Alexis Filippakopoulos, Dimitris Kastaniotis, Christos Theocharatos, Vangelis Vassalos

Irida Labs

Personalization in Distributed tinyML Applications via Adaptive Clustered Federated Learning

Introduction

From smart gadgets to cameras and sensors, IoT devices produce an immense volume of data. Due to the distributed nature of these devices, the data produced is highly decentralized and heterogeneous. Therefore, a need for AI solutions that consider this decentralized nature of the data, along with the constraints imposed by the IoT setting, while operating in a privacy-preserving manner becomes increasingly apparent.

Federated Learning (FL) allows multiple edge devices, referred to as clients that constitute a network, to collaboratively train a shared global model, capable of inferring from all tasks within that network. Each client trains its model locally, on its own data, sharing only its updated parameters. Training is coordinated by a central entity, referred to as the server, whose role is to broadcast the initial weights within the network, receive the updated parameters of all participants, aggregate the global model (*FederatedAveraging*) and broadcast it within the network. This constitutes a complete global training round in a federated setting. As a result, local data remains on the edge, rendering it decentralized, while the parameter-only exchange between clients and the server preserves, to an extent, both user and data privacy. However, real-world applications are defined by statistical heterogeneity among local distributions, diversity in tasks, as well as non-IID network-wide distributions. Due to these inherent problems, FL's convergence, along with the generalization capabilities of the global model, can be severely hindered, resulting in many clients being better off training solely on their local data.

In this light, *Personalized Federated Learning* (PFL) acknowledges the benefits of - accessing rather inaccessible data in a privacy-preserving manner and aims to equip each client with a unique and personalized model instance, tailored to its local data and client-specific tasks. The dominant approaches for achieving personalization, include but are not limited to, local fine-tuning, multi-task learning, model regularization and model interpolation by mixing the global model with each local model.

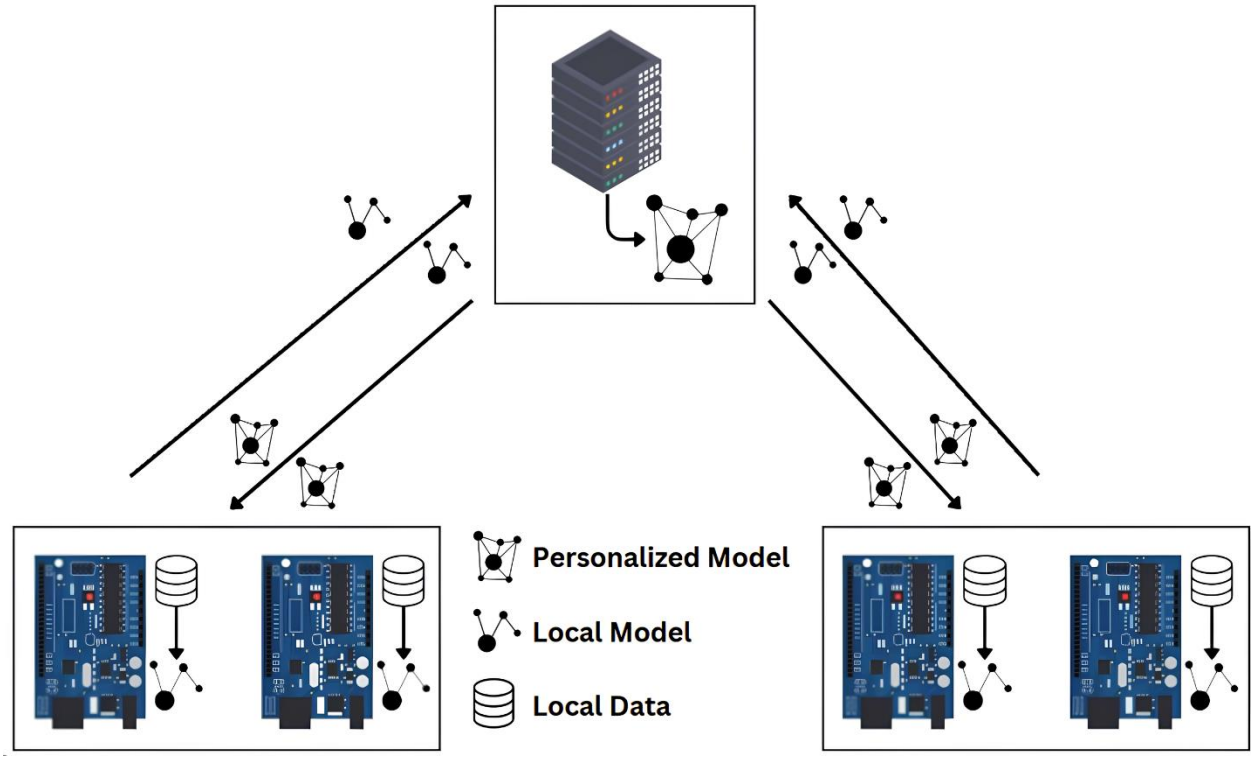

Figure 1. Visual representation of the federated framework.

Our approach, adopts model interpolation for obtaining robust client-specific models. Focusing on personalization and aiming to improve in-distribution generalization, we opt for Hierarchical Clustering based on the cosine similarities between local models' weights. We adaptively construct clusters of clients with similar tasks, effectively minimizing the divergence between local and global distributions. Consequently, we observe a significant increase in

performance, while exploring non-IID settings of increasing statistical heterogeneity and task diversity, among all local distributions.

Problem Formulation

Let a population of $k \in \{1, \dots, K\}$ clients, each maintaining a local distribution D_k, of size $|D_k|$, on domain $\Xi \coloneqq \mathrm{X} \times Y$, where $X \in \mathbb{R}^d$ is the input domain and $Y \in \mathbb{R}^T$ is the label domain. We denote as $\overline{D} = \cup_{k=1}^{K} D_k$ the network-wide (global distribution), which corresponds to the joint distribution of all K local distributions D_k.

For any model $h \in \mathcal{H}$, the loss function is defined as $\ell: \mathcal{H} \times \Xi \to \mathbb{R}^+$, while the true and empirical risks for said model h at a specific local distribution D_k are denoted as $\mathcal{L}_{D_k}(h) = \mathbb{E}_{(\boldsymbol{x},y)\sim D_k}[\ell(h(\boldsymbol{x}), y]$ and $\hat{\mathcal{L}}_{D_k}(h)$ respectively. Thus, any client k training its model (h_k) on D_k will obtain $h_k^* = argmin\hat{\mathcal{L}}_{D_k}(h_k)$.

In this setting, FL aims at obtaining a global model $\bar{h}^* = argmin\hat{\mathcal{L}}_{\overline{D}}(\sum_{k=1}^{K} \frac{|D_k|}{|\overline{D}|} h_k)$.

Similarly, PFL via model mixing constructs the personalized model of a client k with the following convex combination: $h_k^* = \lambda h_k^* + (1-\lambda)\bar{h}^*$, where λ is the mixing parameter and thereby $h_k^* = argmin\hat{\mathcal{L}}_{D_k}(\lambda h_k^* + (1-\lambda)\bar{h}^*)$.

The primary issue of this approach stems from the mixing of the global model $\bar{h}^*$ with every client model h_k^*. Specifically, the greater the divergence between $\overline{D}$ and D_k, the more likely we are to obtain a suboptimal set of parameters for h_k^* and subsequently a worse fit for D_k.

Motivation

An FL protocol can host thousands of edge devices simultaneously, thus, under the assumption of task diversity, there is a high probability that the network-wide distribution $\overline{D}$ will significantly diverge from some local distributions D_k. Hence, a mechanism for identifying and

grouping similar clients into subpopulations, whose joint distribution will diverge less from D_k compared to $\overline{D}$, is critical if we wish to maximize personalization.

In this light, *Clustered Federated Learning* (CFL) aims to do exactly that. However, how should one measure distribution/task similarity with no apriori knowledge of both? Both empirically and theoretically, parameters that are updated based on a fixed distribution are reflective of that distribution. Therefore, each client could pre-train its model for some *E* epochs and transmit its weights to the server to be used for measuring distribution similarity. There are many approaches for measuring similarity in high-dimensional spaces such as the Euclidean Distance, the Manhattan Distance and the Cosine Similarity, which are all valid options.

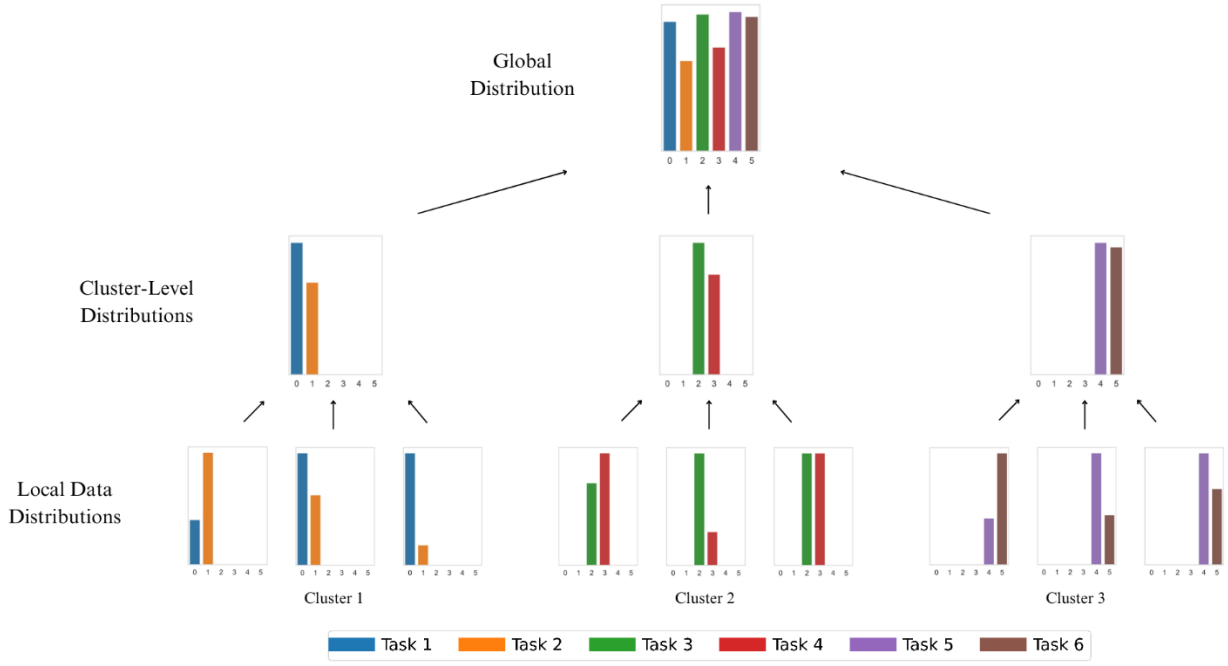

Figure 2. Visualization of the difference between local, cluster-level and global distributions.

Methodology

We opt for Hierarchical Clustering based on the cosine similarities of all client weights, to deduce the optimal subpopulations with similar tasks and distributions. The choice for cosine

similarity is based on empirical results drawn from extensive experimentation. We have found that cosine similarity with a threshold t=0.9, consistently finds the optimal clusters. Thereon, we do not diverge from the typical FL training procedure and our framework can be summarized in the following algorithm:

/* Runs on Server-Side */
for each client $k \in \{1, 2, \dots K\}$ **do**
    Transmit h^0 // Initial weights
    Receive h_k^E // Pre-trained client weights
end for
Calculate clients' **cosine similarities** using h_k^E
Cosine Similarity **Hierarchical Clustering** to form C clusters
for each cluster $c \in \{1, 2, \dots C\}$ **do**
    $h_c^0 \leftarrow \sum_{k=1}^{|c|} \frac{|D_k|}{|D_c|} h_k^E$ // Aggregate cluster-level initial weights
    for each client $k \in c$ **do**
        Transmit h_c^0
    end for
end for
for each global round $t = 0, 1, \dots T-1$ **do**
    for each client $k \in \{1, 2, \dots K\}$ **do**
        Receive h_k^{t+1} // Client's updated weights
    end for
    for each cluster $c \in \{1, 2, \dots C\}$ **do**
        $h_c^{t+1} \leftarrow \sum_{k=1}^{|c|} \frac{|D_k|}{|D_c|} h_k^{t+1}$ // Aggregate cluster-level global model
        for each client $k \in c$ **do**
            $h_k^{t+1} \leftarrow \lambda h_k^{t+1} + (1-\lambda) h_c^{t+1}$ // Aggregate client-specific model
            Transmit h_k^{t+1}
        end for
    end for
end for

/* Runs on Client-Side */
$h_k^0 = h^0$ //Initialize local model with received weights
for each pretraining round $e = 0, 1, \dots E-1$ **do**
    $h_k^{e+1} = h_k^e - \eta_e \widetilde{\nabla} \ell(h_k^e(\mathbf{x}), y)$ // Update local model
end for
Transmit h_k^{e+1} // Pre-trained local weights
Receive h_c^0 // Cluster-level initial weights
$h_k^{e+1} = h_c^0$ // Load cluster-level initial weights
for each global round $t = 0, 1, \dots T-1$ **do**
    $h_k^{t+1} = h_k^t - \eta_t \widetilde{\nabla} \ell(h_k^t(\mathbf{x}), y)$ // Update local model
    Transmit h_k^{t+1} // The updated local weights
    Receive h_k^{t+1} // The aggregated personalized weights
end for

Similarity Experiments

Since obtaining the optimal clusters is critical for improving personalization, we must define a robust mechanism for measuring distribution similarity. It is proven that model parameters are highly associated with the underlying distribution that is modeled. Hence, after each client receives the initial weights (h^0), it trains on its local distribution for *E* epochs before transmitting those weights (h_k^E) back to the server to be used for calculating distribution similarities.

We aim to answer the following questions:

1. *Should we use all layers' weights or partial weights?*
2. *How long should a client train for his weights to adequately reflect its local distribution?*

We set up three experiments with increasing amounts of task overlap and statistical heterogeneity among the clients. We utilize a CNN, comprised of 5 convolutional and 1 full connected layers, for a total size of 511KBs. Each client is trained and tested on a subset of 2 or 3 of the CIFAR-10 classes. We create a pathological Non-IID setting by constructing imbalanced subsets of the CIFAR-10 dataset.

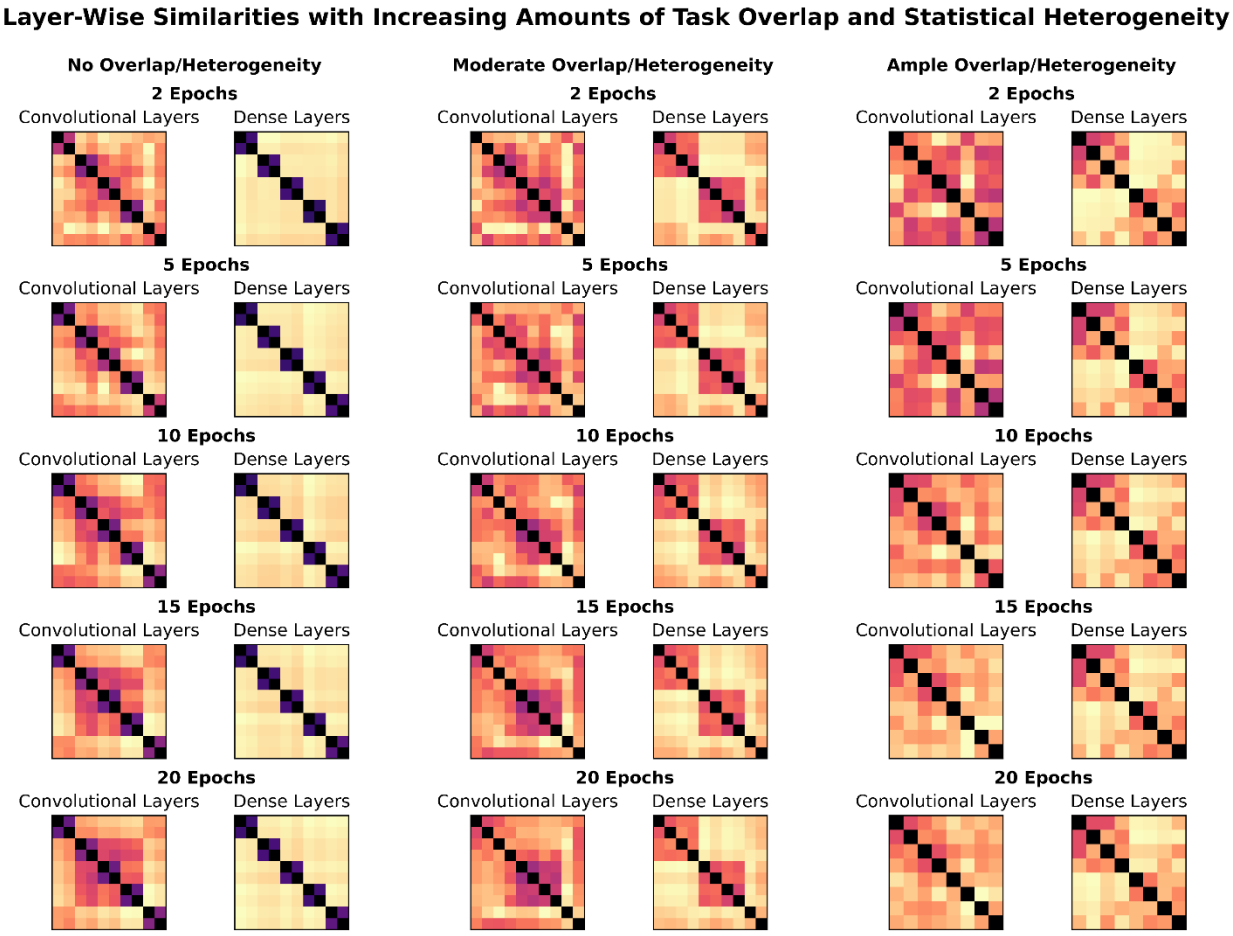

Figure 3. Visualizing the similarities between convolutional and fully connected layers of different client for different amounts of task overlap and statistical heterogeneity, in heatmap form.

The insights to be drawn from the figure above could be summarized in the following key points.

- Firstly, as other empirical experiments have shown, convolutional layers are not task-specific but rather general feature extractors, thus considering them when calculating any distance/similarity measure between parameters is not beneficiary.
- On the contrary, dense layers are task-specific and thus considering only them, will result in more precise measurements with less computations needed, compared to considering all layers.

- Secondly, pre-training for prolonged periods does not necessarily translate to greater precision in our similarity measurements. For the first two experiments the optimal clusters are formed with 2 epochs of training and maintained as the epochs increase. However, for the third experiment, as the epochs progress the second cluster loses its structure.

This argument is further reinforced by the following figure, where we plot the similarities computed during the different training checkpoints on the same client population.

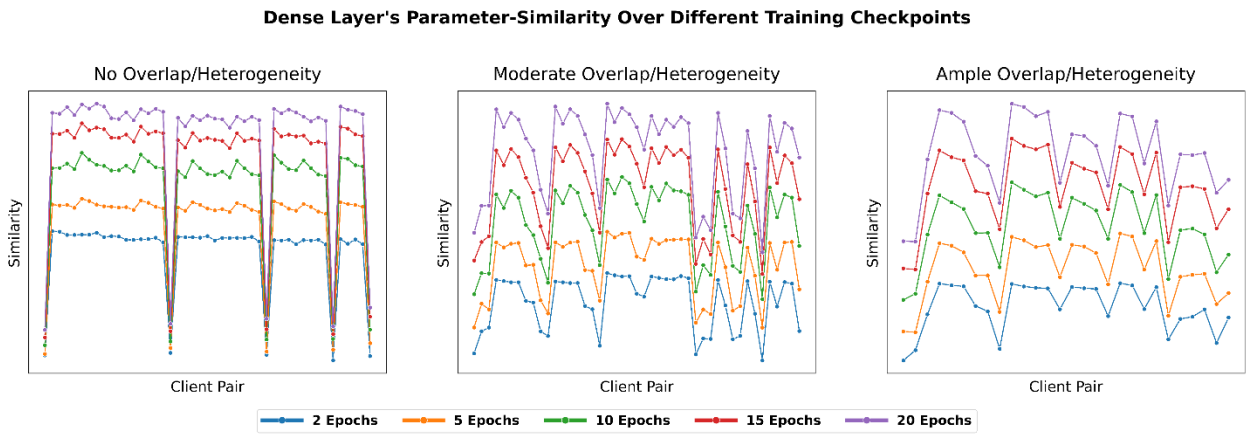

Figure 4. Visualizing the fully connected layer similarity between clients with different amounts of statistical heterogeneity in simple line form.

We observe that extended training epochs lead to increased parameter similarity across clients, but the relative similarity between specific pairs remains relatively unchanged. This is indicative that there is no need for excessive pre-training, since with few epochs, model parameters seem to encapsulate enough information about the underlying distribution, to enable adequate similarity measurements.

Performance Experiments

To evaluate the framework's performance, we employ 16 clients and set up two pathological Non-IID settings, with imbalanced local distributions, using CIFAR-100 classes. Firstly, simulating a scenario where all tasks within a cluster are shared by all its clients, we assign two tasks per client for a total of 16 tasks with the optimal number of clusters being 8. Conversely, we simulate a scenario where within each cluster there are tasks that do not concern all clients, we assign three tasks per client for a total of 20 tasks with 4 optimal clusters. Again, we opt for the same CNN, comprised of 5 convolutional and 1 full connected layers, for a total size of 511KBs. To assess the network's overall performance, we average all metrics across our clients.

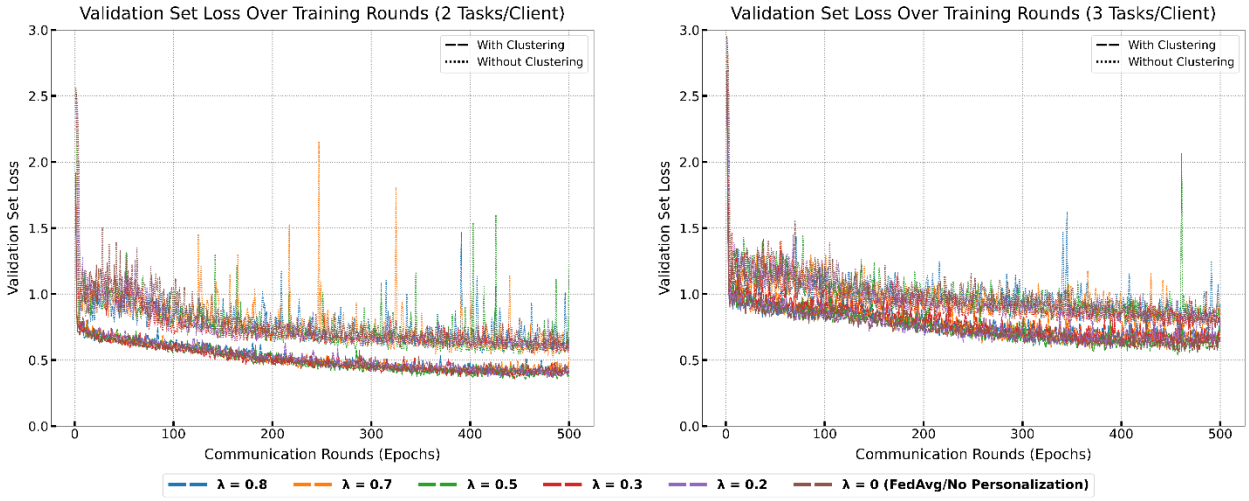

Figure 5. Validation loss among clients in a Non-IID setting, with different amounts of statistical heterogeneity and task overlap, with respect to λ. Comparing the effectiveness of clustering similar clients as previously described.

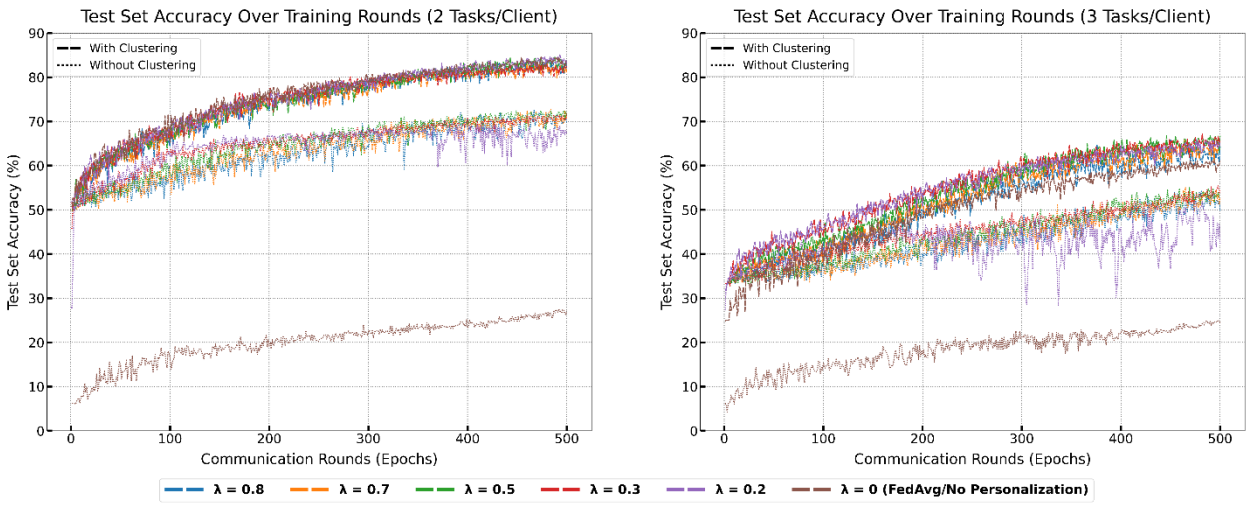

Figure 6. Test accuracy among clients in a Non-IID setting, with different amounts of statistical heterogeneity and task overlap, with respect to λ. Comparing the effectiveness of clustering similar clients as previously described.

Our conclusions can be summarized to the following points:

1. It is evident that clustering and cluster-level global models have a tremendous impact in model convergence and performance, compared to plain personalization. Furthermore, the difference between *FedAvg* (no clustering $\lambda = 0$) with all the other approaches, highlights the need for personalization in federated settings.
2. From the validation figure, we deduce that clustering reduces oscillations and smoothens the loss's convergence, by preventing the models from drifting too much from their local distributions. Additionally, clustered clients reach lower convergence points, indicating a better fit on the data.
3. From the accuracy figure, it is conspicuous that the clustering framework steadily outperforms its counterpart, reaching an average of +12% and +11%, on our respective experiments, in terms of test set accuracy.
4. Regardless of clustering or not, no particular λ stands out compared to others, besides no--clustering $\lambda = 0$, which does not personalize and is the worst performing. Interestingly

though, on the 2-tasks experiment, the highest recorded precision belonged to no-clustering $\lambda = 0$. This suggests, that in Non-IID and imbalanced settings, constructing only one global model will result in a bias towards the majority tasks, leading to fewer predictions for underrepresented tasks, which tend to be correct and hence inflate the model's precision.

5. A general rule for choosing an appropriate λ is the following. As λ increases we incorporate more of the locally trained model in each aggregation and vice versa. Thus, the more we expect a local distribution to diverge from its cluster-level or global distribution, the greater λ should be.
6. The difference in performance between the two experiments is attributed to the constrained capacity of our classifier. Given more parameters, the models would have performed better but with notable differences between using and not using clustering.

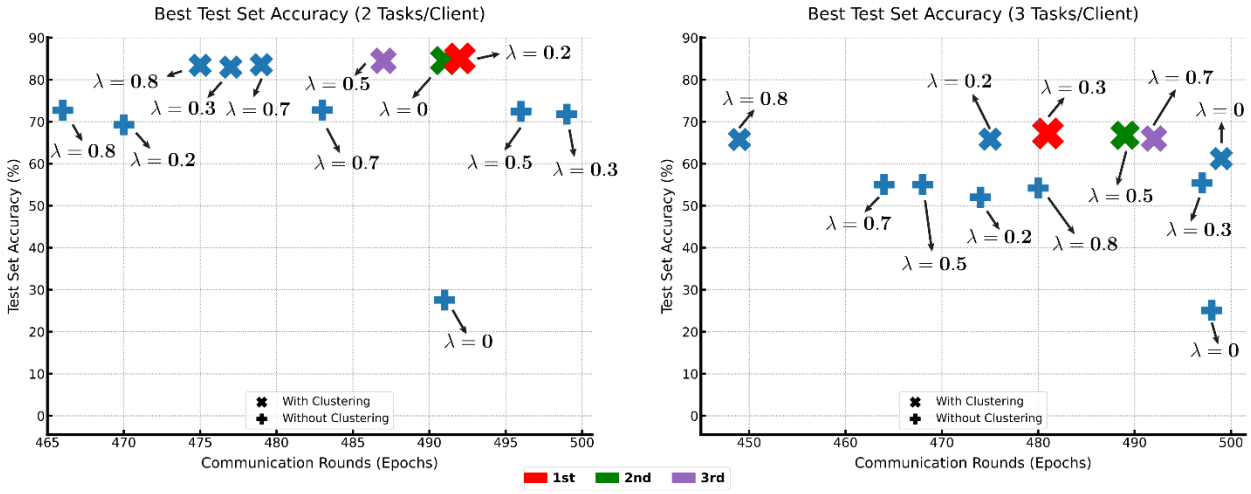

Figure 7. Test accuracy among clients in a Non-IID setting, with different amounts of statistical heterogeneity and task overlap, with respect to λ. Comparing the effectiveness of clustering similar clients as previously described.