50% of the clients of each cluster are selected in each round. The random selection performers close to POC, even using one client per cluster and not half of the clients. This means that the FedSCCS clustering helps to select representative clients for their data distributions, improving communication efficiency even when selecting fewer clients. Figure 7 shows that, in Random Selection, most of the clients were selected fewer times than in POC-based selection and yet reached similar accuracy. Hence, the client resource consumption is lower, demonstrating another advantage of using FedSCCS clustering. Moreover, FedSCCS outperforms the FedAvg with all clients selected in both the client selection method and the

other literature methods. This indicates that even using fewer clients and reducing communication costs, FedSCCS performs better.

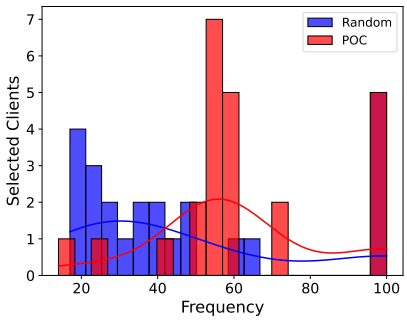


Figure 7. Frequency of different selected clients for Random and POC selection methods.

D. Communication Analysis

This analysis assesses the communication cost (uplink between the client and the server) and how we can improve it with client selection techniques.

Our experiments compared FedSCCS to FedAvg with all clients in every round. We found that for 10 clusters, the POC-based solution only consumed 62.5% of the communication cost of the complete FedAvg approach, while random client selection consumed only 47.5%. This efficiency gain is shown in Figure 8 for different clusters, displaying the communication cost of distinct selection methods in FedSCCS. The client selection techniques reduce the costs, especially in random selection, which chooses only one client per cluster and maintains accuracy and convergence.

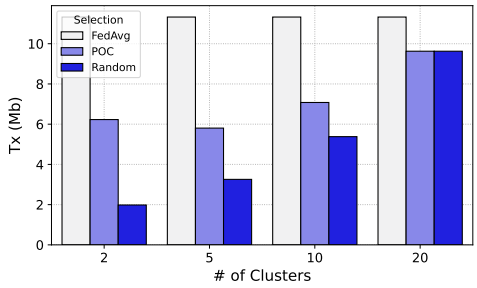


Figure 8. Communication costs with different number of clusters and different selection methods in FedSCCS.

Based on these analyses, we can draw the following conclusions: (*i*) FedSCCS produces significant clusters through the CKA similarity measure, which leads to better performance and improved management; (*ii*) the management of multiple hierarchical models independently for each cluster offers a wide range of customization options, from generic to specialized, making it adaptable for specific applications; and (*iii*) FedSCCS is well-suited for implementing alternative client selection strategies that help to reduce communication costs and optimize resource utilization while maintaining overall performance.

V. Conclusions

This paper introduced FedSCCS, a framework aimed at clustering clients based on their model similarity and selecting clients to enhance efficiency. FedSCCS employed CKA similarity and combined it with Ward's hierarchical clustering method, facilitating the creation of representative clusters and subsequent client selection. Experimental results show that FedSCCS achieves superior accuracy when it was compared to state-of-the-art approaches. Furthermore, by incorporating client selection techniques, FedSCCS effectively reduces communication costs and prevents client overload during training.

A significant limitation of FedSCCS lies in determining the optimal number of clusters in each case, striking a balance between avoiding overly specialized models and excessive generalization. Therefore, as part of our future work, we will address this limitation and investigate how FedSCCS performs when incorporating various algorithms for aggregating, configuring, and selecting clients within each cluster. The ultimate objective is to develop a dynamic approach that comprehends and defines which algorithm should be employed in each cluster, ultimately enhancing its effectiveness.

References

[1] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.

[2] S. Abdulrahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, and M. Guizani, "A survey on federated learning: The journey from centralized to distributed on-site learning and beyond," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5476–5497, 2021.

[3] B. Liu, M. Ding, S. Shaham, W. Rahayu, F. Farokhi, and Z. Lin, "When machine learning meets privacy: A survey and outlook," *ACM Computing Surveys (CSUR)*, vol. 54, no. 2, pp. 1–36, 2021.

[4] A. Imteaj, U. Thakker, S. Wang, J. Li, and M. H. Amini, "A survey on federated learning for resource-constrained iot devices," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 1–24, 2022.

[5] D. K. Dennis, T. Li, and V. Smith, "Heterogeneity for the win: One-shot federated clustering," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 2611–2620.

[6] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 8, pp. 3710–3722, 2021.

[7] O. Sener and S. Savarese, "Active learning for convolutional neural networks: A core-set approach," *arXiv preprint arXiv:1708.00489*, 2017.

[8] 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.

[9] M. Morafah, S. Vahidian, W. Wang, and B. Lin, "Flis: Clustered federated learning via inference similarity for non-iid data distribution," *IEEE Open Journal of the Computer Society*, vol. 4, pp. 109–120, 2023.

[10] C. Palihawadana, N. Wiratunga, A. Wijekoon, and H. Kalutarage, "Fedsim: Similarity guided model aggregation for federated learning," *Neurocomputing*, vol. 483, pp. 432–445, 2022.

[11] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–7.

[12] Y. Deng, F. Lyu, J. Ren, H. Wu, Y. Zhou, Y. Zhang, and X. Shen, "Auction: Automated and quality-aware client selection framework for efficient federated learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 8, pp. 1996–2009, 2022.

[13] M. Ribero, H. Vikalo, and G. de Veciana, "Federated learning under intermittent client availability and time-varying communication constraints," *IEEE Journal of Selected Topics in Signal Processing*, vol. 17, no. 1, pp. 98–111, 2023.

[14] Y. J. Cho, J. Wang, and G. Joshi, "Towards understanding biased client selection in federated learning," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2022, pp. 10 351–10 375.

[15] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-iid data with reinforcement learning," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, 2020, pp. 1698–1707.

[16] B. McMahan, E. Moore, D. Ramage, S. Hampson, , and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," 2017.

[17] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton, "Similarity of neural network representations revisited," in *International conference on machine learning*. PMLR, 2019, pp. 3519–3529.

[18] J. H. Ward Jr, "Hierarchical grouping to optimize an objective function," *Journal of the American statistical association*, vol. 58, no. 301, pp. 236–244, 1963.

[19] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K. H. Li, T. Parcollet, P. P. B. de Gusmão *et al.*, "Flower: A friendly federated learning research framework," *arXiv preprint arXiv:2007.14390*, 2020.

[20] K. Sozinov, V. Vlassov, and S. Girdzijauskas, "Human activity recognition using federated learning," in *2018 IEEE Intl Conf on Parallel Distributed Processing with Applications, Ubiquitous Computing Communications, Big Data Cloud Computing, Social Computing Networking, Sustainable Computing Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, 2018, pp. 1103–1111.

[21] M. Malekzadeh, R. G. Clegg, A. Cavallaro, and H. Haddadi, "Mobile sensor data anonymization," in *Proceedings of the international conference on internet of things design and implementation*, 2019, pp. 49–58.