Improving Learning of New Diseases through Knowledge-Enhanced Initialization for Federated Adapter Tuning

Danni Peng, *Member, IEEE*, Yuan Wang, *Member, IEEE*, Kangning Cai, Peiyan Ning, Jiming Xu, Yong Liu, *Senior Member, IEEE*, Rick Siow Mong Goh, *Senior Member, IEEE*, Qingsong Wei, *Senior Member, IEEE*, and Huazhu Fu, *Senior Member, IEEE*

***Abstract*—In healthcare, federated learning (FL) is a widely adopted framework that enables privacy-preserving collaboration among medical institutions. With large foundation models (FMs) demonstrating impressive capabilities, using FMs in FL through cost-efficient adapter tuning has become a popular approach. Given the rapidly evolving healthcare environment, it is crucial for individual clients to quickly adapt to new tasks or diseases by tuning adapters while drawing upon past experiences. In this work, we introduce Federated Knowledge-Enhanced Initialization (FedKEI), a novel framework that leverages cross-client and cross-task transfer from past knowledge to generate informed initializations for learning new tasks with adapters. FedKEI begins with a global clustering process at the server to generalize knowledge across tasks, followed by the optimization of aggregation weights across clusters (inter-cluster weights) and within each cluster (intra-cluster weights) to personalize knowledge transfer for each new task. To facilitate more effective learning of the inter- and intra-cluster weights, we adopt a bi-level optimization scheme that collaboratively learns the global intra-cluster weights across clients and optimizes the local inter-cluster weights toward each client's task objective. Extensive experiments on three benchmark datasets of different modalities, including dermatology, chest X-rays, and retinal OCT, demonstrate FedKEI's advantage in adapting to new diseases compared to state-of-the-art methods.**

***Index Terms*—Federated Learning, New Disease Adaptation, Foundation Model Adapter Tuning, Knowledge Transfer, Learned Initialization**

I. Introduction

Manuscript submitted 2 Dec 2024. This Research is supported by the RIE2025 Industry Alignment Fund – Industry Collaboration Project (IAF-ICP) (Award No: I2301E0020) and Japan-Singapore Joint Call: Japan Science and Technology Agency (JST) and Agency for Science, Technology and Research (A*STAR) 2024 (Award No: R24I6IR141), administered by A*STAR *(Corresponding author: Qingsong Wei (wei_qingsong@ihpc.a-star.edu.sg))*.

Danni Peng, Yuan Wang, Yong Liu, Rick Siow Mong Goh, Qingsong Wei, and Huazhu Fu are with the Institute of High Performance Computing (IHPC), Agency for Science, Technology and Research (A*STAR), Singapore 138632 (e-mail: dannip@ihpc.a-star.edu.sg; wang_yuan@ihpc.a-star.edu.sg; liuyong@ihpc.a-star.edu.sg; gohsm@ihpc.a-star.edu.sg; wei_qingsong@ihpc.a-star.edu.sg; hzfu@ieee.org).

Kangning Cai, Peiyan Ning, and Jiming Xu are with EVYD Technology (e-mail: kangning.cai@evydtech.com; peiyan.ning@evydtech.com; jiming.xu@evydtech.com).

FEDERATED learning (FL) has gained traction in healthcare by enabling collaborative model training across institutions without sharing sensitive data [1]. With large foundation models (FMs) demonstrating strong performance across various tasks [2], [3], integrating FMs into FL presents new opportunities for medical imaging [4]. A common approach involves fine-tuning pre-trained FMs for downstream tasks in FL [5], [6]. To mitigate the substantial computation and communication overhead of full fine-tuning, recent methods adopt federated adapter tuning [7], which updates and transmits only lightweight adapters between the server and clients [8]. This approach enables efficient use of FMs in FL, making it particularly suitable for resource-constrained clients such as small clinics [9], [10].

In dynamic healthcare environments, FL clients often face previously unseen diseases, such as rare conditions or emerging outbreaks like COVID-19 [11], [12]. To remain effective, they must continually adapt to new diagnostic tasks while leveraging prior knowledge. Federated continual learning (FCL) supports this by enabling clients to learn from evolving local task streams without sharing sensitive data [13]. Traditional FCL methods focus on mitigating forgetting within a single shared model, under the constraint that assigning a separate model per task would incur large storage costs [14], [15]. However, this constraint is eased in federated adapter tuning, which fine-tunes lightweight adapters on a fixed pre-trained FM [7], [8]. Since adapters are small (e.g.,$<1\%$ of ViT with LoRA), assigning one per task is feasible, shifting the focus from "unforgetting" to knowledge transfer for better new task adaptation [16]. We adopt this setup by assigning a unique adapter per task and addressing the underexplored challenge of improving adaptation to new tasks.

Transferring knowledge from related medical conditions is crucial for quickly adapting to new diseases. For example, insights from SARS can inform COVID-19 treatment due to their shared coronavirus origin [17], [18]. Yet, individual hospitals often have limited, heterogeneous data. FCL enables both temporal transfer (from past tasks) and spatial transfer (from other clients), broadening each client's knowledge base. As shown in Figure 1, traditional FL (top left) transfers knowledge spatially across clients, and CL (bottom left) transfers knowledge temporally across tasks. Our FCL approach

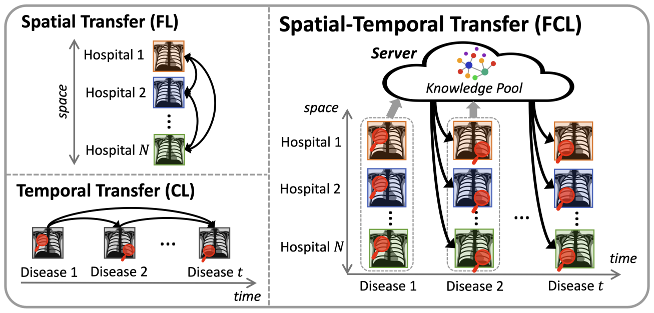

Fig. 1. Illustration of spatial-temporal knowledge transfer in FCL. Left: traditional FL enables spatial transfer across clients (top), and CL enables temporal transfer across tasks (bottom). Right: our FCL framework unifies both by pooling task-specific modules from all clients at the server. When a new disease arises, each client adapts more effectively by drawing on diverse knowledge from this shared pool.

(right) combines both: at each step, clients upload task-specific modules (e.g., adapters) to a shared knowledge pool. When encountering a new disease, a client retrieves relevant knowledge—including its own and others' past experiences—for better adaptation. For example, when diagnosing Atelectasis, a hospital lacking sufficient prior data may benefit from others' clearer or more diverse cases of related conditions (e.g., Effusion or Pneumothorax), improving learning via shared visual patterns like lung collapse or opacity [19].

In this work, we aim to explore the potential of leveraging spatial-temporal transfer for learning new diseases with FM adapter tuning. To this end, we propose a novel FCL framework, termed **Fed**erated **K**nowledge-**E**nhanced **I**nitialization (FedKEI), which selectively transfers valuable knowledge across clients and tasks to construct informed initializations for the adapters to learn new tasks. Specifically, the adapter and head tuned locally for each task (referred to as the task-specific modules) are sent to server, where a pool of task-specific modules is accumulated over time. Our FedKEI performs two key steps to effectively extract and transfer knowledge from the stored task-specific modules to improve new task learning: (1) global clustering of task-specific modules to generalize knowledge across tasks and clients, and (2) learning the aggregation weights across clusters (i.e., inter-cluster weights) and within each cluster (i.e., intra-cluster weights)—collectively referred to as the bi-level weights—to aggregate the task-specific modules. The resulting aggregated modules are then used to initialize the adapter and head for fine-tuning on new tasks. By this means, knowledge from previous tasks is selectively transferred and integrated into informed initializations to enhance new task learning.

To facilitate effective learning of the bi-level aggregation weights, we adopt a bi-level optimization scheme where intra-cluster weights are learned at the server to enhance inter-cluster weight learning at the client. Inspired by meta-learning [20], this approach ensures that the cluster-specific modules, formed using shared intra-cluster weights, are more informative and capable of supporting the learning of local inter-cluster weights, leading to more effective initializations.

Overall, we summarize our main contributions as follows:

- We propose a novel FCL framework, FedKEI, for improving adaptation to new diseases. The framework focuses on integrating the spatial-temporal knowledge transfer into a more informed initialization for FM adapter tuning.
- In our framework, we employ global clustering and bi-level aggregation learning to achieve effective knowledge transfer, where the former generalizes knowledge across different tasks and clients, and the latter selectively personalizes knowledge transfer for each task.
- To facilitate more effective bi-level aggregation weight learning, we introduce a novel bi-level optimization scheme to learn the global intra-cluster weights such that the subsequent learning of local inter-cluster weights is enhanced for generating effective initializations.
- We extensively evaluate FedKEI on three large-scale FCL datasets across different modalities and show that it achieves better performance in adapting to new diseases compared to state-of-the-art methods.

II. Related Work

A. Task Adaptation in Federated Learning

FL has recently gained popularity for supporting privacy-preserving collaboration among clients [1], [21]. Traditional FL algorithms, such as FedAvg [22], FedProx [23], and SCAFFOLD [24], focus on learning a global model that performs well across all clients. While FedProx and SCAFFOLD introduce mechanisms to mitigate data heterogeneity—such as proximal regularization and control variates—they still follow a one-model-fits-all paradigm. This paradigm often falls short in scenarios with severe client heterogeneity and dynamic task distributions. Personalized federated learning (PFL) addresses this by developing personalized models tailored to each client's local objective, including meta-learning–based approaches, which learn a global initialization to facilitate client-side adaptation [25], and personalized aggregation approaches, which learn client-specific aggregation weights optimized towards local objective [26], [27]. However, most existing PFL methods assume static client objectives and struggle with evolving tasks or shifting data, limiting their effectiveness in dynamic settings requiring continual knowledge transfer.

FCL is a recent approach focused on continual task learning at clients, primarily aiming to mitigate forgetting. FCL methods can be broadly categorized into three classes. First, regularization-based approaches retain knowledge of previous tasks by constraining model updates, through explicit regularization terms or via knowledge distillation [14], [15]. Second, replay–based methods store raw samples from previous tasks or generate pseudo-examples to be used alongside new task data during training [28]. Third, architecture-based approaches assign isolated model parameters to different tasks to preserve past knowledge [13]. Recently, [29] explicitly identified the issue of spatial-temporal catastrophic forgetting in FCL and addressed it with a gradient-free approach. Despite recent developments, most FCL methods mainly focus on preserving performance on past tasks and preventing forgetting, with limited exploration of how knowledge from previous tasks can be harnessed to improve adaptation to new ones.

Another related branch is federated domain generalization (FDG) [30], which aims to develop federated models that generalize well to unseen domains or tasks by learning domain-invariant features from clients with distribution shifts. ELCFS [31] focuses on client-side learning by encouraging unbiased local training through amplitude spectrum transfer across clients, enriching local distributions for more effective domain-invariant feature extraction. In contrast, FedGA [32] improves generalizability through server-side aggregation, adjusting client weights based on their generalization gaps. Caldarola et al. [33] proposes a hybrid strategy, applying sharpness-aware minimization for local training at the client side and stochastic weight averaging for model aggregation at the server. While effective, these methods address only inter-client domain shifts, assuming static local distributions. Moreover, FDG develops a generalized global model without personalization for heterogeneous local tasks. In contrast, our framework transfers knowledge across both clients and time, tailored specifically to improve learning for individual tasks.

B. FM Adapter Tuning in Federated Learning

With the rise of powerful FMs [2], [3], there has been growing interest in integrating FMs into FL [4]. Instead of full fine-tuning, tuning only the lightweight adapters provides a cost-efficient way for leveraging large FMs in FL, incurring only minimal client computation and communication [8].

Among the earliest works, FedCLIP [9] demonstrates that adapter tuning outperforms full fine-tuning in FL by better retaining the rich priors of pre-trained CLIP [3], benefiting data-scarce local tasks. Building on this, FACMIC [34] applies adapter tuning of CLIP in the medical domain, incorporating a domain adaptation loss to mitigate client distribution shifts. In contrast, FLoRA [35] and FFA-LoRA [36] focus on integrating and aggregating the existing LoRA adapters on pre-trained LLMs. FLoRA introduces a stacking-based aggregation strategy, while FFA-LoRA proposes to fine-tune only the zero-initialized matrix of LoRA to address noise during convergence. While these studies mainly concern with better implementation of adapter tuning in standard FL, our work focuses on the continual setting, leveraging knowledge transfer to improve adaptation to new tasks.

C. Federated and Continual Learning in Medical Imaging

FL has gained traction in medical imaging, with various strategies to address data heterogeneity and client-side adaptation. Feng et al. [37] propose a shared encoder with client-specific decoders for MR image reconstruction. Xu et al. [38] introduce an ensemble framework combining global and personalized models with a model selector to handle client shift. Li et al. [39] apply domain adaptation with noise-augmented fMRI data and a domain discriminator to reduce inter-client distribution gaps. ELCFS [31], a federated domain generalization method designed for medical image segmentation, employs a boundary-oriented episodic learning scheme to simulate domain shifts at training.

CL has also been explored in healthcare. Wu et al. [40] mitigate forgetting in class-incremental nuclei segmentation through regularization. Amrollahi et al. [41] enable center-incremental sepsis prediction using a hybrid of weight regularization and representation replay. Zheng et al. [42] propose asynchronous federated continual learning with reinforcement learning and selective experience replay for modality-incremental landmark localization in 3D imaging.

Despite recent advances, few works unify FL and CL to enable spatial-temporal knowledge transfer across clients and tasks in the medical domain. Additionally, most efforts focus on mitigating forgetting, with limited emphasis on improving new task adaptation, especially for FM adapter tuning.

III. METHODOLOGY

A. Preliminaries

1) Problem Setup: A standard FL setup involves N clients and a central server, where each client $i \in [N]$ owns a local dataset $\mathcal{D}_i$. In real-world scenarios, the local data distribution at each client is not static, i.e., each client i continuously encounters a local stream of tasks $\{\mathcal{T}_i^1, \mathcal{T}_i^2, \cdots\}$, where each task $\mathcal{T}_i^t \in \mathcal{D}_i$ is a subset of $\mathcal{D}_i$. To study the problem of new disease learning, we define each task to have a label set (i.e., disease classes) that differs from all the previously seen tasks of the same client, i.e., $\mathcal{Y}_i^t \neq \mathcal{Y}_i^j, \forall j < t$, where $\mathcal{Y}_i^t$ denotes the label set of $\mathcal{T}_i^t$ of client i. To reflect realistic medical scenarios such as emerging diseases, we assume that clients observe largely similar tasks at a given time t.

In this work, we focus on improving adaptation to new tasks by leveraging knowledge from past tasks. As sharing clients' local data is prohibited in FL, we instead utilize the task-specific models learned from previous tasks to guide current learning. Formally, at time t, given a new task $\mathcal{T}_i^t$ for each client i, we aim to learn task-specific models $\{\theta_i^t\}_{i=1}^N$ by utilizing the set of prior task-specific models $\Theta^{t-1} = \{\theta_i^j \mid i \in [N], j \in [t-1]\}$ obtained from all clients. The learning objective is defined as follows:

$$\min_{\{\theta_1^t,\cdots,\theta_N^t\}} \sum_{i=1}^N \mathcal{L}_{\mathcal{T}_i^t}(\theta_i^t; \Theta^{t-1}), \tag{1}$$

where $\mathcal{L}_{\mathcal{T}_i^t}$ denotes a classification loss (e.g., cross-entropy) on the current task $\mathcal{T}_i^t$, augmented by knowledge distilled or transferred from relevant past models in Θ^{t-1}.

2) Federated Adapter Tuning for New Task Adaptation: Adopting adapter tuning in FL offers a computation- and communication-efficient way for clients to harness the power of FMs when adapting to new tasks. Let $F^*(\cdot)$ represent a fixed, pre-trained FM backbone (e.g., ViT [2]) retained locally at each client. We attach to it an adapter module $g_\omega(\cdot)$, parameterized by ω (e.g., LoRA [8]). Together, the backbone and adapter form a composite feature extractor: $f_\omega(\cdot) = F^*(g_\omega(\cdot))$. Each client also maintains a classification head $h_\phi(\cdot)$, parameterized by ϕ. Thus, the learnable parameters of the new task $\mathcal{T}_i^t$ consist of the adapter and the classification head: $\theta_i^t = (\omega_i^t, \phi_i^t)$. To leverage prior knowledge of the past task-specific modules $\Theta^{t-1} = \{\theta_i^j \mid i \in [N], j \in [t-1]\}$, where each $\theta_i^j = (\omega_i^j, \phi_i^j)$, the federated adapter tuning loss $\mathcal{L}_{\mathcal{T}_i^t}(\theta_i^t; \Theta^{t-1})$ for each new task $\mathcal{T}_i^t$ is defined as:

$$\mathcal{L}_{\mathcal{T}_i^t}(\theta_i^t; \Theta^{t-1}) = \sum_{(x,y)\in\mathcal{T}_i^t} \ell\left(h_{\phi_i^t}(f_{\omega_i^t}(x)), y | \Theta^{t-1}\right). \tag{2}$$

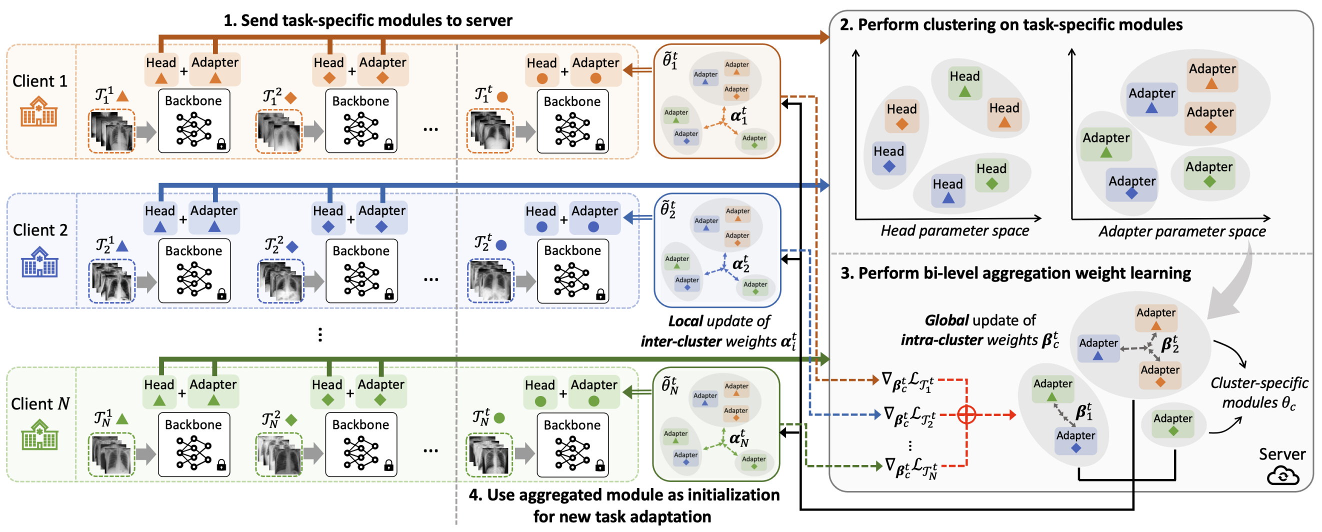
1. Send task-specific modules to server
2. Perform clustering on task-specific modules
Client 1
Client 2
Client N
Head
Adapter
Backbone
Head parameter space
Adapter parameter space
3. Perform bi-level aggregation weight learning
Local update of inter-cluster weights α_i^t
Global update of intra-cluster weights β_c^t
Cluster-specific modules θ_c
Server
4. Use aggregated module as initialization for new task adaptation

Fig. 2. An overview of FedKEI. After local fine-tuning on each new task, the task-specific modules (i.e., the adapter and head) are sent to the server, where global clustering is applied to all stored modules. When new tasks arrive, the server and clients collaboratively learn bi-level aggregation weights—the local inter-cluster weights α_i^t and global intra-cluster weights β_c^t—to combine the stored task-specific modules, optimizing towards new tasks' objectives through a bi-level optimization process. Specifically, at each outer-loop step, cluster-specific modules $\boldsymbol{\theta_c}$ are sent to clients, which perform inner-loop updates of α_i^t by optimizing on local tasks. Gradients w.r.t. $\boldsymbol{\theta_c}$ are then back-propagated through the inner-loop and returned to the server to update β_c^t (for simplicity, we illustrate this process using the adapter, though the same applies to the head). The final aggregated module $\tilde{\boldsymbol{\theta}}_i^t$, computed using the learned bi-level weights, is used to initialize learning for the new task.

B. Overview of Federated Knowledge-Enhanced Initialization (FedKEI) Framework

To achieve the objective of improving adaptation to new tasks as described in (2), we propose FedKEI — a framework that leverages *global clustering* and *bi-level aggregation weight learning* to generate informed initializations for the adapter and head for fine-tuning on new tasks. An overview of FedKEI is shown in Figure 2.

Generally, when a client encounters a new task, it locally fine-tunes the adapter and head (with the FM backbone fixed) and sends the task-specific modules to the server. These modules are accumulated over time into a knowledge pool, which can be leveraged for knowledge transfer to new tasks. Before local fine-tuning on a new task, the server performs two steps: (1) applies a global clustering algorithm to generalize knowledge across tasks (Section III-C); and (2) optimizes bi-level aggregation weights—global intra-cluster and personalized inter-cluster—to tailor knowledge transfer. This is done through a bi-level optimization scheme involving communication of cluster-specific modules and local task gradients between server and clients (Section III-D). The modules obtained by aggregating with the learned bi-level weights are used as initializations for adapter tuning on new tasks, effectively incorporating useful knowledge across time and space to facilitate new task learning (Section III-E).

In what follows, we describe each process in detail. Since our framework applies similarly to both adapters and heads—with clustering and weight learning performed separately in their respective parameter spaces—we use the term “module” and the symbol θ to refer to both the adapter and the head throughout the description.

C. Global Clustering of Task-Specific Modules

After local fine-tuning for each task at client side, the learned task-specific modules are sent to the server. Suppose we are at time t, the collected task-specific modules are up to time $t-1$, denoted by $\Theta^{t-1} = \{\theta_i^j | i \in [N], j \in [t-1]\}$. We apply a clustering algorithm (e.g., k-means++ [43]) on Θ^{t-1} to group task-specific modules with similar patterns. This serves to encourage generalization of related features among diverse tasks and facilitate knowledge transfer [44]–[46]. After that, task-specific modules belonging to the same cluster are aggregated to form a set of cluster-specific modules.

Formally, let $M = N \times (t-1)$ denote the total number of task-specific modules in Θ^{t-1}, and K denote the number of clusters. The cluster assignment outcome is denoted by a binary matrix $\mathbf{B} \in \{0,1\}^{K \times M}$, where $\mathbf{B}_{c,j}$ indicates whether the j-th task-specific module is assigned to cluster c. The cluster-specific module θ_c corresponding to cluster $c \in [K]$ is obtained by aggregating the task-specific modules assigned to that cluster:

$$\theta_c = \sum_{j=1}^{M} \frac{\mathbf{B}_{c,j}}{\sum_{j'=1}^{M} \mathbf{B}_{c,j'}} \cdot \theta_j, \quad \forall c \in [K]. \tag{3}$$

D. Bi-Level Aggregation Weight Learning

Given K cluster-specific modules, each client must determine a good initialization for learning a new task. A straightforward approach is to select the best-performing cluster module [44], but this overlooks useful information from other clusters. To address this, we propose a learning-based strategy that personalizes transfer from the clustered knowledge to each new task [27]. Specifically, we learn inter-cluster weights to combine cluster-specific modules and intra-cluster weights to refine aggregation within each cluster—together forming the bi-level aggregation weight learning. The inter-cluster weights assign importance to each cluster based on its utility for the new task, while the intra-cluster weights calibrate aggregation within each cluster to produce improved cluster-specific modules for learning personalized inter-cluster weights.

Inspired by meta-learning, which optimizes shared components (e.g., initializations or optimizers) to improve learning across tasks [20], we adopt a bi-level optimization scheme to learn these aggregation weights. We treat the intra-cluster weights as shared components across all clients' new tasks at time t and optimize them to enhance the learning of inter-cluster weights for individual tasks. Following the episodic learning paradigm of meta-learning, we optimize the global intra-cluster weights in the outer loop (at the server) and update the local inter-cluster weights in the inner loop (at the clients). We next describe the optimization procedure for both the inter- and intra-cluster weights.

1) Inner Updates of Local Inter-Cluster Weights: The inter-cluster weights are updated locally at the client side to directly optimize for the performance of the aggregated module on the new tasks, using the K cluster-specific modules $\{\theta_c\}_{c=1}^K$ received from the server.

Recalling Section III-C, the cluster-specific module θ_c is obtained by aggregating the M task-specific modules collected at the server with intra-cluster weights $\boldsymbol{\beta}_c = [\beta_{c,1}, \cdots, \beta_{c,M}] \in \mathbb{R}^M$, whose values are initialized based on the cluster assignment from the global clustering process, $\boldsymbol{\beta_c} \leftarrow \mathbf{B}_{c,:}/\sum_{j=1}^M \mathbf{B}_{c,j} \in \mathbb{R}^M$. The cluster-specific module θ_c computed using $\boldsymbol{\beta}_c$ is represented as:

$$\theta_c(\boldsymbol{\beta}_c) = \textstyle\sum_{j=1}^M \beta_{c,j} \cdot \theta_j, \quad \forall c \in [K]. \tag{4}$$

Let $\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_K] \in \mathbb{R}^K$ denote the inter-cluster weights of aggregating the K cluster-specific modules, initialized by $\frac{1}{K}$. The final aggregated module is obtained by:

$$\tilde{\theta}(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \textstyle\sum_{c=1}^K \alpha_c \cdot \theta_c(\boldsymbol{\beta}_c). \tag{5}$$

To adapt to the new task $\mathcal{T}_i^t$ of client i, we update the inter-cluster weight $\boldsymbol{\alpha}$ by optimizing $\tilde{\theta}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ on the task objective $\mathcal{L}_{\mathcal{T}_i^t}$, while keeping $\boldsymbol{\beta}$ fixed. That is, given the set of cluster-specific modules (obtained with some fixed $\boldsymbol{\beta}$) received from the server, we perform one or several steps of updates on $\boldsymbol{\alpha}$ at the client, which we refer to as the inner-loop updates. Formally, when performing one inner-loop update, the inter-cluster weight $\boldsymbol{\alpha}_i^t$ for task $\mathcal{T}_i^t$ is obtained by:

$$\boldsymbol{\alpha}_i^t = \boldsymbol{\alpha} - \eta_1 \nabla_{\boldsymbol{\alpha}} \mathcal{L}_{\mathcal{T}_i^t}(\tilde{\theta}(\boldsymbol{\alpha}, \boldsymbol{\beta})), \tag{6}$$

where η_1 is the inner-loop learning rate. The gradient $\nabla_{\boldsymbol{\alpha}} \mathcal{L}_{\mathcal{T}_i^t}$ is computed as $(\nabla_{\boldsymbol{\alpha}} \tilde{\theta})^\top \nabla_{\tilde{\theta}} \mathcal{L}_{\mathcal{T}_i^t}$, where $\nabla_{\boldsymbol{\alpha}} \tilde{\theta}$ is a matrix with the cluster-specific modules $[\theta_1, \cdots, \theta_K]$ as the column vectors (from (5)). For brevity, we describe our method with a single step of inner-loop update, but the approach can easily extend to multiple updates.

2) Outer Updates of Global Intra-Cluster Weights: The intra-cluster weights $\boldsymbol{\beta} = [\boldsymbol{\beta}_1, \cdots, \boldsymbol{\beta}_K]$ determine how the cluster-specific modules are formed. To ensure that the cluster-specific modules are optimized for learning the new task at time t across all clients, we learn the intra-cluster weights by (1) collaboratively optimizing them across all clients to enhance generalizability, and (2) ensuring that the resulting cluster-specific modules are optimal for learning the downstream inter-cluster aggregation.

The two objectives can be achieved by optimizing the intra-cluster weights collaboratively in the outer loop such that the inter-cluster weights updated for each client in the inner loop (i.e., $\boldsymbol{\alpha}_i^t, \forall i \in [N]$) perform the best. More concretely, at time t, the intra-cluster weights $\boldsymbol{\beta}^t$ is optimized by encouraging better performance of the updated inter-cluster aggregated module $\tilde{\theta}(\boldsymbol{\alpha}_i^t, \boldsymbol{\beta})$ across all new tasks $\{\mathcal{T}_i^t\}_{i=1}^N$ of N clients:

$$\boldsymbol{\beta}^t = \arg\min_{\boldsymbol{\beta}} \textstyle\sum_{i=1}^N \mathcal{L}_{\mathcal{T}_i^t}(\tilde{\theta}(\boldsymbol{\alpha}_i^t, \boldsymbol{\beta})). \tag{7}$$

Considering the intra-cluster weight $\boldsymbol{\beta}_c$ for each cluster c separately, the gradient descent update is given by:

$$\boldsymbol{\beta}_c \leftarrow \boldsymbol{\beta}_c - \eta_2 \textstyle\sum_{i=1}^N \nabla_{\boldsymbol{\beta}_c} \mathcal{L}_{\mathcal{T}_i^t}(\tilde{\theta}(\boldsymbol{\alpha}_i^t, \boldsymbol{\beta})), \quad \forall c \in [K], \tag{8}$$

where η_2 is the outer-loop learning rate. We can see that the gradient for updating $\boldsymbol{\beta}_c$ is the sum of the gradients of individual tasks $\nabla_{\boldsymbol{\beta}_c} \mathcal{L}_{\mathcal{T}_i^t}, \forall i \in [N]$, which by chain rule, can be expressed as:

$$\nabla_{\boldsymbol{\beta}_c} \mathcal{L}_{\mathcal{T}_i^t} = (\nabla_{\boldsymbol{\beta}_c} \theta_c)^\top (\nabla_{\theta_c} \tilde{\theta}(\boldsymbol{\alpha}_i^t, \boldsymbol{\beta}))^\top \nabla_{\tilde{\theta}(\boldsymbol{\alpha}_i^t, \boldsymbol{\beta})} \mathcal{L}_{\mathcal{T}_i^t}. \tag{9}$$

Here, the first term $\nabla_{\boldsymbol{\beta}_c} \theta_c$ is simply a matrix with the task-specific modules $[\theta_1, \cdots, \theta_M]$ as the column vectors (from (4)). The second term $\nabla_{\theta_c} \tilde{\theta}(\boldsymbol{\alpha}_i^t, \boldsymbol{\beta})$ is the derivative of the updated inter-cluster aggregated module $\tilde{\theta}(\boldsymbol{\alpha}_i^t, \boldsymbol{\beta})$ w.r.t. the cluster-specific module θ_c, where $\boldsymbol{\alpha}_i^t$ is obtained by updates in the inner loop leveraging the cluster-specific modules $\{\theta_c\}_{c=1}^K$ (see (6) and (5)). Hence, evaluating $\nabla_{\theta_c} \tilde{\theta}(\boldsymbol{\alpha}_i^t, \boldsymbol{\beta})$ involves back-propagating through the inner loop conducted at the client side. Deriving from (6), it can be obtained that $\nabla_{\theta_c} \tilde{\theta}(\boldsymbol{\alpha}_i^t, \boldsymbol{\beta}) = \alpha_{i,c}^t I - \eta_1 \theta_c (\nabla_{\tilde{\theta}(\boldsymbol{\alpha}, \boldsymbol{\beta})} \mathcal{L}_{\mathcal{T}_i^t})^\top$, where $\alpha_{i,c}^t$ is the updated inter-cluster weight associated with cluster c, and $\nabla_{\tilde{\theta}(\boldsymbol{\alpha}, \boldsymbol{\beta})} \mathcal{L}_{\mathcal{T}_i^t}$ is the task gradient w.r.t. the inter-cluster aggregated module using the initial $\boldsymbol{\alpha}$.

Generally, computing the outer-loop gradients in (9) requires (a) the task-specific modules stored at the server, which constitutes the first term $\nabla_{\boldsymbol{\beta}_c} \theta_c$, and (b) the gradients w.r.t. θ_c derived through the inner-loop $\nabla_{\theta_c} \mathcal{L}_{\mathcal{T}_i^t} = (\nabla_{\theta_c} \tilde{\theta}(\boldsymbol{\alpha}_i^t, \boldsymbol{\beta}))^\top \nabla_{\tilde{\theta}(\boldsymbol{\alpha}_i^t, \boldsymbol{\beta})} \mathcal{L}_{\mathcal{T}_i^t}$. Since the inner-loop updates of $\boldsymbol{\alpha}$ is conducted at the client side, we compute $\nabla_{\theta_c} \mathcal{L}_{\mathcal{T}_i^t}$ at clients and send it to the server to compute the outer-loop gradient $\nabla_{\boldsymbol{\beta}_c} \mathcal{L}_{\mathcal{T}_i^t}$ in (9). We then aggregate $\nabla_{\boldsymbol{\beta}_c} \mathcal{L}_{\mathcal{T}_i^t}$ across N clients to perform updates on $\boldsymbol{\beta}_c$ in (8). In our implementation, we perform only one outer-loop update. Hence, the additional computation and communication overhead introduced remains manageable. Since only gradients related to the cluster-specific modules are shared with the server, privacy is preserved.

3) Actual Learning of Inter-Cluster Weights: Upon completion of the bi-level optimization, we obtain a set of effective cluster-specific modules with updated intra-cluster weights $\boldsymbol{\beta}^t$, generalized for all the tasks at time t. On the updated cluster-specific modules, each client i then performs actual learning of personalized inter-cluster weight $\hat{\boldsymbol{\alpha}}_i^t$ for the new task $\mathcal{T}_i^t$:

$$\hat{\boldsymbol{\alpha}}_i^t = \arg\min_{\boldsymbol{\alpha}} \mathcal{L}_{\mathcal{T}_i^t}(\tilde{\theta}(\boldsymbol{\alpha}, \boldsymbol{\beta}^t)). \tag{10}$$

Note that the inner-loop updates described earlier are only auxiliary for optimizing the intra-cluster weights. Leveraging

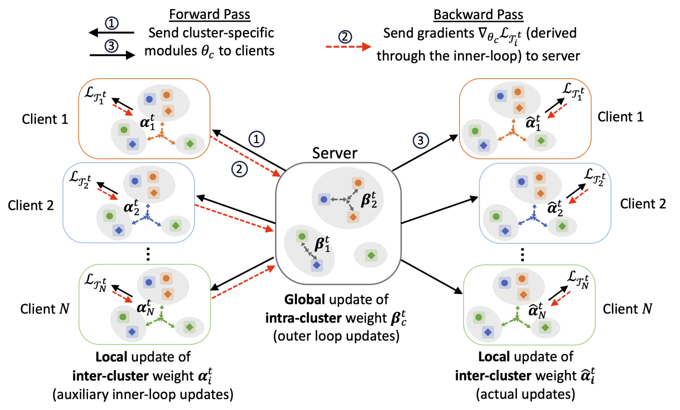
Forward Pass
① ③ Send cluster-specific modules θ_c to clients
Backward Pass
② Send gradients $\nabla_{\theta_c}\mathcal{L}_{\mathcal{T}_i^t}$ (derived through the inner-loop) to server
Client 1
Client 2
Client N
Server
Global update of intra-cluster weight β_c^t (outer loop updates)
Local update of inter-cluster weight α_i^t (auxiliary inner-loop updates)
Local update of inter-cluster weight $\hat{\alpha}_i^t$ (actual updates)

Fig. 3. An illustration of bi-level aggregation weight learning. The cluster-specific modules $\boldsymbol{\theta_c}$, weighted by global intra-cluster weights $\boldsymbol{\beta_c}$, are sent to clients (outer-loop forward pass). Based on the received $\boldsymbol{\theta_c}$, each client locally updates inter-cluster weights $\boldsymbol{\alpha}_i^t$ by minimizing the task loss $\mathcal{L}_{\mathcal{T}_i^t}$ (inner-loop). Upon completion, the gradients $\nabla_{\theta_c}\mathcal{L}_{\mathcal{T}_i^t}$ are derived through the inner-loop updates and sent back to the server (outer-loop backward pass). After outer-loop optimization, the learned $\boldsymbol{\beta}_c^t$ is forwarded for learning the actual inter-cluster weight $\hat{\boldsymbol{\alpha}}_i^t$.

the optimized intra-cluster weights, the inter-cluster weights $\hat{\boldsymbol{\alpha}}_i^t$ computed here contribute to the actual initialization for learning task $\mathcal{T}_i^t$. Figure 3 illustrates the full process of bi-level aggregation weight learning.

E. Initialization for New Task Adaptation

Once we obtain the learned intra-cluster weights $\boldsymbol{\beta}^t$ and the personalized inter-cluster weights $\hat{\boldsymbol{\alpha}}_i^t$ tailored to task $\mathcal{T}_i^t$, we generate the final aggregated module $\tilde{\theta}_i^t = \tilde{\theta}(\hat{\boldsymbol{\alpha}}_i^t, \boldsymbol{\beta}^t)$ and use it as initialization for adapter tuning on new task $\mathcal{T}_i^t$:

$$\min_{\theta_i^t \leftarrow \tilde{\theta}_i^t} \mathcal{L}_{\mathcal{T}_i^t}(\theta_i^t). \tag{11}$$

Generally, the learned bi-level weights $\hat{\boldsymbol{\alpha}}_i^t, \boldsymbol{\beta}^t$ collectively determine how to leverage knowledge from tasks stored in the knowledge pool to facilitate learning of new tasks.

IV. Experiments

A. Experimental Setup

1) Datasets and Settings: We evaluate FedKEI on three FL datasets across different medical imaging modalities for disease classification: (1) skin lesion identification from dermoscopic images; (2) chest disease diagnosis from X-rays; and (3) eye disease classification from retinal OCT. These datasets cover diverse imaging techniques and conditions, allowing us to assess robustness across clinical scenarios. We construct tasks to simulate the FCL setting, defining a task order such that, at each time step, the task (i.e., the disease) encountered by all clients is largely the same. This synchronous task order setup reflects real-world scenarios in which medical institutions often face the same new diseases—such as emerging illnesses or pandemic outbreaks—around the same time.

Derm-FL Dataset: Following [47], we collect data from four public skin datasets — ISIC19 [48], HAM10000 [49], PAD-UFES [50], and Derm7pt [51] — and split them into ten clients to simulate an FL setting ($N = 10$). Figure 4a shows the data distribution among ten clients. The combined dataset includes 37,607 dermoscopic images across eight skin lesion types: MEL, NV, BCC, AK, VASC, BKL, DF, and SCC. To simulate continual adaptation, we construct five sequential tasks per dataset ($M = 5$), each involving identifying a new disease from those previously seen. Figure 5a shows the order of disease learning for the four datasets. For example, the first task is to identify NV from MEL, the second BCC from {MEL, NV}, and so on. While tasks are mostly aligned across clients, some datasets lack certain diseases (e.g., PAD-UFES lacks DF, Derm7pt lacks AK). In such cases, we substitute available alternatives (e.g., VASC for AK in Derm7pt). This mild relaxation of task alignment reflects real-world variability in disease prevalence and data availability, allowing us to evaluate FedKEI's robustness under such conditions.

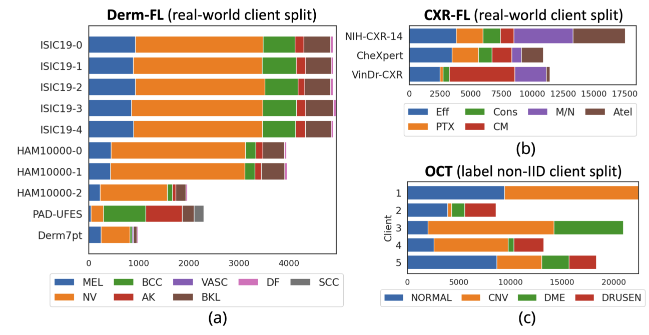

Fig. 4. Data distribution among clients for (a) Derm-FL, (b) CXR-FL, and (c) OCT datasets.

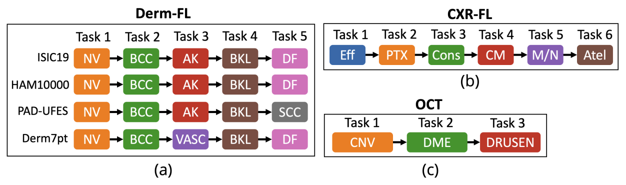
Derm-FL
Task 1
Task 2
Task 3
Task 4
Task 5
ISIC19: NV → BCC → AK → BKL → DF
HAM10000: NV → BCC → AK → BKL → DF
PAD-UFES: NV → BCC → AK → BKL → SCC
Derm7pt: NV → BCC → VASC → BKL → DF
(a)
CXR-FL
Task 1 Task 2 Task 3 Task 4 Task 5 Task 6
Eff → PTX → Cons → CM → M/N → Atel
(b)
OCT
Task 1 Task 2 Task 3
CNV → DME → DRUSEN
(c)

Fig. 5. Order of disease learning for (a) Derm-FL, (b) CXR-FL, and (c) OCT datasets.

CXR-FL Dataset: We compile chest X-ray images from three public datasets—NIH-CXR-14 [19], CheXpert [52], and VinDr-CXR [53]—treating each as a client to simulate an FL setting ($N = 3$). We retain seven common classes (including 'No Finding' and six diseases: Eff, PTX, Cons, CM, M/N, and Atel), resulting in 79,010 images. Figure 4b shows the data distribution among three clients. By treating each disease as a task, we construct six sequential tasks ($M = 6$). The first task is to identify Eff from No Finding, the second task is to identify PTX from {No Finding, Eff}, etc. Figure 5b shows the task order for CXR-FL dataset. Since all three clients contain the six diseases, the same task order is applied to all clients.

OCT Dataset: We also evaluate our method on the OCT dataset [54], which contains 84,495 retinal images spanning four conditions: Normal, CNV, DME, and Drusen. To simulate label non-IID scenario, we split each class among five clients following $Dir(0.5)$ ($N = 5$). Figure 4c shows the data distribution among five clients. We construct three tasks from the four classes ($M = 3$). The first task is to identify CNV from Normal, the second is to identify DME from {Normal, CNV}, etc. Figure 5c shows the task order for OCT dataset.

2) Implementation Details: For fair comparisons, all methods use the same FM adapter tuning setup. For our main experiments, we adopt ViT-B/16 [2] pre-trained on ImageNet as the fixed backbone F^* and fine-tune LoRA adapters [8]. For each new task, the adapter and head are fine-tuned locally for 30 epochs with a learning rate of 0.005 and batch size of 64 across all three datasets. Communication between server and clients occurs only at the start of each new task to transmit aggregated knowledge. All images are preprocessed to match the ViT-B/16 input: grayscale images (from CXR-FL and OCT) are converted to pseudo-RGB by channel replication, resized to 224×224, and normalized using ImageNet statistics, consistent with the FM's pretraining protocol.

For our FedKEI, we set both the inner-loop learning rate η_1 (for inter-cluster weight $\boldsymbol{\alpha}$) and the outer-loop learning rate η_2 (for intra-cluster weight $\boldsymbol{\beta}$) to 0.05. In each inner loop, we update the inter-cluster weight $\boldsymbol{\alpha}$ on local task for 1 epoch with a batch size of 64 (i.e., the number of inner-loop steps varies based on task data size). The number of outer-loop steps is set to 1. For clustering, we perform k-means++ [43] and tune the number of clusters K for both adapters and heads in $\{3, 5, 7, 9\}$. We use SGD optimizer for all gradient updates.

3) Evaluation Metrics: We evaluate new task adaptation using two metrics: (1) final AUC at the last epoch of each task, and (2) Learning Curve Area (LCA), calculated by averaging the AUC across all epochs of each task. The former assesses how well a task is learned and the latter indicates the speed of learning a task [55]. A larger LCA reflects faster learning, as it indicates that higher performance is reached earlier during training—often due to better initialization or effective knowledge transfer from prior tasks—resulting in a greater cumulative area under the learning curve. For all experiments, we conduct three trials with different seeds and report the mean and standard deviation.

B. Baseline Comparison

We compare FedKEI with 10 baselines across four FL categories. Traditional FL methods include: (1)FedAvg [22], which averages task-specific modules from all clients for initialization; (2)FedProx [23], which adds a proximal term to improve generalization; and (3)FFA-LoRA [36], which enhances LoRA aggregation by partially freezing LoRA's weights. FCL methods prevent forgetting: (4) FedCurv [14], a regularization-based method preserving prior tasks; and (5) FLwF [15], a distillation-based method aligning current and previous task logits. FDG methods improve generalization to unseen tasks: (6)ELCFS [31], which shares amplitude spectra across clients; and (7)FedGA [32], which enhances aggregation for better global generalization. PFL methods personalize models: (8) Per-FedAvg [25], a meta-learning method for fast adaptation; (9) FedAMP [26], which computes aggregation weights based on model similarity; and (10) APPLE [27], which learns aggregation weights by optimizing client objectives. Lastly, we include a naive baseline Rand, which randomly initializes the adapter and head for new task.

Table I, II and III present results on Derm-FL, CXR-FL, and OCT datasets, respectively, which include the final AUC of each individual task and the overall AUC and LCA averaged across all tasks. To assess the significance of FedKEI's performance gains over the baselines, we conducted Welch's t-test. The resulting p-values for AUC and LCA are denoted by superscripts following each method name in the tables: ‡ for $p < 0.001$, † for $p < 0.01$, and * for $p < 0.05$. For all columns, the best and second-best scores are shown in bold and underline, respectively. Note that no score is highlighted for Task 1, as all methods perform the same as Rand—except FedProx, FFA-LoRA, ELCFS, and Per-FedAvg, which involve modifications to the local fine-tuning process.

TABLE I

BASELINE COMPARISONS ON DERM-FL DATASET. WE REPORT THE INDIVIDUAL TASK PERFORMANCE (AUC) AVERAGED OVER 10 CLIENTS AS WELL AS THE OVERALL MEAN AUC AND LCA OVER 5 TASKS.

Method	Individual Task AUC					Overall	
	Task 1	Task 2	Task 3	Task 4	Task 5	AUC	LCA
Rand‡	87.92	93.33	87.16	78.38	71.04	83.57$\pm$0.12	81.15$\pm$0.05
FedAvg‡	87.92	93.40	88.94	78.64	72.89	84.36$\pm$0.11	81.50$\pm$0.10
FedProx†	86.54	91.88	<u>88.95</u>	78.31	74.01	84.14$\pm$0.18	<u>82.18</u>$\pm$0.23
FFA-LoRA†	88.19	93.38	88.58	<u>78.71</u>	73.66	84.50$\pm$0.35	81.56$\pm$0.30
FedCurv‡	87.92	93.00	88.43	77.60	69.39	83.27$\pm$0.00	80.32$\pm$0.00
FLwF‡	87.92	93.21	88.18	78.61	70.04	83.59$\pm$0.07	80.50$\pm$0.18
ELCFS‡	87.37	92.51	87.36	77.18	71.38	83.16$\pm$0.09	78.96$\pm$0.12
FedGA‡	87.92	93.51	88.91	78.69	73.81	84.57$\pm$0.06	81.79$\pm$0.06
Per-FedAvg‡	88.06	93.51	87.51	77.82	75.45	84.47$\pm$0.15	81.21$\pm$0.09
FedAMP‡	87.92	91.72	87.04	76.99	75.40	83.82$\pm$0.04	79.98$\pm$0.00
APPLE‡	87.92	<u>93.65</u>	87.66	78.06	<u>75.83</u>	<u>84.63</u>$\pm$0.16	81.61$\pm$0.07
FedKEI	87.92	**94.71**	**89.93**	**79.55**	**80.48**	**86.52**$\pm$0.10	**85.07**$\pm$0.08

First, from Table I, we observe that FedKEI consistently outperforms all baselines on the Derm-FL dataset. Specifically, it achieves improvements of 1.89% and 2.89% in overall AUC and LCA, respectively, over the second-best methods (APPLE, FedProx). These results suggest that FedKEI is effective in enhancing both the accuracy and efficiency of new task adaptation. Among the baselines, FCL methods (FedCurv, FLwF) perform poorly—often worse than the naive Rand—since merely preserving performance on old tasks does not guarantee effective transfer to new tasks and limits adaptability. FDG method ELCFS also underperforms, suggesting that enhancing local diversity without task-specific focus can hurt performance. Traditional FL methods (FedAvg, FedProx, FFA-LoRA) generally perform well, showing that aggregating past task-specific modules offers more useful initializations than Rand. Among PFL baselines, rule-based FedAMP lags behind, while learning-based Per-FedAvg and APPLE perform well, highlighting the benefits of direct task-specific optimization. Our FedKEI, by learning the bi-level aggregation specifically on the new tasks, generates the most effective initializations and achieves the best overall performance.

For the CXR-FL dataset (Table II), FedKEI outperforms the strongest baselines (APPLE, Per-FedAvg) by 0.98% and 1.16% in AUC and LCA, respectively. All FL methods (except ELCFS) outperform Rand, highlighting the value of leveraging previous tasks (through model aggregation, knowledge preservation or personalization) in facilitating new task learning for this dataset. Our FedKEI further boosts performance by

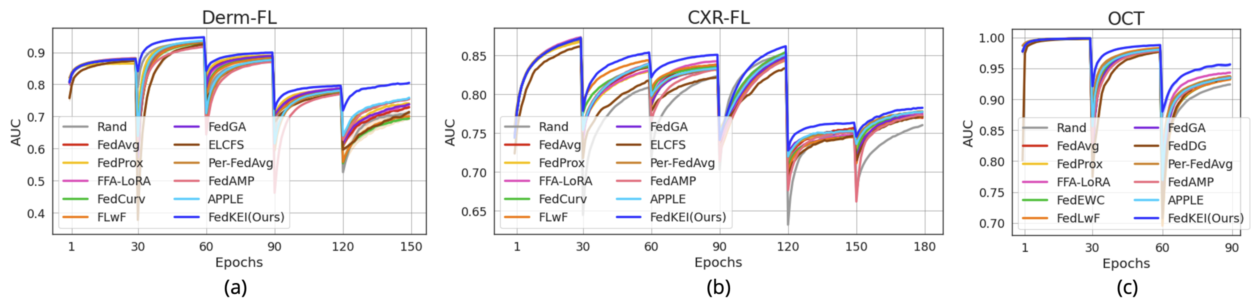

Fig. 6. Learning curves for individual tasks of FedKEI and the compared baselines on (a) Derm-FL, (b) CXR-FL, and (c) OCT datasets.

TABLE II
BASELINE COMPARISONS ON CXR-FL DATASET. WE REPORT THE INDIVIDUAL TASK PERFORMANCE (AUC) AVERAGED OVER 3 CLIENTS AS WELL AS THE OVERALL MEAN AUC AND LCA OVER 6 TASKS.

Method	Individual Task AUC						Overall	
	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	AUC	LCA
Rand*	87.23	80.88	82.35	85.28	75.23	76.03	81.17±0.44	78.40±0.54
FedAvg*	87.23	83.56	83.80	84.87	75.68	76.96	82.02±0.02	79.93±0.24
FedProx*	86.74	82.99	83.17	84.36	75.32	77.19	81.63±0.61	79.84±0.60
FFA-LoRA‡	87.33	83.09	84.24	85.26	74.78	77.56	82.04±0.01	79.95±0.02
FedCurv*	87.23	83.60	83.50	85.42	74.64	77.14	81.92±0.20	80.13±0.24
FLwF†	87.23	84.36	83.75	84.60	74.45	77.06	81.91±0.18	80.04±0.03
ELCFS*	86.14	81.77	82.19	83.38	74.61	77.10	80.86±0.41	78.63±0.37
FedGA†	87.23	83.87	83.23	84.79	75.06	77.33	81.92±0.15	79.92±0.17
Per-FedAvg*	87.09	83.85	83.22	85.16	75.13	77.78	82.04±0.23	80.17±0.19
FedAMP*	87.23	83.94	83.22	84.44	75.16	77.27	81.88±0.26	79.49±0.21
APPLE*	87.23	83.91	83.34	85.11	75.23	77.73	82.09±0.25	80.17±0.21
FedKEI	87.23	**85.36**	**85.07**	**86.16**	**76.33**	**78.25**	**83.07**±0.05	**81.33**±0.06

TABLE III
BASELINE COMPARISONS ON OCT DATASET. WE REPORT THE INDIVIDUAL TASK PERFORMANCE (AUC) AVERAGED OVER 5 CLIENTS AS WELL AS THE OVERALL MEAN AUC AND LCA OVER 3 TASKS.

Method	Individual Task AUC			Overall	
	Task 1	Task 2	Task 3	AUC	LCA
Rand‡	99.82	97.92	92.39	96.71±0.00	95.12±0.08
FedAvg†	99.82	97.90	93.65	97.12±0.12	95.32±0.18
FedProx†	99.74	97.67	93.28	96.90±0.20	95.31±0.16
FFA-LoRA†	99.83	97.93	94.30	97.35±0.04	95.67±0.02
FedCurv‡	99.82	97.92	93.30	97.01±0.00	94.96±0.10
FLwF‡	99.82	98.26	93.18	97.08±0.00	95.07±0.01
ELCFS*	99.77	97.68	93.51	96.99±0.40	94.34±0.72
FedGA†	99.82	97.91	93.45	97.06±0.10	95.20±0.04
Per-FedAvg†	99.83	98.03	93.70	97.19±0.09	95.91±0.01
FedAMP†	99.82	97.87	93.49	97.06±0.09	95.29±0.05
APPLE‡	99.82	98.02	93.47	97.10±0.02	95.56±0.08
FedKEI	99.82	**98.74**	**95.62**	**98.06**±0.02	**97.11**±0.14

effectively learning the bi-level aggregation. Similar trends are observed on the OCT dataset (Table III), where our FedKEI again achieves the best performance, surpassing the strongest baselines (FFA-LoRA, Per-FedAvg) by 0.71% and 1.20% in AUC and LCA, respectively.

Figure 6 shows the learning curves of FedKEI and the baselines across all three datasets. FedKEI consistently outperforms all baselines on each task and also enables faster task learning as observed from the learning curves. For instance, for the last task of Derm-FL (as shown in Figure 1a), our FedKEI is able to achieve the final-epoch performance of the baselines within only a few initial epochs, which means that only a fraction of the training time is required to achieve similar performance as the baselines. More concretely, the strongest baseline APPLE achieves 75.83 AUC for the last task of Derm-FL at epoch 30 in 146.9 seconds, while our FedKEI surpasses this with 76.21 AUC at epoch 4, requiring just 20.3 seconds. This advantage is attributed to the high-quality initializations generated by FedKEI (as shown by its strong performance at the start of each task), which facilitate not just better final performance but also faster adaptation to new tasks.

C. Computation & Communication Costs

In Table IV, we summarize the computation and communication costs on Derm-FL dataset. The computation cost is measured in two ways: (1) the average GPU execution time per task, recorded on a single NVIDIA RTX 3090 GPU with 24GB memory; and (2) the number of floating-point operations required per task, including both client-side and server-side computations, reported in teraFLOPs (TFLOPs). For communication cost, we report the theoretical download and upload size per client, where $|\theta|$ denotes the parameter size of the adapter and the head, $|\mathcal{D}_\mathcal{T}|$ denotes the total storage size of the task dataset, N denotes the number of clients, and K is the number of clusters in our method.

From the results, we observe that FedKEI's computation cost is comparable to the baselines, ranking around the middle among the 11 methods in terms of both GPU time and FLOPs. Compared to FedAvg, it adds only 2.92 minutes of GPU time and 10.08 TFLOPs per task. FFA-LoRA incurs the lowest TFLOPs due to partial freezing of LoRA. While the cost of FedKEI is higher than that of FedProx and FFA-LoRA, it is considerably lower than more sophisticated methods such as FLwF, FedGA, and Per-FedAvg, which involve additional inference or complex meta-learning procedures that introduce substantial computation during the task learning phase. In contrast, our FedKEI performs clustering and bi-level aggregation learning only once prior to task learning, keeping the overall computation cost relatively modest.

For communication cost, FedKEI requires transmitting data equivalent to K cluster-specific modules three times between the server and clients during bi-level aggregation learning (as shown in Figure 3), and uploading the learned task-specific modules once to the server after local task learning, leading

TABLE IV
COMPUTATION AND COMMUNICATION COSTS ON DERM-FL DATASET.

Method	Computation Cost		Communication Cost
	GPU time / task	TFLOPS / task	Theoretical cost
FedAvg	20.61 min	128.54	$2 \times \|\theta\|$
FedProx	20.66 min	128.54	$2 \times \|\theta\|$
FFA-LoRA	20.59 min	85.68	$\|\theta\|$
FedCurv	21.50 min	156.48	$(2N+2) \times \|\theta\|$
FLwF	29.75 min	192.90	$2 \times \|\theta\|$
ELCFS	36.23 min	130.68	$(N+1) \times \|\mathcal{D}_\mathcal{T}\|$
FedGA	32.39 min	157.10	$2 \times \|\theta\|$
Per-FedAvg	36.64 min	170.48	$4 \times \|\theta\|$
FedAMP	20.92 min	128.78	$2 \times \|\theta\|$
APPLE	21.39 min	149.64	$(N+1) \times \|\theta\|$
FedKEI	23.52 min	138.62	$(3K+1) \times \|\theta\|$

to a total communication size of $(3K+1) \times |\theta|$. Although it is higher than methods like FedAvg involving only constant multiples of $|\theta|$, the number of clusters K is typically smaller than the number of clients N, making FedKEI more efficient than FedCurv and APPLE. ELCFS incurs far higher cost than the others by transmitting amplitude spectra as large as the raw dataset $|\mathcal{D}_\mathcal{T}|$. Since $|\theta|$ consists only of lightweight components (e.g., a LoRA adapter on ViT-B/16 is only 0.28 MB), the actual communication cost of our FedKEI remains low—approximately 2.86 MB with $K = 3$ in our experiments.

D. Ablation Studies

1) Effect of Key Components in FedKEI: To assess the contribution of each component in FedKEI, we compare it with three incremental variants: Variant A aggregates all previous task-specific modules within each client into client-specific modules and learns the weights $\boldsymbol{\alpha}$ to combine them for initialization; Variant B adds global clustering, replacing client-specific with cluster-specific modules for aggregation; Variant C learns the intra-cluster weights $\boldsymbol{\beta}$ via direct gradient descent on new task objectives, similarly to $\boldsymbol{\alpha}$. Our FedKEI further employs a bi-level optimization scheme, optimizing the intra-cluster weight $\boldsymbol{\beta}$ collaboratively across clients to facilitate the local adaptation of the inter-cluster weight $\boldsymbol{\alpha}$.

Table V presents ablation results on the three datasets, along with the performance gain of each variant over its predecessor. We observe consistent improvements with each added component across all datasets. Notably, learning the inter-cluster (or inter-client) weight $\boldsymbol{\alpha}$ yields significant gains over Rand, especially for Derm-FL and CXR-FL. This is because the mechanism of learning to combine past modules towards new task objectives serves to generate a much more informative initialization than the random initialization. Adding clustering further boosts performance, as the task-specific modules within each cluster are now more relevant, producing diverse clusters that enable more flexible personalized aggregation. Allowing the intra-cluster weight $\boldsymbol{\beta}$ to be learned towards new tasks further increases the flexibility of customizing the initializations. Our FedKEI, by employing the bi-level optimization scheme, shows notable improvements over Variant C, demonstrating its effectiveness in learning better bi-level weights and producing better initializations for improved task adaptation.

2) Effect of FM and Adapter Choices: To validate FedKEI's compatibility with different FMs and adapters, we conduct two experiments: (1) fine-tuning **Swin-B** [56] (pretrained on ImageNet) with LoRA, and (2) fine-tuning ViT-B/16 with **IA3 adapter** [57]. Swin-B is a variant of Swin Transformer that employs hierarchical structure and shifted windows for multi-scale feature extraction, and IA3 adapts attention layers via learned rescaling vectors for the keys and values. The results on Derm-FL are summarized in Table VI.

From the results, we observe two key findings. First, Swin-B with LoRA outperforms ViT-B/16 with LoRA (Table I), likely due to its superior multi-scale feature processing for medical images, while ViT-B/16 with IA3 achieves performance comparable to LoRA. Second, FedKEI consistently achieves top performance: with Swin-B + LoRA, it surpasses the second-best performer (FedGA)) by 0.96% in AUC and 1.88% in LCA; with ViT-B/16 + IA3, it outperforms the second-best performer (APPLE) by 1.27% in AUC and 2.75% in LCA. These results highlight FedKEI's robustness and adaptability across diverse FMs and adapters. Other baseline trends remain consistent with Table I. Note that since FFA-LoRA is designed specifically for LoRA, it is omitted from IA3 experiments.

3) Effect of Task Order: In this section, we assess FedKEI's robustness to task order variations on Derm-FL. The first experiment retains synchronous alignment but reverses the task order for each of the four source datasets (e.g., ISIC19: DF $\rightarrow$ BKL $\rightarrow$ AK $\rightarrow$ BCC $\rightarrow$ NV). The second adopts an asynchronous setup, randomly shuffling the task order for each client so that all clients follow different task orders. The former tests the effect of task order in the synchronous setup, while the latter tests the asynchronous setup, where a disease encountered by one client may have been seen by other clients.

As shown in Table VII, reversing the task order slightly improves overall performance compared to the original (Table I), suggesting that learning certain tasks earlier may benefit subsequent ones—likely due to shared visual features or semantic similarities that enhance knowledge transfer. In this setting, FedKEI outperforms all baselines, with margins of 1.40% in AUC and 1.58% in LCA over the second-best performer Per-FedAvg. Shuffling task orders across clients yields notable AUC gains of 1.5–3% compared to Table I, likely because when clients follow different task orders, a new task encountered by one client may have already been learned by others, allowing it to benefit from previously learned modules. Under this asynchronous setup, FedKEI again achieves the best results, surpassing the next-best methods (APPLE, FedGA) by 1.08% in AUC and 1.81% in LCA, demonstrating its robustness across different task order settings.

E. Qualitative Analysis

In this section, we qualitatively analyze FedKEI's capability of learning meaningful initializations by visualizing the clustering results and the learned inter-cluster weights $\boldsymbol{\alpha}$ for assigning importance to different clusters. For this analysis, we use CXR-FL dataset, which consists of three clients and six tasks for each client. We investigate the quality of the inter-cluster weight $\boldsymbol{\alpha}$ learned for the last task (i.e., the 6^{th} task)

TABLE V

ABLATION STUDIES OF FEDKEI'S KEY COMPONENTS. EACH VARIANT INCLUDES ONE MORE DESIGN AT A TIME. FOR EASE OF COMPARISON, WE ALSO REPORT THE **PERFORMANCE INCREMENT OF EACH VARIANT OVER THE PREVIOUS VERSION** AFTER ADDING A NEW COMPONENT.

Variant	learned α	modules clustering	learned β	bi-level optimization	**Derm-FL**				**CXR-FL**				**OCT**			
					AUC	Incr.	LCA	Incr.	AUC	Incr.	LCA	Incr.	AUC	Incr.	LCA	Incr.
Rand					83.57$\pm$0.12	-	81.15$\pm$0.05	-	81.17$\pm$0.44	-	78.40$\pm$0.54	-	96.71$\pm$0.00	-	95.12$\pm$0.08	-
Variant A	✓				85.62$\pm$0.00	2.05	84.19$\pm$0.01	3.04	81.97$\pm$0.23	0.8	80.15$\pm$0.09	1.75	96.93$\pm$0.20	0.22	95.84$\pm$0.30	0.72
Variant B	✓	✓			85.97$\pm$0.03	0.35	84.24$\pm$0.08	0.05	82.11$\pm$0.40	0.14	80.28$\pm$0.32	0.13	97.37$\pm$0.32	0.44	96.33$\pm$0.40	0.49
Variant C	✓	✓	✓		86.02$\pm$0.12	0.05	84.41$\pm$0.12	0.17	82.49$\pm$0.07	0.38	80.61$\pm$0.13	0.33	97.50$\pm$0.23	0.13	96.43$\pm$0.09	0.10
FedKEI	✓	✓	✓	✓	**86.52**$\pm$0.02	0.50	**85.07**$\pm$0.03	0.66	**83.07**$\pm$0.05	0.58	**81.33**$\pm$0.06	0.72	**98.06**$\pm$0.02	0.56	**97.11**$\pm$0.14	0.68

TABLE VI

BASELINE COMPARISONS ON DERM-FL DATASET WITH (1) FINE-TUNING **SWIN-B** WITH LORA, AND (2) FINE-TUNING VIT-B/16 WITH **IA3**.

Method	**Swin-B** + LoRA		ViT-B/16 + **IA3**	
	AUC	LCA	AUC	LCA
Rand	86.54$\pm$0.08	84.44$\pm$0.12	83.99$\pm$0.10	82.70$\pm$0.16
FedAvg	87.85$\pm$0.10	85.15$\pm$0.17	84.39$\pm$0.14	82.19$\pm$0.21
FedProx	87.79$\pm$0.21	85.38$\pm$0.31	84.53$\pm$0.13	82.56$\pm$0.18
FFA-LoRA	88.00$\pm$0.11	85.50$\pm$0.25	-	-
FedCurv	86.60$\pm$0.01	83.86$\pm$0.06	82.96$\pm$0.07	80.25$\pm$0.15
FLwF	86.31$\pm$0.18	83.36$\pm$0.13	83.17$\pm$0.19	80.48$\pm$0.20
ELCFS	86.47$\pm$0.09	82.96$\pm$0.16	83.78$\pm$0.18	80.56$\pm$0.25
FedGA	$\underline{88.08}$$\pm$0.08	$\underline{85.58}$$\pm$0.10	84.63$\pm$0.11	82.64$\pm$0.11
Per-FedAvg	87.96$\pm$0.10	85.07$\pm$0.20	84.65$\pm$0.20	82.55$\pm$0.14
FedAMP	85.60$\pm$0.02	81.99$\pm$0.11	82.86$\pm$0.07	78.50$\pm$0.11
APPLE	87.31$\pm$0.11	84.80$\pm$0.16	$\underline{84.82}$$\pm$0.09	$\underline{82.81}$$\pm$0.22
FedKEI	**89.04**$\pm$0.10	**87.46**$\pm$0.15	**86.09**$\pm$0.09	**85.56**$\pm$0.11

TABLE VII

BASELINE COMPARISONS ON DERM-FL DATASET WITH (1) SYNCHRONOUS **REVERSED** TASK ORDER, AND (2) ASYNCHRONOUS **SHUFFLED** TASK ORDERS ACROSS CLIENTS.

Method	**Reversed** Task Order		**Shuffled** Task Order	
	AUC	LCA	AUC	LCA
Rand	85.01$\pm$0.11	82.99$\pm$0.18	86.44$\pm$0.13	85.34$\pm$0.20
FedAvg	85.24$\pm$0.08	83.39$\pm$0.13	86.57$\pm$0.21	85.42$\pm$0.19
FedProx	85.14$\pm$0.22	83.31$\pm$0.17	86.64$\pm$0.09	85.52$\pm$0.13
FFA-LoRA	85.01$\pm$0.23	83.44$\pm$0.15	86.85$\pm$0.22	85.57$\pm$0.31
FedCurv	85.07$\pm$0.09	82.85$\pm$0.09	86.39$\pm$0.12	84.70$\pm$0.15
FLwF	84.91$\pm$0.16	82.80$\pm$0.20	86.46$\pm$0.11	84.67$\pm$0.19
ELCFS	84.36$\pm$0.05	81.58$\pm$0.11	85.87$\pm$0.24	82.62$\pm$0.21
FedGA	85.28$\pm$0.19	83.41$\pm$0.11	86.79$\pm$0.07	$\underline{85.76}$$\pm$0.13
Per-FedAvg	$\underline{85.38}$$\pm$0.08	$\underline{83.76}$$\pm$0.15	86.66$\pm$0.06	85.38$\pm$0.14
FedAMP	84.40$\pm$0.01	81.92$\pm$0.04	86.29$\pm$0.04	84.63$\pm$0.05
APPLE	85.37$\pm$0.06	83.62$\pm$0.18	$\underline{86.86}$$\pm$0.11	85.74$\pm$0.15
FedKEI	**86.78**$\pm$0.07	**85.34**$\pm$0.11	**87.94**$\pm$0.14	**87.57**$\pm$0.09

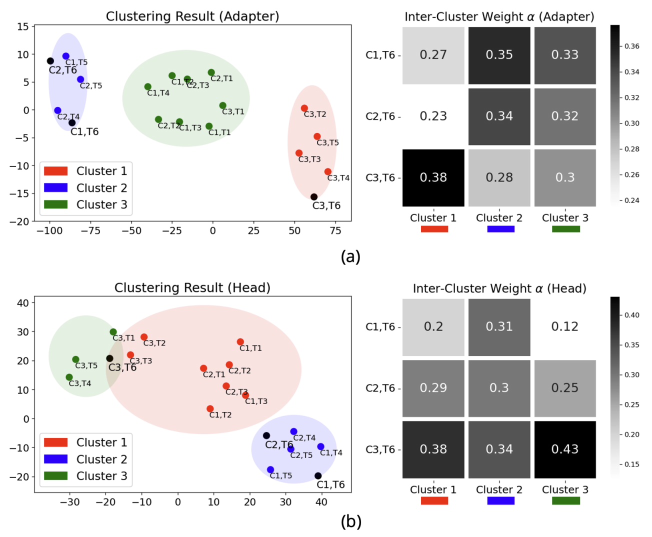

Fig. 7. Clustering results (t-SNE) and the inter-cluster weight α learned for the 6th task of three clients of CXR-FL for (a) adapter and (b) head. Each point (Ci,Tt) is the task-specific module of task t of client i.

by assessing how closely α aligns with the optimized adapter and head of the 6th task (obtained after local fine-tuning) in terms of assigning importance to different clusters to generate the initializations. Here, we set the number of clusters $K = 3$ for both adapters and heads.

Figure 7 shows the t-SNE plots of the clustering results and the learned α for the 6th task of three clients for the adapters and heads, respectively. In the t-SNE plots, each point (Ci,Tt) represents the task-specific module of task t of client i, obtained after local fine-tuning on that task. To generate initializations for the 6th task, we perform clustering on all the $3 \times 5 = 15$ previous task-specific modules stored at the server. The clustering results are shown in three different colors, and the black dots represent the optimized modules of the 6th task of three clients after local fine-tuning (which are not involved in the clustering). From the plots, we see that FedKEI tends to assign larger weights to clusters closer to the optimized modules of the 6th task. For instance, in Figure 7a, the optimized adapter of the 6th task of client 3 (i.e., point C3,T6) is located closer to cluster 1 and more distant from cluster 2. The weight α learned to generate the adapter's initialization for this task assigns a higher value to cluster 1 and a lower value to cluster 2. Similar trends are also observed in Figure 7b for the heads. This alignment shows that FedKEI effectively identifies clusters more relevant to the optimally learned adapter and head of the new task, generating well-informed initializations that facilitate new task learning.

F. Evaluation on Medical Pretrained FMs and 3D Images

To further assess FedKEI's effectiveness, we conduct experiments under two additional setups: (1) using a larger medically-pretrained FM and (2) applying it to 3D medical imaging. For the first, we evaluate FedKEI with RETFound

TABLE VIII
BASELINE COMPARISONS BY FINE-TUNING LORA ON (1) RETFOUND WITH THE OCT DATASET, AND (2) VOCO WITH THE CC-CCII DATASET.

Method	RETFound + LoRA on OCT		VoCo + LoRA on CC-CCII	
	AUC	LCA	AUC	LCA
Rand	97.92±0.03	95.21±0.09	85.70±0.09	86.32±0.04
FedAvg	98.85±0.08	96.19±0.05	86.09±0.05	87.30±0.21
FedProx	98.66±0.07	96.54±0.03	85.74±0.14	86.79±0.07
FFA-LoRA	98.77±0.13	96.14±0.10	86.19±0.08	87.12±0.05
FedCurv	98.51±0.04	95.14±0.01	85.61±0.05	86.57±0.07
FLwF	98.55±0.07	96.63±0.11	85.62±0.10	86.61±0.19
ELCFS	98.10±0.13	95.52±0.04	85.59±0.05	85.74±0.17
FedGA	98.78±0.05	96.20±0.09	86.26±0.11	87.31±0.10
Per-FedAvg	98.58±0.11	96.92±0.05	86.24±0.09	87.52±0.05
FedAMP	98.85±0.05	96.40±0.08	85.77±0.03	86.42±0.10
APPLE	98.86±0.08	96.77±0.10	86.30±0.07	87.62±0.13
FedKEI	**99.07**±0.07	**96.98**±0.04	**87.14**±0.07	**88.58**±0.11

[58], fine-tuning its ViT-L/16 encoder with LoRA on our federated OCT dataset. For the second, we use the CC-CCII dataset [59], which contains CT scans from three classes: Normal, Common Pneumonia (CP), and Novel Coronavirus Pneumonia (NCP). We simulate the FCL setting by distributing each class across 5 clients using $Dir(0.5)$ and defining two sequential tasks: identifying CP from Normal, and NCP from $\{$Normal, CP$\}$. We adopt the VoCo model [60] pretrained on 3D CT data and fine-tune it with LoRA on CC-CCII.

The results, summarized in Table VIII, demonstrate the robustness of FedKEI across both settings. In the first setup, FedKEI achieves top performance with RETFound, outperforming the second-best method APPLE by 0.21% in AUC—even in a near-saturated regime where most methods approach 99% AUC due to strong pretraining on highly relevant OCT data. In the second setup with 3D CT images, FedKEI again leads, surpassing APPLE by 0.84% in AUC and 0.96% in LCA. These results demonstrate FedKEI's effectiveness with larger medical FMs and in challenging 3D imaging tasks.

V. CONCLUSION

In this work, we propose FedKEI, a novel FL framework that enhances adaptation to new diseases for FM adapter tuning. FedKEI selectively transfers knowledge from previous task-specific modules to generate informed initializations for new tasks. It performs global clustering at the server to generalize knowledge across tasks, followed by bi-level aggregation weight learning to personalize transfer for each new task. Extensive experiments on medical datasets demonstrates FedKEI's advantage in adapting to new diseases compared to state-of-the-art methods.

Despite its strong performance, FedKEI has several limitations that offer directions for future work: (1) It has yet to be evaluated in real-world large-scale FL systems, where challenges like device heterogeneity and connectivity instability may arise; (2) Testing on additional modalities and multimodal settings (e.g., with EHRs or genomics) could further assess its generalizability; and (3) While the added overhead is moderate, future work will explore first-order approximations and model compression to improve efficiency.

REFERENCES

[1] N. Rieke, J. Hancox, W. Li, F. Milletari, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein *et al.*, "The future of digital health with federated learning," *NPJ digital medicine*, vol. 3, no. 1, p. 119, 2020.

[2] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2020.

[3] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[4] W. Zhuang, C. Chen, and L. Lyu, "When foundation model meets federated learning: Motivations, challenges, and future directions," *arXiv preprint arXiv:2306.15546*, 2023.

[5] H. Woisetschläger, A. Isenko, S. Wang, R. Mayer, and H.-A. Jacobsen, "A survey on efficient federated learning methods for foundation model training," *arXiv preprint arXiv:2401.04472*, 2024.

[6] J. Bian, Y. Peng, L. Wang, Y. Huang, and J. Xu, "A survey on parameter-efficient fine-tuning for foundation models in federated learning," *arXiv preprint arXiv:2504.21099*, 2025.

[7] S. Li, F. Ye, M. Fang, J. Zhao, Y.-H. Chan, E. C.-H. Ngai, and T. Voigt, "Synergizing foundation models and federated learning: A survey," *arXiv preprint arXiv:2406.12844*, 2024.

[8] E. J. Hu, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen *et al.*, "Lora: Low-rank adaptation of large language models," in *International Conference on Learning Representations*, 2021.

[9] W. Lu, H. Xixu, J. Wang, and X. Xie, "Fedclip: Fast generalization and personalization for clip in federated learning," in *ICLR 2023 Workshop on Trustworthy and Reliable Large-Scale Machine Learning Models*, 2023.

[10] H. Zhao, W. Du, F. Li, P. Li, and G. Liu, "Fedprompt: Communication-efficient and privacy-preserving prompt tuning in federated learning," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[11] A. Youssef, T. Zhu, A. Thakur, P. Watkinson, P. Horby, D. W. Eyre, and D. A. Clifton, "Rapid_ai: A framework for rapidly deployable ai for novel disease & pandemic preparedness," *medRxiv*, pp. 2022–08, 2022.

[12] M. M. Derakhshani, I. Najdenkoska, T. van Sonsbeek, X. Zhen, D. Mahapatra, M. Worring, and C. G. Snoek, "Lifelonger: A benchmark for continual disease classification," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2022, pp. 314–324.

[13] J. Yoon, W. Jeong, G. Lee, E. Yang, and S. J. Hwang, "Federated continual learning with weighted inter-client transfer," in *International Conference on Machine Learning*. PMLR, 2021, pp. 12 073–12 086.

[14] N. Shoham, T. Avidor, A. Keren, N. Israel, D. Benditkis, L. Mor-Yosef, and I. Zeitak, "Overcoming forgetting in federated learning on non-iid data," *arXiv preprint arXiv:1910.07796*, 2019.

[15] A. Usmanova, F. Portet, P. Lalanda, and G. Vega, "A distillation-based approach integrating continual learning and federated learning for pervasive services," in *3rd Workshop on Continual and Multimodal Learning for Internet of Things–Co-located with IJCAI 2021*, 2021.

[16] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," *arXiv preprint arXiv:1606.04671*, 2016.

[17] M. Hoffmann, H. Kleine-Weber, S. Schroeder, N. Krüger, T. Herrler, S. Erichsen, T. S. Schiergens, G. Herrler, N.-H. Wu, A. Nitsche *et al.*, "Sars-cov-2 cell entry depends on ace2 and tmprss2 and is blocked by a clinically proven protease inhibitor," *cell*, vol. 181, no. 2, pp. 271–280, 2020.

[18] B. Rockx, T. Kuiken, S. Herfst, T. Bestebroer, M. M. Lamers, B. B. Oude Munnink, D. de Meulder, G. van Amerongen, J. van den Brand, N. M. Okba *et al.*, "Comparative pathogenesis of covid-19, mers, and sars in a nonhuman primate model," *Science*, vol. 368, no. 6494, pp. 1012–1015, 2020.

[19] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, "Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2097–2106.

[20] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International conference on machine learning*. PMLR, 2017, pp. 1126–1135.

[21] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, Y. Li, X. Liu, and B. He, "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 4, pp. 3347–3366, 2021.

[22] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[23] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

[24] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International conference on machine learning*. PMLR, 2020, pp. 5132–5143.

[25] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: a model-agnostic meta-learning approach," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2020, pp. 3557–3568.

[26] Y. Huang, L. Chu, Z. Zhou, L. Wang, J. Liu, J. Pei, and Y. Zhang, "Personalized cross-silo federated learning on non-iid data," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 9, 2021, pp. 7865–7873.

[27] J. Luo and S. Wu, "Adapt to adaptation: Learning personalization for cross-silo federated learning," in *IJCAI: proceedings of the conference*, vol. 2022. NIH Public Access, 2022, p. 2166.

[28] J. Dong, L. Wang, Z. Fang, G. Sun, S. Xu, X. Wang, and Q. Zhu, "Federated class-incremental learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 164–10 173.

[29] J. Tang, H. Zhuang, J. He, R. He, J. Wang, K. Fan, A. Liu, T. Wang, L. Wang, Z. Zhu *et al.*, "Afcl: Analytic federated continual learning for spatio-temporal invariance of non-iid data," *arXiv preprint arXiv:2505.12245*, 2025.

[30] Y. Li, X. Wang, R. Zeng, P. K. Donta, I. Murturi, M. Huang, and S. Dustdar, "Federated domain generalization: A survey," *arXiv preprint arXiv:2306.01334*, 2023.

[31] Q. Liu, C. Chen, J. Qin, Q. Dou, and P.-A. Heng, "Feddg: Federated domain generalization on medical image segmentation via episodic learning in continuous frequency space," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 1013–1023.

[32] R. Zhang, Q. Xu, J. Yao, Y. Zhang, Q. Tian, and Y. Wang, "Federated domain generalization with generalization adjustment," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 3954–3963.

[33] D. Caldarola, B. Caputo, and M. Ciccone, "Improving generalization in federated learning by seeking flat minima," in *European Conference on Computer Vision*. Springer, 2022, pp. 654–672.

[34] Y. Wu, C. Desrosiers, and A. Chaddad, "Facmic: Federated adaptative clip model for medical image classification," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2024, pp. 531–541.

[35] Z. Wang, Z. Shen, Y. He, G. Sun, H. Wang, L. Lyu, and A. Li, "Flora: Federated fine-tuning large language models with heterogeneous low-rank adaptations," *arXiv preprint arXiv:2409.05976*, 2024.

[36] Y. Sun, Z. Li, Y. Li, and B. Ding, "Improving lora in privacy-preserving federated learning," in *The Twelfth International Conference on Learning Representations*.

[37] C.-M. Feng, Y. Yan, S. Wang, Y. Xu, L. Shao, and H. Fu, "Specificity-preserving federated learning for mr image reconstruction," *IEEE Transactions on Medical Imaging*, vol. 42, no. 7, pp. 2010–2021, 2022.

[38] A. Xu, W. Li, P. Guo, D. Yang, H. R. Roth, A. Hatamizadeh, C. Zhao, D. Xu, H. Huang, and Z. Xu, "Closing the generalization gap of cross-silo federated medical image segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 20 866–20 875.

[39] X. Li, Y. Gu, N. Dvornek, L. H. Staib, P. Ventola, and J. S. Duncan, "Multi-site fmri analysis using privacy-preserving federated learning and domain adaptation: Abide results," *Medical image analysis*, vol. 65, p. 101765, 2020.

[40] H. Wu, Z. Wang, Z. Zhao, C. Chen, and J. Qin, "Continual nuclei segmentation via prototype-wise relation distillation and contrastive learning," *IEEE Transactions on Medical Imaging*, vol. 42, no. 12, pp. 3794–3804, 2023.

[41] F. Amrollahi, S. P. Shashikumar, A. L. Holder, and S. Nemati, "Leveraging clinical data across healthcare institutions for continual learning of predictive risk models," *Scientific reports*, vol. 12, no. 1, p. 8380, 2022.

[42] G. Zheng, V. Braverman, J. Leal, S. Rowe, D. Leung, M. A. Jacobs, and V. S. Parekh, "Towards a collective medical imaging ai: Enabling continual learning from peers," in *Medical Imaging with Deep Learning*, 2024.

[43] D. ARTHUR, "k-means++: the advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, New Orleans, Louisiana, 2007*. Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[44] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 19 586–19 597, 2020.

[45] G. Long, M. Xie, T. Shen, T. Zhou, X. Wang, and J. Jiang, "Multi-center federated learning: clients clustering for better personalization," *World Wide Web*, vol. 26, no. 1, pp. 481–500, 2023.

[46] H. Yao, Y. Wei, J. Huang, and Z. Li, "Hierarchically structured meta-learning," in *International conference on machine learning*. PMLR, 2019, pp. 7045–7054.

[47] R. Yan, L. Qu, Q. Wei, S.-C. Huang, L. Shen, D. L. Rubin, L. Xing, and Y. Zhou, "Label-efficient self-supervised federated learning for tackling data heterogeneity in medical imaging," *IEEE Transactions on Medical Imaging*, vol. 42, no. 7, pp. 1932–1943, 2023.

[48] N. Codella, V. Rotemberg, P. Tschandl, M. E. Celebi, S. Dusza, D. Gutman, B. Helba, A. Kalloo, K. Liopyris, M. Marchetti *et al.*, "Skin lesion analysis toward melanoma detection 2018: A challenge hosted by the international skin imaging collaboration (isic)," *arXiv preprint arXiv:1902.03368*, 2019.

[49] P. Tschandl, C. Rosendahl, and H. Kittler, "The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," *Scientific data*, vol. 5, no. 1, pp. 1–9, 2018.

[50] A. G. Pacheco and R. A. Krohling, "The impact of patient clinical information on automated skin cancer detection," *Computers in biology and medicine*, vol. 116, p. 103545, 2020.

[51] J. Kawahara, S. Daneshvar, G. Argenziano, and G. Hamarneh, "Seven-point checklist and skin lesion classification using multitask multimodal neural nets," *IEEE journal of biomedical and health informatics*, vol. 23, no. 2, pp. 538–546, 2018.

[52] J. Irvin, P. Rajpurkar, M. Ko, Y. Yu, S. Ciurea-Ilcus, C. Chute, H. Marklund, B. Haghgoo, R. Ball, K. Shpanskaya *et al.*, "Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 590–597.

[53] H. Q. Nguyen, K. Lam, L. T. Le, H. H. Pham, D. Q. Tran, D. B. Nguyen, D. D. Le, C. M. Pham, H. T. Tong, D. H. Dinh *et al.*, "Vindr-cxr: An open dataset of chest x-rays with radiologist's annotations," *Scientific Data*, vol. 9, no. 1, p. 429, 2022.

[54] D. Kermany, K. Zhang, and M. Goldbaum, "Large dataset of labeled optical coherence tomography (oct) and chest x-ray images," *Mendeley Data*, vol. 3, no. 10.17632, 2018.

[55] A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny, "Efficient lifelong learning with a-gem," in *International Conference on Learning Representations*, 2018.

[56] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[57] H. Liu, D. Tam, M. Muqeeth, J. Mohta, T. Huang, M. Bansal, and C. A. Raffel, "Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 1950–1965, 2022.

[58] Y. Zhou, M. A. Chia, S. K. Wagner, M. S. Ayhan, D. J. Williamson, R. R. Struyven, T. Liu, M. Xu, M. G. Lozano, P. Woodward-Court *et al.*, "A foundation model for generalizable disease detection from retinal images," *Nature*, vol. 622, no. 7981, pp. 156–163, 2023.

[59] K. Zhang, X. Liu, J. Shen, Z. Li, Y. Sang, X. Wu, Y. Zha, W. Liang, C. Wang, K. Wang *et al.*, "Clinically applicable ai system for accurate diagnosis, quantitative measurements, and prognosis of covid-19 pneumonia using computed tomography," *Cell*, vol. 181, no. 6, pp. 1423–1433, 2020.

[60] L. Wu, J. Zhuang, and H. Chen, "Voco: A simple-yet-effective volume contrastive learning framework for 3d medical image analysis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 22 873–22 882.