On the Tradeoff Between Heterogeneity and Communication Complexity in Federated Learning

Priyanka Sinha, Jacek Kibilda, and Walid Saad

***Abstract*—The heterogeneity-complexity tradeoff in federated learning (FL) is a key challenge facing the deployment of FL in the real world. While learning on heterogeneous data is key to required performance improvement in many applications, training with state-of-the-art uncoordinated FL methods in the presence of substantial heterogeneity can lead to prohibitively large communication complexity. Most state-of-the-art techniques designed to deal with heterogeneity employ sampling across the client pool, inducing low aggregation or coverage in the datasets of the out-of-sample clients. In this paper, we challenge the common wisdom of learning a single global model across multi-distribution data and propose a federated learning mechanism (without money) that partitions the pool of clients into a minimal number of subsets of clients and then learns a global model for each subset of the clients. This leads to a reduced degree of heterogeneity across each subset of clients while ensuring positive aggregation across all client datasets. In particular, we pose the problem of federated learning as a repeated multi-agent basic utility game and use no-regret algorithms find approximate solutions. Experiments on multi-distribution datasets show that the proposed method outperforms the key benchmarks in terms of both communication complexity and global accuracy.**

***Index Terms*—Fairness, Multi-distribution learning, heterogeneity, federated clustering, communication complexity, aggregation, coverage, no-regret learning, mechanism without money.**

I. Introduction

Due to its effectiveness in addressing challenges like data security, privacy, and access to heterogeneous data, federated learning has been increasingly popular in recent years. Examples of typical applications include smart manufacturing, digital health, and vehicular communications, etc. One of the main obstacles in the adoption of federated learning is the long training time. In many federated learning scenarios the clients are only online for a short time, therefore having long training time or high communication complexity leads to higher training errors. At the same time long training times are unavoidable when training on multiple distributions. Thus there is still a need for a more efficient training paradigm that can learn from the history of the past rounds to adjust its current behavior.

The most prominent method of dealing with heterogeneity-communication trade-offs is known as client sampling. While this line of work has been extensively explored by researchers, these methods still have the major drawback of being inherently unfair in terms of aggregation across the clients. The outcome of the sampling methods differ based on the author's preference over different objectives. For example, methods such as [1], [2] favor convergence speed and communication complexity at the cost of aggregation, while the papers [3] favor aggregation more. A more comprehensive list of sampling techniques can be found in [4]. Different from the sampling techniques, our work proposes periodic clustering across the client pool. Recently there have been some other works [5]–[7] in the literature that also consider clustering for federated learning. Although these papers consider dividing the data into different clusters as a prepossessing step, our work considers an online clustering technique that keeps assigning clients to the correct clusters as they keep joining in.

A more comprehensive approach to solving multiple trade-offs in the FL process is considered in the FL-Mechanism literature. This literature can be broadly categorized into two classes: mechanisms with money and mechanisms without money. In the former category, the clients are paid monetary rewards for joining the FL training and in the second category only intrinsic motivations such as global accuracy, local accuracy, statistical utility, system utility, etc. are considered. While most FL mechanisms such as [8] and [9] involve either full or partial monetary rewards to the clients, our proposed mechanism is completely intrinsically motivated. A more comprehensive description of the FL mechanisms can be found in [10].

Finally, we summarize the main contributions of the proposed work in this section. i) while the state-of-the-art techniques for mitigating the heterogeneity-complexity trade-off by sampling a subset of clients at each round and learning a single averaged model across the entire pool of clients. This creates a trade-off between the desired coverage gains of the clients and the achievable communication complexity of the algorithm. In our work, we replace the sampling step with a periodic clustering step, that allows us to learn a different averaged model for each cluster and therefore improve the heterogeneity-complexity trade-off without hurting the coverage gains of certain clients in the pool. ii) The proposed clustering method is an online clustering therefore any newcomer client can be readily assigned a cluster and no pre-processing step is required. Overall the larger goal of our current and future work on this topic is to identify a suitable solution concept for the FL framework, that can help us build a mechanism that can address the trade-off of interest more predictably. While in

This research was supported by the US National Science Foundation under Grant CNS- 2114267.

this paper, we focus on the aggregation-complexity trade-off, a well-suited solution concept can be extended to handle other related trade-offs between competing objectives such as fairness and robustness with a minimum shift of perspective.

Notation: For a natural number $N \in \mathbb{N}$, $[N]$ denotes the set of all natural numbers from 1 to N, i.e. $[N] = \{1, 2, 3, ..., N\}$. For all $i \in [N]$, $\boldsymbol{\theta}_i \in \mathbb{R}^p$ denotes a $p-$dimensional vector, and $\boldsymbol{\theta} = \{\boldsymbol{\theta}_i \in \mathbb{R}^p\}_{i\in[N]}$ denotes a collection of N, $p-$dimensional vectors $\boldsymbol{\theta}_i$.

II. Setup

A. Federated Learning in Heterogeneous Setting

In federated learning, a large number of clients (let N be the number of the clients), each with heterogeneous datasets $\mathcal{D}_i \sim \mathcal{P}_i$ collaborate to learn a single model that will achieve high training and testing accuracy across all datasets. These clients are heterogeneous in the sense that the data samples $(X_m, Y_m)_{m\in[N_i]} \in \mathcal{D}_i$ in their respective datasets follow different distributions $\mathcal{P}_i$ on the dataspace $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are the input and output domains of the task of interest. Thus each client has access to a different expected loss function $l_i : \mathbb{R}^p \to \mathbb{R}$, where $l_i(\boldsymbol{\theta}_i) = \mathbb{E}_{(X,Y)\sim\mathcal{P}_i}\Big[l(X, Y, \boldsymbol{\theta}_i)\Big]$ parameterized its local model weights $\boldsymbol{\theta}_i$. Here $l : \mathcal{X} \times \mathcal{Y} \times \mathbb{R}^p \to \mathbb{R}$ is the per-sample common task loss. The clients are interested in minimizing the global loss function $L : \mathbb{R}^p \to \mathbb{R}$, where $L(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i\in[N]} l_i(\boldsymbol{\theta})$, and finding a single global model, $\boldsymbol{\theta}_a^* \in \mathbb{R}^p$ that satisfies the accuracy requirements for all clients in $[N]$:

$$\begin{cases} \boldsymbol{\theta}_a^* \in \arg\min\limits_{\boldsymbol{\theta}\in\mathbb{R}^p} L \\ l_i(\boldsymbol{\theta}_a^*) \leq \epsilon^* \ , \forall i \in [N] \ . \end{cases} \tag{1}$$

Due to privacy concerns, the optimization task in (1) is decomposed into a minimization over the model parameters space, $\mathbb{R}^p$ at the client side and an optimization over the probability simplex Δ^{N-1} at the server side:

$$\textbf{At each client:} \quad \underset{\boldsymbol{\theta}_i^k\in\mathbb{R}^p}{\text{minimize}}\, l_i(\boldsymbol{\theta}_i^k) \ ,$$

$$\textbf{At the server:} \quad \underset{p^k\in\Delta^{N-1}}{\text{minimize}} \bigoplus_{i\in[N]} l_i(\sum_{i=1}^{N} p_i^k \boldsymbol{\theta}_i^k) \ . \tag{2}$$

At each round $k \in \mathbb{N}$, the clients present a set of vectors $\{\boldsymbol{\theta}_i{}^k \in \text{dom}(l_i)\}_{i\in[N]}$ and the server is required to choose a probability vector $p^k \in \Delta^{N-1}$ such that the combined model, $\boldsymbol{\theta}_a^k = \sum_{i=1}^N p_i{}^k \boldsymbol{\theta}_i^k$ satisfies (1).

However, with a high degree of heterogeneity among the clients the above setting is known to increase the communication complexity, i.e. it requires a very large number of the model combining steps at the server before converging to an acceptable loss value, or the loss never drops below the desired threshold (in this case the communication complexity is almost infinity). To improve the communication complexity, the state-of-the-art methods rely on sampling the most useful clients from the pool, based on various metrics of client utility. This in turn introduces aggregation or coverage deficiency and gives rise to fairness issues [11] in federated learning. In this paper, we improve the achievable complexity-coverage tradeoff by employing an online clustering method that partitions the client pool into subsets of clients with a smaller degree of intra-cluster heterogeneity. In the following subsections, we provide a more precise characterization of the communication complexity and coverage gain in federated learning.

B. Characterization of Communication Complexity in FL

In the literature [3] the convergence rate of an arbitrary real-valued function $g : \mathbb{R}^p \to \mathbb{R}$ under any iterative method $\boldsymbol{\theta}_{k+1} = \Phi(\boldsymbol{\theta}_k)$ is defined as an upper bound on the difference between the value of the function at round-k, $g(\boldsymbol{\theta}_k)$ and the desired optimal value ϵ_g: $g(\boldsymbol{\theta}_k) - \epsilon_g \leq \mathcal{O}(f(k))$. Similarly, if the desired communication complexity in federated learning is $\mathcal{O}(f(k))$ then we need to ensure the following upper bound on the global loss:

$$\begin{cases} \frac{1}{N}\sum_{i\in[N]} l_i(\boldsymbol{\theta}_a^*) \leq \epsilon^* \ , \\ \frac{1}{N}\sum_{i\in[N]} \big[l_i(\boldsymbol{\theta}_a^k) - l_i(\boldsymbol{\theta}_a^*)\big] \leq \mathcal{O}(f(k)) \ . \end{cases} \tag{3}$$

C. Characterization of Coverage Gain in FL

The coverage gain of a client participating in federated learning is measured by a net decrease in the loss incurred by the client on the local and non-local datasets. Let the optimal local model parameter for a client-$w \in [N]$ be $\boldsymbol{\theta}_w^*$, and let the FL global model under the choice of combining vector, $p \in \Delta^{N-1}$, be $\boldsymbol{\theta}_a$. Then we define the local gain of client-w, $g_w^l(\boldsymbol{\theta}_a; p)$ and the non-local gain local gain of client-w, $g_w^n(\boldsymbol{\theta}_a; p)$ as the relative changes in the loss incurred induced by the combination vector, p, respectively on the datasets $\mathcal{D}_w$ and $\mathcal{D}_i, \forall i \neq w$:

$$\begin{cases} g_w^l(\boldsymbol{\theta}_a; p) = l_w(\boldsymbol{\theta}_w^*) - l_w(\boldsymbol{\theta}_a; p) \\ g_w^n(\boldsymbol{\theta}_a, p) = \sum_{i\in[N]:i\neq w} \big[l_i(\boldsymbol{\theta}_w^*) - l_i(\boldsymbol{\theta}_a, p)\big] \ . \end{cases} \tag{4}$$

Thus the net gain or utility $u_w(\boldsymbol{\theta}_a; p) = g_w^l(\boldsymbol{\theta}_a, p) + g_w^n(\boldsymbol{\theta}_a, p)$ that the client-w derives from the model combination induced by the vector p, is defined as below:

$$u_w(\boldsymbol{\theta}_a, p) = \sum_{i\in[N]} \big[l_i(\boldsymbol{\theta}_w^*) - l_i(\boldsymbol{\theta}_a, p)\big] \tag{5}$$

In order to maximize the total coverage we require that all clients in $[N]$ derive positive utility from the FL process

$$\inf_{w\in[N]} \sum_{i\in[N]} \big[l_i(\boldsymbol{\theta}_w^*) - l_i(\boldsymbol{\theta}_a)\big] \geq \delta \in \mathbb{R}^{++} \ . \tag{6}$$

Combining (6) and (3) the complete optimization problem characterizing the complexity-coverage tradeoff can be given as below:

$$\left\{\begin{array}{l} \min_{p^k\in\Delta^{N-1}} \sum_{i\in[N]} \left[l_i(\boldsymbol{\theta}_a^k;p^k)-l_i(\boldsymbol{\theta}_a^*)\right]- \\ \inf_{w\in[N]}\sum_{i\in[N]}\left[l_i(\boldsymbol{\theta}_w^*)-l_i(\boldsymbol{\theta}_a^k)\right] \text{ where} \\ \frac{1}{N}\sum_{i\in[N]} l_i(\boldsymbol{\theta}_a^*)\le \epsilon^* \ . \end{array}\right. \tag{7}$$

Recognizing the necessity of multiple global models and multiple combination vectors in improving the complexity-coverage tradeoff, we formulate the problem in a way such that the history of the past FL rounds can be used to assess pairwise compatibility of the clients in the pool, which then dictates the choice of the model combining vector. We identify that no-regret learning in a multi-player (each client is a player with its own utility function and action set) repeated game framework lets us capture the desired properties of the federated learning process. In the next section we describe our alternative framework.

III. Federated Learning as a Repeated Convex Game:

In order to address the coverage-complexity trade-off in the heterogeneous setting we pose the problem of federated learning as that of no-regret learning in a repeated multi-agent convex game [12]–[14]. Similar to the communication or model averaging step in the FL process, a repeated game is played in a sequence of rounds. The game is described as the tuple $\left[i\in[N],\boldsymbol{\Lambda}_i,u_i(\boldsymbol{\Lambda},\boldsymbol{\lambda})\right]$ where each client is a player with an action set $\boldsymbol{\Lambda}_i\in\Delta^{N-1}$, and utility function $u_i:\bigotimes_{i\in[N]}\boldsymbol{\Lambda}_i\to\mathbb{R}$ described as below

$$u_i(\boldsymbol{\Lambda},\boldsymbol{\lambda})=\sum_{i\in[N]}\left[\boldsymbol{\lambda}_i l_i\left(\sum_{j\in[N]}\boldsymbol{\Lambda}_{i,j}\theta_j\right)-l_j(\theta_i^*)\right] \ , \tag{8}$$

where $\boldsymbol{\Lambda}_{i,j}$ represents the preference of client-i to combine models with client-j, and $\boldsymbol{\Lambda}_{i,j}$ is a mixed strategy over the set of the client model $\boldsymbol{\theta}_j,\forall j\in[N]$. $u_i(\boldsymbol{\Lambda},\boldsymbol{\lambda})$ is the same utility function defined in (5) with $p\in\Delta^{N-1}$, replaced by a stochastic matrix $\boldsymbol{\Lambda}\in\Delta^{N-1}\times\Delta^{N-1}$ and a probability vector $\boldsymbol{\lambda}\in\Delta^{N-1}$ and $\boldsymbol{\lambda}$ is a distribution over the set of convex loss functions $L=\{l_i:\mathbb{R}^p\to\mathbb{R},\forall i\in[N]\}$ of all the clients. Upon receiving the set of local model parameters θ_i^k from all the clients the server first computes or updates the stochastic matrix $\boldsymbol{\Lambda}$ associated with preference vectors $\boldsymbol{\Lambda}_i$, and then computes the final model combination vector $\boldsymbol{\lambda}^k\in\Delta^{N-1}$, that produces the averaged model θ_a^k as below:

$$\begin{aligned} \theta_a^k &= \sum_{i\in[N]}\boldsymbol{\lambda}^k\theta_i^k, \\ \boldsymbol{\lambda}^k &= \boldsymbol{\lambda}^k\boldsymbol{\Lambda}^k \ . \end{aligned} \tag{9}$$

The above game when solved leads to a correlated equilibrium, however, due to computational intractability we approximately solve the game by employing the regret matching methods in III-A and III-B. The goal of the clients and the server is to learn a sequence of $\{\boldsymbol{\lambda}^k\}_{k\in[K]}$ and $\{\boldsymbol{\Lambda}^k\}_{k\in[K]}$ that will minimize the cumulative global loss below:

$$\min_{\boldsymbol{\lambda}^k\in\Delta^{N-1},\boldsymbol{\Lambda}^k\in(\Delta^{N-1})^2}\sum_{k\in[K]}\bigoplus_{i\in[N]}\boldsymbol{\lambda}_i^k l_i\left(\sum_{j=1}^{N}\boldsymbol{\Lambda}_{i,j}^k\boldsymbol{\theta}_j^k\right) \ , \tag{10}$$

Intuitively, we can interpret $\boldsymbol{\lambda}_i=\mathbb{P}(l=l_i)$ as the relative size of the dataset $\mathcal{D}_i$, compared to the size of the union of all the disjoint sets $\bigcup_{i=1}^N\mathcal{D}_i$. Similarly, the (i,j) -element of the stochastic matrix $\boldsymbol{\Lambda}$ can be interpreted as the conditional probability of selecting action $\boldsymbol{\theta}_j$, given we have already chosen l_i, i.e. $\boldsymbol{\Lambda}(i,j)=\mathbb{P}(\boldsymbol{\theta}=\boldsymbol{\theta}_j|l=l_i)$.

A. External Regret Procedure for Reducing Complexity

Now we establish the connection between the external regret of each client against their local model θ_i^k and the upper bound on the local losses related to the communication complexity in (3). It is easy to check that after some simple manipulation, the desired FL-convergence properties of (3) can be summarized as follows :

$$\left\{\begin{array}{l} \sum_{k=1}^T l_i(\boldsymbol{\theta}_a^k)-\sum_{k=1}^T l_i(\boldsymbol{\theta}_i^k)\le\mathcal{O}(f(T))+\epsilon(T),\forall i\in[N] \ , \\ \sum_{k=1}^T\left[l_i(\boldsymbol{\theta}_i^k)-\epsilon^*\right]\le\sum_{k=1}^T\epsilon(k)=\epsilon(T),\forall i\in[N] \ , \\ \sup_{i\in[N]} l_i(\boldsymbol{\theta}_a^*)\le\epsilon^* \ , \end{array}\right. \tag{11}$$

where $\epsilon(k)$ and $\epsilon(T)$ are upper bounds on the local error of client-i at round-k and T respectively. In order to reduce the communication complexity of FL we need to minimize the upper bounds on (11). In a regret learning framework [15], the learner optimizes the cumulative loss $L_T(\boldsymbol{\theta}_a^k)=\sum_{k=1}^{k=T}l_i(\boldsymbol{\theta}_a^k)$ over the sequence of actions $\boldsymbol{\theta}_a^k$. Below we consider client-i regret for using $\boldsymbol{\theta}_a^k$ at each round-k, instead of the pure action $\boldsymbol{\theta}_i^k$, towards minimization of $L_T(\boldsymbol{\theta}_a^k)=\sum_{k=1}^{k=T}l_i(\boldsymbol{\theta}_a^k)$:

Algorithm 1 External Regret Minimization

```
procedure RegEXT($\boldsymbol{\Lambda}_k^{\nu(t)}, W_k^{\nu(t)}, \boldsymbol{\theta}_{k+1}^{\nu(t)}, l_i(\boldsymbol{\theta}_{k+1,i}^{\nu(t)}), l_i(\boldsymbol{\theta}_{a,k}^{\nu(t)})$)
    $\eta\in(0,1)$ —- decay parameter
    Estimate local gradient: $\nabla_{\boldsymbol{\theta}}\hat{l}_i(\boldsymbol{\theta}_i^k) = \frac{l_i(\boldsymbol{\theta}_i^k)-l_i(\boldsymbol{\theta}_a^k)}{||\boldsymbol{\theta}_i^k-\boldsymbol{\theta}_a^{k-1}||}(\boldsymbol{\theta}_i^k - \boldsymbol{\theta}_a^{k-1})$

    Estimate cross error: $\hat{l}_i(\boldsymbol{\theta}_j^k) = l_i(\boldsymbol{\theta}_i^K) + \langle\nabla_{\boldsymbol{\theta}}\hat{l}_i(\boldsymbol{\theta}_i^k),\boldsymbol{\theta}_j^k-\boldsymbol{\theta}_i^k\rangle$
    $W_{k+1}^{\nu(t)}(i,j) = W_k^{\nu_t}(i,j)[1-\eta\hat{l}_i(\boldsymbol{\theta}_j^k)]$
    $\boldsymbol{\Lambda}_{k+1}^{\nu(t)}(i,j) = \frac{W_{k+1}^{\nu(t)}(i,j)}{\sum_{j\in C_\nu}W_{k+1}^{\nu(t)}(i,j)}$
     return $\boldsymbol{\Lambda}_{k+1}^{\nu(t)}$
```

$$\mathcal{R}_i^{Ext}(T)=\sum_{k=1}^T l_i(\boldsymbol{\theta}_a^k)-\sum_{k=1}^T l_i(\boldsymbol{\theta}_i^k) \tag{12}$$

Comparing (11) and (12) reveals that minimizing external regret $\mathcal{R}_i^{Ext}(T)$ of each client-i in [N] is equivalent to minimizing the communication complexity. Intuitively $\mathcal{R}_i^{Ext}(k)$ can be understood as the increase in client-i local error beyond $l_i(\boldsymbol{\theta}_i^k)$, due to combining models with other clients in $[N]$ with the combining distribution of $\boldsymbol{\Lambda}_i^k$ instead of sticking to the fixed distribution, $\mathbf{\Lambda_i}$ where $\mathbf{\Lambda}_{i,i} = 1, \mathbf{\Lambda}_{i,j} = 0, \forall j \neq i$.

Algorithm: In Algo. 1, we describe a procedure called RegEXT that updates the stochastic matrix $\mathbf{\Lambda}^k$ using a polynomial weights algorithm [16]. Using the upper bound on convex function $l_i(\boldsymbol{\theta}_a^k) \leq \sum_{j=1}^{N} \mathbf{\Lambda}_{i,j}^k l_i(\boldsymbol{\theta}_j^k)$ we get the following upper bound on $\mathcal{R}_i^{Ext}(k)$:

$$R_i^{Ext}(k) \leq \sum_{j=1}^{N} \mathbf{\Lambda}_{i,j}^k \left[l_i(\boldsymbol{\theta}_j^k) - l_i(\boldsymbol{\theta}_i^k)\right] \tag{13}$$

Based on (18), the server reduces $\mathbf{\Lambda}_{i,j}^k$ by a fraction of $(1-\eta)\hat{l}_i(\boldsymbol{\theta}_j^k)$, whenever the estimated loss $\hat{l}_i(\boldsymbol{\theta}_j^k) \geq 0$. At each communication round-k, the server asks each client to share their losses $l_i(\boldsymbol{\theta}_i^k)$ and $l_i(\boldsymbol{\theta}_a^{k-1})$ along with the model parameters $\boldsymbol{\theta}_i^k$, and uses the information from all clients together to estimate the quantity $\hat{l}_i(\boldsymbol{\theta}_j^k)$. Since the losses are just one-dimensional real numbers, they do not lead to any considerable increase in the size of the data shared on the client's side.

B. Swap Regret Procedure for Maximizing Client Utility

The condition in (6) can be stated in a per-round cumulative manner as follows:

$$\begin{cases} \sup_{w\in[N]} \sum_{k=1}^{T} \sum_{i\in[N]} \left[l_i(\boldsymbol{\theta}_a^k) - l_i(\boldsymbol{\theta}_w^k)\right] \leq 0 \ . \\ \sum_{k=1}^{T} \left[l_i(\boldsymbol{\theta}_w^k) - l_i(\boldsymbol{\theta}_w^*)\right] \leq \sum_{k=1}^{T} \mathbf{\Lambda}_{i,w}\epsilon(k) + l_i(\boldsymbol{\theta}_w^*) \ . \end{cases} \tag{14}$$

Having computed the stochastic matrix, $\mathbf{\Lambda}^k$, we now solve for the final model combination vector λ^k in (9) by minimizing the largest utility regret of a client in the given cluster $[N]$ or the swap utility regret within the cluster. We define the per-round swap regret of client-w in the pool $[N]$ as follows:

$$\mathcal{R}_{[N]}^{\text{Swp},w}(k) = \sup_{w\in[N]} \sum_{i\in[N]} \left[l_i(\boldsymbol{\theta}_a^k;\lambda^k) - l_i(\boldsymbol{\theta}_w^k)\right] \tag{15}$$

Our goal is to find a sequence of probability vectors $\lambda_k, \forall k \in [T]$ that minimizes the cumulative swap regret $\mathcal{R}_{[N]}^{\text{Swp},w}(T)$:

$$\min_{\lambda^k\in\Delta^{N-1}} \sum_{k=1}^{T} \mathcal{R}_{[N]}^{\text{Swp},w}(\lambda^k, \boldsymbol{\theta}^k) \ . \tag{16}$$

Reducing external regret to swap regret: we now follow a reduction technique [17] to leverage the existing external regret minimization procedure and the resulting distributions $\mathbf{\Lambda}_{i i\in[N]}$ to obtain a solution to the coverage maximization problem. Using Jensen's inequality, the swap regret can be bounded above as below:

$$\sum_{k=1}^{T} \mathcal{R}_{[N]}^{\text{Swp},w}(k) \leq \sum_{k=1}^{T}\sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_j^k \left[l_i(\boldsymbol{\theta}_j^k) - l_i(\boldsymbol{\theta}_w^k)\right] \tag{17}$$

Comparing (17) to the following upper bound on total external regret obtained from (18)

$$\sum_{k=1}^{T}\sum_{i=1}^{N} \lambda_i^k R_i^{\text{Ext}}(k) \leq \sum_{k=1}^{T}\sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i^k \mathbf{\Lambda}_{i,j}^k \left[l_i(\boldsymbol{\theta}_j^k) - l_i(\boldsymbol{\theta}_i^k)\right] \ , \tag{18}$$

we see that if $\lambda_j^k = \sum_{i=1}^{N} \lambda_i^k \mathbf{\Lambda}_{i,j}^k, \forall j \in [N]$, then the losses incurred by the external and swap regret minimizers coincide and we obtain a swap regret of at most $\epsilon(T)$, after playing λ^k for T rounds. Moreover, it is well known that setting the swap regret minimizing distribution, $\lambda_{sw}^k \in \Delta^{N-1}$ as the invariant probability vector of the stochastic matrix, $\mathbf{\Lambda}$ obtains the minimum swap regret. Therefore given a choice of $\mathbf{\Lambda}$ that minimizes the external regret, we set the maximum-coverage combination vector as its invariant probability vector. From Perron–Frobenius [18] theorem of non-negative matrices we know that such a unique probability vector always exists for any non-negative stochastic matrices.

$$\lambda_{sw}^k \mathbf{\Lambda}^k = \lambda_{sw}^k \tag{19}$$

IV. Clustering in Heterogeneous FL

A. Clustering Objective

From the external and swap regret guarantees we know that at the end of T communication steps the sum of the negative client utility and the total global loss will be bounded by the min achievable $\epsilon(T)$ in (11), which is the maximum of the local losses across the clients, and therefore depends on properties such as diameter or maximum subgradient in the global solution set $S_{\text{global}} = \bigotimes_{i\in[N]} \text{dom}(l_i)$. While optimization over $\mathbf{\Lambda}$ and λ lets us get close to the minima in the convex hull conv$(\{\boldsymbol{\theta}_i\}_{i\in[N]})$ the global loss $\sum_{i\in[N]} l_i(\boldsymbol{\theta}_a)$ will still be higher than desired if the degree of heterogeneity (as measured by the negative utility of the clients) between the clients is very high. Thus the purpose of the clustering algorithm here is to partition the global solution set S_{global} into smaller solution sets S_ν per cluster such that the maximum local loss value falls below the desired global loss threshold ϵ^* in (3).

$$\begin{cases} \bigotimes_{i\in[N]} \text{dom}(l_i) = \bigcup_{\nu\in 2^{[N]}} \bigotimes_{i\in[\nu]} \text{dom}(l_\nu) \text{ such that,} \\ \sup_{S_\nu\in S_{\text{global}}} \sup_{i\in S_\nu} l_i(S_\nu) \leq \epsilon^* \ . \end{cases} \tag{20}$$

We achieve this by running the regret minimization algorithm for a couple of FL rounds and removing the clients with low (or negative) utility or high external regret from the pool into a different cluster. That is we try to assign each client to a subset $C_\nu \subset [N]$ of the pool $[N]$ that maximizes its utility. Let $\mathcal{C} \in 2^{[N]}$ be a partition of $[N]$, i.e. $\bigcup_{\nu\in|\mathcal{C}|} C_\nu = [N]$, and $\mathcal{C}_\nu$ and $\mathcal{C}_{\nu'}$ two elements in $\mathcal{C}$ and therefore $\mathcal{C}_\nu \cap \mathcal{C}_{\nu'} = \emptyset$. By $\mathbf{\Lambda}_\nu \in \left(\Delta^{|\mathcal{C}_\nu|-1}\right)^2$ we denote the $|\mathcal{C}_\nu| \times |\mathcal{C}_\nu|$ stochastic matrix that has been extracted from the $N \times N$ root stochastic matrix, $\mathbf{\Lambda} \in \left(\Delta^{N-1}\right)^2$. We say that $\mathcal{C}$ is a successful partitioning of the client pool, if:

$$u_i(\mathbf{\Lambda}_\nu) \geq u_i(\mathbf{\Lambda}_{\nu'}), \forall i \in \mathcal{C}_\nu, \forall(\mathcal{C}_\nu, \mathcal{C}_{\nu'}) \in \mathcal{C}^2 \ . \tag{21}$$

Following the same argument we state the clustering objective function as below:

$$\max_{\mathcal{C}\in 2^{[N]}} \sum_{(\nu,\nu')\in\mathcal{C}^2} \inf_{i\in C_\nu} \left[u_i(\boldsymbol{\Lambda}_\nu) - u_i(\boldsymbol{\Lambda}_{\nu'})\right] \ . \qquad (22)$$

B. The Overall FL Mechanism

In **Procedure. 2** we put the regret minimizing subroutines with clustering subroutines within the generic flow of a federated learning scheme. We use $t \in \mathbb{N}$ to indicate the clustering steps, $k \in \mathbb{N}$ to indicate communication steps in FL. At $t = 1$, the total number of clusters is $|\mathcal{C}_t| = 1$. After running K rounds of FL method, where the model combination vectors of each cluster $\boldsymbol{\Lambda}_K^{C_{\nu(t)}}$ is computed using the swap regret minimization method, we check if the total external $\mathcal{R}_{\text{ext}}^{\nu(t)}$ for any cluster $C_{\nu(t)}$ is still beyond the desired complexity bound, $\delta f(Kt)$ (as shown in line-12 of Procedure. 2), where $\delta \in \mathbb{R}^{++}$ is a proportionality constant that can be tuned. If yes, then we decompose this cluster using the method shown in **Algorithm. 3**.

Here we apply the clustering principle stated in (22). i.e. expel the smallest subset of minimum-weight clients needed to meet the desired upper bound on regret at a given round, where the weight of a client is proportional to its utility in the cluster (based on pointwise approximation of utility [19]). In order to check if client-$w \in C_{\nu(t)}$ is the lowest utility client in cluster $C_{\nu(t)}$, we check if w has the least preference $\boldsymbol{\Lambda}_{j,w}^{\nu(t),K}$ from the other clients $j \in C_{\nu(t)}, j \neq w$ .i.e. if $w = \arg\min_{j\in C_{\nu(t)}} \sum_{i\in C_v} \boldsymbol{\Lambda}_{i,j}^{K,\nu(t)}$ (as shown in line-1 of Algorithm.3). After refining the clusters in this way then compute their averaged model using the updated stochastic matrices for the clusters, which will be sent to the clients in the next round of federated learning.

Algorithm 2 Clustering-based Federated Learning Mechanism

```
1:  procedure FLCL(S: Server, C: $N$ Clients)
2:      for $t = 1 \rightarrow T$ do
3:          Init: $W^{t,1} = I_{N\times N}$, $\boldsymbol{\Lambda}^{t,1} = \frac{1}{N} I_{N\times N}$
4:          for $v = 1 \rightarrow |\mathcal{C}_t|$ do (in each cluster $C_v$)
5:              for $k = 1 \rightarrow K$ do (FL-rounds begin)
6:                  S: send $\boldsymbol{\theta}_k^{a,\nu(t)}$ to C
7:                  C: send $L_k^t = \{\boldsymbol{\theta}_{k+1}^{i,\nu(t)}, l_i(\boldsymbol{\theta}_{k+1}^{i,\nu(t)}), l_i(\boldsymbol{\theta}_k^{a,\nu(t)})\}$ to
    S
8:                  S: $\boldsymbol{\Lambda}_{k+1}^{\nu_(t)}$ = RegExt($\boldsymbol{\Lambda}_k^{\nu(t)}, L_{t,k}$)
9:                  S: $\lambda_{k+1}^{\nu(t+1)} = \lambda_{k+1}^{\nu(t)} \boldsymbol{\Lambda}_{k+1}^{\nu(t)}$
10:                 S: $\boldsymbol{\theta}_{a,k+1}^{\nu(t)} = \lambda_{k+1}^{\nu(t)}, \boldsymbol{\theta}_t^t$
11:             $\mathcal{R}^{\nu(t)} = \sum_{i\in C_{\nu(t)}} \mathcal{R}_{i,K}^{\nu(t)}$
12:         while $\mathcal{R}^v(t) \geq \delta f(Kt)$ do
13:             $C^{\nu(t+1)}, \boldsymbol{\theta}_a^{\nu(t+1)}$ =DecompCL($\boldsymbol{\Lambda}_K^{\nu(t)}, \boldsymbol{\theta}_K^{\nu(t)}$)
        return $C^\nu, \boldsymbol{\theta}_a^\nu, \forall\nu \in [\mathcal{C}(t)]$
14:
15:
```

Algorithm 3 Hierarchical Decomposition-based Clustering

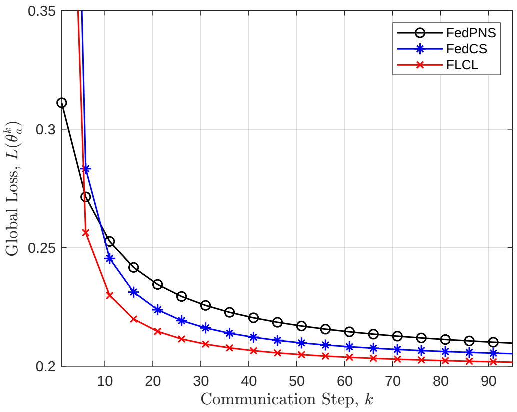
```
procedure DecompCL($\boldsymbol{\Lambda}_K^{\nu(t)}, \boldsymbol{\theta}_K^{\nu(t)}$)
    $w = \arg\min_{j\in C_v} \sum_{i\in C_v} \boldsymbol{\Lambda}_{i,j}^{K,t}$
    $\lambda_w^{\nu(t+1)} = 0$
    $E_{t+1} = E_t \cup w$
    $C^{\nu(t+1)} = E_t$
    $\boldsymbol{\Lambda}^{\nu(t+1)} = \boldsymbol{\Lambda}^{\nu(t)}(i, j \in {E_t}^2)$
    $\lambda^{\nu(t+1)} = \lambda : \lambda\boldsymbol{\Lambda}^{\nu(t+1)} = p$
    $\boldsymbol{\theta}_a^{\nu(t+1)} = \sum_{i\in C_{\nu(t+1)}} \lambda_i^{\nu(t+1)} \boldsymbol{\theta}_{i,K}^{\nu(t+1)}$
     return $C^{\nu(t+1)}, \boldsymbol{\theta}_a^{\nu(t+1)}$
```

=0

V. Numerical Analysis

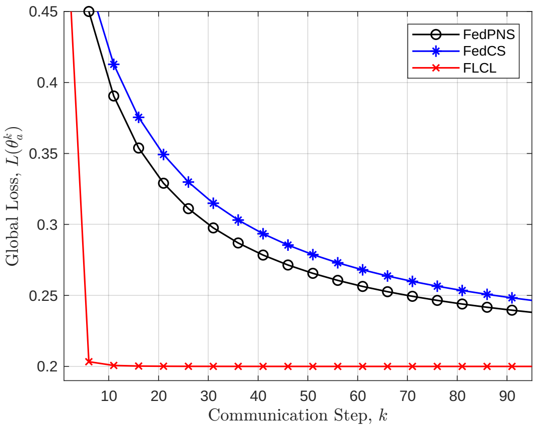
We evaluate our training mechanism in the context of a binary classification task where the data $\mathcal{D} \subset \mathcal{X} \times \{+1, -1\}$ is distributed according to a joint distribution, $\mathcal{P}(X, Y)$ on $\mathcal{X} \times \{+1, -1\}$. In particular, we consider a mixture model setting, and our assumptions about the data distribution resemble the setup used in [20], [21]. Although the focus of these works (impact of gradient descent on homogeneous data on margin and overfitting of resulting models) is quite different from ours, we extend this setup to generate multiple related but heterogeneous distributions. More precisely for client-i, we consider positive examples are distributed as $\mathcal{P}_i(X|Y = +1) \sim \mathcal{N}(\boldsymbol{\mu_i}, \sigma_i^2 \boldsymbol{I}_{d\times d})$ and the negative examples are distributed as $\mathcal{P}_i(X|Y = -1) \sim \mathcal{N}(-\boldsymbol{\mu_i}, \sigma_i^2 \boldsymbol{I}_{d\times d})$. Moreover we consider the hinge loss $l(z) = \max(0, 1-z)$ and then generate the loss function of client-i, $l_i : \mathbb{R}^p \rightarrow \mathbb{R}$, where $l_i(\boldsymbol{\theta}) = \mathbb{P}(Y = +1)\mathbb{E}_{X\sim\mathcal{N}(\boldsymbol{\mu_i}, \sigma_i^2 \boldsymbol{I}_{d\times d})}\big[\max\big(0, 1 - \boldsymbol{\theta}^T x\big)\big] + \mathbb{P}(Y = -1)\mathbb{E}_{X\sim\mathcal{N}(-\boldsymbol{\mu_i}, \sigma_i^2 \boldsymbol{I}_{d\times d})}\big[\max\big(0, 1 + \boldsymbol{\theta}^T x\big)\big]$. Next, in order to introduce heterogeneity, we create a collection of distinct distribution parameters $\{\boldsymbol{\mu_i}, \sigma_i^2 \boldsymbol{I}_{d\times d}\}_{i\in[N]}$. We sample σ_i^2 uniformly from an interval $[0.1, 1]$. For the mean parameters, we select a $\boldsymbol{\mu_i} \in \mathbb{R}^d$ randomly and then select a set of transformations $\Phi = \{\phi_i : \mathbb{R}^d \rightarrow \mathbb{R}^d, \phi_i(\boldsymbol{\mu_i}) = r_i \exp(-j\psi_i)\}$, that can be used to create the mean of each client's distribution, $\boldsymbol{\mu_i} = \phi_i(\boldsymbol{\mu_i})$. The magnitude $r_i \sim \mathcal{U}[1, 10]$ and the rotation $\psi_i \sim \mathcal{U}[0, \pi]$ are chosen from the said uniform distributions.

We compare our results against two sampling techniques from the literature, namely FedCS [3], and FedPNS [1]. In FedCS, the focus is on maximizing the aggregation or the coverage even at the expense of communication complexity, and therefore clients with a higher degree of heterogeneity. FedPNS on the other hand rejects clients with a higher degree of heterogeneity to minimize the communication complexity. In Fig. 1 and Fig. 2 we plot the number of communications rounds, k, in the x-axis, and the global loss, $l_g(\boldsymbol{\theta}_a^k)$ at these rounds on the y-axis. Please note that for FLCL we plot $l_g(\boldsymbol{\theta}_a^k) = \sum_{\nu\in\mathcal{C}_t} \frac{1}{|C_\nu|} \sum_{i\in C_\nu} l_i(\boldsymbol{\theta}_a^{k,\nu})$, i.e. the average global loss across the clusters on the y-axis, and for FedCS and FedPNS we plot the global loss of the averaged model, $\boldsymbol{\theta}_a^k$ on both in-sample ($S \subset [N]$) and out-of-sample $\tilde{S} \subset [N], \tilde{S} \cap S = \emptyset$) client datasets, i.e. we plot $l_g(\boldsymbol{\theta}_a^k) = \frac{1}{|S|} \sum_{i\in S} l_i(\boldsymbol{\theta}_a^k) + \frac{1}{|\tilde{S}|} \sum_{i\in\tilde{S}} l_i(\boldsymbol{\theta}_a^k)$. Therefore in the case of FLCL $l_g(\boldsymbol{\theta}_a^k)$ is the training loss, but in case of FedCS and FedPNS, $l_g(\boldsymbol{\theta}_a^k)$ is more than the training error, $\frac{1}{|S|} \sum_{i\in S} l_i(\boldsymbol{\theta}_a^k)$, i.e. it helps us capture the impact of reduced coverage due to sampling process. The rate of decay of these curves indicates the communication complexity and the error level at the terminal



(a) Global training loss of averaged model vs the number FL communication round in presence of input heterogeneity.

Fig. 1: Accuracy-complexity trade-off in the presence of input heterogeneity



(a) Global training loss of averaged model vs FL round in presence of output heterogeneity.

Fig. 2: Accuracy-complexity trade-off in the presence of output heterogeneity.

rounds indicates the minimum global error achieved by the the averaged models.

In Fig. 1, we consider a pool of clients where $< \boldsymbol{\mu_i}_i^+ | \boldsymbol{\mu_i}_j^+ >\leq 0$ are drawn with zero probability. I.e. the models in this case are still heterogeneous enough to slow down the learning process and increase global error, but the degree of heterogeneity is not at a detrimental level. Fig. 1 shows that in this case, the FedCS achieves the smaller global loss in the final round but the rate of decay is higher for FedPNS in the initial rounds. While this is the expected behavior for the benchmark methods, we see that in the lower error regime, i.e. for all rounds where the global loss is below 0.28, the proposed method FLCL achieves the best communication complexity compared to the others, and in the terminal rounds achieves the smallest global loss compared to the others as well, illustrating the benefits of learning multiple models for both the communication complexity and global loss value.

In Fig. 2, we consider a pool of clients where $< \boldsymbol{\mu_i}_i^+ | \boldsymbol{\mu_i}_j^+ >\leq 0$ and $< \boldsymbol{\mu_i}_i^+ | \boldsymbol{\mu_i}_j^+ >= -1$ are drawn with non-zero probability. The case of $< \boldsymbol{\mu_i}_i^+ | \boldsymbol{\mu_i}_j^+ >= -1$ could indicate the presence of clients with poor labeling or even clients that are maleficent attackers wanting to stop the FL training from succeeding. In this case, we see that the proposed method FLCL outperforms both the benchmarks by large margins both in terms of communication complexity and value of global loss. The reason for such drastic gain of the FLCL is that in the presence of data that is similar in the input domain but opposite in the output domain is akin to adversarial samples, and therefore by clustering we prevent large distortion in the function being learned.

VI. Conclusion

In this paper we propose an intrinsically motivated federated learning mechanism that achieves improved coverage-complexity tradeoff by partitioning the client pool into a set of clusters and then learning a global model for each of these clusters. While we consider the clients to have convex loss functions in the future we will look into the possibility of designing a similar mechanism for clients with non-convex losses.

References

[1] H. Wu and P. Wang, "Node selection toward faster convergence for federated learning on non-iid data," 2022.

[2] C. Li, X. Zeng, M. Zhang, and Z. Cao, "Pyramidfl: A fine-grained client selection framework for efficient federated learning," in *Proceedings of the 28th Annual International Conf. Mob. Comp. and Ntwrkng.*, ser. MobiCom '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 158–171.

[3] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *ICC 2019 IEEE International Conf. Comm. (ICC)*. IEEE, May 2019.

[4] L. Fu, H. Zhang, G. Gao, M. Zhang, and X. Liu, "Client selection in federated learning: Principles, challenges, and opportunities," 2023.

[5] E. S. Lubana, C. I. Tang, F. Kawsar, R. P. Dick, and A. Mathur, "Orchestra: Unsupervised federated learning via globally consistent clustering," 2022.

[6] D. K. Dennis, T. Li, and V. Smith, "Heterogeneity for the win: One-shot federated clustering," 2021.

[7] F. E. Castellon, A. Mayoue, J.-H. Sublemontier, and C. Gouy-Pailler, "Federated learning with incremental clustering for heterogeneous data," 2022.

[8] B. Luo, Y. Feng, S. Wang, J. Huang, and L. Tassiulas, "Incentive mechanism design for unbiased federated learning with randomized client participation," 2023.

[9] Y. Zhan, J. Zhang, Z. Hong, L. Wu, P. Li, and S. Guo, "A survey of incentive mechanism design for federated learning," *IEEE Trans. Emerging Topics in Comp.*, vol. 10, no. 02, pp. 1035–1044, apr 2022.

[10] J. Yang, S. Cao, C. Zhao, W. Niu, and L.-C. Tsai, "Portfolio-based incentive mechanism design for cross-device federated learning," 2023.

[11] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," 2020.

[12] I. Anagnostides, C. Daskalakis, G. Farina, M. Fishelson, N. Golowich, and T. Sandholm, "Near-optimal no-regret learning for correlated equilibria in multi-player general-sum games," in *Proceedings. 54th Annual ACM SIGACT Symp. Theory Comp.*, ser. STOC 2022. New York, NY, USA: Association for Computing Machinery, 2022, p. 736–749.

[13] P. G. Sessa, I. Bogunovic, M. Kamgarpour, and A. Krause, "No-regret learning in unknown games with correlated payoffs," 2019.

[14] Z. Wang, Y. Shen, and M. Zavlanos, “Risk-averse no-regret learning in online convex games,” in *Proceedings. 39th International. Conf. Machine. Learn.*, ser. Proceedings of Machine Learning Research, vol. 162. PMLR, 17–23 Jul 2022, pp. 22 999–23 017.

[15] G. J. Gordon, A. Greenwald, and C. Marks, “No-regret learning in convex games,” in *Proceedings of the 25th International Conference on Machine Learning*, ser. ICML ’08. New York, NY, USA: Association for Computing Machinery, 2008, p. 360–367.

[16] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic Game Theory*. New York, NY, USA: Cambridge University Press, 2007.

[17] A. Blum and Y. Mansour, “From external to internal regret,” *Journal of Machine Learning Research*, vol. 8, no. 47, pp. 1307–1324, 2007.

[18] R. B. Bapat and T. E. S. Raghavan, *Perron-Frobenius theory and matrix games*, ser. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1997, p. 1–58.

[19] A. Rubinstein and S. M. Weinberg, “Simple mechanisms for a subadditive buyer and applications to revenue monotonicity,” 2018.

[20] S. Frei, N. S. Chatterji, and P. L. Bartlett, “Benign overfitting without linearity: Neural network classifiers trained by gradient descent for noisy linear data,” 2023.

[21] N. S. Chatterji and P. M. Long, “Finite-sample analysis of interpolating linear classifiers in the overparameterized regime,” 2021.