Controllable Heterogeneous Model Aggregation for Personalized Federated Learning

Jiaqi Wang[1] **Qi Li**[2] **Lingjuan Lyu**[3] **Fenglong Ma**[1]*
[1]The Pennsylvania State University [2]Iowa State University [3]Sony AI
{jqwang,fenglong}@psu.edu, qli@iastate.edu, lingjuan.lv@sony.com

Abstract

Federated learning, a pioneering paradigm, enables collaborative model training without exposing users' data to central servers. Most existing federated learning systems necessitate uniform model structures across all clients, restricting their practicality. Several methods have emerged to aggregate diverse client models; however, they either lack the ability of personalization, raise privacy and security concerns, need prior knowledge, or ignore the capability and functionality of personalized models. In this paper, we present an innovative approach, named `pFedClub`, which addresses these challenges. `pFedClub` introduces personalized federated learning through the substitution of controllable neural network blocks/layers. Initially, `pFedClub` dissects heterogeneous client models into blocks and organizes them into functional groups on the server. Utilizing the designed `CMSR` (Controllable Model Searching and Reproduction) algorithm, `pFedClub` generates a range of personalized candidate models for each client. A model-matching technique is then applied to select the optimal personalized model, serving as a teacher model to guide each client's training process. We conducted extensive experiments across three datasets, examining both IID and non-IID settings. The results demonstrate that `pFedClub` outperforms baseline approaches, achieving state-of-the-art performance. Moreover, our model insight analysis reveals that `pFedClub` generates personalized models of reasonable size in a controllable manner, significantly reducing computational costs[2].

1 Introduction

Federated learning (FL) [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15] is a prevalent method to train machine learning models collaboratively without centralizing clients' data on a cloud server. However, many current FL training frameworks demand uniformity in deep neural network structures among client models, a requirement often too stringent for practical, real-world applications. An alternative approach involves equipping clients with heterogeneous models, introducing a new challenge: *how to aggregate these diverse models within the federated learning framework effectively*.

Recently, various approaches have emerged to address the challenge of aggregating heterogeneous models, particularly in the context of personalized FL. Some methods leverage *additional information*, such as class information [16], logits [17, 18], and label-wise representations [19], as intermediaries for exchanging information between clients and the server. While seemingly straightforward, this approach raises significant privacy concerns, especially regarding the potential exposure of sensitive client data. To mitigate these privacy concerns, techniques such as *distillation* [20, 21, 22] and *model reassembly* [23] have been introduced in heterogeneous FL, wherein only model parameters are exchanged, akin to traditional FL approaches [1]. Despite demonstrating effectiveness in aggregating

*Corresponding author.

[2]The source code can be found at `https://github.com/JackqqWang/24club`.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

https://doi.org/10.52202/079017-1733

heterogeneous models, both distillation and model reassembly techniques in FL suffer from a common drawback – **lack of control over personalized model generation**.

Distillation-based approaches inherently necessitate the establishment of a unified model as the global model, informed by prior insights [21]. This global model is then distributed to clients to guide their training efforts. However, a smaller consensus global model may struggle to extract heterogeneous knowledge from clients, often resulting in a larger size. This poses challenges for smaller clients with limited computational resources to run the shared large global model effectively. Similarly, model reassembly-based approaches also encounter a related issue. While they generate personalized models for each client, these personalized models may be significantly larger than the clients' capacity, posing challenges for implementation.

To further investigate these limitations, we conducted an experiment using the state-of-the-art model reassembly approach, pFedHR [23], on the SVHN dataset, where each client operated a distinct model. (*Additional details and further discussions can be found in Section 4.4.*) We evaluated the parameter size of the original models and the average number of parameters in the personalized models received across communication rounds. Subsequently, we illustrated the parameter size differences between these two types of models in Figure 1. The entire bar represents the average parameter size of the generated personalized model, with the original model size depicted in blue and the increased size shown in red. Our preliminary findings indicate that the personalized models produced by pFedHR are notably larger than the original local models.

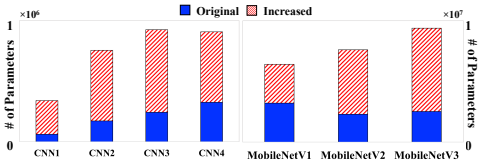

Figure 1: Lack of controllability demonstration using pFedHR [23] on the SVHN dataset with seven different client models by comparing the original model size (blue bars) to the generated personalized client model size (entire bars).

To address this issue, in this paper, we introduce a novel approach for heterogeneous model aggregation, named `pFedClub`, aiming to achieve **p**ersonalized **Fed**erated learning through **C**ontrollable neural network b**l**ock s**ub**stitution, as depicted in Figure 2. `pFedClub` receives heterogeneous models uploaded from clients on the server. `pFedClub` first decomposes the heterogeneous client models into different blocks and subsequently clusters these blocks based on their functionalities (refer to Section 3.2.1). `pFedClub` explores a novel neural network block substitution technique to achieve this objective, as detailed in Section 3.2.2. Specifically, `pFedClub` aims to substitute the r-th block $\mathbf{B}_{m,r}^t$ within the m-th client model $\mathbf{w}_m^t$ with a block selected from the same group as $\mathbf{B}_{m,r}^t$ during communication round t. This approach ensures both the functionality of personalized models and their similarity to the original models. To enhance the diversity of the generated models, we permit arbitrary substitutions from the group for the first block (**Step 1**). For subsequent blocks, we introduce *an order-constrained block search strategy* (**Step 2**), ensuring the quality of the generated models. If the order constraint halts the substitution prematurely, the remaining blocks are directly added to the substituted ones (**Step 3**). This completion strategy not only reduces the number of newly added parameters in the stitching blocks but also ensures similar functionality. As `pFedClub` may generate multiple personalized candidate models for each client, we employ the *similarity-based model-matching* technique to select the personalized model, as discussed in Section 3.2.3. The selected personalized model is then distributed to the respective client as a teacher model, guiding the client model training process through knowledge distillation.

It is essential to emphasize that the proposed `pFedClub` framework is designed to be both general and flexible, allowing for the incorporation of strict controllable constraints during the personalized candidate generation process. For instance, constraints such as the model size of the generated candidates can be easily integrated into the framework. Furthermore, introducing controllability within `pFedClub` enables the mitigation of computational and communication costs compared to current model reassembly-based approaches. By incorporating controllable constraints, `pFedClub` offers greater adaptability and efficiency in handling personalized model generation. Experimental results demonstrate that `pFedClub` achieves state-of-the-art performance on three benchmark datasets under both IID and non-IID settings, demonstrating the effectiveness of the proposed aggregation strategy.

https://doi.org/10.52202/079017-1733

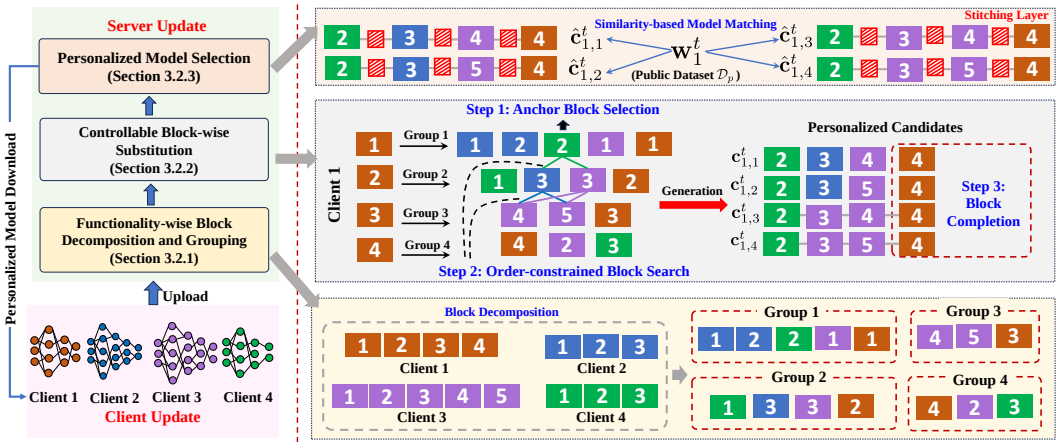

Figure 2: Overview of the proposed pFedClub. We take four clients with heterogeneous models as an example. The numbers denote the blocks' indexes. The number of functional groups is 4. We take Client 1 as an example to demonstrate how pFedClub works to generate personalized candidate models. Note that the arbitrary substitution for Block 1 of Client 1 is the second block from Client 4.

2 Related Work

Model Heterogeneity in Federated Learning. Heterogenous model cooperation is a challenging task in FL. Researchers have explored submodel training techniques [24, 25], focusing on training a shared large global model by sending masked heterogeneous models to appropriate clients. However, these approaches often fail to provide personalized models for individual clients. Furthermore, they are considerably constrained by the limitations in freedom of model selection. In addition, FedDF [20] and FedKEMF [21] conduct ensemble distillation, but the settings of FedDF are different from ours and not for model personalization. FedKEMF utilizes mutual knowledge distillation with the requirement of predefined model structures. Besides, HeteroFL [24] and FlexiFed [25] are restricted to the requirements of the client model structures. Other related research work needs extra information to be exchanged between the server and clients, e.g., logits in FCCL [17], class scores in FedMD [16], and label-wise representation in FedGH [19], which raises the concerns of privacy [26]. The most recent work pFedHR [23] provides a layer-wise model reassembly approach to solve the challenge of model heterogeneity in federated learning. However, it has several limitations, as we discuss in Section 1.

Personalized Federated Learning. Instead of maintaining one global model, personalized FL cares more about each local model's performance, which is more sufficient and practical. In [27], the authors add a proximal term to the local optimization loss function to bound the difference between the local and global model updates. The aggregated global model is treated as the initial shared model from the meta-learning perspective in [28]. In [29], the authors design a new regularized client loss to optimize the local model to achieve personalization. However, the discussed personalized federated learning work assumes that the clients have to share identical model structures.

3 Methodology

3.1 Overview

Let $\mathcal{D}_n = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{|\mathcal{D}_n|}$ denote the training data stored in the n-th local client L_n, where $\mathbf{x}_i$ denotes the data, $\mathbf{y}_i$ is the ground truth, and $|\mathcal{D}_n|$ is the number of training data. Each client employs a deep neural network-based model $\mathbf{w}_n$ to train on its training data. Note that the client models $\{\mathbf{w}_1, \cdots, \mathbf{w}_N\}$ do not share identical network structures, where N is the total number of clients. The overview of the proposed pFedClub is depicted in Figure 2, containing server update and client update. During each communication round t, the randomly chosen M ($M \ll N$) client models $\{\mathbf{w}_1^t, \cdots, \mathbf{w}_M^t\}$ will be uploaded to the server to generate their personalized models $\{\mathbf{w}*_1^t, \cdots, \mathbf{w}*_M^t\}$. These personalized models will be sent to the corresponding clients as teachers

https://doi.org/10.52202/079017-1733

to guide the update of client models $\{\mathbf{w}_1^{t+1}, \cdots, \mathbf{w}_M^{t+1}\}$ in the next communication round. Next, we provide the details of the model design.

3.2 Server Update

The model heterogeneity of the uploaded M client models $\{\mathbf{w}_1^t, \cdots, \mathbf{w}_M^t\}$ at the t-th communication round makes it challenging to generate personalized client teachers $\{\mathbf{w}*_1^t, \cdots, \mathbf{w}*_M^t\}$. To tackle this challenge, a new controllable block-wise substitution-based personalized model aggregation approach is proposed, which not only maintains the functionality of each block in the originally uploaded client models but also injects new knowledge provided by other models to further achieve model personalization. To this end, we first divide each client model $\mathbf{w}_m^t$ into blocks and then group blocks into different functionality clusters.

3.2.1 Functionality-wise Block Decomposition and Grouping

Block Decomposition. Except for recurrent deep neural networks such as the long-short term memory network [30], most of the remaining ones, such as the family of convolutional neural networks (CNN), can be treated as block-stacked neural networks[3]. Let R_m denote the number of blocks in client model $\mathbf{w}_m^t$. We then decompose $\mathbf{w}_m^t$ into blocks $\{\mathbf{B}_{m,1}^t, \cdots, \mathbf{B}_{m,r}^t, \cdots, \mathbf{B}_{m,R_m}^t\}$. Note that a block is either a convolutional net, a fully connected layer, or a building block associated with a shortcut connection such as residual neural networks [31].

Block Grouping. After decomposing each model, we then apply the K-means algorithm to group blocks based on their functionality. Specifically, we apply the centered kernel alignment (CKA) technique [32] to calculate the similarity between two blocks as follows:

$$\textbf{sim}(\mathbf{B}_{m,i}^t, \mathbf{B}_{n,j}^t) = \textbf{CKA}(\mathbf{x}_{m,i}^t, \mathbf{x}_{n,j}^t) + \textbf{CKA}(\mathbf{B}_{m,i}^t(\mathbf{x}_{m,i}^t), \mathbf{B}_{n,j}^t(\mathbf{x}_{n,j}^t)), \tag{1}$$

where $\mathbf{x}_{\cdot,i}^t$ denotes the input of the block $\mathbf{B}_{\cdot,i}^t$, and $\mathbf{B}_{\cdot,i}^t(\mathbf{x}_{\cdot,i}^t)$ represents the output of the block $\mathbf{B}_{\cdot,i}^t$ ($\cdot = m$ or n). Let $\{\mathcal{G}_1^t, \cdots, \mathcal{G}_K^t\}$ denote the K functionality groups generated by the K-means algorithm, where each group $\mathcal{G}_k^t$ contains multiple blocks from different models with similar functions.

3.2.2 Controllable Block-wise Substitution

Intuitively, if each pair of corresponding blocks of two models has similar functionality, the whole models should also be similar. Based on this straightforward intuition, we propose to replace each block $\mathbf{B}_{m,r}^t$ with a function-similar one chosen from the group $\mathcal{G}_k^t$, where $\mathbf{B}_{m,r}^t \in \mathcal{G}_k^t$. Let G_k denote the number of function-similar blocks in each group $\mathcal{G}_k^t$. Arbitrarily replacing each block without any constraints will produce a vast number of candidate models, which is approximately equal to $G_k^{R_m}$, where R_m is the number of blocks in model $\mathbf{w}_m^t$. It is time-consuming to update all candidates.

To reduce the size of the candidate model pool, a naive solution is to randomly choose a fixed number of models from the pool first and then use the model similarity score to select the personalized teacher model. Although this approach increases the diversity of the candidate model generation, it is uncontrollable to introduce too much randomness, and the low-quality candidates may reduce the convergence rate of the federated system training. To solve these issues, we design a **C**ontrollable **M**odel **S**earching and **R**eproduction (`CMSR`) algorithm (as depicted in Algorithm 1), which is a greedy-based search approach to reproduce a set of personalized candidate models for each $\mathbf{w}_m^t$ by considering both **diversity** and **quality**. In particular, `CMSR` consists of three key steps: anchor block selection, order-constrained block search, and block completion. Next, we describe the details of each step.

Step 1: Anchor Block Selection (Alg. 1 lines 3-7). Assume that the first block $\mathbf{B}_{m,1}^t$ in $\mathbf{w}_m^t$ belongs to the group $\mathcal{G}_k^t$. `CMSR` then randomly selects one block in $\mathcal{G}_k^t$ as the substitution. The substituted block will be treated as the anchor/starting block of all the candidate models. As shown in Figure 2, $\mathbf{w}_{4,2}^t$ is selected as the substitution of $\mathbf{w}_{1,1}^t$. Note that a naive solution to select the anchor block may use the block with the largest similarity score calculated via Eq. (1) instead of randomly choosing one. However, such a solution significantly reduces the diversity of the generated candidates, further limits the extra knowledge borrowed from other models, and finally hurts the personalization of teacher models. Besides, $\mathbf{B}_{m,1}^t$ has a chance to be selected with a probability $\frac{1}{G_k}$.

[3] In this paper, we do not decompose recurrent deep neural networks, which is our future work.

Algorithm 1: The CMSR Algorithm

input : Client model $\mathbf{w}_m^t$ and block clusters $\{\mathcal{G}_1^t, \cdots, \mathcal{G}_K^t\}$
output : Candidate set $\{\mathbf{c}_{m,1}^t, \cdots, \mathbf{c}_{m,S_m}^t\}$

```
1  Initialize candidate blocks C_m^t = {};
2  for r ← 1, ··· , R_m do
3      // Step 1: Anchor Block Section
4      if r = 1 then
5          Randomly select one block B_{p,q}^t from the group that contains B_{m,1}^t;
6          Add the substituted block to C_m^t[1] = [B_{p,q}^t];
7          Record the block index q;
8      // Step 2: Order-constrained Block Search
9      if r > 1 then
10         Initialize block index set I_{m,r}^t = [];
11         // Assume that B_{m,r}^t ∈ G_k^t
12         for B_{·,u}^t ∈ G_k^t do
13             if u > q then
14                 Add the block to C_m^t[r];
15                 Add the block index to I_{m,r}^t;
16         if I_{m,r}^t = ∅ then
17             break;
18         q ← min(I_{m,r}^t);
19 // Candidate Generation
20 Use C_m^t to generate candidates that satisfy the order condition;
21 if the number of blocks of the candidate model is smaller than R_m then
22     Run Step 3: Block Completion to complete the remaining blocks;
return : {c_{m,1}^t, ··· , c_{m,S_m}^t}
```

Step 2: Order-constrained Block Search (Alg. 1 lines 8-18). After selecting the anchor block in Step 1, CMSR then finds the substitutions for the following blocks. The simplest solution is repeating the previous step $R_m - 1$ times to generate a candidate model. As we discussed before, it may generate low-quality candidate models.

To avoid this problem and generate controllable high-quality candidates, we maintain the order of the selected blocks, even from different client models, as a hard constraint [33]. Mathematically, let $\mathbf{B}_{p,q}^t$ denote the substitution of the r-th block of $\mathbf{B}_{m,r}^t$, where p is the model index and q is the block index. For any substitution of the $(r+1)$-th block $\mathbf{B}_{\cdot,u}^t$ should satisfy the constraint $q < u$. As shown in Figure 2, the substitutions of $\mathbf{w}_{1,2}^t$ include $\mathbf{w}_{2,3}^t$ and $\mathbf{w}_{3,3}^t$ since the index of the first block's substitution $\mathbf{w}_{4,2}^t$ is 2.

Only using the order constraint is sufficient for the proposed pFedClub to generate high-quality and informative candidates. First, maintaining the order of block functions can guarantee that the candidate model has functionality similar to the original model. Second, it also avoids pre-defining the order of operation types, which releases the constraints of personalized teacher model generation and further increases the diversity of candidates. Third, such an approach is capable of generating similar-sized personalized models for clients. It is essential for several applications with limited computational resources, such as smart devices. However, pFedHR cannot control the size of the generated candidates.

It is worth noting that, aside from the order constraint, pFedClub is highly adaptable and can easily incorporate other types of constraints. In Section 4.4, we will delve into the integration of the model size constraint, demonstrating the framework's versatility and ability to accommodate various constraints for personalized model generation.

Step 3: Block Completion (Alg. 1 lines 19-22). Step 2 may stop at the certain block $r < R_m$ due to the block order constraint. To maintain the original functional structure, we will add the remaining blocks of $\mathbf{w}_m^t$, i.e., $\{\mathbf{B}_{m,r+1}^t, \cdots, \mathbf{B}_{m,R_m}^t\}$ to the substitutions. In such a way, pFedClub can

https://doi.org/10.52202/079017-1733

generate a set of candidates denoted as $\{\mathbf{c}_{m,1}^t, \cdots, \mathbf{c}_{m,S_m}^t\}$, where S_m is the number of generated candidate models. As shown in Figure 2, `pFedClub` will stop after substituting the third block $\mathbf{w}_{1,3}^t$ since all the block indexes in Group 4 for substituting $\mathbf{w}_{1,4}^t$ are not greater than 4, which is the minimum feasible block index. Thus, `pFedClub` will complete the generated candidates by directly using $\mathbf{w}_{1,4}^t$ as the forth block.

3.2.3 Personalized Model Selection

The final stage of `pFedClub` is to automatically select the "best" candidate teacher model from $\{\mathbf{c}_{m,1}^t, \cdots, \mathbf{c}_{m,S_m}^t\}$ for the m-th client. However, selecting such a model is non-trivial because $\mathbf{c}_{m,s}^t$ is a reassembled, incomplete model via block substitution, and the dimension sizes of different blocks may not be well-aligned.

Block Stitching. We complete each candidate model $\mathbf{c}_{m,s}^t$ using the network stitching technique [34]. We use a nonlinear activation function ReLU$(\cdot)$ on top of a linear layer, i.e., ReLU$(\mathbf{W}^\top \mathbf{X} + \mathbf{b})$ as the dimension mapping function.

Since the parameter values of $\{\mathbf{W}, \mathbf{b}\}$ in the stitching functions are **unknown**, it is essential to learn them with training data. Here, we propose to use a public dataset $\mathcal{D}_p$ to fine-tune the stitched candidate model $\mathbf{c'}_{m,s}^t$. Note that we fix all the parameters in $\mathbf{c}_{m,s}^t$ (denoted as $\boldsymbol{\theta}_{m,s}^*$) and only update $\{\mathbf{W}, \mathbf{b}\}$ in $\mathbf{c'}_{m,s}^t$ using the following loss if the public data are labeled:

$$\mathcal{L}_m = \frac{1}{|\mathcal{D}_p|} \sum_{i=1}^{|\mathcal{D}_p|} \text{CE}(\mathbf{c'}_{m,s}^t(\mathbf{x}_i; \mathbf{W}, \mathbf{b}, \boldsymbol{\theta}_{m,s}^*), \mathbf{y}_i), \tag{2}$$

where $|\mathcal{D}_p|$ denotes the number of data in the public dataset, CE$(\cdot, \cdot)$ means the cross-entropy loss, $\mathbf{c'}_{m,s}^t(\mathbf{x}_i; \mathbf{W}, \mathbf{b}, \boldsymbol{\theta}_{m,s}^*)$ presents the predicted label distribution for the data $\mathbf{x}_i$ by fixing the parameters $\boldsymbol{\theta}_{m,s}^*$, and $\mathbf{y}_i$ is the ground truth vector.

An unlabeled public dataset can be used to fine-tune the parameters $\{\mathbf{W}, \mathbf{b}\}$ using the normalized temperature-scaled cross-entropy loss [35] for a pair of data as follows:

$$\mathcal{L}_m^{i,j} = -\log \frac{\exp(\cos(\mathbf{c'}_{m,s}^t(\mathbf{x}_i), \mathbf{c'}_{m,s}^t(\mathbf{x}_j))/\tau)}{\sum_{k=1}^{2|\mathcal{D}_p|} \mathbb{1}_{[k \neq i]} \exp(\cos(\mathbf{c'}_{m,s}^t(\mathbf{x}_i), \mathbf{c'}_{m,s}^t(\mathbf{x}_k))/\tau)}, \tag{3}$$

where $\cos(\cdot, \cdot)$ is the cosine similarity, and τ is the hyperparamter. $\mathbf{x}_j$ is the augmentation of $\mathbf{x}_i$. We still fix the parameters $\boldsymbol{\theta}_{m,s}^*$ and learn $\{\mathbf{W}, \mathbf{b}\}$.

It is worth noting that block stitching operation will not significantly increase the number of parameters in the candidate model. Besides, some candidate models may be generated using Step 3: block completion. For those candidates, the number of newly added parameters is much smaller. Moreover, the limited number of parameters is helpful for the new candidate models to maintain more original model information. Finally, it makes model computation efficient and speeds up the model training.

Model Selection. Let $\hat{\mathbf{c}}_{m,s}^t$ denote the fine-tuned candidate model via Eq. (2). We then calculate the average cosine similarity scores on logits outputted by the original model $\mathbf{w}_m^t$ and its candidate model $\hat{\mathbf{c}}_{m,s}^t$ as follows:

$$\alpha_{m,s} = \frac{1}{|\mathcal{D}_p|} \sum_{i=1}^{|\mathcal{D}_p|} \cos(\boldsymbol{\beta}_m^t(\mathbf{x}_i), \hat{\boldsymbol{\beta}}_{m,s}^t(\mathbf{x}_i)), \tag{4}$$

where $\boldsymbol{\beta}_m^t(\mathbf{x}_i)$ and $\hat{\boldsymbol{\beta}}_{m,s}^t(\mathbf{x}_i)$ denote the logits of $\mathbf{x}_i$ outputted by the models $\mathbf{w}_m^t$ and $\hat{\mathbf{c}}_{m,s}^t$, respectively. Note that this is a forward propagation and does not need to train the models. Finally, the candidate model with the highest similarity scores in the set $\{\alpha_{m,1}, \cdots, \alpha_{m,S_m}\}$ will be selected as the final personalized teacher model $\mathbf{w}*_m^t$.

3.3 Client Update

When the m-th client is selected again at the j ($j > t$) communication round, the personalized model $\mathbf{w}*_m^t$ generated in the recent communication round will be distributed. Since the teacher model $\mathbf{w}*_m^t$

usually has a different network structure from the client model $\mathbf{w}_m^j$, we propose to use knowledge distillation to update the client model following [36] by optimizing the following loss:

$$\mathcal{L}_n^j = \frac{1}{|\mathcal{D}_n|} \sum_{i=1}^{|\mathcal{D}_n|} \left[\text{CE}(\mathbf{w}_m^j(\mathbf{x}_i), \mathbf{y}_i) + \lambda \text{KL}(\boldsymbol{\beta}_m^j(\mathbf{x}_i), \hat{\boldsymbol{\beta}}_m^t(\mathbf{x}_i)) \right], \tag{5}$$

where λ is a hyperparameter and $\text{KL}(\cdot, \cdot)$ is the Kullback–Leibler divergence.

4 Experiments

4.1 Experimental Setups

Datasets. In our experiments, we utilize three commonly used datasets to validate the performance of the proposed `pFedClub`, including MNIST[4], SVHN[5], and CIFAR-10[6]. We randomly divide the datasets into three parts: 72% for training, 20% for testing, and 8% as the public dataset. We test two data distribution settings in federated learning, i.e., IID and non-IID, following existing work [23]. For the **IID** setting, the training and testing data are randomly distributed to N clients. For the **non-IID** setting, each client randomly holds data belonging to two classes.

Baselines. The proposed `pFedClub` aims to aggregate **heterogeneous** client models to boost federated learning performance. Based on the condition of public datasets, we consider the following approaches as our baselines: (1) without using public datasets: HeteroFL [24] and FlexiFed [25]; (2) using labeled public data: FedMD [16], FedGH [19], and pFedHR [23]; and (3) using unlabeled public data: FCCL [17], FedKEMF [21], and pFedHR [23]. We also compare the proposed `pFedClub` with the general approaches to learning personalized federated learning models, which share the same structure for all clients. The **homogeneous** baselines include: FedAvg [1], FedProx [27], Per-FedAvg [28], PFedMe [29], PFedBayes [37], and pFedHR [23]. The details of all baselines can be found in Appendix **A**.

Client Model Deployment. In our experiments, we employ seven client models with different network structures, including MobileNetV1 [38], MobileNetV2 [39], MobileNetV3 [40], and four manually designed CNN models (denoted as CNN1 to CNN4). Each CNN model contains several convolutional blocks and fully connected blocks. The detailed model structures of the four models can be found in Appendix **B**. We set the number of clients $N = 50$ and the number of active clients $M = 5$ in each communication round. We propose three plans to distribute client models to validate the performance of the proposed `pFedClub` in different scenarios. First, we use all seven types of models and randomly send each model to a client (**Model Zoo I**). Specifically, CNN1 is randomly assigned to 8 clients, and each of the remaining six models is randomly assigned to 7 clients. Second, we distribute the three MobileNet family models to clients (**Model Zoo II**). In particular, we randomly assign MobileNetV1 to 16 clients, and MobileNetV2 and MobileNetV3 are randomly sent to the remaining 34 clients evenly. Finally, we use the four CNNs as the client models (**Model Zoo III**). Each CNN1, CNN2, and CNN3 model is randomly assigned to 12 clients. The remaining 14 clients will use CNN4 as the client model.

Implementation Details. We run all the experiments on NVIDIA A100 with CUDA version 12.0 on a Ubuntu 20.04.6 LTS server. All baselines and the proposed `pFedClub` are implemented in Pytorch 2.0.1. For the proposed `pFedClub` and baseline pFedHR, we set the number of clusters $K = 4$ following [23], and the local training epoch and the server finetuning epoch are equal to 10 and 3, respectively. The hyperparameter λ in Eq. (5) is 0.2. The hyperparameter τ in Eq. (3) is 0.07. We use Adam as the optimizer. The learning rate of the local client learning and the server fine-tuning learning rate equal 0.001. We use **average client accuracy** with three runs as the evaluation metric.

4.2 Performance Comparison

Heterogenous Model Aggregation. Table 1 lists the experimental results regarding the **three-run accuracy** of the proposed `pFedClub` and baselines. Note that HeteroFL and FlexiFed belong to

[4] `https://yann.lecun.com/exdb/mnist/`

[5] `http://ufldl.stanford.edu/housenumbers/`

[6] `https://www.cs.toronto.edu/~kriz/cifar.html`

https://doi.org/10.52202/079017-1733

Table 1: Performance (%) comparison with baselines under the heterogeneous settings.

Setting	Public Data	Dataset / Method	MNIST IID	MNIST Non-IID	SVHN IID	SVHN Non-IID	CIFAR-10 IID	CIFAR-10 Non-IID
Model Zoo I	Labeled	FedMD	91.12 ± 2.44	90.03 ± 2.98	76.22 ± 3.01	75.14 ± 3.75	66.38 ± 3.96	63.10 ± 4.75
		FedGH	92.76 ± 1.93	91.27 ± 2.21	78.41 ± 2.65	75.06 ± 2.87	71.22 ± 2.79	67.37 ± 3.06
		pFedHR	**94.67** ± 1.58	92.88 ± 1.10	81.59 ± 1.40	80.88 ± 1.92	73.21 ± 3.24	69.88 ± 3.45
		pFedClub	94.02 ± 1.41	**93.20** ± 0.85	**84.55** ± 1.17	**82.65** ± 1.56	**76.45** ± 2.87	**73.62** ± 3.01
	Unlabeled	FedKEMF	91.47 ± 1.87	90.60 ± 1.68	77.56 ± 2.47	74.23 ± 2.77	68.77 ± 2.54	65.09 ± 3.12
		FCCL	91.09 ± 2.05	90.21 ± 2.44	79.44 ± 2.33	75.28 ± 2.60	66.85 ± 2.66	64.76 ± 2.98
		pFedHR	92.15 ± 1.69	91.00 ± 1.73	80.66 ± 2.17	78.93 ± 2.55	72.06 ± 2.38	68.54 ± 2.47
		pFedClub	**93.72** ± 1.90	**92.77** ± 1.51	**83.94** ± 2.08	**81.76** ± 2.32	**75.86** ± 1.98	**72.87** ± 2.04
Model Zoo II	Labeled	FedMD	91.98 ± 0.76	92.01 ± 1.05	80.86 ± 1.26	77.53 ± 1.53	68.55 ± 1.89	63.74 ± 2.25
		FedGH	92.13 ± 1.32	91.14 ± 1.59	78.15 ± 1.50	75.47 ± 1.98	71.29 ± 1.77	68.60 ± 2.42
		pFedHR	93.51 ± 1.36	92.77 ± 1.24	82.33 ± 1.86	80.96 ± 1.90	73.60 ± 2.38	71.14 ± 2.76
		pFedClub	**93.62** ± 1.07	**93.11** ± 1.47	**84.25** ± 2.04	**82.47** ± 1.66	**76.02** ± 1.53	**73.15** ± 1.97
	Unlabeled	FedKEMF	92.61 ± 1.25	91.33 ± 1.70	79.62 ± 1.68	77.54 ± 2.04	69.11 ± 2.45	66.07 ± 2.88
		FCCL	92.77 ± 1.77	90.89 ± 1.90	81.88 ± 1.79	77.32 ± 1.68	68.02 ± 2.06	66.43 ± 2.96
		pFedHR	93.21 ± 1.45	92.77 ± 1.69	81.56 ± 1.50	79.68 ± 1.79	71.88 ± 1.75	69.94 ± 2.24
		pFedClub	**93.86** ± 1.34	**93.41** ± 1.85	**83.89** ± 1.22	**81.61** ± 1.47	**75.52** ± 1.39	**72.31** ± 1.98
Model Zoo III	✗	HeteroFL	92.48 ± 1.14	91.25 ± 1.45	80.57 ± 1.37	77.60 ± 1.68	71.08 ± 1.57	67.87 ± 1.66
		FlexiFed	91.08 ± 1.52	90.10 ± 1.66	80.69 ± 1.39	75.30 ± 1.62	68.09 ± 2.79	67.15 ± 2.88
	Labeled	FedMD	92.16 ± 1.32	91.37 ± 1.56	80.22 ± 1.59	76.14 ± 1.86	67.14 ± 1.67	63.50 ± 1.88
		FedGH	92.93 ± 1.52	91.44 ± 1.08	79.03 ± 1.44	75.28 ± 1.75	67.88 ± 1.75	70.77 ± 1.93
		pFedHR	92.25 ± 1.93	91.07 ± 1.64	81.88 ± 2.36	79.25 ± 1.71	72.45 ± 1.81	69.08 ± 1.95
		pFedClub	**93.24** ± 1.36	**92.66** ± 0.98	**82.69** ± 1.61	**81.21** ± 1.68	**74.88** ± 2.02	**71.94** ± 1.82
	Unlabeled	FedKEMF	92.78 ± 0.75	91.60 ± 1.03	78.88 ± 1.68	76.16 ± 1.77	68.04 ± 2.16	65.80 ± 2.84
		FCCL	92.65 ± 1.84	91.07 ± 1.92	80.32 ± 1.71	76.02 ± 1.82	67.16 ± 2.42	66.73 ± 2.68
		pFedHR	**93.84** ± 1.25	93.46 ± 1.50	81.76 ± 2.12	78.40 ± 2.50	71.74 ± 1.68	68.23 ± 1.79
		pFedClub	93.77 ± 1.16	**93.52** ± 1.37	**82.50** ± 1.25	**81.03** ± 1.49	**74.71** ± 1.40	**71.68** ± 1.59

the submodel training technique and require that each client model must be a part of the global model. Thus, they are only tested with Model Zoo 3. These results show that our proposed approach outperforms all the baselines under most settings, especially the more complicated datasets SVHN and CIFAR-10. Compared with the most recent work pFedHR, our proposed work pFedClub shows superior performance over that on SVHN and CIFAR-10 datasets under both the IID and non-IID settings. For pFedClub, the use of the labeled public data is able to boost the performance compared with the setting using unlabeled public data, which aligns with the observations in [23]. In addition, from Model Zoo III to Model Zoo I, the performance of pFedClub on the more complicated datasets, SVHN and CIFAR-10, improves with the increase of diversity of the model zoos.

Homogeneous Model Aggregation. The clients are assigned the same model structures to verify the effectiveness of pFedClub under the homogenous setting. We test the performance with CNN2 and MobileNetV2 under the non-IID setting and compare it with the state-of-the-art homogenous federated learning work. The results are shown in Table 2. Note that pFedHR and pFedClub are under the setting where the public data is labeled. We observe that the results of all the approaches on the MNIST dataset are relatively high, even with a simple CNN2 model, as classification on the MNIST dataset is an easy task. Besides, pFedClub outperforms state-of-the-art baselines on SVHN and CIFAR-10 datasets using the CNN2 model or MobileNetV2 model. These results demonstrate that pFedClub is also effective for the homogeneous setting.

4.3 Ablation Study

We conduct the ablation study to validate the effectiveness of each designed module in our proposed approach with Model Zoo III under the non-IID setting. In particular, we use the following four baselines: (1) $\texttt{pFedClub}_{max}$: in the anchor block selection stage (Step 1 in Section 3.2.2), we naively select the most similar block calculated by Eq. (1) for the first block, instead of randomly selecting one. (2) $\texttt{pFedClub}_{min}$: different from $\texttt{pFedClub}_{max}$, we use the block with the smallest index number as the substitution. The substituted block is either itself or other models' first block. (3) $\texttt{pFedClub}_{noc}$: we conduct the block search without using the order con-

Table 2: Performance (%) comparison with baselines under the homogeneous setting.

Model	Approach	MNIST	SVHN	CIFAR-10
CNN2	FedAvg	90.52	62.49	58.01
	FedProx	90.87	63.77	59.65
	Per-FedAvg	91.04	63.59	59.81
	PFedMe	91.79	64.27	60.14
	PFedBayes	92.54	63.19	60.08
	pFedHR	**92.62**	64.59	61.79
	pFedClub	92.18	**66.97**	**64.25**
MobileNetV2	FedAvg	92.07	79.42	62.13
	FedProx	92.85	80.33	63.60
	Per-FedAvg	92.62	82.45	70.88
	PFedMe	93.05	81.79	72.13
	PFedBayes	**93.71**	83.05	72.44
	pFedHR	93.15	83.88	73.65
	pFedClub	93.68	**85.26**	**74.97**

straint in Step 2; and (4) pFedClub$_{nbc}$: we do not conduct the block completion process (Step 3 in Section 3.2.2).

We report the results in Table 3 and provide the following observations: (1) Removal of any one module will cause the performance drop, thus demonstrating the individual contribution of each design in our proposed pFedClub. (2) When we use simple strategies (i.e., pFedClub$_{max}$ and pFedClub$_{min}$) in the anchor block selection, the drop in the performance is smaller than studies (i.e., pFedClub$_{noc}$ and pFedClub$_{nbc}$). It indicates that maintaining the block number and keeping the model completion matter more than the anchor block selection. Overall, each designed module has its own contribution, and the systematic combination of all the designed modules guarantees the effectiveness of our proposed pFedClub.

Table 3: Ablation study performance (%) comparison.

Dataset	SVHN		CIFAR-10	
Method	IID	Non-IID	IID	Non-IID
pFedClub$_{max}$	78.69	73.50	70.52	66.89
pFedClub$_{min}$	77.50	72.09	69.96	66.84
pFedClub$_{noc}$	65.26	61.08	62.19	58.14
pFedClub$_{nbc}$	63.01	59.47	61.38	56.02
pFedClub	**82.50**	**81.03**	**74.71**	**71.68**

4.4 Controllability Analysis

Experimental Setups. The major advantage of the proposed pFedClub is enabling the generation of controllable personalized candidate models. To clearly exhibit the insights, we use *Model Zoo I* as the heterogeneous model set, and each type of model is assigned to a corresponding client. Besides, each client will be mandatorily active during all the communication rounds. We use unlabeled public data for model training under the non-IID setting. To quantitatively evaluate the controllability of the personalized models generated by pFedClub, we propose to use the average of the model size change percentage over T communication rounds as the metric for each client, which is defined as follows:

$$\phi = \frac{1}{T}\sum_{t=1}^{T}\frac{|\mathbf{w}*_m^t| - |\mathbf{w}_m|}{|\mathbf{w}_m|}, \tag{6}$$

where $|\mathbf{w}*_m^t|$ denotes the parameter size of the personalized teacher model at round t, and $|\mathbf{w}_m|$ denotes the model parameter size of the original client model.

Model Comparison. Since the proposed pFedClub is a model reassembly-based framework, for a fair comparison, we choose to use pFedHR as the baseline. Besides, as mentioned in Section 3.2.2 Step 2, the proposed pFedClub is flexible to incorporate other constraints. In this experiment, we take the model size into consideration and denote the model as pFedClub$^+$. The reason is that for real-world FL applications, such as training a model with smart devices, their computational capability is limited. Larger personalized models may make these devices stop working. To facilitate flexible management of the generated model's size in pFedClub$^+$, we introduce an additional parameter, $\eta > -1$, which provides flexible controllability to decide the generated model size following the constraint: $|\mathbf{w}*_m^t| \leq (1+\eta)|\mathbf{w}_m^t|$. In this experiment, we set $\eta = 0.1$.

Results. Figure 3 illustrates the comparative performance with respect to accuracy and the model size controllability of pFedHR, pFedClub, and pFedClub$^+$ on the SVHN dataset under non-IID conditions. We observe that: **(1)** Both pFedClub and pFedClub$^+$ show a superior performance over pFedHR shown in Figure3(a). **(2)** As for the model size control, we can observe that pFedClub and pFedClub$^+$ both have better effectiveness over pFedHR in Figure 3(b). For example, For example, for client 1, the average received model parameter size is around 4.8 times as the original model for pFedHR and 1.5 times as the original model for pFedClub. For clients 5 - 7 using MobileNets, the average of the received personalized teacher model parameter size using pFedClub is smaller than that of the original model size ($\phi < 0$). However, ϕ is still a large positive number for pFedHR, which means the clients still receive the personalized teacher models larger than their original ones. **(3)** When comparing pFedClub with pFedClub$^+$, the latter shows a slight decrease in accuracy due to the rigorous model size constraint. Nonetheless, pFedClub$^+$ further refines the control over

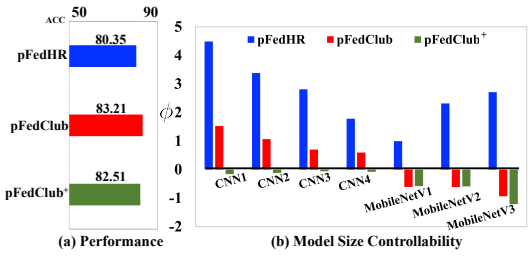

Figure 3: Controllability analysis.

https://doi.org/10.52202/079017-1733

the size of received personalized teacher models across all clients, ensuring they are not larger than than the original models ($\phi \leq 0$). This confirms the capability of both pFedClub and pFedClub$^+$ to effectively manage personalized model sizes, an essential feature for applications sensitive to computational resources. Additional details on model size comparisons between pFedClub and pFedHR across various communication rounds are provided in Appendix **C**.

4.5 Computational Cost Comparison

The proposed model can be treated as a fine-grained model reassembly technique, which uses controllable block-wise substitution to generate personalized candidates. In this experiment, we aim to compare the computational cost of the server between pFedClub and pFedHR using the computational time at each round on the SVHN dataset under the non-IID setting with Model Zoo III for 50 clients. We record the consumed computation time on the server side for each communication round. The results are shown in Figure 4. We can observe that the computation time at the server side of our approach pFedClub is generally shorter than that of the baseline pFedHR. Also, with the algorithm running with respect to the communication round, our approach becomes more consistent and stable compared with the significant shift of pFedHR. These results confirm that pFedClub is an efficient approach for heterogeneous model aggregation compared with pFedHR.

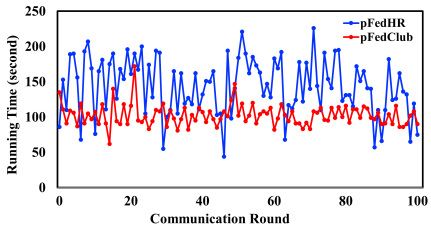

Figure 4: Server running time v.s. communication round.

4.6 System Running Time v.s. Accuracy

Except for controllability and computational costs, system running time is another key factor to evaluate the utility of the proposed pFedClub. Toward this end, we conduct an experiment to compare the consumed time to reach a fixed accuracy. The experimental setting is the same as the one that we described in Section 4.5. We take pFedHR as a baseline for comparison again. The results are shown in Figure 5. We can observe that the proposed approach pFedClub takes less time to achieve the target accuracy on the SVHN and CIFAR-10 datasets under the non-IID setting compared with pFedHR. These results demonstrate the effectiveness of the proposed pFedClub for the heterogeneous model aggregation in federated learning again.

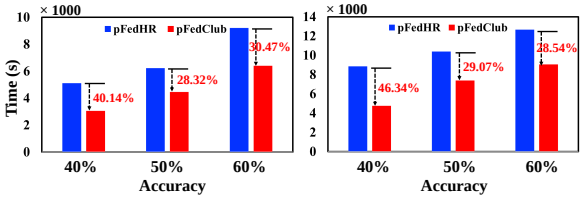

Figure 5: The consumed running time (in seconds) of models to achieve the target accuracy.

4.7 Extra Experimental Results

To validate the **model scalability** of pFedClub, we conduct the experiments by considering different numbers and different active ratios of clients, and the results are shown in Table 4 in Appendix **D**. Besides, in our model design, there is a key parameter K used in Section 3.2.1. We validate the sensitivity of the selection of K, and the results are listed in Table 5 in Appendix **E**.

5 Conclusion

This paper introduces pFedClub designed to revolutionize personalized federated learning. By leveraging a unique network block substitution method, pFedClub effectively creates tailored and functionally analogous personalized models for individual clients. Moreover, pFedClub is highly adaptable and can easily incorporate other types of constraints to achieve application-driven personalized model generation. Our experimental evaluations, conducted on three diverse datasets under both IID and non-IID settings, unequivocally validate the efficacy of pFedClub in the domain of heterogeneous model aggregation for federated learning. The results affirm the accuracy, efficiency, and flexibility of our proposed method, demonstrating its potential for real-world applications.

https://doi.org/10.52202/079017-1733

Acknowledgements This work is partially supported by the National Science Foundation under Grant No. 2348541 and 2238275.

References

[1] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[2] Liang Qu, Ningzhi Tang, Ruiqi Zheng, Quoc Viet Hung Nguyen, Zi Huang, Yuhui Shi, and Hongzhi Yin. Semi-decentralized federated ego graph learning for recommendation. *arXiv preprint arXiv:2302.10900*, 2023.

[3] Chung-ju Huang, Leye Wang, and Xiao Han. Vertical federated knowledge transfer via representation distillation for healthcare collaboration networks. In *Proceedings of the ACM Web Conference 2023*, pages 4188–4199, 2023.

[4] Kaibin Wang, Qiang He, Feifei Chen, Hai Jin, and Yun Yang. Fededge: Accelerating edge-assisted federated learning. In *Proceedings of the ACM Web Conference 2023*, pages 2895–2904, 2023.

[5] Bingyan Liu, Yifeng Cai, Hongzhe Bi, Ziqi Zhang, Ding Li, Yao Guo, and Xiangqun Chen. Beyond fine-tuning: Efficient and effective fed-tuning for mobile/web users. In *Proceedings of the ACM Web Conference 2023*, pages 2863–2873, 2023.

[6] Xiangrong Zhu, Guangyao Li, and Wei Hu. Heterogeneous federated knowledge graph embedding learning and unlearning. In *Proceedings of the ACM Web Conference 2023*, pages 2444–2454, 2023.

[7] Yuke Hu, Wei Liang, Ruofan Wu, Kai Xiao, Weiqiang Wang, Xiaochen Li, Jinfei Liu, and Zhan Qin. Quantifying and defending against privacy threats on federated knowledge graph embedding. In *Proceedings of the ACM Web Conference 2023*, pages 2306–2317, 2023.

[8] Tao Guo, Song Guo, and Junxiao Wang. pfedprompt: Learning personalized prompt for vision-language models in federated learning. In *Proceedings of the ACM Web Conference 2023*, pages 1364–1374, 2023.

[9] Han Xie, Li Xiong, and Carl Yang. Federated node classification over graphs with latent link-type heterogeneity. In *Proceedings of the ACM Web Conference 2023*, pages 556–566, 2023.

[10] Lars Heling and Maribel Acosta. Federated sparql query processing over heterogeneous linked data fragments. In *Proceedings of the ACM Web Conference 2022*, pages 1047–1057, 2022.

[11] Haoyu Wang, Handong Zhao, Yaqing Wang, Tong Yu, Jiuxiang Gu, and Jing Gao. Fedkc: Federated knowledge composition for multilingual natural language understanding. In *Proceedings of the ACM Web Conference 2022*, pages 1839–1850, 2022.

[12] Chengxu Yang, Qipeng Wang, Mengwei Xu, Zhenpeng Chen, Kaigui Bian, Yunxin Liu, and Xuanzhe Liu. Characterizing impacts of heterogeneity in federated learning upon large-scale smartphone data. In *Proceedings of the Web Conference 2021*, pages 935–946, 2021.

[13] Jinze Wu, Qi Liu, Zhenya Huang, Yuting Ning, Hao Wang, Enhong Chen, Jinfeng Yi, and Bowen Zhou. Hierarchical personalized federated learning for user modeling. In *Proceedings of the Web Conference 2021*, pages 957–968, 2021.

[14] Chenglin Li, Di Niu, Bei Jiang, Xiao Zuo, and Jianming Yang. Meta-har: Federated representation learning for human activity recognition. In *Proceedings of the web conference 2021*, pages 912–922, 2021.

[15] Weiming Zhuang, Jian Xu, Chen Chen, Jingtao Li, and Lingjuan Lyu. Coala: A practical and vision-centric federated learning platform. *arXiv preprint arXiv:2407.16560*, 2024.

 https://doi.org/10.52202/079017-1733

[16] Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*, 2019.

[17] Wenke Huang, Mang Ye, and Bo Du. Learn from others and be yourself in heterogeneous federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10143–10153, 2022.

[18] Lichao Sun and Lingjuan Lyu. Federated model distillation with noise-free differential privacy. *arXiv preprint arXiv:2009.05537*, 2020.

[19] Liping Yi, Gang Wang, Xiaoguang Liu, Zhuan Shi, and Han Yu. Fedgh: Heterogeneous federated learning with generalized global header. *arXiv preprint arXiv:2303.13137*, 2023.

[20] Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33:2351–2363, 2020.

[21] Sixing Yu, Wei Qian, and Ali Jannesari. Resource-aware federated learning using knowledge extraction and multi-model fusion. *arXiv preprint arXiv:2208.07978*, 2022.

[22] Jiaqi Wang, Shenglai Zeng, Zewei Long, Yaqing Wang, Houping Xiao, and Fenglong Ma. Knowledge-enhanced semi-supervised federated learning for aggregating heterogeneous lightweight clients in iot. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pages 496–504. SIAM, 2023.

[23] Jiaqi Wang, Xingyi Yang, Suhan Cui, Liwei Che, Lingjuan Lyu, Dongkuan Xu, and Fenglong Ma. Towards personalized federated learning via heterogeneous model reassembly. *Advances in Neural Information Processing Systems*, 2023.

[24] Enmao Diao, Jie Ding, and Vahid Tarokh. Heterofl: Computation and communication efficient federated learning for heterogeneous clients. In *International Conference on Learning Representations*, 2020.

[25] Kaibin Wang, Qiang He, Feifei Chen, Chunyang Chen, Faliang Huang, Hai Jin, and Yun Yang. Flexifed: Personalized federated learning for edge clients with heterogeneous model architectures. In *Proceedings of the ACM Web Conference 2023*, pages 2979–2990, 2023.

[26] Lingjuan Lyu, Han Yu, Xingjun Ma, Chen Chen, Lichao Sun, Jun Zhao, Qiang Yang, and S Yu Philip. Privacy and robustness in federated learning: Attacks and defenses. *IEEE transactions on neural networks and learning systems*, 2022.

[27] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

[28] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *Advances in Neural Information Processing Systems*, 33:3557–3568, 2020.

[29] Canh T Dinh, Nguyen Tran, and Josh Nguyen. Personalized federated learning with moreau envelopes. *Advances in Neural Information Processing Systems*, 33:21394–21405, 2020.

[30] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[31] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[32] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning*, pages 3519–3529. PMLR, 2019.

[33] Xingyi Yang, Daquan Zhou, Songhua Liu, Jingwen Ye, and Xinchao Wang. Deep model reassembly. *Advances in neural information processing systems*, 35:25739–25753, 2022.

[34] Zizheng Pan, Jianfei Cai, and Bohan Zhuang. Stitchable neural networks. *arXiv preprint arXiv:2302.06586*, 2023.

[35] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[36] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4320–4328, 2018.

[37] Xu Zhang, Yinchuan Li, Wenpeng Li, Kaiyang Guo, and Yunfeng Shao. Personalized federated learning via variational bayesian inference. In *International Conference on Machine Learning*, pages 26293–26310. PMLR, 2022.

[38] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[39] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[40] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019.

 https://doi.org/10.52202/079017-1733

Appendix

A. Baselines

In the heterogeneous experiments, we use the following approaches as baselines:
(1) Without using public datasets:

- HeteroFL [24]: Local clients' models are required to belong to the same model class and work together to produce one single global inference mode. Specifically, they revise the batch normalization, conduct a pre-activity scaling, and design a masked loss to solve this research problem.
- FlexiFed [25]: The common parts of the clients' models work together, and the different parts work separately to keep the system updated. They provide a basic-common strategy, cluster-common strategy, and max-common strategy to conduct the heterogeneous model aggregation under the setting.

(2) Using labeled public data:

- FedMD [16]: It uses transfer learning and knowledge distillation with the labeled public data on the server side. Specifically, each client needs to train the local model on both the public dataset and the private dataset. Then, clients upload the class scores on the public dataset to the server, and the server calculates the consensus to send it back for a local update.
- FedGH [19]: The clients have their own feature extractors and share the homogeneous global header. The clients train local models on their local data and upload the representation and label for each label back to the server for the global header update. Then, the clients replace their own headers with the global one for inference.
- pFedHR [23]: This approach utilizes the model disassembly techniques to decompose local models into layers. The server composes the layers back and tunes the models while stitching the layers using the public datasets.

(3) Using unlabeled public data:

- FCCL [17]: FCCL leverages the unlabeled public data and averages the logits from local clients. The approach utilizes a consensus logit to guide the local training.
- FedKEMF [21]: This approach aggregates knowledge from local models and distills it into global knowledge via knowledge distillation. It uses mutual learning to personalize the models on the server side with the unlabeled public data;
- pFedHR [23]: This approach is also able to use the unlabeled public data to tune the stitching layers of the candidate models.

The following baselines are used in the homogeneous experiments:

- FedAvg [1]: It is the vanilla version of federated learning, which averages the model parameters from the local clients;
- FedProx [27]: It adds the proximal term to the local model training based on FedAvg;
- Per-FedAvg [28]: The MAML framework is proposed based on meta learning;
- PFedMe [29]: It uses regularized loss and decouples the personalization problem into a bi-level optimization;
- PFedBayes [37]: It proposes an algorithm to take consideration of the global distribution while conducting local model training;
- pFedHR [23]: It generates personalized models by model decomposition and composition for local clients to guide local model training.

B. CNN Strutures

In our experiments, we have 4 CNN models with different complexity. The details are shown as follows. In each convolutional NN sequential block, there is 1 convolutional layer, a max pooling layer, and a ReLU function.

CNN1: *Cov1:{Conv2d (kernel size = 5) → ReLU → MaxPool2D (kernel size = 2, stride = 2) } → Cov2:{Conv2d (kernel size = 5) → ReLU → MaxPool2D (kernel size = 2, stride = 2) } → FC1:{Linear → ReLU} → Dropout→ FC2:Linear.*

CNN2: *Cov1:{Conv2d (kernel size = 5) → ReLU → MaxPool2D (kernel size = 2, stride = 2) } → Cov2:{Conv2d (kernel size = 5) → ReLU → MaxPool2D (kernel size = 2, stride = 2) } → Cov3:{Conv2d (kernel size = 5) → ReLU} → FC1:{Linear → ReLU} → Dropout→ FC2:Linear.*

CNN3: *Cov1:{Conv2d (kernel size = 5) → ReLU → MaxPool2D (kernel size = 2, stride = 2) } → Cov2:{Conv2d (kernel size = 5) → ReLU → MaxPool2D (kernel size = 2, stride = 2) } → Cov3:{Conv2d (kernel size = 5) → ReLU} → Cov4:{Conv2d (kernel size = 5) → ReLU → MaxPool2D (kernel size = 2, stride = 2) } → Cov5:{Conv2d (kernel size = 5) → ReLU} → FC1:{Linear → ReLU → Dropout} → FC2:{Linear → ReLU}→ FC3:{Linear → ReLU}→FC4:Linear.*

CNN4: *Cov1:{Conv2d (kernel size = 5) → BatchNorm2d → ReLU} → Cov2:{Conv2d (kernel size = 3) → ReLU → MaxPool2D (kernel size = 2, stride = 2) } → Cov3:{Conv2d (kernel size = 3) → BatchNorm2d → ReLU} → Cov4:{Conv2d (kernel size = 5) → ReLU → MaxPool2D (kernel size = 2, stride = 2) →* Dropout *} → Cov5:{Conv2d (kernel size = 3) → BatchNorm2d → ReLU } → Cov6:{Conv2d (kernel size = 3) → ReLU → MaxPool2D (kernel size = 2, stride = 2) } → FC1:{Linear → ReLU → Dropout} → FC2:{Linear → ReLU}→ FC3:{Linear → ReLU}→FC4:Linear.*

C. Generated Model Size Comparison

In Section 4.4, we have validated that the size of the generated personalized models by the proposed `pFedClub` is much smaller than that produced by pFedHR. Figure 3 shows the relatively average change in model size. In this experiment, we aim to show a detailed comparison of model size changes at each communication round for each type of heterogeneous model.

The results are shown in Figure 6. We can observe that our approach `pFedClub` conducts better control over the generated models compared with pFedHR in almost every communication round, especially for the client with the MobileNet models, where the size of our generalized model parameters is always smaller than the original one ($\phi < 0$).

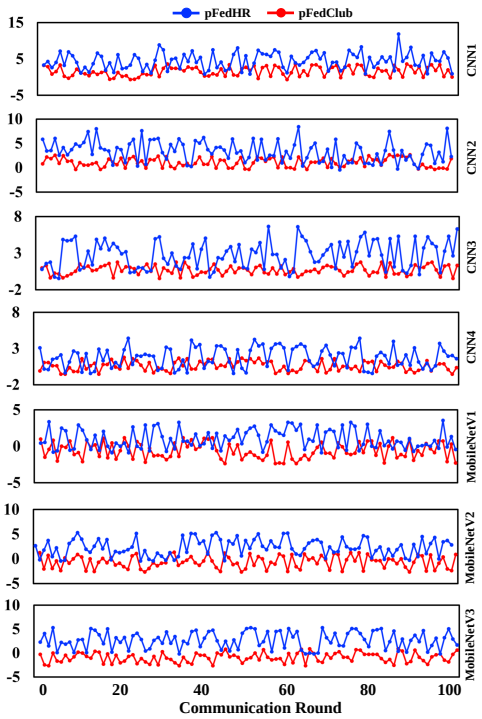

Figure 6: Model size controllability compared with pFedHR with respect to the communication round.

D. Model Scalability Analysis

Besides, model scalability is important in federated learning systems. To validate the scalability of `pFedClub`, we conduct the following experiments by considering different numbers and different active ratios of clients. The results are shown in Table 4.

Given the different settings of the different numbers of clients and active ratios, we can observe that the performance changes according to our expectations. Moreover, given a fixed active ratio, the performance has a slight decrease with the increase in the client number. One possible reason is each client will have a smaller number of training data when the client number increases given a certain number of training data. The change in the local model performance will harm the performance of the whole system in a certain range. Furthermore, given a fixed number of clients, the increase in the active ratio will slightly boost the performance because it enables more clients to contribute to the update process with their local training at each communication round. The experiment results demonstrate the scalability of our proposed approach, considering the change in the client number and active ratio.

https://doi.org/10.52202/079017-1733

Table 4: Scalbility of pFedClub with different numbers of clients and different active ratios on the SVHN dataset.

Data	**IID**			**non-IID**		
Client	**Active Ratio**			**Active Ratio**		
Number	10%	20%	30%	10%	20%	30%
30	82.84	84.68	85.02	81.23	82.58	84.47
50	82.50	84.02	84.89	81.03	82.41	83.15
100	79.26	82.55	83.97	77.06	79.43	80.22

Table 5: Hyperparameter study of the number of clusters. The performance (%) of pFedClub with different values of K on the SVHN and CIFAR-10 datasets.

Public Dataset	**Dataset**	**SVHN**		**CIFAR-10**	
	Cluster	**IID**	**Non-IID**	**IID**	**Non-IID**
Unlabeled	3	80.45	78.21	72.02	68.04
	4	82.50	81.03	74.71	71.68
	5	82.66	81.17	75.26	71.89
Labeled	3	80.76	79.88	73.07	68.95
	4	82.69	81.21	74.88	71.94
	5	82.77	81.35	75.89	72.38

E. Hyperparameter Study

In our design, K is the number of the groups based on the function-wise clustering with K-means. In this experiment, we aim to study how the hyperparameter K affects the performance. We maintain the experimental setting in Sections 4.5 and 4.6. The results are shown in Table 5. We can observe that the performance of pFedClub will slightly increase with the increase of K. In the main experiments, we follow [23] and set $K = 4$. When $K = 5$, the performance can increase slightly. One possible reason is that a large K is able to produce more specific function-based groups and further identify the function of the blocks more accurately. We can also observe that the performance is generally stable with the change of hyperparameter K in a certain range.

F. Limitations and Broader Impacts

This work focuses on the controllable personalized model generation for the heterogeneous federated learning setting. Although the proposed pFedClub outperforms baselines and is able to generate size-controllable client models, it still has several limitations. First, the proposed controllable model searching and reproduction (CMSR) algorithm is a heuristic algorithm that is designed based on intuitions. Thus, the results may be suboptimal. Second, in the system running time experiments, we have demonstrated that the proposed pFedClub can reduce the learning time compared with the state-of-the-art model pFedHR. However, compared with traditional averaging methods, the running time on the server is still much longer. We plan to design a more efficient algorithm for the server update. Finally, in the experiments, we only test the image classification task, which limits the efficacy test on other tasks. We will test more diverse tasks with the proposed pFedClub in the future.

This research significantly enhances federated learning by enabling efficient aggregation of heterogeneous models without compromising data privacy. This approach not only improves user experience across diverse sectors by providing tailored solutions but also supports sustainable computing practices of large-scale machine learning operations.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our abstract and introduction reflect the paper's contributions and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitation in the appendix.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

https://doi.org/10.52202/079017-1733

Justification: Our paper has no theoretical result.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer:[Yes]

   Justification: We provide the codes and experiment details to reproduce the results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer:[Yes]

Justification: We provide the code in the supplementary file and we use the open datasets.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We specify the details as asked.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We run experiment multiple times.We provide the mean and the STD.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

https://doi.org/10.52202/079017-1733

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the details in **Implementation Details** in Sec 4.1. We report the running time in Sec 4.5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We provide the broader impacts in the appendix.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the original paper, code package, and dataset.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

https://doi.org/10.52202/079017-1733

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide the details of the designed model and provide the codes in the supplementary file.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.