



<http://www.diva-portal.org>

Postprint

This is the accepted version of a paper presented at *26th IEEE Symposium on Computers and Communications, ISCC 2021, Athens, 5 September 2021 through 8 September 2021*.

Citation for the original published paper:

Al-Saedi, A A., Boeva, V., Casalicchio, E. (2021)

Reducing Communication Overhead of Federated Learning through Clustering Analysis

In: *26th IEEE Symposium on Computers and Communications (ISCC 2021)* Institute of Electrical and Electronics Engineers (IEEE)

<https://doi.org/10.1109/ISCC53001.2021.9631391>

N.B. When citing this work, cite the original published paper.

©2021 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Permanent link to this version:

<http://urn.kb.se/resolve?urn=urn:nbn:se:bth-22237>

# Reducing Communication Overhead of Federated Learning through Clustering Analysis

Ahmed A. Al-Saedi

Department of Computer Science  
Blekinge Institute of Technology  
Karlskrona, Sweden  
Email: ahmed.a.al-saedi@bth.se

Veselka Boeva

Department of Computer Science  
Blekinge Institute of Technology  
Karlskrona, Sweden  
Email: veselka.boeva@bth.se

Emiliano Casalicchio

Departement of Computer Science  
Sapienza University of Rome, Italy &  
Blekinge Institute of Technology, Sweden  
Email: emiliano.casalicchio@uniroma1.it

**Abstract**—Training of machine learning models in a Datacenter, with data originated from edge nodes, incurs high communication overheads and violates a user’s privacy. These challenges may be tackled by employing Federated Learning (FL) machine learning technique to train a model across multiple decentralized edge devices (workers) using local data. In this paper, we explore an approach that identifies the most representative updates made by workers and those are only uploaded to the central server for reducing network communication costs. Based on this idea, we propose a FL model that can mitigate communication overheads via clustering analysis of the worker local updates. The Cluster Analysis-based Federated Learning (CA-FL) model is studied and evaluated in human activity recognition (HAR) datasets. Our evaluation results show the robustness of CA-FL in comparison with traditional FL in terms of accuracy and communication costs on both IID and non-IID cases.

**Index Terms**—Federated Learning, Communication Costs, Clustering Analysis, Human Activity Recognition

## I. INTRODUCTION

Today, low-end edge devices, such as smart-phones and Internet of Things (IoT) devices, are equipped with powerful and energy-efficient processing units. This unique landscape enables the execution of complex Machine Learning (ML) tasks at the edge of the Internet. Federated Learning (FL) [1] has been proposed by Google in 2017 to enable learning collaboratively over a large number of edge devices (workers hereafter) without the need to centrally collect training data and to train centralized models.

In FL, a global model is initialized in a central location (e.g., a cloud datacenter) and is shared with all participating workers for a set of iterative training round. At each training round, the workers train the model locally (using their local data), and then each worker sends its model parameter update back to the central node. The global model is then updated averaging the local model parameters received by all the workers and shared again with them [1]. These operations are repeated at each iteration round.

The iterative nature of FL, however, does not eliminate the network congestion problem completely due to data offloading from the edge to the cloud. Indeed, for complex models, large scale applications (with hundred thousand or millions of workers), and high frequency update, the communication overhead at each iteration is not negligible, and becomes a

challenge to be addressed [2]–[4]. Several research studies have proposed decreasing the overall number of bits transferred for each worker update which is mostly used by means of data compression. However, none of those works considers the impact of some loss of data compression results that could bring harm to the learning accuracy, resulting in no convergence guarantees [4]. The other direction is to reduce the total number of worker updates transferred during the model training [5].

In this paper, we propose a new federated learning algorithm, entitled Cluster Analysis-based Federated Learning (CA-FL), that is reducing the communication overhead without losing accuracy rate by minimizing the number of worker updates transferred during each FL iteration phase. CA-FL consists of an initialization phase and a sequence of iteration steps (an iterative training phase). At the initialization phase the local updates of the available workers are analysed and partitioned into a number of clusters. Then at each training round (iteration phase), representatives (one or more) are selected for each cluster based on some task specific evaluation criteria. Only the data supplied by the representative workers are used for training during the current round in order to build the global model. As a final operation, at each round the worker partitioning is adapted based on the analysis of the newly arrived information (the local updates of the selected workers). In that way new representatives will be selected at the next training round using the adapted partitioning of the workers. The training process is iterative and continues until a predefined stopping criteria is met. The rationale behind the above idea is that it is not necessary to calculate and upload the local updates that are similar. Instead, we can use representatives for each group of similar updates (model local parameters) and reflect the importance of each group into the aggregation of the global model by weighting its cardinality.

The proposed CA-FL algorithm can be considered as a communication-efficient version of Federated Averaging (FedAvg) algorithm introduced in [2]. Hence in this study we compare CA-FL to FedAvg on two publicly available physical activity monitoring datasets in terms of model accuracy and communication costs. These datasets have been selected since human activity recognition is one of smart monitoring applications that will benefit of federated learning models that

reduce communication costs.

The rest of this paper is organized as follows. Section II contains a brief description of the related works. This is followed by Section III which introduces required background and methods. The proposed approach is explained in Section IV. Details about the data, experimental setting and discussion of the results obtained from initial evaluation of the proposed approach are presented in Section V. Section VI is devoted to conclusions and future work.

## II. RELATED WORK

Asad et al. [6] have evaluated the communication efficiency of FL approaches and additionally provided a detailed review of the state-of-the-art FL frameworks. Solutions that address the problem of reducing network overhead in FL can be classified in two broad categories: works that reduce the communication traffic by means of data compression [3], [7]–[11] and studies that aim at reducing the number of local updates and syncs with the central model [5], [12].

Notice that our proposed FL model falls into the second category. For instance, Gaia approach, introduced in [12], proposes a communication architecture for minimizing the communication cost for geo-distributed ML systems. The Gaia measures the importance of a local update with a predefined threshold. Thus, the insignificant local update will be excluded from uploading to the server. A framework called Communication-Mitigated Federated Learning (CMFL) has been proposed in [5] to specify the relevance of a worker update by calculating the proportion of parameters having different signs of the local update and global update then excluding the irrelevant updates. Chen et al. have proposed a communication-dynamic method for FL where a layer-wise asynchronous update method is used [13]. Jeong et al. have introduced a FL scheme to reduce communication overhead with few losses on the accuracy with a feature fusion technique for aggregating the features [14]. Wu et al. have introduced in [15] a framework (FedMed) for the adaptive aggregation, mediation incentive scheme, and topK strategy to address the model aggregation and communication overhead. Elbir [16] proposes a hybrid FL and centralized learning (HFCL) mechanism that performs gradient computation on only active devices that have enough computational power during model training. Caldas et al. have presented strategies to alleviate communication costs by sending a lossy compressed model to the client and permitting users to train small subsets of the global model (Federated Dropout) [17]. In [18] a practical Robust, and Communication-efficient Semi-supervised FL (RC-SSFL) system that allows the clients to jointly learn a high-quality model is used.

We have been inspired by the above discussed studies and explored an approach that applies clustering analysis to identify the most representative updates made by workers and those are only uploaded to the central server for reducing network communication costs.

## III. BACKGROUND AND METHODS

In this section, we introduce the concept of federated learning upon which our solution is based [4]. In addition, we discuss techniques used for conducting clustering analysis on the workers' local updates.

### A. Federated Learning

In Federated Learning [4], the learning task is conducted by having a coordinator (also referred as server and usually running in a cloud Datacenter) that, throughout a set of iterations, collaborates with all the participating workers (also referred as clients) to learn and build a shared, privacy preserving ML model. Note that private data is kept by the workers and not shared with the Server.

In detail, a model  $M$  is learned iteratively by using a randomly selected subset, denoted by  $W_s$ , of all available workers. The workers in  $W_s$  participate at each round and compute the gradient of the loss over all the data held by them. Each worker  $w \in W_s$  at round  $t$  has its own row of data  $D_w^t$  and a local model  $m_w^t$ . At each round  $t$ , each worker trains its local model by iterating the local update multiple times of Stochastic Gradient Descent (SGD) before sending the next local model  $m_w^{t+1}$  to the server which holds the global model. The server, after collecting all the local models computed at round  $t$ , performs a synchronous update of the global model  $M_{t+1}$ . The global model update can be computed using different criteria. In this paper, we assume it is evaluated by means of federated averaging, that is the local model  $m_w^t$ ,  $w \in W_s$  and global model  $M_t$  are updated by the following equations: [19]:

$$m_w^{t+1} = m_w^t - \eta g_w^t; \quad (1)$$

$$M_{t+1} = \sum_{w \in W_s} \frac{n_w}{n} m_w^{t+1}, \quad (2)$$

where  $m_w^{t+1}$  is the local update,  $g_w^t$  are the updated weights on its local data at the current model  $m_w^t$ ,  $M_{t+1}$  is the next global model,  $\eta$  is a learning rate computed by each worker,  $W_s$  is the set of workers that participate in the training,  $n$  is the sum (total number) of all data points and  $n_w$  is the number of local data points. The server then distributes the global model  $M_{t+1}$  to the workers that can perform another iteration of local training and model update. We refer to this baseline algorithm as FedAvg (FL-baseline hereafter).

### B. Partitioning Algorithms

Three partitioning algorithms are well known and generally used for data analysis to partition the data points into  $k$  disjoint clusters [20]:  $k$ -means,  $k$ -medians, and  $k$ -medoids clustering. All these methods start by initializing a set of  $k$  cluster centers, where  $k$  is preliminarily determined. Then, each data point is assigned to the cluster whose center is the nearest, and the cluster centers are recomputed. This process is repeated until the points inside every cluster become as close to the center as possible and no further item reassignments take place. The three partitioning techniques vary in how the cluster center is located.

In our CA-FL algorithm we use  $k$ -medoids for partitioning the available workers into groups of similar workers w.r.t. their local updates. The medoid is the most centrally located point in a given cluster. Due to this in the CA-FL, the medoid of each cluster can be used as the representative of the cluster in a case the evaluation of the workers' updates is missing.

1) *Silhouette Index*: *Silhouette Index* (SI) is a well-known and widely used internal cluster validation technique [21]. The cluster validation techniques are designed to be used for evaluating the quality of data partitions. Internal cluster validation techniques base their evaluation on the same information used to derive the clusters themselves and can be split with respect to the specific clustering property they assess. A detailed and comparative overview of different types of validation measures can be found in [22].

Suppose  $a_i$  represents the average distance of object  $i$  from all the other objects in the cluster to which the object  $i$  is assigned, and  $b_i$  represents the minimum of the average distances of object  $i$  from objects of the other clusters. Then the *Silhouette Index*  $s(i)$  of object  $i$  can be calculated as

$$s(i) = (b_i - a_i) / \max\{a_i, b_i\}. \quad (3)$$

$s(i)$  measures how well object  $i$  matches the clustering at hand.  $s(i) \in [-1, 1]$  and higher value points out a better quality of the clustering results. For example, when  $s(i)$  is close to 1 this means that object  $i$  is assigned to a very appropriate cluster. A situation is different when  $s(i)$  is about zero. Namely, object  $i$  lies between two clusters. The worst case is when  $s(i)$  is close to  $-1$ . Evidently, this object has been misclassified.

The proposed CA-FL algorithm applies the Silhouette Index at each iteration round for assessing whether the worker representatives are still well tied to their current clusters (see Section IV).

2) *Estimation of the number of clusters*: The partitioning algorithms, such as  $k$ -medoids used in our study, require  $k$ , the number of clusters to be known in advance. However, determining the optimal number of clusters  $k$  may be challenging in solving problems involving real-world data sets. One solution that may apply in such cases is to build a clustering model with a range of values for  $k$  and then evaluate the quality of the generated clustering partitions. For example, different internal cluster validation indices (such as SI) can be applied to recognize the optimal clustering solution.

In this study, we take the advantage of Silhouette Index method for identifying the optimal  $k$ . Namely,  $k$ -medoids is conducted for each value of  $k$  in a given interval. The corresponding scores of SI generated by the different  $k$  are depicted as a function of  $k$ . The optimal  $k$  is the value at which a significant local change in the value of this method observed.

#### IV. PROPOSED CA-FL ALGORITHM

The proposed CA-FL algorithm foresee two distinctive phases, *initialization* and *iteration*, described in what follows.

Let us define  $W = \{w_1, w_2, \dots, w_n\}$  the set of all available workers and  $W^t$  is a subset of  $W$  that contains workers

selected at round  $t$ . The workers in  $W^t$  can be the clusters' representatives or a set of randomly selected workers and  $|W^t| < n$ .

##### Initialization Phase:

- 1) The Server initializes model  $M_t$ ,  $t = 0$ .
- 2) Server sends the initial global model  $M_t$  to a set of workers  $W^t$  ( $W^t \subset W$ ). These are selected to be used for a initial training of FL at round  $t$ .
- 3) Each worker  $w \in W^t$  receives the global model  $M_t$  and optimizes its parameters locally, i.e.  $m_w^t$  initial update is produced and sent back to the server (eq. 1).
- 4) Server aggregates the parameters  $\{m_w^t \mid w \in W^t\}$  uploaded by the selected workers  $W^t$  to update the global model  $M_t$  through the FedAVG algorithm (eq. 2).
- 5) The local updates  $\{m_w^t \mid w \in W^t\}$  of the workers in  $W^t$  are analyzed by using  $k$ -medoids algorithm and the workers are partitioned into  $k$  clusters of similar updates, i.e. an initial clustering  $C^t = \{C_1^t, C_2^t, \dots, C_k^t\}$  of the workers in  $W^t$  is produced.

##### Iteration Phase:

- 1) At each iteration round  $t$  ( $t \geq 0$ ) server evaluates each local update  $m_w^t$ ,  $w \in W^t$  by using an evaluation measure suitable for the task under consideration.
- 2) Server ranks the workers in each cluster  $C_i^t$ ,  $i = 1, 2, \dots, k$  and selects the top-ranked worker (the representative) or the first  $p$  top-ranked workers, where  $p$  is defined to be proportional to the cluster size. The selected representatives form a new set of workers  $W^{t+1} = \{w_1^{t+1}, w_2^{t+1}, \dots, w_k^{t+1}\}$ , where  $k \ll |W^0|$ . Each worker  $w \in W^{t+1}$  will check-in with the server.
- 3) Server sends the global model  $M_t$  to each representative  $w \in W^{t+1}$ .
- 4) Each representative  $w \in W^{t+1}$  receives the global model  $M_t$  and optimizes its parameters locally, i.e.  $m_w^{t+1}$  update is produced (eq. 1) and sent back to the server.
- 5) Server aggregates the received parameters  $\{m_w^{t+1} \mid w \in W^{t+1}\}$  uploaded by the representatives to update the global model through the FedAVG algorithm, i.e. an updated global model  $M_{t+1}$  is produced.
- 6) Server adapts  $C^t$  to the newly arrived local updates, i.e. SI is applied for assessing whether each representative  $w \in W^{t+1}$  is still well tied to its current cluster (eq. 3). It may happen that some workers will change their clusters, i.e. the updated clustering  $C^{t+1}$  of the workers in  $W^0$  is produced.

Then the steps 1 – 6 of the *iteration* phase are repeated until the maximum number  $T$  of training rounds is executed. The CA-FL algorithm iteration phase is illustrated in Figure 1.

## V. EVALUATION

### A. Data and Experimental Setup

For initial evaluation of the proposed CA-FL algorithm, we have used publicly available real-world datasets [23], [24] from UCI Machine Learning repository. We use mHealth and Pampap2 physical activity monitoring datasets. Both datasets

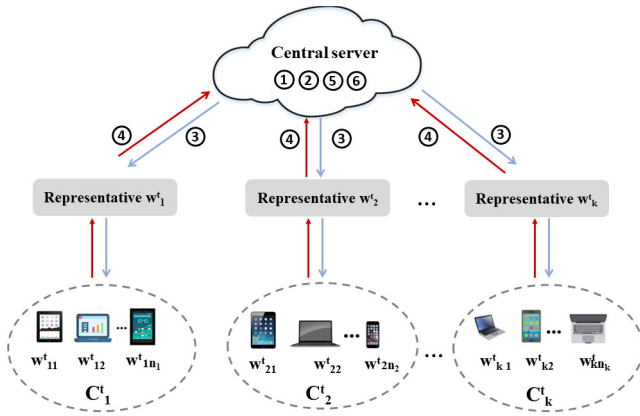


Fig. 1. A schematic illustration of the CA-FL algorithm iteration phase. Circled numbers correspond to the iteration phase.

involve motion sensor data of several physical activities. Each worker trains its local model for a number of epochs on the local dataset using sklearn as the ML library. The updated local model is sent back to the Server. When all workers performed  $e$  number of epochs, the Server updates the global model and sends it again to the workers. The process continues and conducts until max number of training rounds  $T$  is reached.

In order to simulate a distributed scenario each dataset is used to generate 10 experimental datasets by randomly separating the data points into a number of groups. Each group is supposed to represent data supplied by one worker. In addition, we have studied two different experimental data distribution scenarios: IID (Independent and Identically Distributed) and non-IID. In that way for each studied number of workers (20, 30, 40, 50 and 60) and each used dataset (mHealth and Pamap2) we have built 20 experimental datasets, i.e. 200 experimental datasets in total. We compare the CA-FL algorithm against the FL-baseline algorithm [2] on the built experimental datasets.

### B. Evaluation Metrics

The FL communication overhead metrics are defined by the average calculation performed by all workers during the model training. In the first FL round of communication, the server sends the initial global weights ( $N$  bytes) to set of selected workers  $W_s$ . At the end of the round, it downloads the updates of size  $N$  bytes from each participating worker and carries out the aggregation. This is iterated for  $T$  communication rounds. The overall communication cost of the server is given by:

$$(2 \times N \times |W_s|) \times T + N \times |W_s|,$$

where  $N$  is the size of the trained model in bytes,  $|W_s|$  is the number of selected workers and  $T$  is the total number of training rounds. We assume the size of the model updates and the number of training iterations to be fixed [25].

### C. Implementation and Availability

We have compared the proposed CA-FL algorithm against the FL-baseline, briefly explained in Section III-A, using Stochastic Gradient Descent (SGD) classifier as a training model in two data distribution scenarios, i.e. IID versus non-IID. The FL-baseline and CA-FL algorithms are implemented in Python using Scikit-learn library. The machine learning task is a multi-class classification problem of human activity recognition. Silhouette Index is used from the Python library Scikit-learn. The scikit-learn implementation of the F-measure (*micro-average  $F_1$* ) has been used to evaluate the accuracy of the two studied algorithms.

### D. Results and Discussion

We initially evaluate the communication overhead by applying FL-baseline and CA-FL algorithms on the experimental datasets of mHealth and Pamap2 datasets built for 20 numbers of workers under two data distribution scenarios: IID versus non-IID. The results obtained on mHealth dataset are reported in Figure 2. The results generated on the experimental datasets of Pamap2 are similar. We have also studied how the number of communication rounds affects the communication cost and the experiments have been conducted for different numbers of rounds (20, 40, 60, 80 and 100). We can observe that in both data distribution scenarios, CA-FL substantially reduces the communication costs. As it will be shown and discussed later in this section this is achieved without losing the learning correctness. In addition, it is interesting to notice that the communication costs accumulated by FL-baseline increase faster with the higher number of rounds in comparison to the ones generated by CA-FL.

The classification performance of the two compared algorithms has been evaluated by running 10-fold cross-validation on each experimental dataset of mHealth and Pamap2 datasets for 20 communication rounds and different number of workers. Figure 3 shows the average F1 scores produced by FL-baseline and CA-FL algorithms on Pamap2 experimental datasets. We can see that the lost in the learning accuracy of CA-FL algorithm in comparison with the FL-baseline algorithm is insignificant. The results generated on the experimental datasets of mHealth are similar.

We have additionally studied how the number of workers used to train the global model affect the communication cost and classification performance of FL-baseline and CA-FL. Figures 3 and 4 display F1 scores and the generated communication costs of FL-baseline and CA-FL algorithms on Pamap2 experimental datasets in IID and non-IID data scenarios, respectively. The obtained results confirm the discussed above observations. Namely, CA-FL reduces the communication costs significantly and these are not highly dependent on the number of workers in contradiction to the FL-baseline algorithm behaviour in the same context. The classification performance of both algorithms is not affected by the number of workers. Notice that the similar results under both evaluation criteria are generated on the experimental datasets of mHealth.

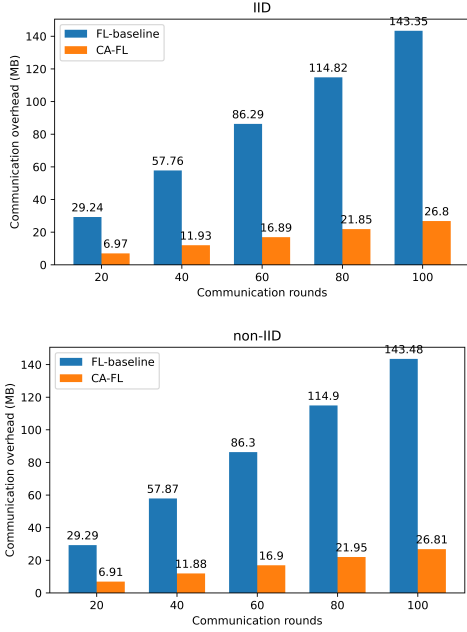


Fig. 2. Communication overhead accumulated by FL-baseline and CA-FL algorithms on mHealth datasets under IID (top) and non-IID (bottom) scenarios for different communication rounds.

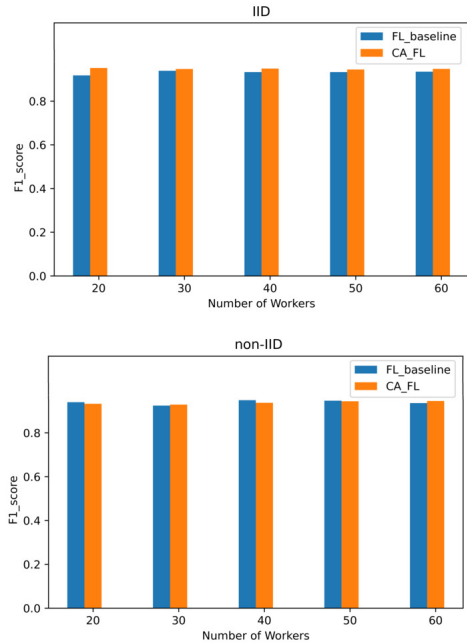


Fig. 3. F1 scores generated by FL-baseline and CA-FL algorithms on Pamap2 datasets under IID (top) and non-IID (bottom) scenarios for different number of workers.

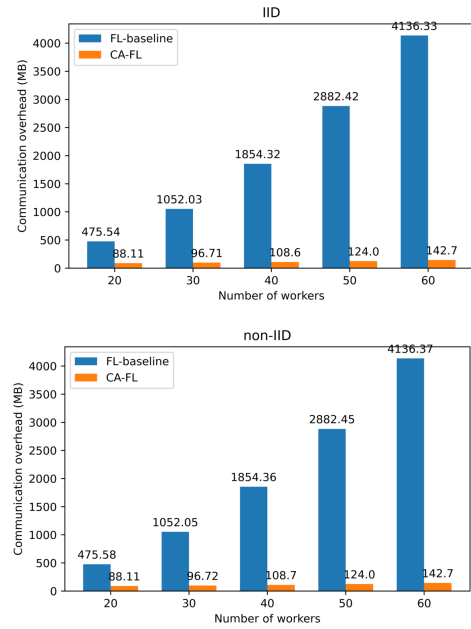


Fig. 4. Communication overhead accumulated by FL-baseline and CA-FL algorithms on Pamap2 datasets under IID (top) and non-IID (bottom) scenarios for different number of workers.

The propose CA-FL approach also supplies with an opportunity to conduct a post-analysis step by evaluating the quality of the workers' data. For example, the number of rounds at which a worker has been used as a representative can eventually be considered as a measure for its data quality.

Table I presents the workers' data quality scores (frequency of being representatives) obtained by averaging over the scores generated on the experimental datasets built on each of the two datasets (mHealth and Pamap2) in the two studied data distribution scenarios (IID versus non-IID) using 20 workers, i.e. four different scores are generated for each worker. It is interesting to notice that the workers' performance is influenced by the used dataset and the data distribution scenario. For example, worker 0 has produced the highest score on Pamap2 data set in IID scenario, but the lowest one in non-IID for the same dataset. In addition, one can see that worker 17 has a high score (4.2) for Pamap2 dataset under IID scenario, but a significantly lower one under the same scenario for mHealth dataset. However, five workers (13, 14, 16, 18 and 19) perform comparatively well under the four studied cases and can be considered more reliable then the others. This is supported by the last column presenting the average score over all four scores for each worker. Such an evaluation procedure may be applied as an initial step for the exploration of the workers in order to be able to select the top ranked (the most reliable) workers for the real training of the model. This is initially studied by the experiments discussed at the end of this section.

We have studied how the number of selected representatives varies with respect to the studied experimental scenarios. The identified optimal number of clusters (representatives) in different experimental scenarios of the two datasets are

TABLE I

EVALUATION OF THE WORKERS’ DATA QUALITY ON THE DATASETS OF mHEALTH AND PAMAP2 IN THE TWO DATA DISTRIBUTION SETTINGS (IID VS NON-IID). THE LAST COLUMN CONTAINS THE AVERAGE SCORES CALCULATED OVER THE STUDIED SCENARIOS.

worker	mHealth		Pamap2		Average Score
	IID	Non-IID	IID	Non-IID	
0	3.5	2.7	<b>4.9</b>	2	3.275
1	3.8	2.8	2.8	3.6	3.25
2	2.3	4	2.7	3.1	3.025
3	3.3	3.3	3.4	2.2	3.05
4	2.7	2.3	2.4	3.7	2.775
5	2.8	2.1	4.6	2.3	2.95
6	3.5	3.9	3.6	2.1	3.275
7	1.4	3.5	2.5	3.7	2.775
8	3	2.6	2.9	3.7	3.05
9	2.7	3.4	2.8	3.5	3.1
10	4.2	1.4	1.7	3.2	2.625
11	2.1	2.5	2.5	2.6	2.425
12	3.2	3.5	3.5	4.3	3.625
<b>13</b>	<b>4.5</b>	3.3	3.7	4	<b>3.875</b>
<b>14</b>	3.4	3.2	3.3	<b>5.1</b>	<b>3.75</b>
15	1.9	3.8	3.3	3.7	3.175
<b>16</b>	4.3	2.6	4.3	3.5	<b>3.675</b>
17	2.7	3.4	4.2	3.4	3.425
<b>18</b>	3.5	3.6	4.3	4.3	<b>3.925</b>
<b>19</b>	3.8	<b>4.1</b>	3.1	3.5	<b>3.625</b>

presented in Table II. Interestingly, the non-IID scenario of mHealth dataset uses less number of representatives under all studied numbers of workers while the rest three experimental scenarios demonstrate very similar patterns.

TABLE II

IDENTIFIED OPTIMAL NUMBER OF CLUSTERS (REPRESENTATIVES) IN DIFFERENT EXPERIMENTAL SCENARIOS (IID AND NON-IID)

No. workers	mHealth		Pamap2	
	IID	Non-IID	IID	Non-IID
20	10	8	10	8
30	16	10	17	16
40	24	17	23	25
50	27	20	28	27
60	38	23	34	36

In addition, we investigate the communication costs and F1-score values in a scenario when the number of workers selected by traditional federated learning is equal to the number of representatives ( $k$ ) used by the CA-FA algorithm. The results obtained on the experimental non-IID datasets of Pamap2 data are given in Table III. It can be observed that under different number of rounds, CA-FL outperforms FL-baseline in term of accuracy. However, since CA-FL communicates with all participating workers in the initialization phase rather than only  $k$  number of workers this costs 7.0974 MB for one round of training when 20 workers involved in federated learning scenario. The latter, as can be seen in Table III, affects the overall communication cost of CA-FL. We can also notice that the server overhead increases linearly with the number of rounds. Furthermore, the results of the FL-baseline and the ones accumulated during the iteration phase

of CA-FL are comparable since both use  $k$  number of workers. Note that the overall communication costs of the CA-FL are calculated by aggregating the communication costs of the initialization and iteration phases. The results generated on Pamap2 experimental IID dataset are similar. Note that the above discussed scenario is artificial, since in our definition of CA-FL algorithm the representative workers are identified from a set of workers ( $W^0$ ) which are randomly selected from all available workers ( $W$ ) during the initialization phase (see Step 2). This means that CA-FL can always use less number of workers than the FL-baseline in training of FL, i.e.  $k$  is always smaller than  $|W^0|$  (see Step 2 in the Iteration Phase).

TABLE III

F1 SCORES AND COMMUNICATION COSTS BY FL-BASELINE (TOP) AND CA-FL (BOTTOM) ON PAMAP2 DATASET UNDER NON-IID DATA

FL-baseline		
No. Rounds	F1 Score	Communication Overhead (MB)
20	0.943	16.606
40	0.944	32.839
60	0.940	49.149
80	0.943	65.439
100	0.953	81.640

CA-FL			
No. Rounds	F1 Score	Communication Overhead (MB)	
		Iteration Phase	Overall Costs
20	0.947	15.575	22.673
40	0.950	31.490	38.587
60	0.951	48.336	55.433
80	0.954	63.900	70.998
100	0.961	81.324	88.421

Finally, we conduct experiments on both datasets for the two studied experimental scenarios (IID and non-IID) in which the FL-baseline algorithm (FedAvg) uses in the training process the five most frequently selected representatives pointed out by our CA-FL algorithm. These are given in bold in Table I (13, 14, 16, 18 and 19). Notice that the used five workers have produced higher F1 scores for the built ML model than those randomly selected eight workers used to generate the results given in Table III. For example, one can compare the scores produced on Pamap2 non-IID experimental datasets (the last column in Table IV) with the F1 scores in Table III. Therefore, we believe the quality of data of the five most frequently selected workers by CA-FL algorithm is better than of those that are randomly selected by FedAvg for the different number of rounds. These initial results are interesting and worth to be further studied in our future work.

## VI. CONCLUSION AND FUTURE WORK

In this paper, we have proposed a FL algorithm (entitled CA-FL), that can mitigate communication overheads via clustering analysis of the worker local updates. The key idea is to cluster the participating workers into groups based on the similarity of their local model parameters. The most representative workers from each cluster are then selected and used in each training round to optimize the global model. CA-FL model has been evaluated on two human activity recognition

TABLE IV

F1 SCORES OF FL-BASELINE ON mHEALTH AND PAMAP2 DATASETS, WHERE THE SERVER USES ONLY THE MOST FREQUENTLY SELECTED REPRESENTATIVES TO TRAIN THE GLOBAL MODEL IN DIFFERENT EXPERIMENTAL SCENARIOS (IID AND NON-IID)

No. Rounds	mHealth		Pamap2	
	IID	Non-IID	IID	Non-IID
20	0.556	0.574	0.962	0.952
40	0.553	0.551	0.958	0.951
60	0.561	0.568	0.943	0.944
80	0.556	0.553	0.951	0.966
100	0.567	0.569	0.945	0.957

datasets in terms of accuracy and communication costs. The CA-FL algorithm has been benchmarked against FL-baseline algorithm under different experimental scenarios. We have studied how the classification performance and communication costs of the two algorithms are affected by the number of workers under two data distribution settings (IID and non-IID). The obtained results have shown that CA-FL can achieve substantial reduction of communication costs in comparison with the traditional FL-baseline without losing the learning correctness.

Our future plans are to pursue further study and evaluate CA-FL algorithm potential for different ML tasks on richer real-world data sets in different application scenarios. We also plan to compare the proposed FL algorithm with some other state-of-the-art communication-efficient FL algorithms.

#### ACKNOWLEDGMENTS

The first author is supported by an Iraq Ministry of Higher Education and Scientific Research PhD Scholarship.

#### REFERENCES

- [1] B. McMahan and D. Ramage, "Federated learning: Collaborative machine learning without centralized training data," *Google Research Blog*, April 2017. [Online]. Available: <https://ai.googleblog.com/2017/04/federated-learning-collaborative.html>
- [2] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," *arXiv preprint arXiv:1602.05629*, 2016.
- [3] W.-T. Chang and R. Tandon, "Communication efficient federated learning over multiple access channels," *arXiv preprint arxiv:2001.08737*, 2020.
- [4] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.
- [5] L. Wang, W. Wang, and B. Li, "Cmfl: Mitigating communication overhead for federated learning," *IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 954–964, 2019.
- [6] M. Asad, A. Moustafa, T. Ito, and A. Muhammad, "Evaluating the communication efficiency in federated learning algorithms," *arXiv preprint arXiv:2004.02738*, 2020.
- [7] F. Seide, H. Fu, J. Droppo, G. Li, and D. Yu, "1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns," in *INTERSPEECH*, 2014.
- [8] W. Wen, C. Xu, F. Yan, C. Wu, Y. Wang, Y. Chen, and H. Li, "Terngrad: Ternary gradients to reduce communication in distributed deep learning," in *NIPS*, 2017.
- [9] S. Zhou, Z. Ni, X. Zhou, H. Wen, Y. Wu, and Y. Zou, "Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients," *arXiv preprint arXiv:1606.06160*, 2016.
- [10] A. F. Aji and K. Heafield, "Sparse communication for distributed gradient descent," *arXiv preprint arXiv:1704.05021*, 2017.
- [11] M. Asad, A. Moustafa, and T. Ito, "Fedopt: Towards communication efficiency and privacy preservation in federated learning," *Applied Sciences*, vol. 10, 2020.
- [12] K. Hsieh, A. Harlap, N. Vijaykumar, D. Konomis, G. Ganger, P. B. Gibbons, and O. Mutlu, "Gaia: Geo-distributed machine learning approaching LAN speeds," in *NSDI*, 2017.
- [13] Y. Chen, X. Sun, and Y. Jin, "Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, pp. 4229–4238, 2020.
- [14] E. Jeong, S. Oh, H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data," *arXiv preprint arXiv:1811.11479*, 2018.
- [15] X. Wu, Z. Liang, and J. Wang, "Fedmed: A federated learning framework for language modeling," *Sensors (Basel, Switzerland)*, vol. 20, 2020.
- [16] A. M. Elbir, "Hybrid federated and centralized learning," *arXiv preprint arXiv:2011.06892*, 2020.
- [17] S. Caldas, J. Konečný, H. McMahan, and A. Talwalkar, "Expanding the reach of federated learning by reducing client resource requirements," *arXiv preprint arXiv:1812.07210*, 2018.
- [18] Y. Liu, X. Yuan, R. Zhao, Y. Zheng, and Y. Zheng, "Re-ssfl: Towards robust and communication-efficient semi-supervised federated learning system," *arXiv preprint arXiv:2012.04432*, 2020.
- [19] H. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Federated learning of deep networks using model averaging," *arXiv preprint arXiv:1602.05629*, 2016.
- [20] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Lucien M. Le Cam and Jerzy Neyman, editors, Proceedings of the Berkley symposium on mathematical statistics and probability*, vol. 1, pp. 281–297, 1967.
- [21] P. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987.
- [22] L. Vendramin, R. Campello, and E. Hruschka, "Relative clustering validity criteria: A comparative overview," *Statistical Analysis and Data Mining*, vol. 3, pp. 209–235, 2010.
- [23] O. Baños, R. García, J. A. H. Terriza, M. Damas, H. Pomares, I. Rojas, A. Saez, and C. Villalonga, "mhealthdroid: A novel framework for agile development of mobile health applications," in *IWAAL*, 2014.
- [24] A. Reiss and D. Stricker, "Introducing a new benchmarked dataset for activity monitoring," *2012 16th International Symposium on Wearable Computers*, pp. 108–109, 2012.
- [25] E. Babu, D. Rueckert, and A. Davison, "Federated deep learning for healthcare data," *Department of Computing, Imperial College London*, June. 13, 2020.