Optimizing Cross-Client Domain Coverage for Federated Instruction Tuning of Large Language Models

Zezhou Wang[1], Yaxin Du[2], Xingjun Ma[3], Yugang Jiang[3], Zhuzhong Qian[1]*, Siheng Chen[2]*,
[1]Nanjing University, [2]Shanghai Jiao Tong University,
[3]Fudan University
{zzw.cs@smail, qzz@}nju.edu.cn, {dorothydu, sihengc}@sjtu.edu.cn, {xingjunma, ygj}@fudan.edu.cn

Abstract

Federated domain-specific instruction tuning (FedDIT) for large language models (LLMs) aims to enhance performance in specialized domains using distributed private and limited data, yet identifying key performance drivers and optimal augmentation strategies remains challenging. We empirically establish that cross-client domain coverage, rather than data heterogeneity, is the pivotal factor. We then introduce FedDCA, an algorithm that explicitly maximizes this coverage through diversity-oriented client center selection and retrieval-based augmentation, constructing diverse, non-redundant cross-client instruction sets. Extensive experiments across multiple domains demonstrate FedDCA's superiority over eleven baselines, achieving performance gains of up to 29.19% and domain coverage improvements of 4.82%-21.36%. FedDCA maintains its effectiveness in diverse and challenging scenarios, including data selection, held-out settings where task-specific public data is scarce and various data heterogeneity, with manageable privacy risks. This work clarifies critical FedDIT dynamics and presents FedDCA as an effective, privacy-preserving, and scalable solution for advancing domain-specific LLM tuning.

1 Introduction

Recently, federated instruction tuning (FedIT) has gained attention as a novel approach that leverages the principles of federated learning (FL) to facilitate collaborative training of large language models (LLM) in distributed environments while maintaining the confidentiality of private data (McMahan et al., 2017; Ye et al., 2024c; Zhang et al., 2023b). This methodology allows for the exchange of model parameters among distributed data holders, thereby achieving a careful balance between privacy preservation and efficient model optimization. Despite the establishment of various FedIT frameworks (Ye et al., 2024c; Kuang et al., 2023; Zhang et al., 2023b), existing literature has not adequately addressed the practical challenges that Federated Domain-specific Instruction Tuning (FedDIT) may encounter in real-world applications. For instance, FedIT generally necessitates a sufficient amount of instruction data for fine-tuning, which is often a shortage in domain-specific fine-tuning contexts (Zhang et al., 2024c).

We explore Federated Domain-specific Instruction Tuning (FedDIT), an innovative approach that harnesses federated learning (FL) to unlock the potential of Large Language Models (LLMs) in specialized domains. A significant hurdle in deploying such models arises when multiple entities hold valuable but limited and privacy-sensitive data—a common scenario in fields like medical diagnosis (Guan and Liu, 2023; Hu et al., 2023) and financial risk assessment (Abadi et al., 2024). These entities require collaborative training of domain-specific LLMs without direct data sharing (Xu et al., 2024a; Zhang et al., 2024c). Our findings underscore a key limitation: exclusive reliance on local, in-domain data, despite its quality, often results in subpar performance due to insufficient scale (see Table 1). FedDIT tackles this by integrating carefully designed instruction augmentation strategies (detailed in Section 2) that expand local datasets while upholding privacy. This augmentation is not merely a supplement but a crucial enabler for robust instruction tuning and for averting performance decline. Focusing on practical and high-quality augmentation (Zhang et al., 2024c; Toshniwal et al., 2024), we investigate a FedDIT setup employing a server-hosted public dataset (a cross-domain instruction set, detailed in Appendix B.1) and utilize various sampling strategies to realize the data augmentation (discussed in Section 3). This dataset, abstracting diverse open-source instructions, provides a versatile foundation for augmentation techniques designed to elevate model performance within spe-

*Corresponding author.

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 991–1011
November 4-9, 2025

cific target domains.

Additionally, the factors affecting FedDIT are still unclear. Compounding this uncertainty, introducing augmented instructions may further complicate results, making it difficult to ascertain effective improvement strategies. We conduct experiments to unveil a significant finding: there is no monotonic correlation between the degree of non-independent and identically distributed (non-iid) and LLM's performance in the context of FedDIT. Inspired by Explore-Instruct (Wan et al., 2023), which highlights the potential of domain coverage in domain-specific instruction tuning. Unlike previous metric that measures domain coverage using the distribution of verb-noun pairs, we define a more general, representation-based cross-client domain coverage metric and investigate its impact on FedDIT. Our results show that domain coverage significantly influences model performance within the corresponding domain.

To maximize cross-client domain coverage without compromising client data privacy, we propose a novel FedDIT algorithm, **FedDCA** (**D**omain **C**overage **A**ugmentation). The inability of direct data sharing under privacy constraints (detailed in Section 5.1) motivates our approach. Initially, each client computes a set of candidate cluster centers from its local data; these serve as privacy-preserving proxies for its semantic distribution. FedDCA then employs a server-side, domain-coverage-oriented selection algorithm (detailed in Section 5.2) to choose a strategic subset (denoted as client centers) from all uploaded candidate centers. These client centers then guide server-side dense retrieval for instruction augmentation. This selection process serves two critical purposes: 1) it provides a privacy-preserving way to capture the semantic diversity of distributed data, and 2) it enables efficient retrieval of relevant public instructions that complement the collective needs of the clients. By strategically optimizing the cross-client domain coverage, FedDCA efficiently constructs an augmented train set that enhances the model's generalization capability, leading to superior performance on domain-specific tasks.

To substantiate our claims, we conduct a rigorous empirical study. This includes foundational experiments identifying the key performance drivers in FedDIT, the development and evaluation of our proposed FedDCA algorithm across multiple specialized domains (medical, financial, and mathematical) against eleven baselines, and in-depth analyses of its practical aspects such as privacy, scalability, and robustness in challenging settings (detailed in Section 6 and Appendix B). Our key contributions are as follows:

- We empirically reveal a critical finding: in Federated Domain-specific Instruction Tuning (FedDIT), cross-client domain coverage, rather than data heterogeneity, substantially impacts LLM effectiveness. We propose a novel representation-based metric to quantify this cross-client domain coverage.
- We propose FedDCA, an algorithm that maximizes cross-client domain coverage through diversity-oriented client center selection and retrieval-based instruction augmentation.
- Extensive experiments across diverse domains demonstrate FedDCA's superior effectiveness and plug-and-play capability, outperforming baselines by at most 29.19% in model performance and improving relative domain coverage by 4.82% to 21.36%. We also show its privacy-preserving capability in Appendix B.5.
- We demonstrate FedDCA's robustness and practical applicability by evaluating its performance in data selection scenarios (achieves comparable performance using only 10% of baseline's data), held-out settings where the server lack of task-specific public data, and its consistent effectiveness across varying inter-client data heterogeneity (Section 6.2 and Appendices B and C).

2 Related Work

Federated Instruction Tuning. Instruction tuning has been widely applied across various application areas of large language models (LLM), serving as a key technique to enhance the capabilities and controllability of LLM (Zhang et al., 2023c; Wei et al., 2022). Recently, federated instruction tuning (FedIT) has emerged as an effective strategy for the distributed optimization of LLMs, leveraging federated learning (FL) protocols to improve the handling of privacy-sensitive tasks in real-world scenarios. So far, several FedIT frameworks (Ye et al., 2024c,b; Zhang et al., 2023b) have been established to evaluate the effectiveness of FedIT across multiple datasets, tasks, and FL methods. While these platforms provide a foundation for research, they have not yet introduced more complex federated algorithms and deeply investigated the

challenging problems and factors affecting FedIT, which are crucial for advancing this field.

Some progress has been made in the study of FedDIT (Zhang et al., 2024c; Wang et al., 2024; Ye et al., 2024a), such as FewFedPIT, which addresses data scarcity by locally generating data using pre-trained LLMs and is the first to explore memory extraction attacks within FedIT. Another relevant approach is FedIT-U2S (Ye et al., 2024a), which focuses on enabling FedIT when clients only possess raw, unstructured documents. Similar to our conceptualization of a server-hosted public dataset for augmentation, FedIT-U2S also leverages server-side resources (the example database and the LLM for instruction generation) to enrich client data, albeit with a focus on structuring initially unstructured data rather than augmenting existing instructions with diverse new ones.

Non-IID data distribution is a common challenge in FL (Karimireddy et al., 2020; Li et al., 2020; Sattler et al., 2019; Li et al., 2019, 2021). However, the impact of data heterogeneity on FedDIT has not been fully explored. To fully understand the impact of data heterogeneity on FedDIT and what truly matters in FedDIT, we conduct a comprehensive analysis in Section 4.

Domain Instruction Augmentation. In the real world, there is an urgent need for training LLMs with specific functionalities (e.g., reasoning capabilities) or domain-specific LLMs (e.g., code (Nijkamp et al., 2023; Luo et al., 2024), medical (Zhang et al., 2023d), financial (Yang et al., 2023b,a; Zhang et al., 2023a; Wu et al., 2023), mathematical (Yue et al., 2024; Luo et al., 2023)).

Existing works tend to use open-source domain-specific instruction tuning datasets for training. However, the target domain may not always have corresponding ready-made domain-specific instruction datasets. Even if they exist, these datasets are often limited in scale.

Augmentation strategies broadly fall into two categories: those leveraging existing/mined data and those generating new instructions. Methods based on **existing or mined data** include reusing human-curated public datasets (Wang et al., 2023; Zhang et al., 2023e), retrieving from large instruction pools (Xia et al., 2024; Jiao et al., 2023), or scaling instruction acquisition from the web (Yue et al., 2024; Zhou et al., 2024). These approaches offer access to diverse, potentially high-quality instructions with generally favorable efficiency and privacy, but depend on the availability and relevance of such external data for the target domain. Conversely, **generative methods** (Wang et al., 2022; Zhang et al., 2024c; Wang et al., 2024) create new instructions using LLMs (locally or via APIs). While potentially highly tailored, they often incur significant computational/API costs, raise privacy concerns, and face challenges in ensuring consistent instruction quality and diversity, as detailed in Appendix B.

In summary, adapting existing instruction augmentation techniques to federated learning presents significant challenges in balancing data quality, diversity, efficiency, privacy, and scalability. Current methods often involve trade-offs between costly, quality-variable generative approaches and data-dependent retrieval strategies. This necessitates robust augmentation frameworks tailored for FedDIT, motivating our work in this area.

3 Problem Formulation

Federated Domain-specific Instruction Tuning (FedDIT) is a federated learning approach designed to improve the performance of LLMs in specific domains by utilizing limited cross-client private data in combination with domain-specific instruction augmentation strategies (Zhang et al., 2024c; Xia et al., 2024; Wang et al., 2022).

For practical and scalable augmentation within FedDIT, we conceptualize the primary source of augmentation instructions as a **server-hosted, multi-domain public dataset** (detailed in Appendix B.1). This strategic choice offers several key advantages in a federated context. Firstly, it provides a unified and manageable resource pool, abstracting the complex origins of public instructions (whether curated, mined, or pre-generated). Secondly, server management of this dataset decouples data sourcing from the core federated augmentation logic (such as retrieval strategies), allowing the latter to focus on optimizing instruction selection and distribution. Finally, a centralized public dataset enables efficient server-side pre-processing and indexing, which ensures consistent data quality and supports sophisticated retrieval mechanisms beneficial to all participating clients.

Consider N distributed clients, each with local private data D_k^l of size N_k^l, and augmented data D_k^g from the server's public dataset D^p. Due to constraints like memory and computation, client c_k can accept at most N_k^p public instructions. The server

maintains D^p that spans multiple domains and is responsible for data augmentation strategies Λ and parameter aggregation. For efficiency, we adopt Low-Rank Adaption (LoRA) (Hu et al., 2022) as the fine-tuning method, tuning additional parameters $\Delta\phi$ while keeping pre-trained LLM parameters ϕ frozen.

The objective of FedDIT is to enhance the domain-specific performance of LLMs through FL without sharing private data (Ye et al., 2024c; Zhang et al., 2024c, 2023b), which is defined as:

$$\arg\min_{\Delta\phi}\{F(\phi,\Delta\phi) \triangleq \sum_{k=1}^{N} p_k F_k\left(\phi,\Delta\phi_k; D_k; \alpha_k\right)\}, \tag{1}$$

where $F_k(\cdot)$ represents the accumulated instruction fine-tuning loss of model $w_{\phi+\Delta\phi_k}$ evaluated on client c_k's augmented dataset $D_k^l \cup D_k^g$. Here, p_k denotes client k's weight based on data ratio, and $\alpha_k = \frac{N_k^p}{N_k^l+N_k^p}$ represents the proportion of public data. Equation 1 optimizes the global model by minimizing the weighted sum of empirical losses across clients' augmented instructions, thereby enhancing in-domain utility. The empirical loss for client c_k, $F_k(\phi,\Delta\phi_k;\mathcal{D})$, is computed as $\frac{1}{|\mathcal{D}|}\sum_{j=1}^{|\mathcal{D}|} l(w_{\phi+\Delta\phi_k}; x_j)$, where $x_j \in D, \forall j \in \{1,2,\ldots,|D|\}$ and $l(\cdot)$ is the instruction tuning loss function.

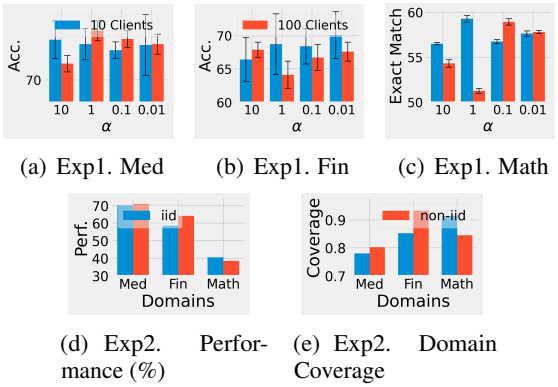
For a client c_k with private instruction set $\mathcal{D}_k$, we first encode instructions into embeddings using encoder w_{enc}. These d-dimensional vectors are then clustered into ξ groups via k-means, yielding centroid set $\mathcal{C}_k = \{\mathbf{c}_{k,1}, \mathbf{c}_{k,2}, \ldots, \mathbf{c}_{k,\xi}\}$, which is subsequently transmitted to the server.

We first introduce the **Direct Retrieval** baseline, which performs independent dense retrieval (Zhao et al., 2022) for each client without explicit consideration of cross-client domain coverage. For each centroid $\mathbf{c}_{k,j}$, the server retrieves the top-$\frac{N_k^p}{\xi}$ most similar public instructions using cosine similarity. The retrieved instructions are aggregated as client c_k's augmented data $D_k^p = \bigcup_{j=1}^{\xi} \mathcal{R}_{k,j}$. Finally, client c_k fine-tunes its model w_k on the combined dataset $D^k = D_k^l \cup D_k^g$ of private and retrieved public instructions.

4 What Truly Counts in FedDIT

4.1 Data Heterogeneity: A Minor Factor

We adopt the Dirichlet distribution (Wang et al., 2020; Yurochkin et al., 2019) to construct various heterogeneity ($\beta = [0.01, 0.1, 1, 10]$) and use k-means with $\xi = 100$ to pseudo labeling instructions.



(a) Exp1. Med (b) Exp1. Fin (c) Exp1. Math

(d) Exp2. Performance (%) (e) Exp2. Domain Coverage

Figure 1: (a)-(c) Performance (%) of different heterogeneity in each domain with 10 and 100 clients. (d)-(e) Performance and domain coverage of iid and non-iid settings on different domains. Experiments show that cross-client domain coverage significantly impacts model effectiveness, while data heterogeneity shows no monotonic correlation with performance.

We then perform instruction tuning on both 10 and 100 clients with 2 randomly selected clients participating in each round. For each domain, we only use the in-domain data and then perform FedIT. We repeat the experiments for 3 times with different random seeds (42, 43 and 44) and report the average performance and the standard deviation in each domain. As shown in Figures 1(a) to 1(c), the performance of LLM shows a non-monotonic correlation with the data heterogeneity, which indicates that the performance of LLM does not directly depend on data heterogeneity and exist other factors playing a key role.

4.2 Domain Coverage: A Key Factor

Different from Explore-Instruct, which defines domain coverage through the distribution of verb-noun pairs, we attempt to conduct more in-depth and extensive experiments to study the effect of domain coverage on FedDIT. Firstly, we define the domain coverage in the FL setting, considering the cross-client data distribution. Assume the dataset of in-domain data D^d represents the latent data distribution of this domain and the cross-client data is defined as $D^c = \cup_{k=1}^{N}\left(D_k^l \cup D_k^g\right)$.

Inspired by the submodular function (Krause and Golovin, 2014), we first define a general coverage metric. For any two sets of item embeddings, S_1 (the reference set) and S_2 (the covering set), their

coverage, denoted $d(S_1, S_2)$, is given by:

$$d(S_1, S_2) = \frac{1}{|S_1|} \sum_{s_1 \in S_1} \max_{s_2 \in S_2} \text{sim}(s_1, s_2), \quad (2)$$

where $\text{sim}(\cdot, \cdot)$ is the cosine similarity between embeddings. Then the domain coverage of D^c respect to D^d is $d(D^d, D^c \cap D^d)$. The use of $D^c \cap D^d$ ensures that we only consider the in-domain portion of the aggregated client data for this particular coverage calculation, preventing misleading scores from out-of-domain client data, consistent with our original formulation.

To better align with our setup, we explore instruction augmentation based on both iid and non-iid cross-client local data distribution and then perform dense retrieval (Zhao et al., 2022) for FedDIT (the Direct Retrieval method). We set the number of clients to 10, while each client has 100 local instructions and obtains 1000 augmented public instructions from the server. For the iid setting, we randomly sample 1000 from the public dataset and divide them into 10 shards as each client's local data. For the non-iid distribution, we perform k-means clustering with $\xi = 100$. Each client randomly samples 100 instructions from randomly selected distinct clusters and then performs retrieval for both settings.

Figures 1(d) and 1(e) presents the performance and the according domain coverage of FedDIT in different domains with iid and non-iid settings. We can observe that both iid and non-iid settings outperform in some domains, but both collectively indicate that higher domain coverage correlates with better performance.

5 Method

Based on the above empirical observations, we propose FedDCA, which enhances domain coverage to obtain a LLM that performs well on domain-specific tasks (shown in Figure 2). We formulate the cross-client domain coverage optimization problem and then introduce the FedDCA algorithm.

5.1 Optimization Problem

As domain coverage directly affects the in-domain performance of the LLM, FedDCA aims to maximize the domain coverage of the cross-client augmented data D^c with respect to the in-domain data distribution D^d. However, privacy constraints prevent clients from sharing local data, meaning the server lacks direct knowledge of clients' domain information. Consequently, directly identifying a cross-client dataset that maximizes domain coverage is infeasible. Instead, we leverage client-provided cluster centers (ξ clusters per client, obtained by k-means algorithm (Wu, 2012)) as privacy-preserving proxies for local data. These centers, typically vector averages, effectively represent local distributions with manageable privacy risks (elaborated in Appendix B.5). The core challenge thus shifts to selecting an optimal set of these client centers, $\mathcal{P}$ (one per client), such that retrieving public instructions based on $\mathcal{P}$ enhances cross-client domain coverage. This formulation provides a tractable approximation to the original optimization problem under inherent privacy constraints.

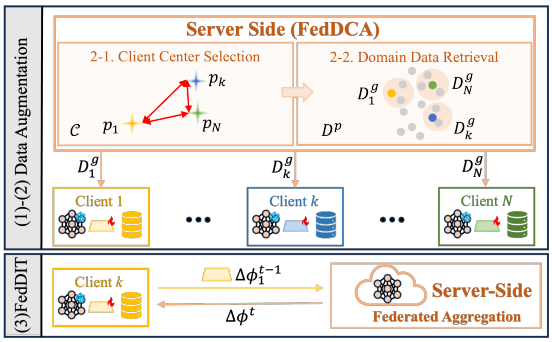


Figure 2: Overview, a three-stage process. **1) Client-side**: Clients c_k perform local instructions clustering and send cluster centers $\mathcal{C}_k$ to the server. **2) Server-side Augmentation:** The server executes diversity-oriented client center selection to maximize domain coverage (obtaining $\mathcal{P}$), retrieves augmented data D_k^g based on these centers, and distributes it to clients. **3) Collaborative Fine-tuning:** Clients collaboratively fine-tune the LLM, exchanging LoRA parameters $\Delta\phi$ with the server.

Additionally, in the FL setting, communication cost is always a critical factor. Thus, we formulate the optimization problem as follows:

$$\arg\min_{\mathcal{P}} \left\{ \eta(\mathcal{P}) - d(D^d, \mathcal{P}) \right\}, \quad (3)$$

where the first term $\eta(\mathcal{P}) = \sum_{i=1}^{N} |\mathcal{P}_i|$ represents the communication overhead of transmitting client centers. And the second term is the domain coverage of the selected client center set $\mathcal{P}$, defined in Eq.2.

For simplicity, we let ξ be the constant N (further discussed in Section 6.3). To better fit the FL environment and enhance the computational efficiency, we propose FedDCA as follows.

5.2 FedDCA

Optimizing Eq. 3 to find the ideal client center set $\mathcal{P}$ presents significant practical hurdles. First, an exhaustive search over all $C_{\xi N}^{N}$ candidate center

combinations is computationally infeasible due to its exponential complexity. Second, directly leveraging a server-hosted public dataset D^p to guide client center selection for optimal domain coverage is often impractical. This stems from several factors: 1) privacy constraints limit the server's knowledge of individual client data distributions and their specific domain focuses; 2) the inherent ambiguity and hierarchical nature of 'domains' (e.g., 'medical' versus specific medical sub-tasks) complicate precise server-side domain coverage calculation, which is already costly over a potentially massive D^p; 3) the assumption of a relevant D^p being available on the server may not hold, particularly if clients utilize external retrieval or local *self-instruct* for augmentation.

These challenges underscore the need for a more pragmatic selection strategy. Therefore, we propose FedDCA. As detailed in Alg. 1, FedDCA seeks a sub-optimal solution in polynomial time (discussed in Appendix A) and operates on the server through two main stages: 1) a coverage-oriented client center selection mechanism, and 2) client-center-based dense retrieval for data augmentation.

Coverage-oriented Client Center Selection. To address these challenges, we propose a novel client center selection strategy that, crucially, *does not rely on a server-side public dataset* D^p. Let $\mathcal{C}_{\text{all}} = \bigcup_{i=1}^{N} \mathcal{C}_i$ be the set of all candidate cluster centers uploaded by the N clients (where each client c_i provides ξ centers in $\mathcal{C}_i$). Our approach leverages $\mathcal{C}_{\text{all}}$ in two ways: 1) We use $\mathcal{C}_{\text{all}}$ as a proxy for the target domain distribution. Domain coverage is then approximated by evaluating how well a selected subset of centers covers $\mathcal{C}_{\text{all}}$ itself, making the computation tractable and independent of D^p. 2) Instead of an exhaustive search, we employ an iterative, coordinate ascent-style algorithm (detailed in Alg. 1) to select a final set $\mathcal{P}$ of N client centers. This efficiently yields a high-quality, albeit potentially suboptimal $\mathcal{P}$. Based on the $\mathcal{C}_{\text{all}}$, the domain coverage could be calculated as $d(\mathcal{C}_{\text{all}}, \mathcal{P})$.

Specifically, each client c_i locally runs k-means to produce ξ candidate centers $\mathcal{C}_i$, then uploads these centers to the server. The server initializes a selection $\mathcal{P}$ by picking one cluster center from each client (thus $|\mathcal{P}| = N$) and computes the overall coverage solely based on these uploaded centers. In each iteration, FedDCA performs the following update (as shown in Alg. 1): it removes the currently chosen center from $\mathcal{P}$, tries replacing it with other candidate cluster centers in $\mathcal{C}_{\text{all}}$, and keeps the replacement if it improves the cross-client domain coverage; this process is repeated until no further improvement is found.

Algorithm 1 FedDCA: Client Center Selection

Initialize:
  For each client c_i, pick any initial center $p_i \in \mathcal{C}_i$.
  Let $\mathcal{P} = \{p_1, \ldots, p_N\}$ and compute its coverage $C(\mathcal{P})$.
  $\Omega^* \leftarrow C(\mathcal{P})$; $i \leftarrow 0$
while True **do**
  $\mathcal{P}_{-i} \leftarrow \mathcal{P} \setminus \{p_i\}$ ▷ Remove p_i from $\mathcal{P}$
  $\nu^* \leftarrow p_i$; $\Omega_{\text{old}} \leftarrow \Omega^*$
  for $q \in (\mathcal{C}_{\text{all}} - \mathcal{P}_{-i})$ **do**
    $\mathrm{C} = d(\mathcal{C}_{\text{all}}, \mathcal{P}_{-i} \cup \{q\})$ ▷ Defined in Eq. 2
    if $\mathrm{C} > \Omega^*$ **then**
      $\Omega^* \leftarrow \mathrm{C}$; $\nu^* \leftarrow q$
    end if
  end for
  if $\Omega^* = \Omega_{\text{old}}$ **then**
    break
  end if
  $p_i \leftarrow \nu^*$; $\mathcal{P} \leftarrow \mathcal{P}_{-i} \cup \{p_i\}$; $C(\mathcal{P}) \leftarrow \Omega^*$; $i \leftarrow (i+1) \bmod N$
end while
Output: The selected client center set $\mathcal{P}$.

This D^p-agnostic selection core is computationally tractable and versatile, effective for both data augmentation and targeted data selection through the selected client centers $\mathcal{P}$.

Domain Data Retrieval. For each client center p_k, the server performs dense retrieval (Zhao et al., 2022) on public dataset D^p to get the top-N_k^p similar public instructions, then sends retrieved public datasets $\{D_1^g, \ldots, D_N^g\}$ to each clients.

Specifically, to avoid the overlap between public data and local private data, we set a threshold α to filter the public instructions that have a similarity score larger than α with the client center.

In summary, through the coverage-oriented client center selection and domain data retrieval, FedDCA obtains a better cross-client domain coverage of the augmented data D_k^g, which leads to a better model performance on the target domain. For the **Discussions** part please refer to Appendix A.

6 Experiments

To demonstrate the effectiveness of FedDCA, we conduct extensive experiments across various domains and with several baselines. For additional results and analysis, please refer to Appendix B.

6.1 Experimental Setup

Dataset and Evaluation Metrics. To evaluate the performance of FedDCA, we conduct experi-

Method	MMLU-Med	FPB	FiQA	TFNS	GSM8K
Zero-shot	70.60/-	55.94/-	18.54/-	59.21/-	23.27/-
FedAvg (McMahan et al., 2017)	68.40/0.6990	58.25/0.8529	14.18/0.8529	66.62/0.8529	47.46/0.7871
FedProx (Li et al., 2020)	69.10/-	56.51/-	14.90/-	66.45/-	47.15/-
SCAFFOLD (Karimireddy et al., 2020)	70.20/-	62.71/-	15.27/-	66.49/-	49.27/-
FedAvgM (Hsu et al., 2019)	64.70/-	68.14/-	29.27/-	70.32/-	46.85/-
Random Sampling	71.30/0.7940	64.19/0.9196	13.09/0.9196	65.53/0.9196	47.38/0.8651
Direct Retrieval	72.20/0.8830	66.31/0.9293	19.11/0.9293	67.62/0.9293	50.87/0.8967
LESS (Xia et al., 2024)	71.00/0.7737	60.56/0.8917	16.00/0.8917	61.14/0.8917	43.13/0.8352
FewFedPIT (Zhang et al., 2024c)	68.50/0.8250	56.30/0.8780	14.20/0.8780	59.10/0.8780	42.30/0.8380
Self-Instruct (Wang et al., 2022)	71.90/0.8586	59.73/0.9015	20.67/0.9015	66.54/0.9015	50.79/0.8811
KnowledgeSG (Wang et al., 2024)	73.13/0.9171	66.00/0.9490	32.72/0.9490	71.10/0.9490	51.20/0.9088
FedDCA+FedAvg	**74.50**/0.9348	67.24/0.9815	35.27/0.9815	73.32/0.9815	**52.46**/0.9320
FedDCA+FedProx	72.40/-	**72.93**/-	**38.18**/-	**77.55**/-	51.25/-
FedDCA+SCAFFOLD	73.20/-	72.68/-	33.09/-	75.50/-	50.26/-
FedDCA+FedAvgM	68.90/-	71.45/-	31.45/-	72.52/-	49.76/-

Table 1: Performance (%) and domain coverage of FedDCA and other eleven baselines. Due to limited local data or inappropriate data augmentation strategy, occurs performance degradation in the Unaugmented and Random Sampling settings (underlined values). Different FL strategy does not affect domain coverage. We can see that FedDCA outperforms all other baselines.

ments utilizing a server-hosted public dataset consisting of multiple domains' data (detailed in Appendix B.1): Alpaca, MedAlpaca (Zhang et al., 2023d), FinGPT (Yang et al., 2023a), and Math-Instruct (Toshniwal et al., 2024). Performance is evaluated on MMLU-Med (medical); FPB, FiQA, and TFNS (financial); and GSM8K (math). The former two domains' metric is Acc. and math's is Exact Match. We set the number of clients to 10 by default, while each client has 100 local instructions and obtains 1k augmented public instructions by default from the server. Specifically, following the construction of the client's local data heterogeneity in Section 4, we set the β to 0.1 by default. We also exhibits FedDCA's strong robustness to different levels of data heterogeneity in Appendix C.

Baselines. We compare FedDCA against a range of baseline methods, categorized into unaugmented and augmented approaches (further details are available in Appendix B). **Unaugmented Methods.** These include **Zero-shot inference**, where the LLM directly predicts without any fine-tuning, and **`FedIT`**. `FedIT` represents the application of standard federated learning to clients' local data, for which we employ four widely recognized FL algorithms: FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020), SCAFFOLD (Karimireddy et al., 2020), and FedAvgM (Hsu et al., 2019). **Augmented Methods.** This category encompasses several strategies. **Random Sampling** serves as a simple baseline, randomly selecting N_k^p public data instances for each client. **Direct Retrieval**, a precursor to our method, performs dense retrieval from public data based on each client's local instructions independently, without optimizing for cross-client domain coverage. **LESS** (Xia et al., 2024) utilizes gradients from a warmed-up LLM on both training and a required validation set for its similarity-based retrieval. For generation-based approaches, **Self-Instruct** (Wang et al., 2022) generates new instructions by prompting a powerful external LLM (e.g., GPT-3.5). **FewFedPIT** (Zhang et al., 2024c) is a federated few-shot method where the global model is used for local in-context learning to synthesize data, aiming to improve both performance and privacy. Finally, **KnowledgeSG** (Wang et al., 2024) employs a server-hosted expert model to generate and refine synthetic data, which clients then use to enhance their local models.

6.2 Main Results

Performance & Domain Coverage. Table 1 showcases the performance and corresponding domain coverage of FedDCA against various baselines across three distinct domains. FedDCA consistently outperforms all eleven baselines, demonstrating substantial performance improvements of up to 29.19%. This superior performance is strongly correlated with its ability to achieve the highest domain coverage, surpassing other methods by an average of 4.82% to 21.36%.

While unaugmented methods naturally struggle due to limited local data and thus restricted domain coverage, many existing augmentation strategies also exhibit significant drawbacks. 1) Generation-based methods present a mixed picture: Knowl-

+ FedDCA	MMLU-Med	FPB	GSM8K
Random Sampling	71.80 (**+0.50**)	63.92 (-0.27)	47.82 (**+0.44**)
Direct Retrieval	72.10 (-0.10)	67.78 (**+1.47**)	52.53 (**+1.66**)
LESS	70.80 (-0.20)	61.03 (**+0.47**)	43.92 (**+0.79**)
Self-Instruct	70.70 (-1.20)	60.21 (**+0.48**)	50.25 (-0.54)

Table 2: Plug-and-Play. Performance (%) of FedDCA with different data augmentation methods. FedDCA matches baseline's performance by just 10% of its data through selection.

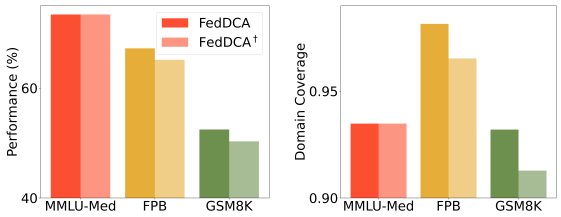
edgeSG, leveraging a powerful server-hosted expert model, achieves strong results while FewFedPIT underperform even unaugmented baselines, potentially due to inconsistent synthetic data quality and diversity. Moreover, these approaches often entail significant resource demands: KnowledgeSG and FewFedPIT involve high computational costs (server-side and client-side), while Self-Instruct incurs monetary expenses for API calls with the privacy leakage. The potential lack of quality and diversity in synthetic instructions further constrains their effectiveness. 2) Random Sampling and Direct Retrieval, while less costly, lack a strategic mechanism to ensure comprehensive cross-client domain coverage. 3) LESS, while performs domain specific retrieval, requiring access to a validation set, and additional computation (warmup training).

In conclusion, FedDCA's strategic data augmentation substantially enhances domain coverage and model performance, efficiently circumventing the computational burden and inherent quality/redundancy challenges of data generation.

Plug-and-Play. We demonstrate FedDCA's plug-and-play capability by combining it with other augmentation methods. After sampling 1k instructions per client through different baselines, FedDCA performs retrieval by the selected client center (200 per client). As shown in Table 2, despite using only 20% of the sampled data, FedDCA achieves comparable performance with baselines. This highlights FedDCA's efficient data selection capability based on the cross-client domain coverage, which is further discussed in Appendix B.7.

6.3 Ablation Study

Impact of Different Cluster Number. The hyperparameter ξ is the number of clusters in the k-means algorithm. The experiment is conducted on $\xi = [N, 2N, 4N, 8N]$, where N is the number of clients. We report the domain coverage of the augmented dataset via FedDCA with different ξ on the three domains in Table 3. Results show that there is no best ξ for all domains and $\xi = N$ is usually an acceptable choice.



(a) Performance (b) Domain Coverage

Figure 3: Impact of similarity threshold α on FedDCA's (a) performance (%) and (b) domain coverage. Results highlight the role of the similarity threshold in enhancing diversity. Lighter bars indicate FedDCA†.

Domain	$\xi = 10$	$\xi = 20$	$\xi = 40$	$\xi = 80$
Med.	0.9348	0.9478	0.9466	0.9618
Fin.	0.9815	0.9814	0.9819	0.9813
Math.	0.9320	0.9348	0.9344	0.9337

Table 3: Domain Coverage for Different ξ Values. $\xi = N$ is usually an acceptable choice.

Similarity Threshold. Figure 3 presents an ablation study comparing FedDCA (w similarity threshold α) against FedDCA† (w/o α). FedDCA achieves superior performance and domain coverage, highlighting the crucial role of the similarity threshold in effectively amplifying the diversity of augmented instructions and subsequently boosting model efficacy. Note that the medical domain remains unchanged, as the similarity of all pairwise embeddings is below 0.7 (the default value for the similarity threshold α).

We also investigate the impact of different retrieval amounts on FedDCA's performance, detailed in Appendix B.4.

7 Conclusion

This work advances our understanding of Federated Domain-specific Instruction Tuning (FedDIT), establishing cross-client domain coverage as paramount over data heterogeneity for high performance. FedDCA provides a practical pathway to optimize this coverage, demonstrating significant gains through strategic data augmentation. FedDCA's plug-and-play capability and validated robustness in challenging scenarios (e.g., scarcity of task-specific public data, large-scale deployments) further positions it as a reliable building block for real-world, privacy-preserving specialized LLM applications. These findings offer both an applicable solution and crucial guidance for future federated system design where domain coverage is the key.

Limitations

While FedDCA provides a robust and scalable framework for federated domain-specific instruction tuning, several avenues remain for further exploration. Although we have empirically demonstrated FedDCA's resilience to the similar cross-client data distributions (see Appendix C), its performance may still be influenced by the quality and diversity of both the server-hosted public data and the client-provided candidate centers. Future work could investigate the adaptive client-side clustering to further enhance coverage and robustness. Additionally, automating hyperparameter selection (e.g., the number of clusters per client) and extending FedDCA to handle evolving data distributions or continual learning scenarios are promising directions to further improve its practicality and generalization in real-world deployments.

Acknowledgments

This work was supported in part by the Nanjing University-ChinaMobile Communications Group Co., Ltd. Joint Institute and Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

A. Abadi, Bradley Doyle, Francesco Gini, Kieron Guinamard, S. K. Murakonda, Jack Liddell, Paul Mellor, S. Murdoch, Mohammad Naseri, Hector Page, George Theodorakopoulos, and Suzanne Weller. 2024. Starlit: Privacy-preserving federated learning to enhance financial fraud detection. *IACR Cryptol. ePrint Arch.*, 2024:90.

Abbas Acar, Hidayet Aksu, A. Selcuk Uluagac, and Mauro Conti. 2018. A Survey on Homomorphic Encryption Schemes: Theory and Implementation. *ACM Comput. Surv.*, 51(4):79:1–79:35.

Dor Bernsohn, Gil Semo, Yaron Vazana, Gila Hayat, Ben Hagag, Joel Niklaus, Rohit Saha, and Kyryl Truskovskyi. 2024. Legallens: Leveraging llms for legal violation identification in unstructured text. pages 2129–2145.

Fabian Biester, Mohamed Abdelaal, and Daniel Del Gaudio. 2024. Llmclean: Context-aware tabular data cleaning via llm-generated ofds. *ArXiv*, abs/2404.18681.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, and 1 others. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650.

Amit Deshpande and Rameshwar Pratap. 2023. Improved outlier robust seeding for k-means. *ArXiv*, abs/2309.02710.

Hao Guan and Mingxia Liu. 2023. Federated learning for medical image analysis: A survey. *Pattern recognition*, 151.

Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. 2019. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, and 1 others. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Zhaoyu Hu, Leyin Li, An Sui, Guoqing Wu, Yuanyuan Wang, Zhifeng Shi, Jinhua Yu, Liang Chen, Guiguan Yang, and Yuhao Sun. 2023. Ocif: automatically learning the optimized clinical information fusion method for computer-aided diagnosis tasks. *International Journal of Computer Assisted Radiology and Surgery*, 18:2273–2286.

Wenxiang Jiao, Jen-tse Huang, Wenxuan Wang, Zhiwei He, Tian Liang, and 1 others. 2023. ParroT: Translating during chat using large language models tuned with human translation and feedback. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15009–15020, Singapore. Association for Computational Linguistics.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. 2020. SCAFFOLD: Stochastic controlled averaging for federated learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5132–5143. PMLR.

Andreas Krause and Daniel Golovin. 2014. Submodular function maximization. *Tractability*, 3(71-104):3.

Weirui Kuang, Bingchen Qian, Zitao Li, Daoyuan Chen, Dawei Gao, and 1 others. 2023. Federatedscope-llm: A comprehensive package for fine-tuning large language models in federated learning. *arXiv preprint arXiv:2309.00363*.

Qinbin Li, Bingsheng He, and Dawn Song. 2021. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10713–10722.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated Optimization in Heterogeneous Networks. *Proceedings of Machine Learning and Systems*, 2(arXiv:1812.06127):429–450.

Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2019. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*.

Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. 2024. Wizardcoder: Empowering code large language models with evol-instruct. In *The Twelfth International Conference on Learning Representations*.

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.

Erik Nijkamp, Hiroaki Hayashi, Caiming Xiong, Silvio Savarese, and Yingbo Zhou. 2023. Codegen2: Lessons for training llms on programming and natural languages. *arXiv preprint arXiv:2305.02309*.

Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. 2019. Robust and communication-efficient federated learning from non-iid data. *IEEE transactions on neural networks and learning systems*, 31(9):3400–3413.

Shubham Toshniwal, Ivan Moshkov, Sean Narenthiran, Daria Gitman, Fei Jia, and Igor Gitman. 2024. OpenMathInstruct-1: A 1.8 Million Math Instruction Tuning Dataset. *Preprint*, arXiv:2402.10176.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605.

Fanqi Wan, Xinting Huang, Tao Yang, Xiaojun Quan, Wei Bi, and Shuming Shi. 2023. Explore-Instruct: Enhancing Domain-Specific Instruction Coverage through Active Exploration. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9435–9454, Singapore. Association for Computational Linguistics.

Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. 2020. Federated learning with matched averaging. In *International Conference on Learning Representations*.

Wenhao Wang, Xiaoyu Liang, Rui Ye, Jingyi Chai, Siheng Chen, and Yanfeng Wang. 2024. Knowledgesg: Privacy-preserving synthetic text generation with knowledge distillation from server. pages 7677–7695.

Xiao Wang, Weikang Zhou, Can Zu, Han Xia, Tianze Chen, Yuansen Zhang, Rui Zheng, Junjie Ye, Qi Zhang, Tao Gui, and 1 others. 2023. Instructuie: Multi-task instruction tuning for unified information extraction. *arXiv preprint arXiv:2304.08085*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Junjie Wu. 2012. *Advances in K-means clustering: a data mining thinking*. Springer Science & Business Media.

Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, and 1 others. 2023. Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2303.17564*.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. LESS: Selecting influential data for targeted instruction tuning. In *International Conference on Machine Learning (ICML)*.

Binqian Xu, Xiangbo Shu, Haiyang Mei, Zechen Bai, Basura Fernando, Mike Zheng Shou, and Jinhui Tang. 2024a. Dofit: Domain-aware federated instruction tuning with alleviated catastrophic forgetting.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. 2024b. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing. *ArXiv*, abs/2406.08464.

Hongyang Yang, Xiao-Yang Liu, and Christina Dan Wang. 2023a. Fingpt: Open-source financial large language models. *FinLLM Symposium at IJCAI 2023*.

Yi Yang, Yixuan Tang, and Kar Yan Tam. 2023b. Investlm: A large language model for investment using financial domain instruction tuning. *arXiv preprint arXiv:2309.13064*.

Rui Ye, Rui Ge, Yuchi Fengting, Jingyi Chai, Yanfeng Wang, and Siheng Chen. 2024a. Leveraging unstructured text data for federated instruction tuning of large language models. *ArXiv*, abs/2409.07136.

Rui Ye, Rui Ge, Xinyu Zhu, Jingyi Chai, Yaxin Du, Yang Liu, Yanfeng Wang, and Siheng Chen. 2024b. Fedllm-bench: Realistic benchmarks for federated learning of large language models. *arXiv preprint arXiv:2406.04845*.

Rui Ye, Wenhao Wang, Jingyi Chai, Dihan Li, Zexi Li, Yinda Xu, Yaxin Du, Yanfeng Wang, and Siheng Chen. 2024c. Openfedllm: Training large language models on decentralized private data via federated learning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 6137–6147, New York, NY, USA. Association for Computing Machinery.

Xiang Yue, Tuney Zheng, Ge Zhang, and Wenhu Chen. 2024. Mammoth2: Scaling instructions from the web. *arXiv preprint arXiv:2405.03548*.

Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. 2019. Bayesian nonparametric federated learning of neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7252–7261. PMLR.

Boyu Zhang, Hongyang Yang, and Xiao-Yang Liu. 2023a. Instruct-fingpt: Financial sentiment analysis by instruction tuning of general-purpose large language models. *FinLLM Symposium at IJCAI 2023*.

Jianyi Zhang, Saeed Vahidian, Martin Kuo, Chunyuan Li, Ruiyi Zhang, Guoyin Wang, and Yiran Chen. 2023b. Towards building the federated gpt: Federated instruction tuning. *arXiv preprint arXiv:2305.05644*.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, and 1 others. 2023c. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*.

Shuo Zhang, Zezhou Huang, and Eugene Wu. 2024a. Data cleaning using large language models. *ArXiv*, abs/2410.15547.

Xinlu Zhang, Chenxin Tian, Xianjun Yang, Lichang Chen, Zekun Li, and Linda Ruth Petzold. 2023d. Alpacare: Instruction-tuned large language models for medical application. *arXiv preprint arXiv:2310.14558*.

Yiming Zhang, Nicholas Carlini, and Daphne Ippolito. 2024b. Effective prompt extraction from language models. In *First Conference on Language Modeling*.

Yue Zhang, Leyang Cui, Deng Cai, Xinting Huang, Tao Fang, and Wei Bi. 2023e. Multi-task instruction tuning of llama for specific scenarios: A preliminary study on writing assistance. *arXiv preprint arXiv:2305.13225*.

Zhuo Zhang, Jingyuan Zhang, Jintao Huang, Lizhen Qu, Hongzhi Zhang, Qifan Wang, Xun Zhou, and Zenglin Xu. 2024c. FewFedPIT: Towards Privacy-preserving and Few-shot Federated Instruction Tuning. *arXiv preprint arXiv:2403.06131*.

Wayne Xin Zhao, Jing Liu, Ruiyang Ren, and Ji rong Wen. 2022. Dense text retrieval based on pretrained language models: A survey. *ACM Transactions on Information Systems*, 42:1 – 60.

Kun Zhou, Beichen Zhang, Jiapeng Wang, Zhipeng Chen, Wayne Xin Zhao, Jing Sha, Zhichao Sheng, Shijin Wang, and Ji-Rong Wen. 2024. Jiuzhang3.0: Efficiently improving mathematical reasoning by training small data synthesis models. *arXiv preprint arXiv:2405.14365*.

A Methodology

A.1 Discussions

Computation. The client center selection process, as detailed in Alg. 1, operates on $\mathcal{C}_{\text{all}}$, the set of all $N \cdot \xi$ candidate centers uploaded by N clients (each providing ξ centers). Since FedDCA is an iterative optimization algorithm, let T be the number of passes until convergence. In each pass, the algorithm tries to swap the currently selected center p_i with one of the other candidate centers from $\mathcal{C}_{\text{all}}$ ($N\xi - N$ choices). Each potential swap requires re-evaluating the domain coverage C, which is $N\xi(N + N\log N)$. Therefore, the total complexity for one pass is approximately $O(N\xi \cdot (N\xi(N + N\log N))) = O(N^3\xi^2(1 + \log N))$. If the algorithm converges in T passes, the total complexity for client center selection is $O(T \cdot N^3\xi^2(1 + \log N))$. Overall, this complexity is polynomial in the number of clients N ($\xi = N$ by default), rendering the selection process tractable for typical federated learning scenarios where N and ξ are manageable.

To empirically validate the convergence speed (T) of our client center selection algorithm, we conducted the client center selection for each domain under the default setup for 100 times. The results show that FedDCA converges remarkably quickly, with an average of 1.1N passes across all experiments. This rapid convergence indicates that the algorithm typically requires only one full iteration through all clients, with occasional minor adjustments in a second round of iteration, making it highly efficient in practice.

Communication. The communication overhead of FedDCA is mainly incurred in two phases: (1) In the domain instruction augmentation phase, each client uploads ξ cluster centers (typically low-dimensional vectors, e.g., 1024-d) to the server, and the server returns N_k^p retrieved public instructions to each client. (2) During model parameter synchronization, FedDCA follows standard federated learning (FedIT) procedures, requiring only the exchange of LoRA parameters $\Delta\phi_k$, which significantly reduces communication cost.

Privacy. Comparing FedDCA with other FedIT methods (Zhang et al., 2024c; Ye et al., 2024c), the difference lies in the client center selection stage. In this stage, the client only uploads the cluster center to the server, which is the average of embeddings to its cluster. In addition, the potential privacy leakage can be further avoided through homomorphic encryption (Acar et al., 2018), which allows the server to directly compute on ciphertext for matrix multiplication for dense retrieval.

Robustness to Outliers. The k-means clustering, a component of our client-side processing, can be sensitive to outliers in the local data. If a client's local dataset contains significant outliers, these could potentially skew the resulting cluster centers, and subsequently affect the server-side selection and retrieval process. To mitigate this, two main approaches could be considered. **Firstly**, more robust clustering algorithms from existing literature could be adopted by clients. For instance, Deshpande and Pratap, 2023 enhances k-means++'s robustness by capping the selection probability for new centers, based on the overall clustering cost and an allowed number of outliers. This prevents distant, potentially outlier, points from dominating the initialization process and improves clustering in noisy data. **Secondly**, clients could perform preliminary data filtering or cleaning steps before initiating the FedDCA process. This might involve simple heuristic-based filtering (e.g., keyword-based removal of irrelevant instructions) or leveraging lightweight LLM-based tagging to identify and exclude potential outliers (Bernsohn et al., 2024; Biester et al., 2024; Zhang et al., 2024a). A comprehensive investigation into the impact of outliers and the optimal strategies for enhancing FedDCA's robustness in such scenarios is a promising direction for future work.

A.2 Theoretical Foundation of FedDCA

To establish the theoretical soundness of FedDCA's greedy approach, we demonstrate that our optimization objective is equivalent to the classic submodular facility location problem. This equivalence provides crucial theoretical guarantees: it ensures that our computationally efficient greedy algorithm achieves at least a $1 - 1/e$ approximation ratio relative to the optimal solution for selecting the client centers $\mathcal{P}$, while remaining tractable for practical federated deployments.

The Optimization Problem in FedDCA. The overall optimization problem in our paper is formulated as Eq. 3. In our specific problem setting, the number of cluster centers (ξ) per client is considered fixed. Therefore, minimizing the objective function is equivalent to maximizing the domain coverage term $d(D^d, \mathcal{P})$ through the selected client centers $\mathcal{P}$.

The Submodular Facility Location Problem. In its abstract form, we are given: a set of "locations" or "customers" $\mathcal{U}$ that need to be serviced; a set of potential "facilities" $\mathcal{F}$ that can be opened; a benefit function $w(i, j)$ that quantifies the value of servicing customer $j \in \mathcal{U}$ with facility $i \in \mathcal{F}$. The objective is to select a subset of facilities $\mathcal{S} \subseteq \mathcal{F}$ of a given size k to maximize the total benefit provided to all customers:

$$g(\mathcal{S}) = \sum_{j \in \mathcal{U}} \max_{i \in \mathcal{S}} w(i, j) \tag{4}$$

Formal Equivalence. The connection between our domain coverage problem and the facility location problem becomes clear when we map the components: **1) Customers ($\mathcal{U}$)** $\rightarrow$ The set of in-domain data, D^d; **2) Facilities ($\mathcal{F}$)** $\rightarrow$ The set of all candidate centers, $\mathcal{C}_{\text{all}}$; **3) Benefit ($w(i, j)$)** $\rightarrow$ The cosine similarity between two centers, $\text{sim}(p, c)$.

Candidate centers are obtained by clustering each client's local data, whereas D^d is the union of all public in-domain data points. This "non-diagonal" mapping still matches the standard facility-location structure because every data point is evaluated against the chosen center set, regardless of origin.

With this mapping, our domain coverage objective $d(D^d, \mathcal{P})$ has the form of the classic facility location function $g(\mathcal{P})$ normalized by the size of the in-domain data, $|D^d|$. That is, $d(D^d, \mathcal{P}) = \frac{1}{|D^d|} g(\mathcal{P})$.

Monotonicity and Submodularity. The facility location function $g(\mathcal{P})$ is known to be both monotone and submodular. Our objective $d(D^d, \mathcal{P})$ inherits these properties.

Monotonicity: This property is guaranteed because our benefit function, $\text{sim}(p, c)$, is the cosine similarity from a sentence encoder, which is non-negative[1] (we have plotted histograms of the similarity scores for each domain, which fall within

[1] https://huggingface.co/BAAI/bge-large-en-v1.5

Dataset	Size	Metric
Alpaca	52,002	-
MedAlpaca	33,955	-
MMLU-Med†	1,089	Acc.
FinGPT	76,772	-
FPB†	152	Acc.
FiQA†	35	Acc.
TFNS†	299	Acc.
MathInstruct	224,567	-
GSM8K†	1,319	Exact Match

Table 4: Dataset information of each domain. Public dataset is composed of these five train sets (Alpaca, MedAlpaca, FinGPT, MathInstruct, and GSM8K). Test sets are marked with † and used for evaluation.

the range about [0.2, 0.85]). When adding a new center to $\mathcal{P}$, the maximum similarity for any data point c can only increase or stay the same. If a negative value ever arises, we can rectify the score via applying an affine shift, which preserves the order of similarities and therefore the monotonicity. Consequently, the total sum of maximum similarities, $g(\mathcal{P})$, cannot decrease.

Submodularity (Diminishing Returns): To formally show this, let's define the marginal gain for a single data point $c \in D^d$ when adding a new center x to a set of centers S as:

$$\Delta_c(S, x) = \max\big(0, \mathrm{sim}(x, c) - \max_{p \in S} \mathrm{sim}(p, c)\big) \tag{5}$$

This formula calculates the new coverage a center x provides to a point c, beyond what is already provided by the existing set S. Now, consider two sets of centers A and B such that $A \subseteq B$. For any data point c, the maximum similarity from set B must be at least as large as from set A, so:

$$\max_{p \in A} \mathrm{sim}(p, c) \leq \max_{p \in B} \mathrm{sim}(p, c) \tag{6}$$

Consequently, the marginal gain from adding x must be smaller for the larger set B. That is, for any c:

$$\Delta_c(A, x) \geq \Delta_c(B, x) \tag{7}$$

The total marginal gain of adding center x to a set S is:

$$g(S \cup \{x\}) - g(S) = \sum_{c \in D^d} \Delta_c(S, x) \tag{8}$$

Since the inequality holds for each individual term in the sum, it must hold for the sum itself. This demonstrates the diminishing returns property:

$$g(A \cup \{x\}) - g(A) \geq g(B \cup \{x\}) - g(B) \text{ for } A \subseteq B \tag{9}$$

Overall, our problem of maximizing the domain coverage $d(D^d, \mathcal{P})$ is equivalent to solving the facility location problem of maximizing $g(\mathcal{P})$, ensuring theoretical guarantees for FedDCA.

In addition, Appendix B.9 empirically validates these theoretical guarantees, demonstrating that FedDCA achieves 86.30%-99.85% of the optimal coverage—significantly exceeding the theoretical lower bound of $1 - 1/e \approx 63\%$.

B Experiments

B.1 Train and Test Dataset Information

We evaluate our method on multiple domains including medical (MedAlpaca, MMLU-Med), financial (FinGPT, FPB, FiQA, TFNS), and mathematical reasoning (MathInstruct, GSM8K). Table 4 shows the dataset statistics for each domain. The public dataset used for retrieval consists of the training sets from these domains and the general instruction dataset (Alpaca) through concatenation and random shuffling, while the test sets (marked with †) are used for evaluation.

B.2 Baselines

To highlight FedDCA's advantages, Table 5 provides a comparative overview with existing baseline methods along six critical dimensions: Privacy Preserving, API Cost, Additional Information requirements, potential for Performance Degradation, orientation towards Domain Coverage, and the Computational Overhead of Data Augmentation. These dimensions are crucial for assessing the practical viability of federated instruction tuning strategies. We analyze each dimension in detail below:

- **Privacy Preserving**: Most evaluated methods, including FedDCA, inherently preserve client data privacy by design, as they do not share raw local data. Self-Instruct is a notable exception as it involves sending prompts containing potentially sensitive information to an external API. Our detailed privacy analysis for FedDCA can be found in Appendices A.1 and B.5.
- **API Cost**: FedDCA operates without requiring external API calls, thus incurring no associated costs. In contrast, Self-Instruct, by leveraging large models like GPT for instruction generation, entails API expenses.
- **Additional Information**: FedDCA primarily relies on clients' local training data for its op-

Method	Privacy Preserving	API Cost	Additional Information	Performance Degradation	Domain Coverage Oriented	Computational Overhead of Data Augmentation
FedAvg (McMahan et al., 2017)	✔	✗	✗	✔	✗	-
Random Sampling	✔	✗	✗	✔	✗	-
LESS (Xia et al., 2024)	✔	✗	✔	✔	✗	high
FewFedPIT (Zhang et al., 2024c)	✔	✗	✗	✗	✗	high
KnowledgeSG (Wang et al., 2024)	✔	✗	✗	✗	✗	high
Self-Instruct (Wang et al., 2022)	✗	✔	✗	✗	✗	low
Direct Retrieval	✔	✗	✗	✗	✗	low
FedDCA (ours)	✔	✗	✗	✗	✔	low

Table 5: Key differences between FedDCA and other baselines. FedDCA demonstrates the ability of: 1) privacy-preserving, 2) no API cost, 3) no additional information required, 4) avoiding performance degradation, and 5) aiming at domain coverage optimization. LESS requires additional information, as it needs access to the validation set for gradient-based retrieval.

erations. Some methods, such as LESS (Xia et al., 2024), necessitate additional resources like a validation set for its gradient-based retrieval mechanism.

- **Performance Degradation Avoidance**: A key objective is to enhance model performance or, at minimum, avoid degradation. As indicated in our main results (Table 1), FedDCA is designed to prevent performance drops. In contrast, methods like FedAvg (when unaugmented), Random Sampling, and even some generation-based approaches like FewFedPIT and KnowledgeSG (if generation quality is not optimal or well-aligned) can risk performance degradation.
- **Domain Coverage Oriented**: FedDCA is uniquely engineered to explicitly optimize cross-client domain coverage in distributed environments. This strategic focus is a distinguishing feature largely absent in the other compared baselines, which typically do not have a direct mechanism for maximizing instruction diversity across the federation.
- **Computational Overhead of Data Augmentation**: FedDCA maintains a low computational overhead for its data augmentation phase, comparable to methods like LESS (due to gradient computation and warm-up), FewFedPIT (local LLM generation), and KnowledgeSG (expert model generation on the server) inherently involve higher computational demands for data augmentation.

In conclusion, these comparisons underscore FedDCA's well-rounded design. It effectively addresses key practical challenges by preserving privacy, avoiding API costs, minimizing reliance on additional information, preventing performance degradation, strategically optimizing domain coverage, and maintaining a low computational overhead for augmentation.

B.3 Implementation Details

We consider FedDIT in the cross-device scenario, $N = 10$ clients, $\mathcal{R} = 30$ rounds, where we randomly sample 2 clients to be available for each round. Then, each available client performs FedDIT for 10 steps with AdamW optimizer, and the batch size is $B = 32$ in a round. The initial learning rate is $5e-5$ with a cosine learning rate scheduler. Our experiment utilizes the widely used LLM, Llama3-8B[2] as the base model with 2048 max sequence length and adopts LoRA tuning method. The rank of LoRA is 16, and the scalar alpha is 16. For k-means (Wu, 2012), we set cluster num $\xi = 10$ and for FedDCA we set the similarity threshold $\alpha = 0.7$. We utilize `bge-large-en-v1.5`[3] as both the client and server's encoder as default, which outputs embeddings of 1024 dimensions.

Specifically, for LESS, we uses LoRA with rank 128 and dropout 0.1 for warmup training on a random 5% subset of the public dataset for 4 epochs. We apply a learning rate peak of 2×10^{-5} with linear warm-up and cosine decay. Gradients are extracted from each epoch checkpoint and projected to a 1024-dimensional space using Johnson-Lindenstrauss random projections. Data selection is performed by computing cosine similarity between training and validation gradient features aggregated across epochs. The top N_k^p scoring examples are selected for final client c_k's instruction tuning. Additionally, for KnowledgeSG, we utilize

[2] https://huggingface.co/meta-llama/Meta-Llama-3-8B

[3] https://huggingface.co/BAAI/bge-large-en-v1.5

You are asked to come up with instructions. Don't repeat instructions in examples. Here are some examples:
Instruction 1: {}
Instruction 2: {}
Provide a new instruction below:

Example 1:
Instruction: {}
Response: {}
Generate the response of this instruction,
Instruction: {}

Figure 4: Prompts used in the Self-Instruct data generation. (a) Prompt for generating new instructions. Two examples are randomly sampled from the client's local data for in-context demonstration. (b) Prompt for generating responses. We prompt GPT-3.5 to generate responses with a randomly selected example for one-shot in-context learning.

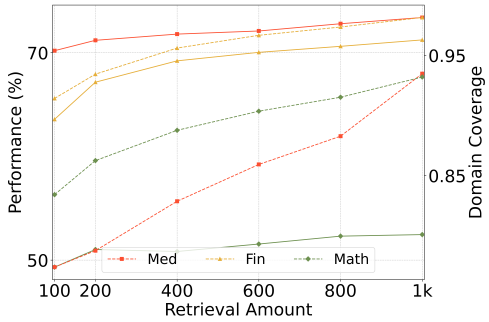


Figure 5: Impact of different retrieval amounts on FedDCA's performance (%) and domain coverage. The dotted lines represent the domain coverge. Results demonstrate FedDCA's robust applicability across different augmentation scales.

AlpacaCare (Zhang et al., 2023d), FinGPT (Yang et al., 2023a) and WizardMath (Luo et al., 2023) as the server-hosted expert model for each domain respectively. To generate the Self-Instruct data, we prompt GPT-3.5 to generate the instruction with the designed prompt in Figure 4. Specifically, we randomly sample two examples from the client's local data to guide GPT-3.5 generating the in-domain instruction and one example from the client's local data for one-shot in-context learning to guide GPT-3.5 generating responses into the example's format.

B.4 Effect of Retrieval Number

Experimental Setup. To assess the impact of the volume of augmented data, we vary the number of public instructions retrieved per client by FedDCA. Specifically, we test retrieval amounts of [100, 200, 400, 800, 1000] instructions per client, while keeping other experimental parameters consistent with the main setup described in Section 6.1. The experiments are conducted across the three primary domains (medical, financial, and mathematical).

Results. Figure 5 shows FedDCA's performance and domain coverage with varying retrieval amounts. Both generally increase with more retrieved instructions, indicating richer augmentation enhances learning. The improvement rates slow after about 400 instructions per client, likely due to increasing overlap with already covered semantic space or the model reaching its capacity to benefit from further diverse examples under the given training regime. Overall, performance and coverage continue to rise with retrieval volume.

Notably, domain coverage's impact on performance differs by domain. For example, medical domain gains are modest, while financial domain gains are more aligned with coverage increases. This variance may stem from: 1) The base model's initial domain proficiency, which influences its baseline performance and the marginal benefits of additional data. 2) The degree of similarity between the public in-domain data distribution and the test set distribution, where higher relevance yields more proportional performance benefits.

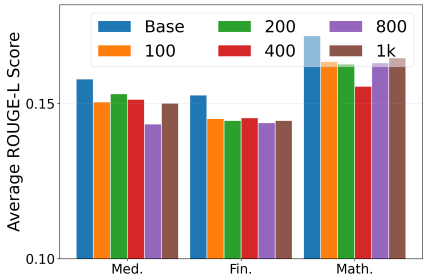
Below is an instruction that describes a task. Write a response that appropriately completes the request.
###Instruction:

Figure 6: Prompt used for memory extraction attack.

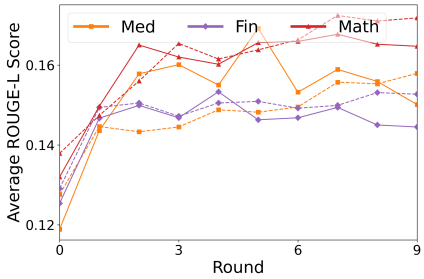
B.5 Privacy Analysis

Memory Extraction Attack. We evaluate the privacy-preserving capability of different ratios of public data against memory extraction attacks (Carlini et al., 2021; Zhang et al., 2024b), which utilizes the autoregression nature of LLM(Xu et al., 2024b).

We focus on one client's instruction tuning in FedDIT, using FedDCA for instruction augmentation with 100 to 1k public instructions. We set up 10 clients with full participation for 10 rounds. Specifically, we record the average ROUGE-L



(a) Different amounts of augmented public data for FedDIT.



(b) The average ROUGE-L score per round. Dotted lines represent the base-data-only setting.

Figure 7: Privacy analysis of memory extraction attacks. Data augmentation can effectively mitigate the privacy leakage risk.

score (Lin, 2004) for client c_0 in each round (following the setup in (Zhang et al., 2024c)). As we use Llama3-8B as our base model and format the instructions and responses into the Alpaca's format, to utilize the auto-regression nature of LLM to extract the instruction, we prompt the model to generate the instruction using the prompt in Figure 6, which is exactly the prefix of the Alpaca's template.

For each setting, we repeat memory extraction 100 times and report the average ROUGE-L score. Specifically, denote the generated $\mathcal{N}$ instructions as $\mathcal{I}$ and the client's local instructions $\mathcal{I}^l$. The calculation of the average ROUGE-L score is defined as $\frac{1}{\mathcal{N}} \sum_{i=1}^{\mathcal{N}} \texttt{ROUGE-L}(\mathcal{I}_i, \mathcal{I}^l)$.

Figure 7(a) shows no significant correlation between the public data ratio and privacy-preserving capability in the same training round. In addition, only using local data has a higher risk of privacy leakage than augmented methods. Additionally, Figure 7(b) shows the trends of the average ROUGE-L score per round. Initially, the average ROUGE-L score for augmented settings increases, then decreases or converges, while the base-data-only scores continue to rise, especially in the code domain. This indicates that with more training rounds, base-data-only fine-tuning captures more privacy information, while the privacy leakage risk in augmented fine-tuning decreases or converges.

Domain Inference Attack. The server may inference the clients' data domain when the domain data retrieval is performed on the server side. However, the proposed algorithm FedDCA is independent of the presence of a public dataset on the server. Even if the server does not have a public dataset, clients can upload their cluster centers to the server, which selects a set of client centers and sends them back to the clients. Each client can then retrieve data from the website based on the received client center by itself, thereby achieving data augmentation while maximizing the cross-client domain coverage.

In that case, since the server does not know the encoder used by the client, it cannot infer the semantic meaning of the embedding. Thus, for the server, it becomes significantly more challenging to infer the client's domain, let alone apply any privacy protection techniques to the embeddings.

We provide two examples for illustration. Two different encoders are used as client's and server's respectively: `BAAI/bge-large-en-v1.5` (denoted as w_1) and `google-bert/bert-large-uncased` (denoted as w_2). Both encoders output 1024-dimensional features.

Example 1: Both w_1 and w_2 take "`hello world`" as input, and the cosine similarity between their embeddings is **0.1829**.

Example 2: Three instructions are used:

- Instruction 1: `Create an array of length 5 which contains all even numbers between 1 and 10.`
- Instruction 2: `Write a replace method for a string class which replaces the given string with a given set of characters.`
- Instruction 3: `What is the sentiment of this news? Please choose an answer from {negative/neutral/positive}. Teollisuuden Voima Oyj, the Finnish utility known as TVO, said it shortlisted Mitsubishi Heavy's EU-APWR model along with reactors from Areva, Toshiba Corp., GE Hitachi Nuclear Energy, and Korea Hydro & Nuclear Power Co.`

For these instructions, Instruction 1 is passed to w_1 and Instructions 2 and 3 are passed to w_2,

	MMLU-Med	FPB	GSM8K
Zero-shot	70.60/-	55.94/-	23.27/-
Base Data	72.40/0.8377	66.74/0.9339	49.12/0.8709
Random Sampling	69.90/0.8497	61.05/0.9408	47.23/0.8812
FedDCA	**73.30/0.9090**	**67.16/0.9800**	**50.26/0.9118**

Table 6: Scalability. Performance (%) and domain coverage of FedDCA and other baselines. 100 clients with 2 clients per round. Results show the strong scalability of FedDCA.

which will result in three embeddings: e_1, e_2, and e_3. The cosine similarity between e_1 and e_2 is **0.1464**, while the similarity between e_1 and e_3 is **0.1879**. Instructions 1 and 2 are in the same domain, whereas they have a lower cosine similarity.

In conclusion, as demonstrated above, when the clients do not perform domain-specific instruction retrieval on the server side, the server cannot infer the client's domain based on the uploaded embeddings.

B.6 Scalability

Experimental Setup. To evaluate the scalability of FedDCA, we conduct experiments with 100 clients, where 2 clients are randomly selected for FedDIT in each round. As the number of clients increases, the amount of local data on each client gradually grows, allowing us to assess how FedDCA performs in larger-scale distributed settings.

Results. Table 6 presents the performance and domain coverage across different domains. Interestingly, we observe that the model trained on only base data even outperforms Random Sampling in domains other than code and narrows the gap with FedDCA. This suggests that with sufficient local data, the benefits of data augmentation become more nuanced. However, FedDCA maintains its advantage by consistently achieving higher domain coverage and better performance across all domains, demonstrating its effectiveness in large-scale federated settings. This superior performance can be attributed to FedDCA's strategic approach to maximizing cross-client domain coverage, which remains crucial even when dealing with a larger overall volume of local data distributed across a greater number of clients.

B.7 Data Selection Scenarios

Experimental Setup. We conduct the experiments on two settings: A) For each domain's FedDIT, we randomly dispatch the in-domain data equally to each client. For example, in the code domain, as the size of the public dataset is 20,022, each client is assigned about 2,000 samples. Then we perform FedDIT on each client's whole data. B) Following A's setting, we first perform FedDCA to determine the client center set, and then we conduct the data selection based on the client center set through dense retrieval to select 200 samples for each client. Finally, we perform FedDIT on the selected data.

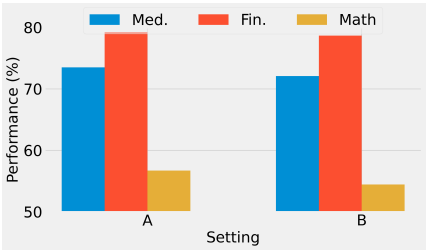


Figure 8: Data selection scenarios. We compare the performance of using the full dataset (Setting A) and using only 10% of data selected through FedDCA (Setting B), demonstrating FedDCA's efficiency in data utilization.

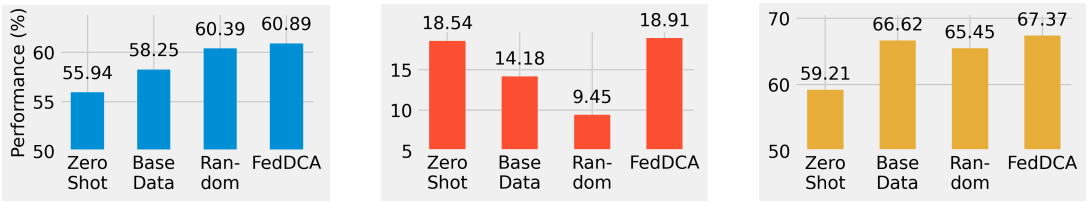
Results. Figure 8 demonstrates that FedDCA can achieve comparable performance to training with the full dataset by using only 10% data through data selection. This efficiency in data utilization highlights the effectiveness of FedDCA's data selection strategy, which maintains model performance while significantly reducing the required training data.

B.8 Held-out Setting

If distributed clients aim to solve tasks based on existing knowledge, the public dataset will inevitably contain knowledge relevant to those domains. This could come from the original corpus (which can be converted into instruction-response pairs by GPT) or from pre-constructed instruction datasets on the website. So the distribution of the public dataset can be categorized as follows: containing **held-in** or **held-out** instructions. The held-in indicates that the public dataset contains instructions for the specific task that clients aim to solve, while the held-out indicates that the public dataset does not contain this task's instructions. The paper's default setting is the held-in setting.

Considering the held-out setting in the financial domain, given that the training set FinGPT and the test sets FPB, FiQA, and TFNS are all related to sentiment analysis tasks. We keep the setting of test sets and replace the FinGPT's instructions in public data with data from a financial QA dataset (`Sujet-Finance-Instruct-177k`[4]). The clients'

[4] https://huggingface.co/datasets/sujet/



(a) FPB (b) FiQA (c) TFNS

Figure 9: Held-out setting. We report the performance of FedDCA and other baselines for (a) FPB, (b) FiQA, and (c) TFNS datasets. Results show the robustness of FedDCA in the held-out setting.

local data are still randomly sampled from FinGPT. This approach yields held-out public data.

Figures 9(a) to 9(c) shows, FedDCA still achieves performance improvements compared to other baselines for the held-out setting. Additionally, using the Random Sampling data augmentation strategy resulted in performance degradation on the FiQA dataset. This further underscores the necessity of selecting an appropriate data augmentation strategy.

B.9 Empirical Tightness of FedDCA's Greedy Approximation

FedDCA's optimization objective is *equivalent to the classic sub-modular facility-location problem*, which guarantees that the simple greedy algorithm attains at least a $1 - 1/e$ fraction of the optimal value in the worst case. To gauge how tight this bound is in practice, we benchmark the FedDCA algorithm against increasingly stronger (but more expensive) baselines on three domains: Medical, Financial and Mathematical. In particular we position FedDCA between the **beam search** (with various beam widths) and an *infeasible* **brute-force solver** that attempts to enumerate all candidate subsets. The beam search, which explores multiple promising branches simultaneously, serves as **a realistic upper bound** that is strictly better than greedy yet still tractable.

Experimental Setup.

1. **Metric:** Domain coverage $d(D^d, \mathcal{P})$ where $\mathcal{P}$ is the selected N client centers, defined as Eq. 2.
2. **Baselines:** $\mathcal{C}_{\text{all}}$ **(default FedDCA)** selects client centers using the union of each client's uploaded cluster center set $\mathcal{C}_{\text{all}}$ as the selection reference to compute marginal gains; $\mathcal{P}_C^*$ is the unreachable optimum under the same selection reference $\mathcal{C}_{\text{all}}$—marked *timeout*, requiring evaluation of $\binom{100}{10} \approx 1.73 \times 10^{13}$ distinct candidate sets; D^d uses D^d as the selection reference to compute marginal gains, which is often impractical in real-world federated settings (please refer to Section 5.2); $\mathcal{P}_D^*$ is the optimum under D^d—also *timeout*; **Beam search** with widths $256, 512, 1024, 2048$ and, for Medical, 9182 (does not exceed the baseline D^d until width 9182).
3. **Approximation ratio:** We report

$$\frac{d(D^d, \mathcal{P}_{\mathcal{C}_{\text{all}}})}{\max_w d(D^d, \mathcal{P}_{\text{beam-}w})} \times 100\%, \qquad (10)$$

where the denominator is the best coverage attained by the beam search. $\mathcal{P}_{\mathcal{C}_{\text{all}}}$ is the selected client center set by FedDCA using $\mathcal{C}_{\text{all}}$ as the selection reference.

Results. As shown in Table 7, FedDCA using $\mathcal{C}_{\text{all}}$ already achieves *nearly optimal* coverage in Medical and Financial domains, within 99.85% and 99.43% of an extensive beam search respectively, and still attains a respectable 86.30% in the more challenging Mathematical domain.

Note that both theoretical optima ($\mathcal{C}_{\text{all}}$ and D^d as reference) are computationally unattainable (*timeout*), so the beam search constitutes the strongest practical competitor.

In conclusion, these real-world ratios vastly exceed the theoretical lower bound of $1-1/e \approx 63\%$, further corroborating that the simple greedy procedure is both **efficient** and **effectively near-optimal** for real-world federated deployments.

C Robustness to Data Heterogeneity

The efficacy of retrieval-based augmentation strategies in FedDIT, such as Direct Retrieval and our proposed FedDCA, is inherently linked to the semantic characteristics of clients' local private data, as these inform the selection of public instructions. A critical consideration is how varying degrees of

`Sujet-Finance-Instruct-177k`

Domain	$\mathcal{C}_{\text{all}}$	$\mathcal{P}_C^*$	D^d	$\mathcal{P}_D^*$	Beam-256	Beam-512	Beam-1024	Beam-2048	Beam-9182	Approx. Ratio
Med.	0.6772	*timeout*	0.6782	*timeout*	0.6776	0.6776	0.6776	0.6778	0.6782	**99.85%**
Fin.	0.8379	*timeout*	0.8418	*timeout*	0.8422	0.8427	0.8427	0.8427	–	**99.43%**
Math.	0.6391	*timeout*	0.6699	*timeout*	0.7401	0.7401	0.7404	0.7405	–	**86.30%**

Table 7: Empirical tightness analysis of FedDCA's greedy approximation. Both theoretical optima ($\mathcal{P}_C^*$ and $\mathcal{P}_D^*$) are computationally unattainable (*timeout*). The approximation ratio is computed against the best beam search result.

inter-client data heterogeneity impact the ability to construct diverse and comprehensively covering augmented datasets. Intuitively, if clients possess highly similar local data, methods relying on these local signals might struggle to achieve broad cross-client domain coverage and instruction diversity in the augmented sets, which are crucial for a well-generalized global model.

This section, therefore, investigates the robustness of FedDCA, Direct Retrieval, and Random Sampling to a spectrum of inter-client data heterogeneity. We focus on these methods as their augmentation processes are either directly influenced by local data distributions (FedDCA, Direct Retrieval) or serve as a data-agnostic baseline (Random Sampling); other generative or complex retrieval methods often have different operational dependencies. We control heterogeneity using a Dirichlet distribution (Dir(β)) over semantic clusters of clients' private instructions, where smaller β values yield higher inter-client heterogeneity (more distinct local data) and larger β values lead to lower heterogeneity (more similar local data). We evaluate performance across $\beta \in \{0.01, 0.1, 1.0, 10\}$.

C.1 Experimental Setup

Inter-Client Data Heterogeneity Control. We focus on the Financial domain, using FinGPT as the source for private instructions. First, we perform k-means clustering ($\xi = 100$ clusters, consistent with Section 4.1) on FinGPT instruction embeddings to create semantic pseudo-labels. To generate local data for 10 clients, each holding 100 instructions, we employ a Dirichlet distribution Dir(β) over these clusters to define each client's data composition, simulating varying degrees of inter-client data heterogeneity for $\beta \in \{0.01, 0.1, 1.0, 10\}$.

Augmentation and Training. Following the general setup in Section 6.1, Random Sampling, Direct Retrieval (introduced in Section 3), and FedDCA are used to obtain 1k augmented instructions per client before commencing FedDIT.

Evaluation Metrics. Model performance is evaluated using FPB, FiQA, and TFNS accuracy. Augmented data characteristics are assessed using three key metrics:

1) **Domain Coverage (↑)**: As defined in Eq. 2, this measures how well the aggregated client data (local + augmented) D^c covers the target financial domain D^d.

2) **ICACS (Inter-Client Augmented Centroids Similarity ↓)**: For each client k, its augmented instruction embeddings are clustered (k-means, $\xi = 10$) into centers $\mathcal{C}_k$. ICACS is the average pairwise cosine similarity between all such cluster centers across different clients. Lower values indicate more distinct augmented sets at a granular level.

3) **RUAI (Ratio of Unique Augmented Instructions ↑)**: The ratio of unique instructions within the total cross-client augmented data pool D^c. Higher values signify less redundancy.

C.2 Results and Analysis

Table 8 systematically quantifies the impact of inter-client data heterogeneity on augmentation strategies. For small β (i.e., high heterogeneity), the cross-client data exhibits substantial diversity, enabling Direct Retrieval to access a broader range of semantic regions in the public pool. This diversity translates into clear gains over Random Sampling in both performance (e.g., +6% FiQA at $\beta = 0.01$) and coverage metrics. However, as β increases and client data distributions become more homogeneous, the advantage of Direct Retrieval diminishes, with its performance and diversity metrics converging toward those of Random Sampling. This trend underscores that the effectiveness of Direct Retrieval is fundamentally driven by the diversity present in cross-client data.

Conversely, as inter-client heterogeneity decreases (i.e., local data distribution becomes more similar), Direct Retrieval's effectiveness continuously degrades. Its model performance and domain coverage drop, showing a more sensitive effectiveness to the heterogeneity. Results indicate that with

Dirichlet β	Augmentation Strategy	Model Performance (%)			Augmented Data Characteristics		
		FPB	FiQA	TFNS	Domain Coverage (↑)	ICACS (↓)	RUAI (↑)
0.01	Random Sampling	64.20	13.10	65.55	0.920	0.472	0.867
	Direct Retrieval	67.20	21.50	69.00	0.938	0.434	0.909
	FedDCA	**67.90**	**36.30**	**74.40**	**0.985**	**0.405**	**0.943**
0.1	Random Sampling	64.19	13.09	65.53	0.920	0.472	0.867
	Direct Retrieval	66.31	19.11	67.62	0.929	0.441	0.882
	FedDCA	**67.24**	**35.27**	**73.32**	**0.982**	**0.411**	**0.938**
1.0	Random Sampling	64.19	13.09	65.53	0.919	0.472	0.867
	Direct Retrieval	65.50	17.80	66.79	0.924	0.448	0.879
	FedDCA	**66.94**	**34.82**	**72.66**	**0.977**	**0.414**	**0.927**
10	Random Sampling	64.20	13.05	65.45	0.918	0.472	0.867
	Direct Retrieval	64.37	15.20	66.10	0.922	0.452	0.871
	FedDCA	**66.73**	**34.51**	**72.23**	**0.972**	**0.423**	**0.921**

Table 8: Performance (%), domain coverage, ICACS and RUAI across different levels of client data heterogeneity (controlled by Dirichlet parameter β) in the Financial Domain. Best model performance, domain coverage, ICACS and RUAI per β setting are in **bold**.

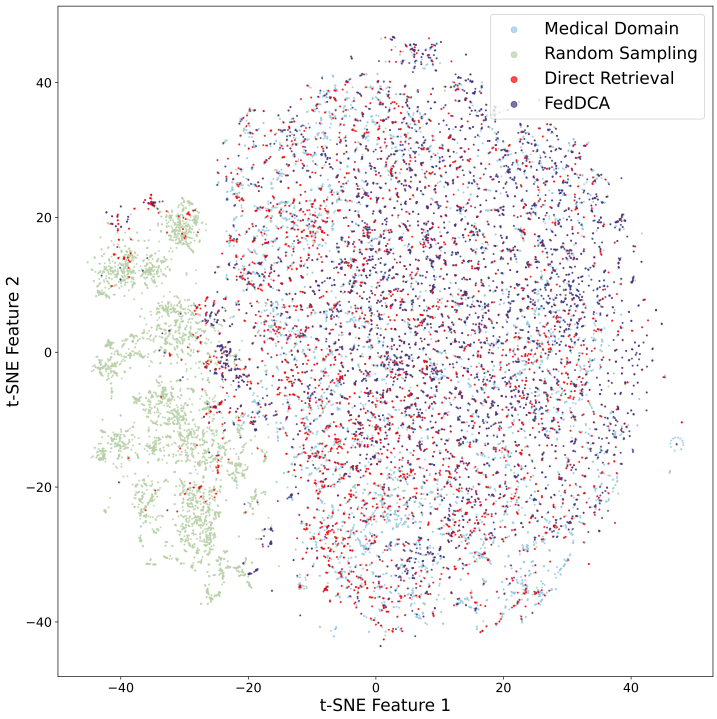
more homogeneous local data distributions, Direct Retrieval tends to fetch more similar public instructions due to the similarity between each client's cluster centers.

Random Sampling exhibits stable, albeit consistently sub-optimal, performance and diversity metrics across all heterogeneity levels. This stability stems from its sampling strategy (oblivious to local data distributions) of the public dataset, which results in similar augmented data.

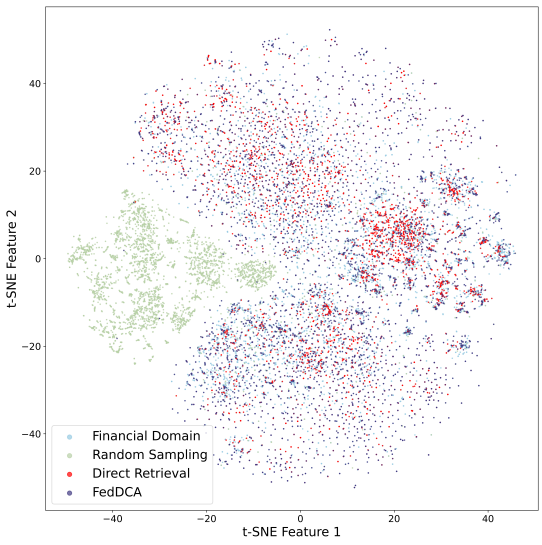
Additionally, FedDCA showcases strong robustness. It consistently achieves the best model performance across all tested β values. While its diversity metrics (domain coverage, ICACS, RUAI) show minor sensitivity to the cross-client data distribution (optimally diverse at highest heterogeneity). This resilience stems from its coverage-oriented client center selection (Algorithm 1), which effectively prevents redundant augmentation by selecting public instructions that add novel semantic aspects to the collective, even with similar cross-client data distributions. In conclusion, these findings underscore FedDCA's robustness and adaptability irrespective of the specific degree of inter-client heterogeneity, makes it a more reliable and effective solution for real-world FedDIT deployments.

D Augmentation Strategy Visualization

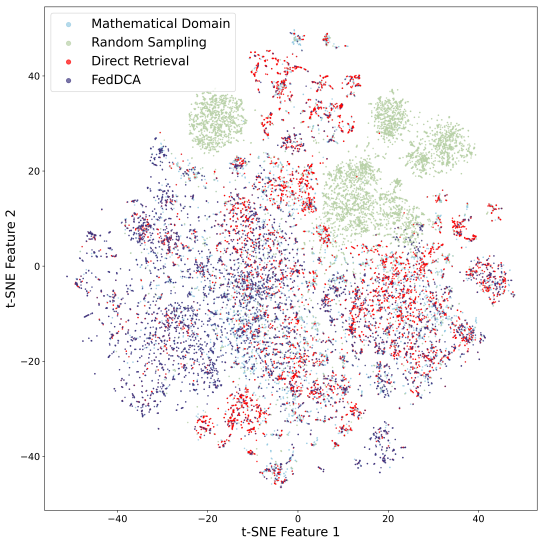
To more intuitively compare the domain coverage of different instruction augmentation methods, we randomly sample 5k cross-client instructions obtained through these methods and 10k in-domain instructions as the background, representing the distribution of specific domains in the public dataset. We then visualized the results using t-SNE (van der Maaten and Hinton, 2008), as shown in Figure 10. The plot shows that FedDCA encompasses most of the in-domain data, which is consistent with FedDCA's domain coverage of each domain shown in Table 1. Also, we can observe that the random sampling strategy selects a lot of out-of-domain data while does not have good coverage in specific domains.



(a) Medical



(b) Financial



(c) Mathematical

Figure 10: Visualization of cross-client data distribution in different domains, performing t-SNE dimensionality reduction on retrieved instructions through various augmentation strategies. We randomly sample 10k in-domain samples as background while randomly sampling 5k samples from the cross-client augmented dataset for different instruction augmentation methods for comparison.