pFedRAG: A Personalized Federated Retrieval-Augmented Generation System with Depth-Adaptive Tiered Embedding Tuning

Hangyu He[1] **Xin Yuan**[2,5] **Kai Wu**[3] **Ren Ping Liu**[3] **Wei Ni**[4,5]

[1]School of Computer Science, University of Sydney, Sydney, Australia
[2]Data61, CSIRO, Sydney, Australia
[3]Global Big Data Technologies Centre & School of Electrical and Data Engineering, University of Technology Sydney, Sydney, Australia
[4]School of Engineering, Edith Cowan University, Perth, Australia
[5]School of Computer Science and Engineering, University of New South Wales, Sydney, Australia

Abstract

Large Language Models (LLMs) can undergo hallucinations in specialized domains, and standard Retrieval-Augmented Generation (RAG) often falters due to general-purpose embeddings ill-suited for domain-specific terminology. Though domain-specific fine-tuning enhances retrieval, centralizing data introduces privacy risks. The use of federated learning (FL) can alleviate this to some extent, but faces challenges of data heterogeneity, poor personalization, and expensive training data generation. We propose pFedRAG, a novel Personalized Federated RAG framework, which enables efficient collaborative fine-tuning of embedding models to address these challenges. The key contribution is a new Depth-Adaptive Tiered Embedding (DATE) architecture, which comprises a Global Shared Layer, combined using FL to capture common knowledge, and a Personalized Layer with adjustable depth tailored for local data and training results of each client. The depth is locally controlled based on crafted metrics and scoring criteria. Also, pFedRAG incorporates a fully client-side pipeline leveraging local small LLMs and vector database filtering to construct high-quality query-document pairs. Experiments on diverse medical non-IID document datasets demonstrate that pFedRAG significantly reduces communication costs, handles data heterogeneity, and improves retrieval performance. Human evaluations confirm the enhanced response quality of pFedRAG.

1 Introduction

Large Language Models (LLMs), such as GPT series (Radford et al., 2018, 2019; Brown et al., 2020) and LLaMA (Touvron et al., 2023a,b), have achieved impressive performance across many natural language processing tasks (Zhao et al., 2023). However, LLMs remain susceptible to hallucinations, producing plausible-sounding but factually incorrect content, which is problematic in domains such as healthcare and law (Ji et al., 2023). Retrieval-Augmented Generation (RAG) mitigates hallucinations by incorporating external knowledge. A retriever selects relevant documents from a knowledge base, and the generator conditions its responses on these documents. The effectiveness heavily depends on retrieval quality.

As illustrated in Figure 1(a), conventional RAG systems typically rely on general-purpose embedding models (Lewis et al., 2020a), which often underperform in specialized domains due to their inability to capture domain-specific semantics and terminology. To enhance retrieval accuracy in such domains, fine-tuning embedding models on domain-specific data has proven beneficial (Gururangan et al., 2020); see Figure 1(b). This allows models to learn domain-relevant representations and improve the relevance of retrieved content. However, centralizing such data for fine-tuning raises significant privacy and governance concerns.

Federated Learning (FL) offers a promising framework for domain-specific fine-tuning in collaborative environments. It enables multiple parties to jointly train models without sharing raw data, instead exchanging and aggregating model parameters, thus preserving data privacy. However, applying FL to the fine-tuning of embedding models in RAG systems introduces several unique challenges:

Cost of Full-Parameter Fine-Tuning. Modern embedding models contain hundreds of millions of parameters, making full-model tuning resource-intensive in FL settings. It imposes high communication and computation costs on clients.

Semantic Non-IID Distribution. In real-world deployments, clients often have non-identically distributed data due to differences in institutional focus or data collection practices. This semantic heterogeneity can lead to local overfitting and degrade global model aggregation, especially with algorithms like Federated Averaging.

Supervised Dataset Construction. Effective fine-

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 14255–14268
November 4-9, 2025

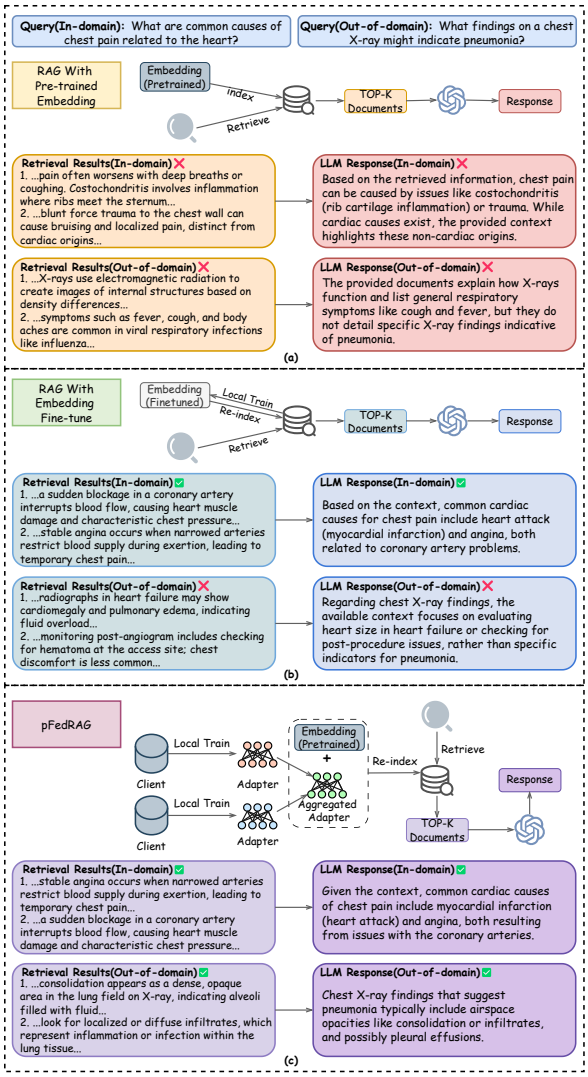

Figure 1: Overview of retrieval-augmented generation workflows under different embedding strategies: Pre-trained, Fine-tuned, and pFedRAG.

tuning requires structured supervised data, specifically for query-document pairs with hard negatives. Generating such data is labor-intensive.

Contributions. In this paper, we propose a novel Personalized Federated RAG (**pFedRAG**) framework designed to address these challenges. Unlike existing federated RAG methods that either perform centralized embedding fine-tuning on pooled data or apply a uniform retrieval mechanism across all clients, pFedRAG integrates a Depth-Adaptive Tiered Embedding (DATE) architecture to balance global knowledge sharing and local personalization, and incorporates a fully client-side supervised dataset generation pipeline to autonomously construct high-quality query–document pairs. The key contributions of this paper are as follows.

Personalized Adaptation to Data Heterogeneity. To address the challenge of semantic non-IID distributions, pFedRAG incorporates the DATE architecture. This includes a shared global layer for knowledge aggregation, a client-specific personalized layer for local adaptation, and a Depth Controller that dynamically adjusts model complexity of the personalized layer. This design allows each client to tailor its retrieval model to its data characteristics, with training performance guiding the dynamic adaptation, improving personalization without sacrificing global generalizability.

Efficient Federated Fine-Tuning. The proposed pFedRAG introduces the Adaptive Dual-Tier Head (ADT-Head), a parameter-efficient architecture that attaches a lightweight, trainable head to a frozen embedding backbone. This design significantly reduces communication and computational overhead, cutting per-round updates to just 4.3% of full model fine-tuning, while preserving strong retrieval performance. It enables the practical deployment of personalized retrieval models in bandwidth-constrained federated settings.

Privacy-Preserving Dataset Generation. We propose a novel client-side pipeline for supervised data generation that avoids central data pooling. It leverages light local LLMs to generate diverse queries and applies vector-based filtering to construct relevant positive and hard negative samples. This approach enables each client to create high-quality retrieval training data autonomously, with reduced annotation costs.

Our proposed pFedRAG is the first framework to integrate personalized embedding tuning through FL into RAG systems, enabling client-level adaptation under data heterogeneity. pFedRAG achieves substantial improvements over traditional pretrained RAG methods, including a 76.0% (local) and 71.6% (global) improvement in Recall@k, and 95.0% of the performance of centralized fine-tuning. Human evaluations validate its impact, with an average score of 8.1 compared to 6.0 for static embeddings, and an 78% expert preference rate. It is evident that the pFedRAG not only advances the status quo of personalized federated retrieval, but also provides strong practical utility in real-world deployment.

2 Related Work

FL for Retrieval-Augmented Generation. FL (McMahan et al., 2017) enables privacy-

preserving collaborative training across decentralized data sources (Kairouz et al., 2021), while RAG (Lewis et al., 2020b) enhances LLM factuality by grounding responses in external knowledge. The integration of these paradigms into FedRAG systems (Jung et al., 2024; Addison et al., 2024) leverages distributed knowledge while maintaining privacy in sensitive domains.

Current FedRAG approaches have addressed various aspects: federated search across distributed clients (Flower, 2025), query overhead reduction through classification-based source selection (Guerraoui et al., 2025), probabilistic search optimization for multi-domain question answering (Shojaee et al., 2025), and privacy enhancement through Confidential Computing (Addison et al., 2024). However, these methods primarily focus on retrieval mechanisms or generator training (Kim et al., 2024; Muhamed et al., 2024), while the optimization of embedding models in FedRAG settings remains virtually unexplored. Our proposed pFedRAG fills this gap by introducing DATE, specifically designed for adapting embedding models within FL contexts.

Personalized FL for Client Heterogeneity. Client heterogeneity in FL encompasses data heterogeneity (non-IID distributions) and system heterogeneity (variations in computational capabilities) (Kairouz et al., 2021), often degrading standard FL algorithm performance. Personalized Federated Learning (PFL) (Tan et al., 2022; Kulkarni et al., 2020) addresses these challenges by customizing models to individual clients while preserving collaborative benefits.

For data heterogeneity, various approaches have been proposed: architectural model decomposition to separate shared and client-specific components (Collins et al., 2021; Arivazhagan et al., 2019), regularization to constrain local updates (Li et al., 2020), meta-learning for rapid client adaptation (Fallah et al., 2020), and client clustering to group similar users (Ghosh et al., 2020). However, these methods often assume uniform model architectures across clients, limiting their applicability in heterogeneous system environments.

For model heterogeneity, researchers have explored knowledge distillation to align diverse architectures (Li and Wang, 2019), parameter importance metrics for dynamic submodel extraction (Su et al., 2024), and Parameter-Efficient Fine-Tuning (PEFT) (Hu et al., 2021) to adapt pre-trained models with reduced parameters. PEFT approaches in FL include homogeneous adapters across varied backbones (Yi et al., 2023) and SVD-based aggregation of different-ranked adaptations (Shen et al., 2024). Managing model heterogeneity dynamically and effectively remains challenging; static decomposition might be suboptimal, and aggregating heterogeneous PEFT parameters can be complex.

3 Problem Formulation

Consider an FL scenario involving N clients, where each client $i \in \mathcal{N} = \{1, \ldots, N\}$ possesses a local dataset $\mathcal{D}_i$ that exhibits significant data heterogeneity. Each client i maintains a dual-tier embedding model, $\Phi_i = \{\theta, \phi^g, \phi_i^p\}$, where θ is a pretrained embedding backbone shared across all clients and remains frozen during training, ϕ^g is a global shared layer updated collaboratively via FedAvg, and ϕ_i^p is a client-specific layer optimized locally to capture personalized information.

Collectively, the clients aim to minimize the average loss, as given by

$$\min_{\phi^g, \{\phi_i^p\}_{i=1}^N} \frac{1}{N} \sum_{i=1}^{N} F_i(\theta, \phi^g, \phi_i^p), \tag{1}$$

where the local objective $F_i(\cdot)$ measures the retrieval embedding quality for client i, based on local query-document pairs sampled from client i's local dataset $\mathcal{D}_i$. Specifically, each client constructs training samples comprising a query, q, and corresponding positive and negative document embeddings, d^+ and d^-.

The retrieval quality is optimized using the InfoNCE loss with an L_2-norm regularization term, which is defined as (van den Oord et al., 2018):

$$F_i = -\mathbb{E}_{(q,d^+,d^-)\sim\mathcal{D}_i}\left[\log \frac{e^{s(q,d^+)}}{e^{s(q,d^+)} + \sum_{d^-} e^{s(q,d^-)}}\right] + \lambda\|\Phi_i\|_2^2, \tag{2}$$

where $s(q, d)$ is the similarity score of embedding vectors, and λ is a regularization factor.

4 Proposed pFedRAG Framework

The objective of pFedRAG is to collaboratively train a retrieval embedding model that captures both globally shared knowledge and personalized features unique to each client's local data distribution. As illustrated in Figure 2, pFedRAG consists of three key aspects: Depth-Adaptive Tiered Embedding (DATE), Federated Learning With Global-Local Adaptation, and Personalized RAG System.

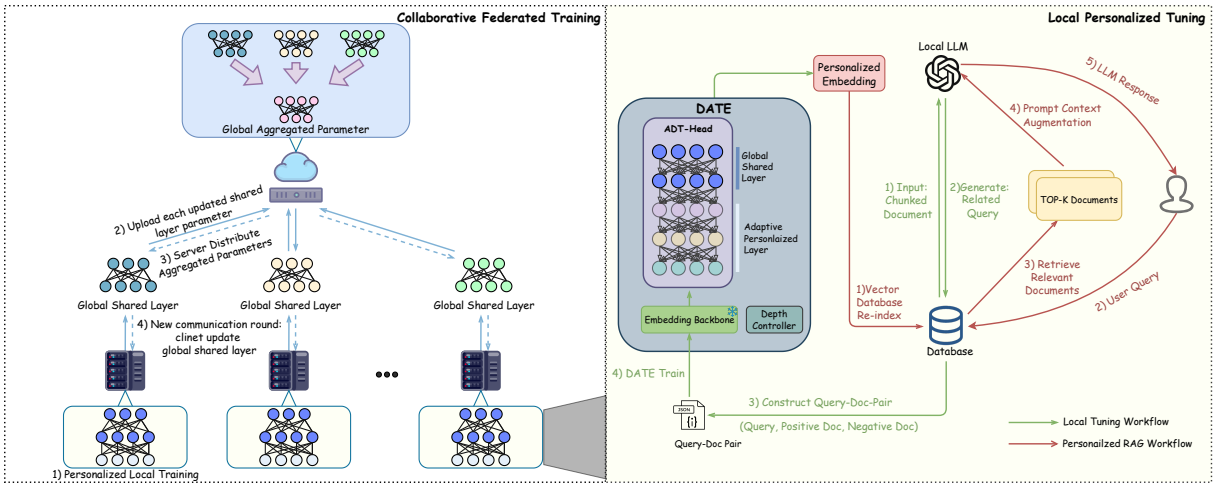

Figure 2: Overall framework of the proposed pFedRAG. The framework combines collaborative federated training of a global shared layer with personalized local training via the DATE architecture. It includes client-side query–document pair generation and a personalized RAG workflow using local LLMs for domain-adaptive retrieval.

4.1 Depth-Adaptive Tiered Embedding

DATE is the architectural foundation of pFedRAG's retrieval model, designed to balance shared representation learning with client-specific personalization. As depicted in Figure 3, it has three components: (i) Embedding Backbone (θ), (ii) ADT-Head, and (iii) Depth Controller. These components work in concert to create a flexible, personalized embedding architecture that adapts to the unique characteristics of each client's data.

Embedding Backbone (θ). The adaptive embedding backbone is built on the pretrained `e5-base-v2` model (Wang et al., 2022), an adaptable text embedding model optimized for retrieval, clustering, and classification tasks. This backbone provides a universal semantic encoder shared across all clients and remains frozen during training, serving as a foundation for embedding computations while enabling adaptation through personalized retrieval.

ADT-Head. This component is designed to efficiently balance global knowledge sharing with client-specific adaptation. This bifurcated structure processes embeddings from the backbone, minimizing communication overhead while preserving personalization capabilities. The ADT-Head contains two complementary layers:

Global Shared Layer (ϕ^g). This layer forms the first stage of ADT-Head and is applied to the embeddings output by θ. It is trained collaboratively across clients to extract generalized, transferable features that support effective federated aggregation. It comprises layer normalization (Ba et al., 2016), dropout (Srivastava et al., 2014), and a pair of linear transformations interconnected by a nonlinear activation (e.g., GELU (Hendrycks and Gimpel, 2016)). By first expanding embedding dimensionality from 1D to 4D and then compressing it to 1D, ϕ^g enhances the stability and generalizability of shared knowledge for robust cross-client representation learning.

Personalized Layer (ϕ_i^p). This layer refines the shared representation to align with each client's local data distribution. It introduces flexibility in model expressiveness by supporting three configurable complexity levels based on varying local data complexities:

- **Base Layer** ($\mathcal{L}_{\text{base}}$)**.** A lightweight configuration with a single linear layer and activation, designed for clients with low data complexity.
- **Advanced Layer** ($\mathcal{L}_{\text{adv}}$)**.** This layer enhances capacity by stacking two linear layers, first expanding to 2D and then projecting back to 1D, thereby allowing for improved personalization for moderate data complexity.
- **Extended Layer** ($\mathcal{L}_{\text{ext}}$)**.** This layer integrates a Self-Attention Interaction Module between the linear transformations. It expands embeddings from 1D to 2D, applies multi-head self-attention (Vaswani et al., 2017), and compresses the result back to 1D, making it suitable for clients with complex or diverse data.

This compact tiered structure significantly reduces communication overhead compared to full-model tuning, making it suitable for FL scenarios.

Depth Controller. We also develop the Depth Controller to dynamically govern personalized layer complexity during training. This component analyzes client data characteristics and monitors training dynamics to determine optimal model capacity, balancing expressiveness and efficiency. The Depth Controller operates via two modules:

Initial Depth Assigner (IDA). The IDA employs a Complexity Scoring Unit to evaluate client data characteristics before training. It sets the Personalized Layer type of client i as $\mathcal{L}_i^{(0)}$ based on its local data complexity score S_i. Here, S_i is calculated by each client i based on local data properties:

$$S_i = w_1 \cdot D_i + w_2 \cdot (\alpha L_{\text{avg},i} + \beta TTR_{\text{norm},i}) + w_3 \cdot PPL_i, \tag{3}$$

where $L_{\text{avg},i}$ is the average token length per document, $TTR_{\text{norm},i}$ is the normalized type-token ratio, and PPL_i is the perplexity (Jelinek et al., 1977) computed over $\mathcal{D}_i$. w_1, w_2, w_3, α, and β are weighting coefficients. This module enables assignment of a suitable initial complexity level to each client based on data characteristics.

Dynamic Depth Scheduler (DDS). To enable clients to progressively refine layer complexity beyond initial assignments, we design the DDS with two units that jointly adapt model complexity during training:

- **API Metrics Analysis Unit:** This unit evaluates the Adaptation Performance Index (API) for each client in fixed time windows to determine when complexity changes are needed. The API combines two key training indicators:

$$\text{API}_{i,t} = w_L \, \Delta L_{i,t}^{\text{norm}} - w_O \, O_{i,t}^{\text{norm}}, \tag{4}$$

  where $\Delta L_{i,t}^{\text{norm}}$ measures normalized training loss reduction (learning momentum), and $O_{i,t}^{\text{norm}}$ quantifies the normalized performance gap between training and validation data (overfitting penalty). Weights w_L and w_O balance these components (with $w_L + w_O = 1$).

  The API trajectory determines whether a layer complexity adjustment is necessary. Layer adjustments are triggered when $\text{API}_{i,t}$ consistently falls outside its dynamic performance band, bounded by thresholds $T_{\text{up}}^{(i,t)}$ and $T_{\text{down}}^{(i,t)}$. $c_s^{(i,t)}$ and $c_l^{(i,t)}$ track consecutive instances of over- or under-performance, guiding upgrade or downgrade decisions. (See Appendix B.1 for details.)

- **Knowledge Distillation Unit:** When a complexity change is triggered, this unit facilitates a smooth transition between different model architectures. It treats the current model as teacher and the newly adjusted model as student, transferring knowledge (Hinton et al., 2015) to ensure the model maintains performance while adapting to its new complexity level. This prevents drastic performance drops during architectural transitions and enables efficient adaptation.

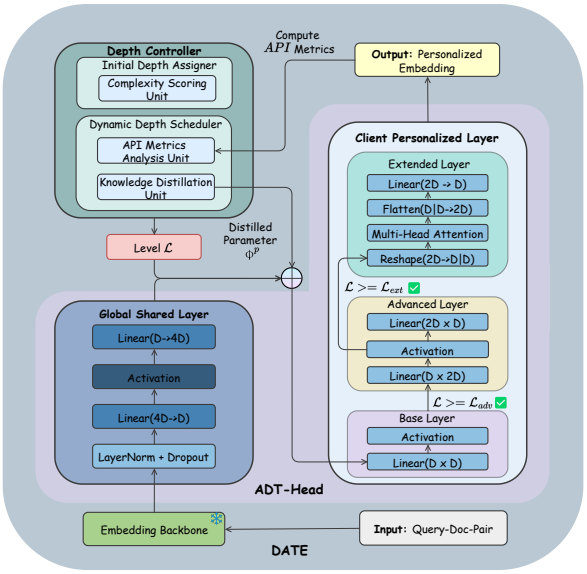

Figure 3: Architecture of DATE in pFedRAG, showing the Embedding Backbone, ADT-Head with its Global and Personalized layers, and Depth Controller with its IDA and DDS modules.

4.2 FL With Global-Local Adaptation

The training process comprises three phases: (i) Model Customization via IDA, (ii) Local Tuning with DDS, and (iii) Global Aggregation on Server.

Model Customization via IDA. At the start of each communication round, the IDA of client i computes a local data complexity score S_i, which is uploaded to the server. After applying min-max normalization across all clients to map scores into the [0,1] range, the server assigns an initial personalized layer configuration $\mathcal{L}_i^0$ based on the normalized complexity score S_i^{norm}:

$$\mathcal{L}_i^0 = \begin{cases} \mathcal{L}_{\text{base}}, & 0 < S_i^{\text{norm}} \leq 0.33, \\ \mathcal{L}_{\text{adv}}, & 0.33 < S_i^{\text{norm}} \leq 0.67, \\ \mathcal{L}_{\text{ext}}, & 0.67 < S_i^{\text{norm}} \leq 1. \end{cases} \tag{5}$$

The thresholds 0.33 and 0.67 divide the normalized range into three equal intervals, corresponding to the Base, Advanced, and Extended layers. This serves as a reasonable initialization, which the Depth Controller further refines during training.

Each client receives the full model but activates only its assigned personalized layer.

Local Tuning with DDS. We put forth an adaptive training strategy at each client, as described in Algorithm 1 of Appendix A.1. During training, the DDS of Depth Controller continuously monitors training dynamics using $\text{API}_{i,t}$ over the fixed time windows T_w. Based on the API metrics and corresponding counters, the Depth Controller determines whether to adjust the layer complexity based on the following decision rule:

$$\text{decision}_{i,t} = \begin{cases} \text{"upgrade"}, & \text{if } c_s^{(i,t)} \geq \tau_s \wedge \mathcal{L}_{i,t-1} \neq \mathcal{L}_{\text{ext}}, \\ \text{"downgrade"}, & \text{if } c_l^{(i,t)} \geq \tau_l \wedge \mathcal{L}_{i,t-1} \neq \mathcal{L}_{\text{base}}, \\ \text{"none"}, & \text{otherwise}, \end{cases} \tag{6}$$

where τ_s is the minimum number of consecutive rounds showing stable improvement required for an upgrade, and τ_l is the maximum number of consecutive rounds showing performance decline tolerated before a downgrade.

When an adjustment is triggered (i.e., decision is not "none"), we propose a novel knowledge preservation mechanism through distillation. This adaptive distillation phase employs the current model as a teacher and the adjusted model as a student:

$$L_{\text{KD},i}(\phi_i^s, \phi_i^t) = -\sum_{x \sim \mathcal{D}_i} p_t(x) \log p_s(x), \tag{7}$$

where ϕ_i^s and ϕ_i^t represent the student and teacher parameters on client i respectively; $p_s(x)$ and $p_t(x)$ are the corresponding output distributions.

During distillation, only the personalized layer parameters ϕ^p is updated while the global shared layer ϕ^g remains frozen:

$$\phi^p \leftarrow \arg\min_{\phi^p} L_{\text{KD},i}(\phi^s, \phi^t). \tag{8}$$

After adaptation, the controller enters a cooling period of T_{cool}, during which it continues to monitor API values but temporarily suspends further structural changes to prevent oscillations.

Global Aggregation on Server. Once local training completes, clients upload only their global shared layer parameters ($\phi_{i,t}^g$) to the server. The server then performs standard federated averaging:

$$\phi_{t+1}^g \leftarrow \frac{1}{|\mathcal{A}|} \sum_{i \in \mathcal{A}} \phi_{i,t}^g. \tag{9}$$

This aggregated global layer is then redistributed to all clients for the next round, while personalized layers (ϕ_i^p) remain local, preserving both personalization and data privacy, as shown in Figure 2.

4.3 Personalized RAG System

The Personalized RAG System uses client-specific embeddings to enhance retrieval relevance. It covers from reconstructing local vector databases to generating context-aware responses via RAG, as shown in Figure 2.

Vector Database Reconstruction. After federated training completes, each client reconstructs its local vector database using personalized embedding model Φ_i (combining frozen backbone with trained ADT-Head). We encode local documents with this model and index the vectors into an optimized vector database (Milvus (Milvus Team, 2019–Present)), improving retrieval accuracy for client-specific data distributions.

Retrieval and Generation. During inference, user queries are encoded with the same personalized embedding model and used to retrieve the top-K relevant document chunks via vector similarity search. These chunks provide contextual knowledge injected into a domain-aware prompt template shown in Table 1. This specially designed prompt bridges retrieved content with the generation capabilities of the local LLM, ensuring responses are contextually grounded and aligned with client-specific domain knowledge.

Domain-Aware Prompt for RAG Inference

Given a user query related to a medical domain, retrieve the most relevant document chunks from the local vector database and use them as context to generate a detailed and informative response. Ensure that the response is coherent and accurately reflects the retrieved information.

`In-Context Few-shot Example`

Query: {User Query}
Retrieved Documents: {Top-K Retrieved Chunks}
Response:

Table 1: LLM Prompt for Personalized RAG.

5 Experiments

We evaluate the proposed pFedRAG framework on a medical document dataset derived from multiple research domains to simulate realistic clinical and research-oriented retrieval scenarios. Due to space limitations, detailed experimental settings are provided in Appendix D.

5.1 Medical Document Datasets Preparation

Data Collection. We construct our dataset by collecting English papers from the PubMed Central

database (National Center for Biotechnology Information (NCBI), Accessed on May 18, 2025) across six medical domains: Cardiology (3125 papers), Medical Informatics (2500), Neuroscience (2188), Oncology (1875), Pharmacy (1563), and Radiology (1250). To ensure dataset quality, we exclude non-peer-reviewed publications, speeches, and incomplete documents. We retain only retrieval-relevant sections (title, abstract, introduction, discussion, conclusion) while removing non-textual elements and privacy-sensitive personal data.

Query-Doc Pair Generation. We segment documents using the e5-base-v2 tokenizer with 512 maximum tokens per chunk, ensuring contextual coherence and compatibility with the embedding model. We leverage a light LLM (QWen2.5-7b-instruct (Qwen Team, 2024)) to generate two diverse queries per document chunk, capturing varied query intents (the prompt used for query generation is provided in Table 6 of Appendix C). This process yields 32619 query-document pairs, with 80% for federated training and 20% for evaluation.

Data Heterogeneity. To simulate realistic non-IID distributions, we partition the dataset across six clients using a Dirichlet distribution ($\alpha = 0.3$) (Hsu et al., 2019), creating significant data heterogeneity that realistically emulates federated environments.

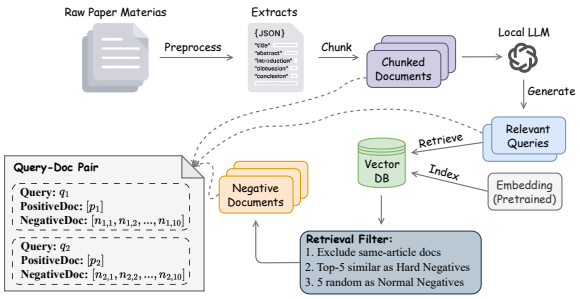

Figure 4: The process of query-doc-pair construction.

Contrastive Pair Construction. For each query, we use its original document chunk as the positive sample. Negative samples are selected through local retrieval from each client's vectorized corpus, excluding chunks from the same source article. We retrieve the top 5 most similar chunks as hard negatives and randomly sample 5 additional documents as normal negatives, maintaining a 1:10 positive-to-negative ratio. We further employ in-batch negative sampling with a batch size of 160, significantly enhancing training effectiveness by increasing negative sample diversity.

5.2 Evaluation Metrics

We evaluate retrieval performance using four metrics: (i) **Recall@k** (proportion of relevant documents in top-k results) (Manning et al., 2008), (ii) **MRR** (position of first relevant document) (Voorhees, 1999), (iii) **NDCG** (ranking quality considering relevance and position) (Järvelin and Kekäläinen, 2002), and (iv) **Average Rank** (average position of relevant documents, lower is better). With one positive sample per query in our setup, Recall@k and NDCG essentially indicate whether the relevant document appears within the top-k results. Detailed definitions of the metrics are provided in Appendix B.2.

5.3 Main Results

To our knowledge, this work is the first to explore adaptive complexity embedding personalization for federated RAG, making direct comparisons with existing algorithms impossible. We therefore constructed baselines representing best practices across the spectrum of embedding model complexity. Since DATE incorporates dynamic adaptation of model complexity to match client characteristics, we first compare it with several complexity-invariant baselines. All architectures shown in Table 2 are built upon pretrained `e5-base-v2` Embedding Backbone (EB), with various configurations: (i) Pretrained Embedding (frozen EB alone); (ii) EB+Global Shared Layer; and (iii) configurations with invariant personalized layer (Base/Advanced/Extended) added on top of both EB and global shared layer.

Results demonstrate DATE's superiority across all test sets. DATE achieves substantial improvements in both local and global evaluations, improving Recall@k by 76.0% (local) and 71.6% (global) and MRR by 70.2% (local) and 71.4% (global) over the Pretrained Embedding baseline. Compared to the best-performing complexity-invariant configuration (EB+Base Layer), DATE still shows consistent gains of 2.2% (local) and 0.7% (global) in Recall@k. These advantages confirm that our adaptive architecture's dynamic complexity adjustment enables effective personalization while maintaining strong generalization capabilities.

5.4 Ablation Study

Does Dynamic Depth Scheduler (DDS) matter? We analyze the DDS effectiveness by comparing it with a static layer allocation strategy. As shown in

Table 2: Performance Comparison of Embedding Architectures (Top-K=5). All results are averaged over three independent runs. Reported values are means with 95% confidence intervals computed using the t-distribution (n=3). Abbreviations: PT=Pretrained EB (frozen), GSL=Global Shared Layer, Adv=Advanced, Ext=Extended.

Method	Recall@k	MRR	NDCG	AvgRank	Recall@1
Local Test Set					
PT (EB only)	0.484±0.007	0.309±0.006	0.323±0.008	14.823±0.288	0.159±0.005
EB+GSL	0.798±0.002	0.496±0.002	0.552±0.001	4.330±0.081	0.295±0.001
EB+Base	0.834±0.004	0.520±0.003	0.583±0.005	3.967±0.165	0.317±0.003
EB+Adv	0.816±0.005	0.504±0.006	0.562±0.006	4.197±0.213	0.301±0.004
EB+Ext	0.792±0.008	0.488±0.009	0.542±0.007	4.471±0.314	0.287±0.006
DATE	**0.852±0.003**	**0.526±0.003**	**0.588±0.004**	**3.834±0.119**	**0.319±0.003**
Global Test Set					
PT (EB only)	0.472±0.010	0.301±0.009	0.314±0.011	15.333±0.399	0.152±0.007
EB+GSL	0.768±0.002	0.492±0.002	0.538±0.002	5.750±0.075	0.290±0.002
EB+Base	0.804±0.005	**0.516±0.006**	**0.571±0.006**	4.896±0.198	0.312±0.004
EB+Adv	0.779±0.007	0.497±0.008	0.546±0.009	5.514±0.285	0.296±0.005
EB+Ext	0.757±0.011	0.482±0.012	0.529±0.010	5.986±0.441	0.282±0.008
DATE	**0.810±0.004**	**0.516±0.004**	**0.571±0.005**	**4.897±0.145**	**0.318±0.002**

Table 3: Effectiveness of DDS (Top-K=5)

Method	Recall@k	MRR	NDCG	AvgRank	Recall@1
Local Test Set					
Pretrained Embedding	0.484	0.309	0.323	14.823	0.159
DC w/o DDS	0.809	0.503	0.560	4.233	0.302
DC w/ DDS	**0.852**	**0.526**	**0.588**	**3.834**	**0.319**
Global Test Set					
Pretrained Embedding	0.472	0.301	0.314	15.333	0.152
DC w/o DDS	0.781	0.500	0.551	5.510	0.292
DC w/ DDS	**0.810**	**0.516**	**0.571**	**4.897**	**0.318**

Table 3, DDS delivers significant improvements on both local and global test sets - increasing NDCG by 5.0% (local) and 3.6% (global) while reducing average rank by 9.4% (local) and 11.1% (global). It is evident that dynamically adjusting layer complexity based on real-time training metrics substantially enhances performance by adapting to each client's evolving needs beyond initial assignments.

5.5 Effectiveness Evaluation

Federated vs Centralized Training Effectiveness. We compare our federated approach with centralized training using the Global Shared Layer. As shown in Table 4, both methods substantially outperform the pretrained baseline. While centralized training shows marginal advantages in each metric, federated training maintains robust performance (95.0% of centralized Recall@k), despite FL's inherent data heterogeneity. This small performance gap confirms our approach effectively balances privacy with distributed knowledge utilization.

Table 4: Federated vs Centralized Training (Top-K = 5)

Training Mode	Recall@k	MRR	NDCG	AvgRank	Recall@1
Pretrained Embedding	0.472	0.301	0.314	15.333	0.152
Centralized Training	**0.808**	**0.524**	**0.582**	**4.297**	**0.332**
Federated Training	0.768	0.492	0.538	5.750	0.290

Table 5: Human Evaluation of End-to-End RAG Effectiveness

Method	Avg. Score	W/T/L	Preferred (%)
Pretrained Embedding	6.0	-	-
DATE	**8.1**	32 / 11 / 7	**78%**

End-to-End RAG Effectiveness via Human Evaluation. We conducted a human evaluation of our RAG system using QWen2.5-7B-Instruct to generate responses from Top-K=5 documents retrieved by either DATE or Pretrained Embedding. Three domain experts blindly evaluated 50 response pairs on correctness, completeness, and coherence. As shown in Table 5, DATE significantly outperforms the baseline (8.1 vs 6.0 average score) with a favorable Win/Tie/Loss ratio of 32/11/7. Experts noted DATE's responses contained more comprehensive coverage of medical concepts with fewer factual errors, confirming that improved retrieval directly translates to better response quality.

We also examined the reliability of human ratings and the stability of model performance. Expert agreement is substantial (Krippendorff's α=0.82), indicating that annotators followed the rubric consistently and that the evaluation results are reproducible. Furthermore, across the 50 items, DATE (pFedRAG) shows tighter score dispersion (8.1 ± 0.95) than the Pretrained Embedding baseline (6.0 ± 1.75). This proves that our DATE does deliver more stable performance across questions.

6 Conclusion

In this paper, we presented pFedRAG to enhance RAG systems in specialized domains while addressing privacy concerns and resource limitations. Our approach tackles key challenges in federated settings, including the high cost of full-model tuning, semantic divergence across heterogeneous client data, and the need for high-quality supervised datasets. We introduced DATE, a comprehensive architecture comprising ADT-Head (a parameter-efficient structure that combines a global shared layer for common knowledge aggregation with dynamically adjusted personalized layers) and Depth

Controller for guiding adaptive complexity adjustments. We also proposed a client-side pipeline leveraging local LLM and vector database filtering for privacy-preserving dataset construction. Experimental evaluations demonstrated that pFedRAG significantly reduces communication and computation costs, effectively handles data heterogeneity through adaptive model complexity, and improves end-to-end RAG performance compared to standard baselines, showcasing its practical viability for collaborative, privacy-conscious enhancement of client-personalized RAG systems.

7 Limitations

Non-Federated Generative Component. Our framework currently personalizes only the retrieval side, leaving the generation component as a standard pre-trained LLM without client-specific adaptation. This may limit response quality in specialized domains. Future work could explore parameter-efficient federated fine-tuning techniques like LoRA adapters for the generation component, enabling end-to-end personalization while maintaining privacy.

Static Hard Negative Sampling Strategy. We employ one-time hard negative mining before training with in-batch negative sampling during iterations. As embeddings evolve, initially identified hard negatives may become less challenging. An iterative re-mining strategy that periodically updates hard negatives based on current embedding spaces could further enhance retrieval performance.

Future Improvements for Dataset Generation. Our client-side pipeline uses lightweight LLMs to generate queries while preserving privacy and accommodating resource constraints. Though effective, query quality might not match that of larger models. Future work could explore privacy-preserving mechanisms to leverage larger LLM capabilities through secure APIs, potentially enhancing dataset quality without compromising privacy.

8 Ethics Statement

This study uses only publicly available data from the PubMed Central Open Access Subset and involves no human subjects or personal data, thus requiring no additional ethical approval. Despite these safeguards, the system could potentially generate inaccurate medical information. We recommend professional reviews of outputs before clinical applications and the implementation of fact-checking mechanisms during deployment.

9 Acknowledgement

We thank the anonymous reviewers and the area chair for their constructive feedbacks.

References

Parker Addison, Minh-Tuan H. Nguyen, Tomislav Medan, Jinali Shah, Mohammad T. Manzari, Brendan McElrone, Laksh Lalwani, Aboli More, Smita Sharma, Holger R. Roth, Isaac Yang, Chester Chen, Daguang Xu, Yan Cheng, Andrew Feng, and Ziyue Xu. 2024. C-FedRAG: A confidential federated retrieval-augmented generation system. In *arXiv preprint arXiv:2412.13163*.

Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Anuvabh Singh, and Sunav Choudhury. 2019. Federated learning with personalization layers. In *arXiv preprint arXiv:1912.00818*.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Liam Collins, Hamed Qi, Mohammad Ghassemi, and Salman Avestimehr. 2021. Exploiting shared representations for personalized federated learning. In *International Conference on Machine Learning*, pages 2089–2099. PMLR.

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

Flower. 2025. Federated retrieval augmented generation (FedRAG) example. https://flower.ai/docs/examples/fedrag.html. Accessed on [Insert Access Date].

Avishek Ghosh, Justin Chung, Dong Yin, and Kannan Ramchandran. 2020. An efficient framework for clustered federated learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 19586–19597.

Rachid Guerraoui, Anusha Gupta, Andreas Hellander, Anne-Marie Kermarrec, Nikola Logic, and Rafael Plassier. 2025. Efficient federated search for retrieval-augmented generation. In *Proceedings of the EuroMLSys Conference*. Based on arXiv:2502.19280.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Tzu-Ming Henry Hsu, Hang Qi, and Matthew Brown. 2019. Measuring the effects of non-identical data distribution for federated visual classification. In *International conference on learning representations*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

Fred Jelinek, Robert L Mercer, Lalit R Bahl, and James K Baker. 1977. Perplexity—a measure of the difficulty of speech recognition tasks. In *The Journal of the Acoustical Society of America*, volume 62, pages S63–S63. Acoustical Society of America.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Jincheol Jung, Hongju Jeong, and Eui-Nam Huh. 2024. Federated learning and RAG integration: A scalable approach for medical large language models. In *arXiv preprint arXiv:2412.13720*.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. 2021. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210.

Eugenia Kim, Jingjing Wang, and Shandong Wu. 2024. Federated learning-enhanced retrieval augmented generation (RAG). Technical Report 8089, Technical Disclosure Commons.

Vinayak Kulkarni, Milind Kulkarni, and Anirudh Pant. 2020. Survey of personalization techniques for federated learning.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Mazar Komeili, et al. 2020a. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Myle Ott, Wen-tau Chen, Alexis Conneau, et al. 2020b. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in*

Neural Information Processing Systems, volume 33, pages 9459–9474.

Daliang Li and Junpu Wang. 2019. FedMD: Heterogenous federated learning via model distillation. In *arXiv preprint arXiv:1910.03581*.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems*, volume 2, pages 429–450.

Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*. Cambridge university press.

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

Milvus Team. 2019–Present. Milvus: A cloud-native vector database for scalable similarity search. https://milvus.io.

Aashiq Muhamed, Ting Zhao, Ahmad Beirami, and Ananda Theertha Suresh. 2024. Cache me if you can: The case for retrieval augmentation (RA) in federated learning. In *ICLR 2024 Workshop on Federated Learning*.

National Center for Biotechnology Information (NCBI). Accessed on May 18, 2025. PubMed Central.

Qwen Team. 2024. Qwen2.5 Technical Report. https://qwenlm.github.io/blog/qwen2.5/.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/language-unsupervised/language understanding paper. pdf*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Han Shen, Lichao Zhang, Han Yu, and Xiaoxiao Liu. 2024. FlexLoRA: A flexible aggregation scheme for federated fine-tuning of large language models. In *International Conference on Learning Representations*.

Parshin Shojaee, Shuai Wang, Smita Sharma, Chenguang Wang, Xiaochuan Liu, and Holger R. Roth. 2025. Federated retrieval augmented generation for multi-product question answering. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. Based on arXiv:2501.14998.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Lili Su, Connor McLaughlin, and Lichao Zhang. 2024. Federated importance-aware submodel extraction. In *Advances in Neural Information Processing Systems*, volume 37. Based on NeurIPS 2024 paper.

Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. 2022. Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *Preprint*, arXiv:1807.03748.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, volume 30.

Ellen M Voorhees. 1999. The trec-8 question answering track report. In *Proceedings of the eighth Text REtrieval Conference (TREC-8)*, volume 99, pages 77–82. National Institute of Standards and Technology (NIST).

Liang Wang, Nan Yang, Ruty Fariha, Fnu Mi, and Bo Zhu. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Liping Yi, Han Yu, Chao Ren, Heng Zhang, Gang Wang, Xiaoguang Liu, and Xiaoxiao Li. 2023. pFedLoRA: Model-heterogeneous personalized federated learning with LoRA tuning. In *arXiv preprint arXiv:2310.19978*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

A ALGORITHM

A.1 Federated Tuning Procedure

Algorithm 1 summarizes the comprehensive algorithm for federated tuning described in Section 4.2.

Algorithm 1: Federated Tuning Procedure

Input : clients $\mathcal{N}$, global rounds T, local epochs E, window size T_w, cooling period T_{cool}, learning rate η, initial parameters (ϕ^g, ϕ^p)
Output : Globally optimized ϕ^g, locally personalized ϕ_i^p

for *each round* $t = 1, 2, ..., T$ **do**
  Sample client set $\mathcal{A} \subseteq [N]$ Send global shared layer ϕ_t^g to clients $i \in \mathcal{A}$ **for** *each client* $i \in \mathcal{A}$ ***in parallel*** **do**
    Initialize local model $(\phi_t^g, \phi_{i,t}^p)$ **for** *epoch* $e = 1, 2, ..., E$ **do**
      Compute local loss F_i via (2) Update $(\phi_t^g, \phi_{i,t}^p) \leftarrow (\phi_t^g, \phi_{i,t}^p) - \eta \nabla F_i$
    Compute Adaptation Performance Index $\text{API}_{i,t}$;
    Update API history buffer $\mathcal{H}_i \leftarrow \mathcal{H}_i \cup \{\text{API}_{i,t}\}$;
    if $|\mathcal{H}_i| \geq T_w$ **then**
      Compute thresholds $T_{\text{up}}^{(i,t)}$, $T_{\text{down}}^{(i,t)}$ based on recent T_w entries in $\mathcal{H}_i$;
      Update counters $c_s^{(i,t)}$, $c_l^{(i,t)}$ based on $\text{API}_{i,t}$;
      if *adjustment condition met via* (6) *and not in cooling period* **then**
        Perform layer adjustment via KD with the current model as Teacher;
        During KD, freeze ϕ^g and update only ϕ^p;
        Start cooling period T_{cool};
    Send updated global layer $\phi_{i,t}^g$ to server
  Aggregate global layer: $\phi_{t+1}^g = \frac{1}{|\mathcal{A}|} \sum_{i \in \mathcal{A}} \phi_{i,t}^g$

B FORMULATION

B.1 API Metrics

Learning Momentum and Overfitting Penalty. The learning momentum $\Delta L_{i,t}$ is calculated as the ratio of loss reduction over consecutive windows:

$$\Delta L_{i,t} = \frac{L_{i,t-2T_w} - L_{i,t}}{L_{i,t-2T_w} - L_{i,t-T_w}}, \tag{10}$$

where T_w is the window size and $L_{i,t}$ is the loss at time step t for client i.

The overfitting score $O_{i,t}$ measures the difference between training and validation performance gains:

$$O_{i,t} = \max\big(0, \Delta\text{Recall}_{i,t}^{\text{train}} - \Delta\text{Recall}_{i,t}^{\text{test}}\big), \tag{11}$$

where $\Delta\text{Recall}_{i,t}^{\text{train}}$ and $\Delta\text{Recall}_{i,t}^{\text{test}}$ are calculated using the same window-based approach as $\Delta L_{i,t}$.

Adaptive Thresholds and Counter Updates. The dynamic thresholds for the API values are calculated as follows:

$$\mu_{i,t} = \frac{1}{T_w} \sum_{k=1}^{T_w} \text{API}_{i,t-k+1}, \tag{12}$$

$$\sigma_{i,t} = \text{std}(\text{API}_{i,t-T_w+1:t}), \tag{13}$$

$$\delta_{i,t} = \min(\sigma_{i,t}, 0.1|\mu_{i,t}|), \tag{14}$$

$$T_{\text{up}}^{(i,t)} = \mu_{i,t} + \delta_{i,t}, \tag{15}$$

$$T_{\text{down}}^{(i,t)} = \mu_{i,t} - \delta_{i,t}. \tag{16}$$

The counters for tracking consistent performance patterns are updated according to:

$$\begin{cases} c_s^{(i,t)} = c_s^{(i,t-1)} + 1,\ c_l^{(i,t)} = 0, & \text{if } \text{API}_{i,t} > T_{\text{up}}^{(i,t)}, \\ c_l^{(i,t)} = c_l^{(i,t-1)} + 1,\ c_s^{(i,t)} = 0, & \text{if } \text{API}_{i,t} < T_{\text{down}}^{(i,t)} \\ c_s^{(i,t)} = c_l^{(i,t)} = 0, & \text{otherwise.} \end{cases} \tag{17}$$

B.2 Evaluation Metrics

The evaluation metrics used in our experiments are formally defined as follows:

$$\text{Recall@k} = \frac{|\{d^+\} \cap R_K(q)|}{|\{d^+\}|}, \tag{18}$$

$$\text{MRR} = \frac{1}{|\mathcal{Q}|} \sum_{q=1}^{|\mathcal{Q}|} \frac{1}{\text{rank}_q}, \tag{19}$$

where $\{d^+\}$ denotes the single positive document for query q, $R_K(q)$ is the set of top-K documents retrieved for q, $\mathcal{Q}$ is the set of all queries, and rank_q is the position of the positive document in the ranking.

$$\text{NDCG} = \frac{1}{\text{IDCG}_K} \sum_{j=1}^{K} \frac{2^{\text{rel}_j} - 1}{\log_2(j+1)}, \tag{20}$$

$$\text{AvgRank} = \frac{1}{|\mathcal{Q}|} \sum_{q=1}^{|\mathcal{Q}|} \text{rank}_q, \tag{21}$$

where $\text{rel}_j \in \{0, 1\}$ indicates the relevance of the document at rank j and IDCG_K is the maximum possible DCG for an ideal ranking.

Since a query has exactly one positive sample, i.e., $\text{IDCG}_K = 1$, Recall and NDCG are binary indicators of whether the true document is within Top-K, while MRR and AvgRank are directly determined by the position of that relevant item.

C PROMPT

In this section, we detail the prompt required for our query generation process. For the query-document pair generation described in Section 5, we utilize a structured prompt with QWen2.5-7b-instruct. This prompt is designed to generate 2 diverse and realistic search queries that a user might ask when seeking information contained in the specific medical document chunk. The prompt template is as follows:

Medical Query Generation Prompt for Searching Document Chunks

You are a medical literature retrieval expert. Your task is to generate exactly 2 search queries based on the following document passage. First, identify the two main themes or core aspects discussed in the document passage. These should reflect the central topics, conditions, treatments, or research questions. Consider focusing on the title, abstract, or key sections to pinpoint these themes. Then, generate two concise yet informative search queries, each focusing on one of the identified themes. Ensure that each query has a distinct search intent and targets a different main theme. Avoid overlap in focus. Queries should be specific and tied to the document's content, avoiding broad or generic terms, to retrieve literature relevant to its core contributions. Do not quote the passage directly; instead, abstract core concepts and rephrase them using keywords and terminology researchers or clinicians would use. Consider what researchers or clinicians would search for to find related or expanded studies. Remain objective, avoiding personal biases or assumptions. Output exactly 2 queries, each on a separate line.

```
Input:   {Document}
Output:  {First Query Here}
         {Second Query Here}
```

Table 6: Prompt template for generating medical chunks search queries

D EXPERIMENTS

All training-based experiments were conducted on 6 NVIDIA RTX Ada 6000 GPUs. Results are reported as the mean over three independent runs to ensure consistency and mitigate randomness.

D.1 License Discussion

In this study, we used the PubMed Central Open Access Subset, whose articles are available under various Creative Commons licenses (e.g., CC0, CC BY, CC BY-SA, CC BY-NC), the Milvus vector database under the Apache License 2.0 (with preservation of LICENSE and NOTICE files on redistribution), and the E5-base-v2 model (intfloat/e5-base-v2) under the MIT License (permitting free use, modification, and redistribution with copyright notice intact). Our use of these artifacts is consistent with their intended research purposes. For the artifacts we create, including embeddings and models derived from these resources, we specify that they are intended for research purposes only and maintain compatibility with the original access conditions of the source materials. Any derivative works produced during this research are not intended for commercial or production use outside research contexts.

Table 7: Distribution of medical domains across federated clients (%) after Dirichlet partitioning ($\alpha = 0.3$)

Client	Card.	Rad.	Med. Info.	Pharm.	Neuro.	Onc.
C1	18.65	44.96	0.38	0.00	35.77	0.25
C2	99.71	0.14	0.01	0.10	0.03	0.02
C3	0.34	0.44	74.25	0.11	0.00	24.86
C4	0.23	25.98	9.44	12.66	51.69	0.00
C5	26.71	0.34	6.83	13.73	36.70	15.69
C6	0.00	12.59	0.18	66.61	7.08	13.53

D.2 Datasets Statistics

The original dataset consists of 32,619 query-document pairs distributed across six medical domains as follows: Cardiology (9,594 pairs), Radiology (1,919 pairs), Medical Informatics (8,315 pairs), Pharmacy (2,558 pairs), Neuroscience (5,756 pairs), and Oncology (4,477 pairs).

To simulate realistic non-IID scenarios in federated learning environments, we employed a Dirichlet distribution ($\alpha = 0.3$) to partition these domain-specific query-document pairs across six client nodes. Table 7 shows the resulting data distribution, with each cell representing the percentage of documents from a specific domain allocated to each client. This approach creates significant heterogeneity in the data distribution across clients, reflecting real-world federated scenarios where institutions specialize in different medical fields.

D.3 Parameter Settings

HyperParameters. This section presents a detailed overview of the hyperparameter settings used in our experiments. As shown in Table 8, the key parameters were carefully selected and tuned to ensure fair comparisons and optimal performance.

Package Parameters. Our implementation leverages the Milvus Standalone version 2.4.13 as the vector database backend with HNSW (Hierarchical Navigable Small World) as the index type and L_2 distance as the metric type. The HNSW con-

Table 8: Hyperparameter settings

Parameter	Value
General Training	
Embedding Model	intfloat/e5-base-v2(109M)
Language Model	QWen/QWen-2.5-7b(7.61B)
Communication rounds (T)	200
Local epochs (E)	3
Learning rate (η)	$1e^{-5}$
Batch size	512
Optimizer	Adam
Weight decay (λ)	0.01
InfoNCE temperature	0.05
Knowledge Distillation	
KD temperature (τ_{KD})	3.0
KD epochs	50
KD learning rate	0.1
Depth Controller	
Data complexity score weights	$w_1 = 0.8, w_2 = 1.5, w_3 = 1.2$
API weights	$w_L = 0.8, w_O = 0.2$
Window size (T_w)	5
Minimum stable rounds (τ_s)	3
Maximum low rounds (τ_l)	4
Cooling period (T_{cool})	5

figuration parameters were set to $M = 16$ and efConstruction $= 256$, balancing search accuracy with indexing efficiency. For text processing and model interactions, we utilized the `transformers` library (version 4.48.3), with `e5-base-v2`'s tokenizer for document chunking operations using a maximum length of 512 tokens. During LLM inference with `QWen2.5-7B-Instruct`, we employed a carefully tuned parameter set including `max_new_tokens=128`, `temperature=0.4`, `do_sample=True`, `top_k=50`, `top_p=0.9`, and `repetition_penalty=1.2`, with `pad_token_id` set to the tokenizer's EOS token ID. Model loading utilized `device_map="auto"` for optimal GPU allocation, float16 precision for memory efficiency, `low_cpu_mem_usage=True` to minimize RAM consumption, and `trust_remote_code=True` to properly handle model-specific optimizations.

D.4 Implementation Details

In our human evaluation process, we provided domain experts with a structured scoring rubric, as shown in Table 9, to ensure consistent and objective assessment of RAG response quality. This rubric guided experts to evaluate responses based on medical accuracy, clinical relevance, terminology precision, and overall coherence. Experts were instructed to focus particularly on whether responses maintained proper distinctions between medical terms, accurately represented clinical concepts, and provided information that would be useful in actual medical contexts.

Table 9: Human Evaluation Scoring Rubric for RAG Response Quality

Score	Description of RAG Response Quality
9–10	**Excellent:** Comprehensive, highly accurate, directly relevant response that fully addresses all query aspects. Demonstrates excellent synthesis of context information, perfectly faithful to the provided context (no hallucinations). Maintains precise distinctions between medical terms and concepts with no terminology confusion.
7–8	**Good:** Largely correct, relevant response addressing main query aspects. Mostly faithful to context with minimal unsupported claims. Medical terminology is used accurately with minimal ambiguity. Generally coherent and understandable.
5–6	**Fair:** Response attempts to answer query but has noticeable issues. May be partially correct/complete, contain some terminology imprecision, or occasional confusion between related medical concepts.
3–4	**Poor:** Mostly irrelevant response with significant factual inaccuracies or superficial query coverage. Contains notable unsupported claims, terminology errors, or conflation of distinct medical concepts.
1–2	**Very Poor:** Completely irrelevant, nonsensical, largely incorrect response with severe medical inaccuracies. Contains fundamental misunderstandings of medical concepts, dangerous terminology confusion.