Fed-RAC: Resource-Aware Clustering for Tackling Heterogeneity of Participants in Federated Learning

Rahul Mishra, *Member, IEEE*, Hari Prabhat Gupta, *Senior Member, IEEE*, Garvit Banga, and Sajal K. Das, *Fellow, IEEE*

***Abstract*—Federated Learning is a training framework that enables multiple participants to collaboratively train a shared model while preserving data privacy. The heterogeneity of devices and networking resources of the participants delay the training and aggregation. The paper introduces a novel approach to federated learning by incorporating resource-aware clustering. This method addresses the challenges posed by the diverse devices and networking resources among participants. Unlike static clustering approaches, this paper proposes a dynamic method to determine the optimal number of clusters using Dunn Indices. It enables adaptability to the varying heterogeneity levels among participants, ensuring a responsive and customized approach to clustering. Next, the paper goes beyond empirical observations by providing a mathematical derivation of the communication rounds for convergence within each cluster. Further, the participant assignment mechanism adds a layer of sophistication and ensures that devices and networking resources are allocated optimally. Afterwards, we incorporate a leader-follower technique, particularly through knowledge distillation, which improves the performance of lightweight models within clusters. Finally, experiments are conducted to validate the approach and to compare it with state-of-the-art. The results demonstrated an accuracy improvement of over 3% compared to its closest competitor and a reduction in communication rounds of around 10%.**

***Index Terms*—Federated learning, heterogeneity, leader-follower technique, resource aware clustering.**

I. Introduction

FEDERATED Learning (FL) is a newly emerging paradigm that enables a distributed training framework where data collection and model training occur locally for each participant. Thus, it preserves data privacy and reduces the communication overhead of transmitting data to the server [1]. Unlike traditional distributed training frameworks that require consensus after each local iteration, either through server or peer communication, FL minimizes the frequency of consensus among distributed participants. FL is initiated by the central server, which broadcasts a randomly initialized model to all participants. Each participant trains the received model using their local dataset and sends the Weight Parameter Matrices (WPM) to the server. The server then aggregates the WPM received from multiple participants and sends back the aggregated one, generating a robust and generalized model [2].

FL participants exhibit significant heterogeneity in terms of devices and networking resources, including processing speed, available memory, and data transmission rate. Each participant uses its resources to load the model and train it locally. The availability of device resources among participants depends on their respective configurations and installed services, leading to irregular intervals between WPM generation [3], [4]. Furthermore, the data transmission rate affects the time required to upload WPM from participants to the server. Consequently, participant heterogeneity hinders the simultaneous transmission and aggregation of WPM. Thus, slower participants (i.e., stragglers) delay the entire training process. The server can mitigate this issue by setting a Maximum Allowable Response (MAR) time for training to minimise the delay caused by stragglers. However, using a fixed MAR time can result in inadequate training due to a reduced number of local updates.

Previous research on FL has addressed the challenge of participant heterogeneity, excluding stragglers from the training process [5]. However, this approach comes with a drawback, as it prevents the system access to valuable datasets held by stragglers, consequently diminishing the model's generalization ability. Next, cluster-based techniques have been proposed in the literature to address participant heterogeneity in FL. These methods leverage various factors such as the relationship between local datasets [5], the similarity of local updates [6], and social relationships between participants [7] to form clusters. However, a notable gap in these studies is the oversight of considering the devices and networking resources of participants during the clustering process. Further, the researchers [8] have highlighted the challenge posed by heterogeneous devices in FL, restricting the size of the global model to accommodate stragglers. Similarly, in the proposal by [9], a technique named HeteroFL was introduced

Manuscript received 17 June 2023; revised 21 February 2024; accepted 4 March 2024. Date of publication 20 March 2024; date of current version 20 May 2024. The work of Hari Prabhat was supported in part by I-DAPT HUB FOUNDATION and in part by SERB under Grant I-DAPT/IIT (BHU)/2023-24/Project Sanction/43 and Grant MTR/2023/000764. The work of Sajal K. Das was supported in part by NSF through FLINT: Robust Federated Learning for Internet of Things under Grant CNS-2008878 and through CANDY: Cyberinfrastructure for Accelerating Innovation in Network Dynamics under Grant OAC-2104078. Recommended for acceptance by M. Li. *(Corresponding author: Rahul Mishra.)*

Rahul Mishra is with the Department of Computer Science and Engineering, Institute of Technology Patna, Patna 801106, India (e-mail: rahul_mishra@iitp.ac.in).

Hari Prabhat Gupta is with the Department of Computer Science and Engineering, Institute of Technology (BHU), Varanasi 221005, India (e-mail: hariprabhat.cse@iitbhu.ac.in).

Garvit Banga is with the Department of Metallurgical Engineering, Indian Institute of Technology (BHU), Varanasi 221005, India (e-mail: garvit.banga.met17@iitbhu.ac.in).

Sajal K. Das is with the Department of Computer Science, Missouri University of Science and Technology, Rolla, MO 65409 USA (e-mail: sdas@mst.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TPDS.2024.3379933, provided by the authors.

Digital Object Identifier 10.1109/TPDS.2024.3379933

to address variations in computational and communication resources by generating multiple-sized models and selecting the optimal one for each participant. Despite these advancements, neither [8] nor [9] have adequately tackled the issue of enhancing the performance of lightweight models used by participants with limited resources. Finally, the work presented in [10], [11] has explored the application of Knowledge Distillation (KD) to improve the performance of lightweight models, which is similar to performance enhancement in the proposed work.

This paper presents a novel approach called Fed-RAC (short for Federated learning with Resource Aware Clustering) to address the negative impact of participant heterogeneity in Federated Learning. We investigate the effect of participant heterogeneity and determine an expression for the required communication rounds per cluster. Fed-RAC is also designed to estimate the error caused by inconsistent objective functions in the presence of heterogeneous devices and networking resources. In particular, we focus on investigating the following problem: *"How can we achieve satisfactory performance while training local models on heterogeneous participants in FL within the given MAR?"* To this end, the major contributions and novelty of this work are as follows:

- *Resource aware clustering:* The first contribution is to conduct resource-aware clustering for identifying the most suitable number of clusters based on the devices and networking resources available to the participants. The server first gathers information regarding the processing speed, data transmission rate, and available memory of all participants to create resource vectors. These vectors are then subjected to unit-based normalization to bring their values within the range of [0,1]. To determine the optimal number of clusters, the server calculates the Dunn Indices [12] among the normalized resource vectors of all participants.
- *Participants assignment to the clusters:* The next contribution is the allocation of participants to the identified clusters, ensuring that the model training within each cluster is performed within a specified maximum allowable response time and communication rounds. Additionally, a mathematical analysis is carried out to derive the expression for the communication round and error caused by an inconsistent objective function in the presence of heterogeneous participants.
- *Leader-follower technique:* Further, our approach introduces the leader-follower technique to enhance the performance of the generic model in low-configuration clusters (followers) by leveraging the model of the highest configuration cluster (leader). In this technique, the leader model is initially trained, and then it guides the training of follower models using knowledge distillation to improve their performance.
- *Experimental validation:* In the end, we conduct experimental evaluations to confirm the effectiveness of the Fed-RAC approach. We validate our proposed method by comparing it with existing baseline techniques [9], [13], [14], [15], using various evaluation metrics and established datasets [16], [17], [18], [19]. The results demonstrate that the proposed approach achieves better performance in the presence of heterogeneous participants.

Paper Organization: Section II provides an overview of the related literature. Section III outlines the preliminary information and problem statement of our proposed approach. Section IV details the Fed-RAC approach and System Implementation in Section V. Section VI evaluates the performance of our approach, while Section VII presents the discussion and future directions for this work. Finally, Section VIII concludes the paper.

II. Background and Motivation

- *Heterogeneous participants in FL:* FL involves a significant number of participant devices with varying resources, leading to degraded performance and increased convergence time when running the same model on all participants. The authors in [8] identified the problem of heterogeneous devices in FL, which limits the size of the global model to accommodate low-resource or slow participants. They proposed a dynamically adaptive approach to model size called ordered dropout, FjORD. HeteroFL [9] introduced the technique to handle variation in computational and communication resources. TailorFL [20] introduced a dual-personalized FL system to address system and data heterogeneity, involving tailored model updates for individual devices and a global aggregation. Next, the authors in [21] involved adaptive model quantization to address device heterogeneity, dynamically adjusting model precision during the federated learning process for improved collaboration among diverse devices. FedRolex [22] proposed FL with rolling sub-model extraction, allowing devices with diverse model architectures to collaboratively train, extract sub-models, and contribute to a shared model.

In previous studies, mechanisms proposed to address the issue of stragglers in FL, including asynchronous [23], [24] and semi-synchronous [25] global update approaches. The authors in [23] introduced an asynchronous algorithm to optimize the FL-based training for stragglers. The algorithm solved the local regularization to ensure convergence in finite time and performed a weighted average to update the global model. Similarly, the authors in [24] introduced the mechanism of asynchronous learning and weighted temporal aggregation on participants and server, respectively. To overcome the problem of higher waiting time, the authors in [25] introduced the semi-asynchronous mechanism, where the server aggregates the weight from a set of participants as per their arrival order.

- *Clustering in FL:* The prior studies utilized the relationship between local datasets [5], the similarity of local updates [6], and social relationship between the participants [7] to form clusters in FL. Authors in [5] exploited the intrinsic relationship between local datasets of multiple participants and proposed a similarity-aware system, namely ClusterFL. The system generated various clusters based on the similarity among local datasets. In [6] authors introduced a modified FL approach, where hierarchical clustering is performed as per the similarity of local updates. Similarly, the authors in [14] proposed a tier-based FL, TiFL, that operates by dividing participants into tiers based on their computational capabilities. The authors in [15] utilized

a hierarchical FL approach by shaping data distribution at the edge to enhance communication efficiency through a multi-level aggregation.

• *KD-based performance improvement:* The existing literature introduced various techniques to improve the performance of the lightweight model using a large-size model via KD [10], [11]. The concept of KD was first introduced by the authors in [10], where the knowledge of a large-size model (teacher) is utilized to improve the performance of the lightweight model (student). The authors in [26] proposed a method for achieving personalized edge intelligence through federated learning and self-knowledge distillation. Next, the authors in [11] introduced pre-trained and scratch teacher-guided KD techniques to improve the performance of students. In [27], the authors proposed FedMD introduces a methodology for heterogeneous FL through model distillation, leveraging a process where a central server distils a unified model from various device-specific models with diverse datasets. Further, the authors in [28] centred on utilizing ensemble distillation to enhance model fusion in FL, involving training an ensemble on global model updates and distilling ensemble knowledge into a centralized model. FedGKT [29] methodology focused on FL, utilizing group knowledge transfer to facilitate the training of large convolutional neural networks at the edge devices.

Motivation: We observed the following limitations in existing literature. Prior studies discarded stragglers from the training to cope up with the heterogeneity of the resources among participants in FL [5]. When the stragglers are discarded, their available local datasets are not utilized during training, which reduces the generalization ability of all the participants. In addition, discarding slow participants hampered their performance improvement via FL. The asynchronous federated learning mechanisms [23], [24] demand the server to wait for stragglers, leading to significant waiting time. The semi-asynchronous global aggregation mechanism [25] is more effective than synchronous, but it discards some participants in each communication round. Suppressing the communication round for aggregation [30] also increases the stale models at participants. The existing work exploited clustering in FL but not considered the devices and networking resources [5], [6], [7].

Communication overhead due to a hierarchy and sensitivity to edge device heterogeneity is observed in [15]. The knowledge distillation poses potential information loss when the teacher model is randomly initialized and training may be delayed when dealing with large model sizes, depending on the resource availability of participants in [27]. Additionally, concerns regarding computational overhead in ensemble training, as discussed in [28], further motivate the exploration of more efficient approaches. Challenges encompassing dual-personalization, increased computational demands, and sensitivity to system and data variations, as indicated in [20], underscore the need for tailored solutions. FedRolex [22], facing complexities in managing heterogeneous models and communication overhead, highlights the demand for effective sub-model compression. Finally, the issues of increased computational demands, communication, and the necessity for efficient coordination in knowledge transfer, as evident in [29].

III. Preliminaries and Problem Statement

A. Preliminaries

This work considers a set $\mathcal{P}$ of N participants and a central server, where $\mathcal{P} = \{p_1, \ldots, p_N\}$. We consider a multi-class classification problem with a set Q of c classes, i.e., $Q = \{1, \ldots, c\}$. Each participant p_i has a local dataset $\mathcal{D}_i$ with n_i number of instances and set of Q classes, where $1 \leq i \leq N$. Let $(\mathbf{x}_{ij}, y_{ij})$ denotes an instance of dataset $\mathcal{D}_i$, where $1 \leq j \leq n_i$. During training the model on the participant p_i learn the mapping between $\mathbf{x}_{ij}$ and $y_{ij}, \forall j \in \{1 \leq j \leq n_i\}$, to build a classifier Π_i. The classifier recognizes the class label of unidentified instances in testing. Let B_i denote the batch size used for the training model on p_i. Further, let τ_i represent the number of Stochastic Gradient Descent (SGD) operations performed in one round of training on p_i. τ_i is estimated as: $\tau_i = \lfloor En_i/B_i \rfloor$, where E is the number of local epochs to train on p_i. We can change B_i and n_i to change τ_i.

B. FL With Heterogeneous Participants

FL begins with the generation and random initialization of a model at the central server that further broadcasts the initialized model to all the participants. Each participant p_i receives and trains the model using local dataset $\mathcal{D}_i$ with n_i instances, where $1 \leq i \leq N$. p_i performs training for E number of local epochs on a batch size of B_i over n_i instances using SGD operations τ_i. The participant minimizes the local loss function $\mathcal{L}_i(\mathbf{w}_i)$, where $\mathbf{w}_i$ is the WPM of p_i. $\mathcal{L}_i(\mathbf{w}_i)$ is estimated as: $\mathcal{L}_i(\mathbf{w}_i) = \frac{1}{n_i} \sum_{j \leftarrow 1}^{n_i} \mathcal{L}_{ij}(\mathbf{w}_{ij})$, where $\mathbf{w}_{ij} \in \mathbf{w}_i$, $1 \leq j \leq n_i$, and $1 \leq i \leq N$. The participant transfers estimated $\mathcal{L}_i(\cdot)$ and $\mathbf{w}_i$ to the server for global aggregation. Upon receiving a local loss and WPM from all the participants, the server estimates global loss ($\mathcal{L}(\mathbf{w})$) and WPM ($\mathbf{w}$) as:

$$\mathcal{L}(\mathbf{w}) = \sum_{i \leftarrow 1}^{N} \left(\frac{n_i}{n_1 \ldots n_N} \right) \mathcal{L}_i(\mathbf{w}_i), \mathbf{w} = \sum_{i \leftarrow 1}^{N} \left(\frac{n_i}{n_1 \ldots n_N} \right) \mathbf{w}_i.$$

The server broadcasts $\mathbf{w}$ for the next round of training. The process of local training and aggregation are orchestrated for $\mathcal{R}$ iterations to achieve a trained model for all the participants. At each global iteration $t \in \mathcal{R}$ the local loss function and WPM are denoted as $\mathcal{L}_i^t(\mathbf{w}_i^t)$ and $\mathbf{w}_i^t$ for participant p_i, respectively, where $1 \leq i \leq N$. $\mathbf{w}_i^t$ at global iteration t ($t \in \mathcal{R}$) of participant p_i is updated as: $\mathbf{w}_i^t = \mathbf{w}_i^{t-1} - \eta \nabla \mathcal{L}_i^t(\mathbf{w}_i^t)$, where η is the learning rate. Using the above equation, we can define the objective function of FL as follows:

$$\min_{\mathbf{w}^{\mathcal{R}}} \mathcal{L}(\mathbf{w}^{\mathcal{R}}) = \sum_{t \leftarrow 1}^{\mathcal{R}} \sum_{i \leftarrow 1}^{N} \left(\frac{n_i}{n_1 + \ldots + n_N} \right) \mathcal{L}_i^t(\mathbf{w}_i^t). \quad (1)$$

1) Heterogeneous Participants: The heterogeneous participants in FL require non-identical training and communication time. Let T_i denotes training and communication time of p_i, estimated as: $T_i = T_i^a.E + T_i^c, \forall i \in \{1, 2, \ldots N\}$, where T_i^a is the training time for one local epoch, and T_i^c is the per-round communication time for sharing WPM from p_i to server. The participants trained the local model and communicated WPM

in parallel. Thus, for each iteration $t \in \mathcal{R}$ the training and communication time T^t depends on the slowest participant, where $T^t = \max_{1 \leq i \leq N}\{T_i\}$. We obtain total training time, denoted as $\mathbb{T}(N, E, \mathcal{R})$, as:

$$\mathbb{T}(N, E, \mathcal{R}) = \sum_{t \leftarrow 1}^{\mathcal{R}} T^t = \sum_{t \leftarrow 1}^{\mathcal{R}} \max_{1 \leq i \leq N}\{T_i\}. \tag{2}$$

2) Objective Inconsistency: The server has a fixed MAR time to complete the global iterations, which reduces the training delay due to slow processing and communication of stragglers. It also minimizes the idle time of faster participants. However, the number of local SGD operations varies over heterogeneous participants within the fixed MAR time. The faster participants perform more local updates in contrast with stragglers. In addition, the number of local updates on the participants also varies across the communication rounds. The objective function of FL given in (1) relies upon the assumptions that the number of local updates, τ_i for p_i $\forall i \in \{1, 2, \ldots, N\}$, remain the same for all participants ($\tau_i = \tau$). However, the variation in the local updates on the heterogeneous participants results in an inconsistent objective function for FL [31]. Let $\bar{\mathcal{L}}(\bar{\mathbf{w}}^{\mathcal{R}})$ denotes the inconsistent objective function, where $\bar{\mathbf{w}}^{\mathcal{R}}$ is the aggregated WPM generated after $\mathcal{R}$ global iterations. The error (err) between actual and inconsistent objective function is defined as $err = |\bar{\mathcal{L}}(\bar{\mathbf{w}}^{\mathcal{R}}) - \mathcal{L}(\mathbf{w}^{\mathcal{R}})|$.

C. Problem Statement and Solution Overview

The fundamental challenges encountered while developing an FL approach to mitigate the heterogeneity are: *1) how to reduce the training and communication time of the stragglers in FL? 2) how to achieve adequate performance within the fixed time interval for communication? and 3) how to minimize the error gap between actual and inconsistent objective functions due to heterogeneous participants.* In this work, we investigate and solve the problem of *training the local model on all the heterogeneous participants within a given maximum allowable response time, achieving adequate performance and minimizing error due to inconsistent objective function.*

Apart from the standard FL workflow, the Fed-RAC trains the local models on all the participants despite higher heterogeneity and reduces training time without compromising performance. Fed-RAC starts with the estimation of the optimal number of clusters to accommodate all N heterogeneous participants. We named the step as *resource aware clustering* (Section IV-A). During clustering, a set $\mathcal{K}$ of k clusters is first identified (Section IV-A1), followed by the generation of a generic model for each cluster (Section IV-A2). Next, the participants are assigned to the empty clusters using *participant assignment mechanism* (Section IV-B). Further, we introduce *leader-follower technique* (Section IV-C) to enhance the performance of the generic models using KD.

IV. Fed-RAC: Federated Learning Via Resource Aware Clustering

In this section, we first cover the details of the Federated learning approach to mitigate the heterogeneity of participants using Resource Aware Clustering (Fed-RAC). The workflow of the Fed-RAC is shown in Fig. 1.

A. Resource Aware Clustering

This sub-section describes the mechanism of dividing the set of N participants into k disjoint clusters. The clustering is performed on the server to preserve the resources of the participants. In doing so, the server fetches three resources from all the participants, i.e., processing speed, data transmission rate, and available memory, denoted as s_i, r_i, and a_i for p_i ($1 \leq i \leq N$), respectively. s_i and a_i are the machine-dependent parameters that rely upon the configuration of the devices. The data transmission rate r_i depends on the bandwidth, channel coefficient, and path loss between participant and server and is estimated. The static information of s_i, r_i, and a_i from the participants are used to initialise the Fed-RAC approach. Afterwards, the approach provides the opportunity to upgrade or downgrade the cluster depending on the available dynamic resources of the participants. If a participant is in the smallest cluster and its resources are dynamically reduced then Fed-RAC sets batch-size and local epochs to continue the training, as discussed in Section IV-B3. It implies the Fed-RAC can easily tackle the dynamic resources of the participants in FL.

All the participants of a cluster possess similar processing speeds, transmission rates, and memory. However, it is tedious to determine the similarity among the three independent resources. Thus, we use a vector $v_i = [s_i, r_i, a_i]$ for participant p_i ($1 \leq i \leq N$) to estimate similarity among resources. We use normalize vector $\bar{v_i} = [\bar{s_i}, \bar{r_i}, \bar{a_i}]$ in place of v_i, to eliminate impact of biasness of high values. The bias value $\bar{s_i}$ is estimated as: $\bar{s_i} = \frac{s_i - \min\{s_i\}_{i=1}^{N}}{\max\{s_i\}_{i=1}^{N} - \min\{s_i\}_{i=1}^{N}}$, $\bar{r_i}$ and $\bar{a_i}$ are also estimated similarly. We further estimate the similarity ($\mathcal{S}_{ij}$) among any two participant p_i and p_j using normalized vectors $\bar{v_i}$ and $\bar{v_j}$, respectively, $\forall i, j \in \{1, 2, \ldots, N\}$ using euclidean distance. $\mathcal{S}_{ij}$ is estimated as: $\mathcal{S}_{ij} = \sqrt{\lambda_1(\bar{s_i} - \bar{s_j})^2 + \lambda_2(\bar{r_i} - \bar{r_j})^2 + \lambda_3(\bar{a_i} - \bar{a_j})^2}$, where λ_1, λ_2 and λ_3 are the contributions of processing capacity, transmission rate, and memory, respectively, $\lambda_1 + \lambda_2 + \lambda_3 = 1$. λ_1, λ_2, and λ_3 can be obtained from [32], [33].

1) Estimating Optimal Number of Cluster k: We introduce a modified version of the conventional Dunn and Dunn-like Indices [12] to estimate the optimal number of clusters using similarity. We use k-means clustering to determine the optimal number of clusters. Dunn index identifies an optimal number of clusters that hold compactness and provide good separation. Let C_f and C_g denote clusters in $\mathcal{K}$ ($C_f, C_g \in \{C_1, \ldots, C_k\}, C_f \neq C_g$). The least distance $\mathbf{dist}(C_f, C_g)$ among C_f and C_g is given as:

$$\mathbf{dist}(C_f, C_g) = \min_{p_i \in C_f, p_j \in C_g, C_f \neq C_g} \mathcal{S}_{ij}. \tag{3}$$

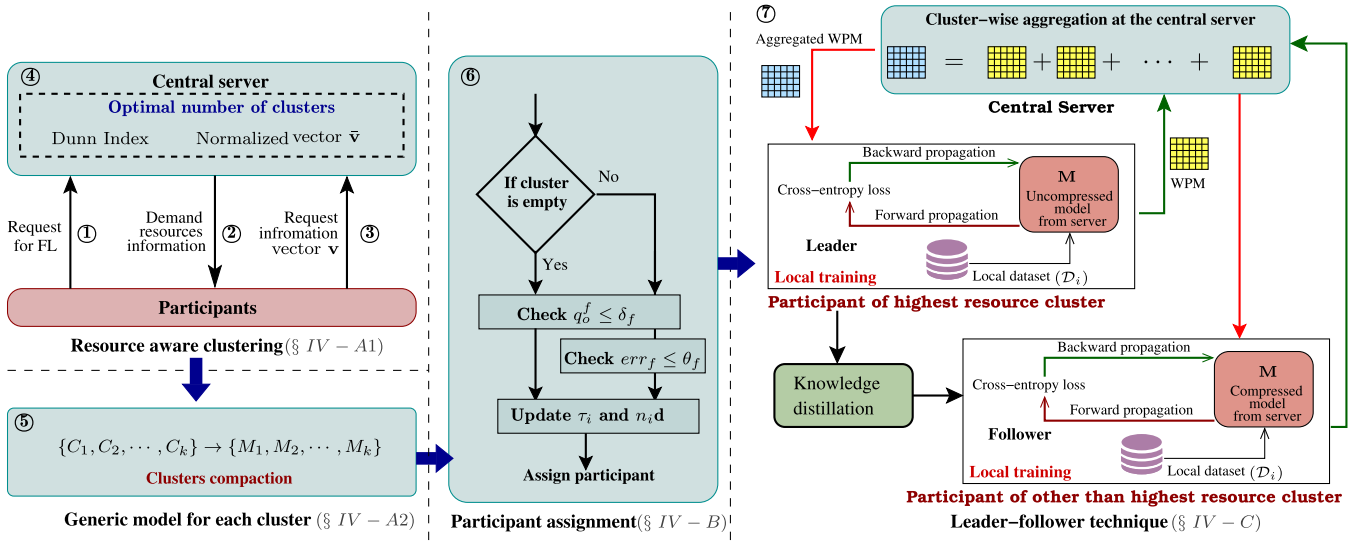


Fig. 1. Workflow for Fed-RAC approach. ○1–○4 steps for resource aware clustering, ○5 generating generic model for each cluster, ○6 participants assignment to the clusters, and ○7 leader follower technique to improve performance.

TABLE I
ILLUSTRATION OF AN EXAMPLE SCENARIO HAVING TEN PARTICIPANTS WITH RESOURCE VECTORS AND NORMALIZED RESOURCE VECTORS

Participant	Resource vector	Normalized resource vector
p_1	$v_1 = [100, 10, 20]$	$\bar{v_1} = [0.5, 0.375, 0.5]$
p_2	$v_2 = [50, 15, 30]$	$\bar{v_2} = [0, 1, 1]$
p_3	$v_3 = [75, 8, 25]$	$\bar{v_3} = [0.25, 0.125, 0.75]$
p_4	$v_4 = [125, 10, 15]$	$\bar{v_4} = [0.25, 0.625, 0.75]$
p_5	$v_5 = [150, 7, 10]$	$\bar{v_5} = [1, 0, 0]$
p_6	$v_6 = [110, 10, 25]$	$\bar{v_6} = [0.6, 0.375, 0.75]$
p_7	$v_7 = [125, 15, 20]$	$\bar{v_7} = [0.75, 1, 0.5]$
p_8	$v_8 = [80, 10, 10]$	$\bar{v_8} = [0.30, 0.375, 0]$
p_9	$v_9 = [75, 15, 20]$	$\bar{v_9} = [0.25, 1, 0.5]$
p_{10}	$v_{10} = [50, 10, 30]$	$\bar{v_{10}} = [0, 0.375, 1]$

The diameter $\mathbf{dia}(C_f)$ of cluster $C_f \in \{C_1, \ldots, C_k\}$ is the distance between participants in C_f. Let p_l^f and p_q^f be the two participants in C_f ($p_l^f \neq p_q^f$), $\mathbf{dia}(C_f)$ is estimated as:

$$\mathbf{dia}(C_f) = \max_{p_l^f, p_q^f \in C_f, p_l^f \neq p_q^f} \mathcal{S}_{lq}^f. \tag{4}$$

Using (3) and (4), we estimate Dunn Index ($DI(k)$) as:

$$DI(k) = \min_{\forall C_f \in \mathcal{K}} \left[\min_{\forall C_g \in \mathcal{K}, C_f \neq C_g} \left(\frac{\mathbf{dist}(C_f, C_g)}{\max_{\forall C_f \in \mathcal{K}} \mathbf{dia}(C_f)} \right) \right]. \tag{5}$$

A high positive value of $DI(\cdot)$ indicates a compact and adequate number of clusters. The divergence-based Dunn and Dunn-like Indices start with $k = 2$ and terminate when $DI(\cdot)$ achieves a higher positive value. We use the maximum number of clusters $k_{max} \leq \sqrt{N}$ as the rule of thumb, inspired from [34]. The complete steps to obtain the optimal number of clusters are given in Procedure 1.

Example 1: Let there are 10 participants denoted as $p_1, \ldots, p_{10}$. The resource and normalized vectors of the example are shown in Table I. Using Procedure 1 with $\lambda_1 = \lambda_2 = \lambda_3 = 1/3$, we obtain $k = 3$ as optimal clusters.

Procedure 1: Optimal Number of Clusters.

Input: Set of N participants $\mathcal{P}$ in FL;
Output: Optimal set of k clusters $\mathcal{K} = \{C_1, C_2, \cdots, C_k\}$;
Initialization: $j \leftarrow 0$, $C_s \leftarrow []$, $\mathcal{K} = \{\}$, $k \leftarrow 2$;
for *each participant* $p_i \in \{p_1, p_2, \cdots p_N\}$ **do**
  Server extracts information of s_i, r_i, and a_i from p_i;
  Estimate resource vector v_i;
for *each participant* $p_i \in \{p_1, p_2, \cdots p_N\}$ **do**
  Estimate $\bar{s_i}$, $\bar{r_i}$, and $\bar{a_i}$ for p_i and vector $\bar{v_i}$;
while $k \leq \sqrt{N}$ **do**
  Perform k-mean and estimate similarity among vectors;
  for *each pair* $C_f, C_g \in \{C_1, C_2, \cdots, C_k\}, C_f \neq C_g$ **do**
    Estimate Dunn index ($DI(k)$) using (5);
  $C_s \leftarrow$ *append*($DI(k)$), $k \leftarrow k + 1$;
$j \leftarrow \arg\max(C_s)$, $k \leftarrow j + 1$; /*Optimal number of clusters*/
return $\mathcal{K} = \{C_1, C_2, \cdots, C_k\}$;

TABLE II
IMPACT OF CLUSTERING TECHNIQUES ON DI VALUES AND ACCURACY AT DIFFERENT VALUES OF k USING MNIST DATASET

Cluster technique	DI values				
	k = 2	**k = 3**	**k = 4**	**k = 5**	**k = 6**
k-means	0.1517	0.1965	0.2165	**0.2317**	0.1750
DBSCAN	**0.2231**	0.1819	0.1642	0.1419	0.1236
OPTICS	0.1165	**0.1208**	0.1037	0.0839	0.0673
Accuracy (±0.30)	94.39%	95.07%	96.32%	**97.73%**	95.67%

Acc = accuracy.

Apart from k-means clustering, we also consider density-based clustering to obtain the optimal number of clusters using normalized resource vectors. We use Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and Ordering points to identify the clustering structure (OPTICS) [35] during the experiment. Table II illustrates the DI-values and accuracy for different k using k-means, DBSCAN, and OPTICS using the resource vectors, discussed in Section VI-E1. From the results

in the table, we observe that for DBSCAN clustering, the DI value decreases with increasing k; thus, it predicts $k = 2$ as an optimal number of the cluster. However, the difference between resources among the participants within a cluster is high, which results in lower accuracy. Moreover, some participants with the least resources can not accommodate a large-size model assigned to the cluster. We can draw similar observations for OPTICS, which gives the optimal number of clusters $k = 3$. k-means clustering results in $k = 5$ optimal number of clusters, where inter-cluster and intra-cluster distances are high and low, respectively. It narrows down the gaps between the resources of the participants within a cluster. Thus, all the participants can easily accommodate the assigned model to a cluster. Such narrow gapping also prevents the bucket effect, where a large model is assigned to the participant with the smallest resources.

2) Generic Model for Each Cluster and Compaction of Clusters: This work considers three resources, i.e., processing speed, data transmission rate, and available memory, to obtain k clusters. However, the cumulative resources are unequal among all the clusters. Thus, the size of the model on the clusters would be non-identical in FL. This work develops a generic model for each cluster and performs cluster compaction afterwards. In doing so, we arrange the k clusters in descending order of their available resources. In other words, the participants in cluster C_1 can train a large-size model and quickly transfer WPM to the server, whereas C_k can train the smallest model and requires more time to share WPM.

Let M denote the initial model generated and randomly initialized by the server. We assume M can directly accommodate on C_1, i.e., training and communication can be performed within the given time. Let M_1 denote the size of the model for C_1, where $M = M_1$. Beyond C_1 other clusters require some compression to train the model and share WPM. Let M_2 denote the compressed version of M that can be deployed on the participants in C_2, consuming less training and communication time. $M_3 - M_k$ are generated for the remaining $k - 2$ clusters. In this work, we consider the model of any cluster C_i is α times smaller than C_{i-1}, i.e., $M_{i-1} = \alpha M_i$, where $\alpha < 1$. It implies $M_k = \alpha^{k-1} M_1 \Rightarrow M_k = \alpha^{k-1} M$.

The compression rate α is not predetermined and invariant. It is determined before the training phase by assessing the resource availability across all participants, and it remains constant during the entire training process. Additionally, it adapts to diverse scenarios, accounting for variations among participants with different resources. Next, to ensure that the α-compressed model aligns with the memory constraints of all participants in real-world applications, our strategy incorporates a dynamic adaptation. It assesses the available resources for each participant and adjusts the compression rate.

• *Cluster compaction:* The estimated k clusters and corresponding models suit the resources of the participants; however, higher compression of the model results in performance compromise. Thus, it is beneficial if all the participants can accommodate in fewer clusters than k. However, it introduces the *straggler effect*, where slow participants do not participate. To overcome the straggler effect, we merge some clusters out of k to obtain m clusters, where $k < m$.

B. Participants Assignment to the Clusters

This sub-section describes the mechanism of assigning N participants to the m clusters. We first deduce the expression to estimate the communication rounds required for the generic model in m different clusters. Next, we define the optimization error due to the heterogeneity of participants. Notably, Fed-RAC initially checks the possibilities of assigning participants to the higher cluster, decreasing as per the assignment criteria.

From Section IV-A2, we have m different models $M_1, M_2, \ldots, M_m$ for clusters $C_1, C_2, \ldots, C_m$, respectively, where the size of models $M_1 > M_2 > \ldots > M_m$ and $M_m = \alpha^{m-1} M_1 = \alpha^{m-1} M$. The server decides $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_m$ communication rounds for training local models of the participants in clusters $C_1, C_2, \ldots, C_m$, respectively. We first determine the expression for communication rounds $\mathcal{R}_f$ for cluster C_f, where $1 \leq f \leq m$.

1) Communication Rounds Per Cluster: Let $\mathcal{P}_f$ denotes the set of F participants to be assigned in C_f, where $\mathcal{P}_f = \{p_1, \ldots, p_F\}$, having loss functions $\mathcal{L}_1, \ldots \mathcal{L}_F$, respectively. We consider the assumptions given in [36] and applied on $\mathcal{L}_1, \ldots, \mathcal{L}_F$ to estimate the round $\mathcal{R}_f$ for cluster C_f.

Assumption 1: Loss $\mathcal{L}_j \in \{\mathcal{L}_1, \mathcal{L}_2, \ldots \mathcal{L}_F\}$ is L-smooth; therefore, for any two WPM $\mathbf{w}_a$ and $\mathbf{w}_b$ on $p_j \in \mathcal{P}_f$, following inequality holds: $\mathcal{L}_j(\mathbf{w}_a) \leq \mathcal{L}_j(\mathbf{w}_b) + (\mathbf{w}_a - \mathbf{w}_b)^T \nabla \mathcal{L}_j(\mathbf{w}_b) + \frac{L}{2} \|\mathbf{w}_a - \mathbf{w}_b\|^2$, where $1 \leq j \leq F$.

Assumption 2: $\mathcal{L}_j$ is μ-strongly convex; the inequality holds: $\mathcal{L}_j(\mathbf{w}_a) \geq \mathcal{L}_j(\mathbf{w}_b) + (\mathbf{w}_a - \mathbf{w}_b)^T \nabla \mathcal{L}_j(\mathbf{w}_b) + \frac{\mu}{2} \|\mathbf{w}_a - \mathbf{w}_b\|^2$.

Assumption 3: Let ε_j^t denote the uniformly and randomly selected sample from the local dataset $\mathcal{D}_j$ of participant p_j on communication round t, where $1 \leq t \leq \mathcal{R}_f$. Let $\nabla \mathcal{L}_j(\varepsilon_j^t, \mathbf{w}_j^t)$ and $\nabla \mathcal{L}_j(\mathbf{w}_j^t)$ denote the gradients of loss function $\mathcal{L}_j(\cdot)$ on ε_j^t samples and entire samples of the local dataset, respectively. The variance of gradients on participant p_j is bounded as: $\mathbb{E}\|\nabla \mathcal{L}_j(\varepsilon_j^t, \mathbf{w}_j^t) - \nabla \mathcal{L}_j(\mathbf{w}_j^t)\|^2 \leq \sigma_f^2$.

Assumption 4: Expected square norm of loss gradient is uniformly bounded as $\mathbb{E}\|\nabla \mathcal{L}_j(\phi_j^t, \mathbf{w}_j^t)\|^2 \leq G_f^2$, $1 \leq t \leq \mathcal{R}_f$ and $1 \leq j \leq F$.

Using Assumptions 1, 2, 3, and 4, we obtain a relation between desired precision (q_o^f), local epoch count E_f, and global iterations $\mathcal{R}^f$ of cluster C_f. The precision is defined as: $q_o^f = \mathbb{E}[\mathcal{L}(\mathbf{w}^{\mathcal{R}_f})] - \mathcal{L}_f^*$, where $\mathbf{w}^{\mathcal{R}_f}$ is the aggregated weight at final global epoch $\mathcal{R}_f$ and $\mathcal{L}_f^*$ is minimum and unknown value of $\mathcal{L}_f$ at the server. Let $\mathcal{L}_j^*$ is the minimum value of $\mathcal{L}_j$ at p_j, where $\forall j \in \{1 \leq j \leq F\}$. In this work, we assume i.i.d datasets on the participants; thus, $\Gamma = \mathcal{L}_f^* - \sum_{i=1}^{F} \mathcal{L}_j^* = 0$, as given [36]. Γ quantifies the degree of non-i.i.d and it goes to zero for i.i.d. Let ϵ_j denotes the weight contribution of participant $p_j \in \mathcal{P}_f$. Let $\beta = \max\{8L/\mu, E_f\}$ and T_f is the total SGD operations on a participant then we obtain the following relation of desired precision (q_o^f) for cluster C_f [36]:

$$\mathbb{E}[\mathcal{L}(\mathbf{w}^{\mathcal{R}_f})] - \mathcal{L}_f^* \leq \frac{L/2\mu^2}{\beta + T_f - 1} \left(4B + \mu^2 \beta \mathbb{E}\| \cdot \|^2\right), \quad (6)$$

where, $B = \sum_{j=1}^{F} \epsilon_j^2 \sigma_f^2 + 8(E-1)^2 G_f^2$. Using upper bound of q_o^f and $T_f = \mathcal{R}_f E_f$, we obtain number of communication

round ($\mathcal{R}_f$) for cluster C_f ($1 \leq f \leq m$) as follows:

$$\mathcal{R}_f = \frac{1}{E_f}\left[\frac{L}{2\mu^2 q_o^f}\left(4B + \mu^2\beta\mathbb{E}\|\mathbf{w}_1 - \mathbf{w}_f^*\|^2\right) + 1 - \beta\right]. \tag{7}$$

From (7), we have fixed communication rounds $\mathcal{R}^f$ for given precision threshold q_o^f and local epochs E_f for cluster C_f, where $1 \leq f \leq m$. In addition, we have $E_f = \frac{B_j \tau_j}{n_j}$; it implies we can change value of B_j, τ_j, and n_j in such a manner, where E_f and R_f remains fixed for $p_j \in \mathcal{P}_f$ and q_o^f changes. We set a threshold over q_o^f, denoted as δ_f for C_f.

2) Optimization Error Due to Participants Heterogeneity: The set of participants $\mathcal{P}_f$ to be assigned in cluster C_f possesses low inter-cluster and high intra-cluster heterogeneity. Therefore, we obtain inconsistency in the objective function of a cluster, discussed in Section III-B2, due to intra-cluster heterogeneity despite using an effective clustering mechanism. To estimate the value of error err_f for cluster C_f, where $C_f \in \{C_1, C_2, \ldots, C_m\}$, we use the assumptions given in [31]. The previous assumptions, i.e., Assumptions 1 and 2 are same for estimating err_f. However, we need to define a new Assumption 5 to calculate err_f.

Assumption 5: Let $\{\epsilon_1, \epsilon_2, \ldots, \epsilon_F\}$ denote a set of weighted contribution of participants in set $\mathcal{P}_f$ of cluster C_f, where $\sum_{j=1}^{F} \epsilon_j = 1$ and $C_f \in \{C_1, \ldots, C_m\}$. There exists two constants $h_1 \geq 1$ and $h_2 \geq 0$ such that $\sum_{j=1}^{F} \epsilon_j \|\nabla \mathcal{L}_j(\mathbf{w}_j)\|^2 \leq h_1^2 \|\sum_{j=1}^{F} \epsilon_j \nabla \mathcal{L}_j(\mathbf{w}_j)\|^2 + h_2^2$.

Using Assumptions 1, 2, and 5, we derive the expression for err_f of cluster C_f. In doing so, let $\mathbf{o}_j$ denote a non-negative vector and define how stochastic gradients are locally accumulated. For example, $\mathbf{o}_j = [1, \ldots, 1] \in \mathbb{R}^{\tau_j}$ for FedAvg [13]. $\|\mathbf{o}_j\|_1$ is the $l1$-norm of $\mathbf{o}_j$ and $o_{[j,-1]}$ is the last element in vector $\mathbf{o}_j$. $\tau_e = \sum_{j=1}^{F} \tau_j / F$, $\tau_j = \lfloor E_f n_j / B_j \rfloor$ and η is the learning rate, where $1 \leq j \leq F$.

$$\begin{aligned} err_f &= \min_{t \in \mathcal{R}_f} \mathbb{E}\left[\|\nabla \bar{\mathcal{L}}(\bar{\mathbf{w}}^t)\|^2\right] \\ &\leq \frac{4b_1}{\eta \tau_e \mathcal{R}_f} + \frac{4\eta L \sigma_f^2 b_2}{F} + 6\eta^2 L^2 \sigma_f^2 b_3 + 12\eta^2 L^2 h_2^2 b_4, \end{aligned} \tag{8}$$

where $b_1 = [\bar{\mathcal{L}}(\bar{\mathbf{w}}^0) - \mathcal{L}_f^*]$, $b_2 = F\tau_e \sum_{j=1}^{F} \frac{\epsilon_j^2 \|\mathbf{o}_j\|_2^2}{\|\mathbf{o}_j\|_1^2}$, $b_3 = \sum_{j=1}^{F} \epsilon_j (\|\mathbf{o}_j\|_2^2 - [o_{j,-1}]^2)$, $b_4 = \max_j\{\|\mathbf{o}_j\|_1 (\|\mathbf{o}_j\|_1 - [o_{j,-1}])\}$. A small err_f indicates lower intra-heterogeneity among the participants. We set error bound for each cluster, i.e., error $err_f \leq \theta_f$ for C_f, where $1 \leq f \leq m$ and $err_f \leq \theta_f$.

3) Participants Assignment: Fed-RAC assigns each participant to an optimal cluster per the available device and networking resources. Such assignment facilitates easier and faster (within MAR time) training and inference of the local model on each participant assigned to a specific cluster. In other words, each participant trains the local model in R_f communication rounds (7) for cluster C_f, $1 \leq f \leq m$. The assignment verifies two conditions: a) precision (6) of cluster C_f must be less than the threshold ($q_o^f \leq \delta_f$) and b) optimization error (8) $err_f \leq \theta_f$. Further, we get two possible cases for assigning participants in each cluster:

- *Case 1 (Cluster is empty):* p_i assigns to empty cluster C_f, if p_i can train the model M_f in given epochs E_f and communication round R_f. The local epoch E_f is high for a single participant as one communication round is required to train the model without multiple participants. In this case, the condition of $q_o^f < \delta_f$ is only verified and the optimization error is zero. It is because the constraint for homogeneity becomes zero with a single participant in (8). If the participant is unable to train M_f in MAR and R_f, it uses the following two steps:
  1) p_i reduces τ_i and n_i, while satisfying $q_o^f \leq \delta_f$.
  2) If $q_o^f \geq \delta_f$ for C_f then the participant switches to the lower cluster and repeats Step 1.
- *Case 2 (Cluster is non-empty):* We initially estimate q_o^f (6). Upon adding p_i to C_f, q_o^f should be less than threshold δ_f. Similar to Case 1, if p_i is incompetent in training M_f in MAR, τ_i and n_i are adjusted until $q_o^f < \delta_f$; otherwise participant switches to the lower cluster. Next, the error (8) is also estimated upon adding p_i to C_f. If estimated $err_f \leq \theta_f$ then p_i is added to C_f, else p_i switches to the lower cluster.

After successfully executing these two cases, N participants are assigned to the m clusters. The assigned participants achieve desired precision and optimization errors less than the corresponding thresholds. The server optimally allocates each participant to a specific cluster as per the resource, precision threshold, and error threshold. Procedure 2 summarizes the steps involved in assigning participants to the clusters.

We prioritize privacy to ensure that only aggregated and anonymized performance data is transmitted to the server. Individual participant details are not exposed, safeguarding the privacy of device performance parameters; however, the server can get the details externally. The first parameter, processing speed, remains constant and is deterministically associated with a specific device manufacturer. Consequently, the server can readily authenticate the accuracy of the processing speed information provided by the participant's device before the commencement of the actual training. With the possibility of participants providing inaccurate information about their transmission rates, the server implements a robust verification process, i.e., data transfer tests. The server initiates data transfer tests by dispatching a randomly initialized weight parameter matrix to all participants in the initial cluster, measuring the time it takes for each device to download it. Subsequently, the server checks the upload time of the trained model's weight matrix after the initial communication round. In cases where a participant reports false information, the server employs its calculated values to reassign it to a different cluster based on its transmission rate.

Further, the participant may falsely report memory information and get a larger model then it may send an updated weight parameter matrix after a delay or can send a partially trained or untrained weight parameter matrix. To circumvent this, the server examines the time taken to receive updated weight matrices from participants. Despite sufficient processing power and transmission rate estimates, a participant consistently exhibits delays in sending updates, and the server intelligently reassigns

Procedure 2: Participants Assignment to the Clusters.

Input: Set of participants $\mathcal{P} = \{p_1, \cdots, p_N\}$;
Initialization: $i \leftarrow 1$, $f \leftarrow 1$;
for *each participant* $p_i \in \{p_1, p_2, \cdots p_N\}$ **do**
  for *each cluster* $C_f \in \{C_1, C_2, \cdots, C_m\}$ **do**
    if $isEmpty(C_f) == True$ **then**
      if p_i *can accommodate* M_f **then**
        Check: Estimate precision q_o^f using (6);
        if $q_o^f \leq \delta_f$ **then**
          Assign p_i to C_f;
        else
          $f \leftarrow f + 1$;
      else
        Reduce τ_i and n_i *s.t.*, p_i can run M_f;
        Goto *Check*;
    else
      if p_i *can accommodate* M_f **then**
        Check-I: Estimate precision q_o^f using (6);
        Calculate error err_f using (8);
        if $q_o^f \leq \delta_f$ *and* $err_f \leq \theta_f$ **then**
          Assign p_i to C_f;
        else
          $f \leftarrow f + 1$;
      else
        Reduce τ_i and n_i *s.t.*, p_i can run M_f;
        Goto *Check-I*;
return Optimal participants in each cluster C_f, $\forall 1 \leq f \leq m$;

the participant to a lower cluster, identifying and addressing potential false information.

C. Leader-Follower Technique

This sub-section introduces the technique of improving the performance of lightweight models $M_2, \ldots, M_m$ in clusters $\{C_2, \ldots, C_m\}$ using generalization ability (or knowledge) of large-size model M_1 in cluster C_1. Along with the logits, the feature information corresponding to the dataset is also provided during the knowledge distillation process. We utilize the assumption that the cluster C_1 is the fastest and can accommodate the server's model without compression, i.e., $M_1 = M$. We use the term *leader* for M_1 and *follower* for models $M_2 - M_m$, thus, named the technique as *leader-follower for performance improvement*. The technique involves the KD technique [10] to improve the performance of the follower model using the trained leader model. MAR time ($\mathbb{T}_{max}$) for training models on all N participants and can be further divided as: $\mathbb{T}_{max} = T_1 + \max\{T_2, T_3, \ldots, T_m\}$, where T_f is the MAR time for training M_f on the participants of C_f, $1 \leq f \leq m$. Since C_m is the slowest cluster and C_1 is the fastest cluster; thus, we can consider the following relation similar to generic models: $T_{f-1} = \kappa T_f$, where $1 \leq f \leq m$ and $\kappa < 1$. It implies $T_1 = \kappa^{m-1} T_m$ then we obtain:

$$\begin{aligned}\mathbb{T}_{max} &= \kappa^{m-1} T_m + \max\{\kappa^{m-2} T_m, \kappa^{m-3} T_m, \ldots, T_m\}, \\ &= \kappa^{m-1} T_m + T_m = (\kappa^{m-1} + 1) T_m. \end{aligned} \tag{9}$$

In special cases, where M_1 is leader of M_2, M_2 is leader of M_3, and so on, i.e., the FL-based training is performed sequentially for each cluster. In this case, $\mathbb{T}_{max}$ is defined as:

$$\begin{aligned}\mathbb{T}_{max} &= \kappa^{m-1} T_m + \kappa^{m-2} T_m + \kappa^{m-3} T_m + \ldots + T_m, \\ &= \{\kappa^{m-1} + \kappa^{m-3} + \ldots + 1\} T_m = \frac{1 - \kappa^m}{1 - \kappa}, \text{ where } \kappa < 1.\end{aligned}$$

This work starts FL-based training from the fastest cluster C_1 with adequate devices and networking resources to train M_1. We train M_1 for E_1 local epochs on the participants of C_1 using $\mathcal{R}_1$ communication rounds. The logits of trained M_1 are next supplied to all the remaining clusters to improve the performance of their generic models using the KD. Algorithm 1 summarizes the steps involved in the Fed-RAC.

By employing a leader model (M1) and follower models (M2-Mk), we aim to facilitate a more controlled and effective knowledge transfer process. The benefits are a) hierarchical knowledge transfer, b) adaptability to varied participant capabilities, and c) enhanced performance (detailed in supplementary [37]). We adjust the learning rate for M2-Mk based on the performance and confidence of M1. When M1's knowledge is less reliable due to a small participant population in C1, lower the learning rate for M2-Mk to make them more receptive to M1's guidance. In addition, we establish clear criteria for model selection. If M1 consistently outperforms M2-Mk on certain metrics, make it a rule to favour M1's updates during the aggregation process. Further, to ensure the higher performance of M1, we selectively aggregate knowledge from the most competent participants to mitigate the impact of a small participant set on the training quality of M1. Additionally, when aggregating updates from multiple models, assign different weights to M1 compared to M2-Mk. This gives M1 a more significant influence on the aggregation.

Fed-RAC commences the training process by training the leader model in C_1, followed by the concurrent training of other models in the remaining clusters through knowledge distillation. To mitigate the extra delay caused by training the leader model, we introduce a workaround. Upon cluster formation, we conduct preliminary local training for the leader model for a few epochs before the actual training begins. To address differences in resource availability among participants across diverse clusters, we modify the sizes of the models. This adjustment aims to minimize delays in aggregation caused by stragglers or participants with lower computational resources.

• *Aggregation of the weight parameter matrices:* To aggregate information, the server employs a layer-wise averaging process across the weight parameter matrices of various models received from the clusters. For instance, let's consider a scenario where only three clusters (C_1, C_2, and C_3) and the server model ms consist of seven layers: one input layer, five hidden layers, and one output layer. Participants in C_1 can train with the full model ($m1 = ms$), while participants in clusters C_2 and C_3 require the removal of one and two hidden layers, respectively. Consequently, the server conducts a layer-wise aggregation of weight parameters, averaging over the available layers, detailed in Supplementary file [37].

Algorithm 1: Fed-RAC Algorithm.

Input: Set $\mathcal{P}$ of N participants with their local datasets;
Call **Procedure 1** to determine a set $\mathcal{K}$ of k clusters;
Merge clusters to obtain m clusters in $\mathcal{K}$, $\{C_1, C_2, \cdots, C_m\}$;
Generate m model for each cluster, $\{M_1, M_2, \cdots, M_m\}$;
/*Participants assignment to the clusters*/
Call **Procedure 2** to assign optimal participants to m clusters;
for *each cluster* $C_f \in \{C_1, C_2, \cdots, C_m\}$ **do**
  if $f == 1$ **then**
    while *communication rounds* $r \leq R_1$ **do**
      Train local models in cluster C_1;
      $r \leftarrow r + 1$;
    Obtained train model M_1 for participants in cluster C_1;
  else
    while *communication rounds* $r \leq R_f$ **do**
      Train local models in cluster C_f under the guidance of model M_1;
      $r \leftarrow r + 1$;
    Obtained train model M_f for participants in cluster C_f;
return Trained model on each participant;

V. System Implementation

The Fed-RAC algorithm and associated procedures were implemented using the Python programming language. The models considered in the study were implemented using the functional API of Keras in Python, chosen for its developer-friendly features. To ensure a fair and comprehensive comparison, all baseline models were reimplemented. Throughout the experiments, the loss function was set to "categorical cross-entropy," and a batch size of 200 was employed. The hyperparameter $\mathcal{L}^*$ ranged between 0.01 and 0.05, with the number of participants denoted as $N = 40$. Local epochs varied across datasets, specifically $E = 1 - 5$ for MNIST and HAR, and $E = 10 - 40$ for CIFAR-10 and SHL. The communication rounds were standardized at 200 for all clusters during the experiments.

Learning rate exploration involved varying it between 0.001 and 0.010. Convolutional layers were selectively compressed to obtain follower models. A dropout of 0.5 was applied, with subsequent layers using fractions of the previous layer's dropout, denoted as $M_2 = 0.5(M_1)$, $M_3 = 0.5(M_2)$, and so forth. This dropout strategy served a dual purpose, contributing to model compression and enhancing training speed and transmission efficiency. The strategic application of dropout introduced regularization, mitigating overfitting and promoting faster convergence during training. The sparsity induced by dropout further aided in reducing communication overhead. All experiments were conducted on datasets built from scratch, with most simulations executed using Colab Pro and some on a computer equipped with an octa-core i7 processor and 32 GB RAM. For access to the Fed-RAC implementation, it is available on the GitHub repository at: *https://github.com/errahulm/Fed-RAC*.

VI. Performance Evaluation

A. Datasets and Models

This work uses four public datasets, including MNIST [16], HAR [17], CIFAR-10 [18], and SHL [19]. MNIST is a handwritten digit dataset containing 50000 images of different digits from $0 - 9$ for training. MNIST also has 10000 images for testing. HAR was collected using the smartphone (Samsung Galaxy S II) sensors, including a tri-axial accelerometer and gyroscope. CIFAR-10 comprises 60000 images of ten different classes. The dataset is balanced and correctly annotated with 6000 images for each class and contains 50000 images for training and 10000 for testing. SHL [19] dataset was collected from the onboard sensors of HUAWEI Mate 9 smartphones to recognize the locomotion modes of the users.

B. Baselines

We considered the existing techniques [9], [13], [14], [15] as baselines, noted as HeteroFL [9], FedAvg [13], Tifl [14] and Share [15], to evaluate and compare the performance. The details of considered baselines are discussed the Section II.

C. Evaluation Strategy

The primary motive of FL is to improve the local performance and generalization ability. We adopt these strategies:

1) *Local performance:* It determines: *how well the local model is trained on the dataset of the participants?*
2) *Cluster performance:* It estimates: *how well the participants can improve the cluster-wise performance through the aggregation of WPM?*
3) *Global performance:* It is the simple average over cluster performance and helps to determine: *how much deviation is observed in the cluster performance from the average value?*

D. Evaluation Metrics

We use the standard metrics, including **accuracy** and **F1-score**, to evaluate the performance of the Fed-RAC. We also introduce a new performance metric, namely **rounds-to-reach x%**. Let $I(x\%)$ denote the symbolic representation of the metric. $I(x\%)$ counts the number of iterations (or rounds) required for achieving the performance of $x\%$. We finally use the leave-one-out-test metric that trains the model for all class labels except for one randomly chosen class label.

E. Ablation Studies

1) Impact of Resource Aware Clustering: This experiment aims to assess the efficacy of resource-aware clustering. The resource vectors of the devices used in the experiment are shown in Table III. The resource vector comprises processing capacity, transmission rate, and memory, and is obtained from a survey conducted on 128 smartphone users, with prior

TABLE III
AVAILABLE RESOURCES SET $\mathcal{P}$ OF 40 PARTICIPANTS

$\mathcal{P}$	*R. Vector*	$\mathcal{P}$	*R. Vector*	$\mathcal{P}$	*R. Vector*	$\mathcal{P}$	*R. Vector*
p_1	$[1.6, 10.88, 8]$	p_{11}	$[1.6, 12.54, 6]$	p_{21}	$[1.6, 40, 1]$	p_{31}	$[3.1, 18.04, 6]$
p_2	$[2.8, 4.1, 3]$	p_{12}	$[0.8, 1.2, 6]$	p_{22}	$[1.1, 11.4, 6]$	p_{32}	$[2.5, 44.13, 6]$
p_3	$[1.1, 1.13, 6]$	p_{13}	$[1.3, 28.41, 6]$	p_{23}	$[2.5, 25, 6]$	p_{33}	$[2.3, 6.5, 6]$
p_4	$[1.6, 11.45, 3]$	p_{14}	$[1.3, 21.9, 3]$	p_{24}	$[2.2, 30, 4]$	p_{34}	$[2.1, 60.21, 6]$
p_5	$[3.2, 8.9, 3]$	p_{15}	$[3.1, 25.99, 6]$	p_{25}	$[1.6, 9.62, 6]$	p_{35}	$[2.1, 61.3, 8]$
p_6	$[2.2, 2, 4]$	p_{66}	$[3.2, 19.43, 4]$	p_{26}	$[2.2, 23.27, 6]$	p_{36}	$[3.2, 19, 6]$
p_7	$[3.1, 8.7, 1]$	p_{17}	$[1.0, 20.98, 3]$	p_{27}	$[1.5, 49.79, 6]$	p_{37}	$[2.7, 32.05, 6]$
p_8	$[1.8, 60, 3]$	p_{18}	$[1.6, 30, 3]$	p_{28}	$[1.7, 37.65, 6]$	p_{38}	$[2.9, 6.52, 6]$
p_9	$[2.7, 8.89, 3]$	p_{19}	$[1.0, 12, 2]$	p_{29}	$[3.1, 15.71, 6]$	p_{39}	$[0.8, 38.8, 6]$
p_{10}	$[1.4, 34.5, 8]$	p_{20}	$[2.7, 10, 6]$	p_{30}	$[2.6, 3, 6]$	p_{40}	$[2.1, 32, 6]$

RV = Resource vector = [processing, transmission rate, and memory].

permission obtained from the relevant authorities. From this survey, we randomly select 40 users to create different clusters using the Fed-RAC approach, as discussed in Section IV-A. Communication rounds are set to 200, and other parameters are described in Section V. The effectiveness of resource-aware clustering is evaluated using three types of resource vectors. The first type uses unnormalized resource vectors of the participants, whereas the second type uses normalized vectors with $\lambda_1 = \lambda_2 = \lambda_3 = 1/3$. The third type is similar to the second but with $\lambda_1 = 0.4, \lambda_2 = 0.4$, and $\lambda_3 = 0.2$. Similarly, other versions of λ_1, λ_2, and λ_3 are in Table IV.

Table IV presents the results of evaluating the impact of normalizing resource vectors on estimating the optimal number of clusters. The findings show that un-normalized vectors yield a limited number of clusters, namely 4 $(C_1 - C_4)$, using Dunn Indices. This is due to the dominance of the transmission rate resource over other resources, resulting in non-optimal clusters. By applying unit-based normalization, all resource values are scaled into the range of [0,1]. The normalized values generate an optimal number of clusters using Dunn Indices, as it removes resource bias. We obtained 6 clusters $(C_1 - C_6)$ by assigning equal contributions of all resources, i.e., λ_1 (processing capacity) $= \lambda_2$ (transmission rate) $= \lambda_3$ (memory) $= 1/3$. When we set the contribution based on the analysis given in [32], [33], $\lambda_1 = 0.4$, $\lambda_2 = 0.4$, and $\lambda_3 = 0.2$, we obtained 5 clusters $(C_1 - C_5)$.

Table IV presents the performance achieved by the Fed-RAC approach using different types of resource vectors on MNIST, HAR, CIFAR-10, and SHL datasets. The results show that normalizing the resource vector leads to improved performance compared to using unnormalized vectors. The normalization process is essential because when using unnormalized vectors, clustering relies on the dominating resource, leading to non-optimal clusters. These clusters may contain participants with non-identical resources that converge at irregular intervals, resulting in reduced cluster performance. Moreover, when the contributions of processing capacity (λ_1) and transmission rate (λ_2) are greater than memory (λ_3), i.e., $\lambda_1 = \lambda_2 = 0.4 > \lambda_3 = 0.2$, the cluster performance is high. An observation is made that an increase in memory contribution for cluster determination results in a decrease in the number of disjoint clusters, particularly when processing power and transmission rate parameters are low. In this scenario, participants with higher memory allocations tend to be assigned to higher clusters, contributing to a reduction in the overall cluster count. However, assigning smaller weights to processing and transmission may result in insufficient and sub-optimal training, leading to a decrease in performance.

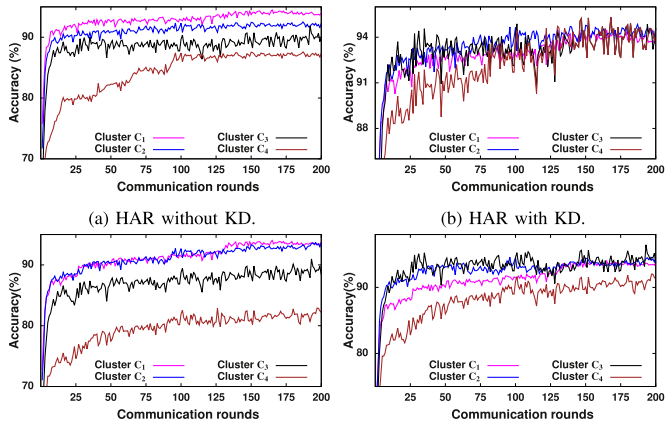


Fig. 2. Impact of leader-follower technique on the performance of models in different clusters using HAR and CIFAR-10 datasets.

2) Impact of Clusters Compaction: Table V illustrates the impact of cluster compaction on the performance of Fed-RAC using MNIST, HAR, CIFAR-10, and SHL datasets. Table V(a) demonstrates the cluster accuracy when all five clusters, estimated previously, are available. The results depicted that the follower clusters, $C_2 - C_5$, achieved comparable performance in contrast with C_1 (leader cluster). Moreover, cluster C_3 achieved a higher performance than C_1. This performance enhancement is due to the distillation of knowledge from leader to follower clusters during training. The details experiment on the impact of using knowledge distillation is elaborated in Section VI-E3. Apart from Table V(a), and (b) illustrates the performance of different clusters on considered datasets after compaction. The results showed a clear margin of improvement in the global and cluster-wise performance while using the cluster compaction in the Fed-RAC. This is due to the increment in the number of participants in each cluster.

3) Impact of Leader-Follower Technique: In this experiment, we aim to evaluate the performance improvement of the follower models assigned to each cluster (other than the leader cluster) using the leader-follower technique discussed in Section IV-C. We consider the four clusters, $C_1 - C_4$, obtained from the compaction in the previous result. The communication round is fixed at 200. However, to ensure brevity, we only present the results on HAR and CIFAR-10.

Fig. 2 illustrates the impact of the leader-follower technique on the performance of models in different follower clusters. Clusters $C_2 - C_4$ gain significant improvement in performance due to the distillation of knowledge from the leader model in C_1, as shown in Fig. 2(b) and (d). The results demonstrated that the improvement in the model's performance is significant at low resource clusters (C_4) and reduced gradually to C_2. It is because if the size of the cluster model is small then the logit difference between leader and follower is higher. Contrarily, if the difference between the size of the cluster model and the leader model is less, the logit difference is limited; thus, the performance gain

TABLE IV
IMPACT OF RESOURCE AWARE CLUSTERING ON CONSIDERED MNIST, HAR, CIFAR-10, AND SHL DATASETS USING DIFFERENT TYPES OF RESOURCE VECTORS

Types of resource vectors	Number of clusters	Datasets							
		MNIST		HAR		CIFAR-10		SHL	
		Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score
Unnormalized	$\mathcal{K}=4$	97.13 ± 0.30	98.06 ± 0.25	91.56 ± 0.45	92.47 ± 0.35	90.12 ± 0.60	90.83 ± 0.30	89.23 ± 0.30	90.41 ± 0.40
Normalized $\{\lambda_1=\lambda_2=\lambda_3=1/3\}$	$\mathcal{K}=6$	97.41 ± 0.25	98.19 ± 0.25	92.46 ± 0.40	93.38 ± 0.30	90.67 ± 0.45	91.41 ± 0.30	89.59 ± 0.25	90.83 ± 0.40
Normalized $\{\lambda_1=\lambda_2=0.4, \lambda_3=0.2\}$	$\mathcal{K}=5$	97.73 ± 0.40	98.37 ± 0.30	93.54 ± 0.50	94.26 ± 0.40	91.01 ± 0.30	92.13 ± 0.40	90.27 ± 0.30	91.19 ± 0.40
Normalized $\{\lambda_1=\lambda_2=0.2, \lambda_3=0.6\}$	$\mathcal{K}=4$	97.29 ± 0.50	98.21 ± 0.40	91.92 ± 0.50	92.61 ± 0.45	90.27 ± 0.60	91.41 ± 0.50	89.73 ± 0.40	90.75 ± 0.50
Normalized $\{\lambda_1=\lambda_2=0.3, \lambda_3=0.4\}$	$\mathcal{K}=4$	97.47 ± 0.50	98.15 ± 0.30	92.15 ± 0.40	93.07 ± 0.35	90.67 ± 0.30	91.87 ± 0.40	89.97 ± 0.40	91.03 ± 0.30

The reported results depict the global accuracy and F1 score achieved by the different models.

TABLE V
IMPACT OF CLUSTER COMPACTION ON THE ACCURACY OF THE FED-RAC USING DIFFERENT DATASETS (MNIST, HAR, CIFAR-10, AND SHL)

(a) Accuracy without compaction (cluster count = 5).

Clusters	Accuracy (in %) on datasets			
	MNIST	HAR	CIFAR-10	SHL
C_1	98.58 ± 0.50	93.31 ± 0.50	92.23 ± 0.30	91.43 ± 0.25
C_2	98.18 ± 0.60	93.68 ± 0.40	91.55 ± 0.25	90.17 ± 0.30
C_3	$\mathbf{98.14 \pm 0.30}$	$\mathbf{94.19 \pm 0.20}$	91.23 ± 0.40	91.76 ± 0.30
C_4	97.04 ± 0.50	93.55 ± 0.50	90.58 ± 0.25	89.88 ± 0.40
C_5	96.71 ± 0.60	93.01 ± 0.60	89.45 ± 0.60	87.82 ± 0.40
Global	$\mathbf{97.73 \pm 0.40}$	$\mathbf{93.54 \pm 0.30}$	$\mathbf{91.01 \pm 0.45}$	$\mathbf{90.27 \pm 0.30}$

(b) Accuracy with compaction (clusters count = 4).

Clusters	Accuracy (in %) on datasets			
	MNIST	HAR	CIFAR-10	SHL
C_1	98.76 ± 0.30	93.47 ± 0.60	92.41 ± 0.40	91.52 ± 0.30
C_2	98.73 ± 0.50	93.76 ± 0.50	92.37 ± 0.25	92.53 ± 0.40
C_3	98.78 ± 0.20	94.92 ± 0.40	92.69 ± 0.35	92.02 ± 0.40
C_4	98.63 ± 0.60	94.17 ± 0.50	91.73 ± 0.30	91.26 ± 0.35
Global	$\mathbf{98.72 \pm 0.25}$	$\mathbf{94.08 \pm 0.35}$	$\mathbf{92.30 \pm 0.30}$	$\mathbf{91.83 \pm 0.30}$

(c) Accuracy with compaction (cluster count = 3).

Clusters	Accuracy (in %) on datasets			
	MNIST	HAR	CIFAR-10	SHL
C_1	98.37 ± 0.45	93.11 ± 0.40	91.81 ± 0.30	91.22 ± 0.40
C_2	97.49 ± 0.35	92.26 ± 0.25	90.36 ± 0.35	89.77 ± 0.40
C_3	95.42 ± 0.40	89.47 ± 0.35	87.41 ± 0.50	86.82 ± 0.30
Global	$\mathbf{97.09 \pm 0.40}$	$\mathbf{91.62 \pm 0.30}$	$\mathbf{89.90 \pm 0.40}$	$\mathbf{89.27 \pm 0.35}$

is low. Cluster C_4 gains accuracy of $\approx 8\%$ for HAR and $\approx 11\%$ for CIFAR-10 datasets, whereas the performance gain for C_2 is $\approx 2\%$ for both datasets. Furthermore, in FL-based training, we considered participants with heterogeneous resources; thus, participants with the highest and lowest resources, respectively, achieved colossal and most minor performances. It also creates a significant difference between the performance of the models in the largest and smallest clusters, which aggregately results in performance compromise despite clustering. Therefore, KD is incorporated to enhance the performance of models in the smaller clusters.

F. Sensitivity Analysis

This experiment aimed to investigate how the learning rate affects the performance of Fed-RAC. MNIST, HAR, CIFAR-10, and SHL datasets were used, and the communication rounds were set to 5, 10, 20, and 20, respectively. The rounds were restricted as the approach converged at any learning rate at higher communication rounds.

Table VI provides a comprehensive overview of how varying learning rates impact the model accuracy within the leader cluster of Fed-RAC across MNIST, HAR, CIFAR-10, and SHL datasets. The observed outcomes underscore the significance of selecting an appropriate learning rate for optimal model performance. Notably, employing a smaller learning rate (e.g., 0.002) yielded favourable results for Fed-RAC across all datasets. The lowest accuracy was recorded for the learning rate of 0.010 due to faster convergence, which led to sub-optimal model performance. Particularly, the MNIST dataset demonstrated accelerated convergence in Fed-RAC, achieving accuracy beyond 90% across all datasets after merely 5 communication rounds. While the accuracy trend for the Fed-RAC approach generally followed a linear pattern across different datasets, a nuanced observation revealed plateaued behaviour for learning rates between 0.006 to 0.008. This plateau suggests a delicate balance in selecting the learning rate to ensure optimal convergence without compromising performance. Furthermore, the substantial difference of over 8% in cluster accuracy between the learning rates of 0.002 and 0.010 underscores the critical importance of judiciously choosing the learning rate during training. This disparity highlights the direct impact of the learning rate on the Fed-RAC approach's efficacy and emphasizes the need for careful consideration in achieving the desired balance between convergence speed and model accuracy.

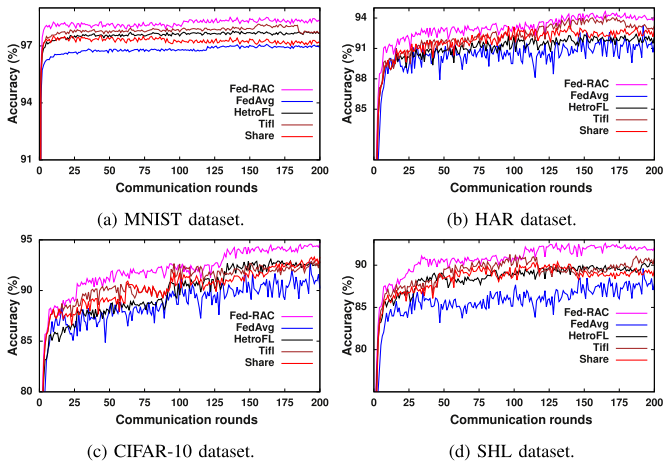
TABLE VI
IMPACT OF LEARNING RATE ON THE ACCURACY OF THE MODEL IN THE LEADER CLUSTER

Datasets	CR	Accuracy (in %) on learning rates				
		0.002	0.004	0.006	0.008	0.010
MNIST	5	98.07	96.44	93.32	92.97	90.37
HAR	10	89.93	87.24	86.05	83.75	79.24
CIFAR-10	20	84.41	83.73	81.29	80.73	77.12
SHL	20	82.14	80.71	79.64	79.32	74.23

CR=Communication rounds.

G. Performance Comparison

1) Impact of Communication Rounds: This experiment investigates the impact of different datasets on the convergence of the Fed-RAC and considered baselines. All 40 participants were involved in the FL operation, and thus FedAvg and Tifl utilized the smallest follower model to ensure deployment and training on all participants. The communication rounds for Fed-RAC were determined as the rounds required for the convergence of the leader model plus the maximum rounds required for the convergence of the slowest follower model.



(a) MNIST dataset. (b) HAR dataset.

(c) CIFAR-10 dataset. (d) SHL dataset.

Fig. 3. Illustration of impact of datasets on the convergence rounds of Fed-RAC, FedAvg, HeteroFL, Tifl, and Shape.

TABLE VII
ILLUSTRATION OF IMPACT OF ROUNDS-TO-REACH $x\%$ GLOBAL ACCURACY ON CONSIDERED DATASETS

Dataset	x%	Fed-RAC (proposed)										A1	A2	A3	A4
		Cluster w/o KD					Cluster w KD								
		C_1	C_2	C_3	C_4	TRR	C_1	C_2	C_3	C_4	TRR				
MNIST	96	2	2	5	9	11	2	2	3	5	**7**	9	11	8	8
HAR	92	36	47	-	-	83	36	17	29	41	**77**	92	102	79	82
CIFAR-10	88	51	59	-	-	110	51	23	37	53	**104**	112	121	108	117
SHL	86	67	74	-	-	141	67	34	39	61	**128**	137	146	139	141

TRR=Total Required Rounds= rounds(C_1)+max rounds {C_2, C_3, C_4}, A1= FedAvg, A2=HetroFL, A3=Tifl, and A4=Shape.

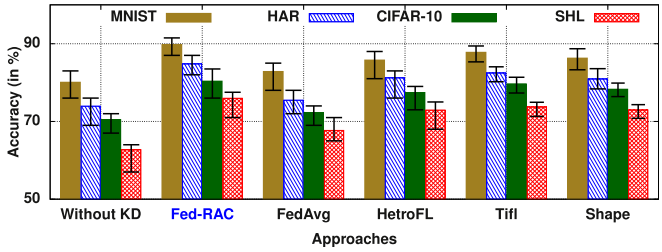


Fig. 4. Impact of leave-one-out test metric on MNIST, HAR, CIFAR-10, and SHL datasets using Fed-RAC (without KD), Fed-RAC (with KD), FedAvg, HeteroFL, Tifl, and Shape approaches.

Fig. 3 shows the impact of the considered datasets, namely MNIST, HAR, CIFAR-10, and SHL, on the convergence of Fed-RAC, FedAvg, HeteroFL, Tifl, and Shape. The learning curve depicted in the figure displays a classic shape with a two-step behaviour. Initially, the performance improves steeply until it reaches a plateau value after some communication rounds. Then, the accuracy increases with more communication rounds. Fed-RAC outperforms the existing approaches on all communication rounds during the experiment. The participants in the leader cluster (C_1) quickly converge due to sufficient resources to train a large-size model. The Fed-RAC approach also incorporates KD to train the models at the participants, leading to well-behaved optimization steps compared to non-KD-based training and reduced communication rounds. On the MNIST dataset, all approaches achieved convergence at lower rounds with marginal improvement afterwards, as shown in Fig. 3. This is due to the balanced and sufficient number of instances for all classes in MNIST. FedAvg achieved slower convergence with minimal accuracy due to incompetence in handling heterogeneity among the participants and using a small-size model to accommodate all 40 participants during training. HeteroFL and Tifl achieved comparable performance to Fed-RAC due to the strategy of addressing heterogeneity.

2) Impact of Rounds-to-Reach X%: The objective of this experiment is to investigate the effectiveness of the proposed Fed-RAC in achieving a global accuracy of $x\%$ within a certain number of communication rounds. To achieve this, we have set the value of x to be 96, 92, 88, and 85 for MNIST, HAR, CIFAR-10, and SHL datasets, respectively, taking into account the convergence rates of these datasets. Fed-RAC involves training the model in the leader cluster followed by parallel training of models in the follower clusters. As such, we define the Total Required Rounds (TRR) for complete training as the sum of rounds required to train the model in the leader cluster (C_1) and the maximum rounds required to train the model in any of the follower clusters ($\max rounds\{C_2, C_3, C_4\}$).

Table VII presents the results of the rounds-to-reach $x\%$ metric on the datasets and illustrates the impact of this metric on the Fed-RAC. The results indicate that the Fed-RAC approach (cluster-wise with KD) outperforms the baseline approaches, including cluster-wise without KD. This can be attributed to two main reasons. First, the participants in the leader cluster (C_1) have sufficient resources to train large models, which leads to quicker convergence. Second, the Fed-RAC approach incorporates KD to train the models at the participants, resulting in well-behaved optimization steps compared to non-KD. Regarding the convergence of *cluster-wise without KD*, the results are not reported for models in clusters C_3 and C_4 on HAR, CIFAR-10, and SHL datasets. This is because, in the absence of KD, the participants in clusters C_3 and C_4 are unable to achieve the desired $x\%$ accuracy within the cap of 200 communication rounds. Further, we used small models in FedAvg, to involve 40 participants. Although the use of KD appears to incur higher computational costs compared to the baselines that do not incorporate KD, Fed-RAC achieves the desired performance in fewer communication rounds, thus reducing the computational cost.

3) Leave-One-Out: The objective of this experiment was to assess the overall performance of Fed-RAC and several baseline approaches in a scenario where instances of a randomly selected class label were not included in the training but appeared in the testing. The class label with the highest number of instances was selected as the leave-out class during the experiment. The communication rounds were set to 200, and the parameters and local epochs were determined according to the implementation details discussed in Section V.

In Fig. 4, the impact of removing instances of one class label from the training of all participants in FL is demonstrated. The results show that Fed-RAC outperforms the existing approaches, which is consistent with the performance pattern observed in previous results. The approach that does not use KD clustering (referred to as the "without KD clustering approach") achieved

TABLE VIII
ILLUSTRATION OF TRAINING DURATION OF DIFFERENT APPROACHES ON THE SEQUENTIAL MACHINE AND HAR DATASET WITH AVAILABLE 40 PARTICIPANTS

Approach	Training Duration (in mins.)	Performance (accuracy)	Participants in Training
Fed-RAC	434	93.54%	40
FedAvg	393	89.86%	32
HertoFL	456	90.76%	36
Tifl	532	91.41%	40
Shape	405	90.93%	34

the lowest performance, likely due to the small-size models trained on follower clusters with a limited number of participants in each cluster. This negatively impacted the overall performance of the approach. The MNIST dataset achieved the highest performance due to the large number of instances for classes other than the excluded one. Conversely, the SHL dataset had the lowest performance due to the excluded class having the highest number of instances.

4) Training Duration: In this section, we analyze the training time duration on a sequential machine, and the results are presented in Table VIII. The findings indicate that the FedAvg approach exhibits the shortest training time, while Tifl requires the most time due to its two-level training and weight exchange among participants. Notably, our proposed approach, Fed-RAC, requires more training time than FedAvg, yet it ensures the participation of all participants, akin to Tifl.

VII. DISCUSSION AND FUTURE WORK

In this section, issues are discussed that need to be addressed in future work in conjunction with the proposed approach. The approach uses a leader-follower technique where logits and features from the leader cluster model are sent to the remaining clusters. However, this could potentially expose private training data or enable participants to reconstruct models. To address these privacy concerns, future work on incorporating security aspects is necessary. Furthermore, while Fed-RAC considers participant heterogeneity, it does not account for noise in data instances and labels. Therefore, future work will involve incorporating such noise into the model training process. Further, we acknowledge the potential significance of fine-tuning this parameter. We are actively exploring further experiments to identify an optimal α value that could enhance the performance of Fed-RAC in our future investigations.

VIII. CONCLUSION

In this paper, a federated learning approach called Fed-RAC is proposed to address the negative impact of heterogeneous participants. Unlike previous studies, Fed-RAC trains local models on all participants despite differences in heterogeneity and training time. The approach first identifies the optimal number of clusters based on available devices and networking resources, then generates and randomly initializes a model that is used for compression to obtain models for all clusters. A participant assignment mechanism and a leader-follower technique are introduced to improve the performance of lightweight models using knowledge distillation. Experimental evaluation is conducted to verify the approach's effectiveness on existing datasets, leading to several key findings: successful federated learning requires proper management of participant heterogeneity, resource-aware clustering helps identify the optimal number of clusters, the number of data instances significantly affects cluster performance, and the leader-follower technique enhances performance based on model size.

REFERENCES

[1] M. Duan et al., "Flexible clustered federated learning for client-level data distribution shift," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 11, pp. 2661–2674, Nov. 2022.

[2] J. S. Ng et al., "Reputation-aware hedonic coalition formation for efficient serverless hierarchical federated learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 11, pp. 2675–2686, Nov. 2022.

[3] X. Ma, C. Wen, and T. Wen, "An asynchronous and real-time update paradigm of federated learning for fault diagnosis," *IEEE Trans. Ind. Informat.*, vol. 17, no. 12, pp. 8531–8540, Dec. 2021.

[4] Z. Wang et al., "Asynchronous federated learning over wireless communication networks," *IEEE Trans. Wireless Commun.*, vol. 21, no. 9, pp. 6961–6978, Sep. 2022.

[5] X. Ouyang, Z. Xie, J. Zhou, J. Huang, and G. Xing, "ClusterFL: A similarity-aware federated learning system for human activity recognition," in *Proc. 19th Annu. Int. Conf. Mobile Syst., Appl., Serv.*, 2021, pp. 54–66.

[6] C. Briggs, Z. Fan, and P. Andras, "Federated learning with hierarchical clustering of local updates to improve training on non-IID data," in *Proc. Proc. Int. Joint Conf. Neural Netw.*, 2020, pp. 1–9.

[7] L. U. Khan, Z. Han, D. Niyato, and C. S. Hong, "Socially-aware-clustering-enabled federated learning for edge networks," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 3, pp. 2641–2658, Sep. 2021.

[8] S. Horvath, S. Laskaridis, M. Almeida, I. Leontiadis, S. Venieris, and N. Lane, "FJORD: Fair and accurate federated learning under heterogeneous targets with ordered dropout," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 12876–12889.

[9] E. Diao, J. Ding, and V. Tarokh, "HeteroFL: Computation and communication efficient federated learning for heterogeneous clients," 2020, *arXiv: 2010.01264*.

[10] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.

[11] H. Zhao, X. Sun, J. Dong, C. Chen, and Z. Dong, "Highlight every step: Knowledge distillation via collaborative teaching," *IEEE Trans. Cybern.*, vol. 52, no. 4, pp. 2070–2081, Apr. 2022, doi: 10.1109/TCYB.2020.3007506.

[12] R. Xu, J. Xu, and D. C. Wunsch, "A comparison study of validity indices on swarm-intelligence-based clustering," *IEEE Trans. Syst., Man, Cybern., Part B. (Cybernetics)*, vol. 42, no. 4, pp. 1243–1256, Aug. 2012.

[13] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[14] Z. Chai et al., "TIFL: A tier-based federated learning system," in *Proc. 29th Int. Symp. High-Perform. Parallel Distrib. Comput.*, 2020, pp. 125–136.

[15] Y. Deng et al., "SHARE: Shaping data distribution at edge for communication-efficient hierarchical federated learning," in *Proc. IEEE 41st Int. Conf. Distrib. Comput. Syst.*, 2021, pp. 24–34.

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[17] D. Anguita, A. Ghio, L. Oneto, X. Parra Perez, and J. L.R. Ortiz, "A public domain dataset for human activity recognition using smartphones," in *Proc. Eur. Symp. Artif. Neural Netw.*, 2013, pp. 437–442.

[18] A. Krizhevsky et al., "Learning multiple layers of features from tiny images," 2009. [Online]. Available: https://www.cs.toronto.edu/kriz/learning-features-2009-TR.Pdf

[19] SHL Challenge, 2023. [Online]. Available: http://www.shl-dataset.org/activity-recognition-challenge/

[20] Y. Deng et al., "TailorFL: Dual-personalized federated learning under system and data heterogeneity," in *Proc. 20th ACM Conf. Embedded Netw. Sensor Syst.*, 2022, pp. 592–606.
[21] A. M. Abdelmoniem and M. Canini, "Towards mitigating device heterogeneity in federated learning via adaptive model quantization," in *Proc. 1st Workshop Mach. Learn. Syst.*, 2021, pp. 96–103.
[22] S. Alam, L. Liu, M. Yan, and M. Zhang, "FedRolex: Model-heterogeneous federated learning with rolling sub-model extraction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 29677–29690.
[23] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," 2019, *arXiv: 1903.03934*.
[24] Y. Chen, X. Sun, and Y. Jin, "Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 10, pp. 4229–4238, Oct. 2020.
[25] Q. Ma, Y. Xu, H. Xu, Z. Jiang, L. Huang, and H. Huang, "FedSA: A semi-asynchronous federated learning mechanism in heterogeneous edge computing," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3654–3672, Dec. 2021.
[26] H. Jin et al., "Personalized edge intelligence via federated self-knowledge distillation," *IEEE Trans. Parallel Distrib. Syst.*, vol. 34, no. 2, pp. 567–580, Feb. 2023.
[27] D. Li and J. Wang, "Fedmd: Heterogenous federated learning via model distillation," 2019, *arXiv:1910.03581*.
[28] T. Lin, L. Kong, S. U. Stich, and M. Jaggi, "Ensemble distillation for robust model fusion in federated learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 2351–2363.
[29] C. He, M. Annavaram, and S. Avestimehr, "Group knowledge transfer: Federated learning of large CNNs at the edge," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 14068–14080.
[30] S. Wang et al., "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, Jun. 2019.
[31] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 7611–7623.
[32] S. Jin, S. Di, X. Liang, J. Tian, D. Tao, and F. Cappello, "DeepSZ: A novel framework to compress deep neural networks by using error-bounded lossy compression," in *Proc. 28th Int. Symp. High-Perform. Parallel Distrib. Comput.*, 2019, pp. 159–170.
[33] S. Yao et al., "FastDeepIoT: Towards understanding and optimizing neural network execution time on mobile and embedded devices," in *Proc. 16th ACM Conf. Embedded Netw. Sensor Syst.*, 2018, pp. 278–291.
[34] J. C. Bezdek and N. R. Pal, "Some new indexes of cluster validity," *IEEE Trans. Syst., Man, Cybern. Part B (Cybernetics)*, vol. 28, no. 3, pp. 301–315, Jun. 1998.
[35] G. H. Shah, "An improved DBSCAN, a density based clustering algorithm with parameter selection for high dimensional data sets," in *Proc. Nirma Univ. Int. Conf. Eng.*, 2012, pp. 1–6.
[36] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FEDAVG on non-IID data," in *Proc. Int. Conf. Learn. Representations*, 2019, pp. 1–26.
[37] "Supplementary material: FedRAC," 2023. [Online]. Available: https://sites.google.com/view/rahulmishracse/fedrac

Rahul Mishra (Member, IEEE) received the PhD degree in computer science and engineering, Indian Institute of Technology (BHU) Varanasi, India, in 2022. He is currently working as an assistant professor Indian Institute of Technology, Patna, India. His research interests include wireless sensor networks, deep learning, fog computing, smart sensing, and federated learning.

Hari Prabhat Gupta (Senior Member, IEEE) received the PhD degree in computer science and engineering from the Indian Institute of Technology Guwahati, India, in 2014. He is currently working as an associate professor in the Department of Computer Science and Engineering, Indian Institute of Technology (BHU), Varanasi, India. His research interests include wireless sensor networks, distributed algorithms, and IoT.

Garvit Banga received the BTech degree in metallurgical engineering from the Indian Institute of Technology (BHU) Varanasi, Varanasi, India, in 2021. His research interests include machine learning, deep learning, and ubiquitous computing.

Sajal K. Das (Fellow, IEEE) is currently a professor of computer science and Daniel St. Clair Endowed Chair with the Missouri University of Science and Technology. His research interests include wireless sensor networks, mobile and pervasive computing, cyber-physical systems, and IoTs, smart environments, cloud and fog computing, cyber security, and social networks. He serves as a founding editor-in-chief of elsevier's pervasive and Mobile Computing Journal, and an associate editor of several journals including *IEEE Transactions of Mobile Computing*, *IEEE Transactions on Dependable and Secure Computing*, and *ACM Transactions on Sensor Networks*.