Appendix

8. Code

We have released the code in github: https://github.com/nlokeshiisc/GCFL_Release/tree/master.

9. Notation

We list the notations and their corresponding description used in GCFL in the Table 2.

Notation	Description
$\mathcal{C}$	Set of clients $\{c_1, c_2, \cdots, c_N\}$
N	No. of clients
D_T	Training data partitioned across N clients i.e. $D_T = \bigcup_{i=1}^{N} D_i$
D_i	Data at each client i
D_S	Server's data
$\mathcal{X}_i^t, w_i^t$	Subset selected at client i on round t and its associated weights
ℓ_S	Server loss computed on D_S
ℓ_i	Loss at each client
η_l	local learning rate at each client
η_g	global learning rate at the server
E	Denotes the number of local gradients update steps that the client performs

Table 2. Important Notations and Descriptions

10. GCFL Algorithm

Algorithm 1 GCFL algorithm

Require: Clients $\mathcal{C} = \{c_1, \cdots, c_N\}$, Server $\mathcal{S}$, Training Data at client i $D_i = \{(x_{ij}, y_{ij})_{j=1}^{n_i}\}_{i=1}^{N}$, Server Data D_S, communication rounds T, budget b, number of clients per round m, local and global learning rates η_l, η_g, , local gradient steps E, rounds at which server gradient is broadcasted K

$\theta_0 \leftarrow$ INIT_f_θ; $\mathcal{X}_i^0 \leftarrow$ INIT_RANDOM_SAMPLES(i) $\forall i \in [N]$
for round $t \in [T]$ **do**
  server does:
  $C^t \leftarrow$ sample m out of N clients
  Broadcast $(\theta^t, \nabla_\theta \ell_S(D_S; \theta^t))$ if $t\%K = 0$ else Broadcast θ^t to C^t

  each client $c_i \in C^t$ **does:**
  $\mathcal{X}_i^t \leftarrow$ solve Eq. (5) using greedy algorithm if $t\%K = 0$ else $\mathcal{X}_i^{t-1}$
  Set $\theta^{'} \leftarrow \theta^t$
  for $e \in [E]$ **do**
    sample a mini-batch $\mathcal{B} \overset{i.i.d.}{\sim} \mathcal{X}_i^t$
    $\theta^{'} \leftarrow \theta^{'} - \eta_l \frac{1}{|\mathcal{B}|} \sum_{(x,y)\in\mathcal{B}} \ell_i(f_{\theta'}(x), y)$
  end for
  Broadcast $\delta_i^t \leftarrow \theta^t - \theta^{'}$ to $\mathcal{S}$

  server does:
  $\theta^{t+1} \leftarrow \theta^t + \eta_g \sum_{i\in S^t} \delta_i^t$

end for
returnFinal model f_{θ^T}

Algorithm 2 Coreset Selection Iteration for the $(k+1)^{\text{th}}$ Point

Require: Coreset $\mathcal{G}_i^k$, Weights $\mathbf{w}_i^k$, Budget b, Validation Gradient θ^t

Set $r^0 \leftarrow$ server's broadcasted validation gradient θ^t if $k = 0$

Calculate error residue: $r^k \leftarrow \sum_{j \in \mathcal{G}_i^k} \mathbf{w}_{ij}^k \nabla_\theta \ell_i^j(\theta^t) - r^{k-1}$

for Data point $j \in [n_i] \setminus \mathcal{G}_i^k$ **do**

  Calculate the distance: $d_j \leftarrow \left\| \nabla_\theta \ell_i^j(\theta^t) - r^k \right\|$

end for

Select the data point minimizing distance: $j^\star \leftarrow \underset{j}{\operatorname{argmin}} \; d_j$

return Next data point $j^\star$

Algorithm 1 presents the pseudocode of our proposed approach. The algorithm depicts the actions performed by both the server and clients at each round. The server's actions are are highlighted in orange and the clients' actions shown in blue color.

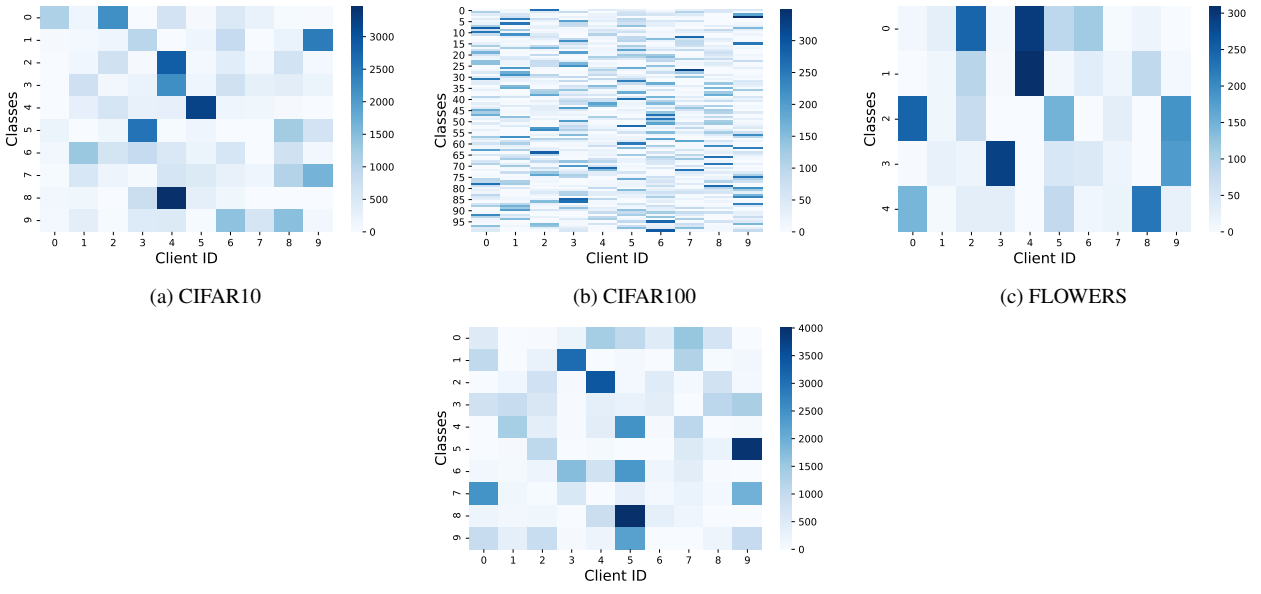
At a communication round t, first the server samples m clients (line 3), and then broadcasts θ^t to them. Additionally, if the clients need to perform coreset selection, the server also broadcasts the validation gradients. The clients subsequently use these to construct the coreset $\mathcal{X}_i^t$ iteratively by solving our label-wise OMP problem (line 8 of the algorithm). Then clients use the selected coreset to train the model for E epochs (line 10–13). Finally, the parameters of the updated model are uploaded back to the server (line 14). The server waits until it receives the updated model from all the sampled clients C^t, then it averages them and the updates the global model with the average(line 17). This completes one round of GCFL execution.

11. Datasets Details

In table 3 we list the details about the datasets used in the experiments. Further, to showcase the challenging nature of datasets used in our experiments, we show the *non-i.i.d.* nature of the partition across clients using heatmaps in Figure 10. We observe that each client is characterized by an allocation of data points D_i such that it is skewed in favor of one or two classes.

Dataset	#Classes ($\lvert\mathcal{Y}\rvert$)	#Clients' data ($\lvert\bigcup_{i=1}^{N} D_i\rvert$)	#Server's data ($\lvert D_S\rvert$)	#Test ($\lvert D_{\text{Test}}\rvert$)
Flowers	5	3270	200	200
FEMNIST	10	60000	5000	5000
CIFAR10	10	50000	5000	5000
CIFAR100	100	50000	5000	5000

Table 3. Dataset statistics



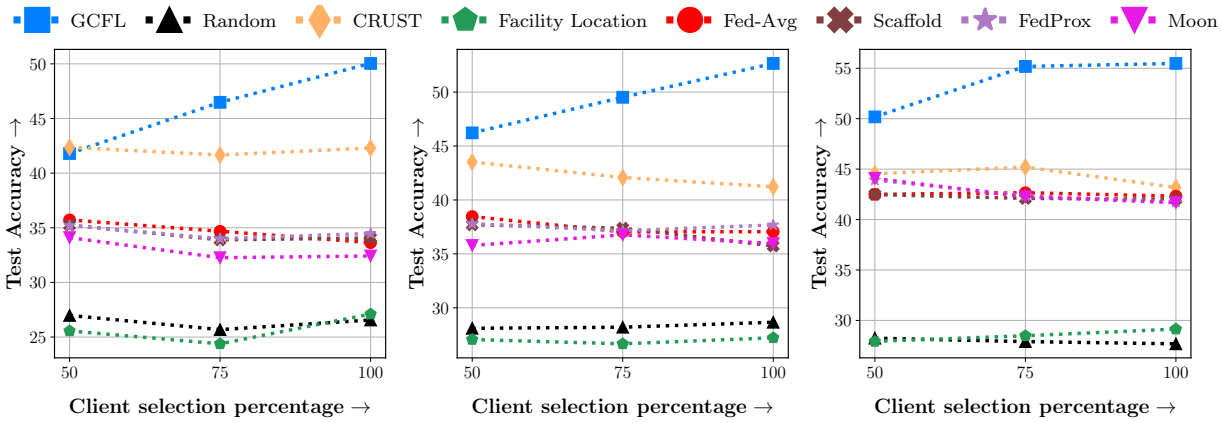
(a) CIFAR10 (b) CIFAR100 (c) FLOWERS

(d) FEMNIST

Figure 10. Classwise distribution of training instances across clients. The X axis spans the clients and the Y axis spans the classes. Each rectangle represents the number of instances that belong to a particular a class in a client. A dark rectangle in the cell (i, j) means that the i^{th} client has more instances of class j.

12. Additional experiments

12.1. Client Selection



(a) 10 clients (b) 15 clients (c) 25 clients

Figure 11. In this experiment we vary the number of participating clients m in each round. We experiment with CIFAR-10 dataset that is injected with 40% closed-set noise. Overall we observe that GCFL performs the best.

To test the robustness of GCFL in a setting where each communication round involves partial participation of clients, we conducted an experiment by varying the number of clients that are sampled in each round. In particular, we experimented with client participation ratios 50%, 75% and 100% and present the results on CIFAR10 dataset under 40% closed-set noise setting. From the figure 11 is it clear that the robustness of GCFL is not affected by the limited participation of clients.

12.2. Ablation study: Varying Budget b

All coreset selection algorithms' performances are affected by the size of the subset they are allowed to select. We present the effect of varying the sampling budget on coreset selection algorithms under 40% close-set noise in Figure 12. For a fair comparison, we make the number of SGD steps consistent across different sampling budgets. For CIFAR10, in Figure 12 a) we see that GCFL's performance is robust to sampling budget and other coreset methods slightly improve as budget size increases. We attribute this robustness to the fact that GCFL selects coreset based on label-wise last layer server gradients. However, for CIFAR100 dataset, in Figure 12 b) we see all the methods improve with increase in the budget size, since it is slightly a more difficult dataset having 100 classes.

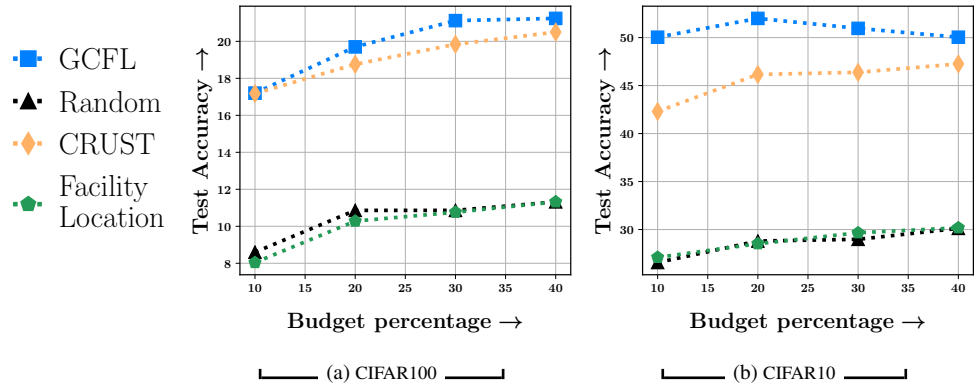


Figure 12. Performance of GCFL and other subset selection method as we vary budget in 40% closeset noise setting.

12.3. Ablation study: Varying *non-i.i.d.* -ness among clients α

We distributed the data to clients following [30] where we simulate the *non-i.i.d.* data partition by sampling class proportions from a symmetric Dirichlet Distribution parameterized with α. Typically, setting a lesser α would result in a very skewed class proportion across clients, and as α increases, we reach the uniform partition (*i.e.* i.i.d.) in the limit. We conduct experiments under the closed set label noise with a noise ratio of 40% to assess the performance of GCFL as against the standard Federated Averaging algorithm. The results are presented in Figure 13 which clearly elucidates the robustness of GCFL primarily attributed to its ability to balance the corset across classes by deliberately running the selection on a per-class basis alongside selecting noise-free informative points. On the other hand, Federated averaging, due to its sensitivity to noise, is unable to recover even when the partition reflects i.i.d. characteristics.

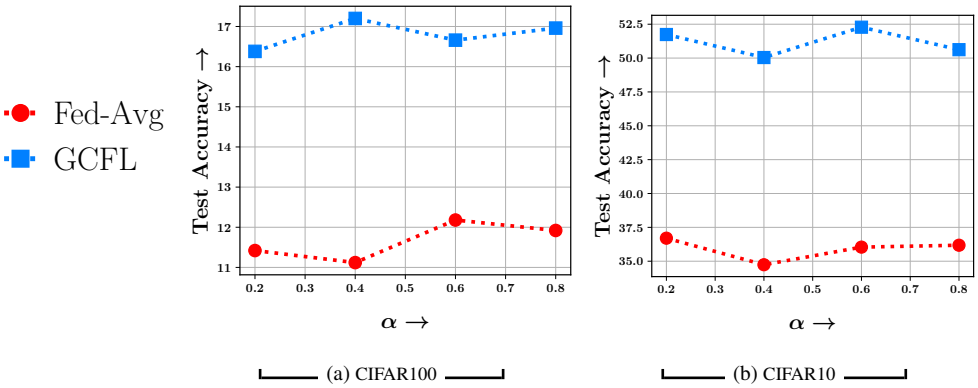


Figure 13. Performance of GCFL and Federated averaging as we vary α(parameter controlling *non-i.i.d.* split among clients) in 40% closeset noise setting.