Addressing Data Heterogeneity, Scarcity, and Training Efficiency in Privacy-Preserving Federated Learning

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig-Maximilians-Universität München

Haokun Chen

München, 2025

Addressing Data Heterogeneity, Scarcity, and Training Efficiency in Privacy-Preserving Federated Learning

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig-Maximilians-Universität München

vorgelegt von
Haokun Chen
aus Hunan, China

München, den 14.06.2025

Erstgutachter:	Prof. Dr. Volker Tresp
Zweitgutachter:	Prof. Dr. Wojciech Samek
Drittgutachter:	Prof. Dr. Ruben Mayer
Tag der Disputation:	22.10.2025

Eidesstattliche Versicherung

(Siehe Promotionsordnung vom 12.07.11, § 8 Abs. 2 Pkt. 5.)

Hiermit erkläre ich, Haokun Chen, an Eides statt, dass die vorliegende Dissertation von mir selbstständig und ohne unerlaubte Beihilfe angefertigt worden ist.

München, den 14.06.2025 *Haokun Chen*

Contents

Abstract **vii**

Zusammenfassung **ix**

Acknowledgments **xi**

List of Publications and Declaration of Authorship **xiii**

1 Introduction **1**

1.1 Overview of Federated Learning 1

1.1.1 Motivation 1

1.1.2 Definition 2

1.1.3 Categorization 3

1.1.4 Application 5

1.2 Threats and Defenses 6

1.2.1 Model Utility Attack 6

1.2.2 Defense Against Model Utility Attack 8

1.2.3 Client Privacy Attack 10

1.2.4 Defense Against Client Privacy Attack 11

1.2.5 Our Contribution 13

1.3 Data Heterogeneity and Scarcity 13

1.3.1 Feature Distribution Skew 14

1.3.2 Label Distribution skew 15

1.3.3 Data Quantity Skew 16

1.3.4 Concept Drift 16

1.3.5 Our Contribution 17

1.4 Communication and Computation 17

1.4.1 Server-Client Communication Overhead 18
1.4.2 Client Computational Burden 18
1.4.3 Our Contribution 19

2 FRAug: Tackling Federated Learning with Non-IID Features via Representation Augmentation 21

3 FedDAT: An Approach for Foundation Model Finetuning in Multi-Modal Heterogeneous Federated Learning 39

4 FedPop: Federated Population-based Hyperparameter Tuning 53

5 FedBiP: Heterogeneous One-Shot Federated Learning with Personalized Latent Diffusion Models 69

6 Conclusion 89

Bibliography 91

Abstract

Over the past decade, deep learning has achieved significant breakthroughs in various domains, including computer vision and natural language processing, with applications spanning industries such as healthcare and manufacturing. However, the success of deep neural networks (DNNs) relies heavily on access to large-scale datasets, which can be difficult for individual organizations to acquire. The reasons behind are the complexities and costs for data collection and annotation at large scale. A straightforward solution is to centralize data from multiple organizations for training. However, this approach raises significant privacy concerns, as such data often contains sensitive or confidential information. Furthermore, regulations like the General Data Protection Regulation (GDPR) emphasize the importance of protecting user privacy in inter-organizational data exchanges. Moreover, transmitting large volumes of data introduces substantial computational overhead, further complicating the process. These challenges highlight the urgent need for methods that facilitate collaborative data use while ensuring privacy preservation.

Federated learning (FL), which enables multiple parties to collaboratively train a DNN with the assistance of a central server, offers an effective solution to the aforementioned problem. Unlike traditional centralized learning, which requires collecting data from each party, FL eliminates the need to upload data for joint training. Instead, locally trained models are exchanged with a central server, which aggregates the knowledge from all of the uploaded models and then distributes a refined global model to each party. This approach allows each party to benefit from the collective contributions, ultimately enhancing the model performance. This thesis makes four key contributions to addressing challenges in federated learning, including data heterogeneity, data scarcity, and system convergence rate, while preserving client data privacy. For each contribution, we propose a novel method and empirically demonstrate its effectiveness within the relevant problem context.

In this thesis, we address key challenges in federated learning (FL) related to data heterogeneity, foundation model adaptation, hyperparameter tuning, and communication efficiency. First, we tackle feature space heterogeneity by introducing a generative augmentation method that aligns diverse client distributions, validated on both public and real-world datasets. We then adapt parameter-efficient fine-tuning techniques for vision-language models in FL by designing a dual-stream adapter that captures both client-specific and client-agnostic knowledge. To optimize performance under limited resources and non-IID conditions, we propose an evolutionary hyperparameter tuning framework that enables efficient online optimization. Lastly, we address data scarcity and heterogeneity in One-Shot FL by personalizing pretrained latent diffusion models, enabling privacy-preserving synthetic data generation that improves performance across challenging domains such as medical and satellite imaging.

Zusammenfassung

In den letzten zehn Jahren hat das Deep Learning bedeutende Durchbrüche in verschiedenen Bereichen erzielt, darunter Computer Vision und die Verarbeitung natürlicher Sprache, mit Anwendungen in Branchen wie dem Gesundheitswesen und der Fertigung. Der Erfolg tief neuronaler Netzwerke (DNNs) beruht jedoch maßgeblich auf dem Zugang zu groß angelegten Datensätzen, die für einzelne Organisationen oft schwer zu beschaffen sind. Gründe dafür sind die Komplexität und die hohen Kosten der Datenerhebung und -annotation im großen Maßstab. Eine naheliegende Lösung besteht darin, Daten mehrerer Organisationen zentral zu sammeln und gemeinsam zu nutzen. Dieses Vorgehen wirft jedoch erhebliche Datenschutzbedenken auf, da solche Daten häufig sensible oder vertrauliche Informationen enthalten. Darüber hinaus betonen gesetzliche Regelungen wie die Datenschutz-Grundverordnung (DSGVO) die Bedeutung des Schutzes der Privatsphäre bei unternehmensübergreifendem Datenaustausch. Zusätzlich verursacht die Übertragung großer Datenmengen einen erheblichen rechnerischen Mehraufwand, was den Prozess weiter erschwert. Diese Herausforderungen verdeutlichen den dringenden Bedarf an Methoden, die eine kollaborative Datennutzung ermöglichen und gleichzeitig den Datenschutz gewährleisten.

Föderiertes Lernen (Federated Learning, FL) bietet hierfür eine effektive Lösung. Es ermöglicht mehreren Parteien, mit Unterstützung eines zentralen Servers gemeinsam ein DNN zu trainieren. Im Gegensatz zum traditionellen zentralisierten Lernen, bei dem alle Daten zusammengeführt werden müssen, entfällt beim FL die Notwendigkeit, Daten zur gemeinsamen Modellbildung hochzuladen. Stattdessen werden lokal trainierte Modelle mit dem zentralen Server ausgetauscht. Dieser aggregiert das Wissen aus allen hochgeladenen Modellen und verteilt anschließend ein verfeinertes globales Modell an alle Teilnehmer. Auf diese Weise profitieren alle Beteiligten von den kollektiven Beiträgen, was letztlich die Modellleistung verbessert. Diese Arbeit leistet vier wesentliche Beiträge zur Lösung zentraler Herausforderungen im Bereich des föderierten Lernens, darunter Datenheterogenität, Datenknappheit und Konvergenzgeschwindigkeit, bei gleichzeitiger Wahrung der Datenprivatsphäre der einzelnen Teilnehmer.

Für jeden dieser Beiträge schlagen wir eine neuartige Methode vor und belegen deren Wirksamkeit empirisch im jeweiligen Problemkontext.

In dieser Arbeit befassen wir uns mit zentralen Herausforderungen des föderierten Lernens (FL), insbesondere im Hinblick auf Datenheterogenität, die Anpassung von Foundation Models, Hyperparameteroptimierung und Kommunikationseffizienz. Zunächst adressieren wir die Heterogenität im Merkmalsraum durch eine generative Augmentierungsmethode, die unterschiedliche Verteilungen der Clients ausgleicht und sowohl auf öffentlichen als auch auf realen Datensätzen validiert wurde. Anschließend passen wir parameter-effiziente Fine-Tuning-Techniken für Vision-Language-Modelle im FL-Kontext an, indem wir eine Dual-Stream-Adapter-Architektur entwerfen, die sowohl client-spezifisches als auch client-unabhängiges Wissen erfasst. Zur Optimierung der Modellleistung unter beschränkten Ressourcen und nicht-i.i.d. Bedingungen schlagen wir ein evolutionäres Hyperparameter-Tuning-Verfahren vor, das eine effiziente Online-Optimierung ermöglicht. Abschließend gehen wir das Problem der Datenknappheit und -heterogenität im One-Shot Federated Learning (OSFL) an, indem wir vortrainierte Latent Diffusion Models personalisieren. Dies erlaubt die datenschutzfreundliche Generierung synthetischer Daten, was die Leistung in anspruchsvollen Domänen wie der medizinischen und satellitengestützten Bildgebung deutlich verbessert.

Acknowledgments

This dissertation is the product of the past four years I have spent at Siemens and Ludwig-Maximilians-Universität München. During this time, I have been fortunate to receive support from many incredible individuals—without whom the completion of this thesis would not have been possible.

First and foremost, I would like to express my deepest gratitude to my supervisor, Prof. Dr. Volker Tresp, for his guidance throughout my entire PhD journey. Volker granted me the freedom to explore exciting research directions and define my own research questions, while always being available for insightful discussions, constructive advice, and invaluable feedback. He is an inspiring role model for what it means to be a great researcher and inventor. I am also deeply honored that Prof. Dr. Wojciech Samek and Prof. Dr. Ruben Mayer have agreed to serve as external examiners for my thesis.

I would like to extend my sincere thanks to Siemens for funding my research. I am especially grateful to Dr. Denis Krompass, who offered me a PhD position in his research group, providing the essential resources and a stimulating, supportive environment. His invaluable guidance, insightful feedback, and unwavering encouragement have been instrumental to my research and publications. Additionally, I would like to thank Dr. Sebastian Szyller, Dr. Weilin Xu, and Dr. Nageen Himayat for their valuable support and supervision during my internship at Intel.

I am profoundly grateful to all my co-authors and collaborators, who have significantly contributed to the publications arising from this work. Special thanks go to Dr. Jindong Gu and Dr. Ahmed Frikha, whose support and guidance have been invaluable throughout my PhD. I also extend my heartfelt appreciation to Dr. Zhiliang Wu and Dr. Felix Buggenthin for their help with company projects. I am grateful to my other co-authors, Yao Zhang, Hang Li, Tong Liu, and Jinhe Bi, for their innovative ideas, insightful discussions, and continuous motivation. Thank you all for your dedication, thought-provoking exchanges, and fruitful collaborations.

Last but not least, I would like to thank my parents for their unconditional love and encouragement in any situation.

List of Publications and Declaration of Authorship

- **Haokun Chen**, Ahmed Frikha, Denis Krompass, Jindong Gu, Volker Tresp. FRAug: Tackling Federated Learning with Non-IID Features via Representation Augmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 4849-4859.

  > *I conceived the original research contributions. I performed all implementations and evaluations. I wrote the initial draft of the manuscript and did most of the subsequent corrections. I regularly discussed this work with my co-author Denis Krompass and Ahmed Frikha. All co-authors contributed to improving the manuscript.*

  This publication serves as Chapter 2 of this thesis.

- **Haokun Chen**, Yao Zhang, Denis Krompass, Jindong Gu, Volker Tresp. FedDAT: An Approach for Foundation Model Finetuning in Multi-Modal Heterogeneous Federated Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024, 38(10): 11285-11293.

  > *I conceived the original research contributions. I performed all implementations and evaluations. I wrote the initial draft of the manuscript and did most of the subsequent corrections. I regularly discussed this work with my co-author Denis Krompass and Yao Zhang. All co-authors contributed to improving the manuscript.*

  This publication serves as Chapter 3 of this thesis.

- **Haokun Chen**, Denis Krompass, Jindong Gu, Volker Tresp. FedPop: Federated Population-based Hyperparameter Tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025, 39(15): 15776-15784.

  > *Denis Krompass and I conceived the original research contributions. I performed all implementations and evaluations. I wrote the initial draft of the manuscript and did most of the subsequent corrections. I regularly discussed this work with my co-author Denis Krompass. All co-authors contributed to improving the manuscript.*

  This publication serves as Chapter 4 of this thesis.

- **Haokun Chen**, Hang Li, Yao Zhang, Gengyuan Zhang, Jinhe Bi, Philip Torr, Jindong Gu, Denis Krompass, Volker Tresp. FedBiP: Heterogeneous One-Shot Federated Learning with Personalized Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

  > *I conceived the original research contributions. I performed all implementations and evaluations. I wrote the initial draft of the manuscript and did most of the subsequent corrections. I regularly discussed this work with my co-author Denis Krompass and Hang Li. All co-authors contributed to improving the manuscript.*

  This publication serves as Chapter 5 of this thesis.

Other Publications

- Yao Zhang, **Haokun Chen**, Ahmed Frikha, Yezi Yang, Denis Krompass, Gengyuan Zhang, Jindong Gu, Volker Tresp. Cl-crossvqa: A continual learning benchmark for cross-domain visual question answering. *arXiv preprint arXiv:2211.10567 (2022).*

- Ahmed Frikha*, **Haokun Chen***, Denis Krompaß, Thomas Runkler, Volker Tresp. "Towards data-free domain generalization. *Asian Conference on Machine Learning. PMLR, 2023*.

- Jinhe Bi, Yujun Wang, **Haokun Chen**, Xun Xiao, Artur Hecker, Volker Tresp, Yunpu Ma. Visual Instruction Tuning with 500x Fewer Parameters through Modality Linear Representation-Steering. *arXiv preprint arXiv:2412.12359 (2024).*

Chapter 1

Introduction

This chapter outlines the technical background required to comprehend the subsequent chapters. Section 1 provides an overview of federated learning, covering its motivation, definition, system categorization, and applications. Sections 2, 3, and 4 delve into the challenges in federated learning that are closely tied to our contributions. In particular, we discuss the motivation for addressing each challenge, review relevant literature, and detail their connection to our proposed solutions.

1.1 Overview of Federated Learning

This section explores the motivation behind federated learning and provides an explanation of how deep learning models are trained within such systems. It also examines the diverse applications of federated learning and the categorization of its systems.

1.1.1 Motivation

Over the past decade, deep learning (DL) has achieved remarkable breakthroughs across various domains. However, its success relies heavily on access to large-scale datasets, which can be challenging for individual organizations to acquire. Key contributing factors include limitations in storage capacity and the significant time and cost complexities associated with data collection. A straightforward solution is to centralize data from multiple organizations onto a single training platform and then perform model optimization. However, this approach raises critical privacy concerns, as the data from different parties often contain sensitive or confidential information. Furthermore, regulations such as the GDPR in European Union and PDPA in Singapore emphasize the importance of protecting user privacy in inter-organizational data exchanges. Additionally, transmitting large volumes of data significantly increases the communication overhead. These challenges highlight the urgent need for methods that enable collaborative data utilization while preserving the privacy of individual parties.

In this context, Federated Learning (FL) —a collaborative learning approach that eliminates the need for sharing users' raw data—has gained significant attention in recent years. While deep learning remains a focal point of research, its integration with federated learning is rapidly emerging as a prominent and dynamic area of exploration.

1.1.2 Definition

The primary goal of FL is to collaboratively train a deep learning model across a set of K clients, each possessing a private dataset D^k that cannot be shared. Before the federated learning process begins, the central server initializes the global model with weights w_0. The model weights are then broadcast to the clients. Subsequently, multiple rounds of server-client communication are conducted, guided by the system's optimization budget.

During each communication round t, the client will optimize the local model w_{t-1}^k using local data:

$$w_t^k = \underset{w}{argmin} \underset{x_i \in D^k}{\mathbb{E}} f(w_{t-1}^k, x_i). \tag{1.1}$$

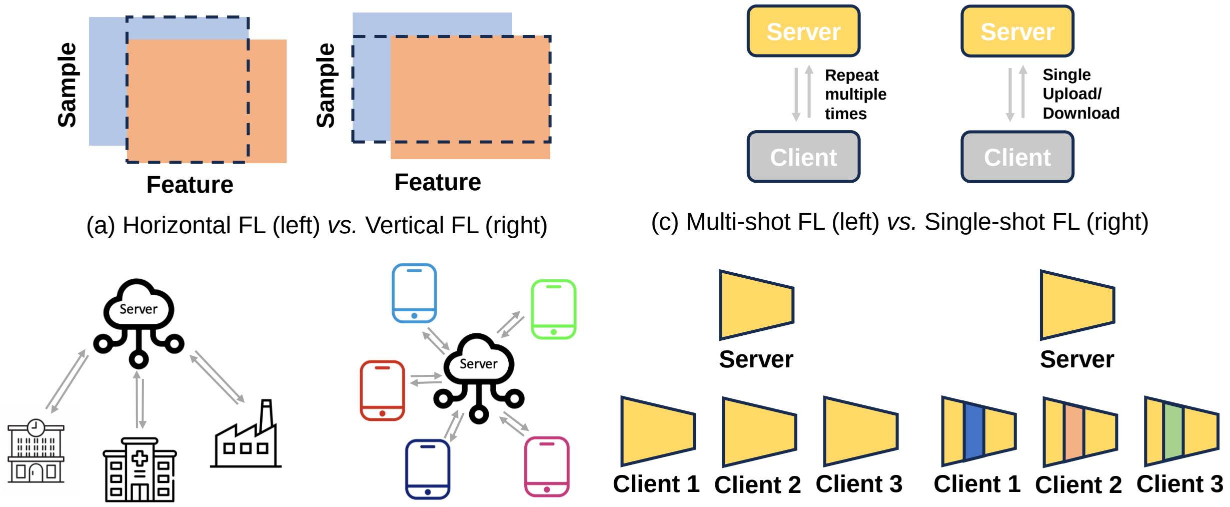
Here, f represents the optimization objective defined by the specific task of the federated learning system. Once all clients complete their local model optimization, the updated model weights are uploaded to the central server, where the model aggregation is performed:

$$w_t = Agg(w_{t-1}, ..., w_t^K, w_t), \tag{1.2}$$

where Agg represents a specific aggregation function. In the most conventional FL aggregation algorithm, i.e., FedAvg [55], the Agg function performs a straightforward averaging of the uploaded client weights:

$$w_t = \frac{1}{K} \sum_{k=1}^{K} w_t^k, \tag{1.3}$$

Once the system's optimization budget is exhausted or the predefined convergence criterion is satisfied, the optimized global model weight are returned.



(a) Horizontal FL (left) *vs.* Vertical FL (right)

(c) Multi-shot FL (left) *vs.* Single-shot FL (right)

(b) Cross-Silo FL (left) *vs.* Cross-Device FL (right)

(d) Global FL (left) *vs.* Personalized FL (right)

Figure 1.1: Categorization of different federated learning systems

1.1.3 Categorization

In the following, we outline the possible categorizations of various FL systems. We provide a schematic illustration of different systems in Figure 1.1.

- **Horizontal vs. Vertical**

  The first type of categorization is based on the nature of data features across different clients. In *vertical* FL, clients possess different features or feature embeddings corresponding to the same set of training entities. For instance, e-commerce platforms may aggregate transaction and payment information from various financial institutions to evaluate user creditworthiness. Similarly, medical institutions may integrate a patient's diagnostic data from multiple hospitals to assess their health status [89].

  In contrast, *horizontal* FL involves clients that share the same feature space but have distinct data samples. This is the more prevalent scenario, where training data is collected independently under different conditions. For example, in industrial anomaly detection, clients may capture images of various machine parts from different experimental setups. Despite the diversity in samples, the extracted features—such as pixel intensities or contextual descriptors—remain consistent across clients [57]. In this work, we focus primarily on the horizontal FL setting.

- **Cross-silo vs. Cross-device**

The second categorization of FL systems is based on the computational and communication capabilities of clients, as well as the size of their private datasets. In *cross-silo* FL, participating entities are typically organizations or data centers. These clients are relatively few in number but possess substantial computational resources. For instance, medical research institutions and hospitals can collaboratively train neural networks for radiographic image classification while keeping patients' chest X-ray data local, as demonstrated in [33].

In contrast, *cross-device* FL involves a large number of clients—primarily mobile devices—with limited computing power and smaller, decentralized datasets. For example, [92] proposed an approach for optimizing a COVID-19 detection model using respiratory sounds and symptom data collected via users' smartphones. Given the constraints of energy consumption and limited hardware capabilities, these devices are often unable to support complex on-device training. As a result, cross-device FL systems must be designed to tolerate unreliable connectivity, device heterogeneity, and intermittent client participation. In this work, we explore both cross-silo and cross-device FL scenarios.

- **Multi-shot vs. One-shot**

  In traditional FL systems, stable model convergence typically requires multiple rounds of communication between clients and a central server—a process commonly referred to as *multi-shot* FL. While effective, this iterative communication model poses significant challenges in resource-constrained environments, where frequent parameter transmission may be impractical or costly. Moreover, repeated exchanges of model updates increase the risk of privacy attacks, such as gradient inversion [106], which can potentially reconstruct sensitive training data from shared gradients.

  As a more communication-efficient alternative, *one-shot* FL [23] limits the client-server interaction to a single round of communication. In this approach, the server first broadcasts an initial model to all clients, who then perform local optimization independently. The updated models are subsequently uploaded in a single step, thereby reducing both communication overhead and exposure to privacy risks.

- **Globalization vs. Personalization**

  A final categorization of FL systems is based on the differences in the final deployed model weights between the server and the clients. In the conventional *global* FL paradigm, the training process culminates in the distribution of a single, globally optimized model to all participating clients. While effective in homogeneous settings, this approach may underperform when client data distributions vary significantly.

In contrast, *personalized* FL introduces greater flexibility by enabling client-specific model adaptation. Personalization can take several forms, including the use of partially or fully client-specific model weights [67], the integration of client-specific modules within a shared architecture [41], or even the deployment of entirely distinct model architectures per client [47]. This tailored approach is especially valuable in addressing the challenges of data heterogeneity and variation in input modalities, ultimately leading to improved performance across diverse client datasets.

1.1.4 Application

In the following, we describe some possible application fields for FL algorithms.

- **Healthcare**

  Protecting patient privacy is a critical concern in the healthcare domain, especially with respect to sensitive data such as Electronic Health Records (EHRs). The collection and sharing of such data are often restricted by stringent regulatory and ethical considerations. FL presents a promising solution to these challenges, particularly in the field of medical imaging. By enabling the distributed training of diagnostic models across multiple hospitals and clinics—using data such as X-rays, MRIs, and CT scans—FL eliminates the need to transfer raw patient data. This decentralized approach allows healthcare institutions to collaboratively develop accurate and robust models for vital tasks such as cancer detection, all while maintaining strict adherence to patient privacy requirements [79].

- **Finance**

  Financial institutions, such as banks, can leverage FL to collaboratively develop more effective fraud detection and credit scoring models. This collaborative approach allows them to improve these models without exposing sensitive financial information like transaction histories and customer details. [50]

- **Industry**

  Within industrial contexts, FL provides a mechanism for companies to develop predictive maintenance models using sensor data collected from machinery deployed across various manufacturing facilities. This approach allows manufacturers to proactively identify potential equipment failures, thereby improving operational efficiency and minimizing downtime, all while maintaining data locality. Moreover, the application of FL extends to collaborative efforts within supply chains, enabling organizations to jointly develop enhanced demand forecasting and inventory management models without compromising the confidentiality of their proprietary data [84].

- **Internet of Things (IoT)**

  FL empowers smart home devices (e.g., thermostats, lighting systems, security cameras) to collectively train models, improving user experience and device performance. Importantly, this is achieved without compromising user privacy, as data remains on individual devices. Likewise, FL enables collaborative model development for health-tracking devices like wearables, allowing for improved models on heart rate, sleep patterns, and physical activity without sharing personal data [7].

In the following, we analyze the main challenges in FL and introduce how we address them in our contributions.

1.2 Threats and Defenses

1.2.1 Model Utility Attack

Unlike traditional centralized learning approaches, Federated Learning (FL) enables collaborative model training without aggregating data in a central repository [32]. Instead, FL distributes the training process across numerous potentially unreliable devices, each retaining private and inaccessible local datasets. This decentralized paradigm introduces new vulnerabilities, as the local training process becomes a potential target for various adversarial attacks [44]. Since only model updates are exchanged while raw data remains on-device, adversaries may exploit this setup to degrade the model's utility or performance.

- **Model Poisoning Attacks**

  Model poisoning attacks [43] involve adversaries directly manipulating local model updates prior to their transmission to the central server. A common approach is to inject fixed perturbations or completely replace benign gradient parameters with malicious ones. Some attacks focus on generating harmful updates by modifying the original gradients

[17, 19]. Although these methods can severely impair the global model performance, the malicious gradients often deviate significantly from the benign ones, making them detectable by defense mechanisms.

To improve stealth and enhance the efficacy of these attacks, more advanced strategies have been proposed. For instance, some methods craft adversarial gradients by altering the statistical properties of original gradients [4], whereas others, such as [112], focus on selectively perturbing only a small subset of the local model parameters to avoid detection. Additionally, Fang et al. [20] propose an adaptive attack that estimates the optimal global model update and constructs adversarial gradients to neutralize it, which effectively circumvents a wide array of defense mechanisms.

- **Data Poisoning Attacks**

  Data poisoning attacks pose a substantial threat to the integrity and reliability of FL systems. In these attacks, adversaries compromise the training data on a subset of participating clients with the objective of degrading the performance of the resulting global model. These attacks are typically classified into two categories: *targeted* attacks, which seek to influence the model's behavior on specific classes or inputs while maintaining general performance degradation, and *non-targeted* attacks, which aim to broadly reduce overall model accuracy.

  A prominent example is the label flipping attack [81], wherein the adversary deliberately mislabels training samples to induce the generation of harmful local updates. Lewis et al. [36] advance this concept by introducing a dynamic poisoning strategy in which the adversary alternates between benign and malicious behavior, thereby improving stealth and persistence. Poisoned data may be injected directly onto selected client devices or indirectly via compromised intermediaries in the communication pipeline [75]. Beyond conventional classification tasks, data poisoning has also been extended to more complex domains such as face recognition [64]. Additionally, Gupta et al. [25] propose a novel approach that involves inverting the loss function to generate gradients that move away from the optimization minima, effectively producing adversarial labels that severely impair model convergence and accuracy.

- **Backdoor Attacks**

  A significant vulnerability inherent to FL systems is their exposure to backdoor attacks. These attacks involve the insertion of covert functionalities into either individual client models or the globally aggregated model. Specifically, the compromised model is designed to behave normally under typical conditions but generates erroneous outputs when a predefined trigger is present [22]. The decentralized architecture of FL exacerbates this threat, as adversaries can amplify the influence of malicious updates with backdoor payloads by simply scaling them to outweigh contributions from benign clients.

Several studies have proposed methodologies for injecting backdoors into FL systems. Nguyen et al. [62] successfully embedded a backdoor into an FL-based intrusion detection system for Internet of Things (IoT) environments, targeting traffic patterns characteristic of specific malware. Bagdasaryan et al. [3] demonstrate that physical artifacts, such as sunglasses, tattoos, and earrings, can act as effective triggers for initiating backdoor behaviors in FL systems. Similarly, Sun et al. [77] propose techniques for embedding covert behaviors within FL models. Xie et al. [93] extend this concept by introducing distributed backdoor attacks, wherein multiple adversarial clients collaborate to implant triggers into the global model. Furthermore, Zhang et al. [109] investigate methods for increasing backdoor persistence by selectively manipulating specific model parameters. More recently, Nguyen et al. [63] introduce a stealthy backdoor approach engineered to evade both manual inspection and automated defense mechanisms.

- **Communication Attacks**

  FL operates through iterative communication between a central server and a network of distributed clients to collaboratively update a global model. However, the substantial volume and high frequency of data exchange inherent in this process give rise to critical communication bottlenecks and expand the system's attack surface. Man-in-the-Middle (MiTM) attacks [1] have been identified as a prominent threat at the network layer: By intercepting communications between the server and clients, adversaries can compromise data integrity, disrupt the training process, and exploit the central server as a single point of failure. Furthermore, Yao et al. [99] investigate targeted adversarial strategies aimed at FL communication channels, including bandwidth throttling, induced latency, and transmission instability. Such disruptions have been demonstrated to impede model convergence significantly and deteriorate overall system performance.

1.2.2 Defense Against Model Utility Attack

- **Data Sanitization**

The distributed and decentralized nature of FL inherently complicates the task of verifying the trustworthiness of individual clients, making the detection of data poisoning both complex and resource-intensive. To mitigate these vulnerabilities, one class of defense strategies focuses on sanitizing data prior to the FL optimization process. For instance, Li et al. [39] propose a method for filtering out malicious or suspicious data before training commences. Tian et al. [80] develope a defense mechanism that identifies and suppresses anomalous data points, thereby diminishing the influence of poisoning attacks. Li et al. [37] further enhance this approach by jointly optimizing a data filtering mechanism alongside the global model. In addition, Cui et al. [16] scale anomaly detection techniques to accommodate the demands of large-scale Internet of Things (IoT) infrastructures within FL settings.

- **Anomalous Client Filtering**

  While data sanitization serves as an essential first line of defense, it may be insufficient in the presence of actively malicious clients within FL systems. To address this challenge, various approaches have been proposed to detect and mitigate harmful client behavior. Li et al. [45] introduce a method that identifies and filters malicious clients by analyzing communication-related information. Yazdinejad et al. [100] propose an Autoencoder-based technique to detect abnormal model weight updates originating from compromised clients. Meng et al. [56] develope a visualization-assisted anomaly detection framework that facilitates the investigation of client behaviors and the assessment of anomaly severity. Qi et al. [69] propose a blockchain-integrated FL framework, wherein client models undergo verification prior to being stored on a consortium blockchain. This method effectively reduces the threat of poisoned updates. Furthermore, Nguyen et al. [61] introduce FLAME, a comprehensive defense framework that detects and eliminates poisoned model updates through a hybrid approach combining model filtering and poison mitigation techniques.

- **Adversarial Training**

  Adversarial training involves the intentional introduction of small perturbations during the model training process to enhance the model's robustness against adversarial attacks [82]. Li et al. [38] employe adversarial training within FL framework to mitigate model drift and accelerate convergence. Hong et al. [27] propose an innovative learning paradigm that enables the transfer of adversarial robustness from high-resource clients to low-resource clients, thereby improving overall system resilience. Additionally, Chen et al. [10]

apply adversarial training as a defense mechanism against evasion attacks, demonstrating improved certifiable robustness in FL settings.

1.2.3 Client Privacy Attack

In Federated Learning (FL), client data privacy is generally maintained by ensuring that raw data remains localized on individual devices. However, recent research has demonstrated that adversaries can still infer sensitive information about local datasets, or even reconstruct the original training data, through analysis of shared model updates [44].

- **Inversion Attacks**

  Although FL is intended to enhance data privacy by transmitting model gradients rather than raw data, this mechanism does not ensure complete protection against information leakage. A growing body of research has revealed that private training data can be reconstructed from shared model updates, thereby exposing a critical vulnerability in FL systems. For instance, Huang et al. [29] conduct a thorough evaluation of gradient inversion attacks, illustrating how gradient information can be exploited to infer sensitive attributes of the underlying data. Similarly, Yin et al. [102] demonstrate that original training data could be reconstructed by utilizing model weights in conjunction with normalization statistics. In another study, Jeon et al. [30] propose a technique that recovers client data by optimizing latent representations to align with observed gradients. More recently, Hatamizadeh et al. [26] introduce a method capable of inverting entire training batches by leveraging gradient vectors, thereby underscoring the privacy risks associated with disclosing model updates.

- **Inference Attacks**

  During the FL training process, the global model may inadvertently capture and encode latent information derived from clients' private data. Consequently, external adversaries can exploit this leakage through inference attacks, aiming to extract sensitive information from the training data. Inference attacks in FL are generally categorized into *Membership Inference Attacks (MIA)* and *Property Inference Attacks (PIA)*, which are introduced below.

  Membership Inference Attacks (MIA) aim to determine whether a specific data sample was part of a client's training dataset. Nasr et al. [59] introduce a gradient ascent-based MIA that amplifies the influence of target data points within others' training sets, thereby increasing inference accuracy. Zhu et al. [114] propose a multi-phase attack strategy that leverages updates from all active clients to enhance the effectiveness of membership inference. Suri et al. [78] focus on black-box subject-level MIA, aligning more closely with practical adversarial objectives in real-world scenarios.

Property Inference Attacks (PIA) attempt to infer latent attributes or properties of the training data without requiring access to the raw data. Wang et al. [87] propose a poisoning-assisted PIA method that exploits periodic patterns in model updates to detect variations in sensitive properties across data distributions. Liu et al. [48] introduce the first PIA framework specifically targeting Federated Graph Neural Networks (GNNs), expanding the threat landscape of inference attacks. Kim et al. [34] highlight the increased vulnerability of Clustered FL to PIA compared to conventional FL and proposed an active inference technique incorporating a scaling mechanism to amplify attack effectiveness.

1.2.4 Defense Against Client Privacy Attack

- **Secure Multi-party Computing (SMC)**

  Secure Multi-Party Computation (SMC), originally introduced through Yao's seminal Millionaire's Problem [98], provides a cryptographic framework that allows multiple parties to jointly compute a function while preserving the confidentiality of their individual inputs. This paradigm has been effectively integrated into FL to enhance the protection of sensitive client data during collaborative model training. Notably, Bogdanov et al. [5] and Bonawitz et al. [6] develope privacy-preserving FL architectures that utilize SMC to securely aggregate model updates from distributed clients. More recently, Xu et al. [95] propose a verifiable private gradient aggregation scheme based on random matrix coding, which further enhances the integrity and trustworthiness of the aggregation process.

  A key advantage of employing SMC in FL lies in the relative size of the data involved: the number of transmitted model parameters is typically several orders of magnitude smaller than the size of clients' local datasets. This enables a practical balance between privacy protection and system efficiency, offering strong privacy guarantees without imposing excessive computational or communication overhead.

- **Differential Privacy (DP)**

  While robust encryption techniques can effectively prevent the direct parsing of individual data points, they do not eliminate the risk of inference attacks aimed at uncovering aggregate characteristics of user groups. Differential Privacy (DP) offers a mathematically rigorous framework to mitigate this threat by injecting carefully calibrated random noise into the data or model updates [18]. This process obscures the true values, making it difficult for adversaries to reconstruct sensitive information, even with auxiliary background knowledge. In the context of FL, DP is typically implemented by adding noise to the transmitted model parameters, thereby preserving privacy without imposing significant communication or computational overhead [88]. One of the key strengths of DP is its formal, quantifiable privacy guarantees, which are provably resistant to a broad spectrum of attacks, regardless of the adversary's prior knowledge.

Extensive research has investigated the integration of DP within FL frameworks. For instance, Geyer et al. [21] propose a DP mechanism specifically adapted for FL, achieving an effective balance between privacy protection and model utility. Zhao et al. [110] apply Local Differential Privacy (LDP) to protect user privacy in IoT crowdsourcing applications. Besides, Triastcyn et al. [83] apply Bayesian DP to achieve tighter privacy guarantees and enhanced model accuracy. Additionally, Huang et al. [28] introduce a DP-based approach aimed at mitigating performance degradation associated with unbalanced client data distributions.

- **Homomorphic Encryption (HE)**

  Homomorphic Encryption (HE) is a cryptographic technique that enables the execution of algebraic operations directly on encrypted data, eliminating the need for prior decryption. Importantly, the decrypted result of these operations is mathematically equivalent to performing the same computations on the original plaintext. In the context of FL, HE facilitates the secure aggregation of encrypted model parameters by a central server, thereby preserving the confidentiality of client-side data throughout the training process.

  Extensive research has explored the integration of HE into FL frameworks. For instance, Zhang et al. [108] introduce a privacy-preserving and verifiable FL scheme based on HE. Similarly, [54] proposed a multi-key HE protocol, wherein model updates are encrypted using aggregated public keys, necessitating collective client participation for decryption and thereby enhancing security assurances. Likewise, Park et al. [65] proposed a HE-based FL scheme that supports encrypted parameter aggregation by the server, while allowing each client to retain a distinct HE private key within a unified distributed cryptosystem, further strengthening privacy protections. Zhang et al. [104] propose an optimization strategy To address the computational overhead often associated with HE.

1.2.5 Our Contribution

The present work addresses the inherent model utility and client privacy challenges in FL through four distinct approaches. *Work 1* introduces an auxiliary generative network optimized and shared to perform data augmentation in the embedding space, thereby reducing the risk of sensitive information leakage compared to augmentation in the raw input space. *Work 2* leverages a frozen pre-trained vision-language model, optimizing and transmitting only a lightweight add-on module. To further enhance privacy, only a subset of the module's parameters is communicated between clients and server, while the rest remain strictly local, thus mitigating inversion and inference risks. *Work 3* employs evolutionary algorithms to eliminate poorly performing client models, which helps preserve global model performance by reducing the influence of ineffective or potentially malicious participants. *Work 4* enforces a stronger privacy-preserving paradigm by avoiding any transmission of client-specific image data; instead, clients send low-dimensional latent vectors, further anonymized via interpolation and the injection of significant random noise.

1.3 Data Heterogeneity and Scarcity

In real-world Federated Learning (FL) scenarios, participating clients are often geographically dispersed and experience varying levels of user activity, leading to inconsistencies in both the volume and distribution of the data they collect. As a result, the data quantity and data distribution across clients are typically heterogeneous in practical FL applications. When facing such heterogeneity, the traditional approach of directly averaging client updates can cause model drift, ultimately impairing the convergence and accuracy of the global model [66]. Therefore, it is essential to design effective FL methods that are tailored to the specific data distribution patterns and application environments encountered in practice.

We first provide the formal description to characterize data heterogeneity in FL: Let $\mathcal{X} \subset \mathbb{R}^{d_i}$ denote the input space, $\mathcal{U} \subset \mathbb{R}^{d_u}$ the feature space, and $\mathcal{Y} \subset \mathbb{N}$ the output space. Let $\boldsymbol{\theta} := [\boldsymbol{\theta}_f, \boldsymbol{\theta}_h]$ represent the parameters of a classification model trained in an FL setting involving a central server and $K \in \mathbb{N}$ clients. The model is composed of two components: a feature extractor $f : \mathcal{X} \to \mathcal{U}$ parameterized by $\boldsymbol{\theta}_f$, and a prediction head $h : \mathcal{U} \to \mathcal{Y}$ parameterized by $\boldsymbol{\theta}_h$. We assume each client holds a private dataset $D^k = \{(\boldsymbol{x}_i^k, y_i^k) \mid i \in 1, \ldots, N_k\}$, where $N^k \in \mathbb{N}$ is the number of local samples and $C \in \mathbb{N}$ is the number of classes. As discussed in [32], data heterogeneity in FL can be modeled through distribution shifts across local datasets, formalized as $P^{k_1}_{\mathcal{XY}} \neq P^{k_2}_{\mathcal{XY}}$ for all $k_1, k_2 \in \{1, \dots, K\}, k_1 \neq k_2$, where $P^{k}_{\mathcal{XY}}$ defines the joint distribution of $\mathcal{X}$ and $\mathcal{Y}$ on client k. In parallel, data scarcity can be represented as $N_{k_1} \neq N_{k_2}$ for all $k_1, k_2 \in \{1, \dots, K\}, k_1 \neq k_2$. In the following, we introduce different data challenges for FL systems:

1.3.1 Feature Distribution Skew

Feature distribution skew, also known as covariate shift, arises in FL when clients exhibit differing distributions of input features ($P_{\mathcal{X}}$), despite sharing a similar underlying relationship between features and labels ($P_{y|\mathcal{X}}$). For instance, in a federated medical imaging scenario, participating institutions may utilize MRI scanners from different manufacturers or follow varying imaging protocols. These differences lead to substantial variability in image characteristics, such as brightness, contrast, and resolution, even when depicting the same underlying medical condition [101]. A similar phenomenon is observed in the FEMNIST dataset, where each user exhibits a distinct handwriting style, resulting in diverse feature representations for identical characters [8]. This type of feature distribution skew poses a significant challenge, as models trained on the feature distribution of a single client may generalize poorly to data from other clients with dissimilar distributions. Overcoming this limitation necessitates federated learning algorithms that can either learn representations robust to such distributional shifts or adapt effectively to client-specific feature variations.

Several approaches have been proposed to mitigate feature distribution skew in federated learning. For instance, Zhou et al. [111] introduce a feature augmentation framework that manipulates local feature statistics using global information, thereby improving model robustness to distributional shifts. Yan et al. [96] propose a data augmentation strategy that integrates global statistical features into local client data, enhancing the generalization capability of learned representations and reducing inter-client feature divergence. Mou et al. [58] employ a variational inference framework by incorporating a Kullback–Leibler (KL) divergence regularization term into the training objective, which constrains the output space of feature extractors while enabling personalized adaptation in the final model layers. Sun et al. [74] propose partial model initialization, a method in which only shared parameters are synchronized across clients, while client-specific components are updated independently to better align with local data characteristics. Finally, Li et al. [41] address feature distribution skew by localizing batch normalization (BN) layers to each client, allowing the model to adapt to domain-specific statistics, while the remaining parameters are aggregated globally.

1.3.2 Label Distribution skew

Label distribution skew, also referred to as prior probability shift, is a prevalent challenge in FL that arises when the marginal label distributions ($P(y)$) vary substantially across clients, even if the conditional feature distributions given the label ($P(\mathcal{X}|y)$) remain relatively consistent. This type of statistical heterogeneity is commonly observed in federated classification tasks [42]. For example, in federated image recognition, some clients may predominantly hold samples from a limited subset of classes—such as digits '1' and '2'—while others contain mostly '8' and '9' [113]. A similar situation occurs in federated social bot detection, where different platforms may naturally exhibit divergent ratios of bot to human user accounts [85]. The presence of label distribution skew can introduce significant bias into the global model, particularly favoring the label distributions of clients with disproportionately large datasets or more frequent participation in training. This bias often leads to suboptimal performance on globally rare or underrepresented classes, highlighting the necessity of developing effective strategies to mitigate such skew.

Several methods have been proposed to address label distribution skew in federated learning. FedProx [40] introduces a proximal regularization term to penalize large deviations from the global model during local updates, thereby enhancing training stability across heterogeneous client distributions. Zhang et al. [105] propose a logit calibration technique that adjusts model predictions based on class occurrence probabilities, reducing the tendency to overfit to majority classes. Zhu et al. [115] mitigate label shift by generating synthetic feature representations that approximate the global data distribution and applying data-free knowledge distillation to align local models. Lutz et al. [53] present a client selection strategy that maximizes the entropy of the global label distribution per communication round, thereby encouraging label diversity and improving generalization. Sheng et al. [73] employ a global Generative Adversarial Network (GAN) to model the overall data distribution, enabling global knowledge distillation without requiring access to local data. Wang et al. [86] propose using private weak learners on the client side to form ensembles with local models, effectively correcting optimization bias and improving performance on underrepresented classes.

There also exist several works that simultaneously address heterogeneity in both the feature and label spaces. Guo et al. [24] introduce a clustering-based framework that groups clients according to data distribution similarity and utilizes bi-level optimization to manage multiple types of distribution shifts. Zhou et al. [111] propose feature anchors to align features and calibrate classifiers simultaneously, enabling consistent model updates and improving model performance under different data heterogeneity. Similarly, Chen et al. [13] propose the use of feature anchors to calibrate classifiers, thereby aligning both feature and label distributions across clients and enhancing robustness to data heterogeneity in FL.

1.3.3 Data Quantity Skew

Data quantity skew refers to the heterogeneity in the amount of local training data held by clients in FL systems. This issue commonly arises in real-world applications such as mobile or IoT environments, where certain users generate substantial volumes of data, while others contribute only sparsely due to lower activity levels or available resources [60]. Under standard aggregation schemes such as Federated Averaging (FedAvg) [55], where client updates are typically weighted by the size of local datasets, clients with larger data volumes tend to exert a disproportionate influence on the global model. This can result in a biased global model that predominantly reflects the data distribution of a small subset of data-rich clients, while undervaluing the contributions of clients with smaller datasets. Such imbalance may lead to the marginalization of rare or minority patterns that are critical for fairness and generalization [103].

Recent works have proposed strategies to mitigate this issue. Zhang et al. [107] provides a systematic analysis on the impact of data imbalance in FL for credit risk forecasting. Chung et al. [15] introduce FedISM, a method that improves training efficiency by initially learning a shared global model from a candidate dataset before proceeding with client-specific model updates. Similarly, Qi et al. [68] propose FedSampling, a data-aware client selection strategy that prioritizes clients based on their data volume rather than selecting them uniformly at random, thereby enhancing model performance under conditions of data quantity skew.

1.3.4 Concept Drift

Concept drift refers to changes in the underlying data-generating process or in the statistical relationship between features and labels, formally characterized as shifts in the conditional distribution $P_{y|\mathcal{X}}$ across different client contexts in FL. For instance, in natural language processing tasks, the semantics or usage of specific terms may vary by geographic region, resulting in heterogeneous feature-label relationships among clients [49]. Concept drift can also arise when identical feature inputs are mapped to different labels, as observed in subjectively annotated sentiment analysis tasks [70]. Conversely, it may occur when disparate feature sets correspond to the same label, such as varying symptom manifestations of a disease across different demographic groups [51]. In the presence of concept drift, a static global model trained on a specific subset of clients or data collected at a fixed point in time may fail to generalize to clients experiencing altered or evolving concepts.

Several studies have explored methods to address concept drift in FL. Casado et al. [9] extend the standard FedAvg algorithm by introducing concept adaptation mechanisms to improve performance in non-stationary environments. Chen et al. [14] propose an asynchronous framework for local concept drift detection, wherein client update strategies are dynamically adjusted based on historical model performance. Jothimurugesan et al. [31] address staggered concept drift by clustering clients according to local drift patterns, thereby improving generalization in the presence of heterogeneous concept shifts.

1.3.5 Our Contribution

The present work addresses challenges related to data heterogeneity and data quantity in FL from multiple perspectives. Work 1 tackles feature space distribution shifts by augmenting the training embedding space through the synthesis of feature embeddings that preserve global knowledge. These synthesized embeddings are then projected back into each client's personalized local space. This approach demonstrates robustness even in scenarios with highly imbalanced client dataset sizes. Work 2 addresses multi-modal (visual and textual) feature distribution shifts by introducing a dual-stream adapter module. This module disentangles and captures both client-specific and client-agnostic knowledge, subsequently distilling the combined knowledge into a target adapter module. Work 3 mitigates issues related to imbalanced data quantity by leveraging evolutionary algorithms, wherein poorly performing client local models are iteratively eliminated and replaced by superior ones. In addition, the application of multi-process training and interleaved elimination steps on the server side helps reduce validation score noise introduced by skewed label distributions. Work 4 addresses feature space heterogeneity by optimizing client-specific soft tokens in the input prompts that characterize local feature distributions, thereby enabling fine-grained and precise personalization in image generation tasks.

1.4 Communication and Computation

In Federated Learning (FL), a large number of distributed clients participate in training and need to frequently communicate model parameters with a server. Usually, the global model requires hundreds or thousands of server-client communication to converge, this makes communication overhead one of the main bottlenecks in FL. Meanwhile, in cross-device setting, the clients usually have limited computing power and energy efficiency. How to reduce the computational burden and improve training efficiency are also key issues.

1.4.1 Server-Client Communication Overhead

Several approaches have been proposed to enhance communication efficiency in federated learning while preserving model performance. Gradient compression is a widely adopted strategy to reduce communication overhead. Lin et al. [46] propose Deep Gradient Compression (DGC) for local gradient clipping and momentum correction and masking. DGC achieves compression ratios up to 600x without accuracy loss, enabling efficient training over low-bandwidth networks and mobile environments. Besides, Aji et al. [2] accelerate gradient communication by exchanging only quantized and sparse gradient updates. Wen et al. [90] improve communication efficiency by transmitting quantized gradients restricted to ternary values {-1,0,1}. Konevcny et al. [35] propose structured updates and sketched updates, both of which aim to reduce the number of transmitted parameters and minimize uplink communication costs. Other works address the challenge from a system perspective: Chen et al. [11] present a communication-efficient FL framework, incorporating probabilistic device selection, parameter quantization, and optimized resource allocation, achieving improvements in both accuracy and communication reduction. Additionally, Chen et al. [12] propose a framework with algorithms for resource allocation and user selection to mitigate issues such as packet errors and limited bandwidth in wireless environments.

1.4.2 Client Computational Burden

To reduce the computational burden at different clients, some methods address the challenges from the optimization perspective. Sattler et al. [72] propose Sparse Ternary Compression (STC), which uses Top-k gradient sparsification, ternary quantization, and efficient encoding to compress both upstream and downstream communication. Luo et al. [52] propose an algorithm to optimally selects the number of clients and local iterations to minimize total cost while ensuring convergence. Wu et al. [91] proposed FedKD, where only probability distributions or embeddings are transmitted between the clients and the server rather than the full model parameters, thereby significantly reducing the amount of data transmitted. Sun et al. [76] adapt and improve Low-Rank Adaptation (LoRA) techniques to enable efficient optimization at client-side.

There are some methods focusing on investigating the tradeoffs between the communication frequency and client local computational burdens. Mcmahan et al. [55] show that local large-batch training, combined with strategies like learning rate decay, has been proven to maintain model accuracy while reducing communication. Besides, Reddi et al. [71] observe that momentum or learning rate decay at server-side helps the global model convergence. From the communication frequency and syncronity, in traditional FL, the server will wait all clients until one round of optimization has been fisnished, which makes slow clients becomes the bottleneck. Therefore, Xie et al. [94] propose FedAsync, which allows clients to upload updates asynchronously, and the server uses a weighted average to aggregate potentially delayed gradients. Also, Yang et al. [97] uses reinforcement learning for client scheduling, selecting a subset of clients based on their contribution to the global model and latency per round, thereby simultaneously optimizing both communication and statistical efficiency.

1.4.3 Our Contribution

In this work, we address communication and computation challenges from multiple angles: Work 1 optimizes only an additional feature embedding generator—significantly more lightweight than full image generation—reducing both communication and computation overhead. Work 2 adapts Parameter-Efficient Fine-Tuning (PEFT) methods to the FL setting, updating only a small add-on module instead of the full large-scale pretrained model. Work 3 introduces multi-process optimization to enhance overall system efficiency. Work 4 adopts a One-Shot FL approach, requiring only a single round of client-server communication with minimal bandwidth use.

Chapter 2

FRAug: Tackling Federated Learning with Non-IID Features via Representation Augmentation

This chapter contains the publication

> **Haokun Chen**, Ahmed Frikha, Denis Krompass, Jindong Gu, Volker Tresp. FRAug: Tackling Federated Learning with Non-IID Features via Representation Augmentation. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 4849-4859

FRAug: Tackling Federated Learning with Non-IID Features via Representation Augmentation

Haokun Chen [1,2] Ahmed Frikha [1,2,3] Denis Krompass [2] Jindong Gu [4*] Volker Tresp [1,3]
[1] Ludwig Maximilian University of Munich [2] Siemens Technology
[3] Munich Center for Machine Learning [4] University of Oxford
{haokun.chen, ahmed.frikha, denis.krompass}@siemens.com,
jindong.gu@outlook.com, volker.tresp@lmu.de

Abstract

Federated Learning (FL) is a decentralized machine learning paradigm, in which multiple clients collaboratively train neural networks without centralizing their local data, and hence preserve data privacy. However, real-world FL applications usually encounter challenges arising from distribution shifts across the local datasets of individual clients. These shifts may drift the global model aggregation or result in convergence to deflected local optimum. While existing efforts have addressed distribution shifts in the label space, an equally important challenge remains relatively unexplored. This challenge involves situations where the local data of different clients indicate identical label distributions but exhibit divergent feature distributions. This issue can significantly impact the global model performance in the FL framework. In this work, we propose Federated Representation Augmentation (FRAug) to resolve this practical and challenging problem. FRAug optimizes a shared embedding generator to capture client consensus. Its output synthetic embeddings are transformed into client-specific by a locally optimized RTNet to augment the training space of each client. Our empirical evaluation on three public benchmarks and a real-world medical dataset demonstrates the effectiveness of the proposed method, which substantially outperforms the current state-of-the-art FL methods for feature distribution shifts, including PartialFed and FedBN.

1. Introduction

Federated Learning (FL) is a machine learning paradigm in which a shared model is collaboratively trained using decentralized data sources. In the classical FL approach, *e.g.*, FedAvg [49], the central server obtains the model by iteratively averaging the optimized model weights uploaded from the active clients. FL has the benefit that it does not require direct access to the client local datasets, resulting in improved client-server communication efficiency and enhanced data confidentiality.

Despite these promising prospects, real-world FL applications encounter practical challenges arising from data heterogeneity, in which the client local datasets are not independent and identically distributed (*non-IID*). Non-IID data from different clients may cause local model drifts during the client update and overfitting to its local objective, making it challenging to obtain a stable and optimal convergence of the aggregated server model [41, 50].

As discussed in [28], data heterogeneity in FL can be categorized into label space heterogeneity and feature space heterogeneity. A variety of methods were developed to tackle problem settings where the client datasets are non-IID in the label space [75, 66]. However, the under-explored problem of feature distribution shift is also prevalent in real-world applications, *e.g.*, in the data collected from different scanners in clinical centers [10], as well as gathered by different machines in industrial manufacturing plants [39]. Most importantly, although these entities may diagnose the same types of cancers or detect the same types of anomalies, *i.e.*, having the same label distribution, they are not willing to share their original data to prevent competitive disadvantage or reverse engineering. Therefore, we propose an effective and privacy-preserving FL algorithm, *i.e.*, Federated Representation Augmentation (*FRAug*), to address this practical problem of feature space heterogeneity.

Unlike previous works that generate synthetic samples in the input space [69, 68] or acquire additional public datasets [44, 17], FRAug applies data augmentation in the low-dimensional feature embedding space, which is more efficient and confronts fewer confidentiality threats. Moreover, the proposed augmentation algorithm is especially suitable for FL applications, where collaborative training is often conducted by multiple edge devices (clients) with limited

[*] Corresponding author

computational powers and data quantities [49]. Specifically, we first aggregate the consensual knowledge from different clients in the embedding space by training a shared representation generator, which produces client-agnostic embeddings. However, solely optimizing the generator might be challenging, given its training representations following different local client feature distributions. Therefore, a Representation Transformation Network (RTNet) is locally trained at each client to transform the client-agnostic synthetic embeddings into client-specific. Hereby, we aim at aligning the client-agnostic embeddings with the local feature distribution. Finally, the local dataset of each client will be augmented by its client-specific synthetic embeddings.

The proposed method FRAug achieves state-of-the-art results on three benchmark datasets with feature distribution shift, surpassing the concurrent FL methods addressing the same problem, including PartialFed [55] and FedBN [43]. Moreover, the superior performance of FRAug on a medical dataset illustrates its applicability in complex real-world FL applications. Our contributions can be summarized as follows:

- We propose a novel representation augmentation algorithm (*FRAug*) to address FL with non-IID features.
- We conduct comprehensive experiments on three public benchmark datasets with feature distribution shifts, in which FRAug achieves SOTA results.
- We verify the maturity and scalability of FRAug on a real-world medical dataset, and further analyze the convergence rate and robustness of FRAug.

2. Related Work

2.1. Federated Learning (FL)

Federated Averaging (FedAvg) [49] is one of the classic FL algorithms for training machine learning models using decentralized data sources. This simple paradigm suffers from performance degradation when there exists data heterogeneity [28, 41]. Numerous studies have been conducted for label space heterogeneity, *i.e*., class distributions are imbalanced across different clients, by adding additional regularization term in the client local update [42, 8, 53, 35, 26, 4, 31, 65], utilizing shared local data [70, 45, 16], introducing additional public datasets [37, 44, 17], fully or partially personalizing the client models [3, 12, 56, 40, 7, 52, 1], or performing data-free knowledge distillation [46] in the input space [20, 69, 68] or the feature space [21, 76, 47]. However, there are only limited studies addressing the heterogeneity in feature space, *i.e*., non-IID features. Recently, [2] showed that Batch Normalization layers (BN) [24] with local statistics improve the robustness of the FL model to inter-center data variability and yield better out-of-domain generalization results, while FedBN [43] provided more theoretical analysis on the benefits of local BN layers for FL with feature non-IID. PartialFed [55] empirically found that partially initializing the client models could alleviate the effect of feature distribution shift. HarmoFL [27] focused on FL applications for heterogeneous medical images and applied amplitude normalization in frequency space and model weight perturbation to harmonize the training process. In this work, we tackle the problem of non-IID features in FL via a client-specific data augmentation approach performed in the embedding space. In particular, client-agnostic embeddings are initially synthesized by a shared generator that captures the knowledge from different distributions, which are then personalized by separate client-specific models. Training the local model with the resulting client-specific embeddings improves its robustness against the feature distribution shift.

2.2. Cross-Domain Learning

The problem of learning on centralized data with non-IID features, *i.e*., cross-domain data, has been widely studied in the context of Unsupervised Domain Adaptation (UDA) [60, 5, 67, 6, 30, 62], where a model is trained using multiple source domains and finetuned using an unlabelled target domain, and Domain Generalization (DG) [71, 13, 72, 14, 36, 29], where the target domain data is not accessible during the training process of UDA. A variety of efforts have been made to tackle the problem of UDA and DG. CROSSGRAD [54] used adversarial gradients obtained from a domain classifier to augment the training data. L2A-OT [73] trained a generative model to transfer the training samples into pseudo-novel domains. MixStyle [74] performed feature-level augmentation by interpolating the style statistics of the output features from different network layers. While the aforementioned methods assume centralized access to all datasets from different domains, we address the problem where the datasets are decentralized and cannot be shared due to privacy concerns.

3. Methodology

3.1. Problem Statement

In this work, we address an FL problem setting with non-IID features, which we describe in the following. Let $\mathcal{X} \subset \mathbb{R}^{d_{in}}$ be an input space, $\mathcal{U} \subset \mathbb{R}^{d_u}$ be a feature space, and $\mathcal{Y} \subset \mathbb{N}$ be an output space. Let $\boldsymbol{\theta} := [\boldsymbol{\theta}_f, \boldsymbol{\theta}_h]$ denote the parameters of the classification model trained in an FL setting involving one central server and $K \in \mathbb{N}$ clients. The model consists of two components: a feature extractor $f : \mathcal{X} \rightarrow \mathcal{U}$ parameterized by $\boldsymbol{\theta}_f$, and a prediction head $h : \mathcal{U} \rightarrow \mathcal{Y}$ parameterized by $\boldsymbol{\theta}_h$. We assume that a dataset $D^k = \{(\boldsymbol{x}_i^k, y_i^k) | i \in \{1, .., N_k\}\}$, containing private data, is available on each client, where $N^k \in \mathbb{N}$ denotes

Method	OfficeHome				
	Art	Clipart	Product	Real	avg
w/o Add. Embeddings	57.47	56.74	73.32	71.25	64.69
w. Add. Embeddings	68.18	72.31	80.04	79.50	**75.01**

Table 1: Evaluation accuracies of models optimized with (w.) and without (w/o) prediction head finetuned using additional embeddings on OfficeHome benchmark, indicating the applicability of the representation generator given the performance increase.

the number of samples in D^k and $C \in \mathbb{N}$ denotes the number of classes. As discussed in [28], FL with non-IID data can be described by the distribution shift on local datasets: $P_{\mathcal{XY}}^{k_1} \neq P_{\mathcal{XY}}^{k_2}$ with $\forall k_1, k_2 \in \{1, ..., K\}, k_1 \neq k_2$, where $P_{\mathcal{XY}}^{k}$ defines the joint distribution of input space $\mathcal{X}$ and label space $\mathcal{Y}$ on D^k. The addressed problem setting, *i.e.*, FL with non-IID features, covers (1) *covariate shift*: The marginal distribution $P_{\mathcal{X}}$ varies across clients, while $P_{\mathcal{Y}|\mathcal{X}}$ is the same, and (2) *concept shift*: The conditional distribution $P_{\mathcal{X}|\mathcal{Y}}$ varies across clients, while $P_{\mathcal{Y}}$ is the same [43]. From the perspective of cross-domain learning literature [60, 71], local data from every client can be viewed as a separate domain.

3.2. Motivational Case Study

To motivate our representation augmentation algorithm, we present an empirical analysis to address the following research question: *Does finetuning only the prediction head using additional synthetic feature embeddings lead to performance improvement?* First, we optimize a classification model $\boldsymbol{\theta}^k$ with 10% of the local dataset D^k following prior FL work [49, 43]. Then, we fix the feature extractor and finetune *only* the prediction head with 100% of D^k. Finally, we evaluate both classification models. Here, we use the representations, extracted by the feature extractor using the additional real images, to simulate the output produced by a ”perfect” embedding generator.

The results in Tab. 1 show that the feature extractor, trained with less data, still captures useful information when exposed to unseen image samples. Most importantly, a substantial average performance boost of 10.32% shows that generating additional representations benefits the client local update, proving the applicability and effectiveness of the proposed method.

3.3. Proposed Method

To tackle FL with non-IID features, we propose Federated Representation Augmentation (FRAug). Our algorithm is built upon FedAvg [49], which is the most widely used FL strategy. In FedAvg, the central server sends a copy of the global model $\boldsymbol{\theta}$ to each client to initialize their local models $\{\boldsymbol{\theta}^k | k \in K\}$. After training on its local dataset D^k, the client-specific updated models are sent back to the central server, where they are averaged and used as the global model. Such communication rounds are repeated until some predefined convergence criteria are met. Similarly, the training process of FRAug (Algorithm 1) can be divided into two stages: (1) The *Server Update*, where the central server aggregates the parameters uploaded by the clients and distributes the averaged parameters to each client, and (2) the *Client Update*, where each client receives the model parameters from the central server and performs local optimization. Unlike FedAvg, where only the local dataset of each client is used for training, FRAug generates additional feature embeddings to finetune the prediction head of the local classification model. Concretely, we train a shared generator and a local Representation Transformation Network (RTNet) for each client, which together produce *domain-specific* synthetic feature embeddings for each client to augment its local data in the embedding space. Hereby, the shared generator captures knowledge from all the clients to generate client-agnostic embeddings, which are then personalized by the local RTNet into client-specific embeddings. In the following, we provide a more detailed explanation of FRAug.

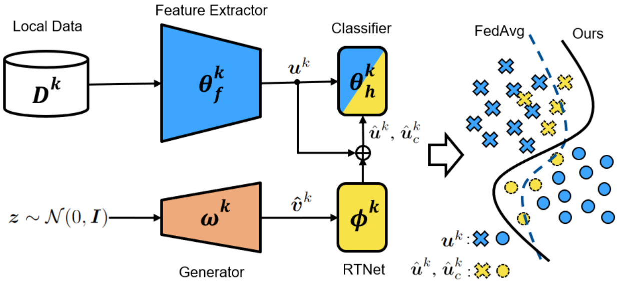

Local Data
$\boldsymbol{D}^{\boldsymbol{k}}$
Feature Extractor
$\boldsymbol{\theta}_{\boldsymbol{f}}^{\boldsymbol{k}}$
$\boldsymbol{u}^k$
Classifier
$\boldsymbol{\theta}_{\boldsymbol{h}}^{\boldsymbol{k}}$
$\hat{\boldsymbol{u}}^k$, $\hat{\boldsymbol{u}}_c^k$
$\boldsymbol{z} \sim \mathcal{N}(0, \boldsymbol{I})$
$\boldsymbol{\omega}^{\boldsymbol{k}}$
Generator
$\hat{\boldsymbol{v}}^k$
$\boldsymbol{\phi}^{\boldsymbol{k}}$
RTNet
FedAvg
Ours
$\boldsymbol{u}^k$:
$\hat{\boldsymbol{u}}^k$, $\hat{\boldsymbol{u}}_c^k$:

Figure 1: Overview of FRAug local update at client k: a shared generator is learned to aggregate knowledge from multiple clients and generate client-agnostic feature embeddings $\hat{\boldsymbol{v}}^k$, which are then fed into the local Representation Transformation Network (*RTNet*) to produce client-specific feature embeddings $\hat{\boldsymbol{u}}^k$ and $\hat{\boldsymbol{u}}_c^k$. Finally, the real feature embeddings $\boldsymbol{u}^k$, extracted by the feature extractor using local dataset D^k, will be augmented with $\hat{\boldsymbol{u}}^k$ and $\hat{\boldsymbol{u}}_c^k$ in the classification model optimization.

3.3.1 Server Update

At the beginning of the training, the server initializes the parameters of the classification model $\boldsymbol{\theta} := [\boldsymbol{\theta}_f, \boldsymbol{\theta}_h]$, as well as the *shared* generator $\boldsymbol{\omega}$. In each communication round r, all clients receive the aggregated model parameters and conduct the *Client Update* procedure in parallel. Subsequently, the server securely aggregates the optimized model parameters from all the clients into a single model that is used in the next communication round.

3.3.2 Client Update

As shown in Fig. 1, at the beginning of the first communication round, each client *locally* initializes a Representation Transformation Network (*RTNet*) parameterized by $\boldsymbol{\phi}^k$. Subsequently, each client receives the classification model parameters $\boldsymbol{\theta}^k$ and the generator parameters $\boldsymbol{\omega}^k$ from the server, and conducts T local update steps. Each local update comprises 2 stages: (1) Classification model optimization, and (2) Generator and RTNet optimization.

(1) Classification Model Optimization: In this stage, the generator and the RTNet are fixed, while the classification model is updated by minimizing the loss $\mathcal{L}_{cls}$, where

$$\begin{aligned} &\mathcal{L}_{cls} = \mathcal{L}_{real} + \mathcal{L}_{syn}, \\ &\text{with} \quad \mathcal{L}_{real} = L_{\text{CE}}(h^k(f^k(\boldsymbol{X}^k)), \boldsymbol{y}^k). \end{aligned} \tag{1}$$

While $\mathcal{L}_{real}$ is minimized to update the model parameter $\boldsymbol{\theta}^k$ by using real training samples from D^k, $\mathcal{L}_{syn}$ is minimized to update only the prediction head h^k as it is computed on synthetically generated samples in the embedding space $\mathcal{U}$. We use cross-entropy (L_{CE}) for both loss functions.

To generate domain-specific synthetic embeddings, the shared generator g^k and local RTNet m^k are used to generate residuals that are added to the embeddings of real examples produced by the local feature extractor f^k. Hereby, we first generate client-agnostic embeddings $\hat{\boldsymbol{v}}^k$ by feeding a batch of random vector $\boldsymbol{z}$, sampled from standard Gaussian distribution $\mathcal{N}(0, \boldsymbol{I})$, and class labels $\boldsymbol{y}$ into the generator g^k. Subsequently, $\hat{\boldsymbol{v}}^k$ are transformed by the local RTNet into client-specific residuals and added to the embeddings of real datapoints. We distinguish two types of synthetic embeddings that we generate to train the local prediction head: domain-specific synthetic embeddings $\hat{\boldsymbol{u}}^k$ and class-prototypical domain-specific synthetic embeddings $\hat{\boldsymbol{u}}_c^k$ for category c. The domain-specific embeddings $\hat{\boldsymbol{u}}^k$ are generated by adding synthetic residuals to the embeddings $\boldsymbol{u}^k$ of real examples from the current batch sampled from D^k. On the other hand, synthetic residuals are added to class-prototypes $\overline{\boldsymbol{u}}_c^k$, *i.e.*, class-wise average embeddings of real examples, to produce $\hat{\boldsymbol{u}}_c^k$, which stabilizes the training and increase the variance of the generated embeddings.

$$\mathcal{L}_{syn} = L_{\text{CE}}(h^k(\hat{\boldsymbol{u}}^k), \boldsymbol{y}) + \sum_{c \in C} L_{\text{CE}}(h^k(\hat{\boldsymbol{u}}_c^k), c), \tag{2}$$

$$\begin{aligned} \text{with} \quad \hat{\boldsymbol{u}}^k &= \boldsymbol{u}^k + \lambda_{syn} \cdot m^k(g^k(\boldsymbol{z}, \boldsymbol{y})), \\ \hat{\boldsymbol{u}}_c^k &= \overline{\boldsymbol{u}}_c^k + \lambda_{syn} \cdot m^k(g^k(\boldsymbol{z}', c)). \end{aligned} \tag{3}$$

To compute the class-wise average embedding $\overline{\boldsymbol{u}}_c^k$, we use the exponential moving average (EMA) scheme, at each local iteration. In particular,

$$\overline{\boldsymbol{u}}_c^k \leftarrow (1 - \lambda_c) \cdot \overline{\boldsymbol{u}}_c^k + \lambda_c \cdot \frac{\sum_{i \in B} \mathbb{1}(\boldsymbol{y}_i = c) \cdot f(\boldsymbol{x}_i)}{\sum_{i \in B} \mathbb{1}(\boldsymbol{y}_i = c) + \epsilon}, \tag{4}$$

where $\mathbb{1}(\cdot)$ denotes the indicator function, B is the batch size of the real samples, and ϵ is a small number added for numerical stability. By using the average embeddings of previous iterations, we enable the examples of previously sampled batches to contribute to the computation of the current average embeddings. The ratio λ_c follows an exponential ramp-up schedule as proposed in [33].

We note that, in Eq. (3), for the generation of $\hat{\boldsymbol{u}}^k$, the original labels $\boldsymbol{y}$ of the sampled data batch are used for the residual generation, since the residuals are added to the embeddings of the examples corresponding to these labels. For $\hat{\boldsymbol{u}}_c^k$, we feed the label c that corresponds to the class of the average embedding $\overline{\boldsymbol{u}}_c^k$. While the residuals produced by the generator and the RTNet are random in early training iterations due to the random initialization of these models, they become more informative as training progresses. To reflect this in our algorithm, we employ the weighting coefficient λ_{syn} (Eq. (3)) that controls the impact of the residuals, and increase it following an exponential schedule throughout training.

To allow the different client-specific models to learn feature extractors tailored to their data distribution D^k, while still benefiting from the collaborative learning, we use local Batch Normalization layers (BN) [24] as introduced in [43].

(2) Generator and RTNet Optimization: In the second stage, the classification model is fixed while the generator and the RTNet are optimized. The class-conditional generator g^k takes a batch of random vectors $\boldsymbol{z}$ and class labels $\boldsymbol{y}$ to produce *client-agnostic* feature embeddings $\hat{\boldsymbol{v}}^k$. $\hat{\boldsymbol{v}}^k$ are then fed into the RTNet m^k to be adapted to the feature distribution of the corresponding client k. The resulting residuals are added on the embeddings of real examples to produce the *domain-specific* synthetic embeddings $\hat{\boldsymbol{u}}^k$ and $\hat{\boldsymbol{u}}_c^k$. The generator will be optimized by minimizing the loss $\mathcal{L}_{gen}$, with

$$\mathcal{L}_{gen} = L_{\text{CE}}(h^k(\hat{\boldsymbol{v}}^k), \boldsymbol{y}) - \alpha L_{\text{MMD}}(\hat{\boldsymbol{v}}^k, \boldsymbol{u}^k). \tag{5}$$

The minimization of the cross-entropy loss L_{CE} incentivizes the shared generator to produce features that are recognized by the prediction heads of all the clients. By sharing and optimizing the generator across all clients, we ensure that the synthetic embeddings produced by the generator, *i.e.*, $\hat{\boldsymbol{v}}^k$, capture client-agnostic semantic information. Additionally, we maximize the statistical distance [61] between $\hat{\boldsymbol{v}}^k$ and the real feature embeddings $\boldsymbol{u}^k$. By doing so, we force $\hat{\boldsymbol{v}}^k$ not to follow any client-specific distribution, and thus enhance the variance of the augmented feature space. Here, we adopt Maximum Mean Discrepancy (MMD) [18] as the distance metric. Subsequently, the client-agnostic embeddings are fed into the RTNet m^k parametrized by $\boldsymbol{\phi}^k$ to produce domain-specific embeddings $\hat{\boldsymbol{u}}^k$ and $\hat{\boldsymbol{u}}_c^k$. $\boldsymbol{\phi}^k$ is optimized by minimizing the loss $\mathcal{L}_{rt}$, where

Algorithm 1 Training procedure of FRAug

```
ServerUpdate
Randomly initialize $\boldsymbol{\theta}_0, \boldsymbol{\omega}_0$
for round $r$ = 1 to $R$ do
  for client $k$ = 1 to $K$ do {in parallel}
    $\boldsymbol{\theta}_r^k, \boldsymbol{\omega}_r^k \leftarrow$ ClientUpdate($\boldsymbol{\theta}_{r-1}, \boldsymbol{\omega}_{r-1}, k, r$)
  $\boldsymbol{\theta}_r \leftarrow \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{\theta}_r^k$
  $\boldsymbol{\omega}_r \leftarrow \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{\omega}_r^k$

ClientUpdate($\boldsymbol{\theta}, \boldsymbol{\omega}, k, r$)
if $r = 1$ then
  Randomly initialize $\boldsymbol{\phi}^k$
$\boldsymbol{\theta}^k \leftarrow \boldsymbol{\theta},\ \boldsymbol{\omega}^k \leftarrow \boldsymbol{\omega}$
for local step $t$ = 1 to $T$ do
  Sample $\{\boldsymbol{X}, \boldsymbol{y}\}$ from $D_k$
  Sample $\boldsymbol{z}, \boldsymbol{z}' \sim \mathcal{N}(0, I)$
  Optimize $\boldsymbol{\theta}^k$ (Eq. (1))
  Optimize $\boldsymbol{\omega}^k$ (Eq. (5)) and $\boldsymbol{\phi}^k$ (Eq. (6))
```

$$\begin{aligned}\mathcal{L}_{rt} = &- L_{\text{ent}}(h^k(\hat{\boldsymbol{u}}^k)) - \sum_{c \in C} L_{\text{ent}}(h^k(\hat{\boldsymbol{u}}_c^k)) \\ &+ \beta(L_{\text{MMD}}(\hat{\boldsymbol{u}}^k, \boldsymbol{u}^k) + \sum_{c \in C} L_{\text{MMD}}(\hat{\boldsymbol{u}}_c^k, \overline{\boldsymbol{u}}_c^k)).\end{aligned} \tag{6}$$

Here, we maximize the entropy (L_{ent}) of the prediction head output on $\hat{\boldsymbol{u}}^k$, $\hat{\boldsymbol{u}}_c^k$ to incentivize the generation of synthetic embeddings that are *hard* to classify for the prediction head h^k. To avoid generating outliers, we align the synthetic embedding distribution with that of the client local data by minimizing their Maximum Mean Discrepancy (MMD). In particular, we penalize high MMD distances between $\hat{\boldsymbol{u}}^k$ and $\boldsymbol{u}^k$, as well as $\hat{\boldsymbol{u}}_c^k$ and $\overline{\boldsymbol{u}}_c^k$ for each class c. α and β denote weighting coefficients in Eq. (5) and Eq. (6), respectively.

4. Experiments and Analyses

We conduct an extensive empirical analysis to investigate the proposed method and its viability. Firstly, we compare FRAug with several FL baseline methods on 3 popular benchmark datasets involving feature distribution shifts. Subsequently, we validate our approach on a real-world medical dataset for genetic treatment classification. We present additional analysis regarding convergence rate, communication overhead, and robustness to input noise. Finally, we demonstrate the ablation studies of FRAug and its comparison with other augmentation-based FL methods.

4.1. Benchmark Experiments

4.1.1 Datasets Description

We conduct experiments on three common image classification benchmarks with domain shift: (1) *OfficeHome* [59], which contains 65 classes in four domains: Art (A), Clipart (C), Product (P) and Real-World (R). (2) *PACS* [38], which includes images that belong to 7 classes from four domains Art-Painting (A), Cartoon (C), Photo (P), and Sketch (S). (3) *Digits* comprises images of 10 digits from the following four datasets: MNIST (MT) [34], MNIST-M (MM) [15], SVHN (SV) [51], and USPS (UP) [23]. Each client contains data from one of the domains, *i.e.*, there exists feature distribution shifts across different clients. To simulate data scarcity described in previous sections, we assume that only 10% (1% for the Digits dataset) of the original data is available for each client, resulting in ca. 100 to 1000 data samples per client following the experimental setup in the previous work [49, 43].

4.1.2 Baselines

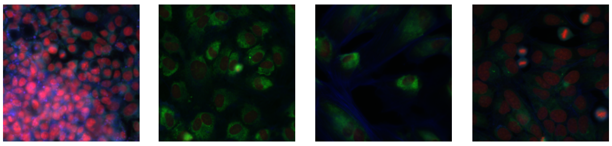
We compare our approach with several baseline methods, including *Single*, *i.e.*, training an individual model on each client separately, *All*, *i.e.*, training a single model at the central server using data aggregated from all clients, *FedAvg* [49], *pFedAvg*, *i.e.*, FedAvg with local model personalization. We also compare FRAug with *FedProx* [42], *FedBABU* [52], and *FedProto* [57], which are strong concurrent methods handling label space heterogeneity in FL. We note that *All* is an oracle baseline as it requires centralizing the data from the different clients, hence infringing the data-privacy requirements. Furthermore, we compare our method with the current state-of-the-art FL methods for non-IID features, *i.e.*, *FedBN* [43] and *PartialFed* [55]. We use the published code of *FedBN* and reimplement *PartialFed* since the original implementation was not made public. We conduct the same hyperparameter search for all methods and report the best results. The detailed hyperparameter search spaces of different methods are provided in Appendix A.

4.1.3 Implementation Details

For the OfficeHome and PACS datasets, we use a ResNet18 [22] pretrained on ImageNet [9] as initialization of the classification model. For Digits, we use a 6-layer Convolution Neural Network (CNN) as the backbone following prior work [43]. We adopt a 2-layer MLP as the generator and RTNet architectures for all datasets. Besides, we apply the same data augmentation techniques on the input images during the classification model training for all clients following the previous work [19]. In Appendix A, we provide further details about model architectures and training hyperparameters. All experiments are repeated with 3 different random seeds.

Benchmark		Single	All	FedAvg	FedProx	FedProto	FedBABU	pFedAvg	PartialFed	FedBN	FRAug
Office Home	A	35.80±0.1	56.65±0.7	57.47±0.6	55.68±0.4	51.44±0.6	49.80±0.4	52.50±0.9	48.83±0.2	57.59±0.8	**57.61**±0.6
	C	45.54±0.8	58.81±1.6	56.74±0.9	56.88±0.5	52.63±0.7	54.23±0.7	52.09±1.1	49.96±0.2	56.52±0.3	**60.03**±0.5
	P	67.04±0.8	71.39±0.3	73.32±0.8	73.84±0.3	70.78±0.7	70.72±0.6	71.78±0.8	72.22±0.8	73.55±1.0	**74.03**±0.8
	R	61.16±0.7	72.63±1.3	71.25±0.3	72.15±0.9	65.13±0.2	66.74±0.5	66.28±0.4	65.82±0.6	72.40±0.9	**74.58**±0.4
	avg	52.42±0.4	64.87±0.9	64.69±0.6	64.63±0.6	60.00±0.3	60.37±0.3	60.67±0.7	59.20±0.5	65.02±0.7	**66.60**±0.3
Digits	MT	96.68±0.2	97.04±0.1	96.85±0.1	96.90±0.1	96.80±0.1	97.38±0.2	96.40±0.2	97.13±0.1	97.03±0.1	**97.81**±0.1
	MM	77.77±0.5	77.04±0.1	73.51±0.2	72.60±0.4	78.16±0.6	79.30±0.8	77.56±0.4	74.21±0.5	77.02±0.2	**81.65**±0.5
	SV	75.55±0.3	77.96±0.5	74.49±0.2	73.01±0.5	77.90±0.2	74.03±0.5	77.50±0.1	78.10±0.5	77.59±0.1	**81.24**±0.3
	UP	79.93±0.8	97.13±0.1	97.62±0.1	97.31±0.3	97.37±0.1	95.37±0.4	96.67±0.1	94.78±0.5	96.80±0.2	**97.67**±0.3
	avg	82.54±0.1	87.29±0.2	85.62±0.2	84.96±0.3	87.50±0.1	86.52±0.4	87.03±0.2	86.05±0.3	87.11±0.2	**89.59**±0.4
PACS	A	82.37±0.6	83.17±0.2	82.72±0.4	80.17±0.4	85.09±0.5	81.25±0.6	88.05±0.8	84.85±0.2	86.60±0.5	**87.34**±0.5
	C	86.08±0.9	86.92±0.8	84.04±1.3	82.04±0.8	86.91±0.3	87.76±1.1	86.20±0.7	87.92±0.5	87.76±1.0	**88.47**±0.9
	P	92.01±1.1	95.95±0.8	96.05±0.5	96.74±1.0	96.49±0.6	94.74±0.4	97.89±0.5	98.24±0.4	97.95±0.4	**98.64**±0.6
	S	87.52±0.8	88.70±0.7	89.50±0.7	88.50±1.0	89.20±0.4	89.41±0.3	88.89±0.9	90.10±0.8	90.75±0.3	**90.95**±0.4
	avg	87.00±0.5	88.68±0.6	88.08±0.9	86.86±0.9	89.42±0.5	88.29±0.6	90.26±0.6	90.28±0.7	90.76±0.3	**91.34**±0.1

Table 2: Evaluation results of different algorithms on three real-world benchmark datasets with feature distribution shift. We report the mean±std accuracy of each client from 3 runs with different seeds. The best results are marked in **bold** (The same applies to the subsequent tables).



(a) HEPG2 (b) HUVEC (c) RPE (d) U2OS

Figure 2: Example images of different cell types, *i.e.*, local data from different clients, in RxRx1 dataset. Strong feature space heterogeneity can be observed between image appearance. *Best viewed in color.*

4.1.4 Results and Discussion

We report the accuracies achieved by the different methods on all three datasets in Tab. 2. We observe that FRAug outperforms all the baselines on all benchmark datasets. On OfficeHome, FRAug outperforms FedAvg and FedBN by 1.91% and 1.58%, respectively. On Digits, FRAug achieves a substantial 2.3% improvement on average compared with all the alternative methods. Likewise, FRAug yields the highest average accuracy on PACS. We note that FRAug achieves an average performance increase of 1.6% compared to FedBN across all three datasets, which surpasses the average performance improvement yielded by FedBN on FedAvg, *i.e.*, 1.5%. Moreover, we find that the performance improvement compared to the best baseline is the highest on the most challenging domains, *i.e.*, on which all methods yield lower results than on other domains. These include MNIST-M and SVHN from Digits, as well as Clipart from OfficeHome, where FRAug achieves impressive improvements of above 3%. Interestingly, our approach outperforms the centralized baseline *All*, demonstrating its effectiveness in aggregating the knowledge from different clients to enable a client-specific augmentation.

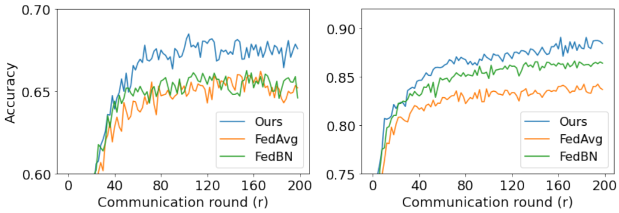
4.2. Validation on a Real-World Medical Dataset

4.2.1 Experimental Setup

To illustrate the effectiveness of FRAug on real-world applications, we further conduct experiments on the RxRx1 [58] medical dataset, which contains images (Fig. 2) of cells obtained by fluorescent microscopy. The task is to classify which genetic treatment the cells received. There are 4 different cell types adopted in the dataset, *i.e.*, HEPG2 (H), HUVEC (V), RPE (R), and U2OS (U), while multiple batches of experiments are executed for each cell type. Despite the careful control of experimental variables, *e.g.*, temperature and humidity, feature space heterogeneity is observed across different batches of experiments [32]. Therefore, we consider 4 different cell types as 4 different domains. We divide the batches of experiments from each domain, *i.e.*, for each cell type, into 4 groups, where each group has the same number of batches and is assigned to one client. By doing so, we simulate a real-world collaborative training setup of different medical institutions where every institution has conducted some batches of experiments on one specific cell type. We note that the number of domains is not equal to the number of clients. Following the FL setting described in the previous section, we select 50 classes from 1139 classes in the original dataset. We adopt ResNet18 [22] pretrained on ImageNet [9] as initialization of the classification model. To further evaluate the scalability of the proposed method, we conduct experiments where 2, 3, and 4 clients from each domain are selected, which gives in total 8, 12, and 16 clients joining the federated communication, respectively. Note that more clients correspond to larger data quantity.

Method	8 clients					12 clients					16 clients				
	H	V	P	U	avg	H	V	P	U	avg	H	V	P	U	avg
FedAvg	24.31 ±0.3	34.39 ±0.8	20.19 ±1.3	17.65 ±0.9	24.14 ±0.8	28.84 ±1.3	40.60 ±0.9	19.72 ±0.7	16.67 ±0.8	26.46 ±0.8	28.17 ±0.7	41.60 ±1.0	23.55 ±0.8	17.65 ±0.8	27.74 ±0.6
HarmoFL	19.61 ±1.0	44.02 ±0.5	20.18 ±0.2	**22.53** ±0.9	26.58 ±1.0	26.61 ±0.8	**49.15** ±0.5	19.27 ±0.7	17.97 ±0.9	28.25 ±0.8	28.57 ±0.9	47.29 ±0.7	22.02 ±0.5	18.05 ±0.7	28.98 ±0.4
FedBN	22.94 ±0.9	43.70 ±0.5	25.92 ±1.0	18.63 ±0.9	27.80 ±1.0	27.22 ±0.4	46.01 ±0.4	26.85 ±0.8	16.95 ±1.1	29.26 ±0.6	29.35 ±0.6	**49.08** ±0.8	29.58 ±0.3	19.97 ±0.2	31.99 ±0.3
FRAug	**28.28** ±0.3	**45.33** ±0.9	**28.74** ±1.2	21.04 ±0.5	**30.84** ±0.5	**30.73** ±0.9	47.36 ±0.8	**30.58** ±0.2	**19.60** ±0.7	**32.07** ±0.5	**32.34** ±0.4	48.05 ±0.5	**31.83** ±1.0	**20.59** ±0.7	**33.20** ±0.8

Table 3: Evaluation results of different methods on real-world medical dataset RxRx1. We conduct experiments with different number of clients for each cell type and report average accuracy of clients holding the same cell type.



(a) OfficeHome

(b) Digits

Figure 3: Convergence analysis of FedAvg, FedBN and FRAug on (a) OfficeHome and (b) Digits benchmarks.

4.2.2 Results and Discussion

In Tab. 3, we compare FRAug with FedAvg, FedBN, and HarmoFL [27], which is a concurrent work that proposed a strong FL method tailored for heterogeneous medical images, and report the average validation accuracy of clients owning data from the same domain (cell type). We observe that FRAug outperforms all competitors over all settings with different numbers of clients and different data amounts. We highlight the performance improvements achieved by FRAug compared with the baselines, *i.e.*, when 8, 12, and 16 clients join the federated collaborative training, our approach surpasses the other methods by at least 3.04%, 2.81%, and 1.21%, respectively. These results indicate the effectiveness of FRAug on settings with larger quantities of training data as well as its scalability to the complex real-world FL scenarios with more clients.

4.3. Additional Analyses

4.3.1 Convergence Analysis

In Fig. 3, we display the convergence analysis of the proposed method compared with the baseline FedAvg and FedBN on the OfficeHome and Digits benchmarks. Hereby, we report the average classification accuracy of all clients on their corresponding local testing set after conducting the communication round r. As shown in the figure, even though FRAug utilizes the representation augmentation technique, the learning curves of FRAug still exhibit better convergence rates. It's also worth noticing that FRAug already achieves distinct performance gain after 50 communication rounds, *i.e.*, 25% of the total rounds.

Model	Parameters(M)	MACs(G)
ResNet18	11.18	1.84
CNN for Digits	18.15	0.08
Generator	0.39	$\ll 0.01$
RTNet	0.26	$\ll 0.01$

Table 4: Parameters number and MACs (Multiply Accumulate operations) comparison of different components in FRAug.

4.3.2 Analysis of Communication Overhead

In Tab. 4, we demonstrate the number of model parameters and computational costs, *i.e.*, the number of operations, of different components used in the proposed method. We observe that both generator and RTNet take only 2-3% of the parameter numbers used in the classification model, proving the communication overhead between client and server is negligible. Besides, we notice that only less than 1% of operations are needed for the newly introduced components in FRAug compared with the classification model. Therefore, we conclude that FRAug is communication efficient and does not impose significant impacts on the clients local training, showing its applicability to clients with edge devices and limited computing power.

4.3.3 Ablation Study

To illustrate the importance of different FRAug components, we conduct an ablation study on three benchmark datasets. The results are shown in Tab. 5. We first notice that applying only the client-specific RTNet solely based on local data is ineffective: Its output $\hat{\boldsymbol{u}}^k$ is restricted in the client local distribution when the client-agnostic feature embeddings are inaccessible, which proves the criticality of optimizing a shared generator G. We further observe that using the client-agnostic synthetic embeddings $\hat{\boldsymbol{v}}^k$ instead of

G ($\hat{\boldsymbol{v}}$)	RTNet ($\hat{\boldsymbol{u}}$)	EMA ($\hat{\boldsymbol{u}}_c$)	OfficeHome	PACS	Digits
	✓		64.58±0.5	88.38±0.5	86.23±0.2
	✓	✓	65.08±0.4	88.50±0.2	86.60±0.1
✓			65.47±0.8	90.82±0.5	87.25±0.1
✓		✓	66.09±0.2	90.74±0.4	88.24±0.3
✓	✓		65.99±0.3	**91.35**±0.1	89.51±0.1
✓	✓	✓	**66.60**±0.4	91.05±0.3	**89.59**±0.2

Table 5: Ablation study for different components of FRAug on three benchmark datasets. The average evaluation accuracy of all clients are reported

the personalized versions leads to slight performance gain. This highlights the importance of the transformation by RTNets into personalized client-specific embeddings. Moreover, the results reveal that both types of synthetic embeddings, *i.e.*, $\hat{\boldsymbol{u}}_c^k$ and $\hat{\boldsymbol{u}}^k$, yield a performance boost when used separately. Employing them together further improves the results, which demonstrates their complementarity.

Additionally, we evaluate the proposed algorithm optimized with different combinations of hyperparameters. From the results, we observe low sensitivity of FRAug to the hyperparameter selection, highlighting its applicability on novel benchmark datasets without time-consuming fine-grained hyperparameter searches. Besides, we conduct experiments with varying numbers of datapoints available on each client. The superior performance of FRAug further indicates its robustness under both data-scarce and data-sufficient scenarios in FL. The detailed evaluation results are provided in Appendix B.

4.3.4 Robustness to Input Noise

Prior works [25, 63, 64] focus on generating or adversarially augmenting the clients local training data. On the contrary, the representation generators used in FRAug extract knowledge from the output of the existing feature extractor, i.e., they do not access the input images. More importantly, FRAug does not impose any constraints on the client local update and model aggregation, which indicates its compatibility with the defensive strategies introduced in [11, 48].

Noise Intensity	Weak	Medium	Strong
FedAvg	63.02±0.4	60.71±0.6	31.26±1.2
FedBN	63.97±0.6	60.12±0.5	30.90±0.9
FRAug	**64.72**±1.0	**61.45**±0.8	**31.65**±0.7

Table 6: Evaluation results of different methods on privatized OfficeHome with different noise intensity. The average accuracy of all clients are reported.

To exhibit the effectiveness of FRAug under the settings with noisy input, we add random noise $\delta \sim \mathcal{N}(0, \sigma^2\mathbf{I})$ to the client local images when optimizing the classification model. More specifically, we select three noise intensities from weak ($\sigma = 0.01$), medium ($\sigma = 0.1$), to strong ($\sigma = 1.0$). The results in Tab. 6 indicate the effectiveness of FRAug under noisy client local data.

Method	A	C	P	R	avg
FedAvg	57.47±0.6	56.74±0.9	73.32±0.8	71.25±0.3	64.69±0.6
$\mathcal{U}(-\gamma, \gamma)$	56.79±0.2	57.47±0.8	72.07±0.2	73.51±0.2	64.96±0.3
$Lap(0, \gamma)$	56.52±0.4	56.37±0.2	72.29±0.2	73.83±0.9	64.75±0.4
$\mathcal{N}(0, \gamma)$	56.93±0.9	57.63±0.5	72.43±0.2	73.27±0.5	65.06±0.4
FAug	50.18±0.5	53.48±0.9	71.82±0.4	66.08±0.8	60.39±0.7
FedReg	53.50±0.3	56.52±0.4	69.36±0.7	68.57±0.2	62.00±0.4
FRAug	**57.61**±0.6	**60.03**±0.5	**74.03**±0.8	**74.58**±0.4	**66.60**±0.3

Table 7: Evaluation results of different augmentation methods on OfficeHome benchmark.

4.3.5 Comparison with Other Augmentation Methods

Since the proposed method applies augmentation in the representation space, we compare FRAug with other augmentation approaches using random noise $\Delta\boldsymbol{u}$ following different distributions. Specifically, we train the prediction head h with real feature embeddings $\boldsymbol{u}$ as well as their augmented variants $\boldsymbol{u} + \Delta\boldsymbol{u}$. We adopt three common distributions for sampling the values of $\Delta\boldsymbol{u}$: Uniform distribution $\mathcal{U}$, Laplace distribution Lap and Gaussian distribution $\mathcal{N}$. We define the standard deviation γ of each distribution as a hyperparameter and report the best results. Moreover, we compare our method with concurrent works applying data augmentation, *i.e.*, *FAug* [25] and *FedReg* [63].

In Tab. 7, we display the evaluation results of representation augmentation approaches with random noise, as well as the concurrent works, on the OfficeHome benchmark. We notice a distinct performance gap between these methods and FRAug, which further highlights the effectiveness of the proposed method.

5. Conclusion

In this work, we present a novel approach to tackle the under-explored feature non-IID problem in FL. The proposed Federated Representation Augmentation (FRAug) method performs client-personalized augmentation in the embedding space to improve the training robustness against feature distribution shift. For that, we optimize a shared generative model to synthesize embeddings by exploiting knowledge from all clients. The output client-agnostic embeddings are then transformed into client-specific embeddings by local Representation Transformation Networks (RTNets). FRAug achieves state-of-the-art results on three benchmark datasets involving feature distribution. Moreover, the superb results of FRAug on a medical dataset illustrate its effectiveness and scalability on complex real-world FL applications.

References

[1] Samiul Alam, Luyang Liu, Ming Yan, and Mi Zhang. Fedrolex: Model-heterogeneous federated learning with rolling sub-model extraction. *arXiv preprint arXiv:2212.01548*, 2022.

[2] Mathieu Andreux, Jean Ogier du Terrail, Constance Beguier, and Eric W Tramel. Siloed federated learning for multi-centric histopathology datasets. In *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning*, pages 129–139. Springer, 2020.

[3] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers. *arXiv preprint arXiv:1912.00818*, 2019.

[4] Chen Chen, Yuchen Liu, Xingjun Ma, and Lingjuan Lyu. Calfat: Calibrated federated adversarial training with label skewness. *arXiv preprint arXiv:2205.14926*, 2022.

[5] Liang Chen, Yihang Lou, Jianzhong He, Tao Bai, and Minghua Deng. Evidential neighborhood contrastive learning for universal domain adaptation. 2022.

[6] Tong Chu, Yahao Liu, Jinhong Deng, Wen Li, and Lixin Duan. Denoised maximum classifier discrepancy for source-free unsupervised domain adaptation. In *Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI-22)*, volume 2, 2022.

[7] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In *International Conference on Machine Learning*, pages 2089–2099. PMLR, 2021.

[8] Yatin Dandi, Luis Barba, and Martin Jaggi. Implicit gradient alignment in distributed and federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6454–6462, 2022.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[10] Qi Dou, Daniel Coelho de Castro, Konstantinos Kamnitsas, and Ben Glocker. Domain generalization via model-agnostic learning of semantic features. *Advances in Neural Information Processing Systems*, 32, 2019.

[11] David Enthoven and Zaid Al-Ars. An overview of federated deep learning privacy attacks and defensive strategies. *Federated Learning Systems: Towards Next-Generation AI*, pages 173–196, 2021.

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

[13] Ahmed Frikha, Haokun Chen, Denis Krompaß, Thomas Runkler, and Volker Tresp. Towards data-free domain generalization. *arXiv preprint arXiv:2110.04545*, 2021.

[14] Ahmed Frikha, Denis Krompaß, and Volker Tresp. Columbus: Automated discovery of new multi-level features for domain generalization via knowledge corruption. *arXiv preprint arXiv:2109.04320*, 2021.

[15] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015.

[16] Xuan Gong, Abhishek Sharma, Srikrishna Karanam, Ziyan Wu, Terrence Chen, David Doermann, and Arun Innanje. Ensemble attention distillation for privacy-preserving federated learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15076–15086, 2021.

[17] Xuan Gong, Abhishek Sharma, Srikrishna Karanam, Ziyan Wu, Terrence Chen, David Doermann, and Arun Innanje. Preserving privacy in federated learning with ensemble cross-domain knowledge distillation. page 3, 2022.

[18] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

[19] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *International Conference on Learning Representations*, 2020.

[20] Weituo Hao, Mostafa El-Khamy, Jungwon Lee, Jianyi Zhang, Kevin J Liang, Changyou Chen, and Lawrence Carin Duke. Towards fair federated learning with zero-shot data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3310–3319, 2021.

[21] Chaoyang He, Murali Annavaram, and Salman Avestimehr. Group knowledge transfer: Federated learning of large cnns at the edge. *Advances in Neural Information Processing Systems*, 33:14068–14080, 2020.

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

[23] Jonathan J. Hull. A database for handwritten text recognition research. *IEEE Transactions on pattern analysis and machine intelligence*, 16(5):550–554, 1994.

[24] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[25] Eunjeong Jeong, Seungeun Oh, Hyesung Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *arXiv preprint arXiv:1811.11479*, 2018.

[26] Wonyong Jeong and Sung Ju Hwang. Factorized-fl: Personalized federated learning with parameter factorization & similarity matching. In *Advances in Neural Information Processing Systems*, 2022.

[27] Meirui Jiang, Zirui Wang, and Qi Dou. Harmofl: Harmonizing local and global drifts in federated learning on heterogeneous medical images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1087–1095, 2022.

[28] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cum-

mings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

[29] Juwon Kang, Sohyun Lee, Namyup Kim, and Suha Kwak. Style neophile: Constantly seeking novel styles for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7130–7140, 2022.

[30] Payam Karisani. Multiple-source domain adaptation via coordinated domain encoders and paired classifiers. *arXiv preprint arXiv:2201.11870*, 2022.

[31] Jinkyu Kim, Geeho Kim, and Bohyung Han. Multi-level branched regularization for federated learning. In *International Conference on Machine Learning*, pages 11058–11073. PMLR, 2022.

[32] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021.

[33] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016.

[34] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[35] Gihun Lee, Minchan Jeong, Yongjin Shin, Sangmin Bae, and Se-Young Yun. Preservation of the global knowledge by not-true distillation in federated learning. *arXiv preprint arXiv:2106.03097*, 2021.

[36] Bo Li, Yifei Shen, Yezhen Wang, Wenzhen Zhu, Dongsheng Li, Kurt Keutzer, and Han Zhao. Invariant information bottleneck for domain generalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 7399–7407, 2022.

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

[38] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pages 5542–5550, 2017.

[39] Li Li, Yuxi Fan, Mike Tse, and Kuo-Yi Lin. A review of applications in federated learning. *Computers & Industrial Engineering*, 149:106854, 2020.

[40] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*, pages 6357–6368. PMLR, 2021.

[41] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

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

[43] Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization. *arXiv preprint arXiv:2102.07623*, 2021.

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

[45] Quande Liu, Cheng Chen, Jing Qin, Qi Dou, and Pheng-Ann Heng. Feddg: Federated domain generalization on medical image segmentation via episodic learning in continuous frequency space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1013–1023, 2021.

[46] Raphael Gontijo Lopes, Stefano Fenu, and Thad Starner. Data-free knowledge distillation for deep neural networks. *arXiv preprint arXiv:1710.07535*, 2017.

[47] Mi Luo, Fei Chen, Dapeng Hu, Yifan Zhang, Jian Liang, and Jiashi Feng. No fear of heterogeneity: Classifier calibration for federated learning with non-iid data. *Advances in Neural Information Processing Systems*, 34:5972–5984, 2021.

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

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

[50] Matias Mendieta, Taojiannan Yang, Pu Wang, Minwoo Lee, Zhengming Ding, and Chen Chen. Local learning matters: Rethinking data heterogeneity in federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8397–8406, 2022.

[51] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[52] Jaehoon Oh, Sangmook Kim, and Se-Young Yun. Fedbabu: Towards enhanced representation for federated image classification. *arXiv preprint arXiv:2106.06042*, 2021.

[53] Yichen Ruan and Carlee Joe-Wong. Fedsoft: Soft clustered federated learning with proximal local updating. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8124–8131, 2022.

[54] Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing across domains via cross-gradient training. *arXiv preprint arXiv:1804.10745*, 2018.

[55] Benyuan Sun, Hongxing Huo, Yi Yang, and Bo Bai. Partialfed: Cross-domain personalized federated learning via partial initialization. *Advances in Neural Information Processing Systems*, 34, 2021.

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

[57] Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. Fedproto: Federated prototype learning across heterogeneous clients. In *AAAI Conference on Artificial Intelligence*, volume 1, page 3, 2022.

[58] J. Taylor, B. Earnshaw, B. Mabey, M. Victors, and J. Yosinski. Rxrx1: An image set for cellular morphological variation across many experimental batches. In *International Conference on Learning Representations (ICLR)*, 2019.

[59] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5018–5027, 2017.

[60] Garrett Wilson and Diane J Cook. A survey of unsupervised deep domain adaptation. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(5):1–46, 2020.

[61] William K Wootters. Statistical distance and hilbert space. *Physical Review D*, 23(2):357, 1981.

[62] Renchunzi Xie, Hongxin Wei, Lei Feng, and Bo An. Gearnet: Stepwise dual learning for weakly supervised domain adaptation. *arXiv preprint arXiv:2201.06001*, 2022.

[63] Chencheng Xu, Zhiwei Hong, Minlie Huang, and Tao Jiang. Acceleration of federated learning with alleviated forgetting in local training. *arXiv preprint arXiv:2203.02645*, 2022.

[64] Tehrim Yoon, Sumin Shin, Sung Ju Hwang, and Eunho Yang. Fedmix: Approximation of mixup under mean augmented federated learning. *arXiv preprint arXiv:2107.00233*, 2021.

[65] Fuxun Yu, Weishan Zhang, Zhuwei Qin, Zirui Xu, Di Wang, Chenchen Liu, Zhi Tian, and Xiang Chen. Fed2: Feature-aligned federated learning. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 2066–2074, 2021.

[66] Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. A survey on federated learning. *Knowledge-Based Systems*, 216:106775, 2021.

[67] Luxin Zhang, Pascal Germain, Yacine Kessaci, and Christophe Biernacki. Interpretable domain adaptation for hidden subdomain alignment in the context of pre-trained source models. In *36th AAAI Conférence on Artificial Intelligence*, 2022.

[68] Lin Zhang, Li Shen, Liang Ding, Dacheng Tao, and Ling-Yu Duan. Fine-tuning global model via data-free knowledge distillation for non-iid federated learning. *arXiv preprint arXiv:2203.09249*, 2022.

[69] Lan Zhang and Xiaoyong Yuan. Fedzkt: Zero-shot knowledge transfer towards heterogeneous on-device models in federated learning. *arXiv preprint arXiv:2109.03775*, 2021.

[70] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

[71] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey. *arXiv e-prints*, pages arXiv–2103, 2021.

[72] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization in vision: A survey. *arXiv preprint arXiv:2103.02503*, 2021.

[73] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Learning to generate novel domains for domain generalization. In *European conference on computer vision*, pages 561–578. Springer, 2020.

[74] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. *arXiv preprint arXiv:2104.02008*, 2021.

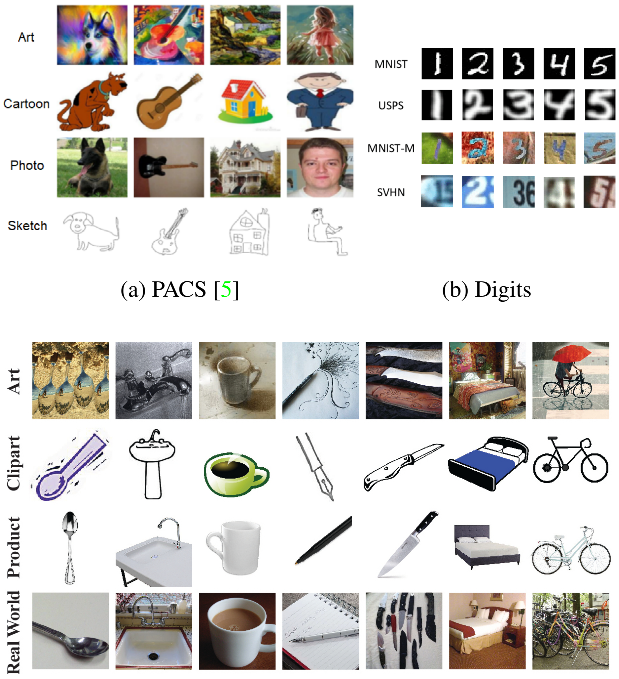
[75] Hangyu Zhu, Jinjin Xu, Shiqing Liu, and Yaochu Jin. Federated learning on non-iid data: A survey. *Neurocomputing*, 465:371–390, 2021.

[76] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. In *International Conference on Machine Learning*, pages 12878–12889. PMLR, 2021.

A. Experimental Details

A.1. Visualization of Benchmark Datasets

In this section, we show example images in different domains from the adopted benchmark datasets, *i.e.*, PACS (Fig. 1a), Digits (Fig. 1b), and OfficeHome (Fig. 1c). We can see that there exists strong appearance variation and distribution shifts across different domains, e.g., in PACS we have both photo-like realistic pictures (*Photo*) and highly abstract human sketches (*Sketch*). Therefore, by assigning data from one of the domains to each client, we are able to simulate the experimental setting with non-IID features in FL.



(a) PACS [5]

(b) Digits

(c) OfficeHome [9]

Figure 1: Example images from the selected benchmark datasets with non-IID features. *Best viewed in color.*

A.2. Hyperparameter Settings

In this section, we provide more details about the training hyperparameters, as well as their search space in Tab. 2. We use 1 NVIDIA GeForce GTX TITAN X with 12GB RAM to run the experiments. We use PyTorch [8] to implement our algorithm. For all baselines and the algorithms proposed in the previous work, we apply the same hyperparameter search as FRAug and report the best performance. For the PACS, OfficeHome, and Digits benchmarks, we apply data augmentation given in Tab. 1 during the training following the previous work [2]. For the medical dataset RxRx1, we follow the setting used in the WILDS benchmark [4] and do not apply any data augmentation. We also display the detailed hyperparameter selection for the proposed method on different benchmarks for possible future reproduction.

Augmentation	Parameters
RandomResizedCrop	portion: $[0.6, 1.0]$
RandomHorizontalFlip	probability: 0.5 (0.0 for Digits)
ColorJitter	jitter degree: 0.3
RandomGrayscale	probability: 0.1
Normalize	ImageNet statistics [1]

Table 1: Data augmentations for OfficeHome, Digits and PACS benchmarks.

In FRAug, the ratio λ_c used for computing the class-wise average embedding $\overline{u}_c^k$ is computed with an exponential ramp-up schedule. Specifically,

$$\lambda_c = \begin{cases} \lambda_0 \cdot exp(-5(1 - \frac{r}{r_0})), & r < r_0 \\ \lambda_0, & r \geq r_0 \end{cases} \tag{1}$$

where λ_0 is set to 0.3, and r_0 is set to 5% of the total communication rounds $(0.05 \cdot R)$.

For the weighting coefficient λ_{syn} that controls the impact of the generated residuals during the training, we use an exponential schedule, *i.e.*, $\lambda_{syn} = e^{0.01(r-R)}$.

A.3. Model Architecture

Following [6], we use a 6-layer CNN with its details listed in Tab. 3 for the Digits dataset: For the convolutional layer (Conv2D), we list parameters with the sequence of input and output dimensions, kernel size, stride, and padding. For the max-pooling layer (MaxPool2D), we list kernel and stride. For the fully-connected layer (FC), we list input and output dimensions. For the Batch Normalization layer (BN), we list the channel dimension. We adopt the last FC layer as the prediction head, which defines the feature dimension with 512.

For the classification models on OfficeHome and PACS datasets, we use the widely adopted backbone ResNet18 [3] and change the output dimension of the last fully-connected layer (FC) to match the class number C of the dataset. We adopt the last FC layer as the prediction head, which defines the feature dimension with 512.

The network architecture of the generator and the Representation Transformation Network (RTNet) are given in Tab. 4 and Tab. 5, respectively. For the generator, we adopt a two-layer MLP, which takes a noise vector $\boldsymbol{z}$ with dimension d_z and a one-hot encoded label y as the input, and outputs a client-agnostic feature representation $\hat{\boldsymbol{v}}$. For RT-

	Hyperparameter	OfficeHome	PACS	Digits	RxRx1
Shared Parameters	Learning rate	0.01	0.01	0.01	0.01
	Image size	224x224	224x224	32x32	256x256
	Optimizer	SGD	SGD	SGD	SGD
	Optimizer momentum	0.5	0.5	0.5	0.5
	Communication rounds (R)	200	200	200	200
Shared Search Space	Local update steps (T)	{5, 10, 20}	{5, 10, 20}	{5, 10, 20}	{5, 10, 20}
	Batch size B	{16, 32, 64}	{16, 32, 64}	{64, 128, 256}	{16, 32}
FRAug	$\boldsymbol{\omega}$ and $\boldsymbol{\phi}$ optimizer	SGD	SGD	SGD	SGD
	Synthetic batch size B_{syn}	16	32	64	32
	η_g	0.05	0.05	0.005	0.01
	η_m	0.025	0.05	0.005	0.01
	α	1.5	1.25	1	1
	β	1.25	1.25	1.5	1
	d_z	128	256	256	128

Table 2: Hyperparameter configurations for different datasets

Layer	Details

1	Conv2D(3, 64, 5, 1, 2)
	BN(64), ReLU(), MaxPool2D(2, 2)

2	Conv2D(64, 64, 5, 1, 2)
	BN(64), ReLU(), MaxPool2D(2, 2)

3	Conv2D(64, 128, 5, 1, 2)
	BN(128), ReLU(), Flatten()

4	FC(6272, 2048)
	BN(2048), ReLU()

5	FC(2048, 512)
	BN(512), ReLU()

6	FC(512, 10)

Table 3: Classification model architecture for the Digits benchmark.

Layer	Details

1	FC($d_z + C$, d_u)
	BN(d_u), ReLU()

2	FC(d_u, d_u)
	BN(d_u), ReLU()

Table 4: Generator architecture.

Layer	Details

1	FC(d_u, d_z)
	BN(d_z), ReLU()

2	FC(d_z, d_u)
	BN(d_u)

Table 5: Representation Transformation Network (*RTNet*) architecture.

Net, we adopt a two-layer MLP, which takes the output of the generator $\hat{\boldsymbol{v}}$ and outputs a client-specific feature residual with dimension d_u.

B. Additional Results and Analyses

B.1. Ablation Study

To illustrate the importance of different FRAug components, we conducted ablation studies on three benchmark datasets, *i.e.*, OfficeHome, Digit and PACS, where the results are shown in Tab. 6, Tab. 7, and Tab. 8, respectively. First, we find that solely applying the RTNet based on the real feature embeddings, *i.e.*, training without a shared generator barely brings performance gain. We assume that RTNet is restricted to the client local distribution and is only helpful when it accesses the client-agnostic. Moreover, using the client-agnostic synthetic embeddings $\hat{\boldsymbol{v}}^k$ leads to only minimal performance gain, highlighting the importance of the proposed representation transformation schema, *i.e.*, RTNet. Moreover, the results demonstrate that using both types of synthetic embeddings, *i.e.*, $\hat{\boldsymbol{u}}_c^k$ and $\hat{\boldsymbol{u}}^k$, yields the largest performance boosts for both benchmarks.

G	RTNet	EMA	OfficeHome				
($\hat{\boldsymbol{v}}$)	($\hat{\boldsymbol{u}}$)	($\hat{\boldsymbol{u}}_c$)	A	C	P	R	avg
	✓		56.61±0.3	57.08±0.5	73.14±0.6	71.49±0.4	64.58±0.5
	✓	✓	57.51±0.2	56.95±0.3	73.58±0.5	72.28±0.5	65.08±0.4
✓			56.24±0.5	59.65±0.3	72.90±0.3	73.09±0.5	65.47±0.8
✓	✓		57.34±0.9	59.10±0.4	73.65±0.7	74.25±0.9	66.09±0.2
✓		✓	56.65±0.2	59.50±0.7	73.50±0.5	74.31±0.9	65.99±0.3
✓	✓	✓	**57.61**±0.9	**60.03**±0.8	**74.03**±0.3	**74.69**±0.2	**66.60**±0.4

Table 6: Ablation study for different components of FRAug on OfficeHome benchmark. The average evaluation accuracy of all clients are reported.

G	RTNet	EMA	Digits				
$(\hat{\boldsymbol{v}})$	$(\hat{\boldsymbol{u}})$	$(\hat{\boldsymbol{u}}_c)$	MT	MM	SV	UP	avg
	✓		97.26±0.2	74.25±0.1	75.42±0.3	**97.98**±0.1	86.23±0.2
	✓	✓	96.97±0.1	75.75±0.2	75.90±0.1	97.79±0.3	86.60±0.1
✓			97.48±0.0	75.98±0.5	77.90±0.3	97.63±0.2	87.25±0.1
✓		✓	97.49±0.1	79.66±0.4	78.80±0.6	96.99±0.4	88.24±0.2
✓	✓		**97.95**±0.1	81.40±0.1	80.78±0.2	97.92±0.1	89.51±0.1
✓	✓	✓	97.81±0.1	**81.65**±0.9	**81.24**±0.3	97.67±0.4	**89.59**±0.4

Table 7: Ablation study for different components of FRAug on Digits benchmark. The average evaluation accu- racy of all clients are reported.

G	RTNet	EMA	PACS				
$(\hat{\boldsymbol{v}})$	$(\hat{\boldsymbol{u}})$	$(\hat{\boldsymbol{u}}_c)$	A	C	P	S	avg
	✓		83.80±0.5	83.95±0.3	96.64±0.2	89.12±0.4	88.38±0.5
	✓	✓	83.43±0.3	84.51±0.2	97.19±0.1	88.86±0.2	88.50±0.2
✓			86.06±0.7	**88.61**±0.9	98.24±0.1	90.37±0.4	90.82±0.5
✓	✓		86.54±0.8	88.19±0.3	98.44±0.3	89.78±1.0	90.74±0.4
✓		✓	87.34±0.5	88.47±0.9	**98.64**±0.6	**90.95**±0.4	**91.35**±0.1
✓	✓	✓	**87.50**±0.9	88.33±0.9	97.66±0.5	90.70±0.6	91.05±0.3

Table 8: Ablation study for different components of FRAug on PACS benchmark. The average evaluation accu- racy of all clients are reported.

B.2. Analysis of Local Dataset Size

In this section, we investigate the effectiveness of our representation augmentation technique for different sizes of client-specific local datasets. Hereby, we vary the number of datapoints available on each client from 100% to 10% of its original local dataset. Tab. 10 depicts the results of this experiment. We compare FRAug with two baseline methods, *i.e.*, FedAvg and Single, as well as FedBN on OfficeHome, and conduct the experiment with 3 different seeds. Compared to FedAvg and FedBN, the improvement achieved by FRAug is stable across different dataset sizes, highlighting the suitability of representation augmentation for scenarios involving non-IID features with scarce and large amounts of data. Compared to local training (*Single*) without collaboration, we observe that the performance improvement yielded by federated learning methods increases as the dataset size decreases. Note that we do not highlight the result of oracle baseline *All* when it achieves the best results, since it does not fulfill the requirement of FL, *i.e.*, datasets from different clients should be decentralized and private.

B.3. Hyperparameter Sensitivity

In this section, we further demonstrate the low sensitivity of the proposed method to the selection of different hyperparameters and present the results of the experiments.

B.3.1 Effects of α, β and d_z

In this section, we show the performance of local classification models in FRAug trained with different combinations of loss ratio in generator optimization, *i.e.*, α, loss ratio in RTNet optimization, *i.e.*, β and dimension of the random noise input of both, *i.e.*, d_z on the OfficeHome and Digits benchmark. We select α and β from $\{0.5, 0.75, 1.0, 1.25, 1.5\}$ and select d_z from $\{64, 128, 256, 512\}$. We display the results in the format of box-plots in Fig. 2 and Fig. 3. From the results, we conclude that FRAug is not sensitive to the selection of these hyperparameters.

B.3.2 Effects of η_g and η_m

In this section, we show the performance of local classification models trained with different combinations of learning rate for the generator, *i.e.*, η_g and learning rate for the RTNet, *i.e.*, η_m. Here, we select $\eta_g, \eta_m \in \{0.05, 0.025, 0.01\}$ and the results is given in Tab. 9. The results show that FRAug is robust to the selection of the learning rate of the generator and the RTNet.

	$\eta_m = 0.05$	$\eta_m = 0.025$	$\eta_m = 0.01$
$\eta_g = 0.05$	66.37	67.00	66.33
$\eta_g = 0.025$	66.69	65.98	66.71
$\eta_g = 0.01$	66.03	66.47	66.19

Table 9: Average test accuracy using different combinations of learning rate η_g and η_m on OfficeHome benchmark.

B.3.3 Effects of T and B_{real}

In this section, we further display the performance of local classification models in FRAug trained with different combinations of local update steps $T \in \{5, 10, 15, 20\}$ and the batch size for the real training samples $B_{real} \in \{16, 32, 64\}$ in Tab. 11. The results show that FRAug can consistently outperform FedAvg and is robust to the selection of the batch size of real samples as well as the local update steps.

B.4. UMAP Visualizations

In Fig. 4, we provide the UMAP [7] visualization of the feature embeddings, extracted by the models optimized by FedAvg and FRAug in PACS benchmark. From the results, we observe that the features extracted by FRAug show better separability, indicating the better robustness of FRAug against the feature distribution shift.

Client	Method	100%	80%	60%	40%	20%	10%
Art	Single	73.06±1.0	69.96±1.2	67.35±1.3	62.28±1.3	50.21±1.8	35.80±0.2
	FedAvg	72.43±0.9	71.06±1.6	68.48±1.4	65.48±1.5	62.28±1.4	56.38±1.1
	FedBN	72.55±0.7	71.78±0.6	68.98±0.5	65.22±0.9	63.58±0.7	57.59±0.8
	FRAug	**73.11**±0.7	**72.99**±1.0	**69.07**±1.4	**66.53**±1.7	**64.20**±0.6	**57.61**±0.6
	All (Orcale)	67.76±1.8	66.12±0.9	67.63±1.9	63.79±0.9	62.41±1.9	56.65±0.7
Clipart	Single	**80.09**±1.6	77.66±0.4	74.29±1.2	68.65±0.3	53.24±1.4	45.54±0.8
	FedAvg	77.57±0.5	77.50±1.2	75.74±1.0	73.03±0.2	67.05±1.3	57.21±0.9
	FedBN	77.96±0.5	77.40±0.8	76.25±0.4	73.46±0.6	67.75±0.9	56.52±0.3
	FRAug	78.92±1.3	77.65±1.0	**76.85**±0.6	**73.53**±1.4	67.66±1.1	**60.03**±0.5
	All (Orcale)	78.71±1.2	78.64±1.3	76.28±1.0	73.30±0.2	68.57±2.0	58.81±1.6
Product	Single	86.51±0.9	85.56±1.8	84.08±1.5	83.03±0.9	73.95±1.8	67.04±0.8
	FedAvg	85.21±1.0	85.14±1.2	83.26±1.0	82.73±1.5	76.35±1.1	73.87±0.8
	FedBN	85.52±0.7	84.46±0.9	84.06±1.0	81.95±0.8	77.95±0.3	73.55±1.0
	FRAug	**86.94**±0.5	**85.91**±1.3	84.42±0.9	**83.55**±1.3	**78.38**±0.4	**74.03**±0.8
	All (Orcale)	85.81±0.3	84.98±1.3	84.68±1.6	83.10±0.9	76.57±1.8	71.39±0.3
Real World	Single	81.57±1.5	79.74±1.3	75.92±1.3	71.94±0.8	65.83±1.5	61.16±0.7
	FedAvg	82.07±0.5	81.65±0.9	80.14±0.9	78.40±1.1	75.61±1.3	70.64±0.3
	FedBN	82.75±0.4	81.73±0.5	80.74±1.0	79.92±0.9	76.61±0.8	72.40±0.9
	FRAug	**84.14**±0.5	**82.95**±0.4	**82.26**±0.2	**81.23**±1.2	**78.06**±1.1	**74.58**±0.4
	All (Orcale)	80.31±0.8	80.28±2.0	78.90±1.5	77.29±1.0	74.62±1.7	72.63±1.3

Table 10: Model performance over different portion of the datasets, *i.e.*, using $\{100\%, 80\%, 60\%, 40\%, 20\%, 10\%\}$ of the original datasets in OfficeHome benchmark. The average accurcay of all clients are reported.

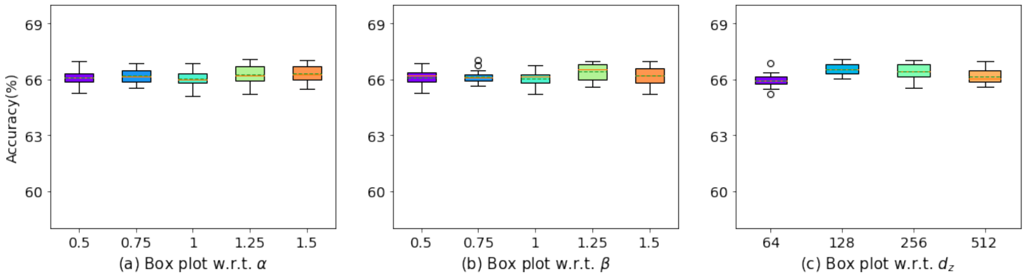


Figure 2: Evaluation results of FRAug with different hyperparameter combinations, *i.e.*, α (loss ratio in generator optimization) and β (loss ratio in RTNet optimization) and d_z (dimension of the random noise input), on OfficeHome benchmark. *Best viewed in color.*

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 1

[2] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *International Conference on Learning Representations*, 2020. 1

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[4] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021. 1

[5] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pages 5542–5550, 2017. 1

[6] Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization. *arXiv preprint arXiv:2102.07623*, 2021. 1

[7] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimen-

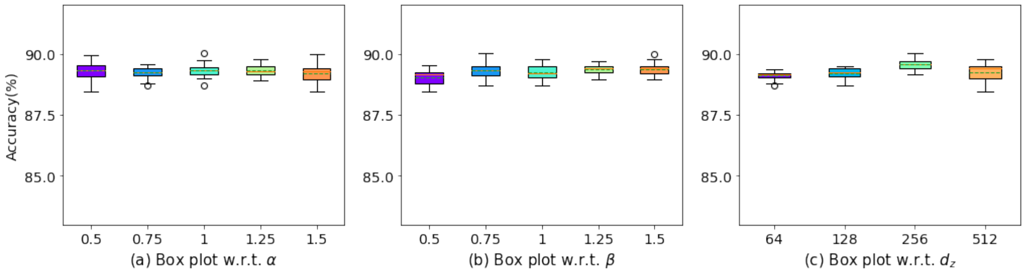


Figure 3: Evaluation results of FRAug with different hyperparameter combinations, *i.e.*, α (loss ratio in generator optimization) and β (loss ratio in RTNet optimization) and d_z (dimension of the random noise input), on Digits benchmark. *Best viewed in color.*

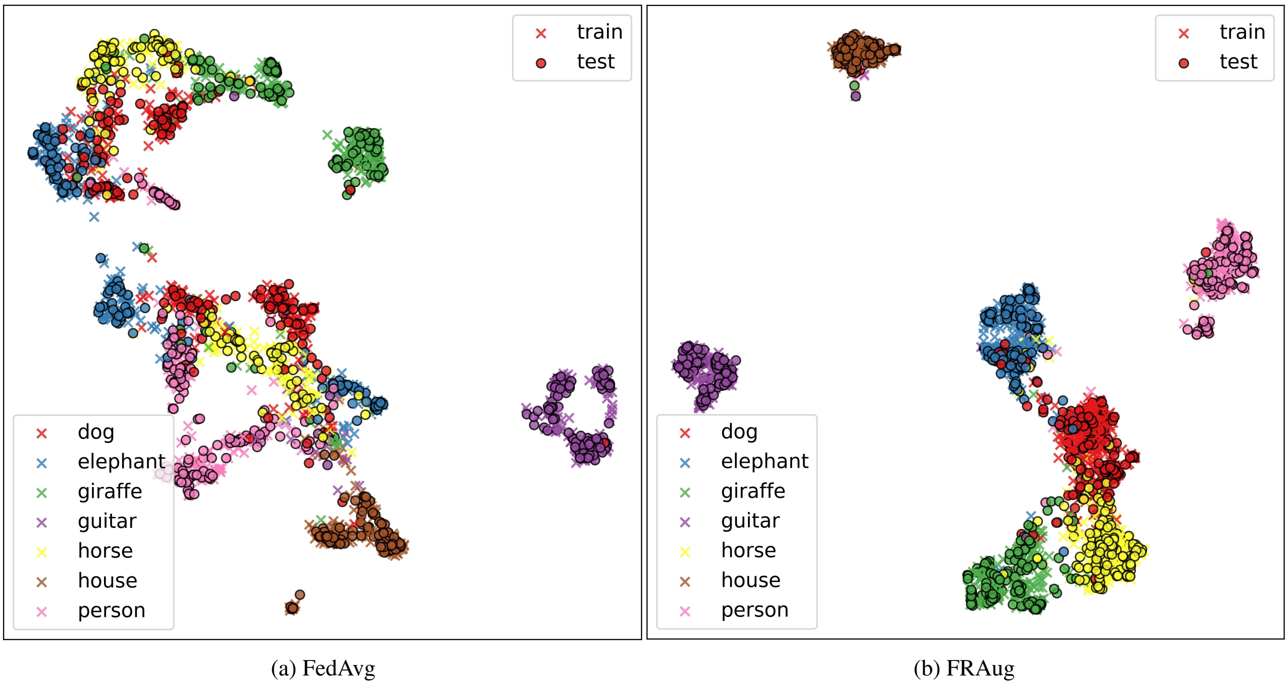
Setting	Method	Art	Clipart	Product	Real-World	Average
B_{real}=16, T=5	FedAvg	59.05±1.2	57.67±0.2	71.17±0.9	72.47±0.3	65.10±0.9
	FRAug	58.23±0.2	57.89±1.0	73.42±0.9	74.89±0.1	**66.11**±0.6
B_{real}=16, T=10	FedAvg	55.97±1.1	58.12±0.1	72.64±0.6	72.71±0.2	64.85±0.4
	FRAug	58.23±0.6	59.04±0.5	73.54±0.9	74.20±0.1	**66.25**±0.6
B_{real}=16, T=15	FedAvg	57.41±1.1	58.35±0.7	72.64±0.8	71.33±0.9	64.93±0.3
	FRAug	58.64±0.6	59.38±0.6	72.64±0.8	74.77±0.5	**66.36**±0.3
B_{real}=16, T=20	FedAvg	56.99±1.2	58.23±0.8	73.42±0.2	72.25±0.9	65.23±0.9
	FRAug	57.61±0.6	60.03±0.5	74.03±0.8	74.58±0.4	**66.60**±0.3
B_{real}=32, T=5	FedAvg	56.17±0.9	55.72±0.3	72.30±0.2	72.48±0.7	64.17±0.5
	FRAug	58.85±1.0	57.44±0.5	74.44±0.8	74.77±0.9	**66.37**±0.7
B_{real}=32, T=10	FedAvg	57.41±1.2	56.29±0.1	72.18±0.1	72.59±0.3	64.62±0.6
	FRAug	57.61±1.2	58.81±0.4	74.55±0.5	75.34±0.1	**66.58**±0.5
B_{real}=32, T=15	FedAvg	59.25±0.8	57.32±0.6	72.41±0.8	71.67±0.8	65.16±0.3
	FRAug	57.61±0.4	58.58±0.2	73.76±0.9	74.77±0.5	**66.18**±0.3
B_{real}=32, T=20	FedAvg	56.38±0.1	56.75±0.7	72.41±0.1	72.59±0.9	64.54±0.5
	FRAug	59.29±0.2	58.58±0.5	73.87±1.2	74.19±0.6	**66.58**±0.4
B_{real}=64, T=5	FedAvg	55.35±0.9	54.92±0.9	72.97±0.3	73.40±0.2	64.16±0.7
	FRAug	58.44±0.9	56.86±0.6	72.97±0.9	74.31±0.5	**65.65**±0.6
B_{real}=64, T=10	FedAvg	55.97±1.3	54.81±0.9	71.62±0.2	72.25±0.7	63.67±0.9
	FRAug	58.44±1.2	56.18±0.9	74.21±0.9	75.11±0.3	**65.99**±0.3
B_{real}=64, T=15	FedAvg	58.23±0.2	53.78±0.9	72.64±0.1	72.48±0.5	64.28±0.5
	FRAug	58.44±1.0	57.78±0.1	73.65±0.9	73.85±0.3	**65.93**±0.6
B_{real}=64, T=20	FedAvg	55.97±1.0	56.18±0.1	72.52±0.5	72.36±0.1	64.25±0.4
	FRAug	58.02±0.1	56.52±0.2	73.42±0.9	73.97±0.1	**65.48**±0.4

Table 11: Test accuracy using different combinations of batch size of real samples B_{real} and local update steps T on Office-Home benchmark.

sion reduction. *arXiv preprint arXiv:1802.03426*, 2018. 3

[8] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 1

[9] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5018–5027, 2017. 1



(a) FedAvg

(b) FRAug

Figure 4: UMAP visualization of the training and testing samples using the model optimized with FedAvg (left) and FRAug (right) on PACS benchmark. *Best viewed in color.*

Chapter 3

FedDAT: An Approach for Foundation Model Finetuning in Multi-Modal Heterogeneous Federated Learning

This chapter contains the publication

> **Haokun Chen**, Yao Zhang, Denis Krompass, Jindong Gu, Volker Tresp. FedDAT: An Approach for Foundation Model Finetuning in Multi-Modal Heterogeneous Federated Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(10), 11285-11293. https://doi.org/10.1609/aaai.v38i10.29007

FedDAT: An Approach for Foundation Model Finetuning in Multi-Modal Heterogeneous Federated Learning

Haokun Chen[1,2], **Yao Zhang**[1,4], **Denis Krompass**[2], **Jindong Gu**[3*], **Volker Tresp**[1,4]

[1] LMU Munich, Munich, Germany
[2] Siemens AG, Munich, Germany
[3] University of Oxford, Oxford, England
[4] Munich Center for Machine Learning (MCML), Munich, Germany
{haokun.chen, denis.krompass}@siemens.com, yzhang@dbs.ifi.lmu.de
jindong.gu@outlook.com, volker.tresp@lmu.de

Abstract

Recently, foundation models have exhibited remarkable advancements in multi-modal learning. These models, equipped with millions (or billions) of parameters, typically require a substantial amount of data for finetuning. However, collecting and centralizing training data from diverse sectors becomes challenging due to distinct privacy regulations. Federated Learning (FL) emerges as a promising solution, enabling multiple clients to collaboratively train neural networks without centralizing their local data. To alleviate client computation burdens and communication overheads, previous works have adapted Parameter-efficient Finetuning (PEFT) methods for FL. Hereby, only a small fraction of the model parameters are optimized and communicated during federated communications. Nevertheless, most previous works have focused on a single modality and neglected one common phenomenon, i.e., the presence of data heterogeneity across the clients. Therefore, in this work, we propose a finetuning framework tailored to heterogeneous multi-modal FL, called Federated Dual-Aadapter Teacher (FedDAT). Specifically, our approach leverages a Dual-Adapter Teacher (DAT) to address data heterogeneity by regularizing the client local updates and applying Mutual Knowledge Distillation (MKD) for an efficient knowledge transfer. FedDAT is the first approach that enables an efficient distributed finetuning of foundation models for a variety of heterogeneous Vision-Language tasks. To demonstrate its effectiveness, we conduct extensive experiments on four multi-modality FL benchmarks with different types of data heterogeneity, where FedDAT substantially outperforms the existing centralized PEFT methods adapted for FL.

Introduction

Recent works have shown the power of foundation models with millions (billions) of parameters (Zhou et al. 2023; Du et al. 2022). These models, represented by Transfomers (Vaswani et al. 2017), achieve promising results when finetuned for real-world multi-modal tasks, including Visual Question Answering (VQA) (Antol et al. 2015), Visual Commonsense Reasoning (VCR) (Zellers et al. 2019), etc. To improve the generalization ability of the foundation models, a substantial amount of data from diverse sectors and application scenarios is typically required for extensive finetuning. However, it becomes challenging to aggregate all training data and perform centralized model finetuning. For instance, collecting data from different clinical centers across multiple countries becomes infeasible due to distinct privacy regulations, such as GDPR in the EU and PDPA in Singapore.

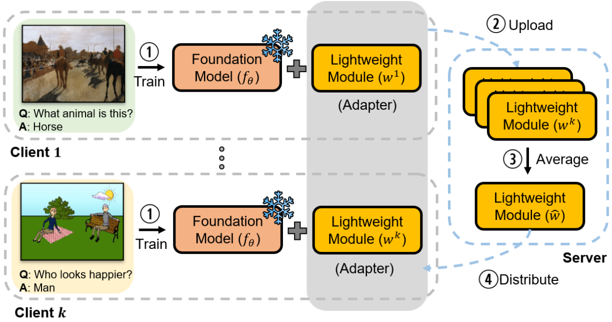

Q: What animal is this?
A: Horse
Client 1
① Train
Foundation Model (f_θ)
Lightweight Module (w^1)
(Adapter)
② Upload
Lightweight Module (w^k)
③ Average
Lightweight Module ($\bar{w}$)
Server
④ Distribute
Q: Who looks happier?
A: Man
① Train
Foundation Model (f_θ)
Lightweight Module (w^k)
(Adapter)
Client k

Figure 1: Schematic illustration of the training procedure for Visual Question Answering (VQA) in Federated Learning.

To address this problem, Federated Learning (FL) emerges as a promising solution, which allows a shared model to be collaboratively optimized using decentralized data sources. In the classical FL approaches, e.g., FedAvg (McMahan et al. 2017), the central server obtains the model by iteratively averaging the optimized model weights uploaded from the active clients. FL offers several advantages, including improved efficiency in client-server communication and enhanced data confidentiality, as it eliminates the need for direct access to the client's local dataset. FL provides promising solutions for various application areas, such as healthcare (Sheller et al. 2020) and industry (Liu et al. 2020), where data privacy is crucial.

Despite its promising prospects, traditional FL is unsuitable for finetuning the entire foundation model. The optimization and transmission of billions of parameters would impose significant client computation burdens and substantial communication overheads. To overcome this challenge,

[*] Corresponding author

parameter-efficient finetuning (PEFT) methods provide a possible solution, where only a small fraction of the model parameters is optimized and communicated during FL.

Existing works have predominantly explored a basic combination of centralized PEFT algorithms and FedAvg. For instance, some approaches focus on training and communicating only the tiny adaptation modules (adapter) (Houlsby et al. 2019; Su et al. 2022) or a small amount of trainable input tokens (Guo et al. 2022; Guo, Guo, and Wang 2023). However, these investigations are limited to single modality scenarios, where only visual or textual tasks are considered. Most importantly, none of these works address the problem of data heterogeneity, in which the data of different clients are not independent and identically distributed (*non-IID*). Data heterogeneity may lead to model drifts during the client local update, as well as an unstable and sub-optimal convergence of the aggregated server model (Li et al. 2020a; Mendieta et al. 2022). Therefore, in this paper, we propose Federated Dual-Adapter Teacher (`FedDAT`), as the first framework to address this challenging yet practical problem, PEFT of foundation models for multi-modal (Vision-Language) heterogeneous FL.

`FedDAT` incorporates a global adapter in the foundation model, which is optimized and transmitted during federated communications. `FedDAT` utilizes a Dual-Adapter Teacher (*DAT*) module, comprising two parallel adapters: one is a copy of the global adapter, kept frozen, while the other is locally optimized at each client. This configuration enables the local adapter to capture client-specific knowledge, which serves to regularize the global adapter and address data heterogeneity. Meanwhile, the frozen adapter preserves client-agnostic knowledge, thereby mitigating the catastrophic forgetting of the global adapter during knowledge transfer. To prevent overfitting of *DAT* to the limited client local dataset, we implement Mutual Knowledge Distillation (*MKD*) between *DAT* and the global adapter. This mechanism ensures efficient knowledge transfer while maintaining the generalization ability of both modules.

The proposed method `FedDAT` achieves state-of-the-art results on four multi-modality benchmarks that include a variety of Vision-Language (VL) tasks with data heterogeneity. Our contributions can be summarized as follows:

- We propose a novel method `FedDAT` for multi-modal heterogeneous FL, which is the first FL framework addressing distributed PEFT of foundation models for Vision-Language tasks.
- We conduct comprehensive experiments on four heterogeneous FL benchmarks with a variety of Vision-Language tasks. The results demonstrate that `FedDAT` achieves SOTA results, indicating better convergence rate and scalability compared to existing PEFT methods.

Related Work

Parameter-Efficient Finetuning (PEFT) for Federated Learning. PEFT has been well studied in centralized machine learning (Houlsby et al. 2019; Liu et al. 2022; Sung, Cho, and Bansal 2022), while its application on FL remains under-explored. Most of the prior work rudimentarily adapted PEFT for FL and focused on single-modal tasks:

(1) Image classification. (Chen et al. 2022; Sun et al. 2022) evaluate the existing PEFT baselines combined with FL, while (Guo et al. 2022; Guo, Guo, and Wang 2023; Li et al. 2023; Lu et al. 2023) finetune the CLIP model (Radford et al. 2021) via tuning and communicating only small amount of learnable (personalized) prompts. (Su et al. 2022) addresses the problem of heterogeneous client images by injecting lightweight adaptation modules (adapters) (Houlsby et al. 2019). (Yang et al. 2023) explores the possibility of finetuning generative foundation models (diffusion models) (Dhariwal and Nichol 2021) via FL.

(2) Language tasks. (Yu, Muñoz, and Jannesari 2023) requires public server dataset and optimize adapter for few-shot finetuning of BERT-like language models (Devlin et al. 2018). (Zhang et al. 2023) builds a distributed instruction tuning (Wei et al. 2021) datasets and finetunes the language model via Low-Rank Adaptation (LoRA) (Hu et al. 2021). (Zhuang, Chen, and Lyu 2023) systematically analyzes the challenges of finetuning large language models in FL.

(Yu et al. 2023) is the first to analyze the situation of having multi-modal client datasets and conducts contrastive representation learning. However, the visual data and the language data are processed by separate networks, i.e., no Vision-Language Foundation Model is involved. In this work, we focus on the under-explored PEFT for large-scale vision-language models in FL and address the problem of client local datasets with heterogeneity in both vision and/or language modality.

Vision-Language Foundation Model. Vision-Language foundation models have significantly advanced the Vision-Language tasks (Antol et al. 2015; Zellers et al. 2019; Suhr et al. 2019; Xie et al. 2019a). Based on the perspective of intra-modality data handling, there are two types of mainstream Vision-Language Foundation model structures: (1) Single-stream Vision-Language Foundation models (Li et al. 2019; Chen et al. 2020; Li et al. 2020b; Su et al. 2020; Kim, Son, and Kim 2021a; Singh et al. 2022), which directly fuse the initial language/visual representation by using the joint cross-modal encoder at the initial state, and (2) Dual-stream Vision-Language foundation models (Lu et al. 2019; Tan and Bansal 2019; Li et al. 2021b; Huo et al. 2021), which separately apply the intra-modality processing to two modalities along with a shared cross-modal encoder. To showcase the applicability of our proposed `FedDAT` to a wide range of Vision-Language foundation models, we carefully select ViLT (Kim, Son, and Kim 2021a) as a representative single-stream Vision-Language foundation model, and ALBEF (Li et al. 2021b) as a representative dual-stream Vision-Language foundation model. By employing these diverse models, we effectively demonstrate the versatility and robustness of `FedDAT` in Vision-Language learning.

Methodology

Problem Statement

In this work, we address a heterogeneous FL problem setting with K clients: Each client k owns its private multi-modal dataset D^k, containing data from visual modality (images) and textual modality (texts). Specifically, we focus on the vision-language tasks and take Visual Question Answering (VQA) as an example. Hereby, the local dataset D^k can be further decomposed into N_k image-question-answer triplets $\{(v_i^k, q_i^k, a_i^k)|i \in \{1, ..., N_k\}\}$. We assume that the marginal distribution of v_i^k and/or q_i^k, a_i^k varies across the clients, i.e., there exists data heterogeneity in the visual space and/or in the textual space. We define the answer pool $A^k = \{a_1^k, ..., a_{C^k}^k\}$ with C^k ground-truth answers for client k and define our task as a C^k-way classification problem following (Antol et al. 2015). Note that the size of the answer pool could be different for different clients. The objective of FL is to collaboratively finetune one *global* foundation model f_θ in a parameter-efficient manner (PEFT) within a pre-defined communication budget, which produces promising results on all client's local data.

PEFT Method: Adapter

In this section, we introduce a traditional parameter-efficient finetuning (PEFT) method, i.e., Adapter (Houlsby et al. 2019), adjusted for FL applications. Here, we adopt the foundation models with common Transformer architecture (Vaswani et al. 2017) consisting of multiple repeated Transformer blocks. Specifically, each block contains a self-attention sub-layer, a fully connected feed-forward network (FFN), and residual connections around the sub-layers followed by layer normalization.

Adapter is a bottleneck network consisting of a down-sample linear layer $W_{down} \in \mathbb{R}^{d\times r}$ and an up-sampling linear layer $W_{up} \in \mathbb{R}^{r\times d}$, where r denotes the down-sampled dimension ($r < d$). A nonlinear activation function $\phi(\cdot)$, such as ReLU, is inserted in between. The adapter is injected after the FFN of each Transformer block and its computation can be formulated as

$$h' = h + \phi(hW_{down})W_{up}, \tag{1}$$

where h is the normalized output of FFN.

Recap: Federated Averaging

In this section, we formally describe the combination of the conventional federated learning algorithm, FedAvg (McMahan et al. 2017), and the centralized PEFT algorithm, i.e., Adapter. Before the client-server communication starts, we deploy the *same* pre-trained foundation model f_θ at different clients. Afterwards, the server randomly initializes the parameter w of the learnable lightweight module, which are the weight matrices of the linear layers W_{down} and W_{up} in the adapters. w is then distributed to all clients for communication and local optimization. We illustrate the procedure of one communication round in the following.

Method	DomainNet						
	C	I	P	Q	R	S	avg
clf-L	72.43	36.13	86.35	55.70	74.07	74.70	66.56
$Adapter$-L	76.05	36.93	88.03	72.40	66.53	78.74	69.78
clf	80.80	44.61	83.47	60.10	84.21	71.69	70.81
$Adapter$	88.59	50.95	87.12	76.00	84.99	74.08	**76.96**

Table 1: Evaluation results of ViT finetuned for DomainNet with/without FL. "L" indicates independent client training, i.e., no federated communication involved.

As shown in Figure 1, each active client k first execute local training to optimize the light-wegiht module w^k combined with the *frozen* foundation model f_θ (①) in parallel, where the following loss L_k is minimized:

$$L_k(w^k) = \frac{1}{N_k}\sum_{i=1}^{N_k}\mathcal{L}(y_i, f_{\theta\cup w^k}(x_i)), \tag{2}$$

where y_i is the ground-truth label of input data x_i, and $\mathcal{L}$ is the loss function, e.g., Cross-Entropy for classification tasks. After the local updates, the central server aggregates $\{w^k|1 \le k \le K\}$, uploaded (②) by all active clients, and executes a parameter aggregation (③):

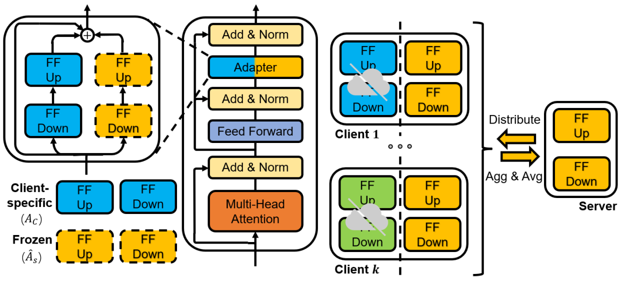
$$\hat{w} \leftarrow \frac{1}{\sum_{k=1}^{K} N_k}\sum_{k=1}^{K} N_k \cdot w^k. \tag{3}$$

Finally, the aggregated weight $\hat{w}$ will be distributed (④) to the active clients for optimization in the next communication round. Note that after exhausting all communication budgets, the global model $f_{\theta\cup w}$ is deployed for the testing.

Motivational Case Study

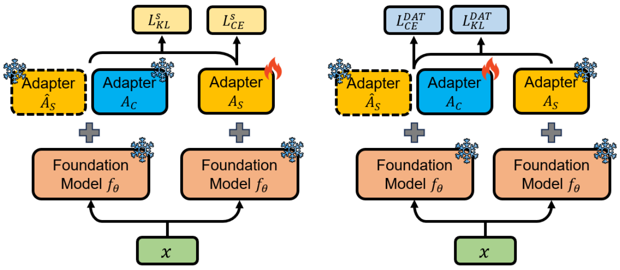
To motivate the architecture design of `FedDAT`, we present an empirical analysis to address the following research question: *Which type of knowledge is more crucial for optimizing a promising ML model in heterogeneous FL, client-specific or client-agnostic?* Therefore, we follow the experiment design proposed in (Tan et al. 2022). Specifically, we take the down-sampled version of DomainNet (Peng et al. 2019), which is an image classification benchmark and contains data from 6 different styles: Clipart (C), Infograph (I), Painting (P), Quickdraw (Q), Real (R), and Sketch (S). By assigning data from one style to each client, we simulate data heterogeneity in the feature space across different clients. We finetune the foundation model, i.e., ViT (Dosovitskiy et al. 2020), with different PEFT methods via FL.

In Table 1, we provide the results of finetuning the classification head (clf) and finetuning with $Adapter$. We also display the performance of client local finetuning (L), i.e., no federated communication involved. We conclude three observations from the results: (1) $Adapter$ is an effective PEFT method in both federated setting and independent finetuning setting compared with clf, providing an average performance increase of 3.22% and 6.15%, respectively. (2) Collaborative training via FL, i.e., finetuning a *client-agnostic* foundation model, generally outperforms local independent finetuning. This can be observed by comparing the average accuracy of models with and without "L". (3) *Client-specific* classification head and adapters show benefits on certain clients (marked with underlines), i.e., clients with Painting (P) and Sketch (S) data and optimized independently. We assume this is due to the large distribution shift in the feature space across different clients' local data, given their different image appearances. This phenomenon



(a) *DAT* Architecture Design (b) Communication

Figure 2: Schematic illustration of the Dual-Adapter Teacher (*DAT*) with local A_c and frozen $\hat{A}_s$. Only the shared adapter A_s is transmitted during federated communication.



(a) Optimization of A_s (b) Optimization of *DAT*

Figure 3: Schematic illustration of the Mutual Knowledge Distillation (*MKD*) between *DAT* and A_s.

answers the previous research question: Both *client-specific* and *client-agnostic* knowledge are crucial and should not be forgotten during federated communication. These observations motivate the proposed method and serve as evidence for its promising applicability and effectiveness.

Proposed Method

In this section, we introduce the proposed method Federated Dual-Adapter Teacher (`FedDAT`). As shown in Algorithm 1, the training process of `FedDAT` can be divided into two functions, which will be introduced in the following:

At the beginning of the training, the server initializes a shared adapter A_s. In each communication round, all active clients receive A_s and conduct $\mathit{Client\ Update}$ in parallel. Subsequently, the server aggregates and averages the optimized parameters $\{A_s^k | 1 \leq k \leq K\}$ uploaded from all clients, which will be used as the initialization of A_s for the next communication round.

The client local update comprises 2 main components, which will be introduced in the following:

(1) Dual-Adapter Teacher (*DAT*). Before the first communication round, each client locally initializes the local adapter A_c as well as the foundation model f_θ with the same pre-trained weights θ. Subsequently, each client receives the parameters of A_s from the server, which is then copied as $\hat{A}_s$ and kept frozen during the client local update. We combine $\hat{A}_s$ and A_c as the Dual-Adapter Teacher (*DAT*) and provide its schematic illustration in Figure 2a.

In *DAT*, we constrain the parameters of A_c strictly local for each client. By personalizing A_c, we force it to focus solely on client-specific knowledge, which is crucial for client data heterogeneity. Meanwhile, the frozen $\hat{A}_s$ is utilized to retain the client-agnostic knowledge captured by the shared adapter A_s. Similar to traditional adapters (Equation 1), given the normalized output of FFN h in a Transformer layer, *DAT* performs the following transformation:

$$h' \leftarrow h + \frac{1}{2}\phi(h \cdot \hat{W}_s^{down}) \cdot \hat{W}_s^{up} + \frac{1}{2}\phi(h \cdot W_c^{down}) \cdot W_c^{up}, \tag{4}$$

where $\hat{W}_s$ and W_c are the weight matrices for $\hat{A}_s$ and A_c, respectively. Afterwards, T local update steps will be executed, in which the shared adapter A_s and the *DAT* module is optimized.

By utilizing *DAT* as a guidance for the local optimization of A_s at each client, our goal is to distill client-specific knowledge into A_s and mitigate the forgetting of A_s on its client-agnostic knowledge. Hereby, we apply Mutual Knowledge Distillation (*MKD*) for an efficient knowledge transfer, which will be introduced in the following.

(2) Mutual Knowledge Distillation (*MKD*). A schematic illustration of *MKD* is provided in Figure 3. *MKD* executes bi-directional knowledge distillation between A_s and *DAT* via L_{KL}^{s} and $L_{\text{KL}}^{\text{DAT}}$, respectively:

$$L_{\text{KL}}^{s} = \mathcal{KL}(z_s(x) || z_{\text{DAT}}(x)), \quad L_{\text{KL}}^{\text{DAT}} = \mathcal{KL}(z_{\text{DAT}}(x) || z_s(x)), \tag{5}$$

where $\mathcal{KL}$ denotes the Kullback-Leibler divergence, z_s and z_{DAT} are the predicted logits of the foundation model injected with A_s and *DAT*, respectively. Hereby, this setup allows the shared adapter A_s to capture both client-specific knowledge and client-agnostic stored in *DAT* (L_{KL}^{s}). Additionally, we apply A_s as guidance for the optimization *DAT* ($L_{\text{KL}}^{\text{DAT}}$) to prevent possible overfitting, considering the scarce local data of each client (McMahan et al. 2017).

MKD is utilized together with the guidance from ground-truth labels of the training data, i.e.,

$$\begin{aligned} L_{\text{CE}}^{s} &= \sum_{c=1}^{C} \mathcal{I}(x,c) \cdot log(\sigma(z_s(x))^{(c)}), \\ L_{\text{CE}}^{\text{DAT}} &= \sum_{c=1}^{C} \mathcal{I}(x,c) \cdot log(\sigma(z_{\text{DAT}}(x))^{(c)}), \end{aligned} \tag{6}$$

where, $\mathcal{I}(x,c)$ is a binary indicator (0 or 1) if c is the ground-truth label for x, σ is the softmax function. Hereby, we aim at training the foundation model, injected with either A_s or *DAT*, to correctly classify the training sample x. Finally, combining *MKD* and L_{CE} produces the optimization objective for A_s and *DAT*:

$$\begin{aligned} L^{s} &= L_{\text{CE}}^{s} + \alpha L_{\text{KL}}^{s}, \\ L^{\text{DAT}} &= L_{\text{CE}}^{\text{DAT}} + \beta L_{\text{KL}}^{\text{DAT}}, \end{aligned} \tag{7}$$

where, α and β are the weighting coefficient. While both *DAT* and A_s are randomly initialized, they become more informative as the training progresses. To reflect this observation, we apply an exponential ramp-up schedule for α and β. Despite the sophisticated design of our method, FedDAT indicates the same inference cost and communication overhead as the PEFT method *Adapter*, where only A_s is transmitted and applied at deployment.

Experiments and Analyses

We conduct extensive empirical analyses to investigate the proposed method. Firstly, we compare FedDAT with other centralized PEFT methods on four heterogeneous FL benchmarks containing different Vision-Language tasks. Afterwards, we demonstrate the effectiveness of FedDAT components via ablation study. Finally, we analyze the promising convergence rate and scalability of FedDAT.

Benchmark Experiments

Datasets Description. We conduct experiments on different Vision-Language (VL) benchmarks with different types of data heterogeneity, including visual, textual, and task heterogeneity. We introduce these benchmarks in the following.

- **Domain**. We adopt 5 common VQA datasets from different domains, i.e., VizWiz (Gurari et al. 2018), COCO QA (Ren, Kiros, and Zemel 2015), Art (Garcia et al. 2020), GQA (Hudson and Manning 2019) and Abstract (Antol et al. 2015). We assign one of the datasets to each client, leading to heterogeneity in both vision and language modality. Example VQA triplets from the benchmark are provided in Figure 4.
- **Function** & **Scene**. We adopt and split the CLOVE benchmark (Lei et al. 2023) into *Scene* and *Function* benchmark, which contains VQA triplets collected from 6 different visual environments and 5 different functions, respectively. Triplets from one scene (function) are allocated to one client, resulting in visual (textual) heterogeneity in the *Scene* (*Function*) benchmark.
- **Task**. We adopt and modify the CLiMB benchmark (Srinivasan et al. 2022), which contains 4 VL tasks, namely VQA (Antol et al. 2015), Natural Language for Visual Reasoning (*NLVR*) (Suhr et al. 2018), Visual Entailment (*VE*) (Xie et al. 2019b), and Visual Commonsense Reasoning (*VCR*) (Zellers et al. 2019). Each client owns data from one of the datasets, introducing task heterogeneity across different clients.

We downsample the original dataset to simulate client local data scarcity described in prior arts (McMahan et al. 2017) and provide more details in the Appendix.

Algorithm 1: Training procedure of FedDAT

```
ServerUpdate
Randomly initialize $A_s$
for round $r$ = 1 to $R$ do
    for client $k$ = 1 to $K$ do {in parallel}
        $A_s^k \leftarrow$ ClientUpdate$(A_s, k, r)$
    end for
    $A_s \leftarrow \frac{1}{K}\sum_{k=1}^{K} A_s^k$
end for
ClientUpdate$(A_s, k, r)$
if $r = 1$ then
    Randomly initialize $A_c$
end if
$\hat{A}_s \leftarrow A_s$
for local step $t$ = 1 to $T$ do
    Sample $\{\boldsymbol{X}, \boldsymbol{y}\}$ from $D_k$
    Optimize $A_s$ via minimizing $L^s$
    Optimize $DAT$ via minimizing $L^{\text{DAT}}$
end for
return $A_s$
```

Implementation Details. For the task-heterogeneous benchmark (*Task*), we adopt the Transformer encoder-only backbones following (Srinivasan et al. 2022), i.e., ViLT (Kim, Son, and Kim 2021b) and VAuLT (Chochlakis et al. 2022). For the rest three benchmarks, we add another encoder-decoder backbone, i.e., ALBEF (Li et al. 2021a). We compare FedDAT with various centralized PEFT methods adapted for FL, including *LoRA* (Hu et al. 2021), *prompt*-tuning (Guo et al. 2022), and *bias*-tuning (Cai et al. 2020). We also provide results of independent client optimization (marked by "L") of the classification head clf and *Adapter*. Moreover, we provide the results of fully finetuning the models ($full$) as an *oracle* method (marked by $*$), given the infeasibility of transmitting the entire foundation model in FL.

To handle the different answer pools in different clients, we incorporate client-specific classification heads for ViLT and VAuLT, and apply client-specific answer lists for ALBEF. To make a fair comparison between different centralized PEFT algorithms and FedDAT, we apply the same hyperparameters search for all methods in different benchmarks. All experiments are repeated with 3 random seeds. The hyperparameters are detailed in the Appendix.

Results and Analyses. In Table 2, we provide the results of FedDAT and the other FL-adapted PEFT methods on our *Domain* benchmark. We observe FedDAT outperforms all the baselines with all the architectures, achieving an average performance improvement of up to 4.55% compared with the most promising baseline *Adapter*. This indicates the easy adaptability of FedDAT for both encoder-based and encoder-decoder-based VL models. Moreover, FedDAT depicts the same communication overhead as a single *Adapter*, which adds and optimizes only less than

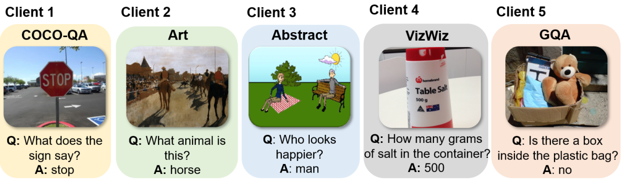


Figure 4: Example VQA triplets of different datasets in *Domain* benchmark with heterogeneity in both Vision and Language modality.

Backbone	Method	Comm. Overhead	VizWiz	COCO	Art	GQA	Abstract	Average
ViLT	clf-L	–	63.13±1.07	36.15±2.92	63.22±0.99	34.90±3.16	52.81±2.67	50.04±1.81
	$LoRA$	$0.60M(0.48\%)$	60.47±1.25	43.28±1.37	62.98±0.75	36.57±2.01	52.04±1.62	51.07±1.41
	$prompt$	$0.60M(0.48\%)$	60.13±1.05	52.13±0.87	63.02±1.58	39.09±0.37	52.88±3.07	53.45±2.04
	$bias$	$0.10M(0.08\%)$	61.83±2.41	49.41±2.36	69.38±1.69	40.43±0.66	60.36±1.92	56.28±1.97
	$Adapter$-L	–	61.72±1.42	46.27±4.58	67.69±0.42	43.62±0.93	54.02±2.16	54.67±2.54
	$Adapter$	$0.89M(0.75\%)$	61.39±1.11	52.39±6.20	68.72±3.20	43.72±0.65	59.43±2.94	57.13±4.08
	FedDAT	$0.89M(0.75\%)$	60.99±2.81	63.81±2.90	71.36±3.34	48.65±2.93	60.75±2.67	**61.11**±2.98
	$full$-L*	–	55.52±1.42	72.97±1.53	73.16±0.28	44.41±3.98	58.78±0.25	60.97±1.45
	$full^*$	$87.40M(100\%)$	56.12±2.55	73.87±0.83	76.24±1.82	50.28±1.59	61.26±0.78	63.55±1.35
VAuLT	clf-L	–	61.83±1.85	32.42±0.04	64.52±1.55	35.08±5.57	48.48±0.77	48.46±1.15
	$LoRA$	$0.60M(0.29\%)$	62.17±1.32	40.56±0.86	63.08±1.13	33.47±3.08	47.34±1.04	49.32±1.16
	$prompt$	$0.60M(0.29\%)$	62.93±0.87	46.52±1.45	64.26±1.03	35.33±2.12	48.91±0.68	51.59±1.63
	$bias$	$0.21M(0.10\%)$	61.12±2.84	43.81±0.35	67.00±1.41	33.30±4.81	51.22±2.07	51.29±1.08
	$Adapter$-L	–	62.33±1.42	47.72±2.83	67.50±2.11	33.75±2.79	54.09±0.93	53.07±1.34
	$Adapter$	$1.79M(0.77\%)$	52.53±3.65	53.63±0.28	66.80±0.53	35.65±1.84	50.03±1.77	51.73±0.46
	FedDAT	$1.79M(0.77\%)$	62.19±1.01	54.83±2.04	67.86±1.93	40.06±3.08	54.48±0.49	**55.88**±1.79
	$full$-L*	–	57.41±2.13	55.68±1.24	70.27±2.11	41.31±1.46	52.66±0.57	55.47±1.85
	$full^*$	$227.77M(100\%)$	45.79±2.12	64.64±3.05	67.89±1.82	41.93±3.85	49.58±0.66	53.97±2.09
ALBEF	$LoRA$	$1.52M(0.53\%)$	60.49±1.32	28.32±0.65	57.04±3.69	28.71±0.42	58.06±2.42	46.52±1.75
	$prompt$	$0.92M(0.32\%)$	63.13±0.65	32.50±1.20	63.45±0.42	32.08±1.07	59.45±1.78	50.12±0.95
	$bias$	$0.93M(0.32\%)$	63.23±0.14	31.23±0.28	61.23±1.12	35.93±1.73	57.88±0.28	49.90±0.87
	$Adapter$-L	–	61.72±1.12	56.32±1.50	65.21±0.35	40.96±2.27	59.51±1.58	56.74±1.38
	$Adapter$	$2.86M(0.98\%)$	59.52±2.44	69.35±2.78	68.32±0.89	41.02±3.12	60.83±2.66	59.81±1.87
	FedDAT	$2.86M(0.98\%)$	61.52±1.51	76.36±0.63	71.04±0.50	49.22±1.60	63.65±1.19	**64.36**±1.39
	$full$-L*	–	61.22±0.14	77.80±1.39	74.45±0.7	50.09±1.06	63.58±2.79	65.43±1.37
	$full^*$	$290.34M(100\%)$	51.91±1.42	78.38±1.11	75.65±0.14	55.91±0.54	70.47±0.83	66.46±0.96

Table 2: Evaluation results of different finetuning methods on our FL benchmark with distribution shift in both Vision and Language space. ”L” indicates client local finetuning where no communication is involved. We report the mean±std accuracy of each client from 3 runs with different seeds.

1% of the total parameters in the foundation model. This further illustrates its applicability to the FL system with constrained communication bandwidths. Besides, FedDAT narrows the performance gap between the PEFT methods and fully-finetuning methods. Interestingly, our approach outperforms the oracle methods $full$-L when applied on ViLT and VAuLT, which demonstrates the effectiveness of introducing client-specific knowledge into the client local optimization. We also note that applying $Adapter$-L for VAuLT, i.e., optimizing adapters for each client independently, achieves better results than $Adapter$, which provides additional evidence for our observation in Section .

Afterwards, we provide the comparison of clients’ average accuracy between FedDAT and different PEFT methods on the other benchmarks. As shown in Table 3, FedDAT provides promising improvements of up to 6.02%, 7.94% and 1.09% on *Function, Scene*, and *Task* benchmark, respectively. More details of the client specific performance are provided in the Appendix.

Ablation Study

To illustrate the importance of different components used in FedDAT, we conduct an ablation study for ViLT on three benchmarks. The results are shown in Table 4. We first investigate the optimization process, where we notice that optimizing without *DAT*, i.e., applying solely the local adapter A_c or the frozen adapter $\hat{A}_s$ as the teacher, leads to only minimal performance increase, which indicates the effectiveness of our Dual-Adapter Teacher design. Besides, distilling only the knowledge from *DAT* to the shared adapter A_s, i.e., omitting the bi-directional *MKD*, brings visible performance gain. Combining both strategies achieves the best results, which further demonstrates their complementarity. Additionally, we validate other inference choices. Specifically, we evaluate the final *DAT* module (combination of A_c and $\hat{A}_s$) and the local adapter A_c at each client. We again note that we are addressing the problem of finetuning a global foundation model via FL, where no further personalization is required. Considering the inference efficiency and the problem setting, we adopt the shared adapter A_s for inference, which also achieves the most promising results.

Convergence Analysis

In Figure 5, we display the convergence analysis of FedDAT compared with the most promising PEFT method *Adapter* on *Domain* benchmark. Hereby, we report the accuracy of the clients on their corresponding local testing set after each communication round. As shown in the figure, even though FedDAT utilizes a more sophisticated optimization schema, i.e., a combination of *DAT* and *MKD*, the learning curves of FedDAT still exhibit faster convergence rates than single $Adapter$. It is also worth noticing that FedDAT already achieves distinct performance gain after 5 communication rounds, i.e., 25% of the total communication budgets.

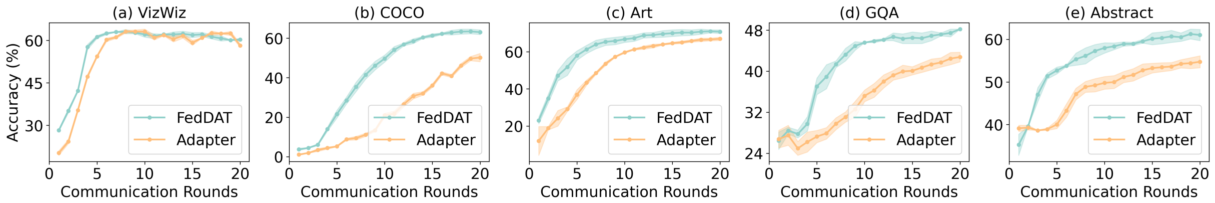


Figure 5: Convergence analysis of ViLT model on different clients in *Domain* benchmark.

Backbone	Method	Function	Scene	Task
ViLT	*clf*-L	31.58±1.97	24.52±0.95	49.46±0.39
	LoRA	32.04±1.12	28.47±1.03	47.82±1.42
	prompt	40.53±1.56	30.53±1.30	49.55±1.14
	bias	43.81±1.39	33.65±1.87	50.71±1.26
	Adapter-L	39.68±2.19	31.91±2.05	49.59±1.74
	Adapter	48.37±1.56	31.07±1.08	51.44±1.34
	FedDAT	**54.39**±2.36	**39.35**±1.25	**52.37**±0.52
	full-L*	56.81±2.97	38.00±1.48	50.64±1.42
	*full**	59.62±2.56	40.62±3.76	53.17±0.69
VAuLT	*clf*-L	27.72±3.05	21.22±2.08	39.63±1.07
	LoRA	29.87±1.86	23.08±1.09	38.35±1.47
	prompt	36.32±2.07	25.63±1.54	38.75±1.34
	bias	36.11±3.05	24.89±2.17	39.46±0.99
	Adapter-L	37.22±2.38	28.57±1.98	40.42±1.21
	Adapter	41.50±3.24	29.39±2.65	40.19±0.89
	FedDAT	**44.54**±2.08	**34.31**±2.87	**41.28**±0.57
	full-L*	49.13±2.68	35.11±1.99	41.66±1.32
	*full**	46.38±1.57	36.72±2.57	42.44±0.71

Table 3: Evaluation results of different methods on *Function*, *Scene*, and *Task* benchmark. "L" indicates independent client finetuning. We report the mean±std accuracy of 3 trials.

Scalability Analysis of FedDAT

To show the effectiveness of FedDAT under various application scenarios, we further conduct experiments with different numbers of clients. More specifically, we split the data of each function in the original CLOVE dataset (Lei et al. 2023) into 5 subsets, where each subset has an equal number of training data and is assigned to one client, following the client data scarcity described in (McMahan et al. 2017). We conduct experiments where 1, 2, 3, 4, and 5 clients (subsets) from each function are selected, which gives in total 5, 10, 15, 20, and 25 clients joining the federated communication for the *Function* benchmark, respectively. We apply also the same split strategy for the 6 different visual environments for the *Scene* benchmark and conduct the same experiment. More details regarding the experimental setups are provided in Appendix.

We observe that FedDAT consistently outperforms *Adapter* across all setups with small or large quantities of training data. Notably, a performance gap of up to 10% for ALBEF and 6% for ViLT is evident. These results indicate the scalability of FedDAT in handling complex FL applications involving a larger number of clients and increased communication budgets.

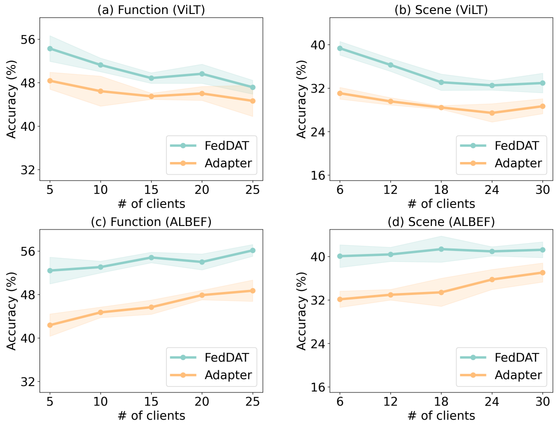


Figure 6: Scalability analysis of FedDAT with different number of clients on *Funciton* and *Scene* benchmarks.

Conclusion

In this work, we propose the first FL framework to address the parameter-efficient finetuning (PEFT) of the foundation model in heterogeneous FL, where various Vision-Language tasks are investigated. The proposed method, named FedDAT, optimizes a shared adapter utilizing the Dual-Adapter Teacher (*DAT*) and Mutual Knowledge Distillation (*MKD*). Compared with existing centralized PEFT methods, FedDAT achieves promising results on the four FL benchmarks with various Vision-Language tasks, demonstrating its effectiveness. Additional experiments indicate its applicability to complex FL setups involving larger distributed systems and training budgets.

Stage	Method	Domain	Function	Scene
-	*Adapter*	57.13±4.08	48.37±1.56	31.07±1.08
Optimization	w/o $\hat{A}_s$	58.24±0.98	50.62±1.45	33.04±0.65
	w/o A_c	57.87±1.24	50.93±0.85	32.45±0.27
	w/o *MKD*	58.41±1.57	52.82±2.98	36.98±1.07
	FedDAT	**61.11**±2.98	**54.39**±2.36	**39.35**±1.25
Inference	$A_c + A_s$	58.45±1.57	50.42±1.87	35.61±2.41
	A_c	55.87±3.35	46.14±2.60	32.84±0.78
	A_s (**FedDAT**)	**61.11**±2.98	**54.39**±2.36	**39.35**±1.25

Table 4: Ablation study for different components in optimization and inference stage of FedDAT on three benchmark datasets.

References

[3] Shen, Y.; and et al. 2022. Cd2-pfed: Cyclic distillation-guided channel decoupling for model personalization in federated learning. In *CVPR*.

Antol, S.; Agrawal, A.; Lu, J.; Mitchell, M.; Batra, D.; Zitnick, C. L.; and Parikh, D. 2015. VQA: Visual Question Answering. In *International Conference on Computer Vision (ICCV)*.

Cai, H.; Gan, C.; Zhu, L.; and Han, S. 2020. Tinytl: Reduce memory, not parameters for efficient on-device learning. *Advances in Neural Information Processing Systems*, 33: 11285–11297.

Chen, J.; Xu, W.; Guo, S.; Wang, J.; Zhang, J.; and Wang, H. 2022. FedTune: A Deep Dive into Efficient Federated Fine-Tuning with Pre-trained Transformers. *arXiv preprint arXiv:2211.08025*.

Chen, S.; Gu, J.; Han, Z.; Ma, Y.; Torr, P.; and Tresp, V. 2023. Benchmarking Robustness of Adaptation Methods on Pre-trained Vision-Language Models. *arXiv preprint arXiv:2306.02080*.

Chen, Y.; Li, L.; Yu, L.; Kholy, A. E.; Ahmed, F.; Gan, Z.; Cheng, Y.; and Liu, J. 2020. UNITER: UNiversal Image-TExt Representation Learning. In Vedaldi, A.; Bischof, H.; Brox, T.; and Frahm, J., eds., *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXX*, volume 12375 of *Lecture Notes in Computer Science*, 104–120. Springer.

Chochlakis, G.; Srinivasan, T.; Thomason, J.; and Narayanan, S. 2022. Vault: Augmenting the vision-and-language transformer with the propagation of deep language representations. *arXiv preprint arXiv:2208.09021*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dhariwal, P.; and Nichol, A. 2021. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34: 8780–8794.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Du, Y.; Liu, Z.; Li, J.; and Zhao, W. X. 2022. A survey of vision-language pre-trained models. *arXiv preprint arXiv:2202.10936*.

Garcia, N.; Ye, C.; Liu, Z.; Hu, Q.; Otani, M.; Chu, C.; Nakashima, Y.; and Mitamura, T. 2020. A Dataset and Baselines for Visual Question Answering on Art. In *Proceedings of the European Conference in Computer Vision Workshops*.

Guo, T.; Guo, S.; and Wang, J. 2023. pFedPrompt: Learning Personalized Prompt for Vision-Language Models in Federated Learning. In *Proceedings of the ACM Web Conference 2023*, 1364–1374.

Guo, T.; Guo, S.; Wang, J.; and Xu, W. 2022. PromptFL: Let Federated Participants Cooperatively Learn Prompts Instead of Models–Federated Learning in Age of Foundation Model. *arXiv preprint arXiv:2208.11625*.

Gurari, D.; Li, Q.; Stangl, A. J.; Guo, A.; Lin, C.; Grauman, K.; Luo, J.; and Bigham, J. P. 2018. VizWiz Grand Challenge: Answering Visual Questions from Blind People. *CVPR*.

Houlsby, N.; Giurgiu, A.; Jastrzebski, S.; Morrone, B.; De Laroussilhe, Q.; Gesmundo, A.; Attariyan, M.; and Gelly, S. 2019. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning*, 2790–2799. PMLR.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Hudson, D. A.; and Manning, C. D. 2019. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 6700–6709.

Huo, Y.; Zhang, M.; Liu, G.; Lu, H.; Gao, Y.; Yang, G.; Wen, J.; Zhang, H.; Xu, B.; Zheng, W.; et al. 2021. WenLan: Bridging vision and language by large-scale multi-modal pre-training. *arXiv preprint arXiv:2103.06561*.

Kim, W.; Son, B.; and Kim, I. 2021a. ViLT: Vision-and-Language Transformer Without Convolution or Region Supervision. In Meila, M.; and Zhang, T., eds., *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, 5583–5594. PMLR.

Kim, W.; Son, B.; and Kim, I. 2021b. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, 5583–5594. PMLR.

Le, H. Q.; Nguyen, M. N.; Thwal, C. M.; Qiao, Y.; Zhang, C.; and Hong, C. S. 2023. FedMEKT: Distillation-based Embedding Knowledge Transfer for Multimodal Federated Learning. *arXiv preprint arXiv:2307.13214*.

Lei, S. W.; Gao, D.; Wu, J. Z.; Wang, Y.; Liu, W.; Zhang, M.; and Shou, M. Z. 2023. Symbolic replay: Scene graph as prompt for continual learning on vqa task. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 1250–1259.

Li, G.; Wu, W.; Sun, Y.; Shen, L.; Wu, B.; and Tao, D. 2023. Visual Prompt Based Personalized Federated Learning. *arXiv preprint arXiv:2303.08678*.

Li, J.; Selvaraju, R.; Gotmare, A.; Joty, S.; Xiong, C.; and Hoi, S. C. H. 2021a. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in neural information processing systems*, 34: 9694–9705.

Li, J.; Selvaraju, R. R.; Gotmare, A.; Joty, S. R.; Xiong, C.; and Hoi, S. C. 2021b. Align before Fuse: Vision and Language Representation Learning with Momentum Distillation. In Ranzato, M.; Beygelzimer, A.; Dauphin, Y. N.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, 9694–9705.

Li, L. H.; Yatskar, M.; Yin, D.; Hsieh, C.; and Chang, K. 2019. VisualBERT: A Simple and Performant Baseline for Vision and Language. *CoRR*, abs/1908.03557.

Li, T.; Sahu, A. K.; Talwalkar, A.; and Smith, V. 2020a. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3): 50–60.

Li, X.; Yin, X.; Li, C.; Zhang, P.; Hu, X.; Zhang, L.; Wang, L.; Hu, H.; Dong, L.; Wei, F.; Choi, Y.; and Gao, J. 2020b. Oscar: Object-Semantics Aligned Pre-training for Vision-Language Tasks. In Vedaldi, A.; Bischof, H.; Brox, T.; and Frahm, J., eds., *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXX*, volume 12375 of *Lecture Notes in Computer Science*, 121–137. Springer.

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

Liu, Y.; Garg, S.; Nie, J.; Zhang, Y.; Xiong, Z.; Kang, J.; and Hossain, M. S. 2020. Deep anomaly detection for time-series data in industrial IoT: A communication-efficient on-device federated learning approach. *IEEE Internet of Things Journal*, 8(8): 6348–6358.

Lu, J.; Batra, D.; Parikh, D.; and Lee, S. 2019. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. In Wallach, H. M.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E. B.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 13–23.

Lu, W.; Hu, X.; Wang, J.; and Xie, X. 2023. FedCLIP: Fast Generalization and Personalization for CLIP in Federated Learning. *arXiv preprint arXiv:2302.13485*.

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

Mendieta, M.; Yang, T.; Wang, P.; Lee, M.; Ding, Z.; and Chen, C. 2022. Local learning matters: Rethinking data heterogeneity in federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8397–8406.

Peng, X.; Bai, Q.; Xia, X.; Huang, Z.; Saenko, K.; and Wang, B. 2019. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, 1406–1415.

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

Ren, M.; Kiros, R.; and Zemel, R. 2015. Exploring models and data for image question answering. *Advances in neural information processing systems*, 28.

Sheller, M. J.; Edwards, B.; Reina, G. A.; Martin, J.; Pati, S.; Kotrotsou, A.; Milchenko, M.; Xu, W.; Marcus, D.; Colen, R. R.; et al. 2020. Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data. *Scientific reports*, 10(1): 12598.

Singh, A.; Hu, R.; Goswami, V.; Couairon, G.; Galuba, W.; Rohrbach, M.; and Kiela, D. 2022. FLAVA: A Foundational Language And Vision Alignment Model. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, 15617–15629. IEEE.

Srinivasan, T.; Chang, T.-Y.; Pinto Alva, L.; Chochlakis, G.; Rostami, M.; and Thomason, J. 2022. Climb: A continual learning benchmark for vision-and-language tasks. *Advances in Neural Information Processing Systems*, 35: 29440–29453.

Su, S.; Yang, M.; Li, B.; and Xue, X. 2022. Cross-domain Federated Adaptive Prompt Tuning for CLIP. *arXiv preprint arXiv:2211.07864*.

Su, W.; Zhu, X.; Cao, Y.; Li, B.; Lu, L.; Wei, F.; and Dai, J. 2020. VL-BERT: Pre-training of Generic Visual-Linguistic Representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Suhr, A.; Zhou, S.; Zhang, A.; Zhang, I.; Bai, H.; and Artzi, Y. 2018. A corpus for reasoning about natural language grounded in photographs. *arXiv preprint arXiv:1811.00491*.

Suhr, A.; Zhou, S.; Zhang, A.; Zhang, I.; Bai, H.; and Artzi, Y. 2019. A Corpus for Reasoning about Natural Language Grounded in Photographs. In Korhonen, A.; Traum, D. R.; and Màrquez, L., eds., *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, 6418–6428. Association for Computational Linguistics.

Sun, G.; Mendieta, M.; Yang, T.; and Chen, C. 2022. Exploring Parameter-Efficient Fine-tuning for Improving Communication Efficiency in Federated Learning. *arXiv preprint arXiv:2210.01708*.

Sung, Y.-L.; Cho, J.; and Bansal, M. 2022. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5227–5237.

Tan, H.; and Bansal, M. 2019. LXMERT: Learning Cross-Modality Encoder Representations from Transformers. In Inui, K.; Jiang, J.; Ng, V.; and Wan, X., eds., *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, 5099–5110. Association for Computational Linguistics.

Tan, Y.; Long, G.; Ma, J.; Liu, L.; Zhou, T.; and Jiang, J. 2022. Federated learning from pre-trained models: A contrastive learning approach. *Advances in Neural Information Processing Systems*, 35: 19332–19344.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wei, J.; Bosma, M.; Zhao, V. Y.; Guu, K.; Yu, A. W.; Lester, B.; Du, N.; Dai, A. M.; and Le, Q. V. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Xie, N.; Lai, F.; Doran, D.; and Kadav, A. 2019a. Visual Entailment: A Novel Task for Fine-Grained Image Understanding. *CoRR*, abs/1901.06706.

Xie, N.; Lai, F.; Doran, D.; and Kadav, A. 2019b. Visual Entailment: A Novel Task for Fine-grained Image Understanding. *arXiv preprint arXiv:1901.06706*.

Yang, M.; Su, S.; Li, B.; and Xue, X. 2023. Exploring One-shot Semi-supervised Federated Learning with A Pre-trained Diffusion Model. *arXiv preprint arXiv:2305.04063*.

Yang, X.; Xiong, B.; Huang, Y.; and Xu, C. 2022. Cross-Modal Federated Human Activity Recognition via Modality-Agnostic and Modality-Specific Representation Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 3063–3071.

Yu, Q.; Liu, Y.; Wang, Y.; Xu, K.; and Liu, J. 2023. Multimodal Federated Learning via Contrastive Representation Ensemble. *arXiv preprint arXiv:2302.08888*.

Yu, S.; Muñoz, J. P.; and Jannesari, A. 2023. Federated Foundation Models: Privacy-Preserving and Collaborative Learning for Large Models. *arXiv preprint arXiv:2305.11414*.

Zellers, R.; Bisk, Y.; Farhadi, A.; and Choi, Y. 2019. From Recognition to Cognition: Visual Commonsense Reasoning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhang, J.; Vahidian, S.; Kuo, M.; Li, C.; Zhang, R.; Wang, G.; and Chen, Y. 2023. Towards Building the Federated GPT: Federated Instruction Tuning. *arXiv preprint arXiv:2305.05644*.

Zhou, C.; Li, Q.; Li, C.; Yu, J.; Liu, Y.; Wang, G.; Zhang, K.; Ji, C.; Yan, Q.; He, L.; et al. 2023. A comprehensive survey on pretrained foundation models: A history from bert to chatgpt. *arXiv preprint arXiv:2302.09419*.

Zhuang, W.; Chen, C.; and Lyu, L. 2023. When Foundation Model Meets Federated Learning: Motivations, Challenges, and Future Directions. *arXiv preprint arXiv:2306.15546*.

A PEFT methods adapted for FL

In this section, we provide additional Parameter-Efficient Finetuning Methods adapted for foundation models in FL.

(1) LoRA (Hu et al. 2021) aims to optimize the projection layers in self-attention sub-layer, W_q (for queries Q) and W_k (for keys K), by optimizing the low-rank decomposition of the optimization update, i.e., $W_{qk}^{down} \in \mathbb{R}^{D\times r}$ and $W_{qk}^{up} \in \mathbb{R}^{r\times d}$ $(r < d)$. The updated block computes Q and K as

$$\begin{aligned} Q &= xW_q + s \cdot W_q^{down} W_q^{up}, \\ K &= xW_k + s \cdot W_k^{down} W_k^{up}, \end{aligned} \tag{1}$$

where x is the block input and s is the scaling factor. Here, the learnable parameters w transmitted in FL is the weight matrices of the low-rank projection layers.

(2) Prompt (Guo et al. 2022) prepends L trainable tokens to the input of a Transformer block. To maintain a similar communication costs compared with other baselines adapted for FL, we apply prompt tuning only to the input embedding for the model, i.e., the learnable prompts will be prepend to the input textual and visual tokens and viewed as the lightweight modules w.

(3) Bias (Cai et al. 2020) aims to adapt the pre-trained foundation model with only fine-tuning a specific group of parameters, the bias term. Here, the learnable w transmitted in FL corresponds to the bias terms in the foundation model parameters θ.

B Experimental Details

B.1 Dataset Details

In this section, we provide more details about the settings of the motivational case study and the other multi-modal FL benchmark datasets.

For the motivation study, we consider an image classification task and adopt the downsampled version of the DomainNet (Peng et al. 2019) dataset following (Tan et al. 2022), where each client containing around 100 to 200 labelled images of the same 10 classes, the image size is set to 64x64. The goal here is to finetune a global vision foundation model that is applicable to all clients. The example images from the benchmark are provided in Figure 1, where we observe different image style and appearance between data from different domains, which simulates a heterogenous FL system.

Domain: We observe that there exists visual data heterogeneity, i.e., from the abstract images to the real-world photo-realistic images, and textual data heterogeneity, i.e., questions and answers from different perspectives. To simulate the client data scarcity in FL following (McMahan et al. 2017), we downsample all datasets. Specifically, we select the triplets containing the top-100 frequent answers and randomly sample the dataset to force each dataset containing around 2.5k triplets.

Function & Scene: In Figure 2, we provide examples from different splits used in the *Function* and *Scene* benchmark. We adapt the 2 splits in the original CLOVE dataset (Lei et al. 2023) to FL. Specifically, we select 5 different functions for *Function* benchmark, namely, Object Recognition, Attribute Recognition, Relation Reasoning, Logic Reasoning, Knowledge Reasoning. We select 6 different scenes for *Scene* benchmark, namely, ShopAndDinning, Workplace, HomeOrHotel, Transportation, SportAndLeisure, Outdoors. Similarly, we downsample all datasets. Specifically, we select the triplets containing the top-100 frequent answers and randomly sample each subdataset to force each dataset containing around 2.5k triplets.

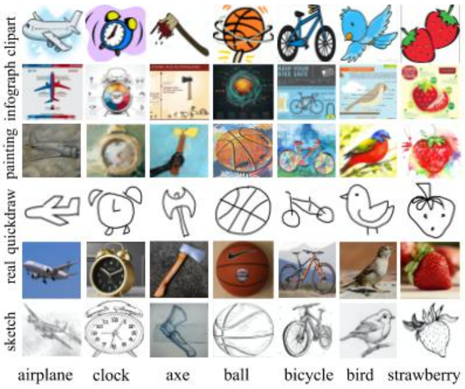


Figure 1: Example data from the DomainNet (Peng et al. 2019) dataset, each client owns data from one of the domain.

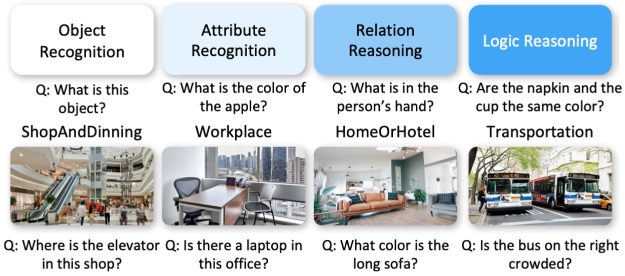

Object Recognition
Attribute Recognition
Relation Reasoning
Logic Reasoning
Q: What is this object?
Q: What is the color of the apple?
Q: What is in the person's hand?
Q: Are the napkin and the cup the same color?
ShopAndDinning
Workplace
HomeOrHotel
Transportation
Q: Where is the elevator in this shop?
Q: Is there a laptop in this office?
Q: What color is the long sofa?
Q: Is the bus on the right crowded?

Figure 2: Examples from *Function* (top) and *Scene* (bottom) benchmark introduced in CLOVE (Lei et al. 2023).

Task: We adapt the CLiMB (Srinivasan et al. 2022) benchmark for FL, where we adop 4 different datasets with different vision-language tasks, namely, *VQA*, *NLVR*, where we should determine whether a natural language sentence is true about a pair of photographs, *SNLI-VE*, where we should determine whether the relation between an image-text pair is entailment, neutral or contradiction, *VCR*. where we should answer the question by selecting from the multiple choices. We provide examples from each dataset in Figure 3. We utilize the code provided in the original paper for applying the dataset downsampling, where we convert NLVR and SNLI-VE to low-shot setup with 1024 samples per each class, while for VQA and VCR we sample 0.01% data of the original datasets.

B.2 Hyperparameter Setups

In this section, we provide the hyperparameters used in our experiments. To make a fair comparison between different PEFT algorithms and `FedDAT`, we apply the same hyperparameters tuning for all methods in different benchmarks. Specifically, for the common hyperparameters used for FL,

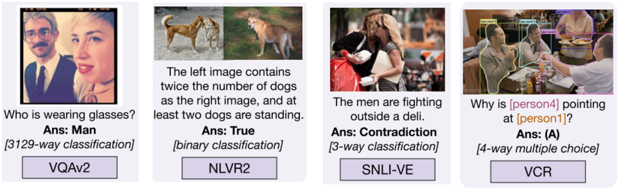

Who is wearing glasses?
Ans: Man
[3129-way classification]
VQAv2
The left image contains twice the number of dogs as the right image, and at least two dogs are standing.
Ans: True
[binary classification]
NLVR2
The men are fighting outside a deli.
Ans: Contradiction
[3-way classification]
SNLI-VE
Why is [person4] pointing at [person1]?
Ans: (A)
[4-way multiple choice]
VCR

Figure 3: Example data from the *Task* benchmark, each client owns data from one of the dataset.

we set batch size to 64, learning rate to 0.0001, number of communication round to 20, number of client local training epoch to 1, random seeds to $\{1, 2, 3\}$. We adopt the AdamW (Loshchilov and Hutter 2017) with weight decay of 0.01. The learning rate is warmed up for the first 10% of the total training steps and is decayed linearly to zero until the end of the local training. For the local PEFT methods, i.e., tuning of the models without any federated communication, we set the total training epoch at each client to 20. All experiments are executed in the GPU GeForce GTX TITAN X with 12GB memory.

We provide detailed hyperparameter for different methods in the following: For *Adapter*, we use the architecture introduced in (Houlsby et al. 2019), set ReLU as the intermediate activation function, and set the reduction factor to 16. For *Prompt* tuning, we set the token size to be 20, i.e., 10 trainable tokens will be prepended before the visual embeddings and another 10 tokens before the textual embeddings. For *LoRA*, we set the reduction factor for the weight matrices to 16. For `FedDAT`, we set the weighting ratio for the losses applied for *MKD* (Equation 7), i.e., α and β, with a ramp-up scheduler based on the local update steps t. Specifically, the formula for α and β is given in the following:

$$x = exp(-5(1 - \frac{t}{T_0})) \tag{2}$$

where T_0 is the total update steps at each client. For the scalability analysis of $FedDAT$, we set the total communication round to $(20\times$number of clients per dataset$)$ to consider the overall data quantity and keep the other hyperparameters unchanged.

C Additional Results

C.1 Client-specific Benchmark Results

In this section, we provide the client-specific performance on the *Funciton*, *Scene*, and *Task* benchmark in Table 1, Table 2, and Table 3, respectively. We observe that `FedDAT` consistently outperforms the most promising centralized PEFT methods, i.e., $Adapter$, with an observable margin, which also narrows the performance gap towards the fully finetuning method $full$.

C.2 Convergence Analysis

In this section, we provide the convergence analysis of ViLT model on *Function* and *Scene* benchmark in Figure 4 and Figure 5, respectively. Here, we provide the client local testing results after each communication round. From the results, we observe that `FedDAT` indicates faster convergence rate and achieves better final results.

References

Cai, H.; Gan, C.; Zhu, L.; and Han, S. 2020. Tinytl: Reduce memory, not parameters for efficient on-device learning. *Advances in Neural Information Processing Systems*, 33: 11285–11297.

Guo, T.; Guo, S.; Wang, J.; and Xu, W. 2022. PromptFL: Let Federated Participants Cooperatively Learn Prompts Instead of Models–Federated Learning in Age of Foundation Model. *arXiv preprint arXiv:2208.11625*.

Houlsby, N.; Giurgiu, A.; Jastrzebski, S.; Morrone, B.; De Laroussilhe, Q.; Gesmundo, A.; Attariyan, M.; and Gelly, S. 2019. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning*, 2790–2799. PMLR.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Lei, S. W.; Gao, D.; Wu, J. Z.; Wang, Y.; Liu, W.; Zhang, M.; and Shou, M. Z. 2023. Symbolic replay: Scene graph as prompt for continual learning on vqa task. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 1250–1259.

Loshchilov, I.; and Hutter, F. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

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

Peng, X.; Bai, Q.; Xia, X.; Huang, Z.; Saenko, K.; and Wang, B. 2019. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, 1406–1415.

Srinivasan, T.; Chang, T.-Y.; Pinto Alva, L.; Chochlakis, G.; Rostami, M.; and Thomason, J. 2022. Climb: A continual learning benchmark for vision-and-language tasks. *Advances in Neural Information Processing Systems*, 35: 29440–29453.

Tan, Y.; Long, G.; Ma, J.; Liu, L.; Zhou, T.; and Jiang, J. 2022. Federated learning from pre-trained models: A contrastive learning approach. *Advances in Neural Information Processing Systems*, 35: 19332–19344.

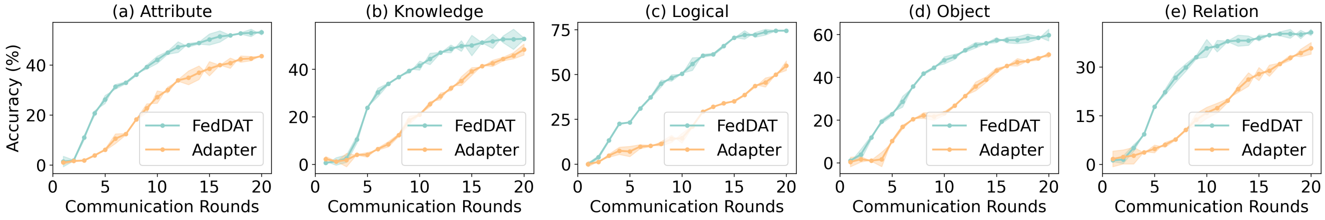


Figure 4: Convergence analysis of ViLT model on different clients in *Function* benchmark. Client local testing accuracy after each communication round is reported.

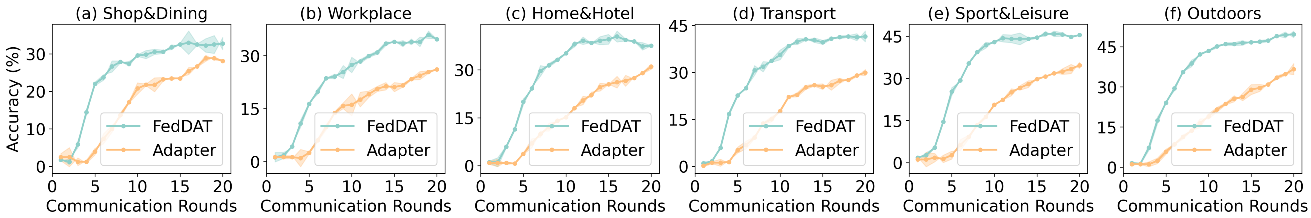


Figure 5: Convergence analysis of ViLT model on different clients in *Scene* benchmark. Client local testing accuracy after each communication round is reported.

Backbone	Method	Attribute	Knowledge	Logical	Object	Relation	Average
ViLT	clf-L	37.40±0.86	28.83±1.43	29.07±1.64	42.58±1.82	20.02±2.25	31.58±1.97
	$prompt$	41.26±0.97	40.27±1.28	40.23±1.56	55.70±0.87	25.21±1.48	40.53±1.56
	$bias$	44.11±0.57	40.42±2.14	48.84±2.19	56.12±1.52	29.55±1.35	43.81±1.39
	$Adapter$-L	37.19±2.88	42.44±3.28	41.86±2.19	48.92±4.41	27.97±0.28	39.68±2.19
	$Adapter$	44.41±1.87	46.37±1.43	58.14±3.30	56.45±1.06	36.45±0.90	48.37±1.56
	FedDAT	50.90±3.52	52.90±3.64	72.15±3.66	59.73±0.82	36.26±2.91	**54.39**±2.36
	$full$-L*	46.00±0.28	57.83±1.28	78.94±5.48	57.83±1.67	43.46±1.20	56.81±2.97
	$full$*	53.86±3.45	57.56±0.14	75.58±1.64	63.33±2.58	47.78±4.65	59.62±2.56
VAuLT	clf-L	28.05±2.30	24.90±4.71	24.03±2.19	40.33±0.46	21.29±1.05	27.72±3.05
	$prompt$	32.15±0.96	35.00±1.29	27.05±0.98	54.09±1.32	33.31±3.08	36.32±2.07
	$bias$	33.95±0.57	33.37±1.85	29.07±4.94	54.84±2.13	29.34±1.56	36.11±3.05
	$Adapter$-L	36.25±0.85	33.47±2.24	30.49±3.98	55.12±2.26	30.79±1.17	37.22±2.38
	$Adapter$	34.69±1.12	36.49±2.45	46.25±1.19	58.49±3.68	31.57±5.54	41.50±3.24
	FedDAT	44.21±1.87	44.45±0.71	40.70±2.74	59.14±2.74	34.21±3.64	**44.54**±2.08
	$full$-L*	44.20±2.16	42.54±4.57	62.79±2.19	52.69±0.30	43.43±0.30	49.13±2.68
	$full$*	42.08±2.02	38.70±0.86	50.00±1.37	62.05±1.07	39.09±2.55	46.38±1.57
ALBEF	$prompt$	35.20±1.17	29.56±0.93	27.28±2.38	40.39±1.05	22.38±1.47	30.96±1.55
	$bias$	30.28±0.86	28.93±1.29	27.13±2.29	41.94±0.61	15.14±1.65	28.68±1.57
	$Adapter$-L	40.38±1.04	36.83±1.64	31.01±0.78	53.26±1.02	32.06±1.71	38.71±1.28
	$Adapter$	40.11±1.12	42.07±1.91	37.47±3.13	58.99±1.01	33.26±1.49	42.38±2.05
	FedDAT	51.02±1.27	53.96±3.50	49.87±4.27	62.15±0.98	45.13±1.81	**52.43**±2.45
	$full$-L*	53.05±1.44	55.85±0.28	72.87±1.10	61.61±2.28	44.28±2.59	57.53±1.78
	$full$*	55.39±1.87	58.87±1.71	70.16±0.54	63.77±0.76	55.93±0.23	60.82±1.04

Table 1: Evaluation results of different finetuning methods on our FL benchmark with distribution shift *Function* in Language space. ”L” indicates client local finetuning where no communication is involved. We report the mean±std accuracy of each client from 3 runs with different seeds.

Backbone	Method	Shop&Dining	Workplace	Home&Hotel	Transport	Sport&Leisure	Outdoors	Average
ViLT	*clf*-L	19.69±0.52	18.84±2.13	22.97±2.33	26.30±1.17	28.30±1.10	31.02±0.45	24.52±0.95
	prompt	31.52±1.26	28.34±1.97	28.71±2.04	32.08±1.49	28.86±0.67	33.69±2.06	30.53±1.30
	bias	27.88±1.04	27.77±2.67	28.68±0.78	34.31±2.17	36.80±2.06	46.48±0.30	33.65±1.87
	Adapter-L	29.59±2.07	23.74±1.96	31.43±2.49	31.72±2.16	37.36±1.90	37.64±2.57	31.91±2.05
	Adapter	28.61±0.15	24.37±0.05	32.42±2.33	30.89±1.34	35.23±1.42	34.86±1.65	31.07±1.08
	FedDAT	34.03±1.06	32.41±1.76	36.65±1.87	39.34±1.92	43.61±0.90	49.05±0.54	**39.35**±1.25
	full-L*	31.05±1.73	29.52±1.95	40.66±1.24	37.73±2.34	41.27±0.47	47.77±1.51	38.00±1.48
	full*	33.74±2.08	31.78±6.93	37.69±1.09	41.27±4.67	44.52±2.53	54.7±2.86	40.62±3.76
VAuLT	*clf*-L	15.89±1.39	16.33±0.35	20.44±0.62	23.47±3.17	20.92±0.79	30.28±0.91	21.22±2.08
	prompt	24.43±1.45	22.06±2.06	21.87±1.75	31.46±0.93	24.99±2.04	28.98±0.65	25.63±1.54
	bias	19.07±3.80	17.71±0.88	21.65±2.02	26.30±0.83	29.98±3.48	34.65±0.45	24.89±2.17
	Adapter-L	20.78±2.97	21.86±2.06	27.54±2.14	29.64±1.57	30.65±1.83	40.94±1.61	28.57±1.98
	Adapter	19.39±3.32	21.44±2.14	29.38±1.65	31.45±2.68	33.48±3.17	41.22±3.67	29.39±2.65
	FedDAT	27.51±2.38	25.87±1.15	32.04±1.92	37.14±2.26	36.75±2.02	46.56±2.37	**34.31**±2.87
	full-L*	32.40±1.90	27.89±2.84	33.30±1.09	35.73±0.83	40.38±0.48	40.94±3.32	35.11±1.99
	full*	34.36±2.60	28.77±3.38	32.42±2.02	37.86±1.51	39.04±1.10	47.87±3.17	36.72±2.57
ALBEF	*prompt*	20.76±2.08	24.32±1.87	21.21±0.32	30.52±0.68	27.45±0.96	29.83±2.23	25.68±1.16
	bias	15.77±2.59	16.21±3.44	18.46±0.10	27.24±1.53	24.72±0.48	27.08±1.51	21.58±1.88
	Adapter-L	25.75±3.74	21.78±3.47	27.69±1.34	34.59±0.59	33.56±1.99	38.88±1.42	30.38±2.68
	Adapter	32.68±1.73	22.19±1.62	29.45±0.58	35.14±1.87	32.29±2.08	41.08±1.18	32.14±1.49
	FedDAT	36.59±2.90	30.57±0.63	37.14±1.32	45.20±0.49	41.46±1.49	49.54±2.10	**40.08**±2.08
	full-L*	31.79±2.07	30.02±2.31	35.16±0.93	41.16±1.17	40.27±0.95	50.43±3.17	38.14±2.03
	full*	37.89±1.04	37.19±0.71	41.65±0.16	46.23±2.33	46.42±1.13	53.52±1.81	43.82±1.72

Table 2: Evaluation results of different finetuning methods on our FL benchmark *Scene* with distribution shift in Vision space. ”L” indicates client local finetuning where no communication is involved. We report the mean±std accuracy of each client from 3 runs with different seeds.

Backbone	Method	VCR	NLVR	VQA	VE	Average
ViLT	*clf*-L	38.98±0.53	58.77±0.27	37.32±0.45	62.78±0.38	49.46±0.39
	LoRA	37.65±0.96	56.26±1.20	36.08±2.08	61.30±1.14	47.82±1.42
	prompt	41.13±1.32	55.86±2.02	38.21±0.60	63.02±0.57	49.55±1.14
	bias	42.05±2.06	58.17±1.57	39.12±1.32	63.50±0.47	50.71±1.26
	Adapter-L	40.52±2.98	58.30±3.12	37.47±0.38	62.08±1.13	49.59±1.74
	Adapter	44.68±1.02	57.69±3.45	39.83±0.21	63.57±0.68	51.44±1.34
	FedDAT	45.58±0.76	58.42±0.86	40.87±0.41	64.62±0.31	**52.37**±0.52
	full-L*	39.86±2.68	57.25±1.04	40.21±0.68	65.36±0.52	50.64±1.42
	full*	44.77±1.78	60.94±0.56	42.25±1.12	64.72±1.35	53.17±0.69
VAuLT	*clf*-L	29.81±1.12	48.44±1.61	28.70±0.67	51.56±0.28	39.63±1.07
	LoRA	29.91±1.05	46.37±0.45	27.52±2.13	49.60±0.87	38.35±1.47
	prompt	30.94±0.76	47.70±1.23	26.79±0.32	49.59±2.56	38.75±1.34
	bias	26.79±1.36	49.61±0.52	29.63±1.16	51.82±1.30	39.46±0.99
	Adapter-L	29.03±0.96	52.54±2.06	29.14±0.09	50.95±1.05	40.42±1.21
	Adapter	28.18±0.27	50.20±0.19	29.51±0.12	52.87±2.01	40.19±0.89
	FedDAT	30.58±0.14	51.48±1.21	30.31±0.82	52.76±1.08	**41.28**±0.57
	full-L*	32.83±1.62	51.56±0.24	31.19±1.04	51.04±2.01	41.66±1.32
	full*	32.57±0.92	52.00±1.07	30.93±0.97	54.27±0.88	42.44±0.71

Table 3: Evaluation results of different finetuning methods on our FL benchmark *Task* with different VL-tasks. ”L” indicates client local finetuning where no communication is involved. We report the mean±std accuracy of each client from 3 runs with different seeds.

Chapter 4

FedPop: Federated Population-based Hyperparameter Tuning

This chapter contains the publication

> **Haokun Chen**, Denis Krompass, Jindong Gu, Volker Tresp. FedPop: Federated Population-based Hyperparameter Tuning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025, 39(15): 15776-15784.

FedPop: Federated Population-based Hyperparameter Tuning

Haokun Chen[1,2*], **Denis Krompaß**[2], **Jindong Gu**[3*], **Volker Tresp**[1,4]
[1] Ludwig Maximilian University of Munich, Munich, Germany
[2] Siemens Technology, Munich, Germany
[3] University of Oxford, Oxford, England
[4] Munich Center for Machine Learning, Munich, Germany
{haokun.chen, denis.krompass}@siemens.com,
jindong.gu@outlook.com, volker.tresp@lmu.de

Abstract

Federated Learning (FL) is a distributed machine learning (ML) paradigm, in which multiple clients collaboratively train ML models without centralizing their local data. Similar to conventional ML pipelines, the client local optimization and server aggregation procedure in FL are sensitive to the hyperparameter (HP) selection. Despite extensive research on tuning HPs for centralized ML, these methods yield suboptimal results when employed in FL. This is mainly because their ”training-after-tuning” framework is unsuitable for FL with limited client computation power. While some approaches have been proposed for HP-Tuning in FL, they are limited to the HPs for client local updates. In this work, we propose a novel HP-tuning algorithm, called Federated Population-based Hyperparameter Tuning (FedPop), to address this vital yet challenging problem. FedPop employs population-based evolutionary algorithms to optimize the HPs, which accommodates various HP types at both the client and server sides. Compared with prior tuning methods, FedPop employs an online ”tuning-while-training” framework, offering computational efficiency and enabling the exploration of a broader HP search space. Our empirical validation on the common FL benchmarks and complex real-world FL datasets, including full-sized Non-IID ImageNet-1K, demonstrates the effectiveness of the proposed method, which substantially outperforms the concurrent state-of-the-art HP-tuning methods in FL.

Introduction

Federated Learning (FL) is an effective machine learning paradigm suitable for decentralized data sources (McMahan et al. 2017). Similar to the conventional ML algorithms, FL exhibits sensitivity to empirical choices of hyperparameters (HPs), such as learning rate, and optimization steps (Kairouz et al. 2021). Hyperparameter Tuning (HPT) is a vital yet challenging component of the ML pipeline, which has been extensively studied in the context of centralized ML (Hutter, Kotthoff, and Vanschoren 2019). However, traditional HPT methods, such as Bayesian Optimization (Snoek, Larochelle, and Adams 2012), are not suitable for FL systems. These methods typically utilize the ”training-after-tuning” framework. Within this framework, a substantial number of HPs needs to be evaluated, which involves repetitive training of models until convergence and subsequent retraining after optimizing the optimal HP. Such approaches can drastically increase the client’s local computational costs and communication overheads, as it needs to execute multiple federated communications when evaluating only one HP. Furthermore, the distributed validation datasets impose a major challenge for HPT in FL, making it infeasible to evaluate HP for a large number of participating clients.

Recently, a few approaches have emerged to address the problem intersection of HPT and FL, but they still exhibit certain limitations: FedEx (Khodak et al. 2021) predefines a narrower HP search space, while FLoRA (Zhou et al. 2023) requires costly retraining after HP-optimization. Moreover, they are only applicable for tuning the HPs used in client local updates. In this paper, we propose Federated Population-based Hyperparameter Tuning (`FedPop`) to address the challenge of tuning HPs for FL. `FedPop` applies population-based evolutionary algorithm (Jaderberg et al. 2017) to optimize the HPs, which adds minimal computational overheads and accommodates various HP types at the client and server sides. Most importantly, `FedPop` employs an online ”tuning-while-training” framework, enhancing efficiency and thereby allowing the exploration of a broader HP search space.

In `FedPop`, we first construct multiple HP-configurations as our tuning population, i.e., we initialize multiple tuning processes (members) with randomly initialized HP-configuration, containing the HPs used in the server aggregation and the local client updates. Afterwards, we apply an evolutionary update mechanism to optimize the HPs of each member by leveraging information across different HP-configurations (`FedPop-G`). Hereby, the HPs in underperforming members will be replaced by a perturbed version of the HPs from better-performing ones, enabling an efficient and effective online propagation of the HPs. To further improve the HPs for the local client updates in a fine-grained manner, we consider the active clients in each communication round as our local population, where each member contains one HP-vector used in the local client update (`FedPop-L`). Similarly, evolutionary updates are executed based on the local validation performance of each member to tune these HP-vectors. Most importantly, all the tuning processes, i.e., members of the population, are decentralized

[*] Corresponding author

and can be asynchronous, aligning perfectly with the distributed system design.

The proposed algorithm `FedPop` achieves new state-of-the-art (SOTA) results on three common FL benchmarks with both vision and language tasks, surpassing the concurrent SOTA HPT method for FL, i.e., FedEx (Khodak et al. 2021). Moreover, we evaluate `FedPop` on large-scale cross-silo FL benchmarks with feature distribution shift (Li et al. 2021), where its promising results demonstrate its applicability to complex real-world FL applications. Most importantly, we demonstrate the scalability of `FedPop`, where we show its applicability to full-sized ImageNet-1K (Deng et al. 2009) with ResNet-50 (He et al. 2016). Our contributions in this paper can be summarized as follows:

- We propose an effective and efficient online hyperparameter tuning (HPT) algorithm, `FedPop`, to address HPT problem for decentralized ML systems.
- We conduct comprehensive experiments on three common FL benchmarks with both vision and language tasks, in which `FedPop` achieves new SOTA results.
- We verify the maturity of `FedPop` for complex real-world cross-silo FL applications, and further analyze its convergence rate on full-sized non-IID ImageNet-1K, as well as its effectiveness when combined with various federated optimization algorithms.

Related Works

Hyperparameter Tuning for FL System

Previous works for tuning hyperparameters in FL focus only on specific aspects: (Wang et al. 2019) tunes only the local optimization epochs based on the client's resources, while (Koskela and Honkela 2018; Mostafa 2019; Reddi et al. 2020) focus on the learning rate of client local training. (Dai, Low, and Jaillet 2020, 2021) apply Bayesian Optimization (BO) (Snoek, Larochelle, and Adams 2012) in FL and optimize a personalized model for each client, while (Tarzanagh et al. 2022) computes federated hypergradient and applies bilevel optimization. (He, Annavaram, and Avestimehr 2020; Xu et al. 2020; Garg, Saha, and Dutta 2020; Seng et al. 2022; Khan et al. 2023) tune architectural hyperparameters, in particular, adapt Neural Architecture Search (NAS) for FL. (Zhang et al. 2022) tunes hyperparameter based on the federated system overheads, while (Maumela, Nelwamondo, and Marwala 2022) assumes the training data of each client is globally accessible. (Mlodozeniec, Reisser, and Louizos 2023) partitions both clients and the neural network and tunes only the hyperparameters used in data augmentation. (Khodak et al. 2020, 2021) systematically analyze the challenges of hyperparameter tuning in FL and propose FedEx for client local hyperparameters. (Zhou et al. 2023) proposes a hyperparameter optimization algorithm that aggregates the client's loss surfaces via single-shot upload. In contrast, the proposed method, FedPop, is applicable to various HP types on the client and server sides. In addition, it does not impose any restrictions on data volume and model architecture.

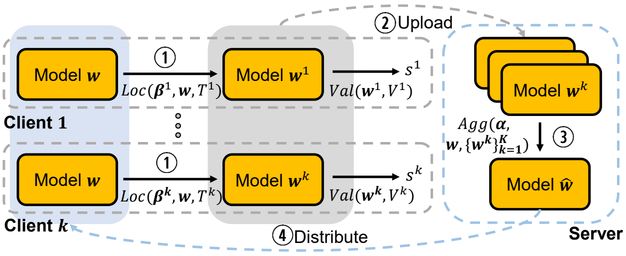

②Upload
Model $\boldsymbol{w}$
①
$Loc(\boldsymbol{\beta}^1, \boldsymbol{w}, T^1)$
Model $\boldsymbol{w}^1$
s^1
$Val(\boldsymbol{w}^1, V^1)$
Client 1
Model $\boldsymbol{w}$
①
$Loc(\boldsymbol{\beta}^k, \boldsymbol{w}, T^k)$
Model $\boldsymbol{w}^k$
s^k
$Val(\boldsymbol{w}^k, V^k)$
Client k
Model $\boldsymbol{w}^k$
$Agg(\boldsymbol{\alpha}, \boldsymbol{w}, \{\boldsymbol{w}^k\}_{k=1}^K)$
③
Model $\hat{\boldsymbol{w}}$
④Distribute
Server

Figure 1: Schematic illustration of the operations involved in one communication round, summarized as `Fed-Opt`.

Evolutionary Algorithms

Evolutionary algorithms are inspired by the principles of natural evolution, where stochastic genetic operators, e.g., mutation and selection, are applied to the members of the existing population to improve their survival ability, i.e., quality (Telikani et al. 2021). Evolutionary algorithms have shown their potential to improve machine learning algorithms, including architecture search (Real et al. 2017; Liu et al. 2017), hyperparameter tuning (Jaderberg et al. 2017; Parker-Holder, Nguyen, and Roberts 2020), and Automated Machine Learning (AutoML) (Liang et al. 2019; Real et al. 2020). FedPop employs an online evolutionary algorithm, which is computationally efficient and explores a broader HP search space. To the best of our knowledge, FedPop is the first work combining evolutionary algorithms with HP optimization in Federated Learning.

Federated Hyperparmater Tuning

Problem Definition

In this section, we introduce the problem setup of hyperparameter tuning for FL. Following the setting introduced in (Khodak et al. 2021), we assume that there are $N_c \in \mathbb{N}^+$ clients joining the federated communication. Each client k owns a training, validation, and testing set, denoted by T^k, V^k, and E^k, respectively. To simulate the communication capacity of a real-world federated system, we presume that there are exactly $K \in \mathbb{N}^+$ active clients joining each communication round. In FedAvg (McMahan et al. 2017), the central server obtains the model weight $\boldsymbol{w} \in \mathbb{R}^d$ by iteratively distributing $\boldsymbol{w}$ to the active clients and averaging the returned optimized weights, i.e., $\{\boldsymbol{w}^k | 1 \leq k \leq K\}$.

More specifically, we denote the server aggregation and the client local training functions as `Agg` and `Loc`, respectively. Our goal is to tune the hyperparameter vectors (**HP-vectors**) used in these two functions. In particular, we denote the HP-vector used in `Agg` and `Loc` as $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, which are sampled from the hyperparameter distribution H_a and H_b, respectively. We define the combination of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ as one **HP-configuration**. In the following, we explain the general steps executed in the communication round, which involves these functions and HP-configurations. We summarize these steps as federated optimization (`Fed-Opt`), which is illustrated in Figure 1. Specifically, all active clients first execute

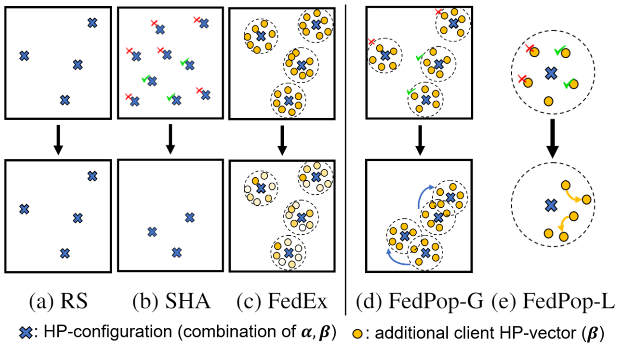

(a) RS
(b) SHA
(c) FedEx
(d) FedPop-G
(e) FedPop-L
✖: HP-configuration (combination of $\boldsymbol{\alpha}, \boldsymbol{\beta}$)
●: additional client HP-vector ($\boldsymbol{\beta}$)

Figure 2: Schematic comparison between `FedPop` and other baselines. `FedEx` optimizes the sampling probabilities of additional $\boldsymbol{\beta}$ based on validation performance (indicated by the brightness of the yellow dots). In contrast, our method supports the tuning of both server (`FedPop-G`) and clients (`FedPop-G` and `-L`) HP-vectors and explores broader search space with the help of evolutionary updates.

function `Loc` (①) in parallel:

$$\boldsymbol{w}^k \leftarrow \texttt{Loc}(\boldsymbol{\beta}^k, \boldsymbol{w}, T^k), \tag{1}$$

which takes the HP-vector $\boldsymbol{\beta}^k$, model parameters $\boldsymbol{w}$ distributed by the central server, and the local training set T^k as inputs, and outputs the optimized model weight $\boldsymbol{w}^k$. Afterwards, the central server aggregates $\boldsymbol{w}^k$, uploaded by the active clients (②) , and executes function `Agg` (③):

$$\hat{\boldsymbol{w}} \leftarrow \texttt{Agg}(\boldsymbol{\alpha}, \boldsymbol{w}, \{\boldsymbol{w}^k | 1 \leq k \leq K\}), \tag{2}$$

which takes HP-vector $\boldsymbol{\alpha}$, current model parameter $\boldsymbol{w}$, updated model parameters from the active clients $\{\boldsymbol{w}^k | 1 \leq k \leq K\}$, and outputs the aggregated model weight $\hat{\boldsymbol{w}}$ which will be distributed to the active clients in the next communication round (④). The goal of the federated hyperparameter tuning method is to find the optimal HP-vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ within a predefined communication budget.

Challenges in Federated Hyperparameter Tuning

Given the problem defined in the previous section, we describe the two main challenges when tuning the hyperparameters for federated learning:

(C1) Extrem resource limitations: The communication budgets for optimizing ML models via FL are always very constrained due to the limited computational power of the clients and connection capacity of the overall system (Li et al. 2020). Therefore, common hyperparameter tuning algorithms, such as extensive local hyperparameter tuning for each client, or experimenting multiple hyperparameter configurations for the overall federated system and then retraining, may not be suitable in the context of FL.

(C2) Distributed validation data: In centralized ML, most hyperparameter tuning algorithms select the HP-configurations based on their validation performance. However, the validation data (V^k) is distributed across the clients in FL. Computing a validation score over all clients is extremely costly and thus infeasible for FL. The alternative is to use the validation performance of client subsets, e.g., the active clients of the communication round, which greatly reduces computational costs. However, this may lead to evaluation bias when the client data are not independent and identically distributed (*Non-IID*).

Method	Number of tried $\boldsymbol{\alpha}$	Number of tried $\boldsymbol{\beta}$	Optim. of $\boldsymbol{\alpha}$	Optim. of $\boldsymbol{\beta}$
`RS`	5	5	✗	✗
`SHA`	27	27	✗	✗
`FedEx`	5	135	✗	✓
`FedPop`	45	>1000	✓	✓

Table 1: Number of HP-vectors tested in different HP-tuning methods on CIFAR-10 benchmark. `FedPop` experiments the largest number of HP-configurations among all methods. Detailed computations are provided in the Appendix.

Baseline Methods

Before introducing the proposed algorithm (`FedPop`) which addresses the challenges of HP-tuning in FL, we illustrate the adaptation of two widely adopted HP-tuning baselines for FL applications and their notations. For the FL setup, we define the maximum communication rounds for the FL system, i.e., total tuning budget, as R_t, and the number of initial HP-configurations as N_c, respectively. We devise two baseline methods for tuning $\boldsymbol{\alpha}, \boldsymbol{\beta}$ as follows:

(1) **Random Search (RS)** first initializes N_c HP-configurations, resulting in a tuning budget of R_c $(= \frac{R_t}{N_c})$ for each HP-configuration. Afterwards, an ML model and N_c tuning processes will be initialized, where each tuning process executes R_c communication rounds to optimize the model using one specific HP-configuration. Finally, the optimized models from all tuning processes will be evaluated and the model exhibiting the highest testing accuracy, as well as its corresponding HP-configuration, are saved.

(2) **Successive Halving (SHA)** is a variation of `RS` which eliminates $\frac{1}{\eta}$-quantile of the under-performing HP-configurations after specific numbers of communication rounds. Within the same tuning budget R_t, `SHA` is able to experiment more HP-configurations compared with `RS`, thus increasing the likelihood of achieving better results. Based on R_t, N_c, and the number of elimination operations, the time step for elimination can be computed. However, the elimination might also discard HP-configurations which lead to promising results but perform poorly at early stages.

Limitations: These baseline methods exhibit two limitations when adapted to FL applications: First, as shown in Figure 2, their numbers of HP-configurations, as well as the HP values, are pre-defined and remain fixed throughout the tuning process. Second, these baseline methods are ”static” and no active tuning is executed inside each tuning process. In other words, the model evaluation results are only obtained and utilized after a specific number of communication rounds. Therefore, we propose `FedPop`, a population-based tuning algorithm that updates the HP-configurations via evolutionary update algorithms. As a result of its high efficiency,

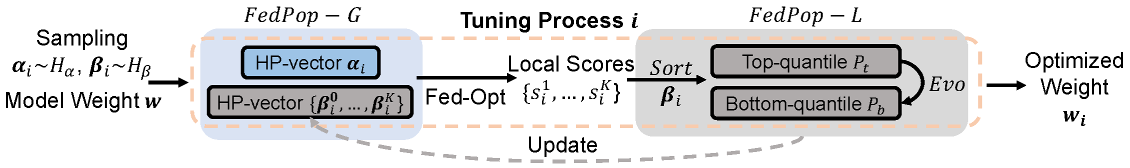


Figure 3: Schematic illustration of `FedPop`, including `FedPop-L` for intra-configuration HP-tuning and `FedPop-G` for inter-configuration HP-tuning. `FedPop` employs an online "tuning-while-training" schema for tuning both server ($\boldsymbol{\alpha}$) and clients ($\boldsymbol{\beta}$) HP-vectors. All functions in `FedPop` can be executed in a parallel and asynchronous manner. *Best viewed in color.*

it experiments the largest number of HP-vectors among all methods (Table 1), which is introduced in the following.

Proposed Method

A schematic illustration of the proposed method, Federated Population-Based Hyperparameter Tuning (`FedPop`), is provided in Figure 3. First, we randomly sample the HP-vectors ($\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$) for each tuning process *in parallel* and execute federated optimization `Fed-Opt` (Figure 1). Subsequently, we conduct `FedPop` based on the validation scores s returned from the active clients in each tuning process. `FedPop` can be divided into 2 sub-procedures: `FedPop-G` aims at tuning both HP-vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ across all HP-configurations (*inter-config*), while `FedPop-L` focuses on a fine-grained search of HP-vector $\boldsymbol{\beta}$ inside each HP-configurations (*intra-config*).

`FedPop` can be wrapped with the aforementioned baselines. Specifically, the primary distinctions between the two wrappers are the number of initialized HP-configurations (N_c) and the execution of the tuning process eliminations in the intermediate steps. In the following, we use the most rudimentary method, `RS`, as our wrapper and elaborate on the proposed method. More details regarding `FedPop` wrapped with `SHA` are provided in the Appendix.

With `RS` as the wrapper, `FedPop` first randomly initializes N_c HP-configurations $(\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i^0)$ as the initial population and copies the model weight vector $\boldsymbol{w}$. Afterwards, we randomly sample addition K HP-vectors, i.e., $\{\boldsymbol{\beta}_i^k | 1 \leq k \leq K\}$, inside a small Δ-ball centered by $\boldsymbol{\beta}_i^0$. Δ is selected based on the distribution of the HP (H_b) and more details are provided in the Appendix. Directly sampling $\boldsymbol{\beta}_i^k$ from H_b is problematic because we find that using too distinct HP-vectors for the active clients would lead to unstable model performance. This phenomenon was also observed by (Khodak et al. 2021). We provide a schematic illustration of the sampling process in Figure 2, where the *yellow dots* ($\{\boldsymbol{\beta}_i^k | 1 \leq k \leq K\}$) are enforced to lie near the *blue crosses* ($\boldsymbol{\beta}_i^0$). Note that this resampling process of $\boldsymbol{\beta}_i^k$ is also executed when $\boldsymbol{\beta}_i^0$ is perturbed via `Evo` in `FedPop-G`. Finally, R_c communication rounds are executed for each tuning process in parallel, where the validation scores s_i^k, of the k_{th} active client in the i_{th} tuning process is recorded. The pseudo codes of the proposed method are given in Algorithm 1.

Evolution-based Hyperparameter Update (`Evo`): Inspired by Population-based Training (Jaderberg et al. 2017), we design our evolution-based hyperparameter update function `Evo` as the following,

$$\texttt{Evo}(\boldsymbol{h}) = \begin{cases} \hat{h}_j \sim U(h_j - \delta_j, h_j + \delta_j) & \text{s.t.} \quad H_j = U(a_j, b_j), \\ \hat{h}_j \sim U\{x_j^{i \pm \lfloor \delta_j \rceil}, x_j^i\} & \text{s.t.} \begin{cases} H_j = U\{x_j^0, ..., x_j^n\}, \\ h_j = x_j^i, \end{cases} \end{cases} \tag{3}$$

where $\boldsymbol{h}$ represents one HP-vector, i.e., $\boldsymbol{\alpha}$ or $\boldsymbol{\beta}$ for our problem setting. We perturb the j_{th} value of $\boldsymbol{h}$, h_j, by resampling it from its possible neighboring values. Concretely, we select the new value of h_j based on the type of its original sampling distribution H_j: (1) If h_j is sampled from a continuous uniform distribution $H_j = U(a_j, b_j)$ (e.g., log-space of learning-rate, dropout), then we perturb h_j by resampling it from $U(h_j - \delta_j, h_j + \delta_j)$, where $\delta_j \leftarrow (b_j - a_j)\epsilon$ and ϵ is the pre-defined perturbation intensity. (2) If $h_j = x_j^i$ is sampled from a discrete uniform distribution $H_j = U\{x_j^0, ..., x_j^n\}$ (e.g., batch-size, epochs), then we perturb h_j by reselecting its value from $\{x_j^{i - \lfloor \delta_j \rceil}, x_j^i, x_j^{i + \lfloor \delta_j \rceil}\}$. To further increase the diversity of the HP search space during tuning, we resample h_j from its original distribution H_j with probability p_{re}.

While the HPs are randomly initialized in the early tuning stages, they become more informative as training progresses. To reflect this in `FedPop`, we employ a cosine annealing schema to control the values of ϵ and p_{re} based on the conducted communication rounds r:

$$x_r = \frac{x_0}{2} \cdot (1 + cos(\pi \frac{r}{R_c})), \tag{4}$$

where x_r and x_0 denote the present and the initial value of the annealed parameter, respectively, x is either ϵ or p_{re}.

`FedPop-G` for Inter-configuration Tuning: In `FedPop-G`, we adopt the average validation loss of all active clients, i.e., $s_i = \frac{1}{K}\sum_{k=1}^{K} s_i^k$, as the performance score for i_{th} HP-configuration. However, s_i may be a biased performance measurement, i.e., the disparity in the difficulty of the validation sets between different clients may lead to noisy s_i. To reduce the impact of the noise, `FedPop-G` is

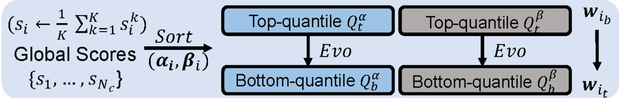

$(s_i \leftarrow \frac{1}{K} \sum_{k=1}^{K} s_i^k)$
Global Scores
$\{s_1, ..., s_{N_c}\}$
Sort
$(\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i)$
Top-quantile Q_t^α
Evo
Bottom-quantile Q_b^α
Top-quantile Q_t^β
Evo
Bottom-quantile Q_b^β
$\boldsymbol{w}_{i_b}$
$\boldsymbol{w}_{i_t}$

Figure 4: Schematic illustration of `FedPop-G`.

Algorithm 1: Federated Population-Based Hyperparameter Tuning.

```
Input: Number of active clients per round $K$, number of HP-configurations $N_c$, total communication budget $R_t$, communication
  budget for each HP-configuration $R_c$ (computed by $\frac{R_t}{N_c}$), perturbation interval for FedPop-G $T_g$, initial model weight $\boldsymbol{w}$, $N_c$
  server HP-vectors $\boldsymbol{\alpha} = \{\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_{N_c}\}$, $N_c$ client HP-vectors $\boldsymbol{\beta} = \{\boldsymbol{\beta}_1^0, ..., \boldsymbol{\beta}_{N_c}^0\}$.
Copy the model weights $\boldsymbol{w}_i \leftarrow \boldsymbol{w}$ for all $N_c$ tuning processes.
for comm. round $r \leftarrow 1$ to $R_c$ do
    for $i \leftarrow 1$ to $N_c$ do
        // in parallel
        if $len(\boldsymbol{\beta}_i) == 1$ then
            Randomly sample $\{\boldsymbol{\beta}_i^k\}_{k=1}^K$ inside $\Delta$-ball of $\boldsymbol{\beta}_i^0$.
        for Client $k \leftarrow 1$ to $K$ do
            // in parallel
            $\boldsymbol{w}_i^k \leftarrow \texttt{Loc}(\boldsymbol{\beta}_i^k, \boldsymbol{w}_i, T^k)$
            $s_i^k \leftarrow \texttt{Val}(\boldsymbol{w}_i^k, V^k)$
        $\boldsymbol{\beta}_i \leftarrow \texttt{FedPop-L}\ (\boldsymbol{\beta}_i, \{s_i^k\}_{k=1}^K, K)$
        $\boldsymbol{w}_i \leftarrow \texttt{Agg}(\boldsymbol{\alpha}_i, \boldsymbol{w}_i, \{\boldsymbol{w}_i^k\})$
        $s_i \leftarrow \frac{1}{K}\sum_{k=1}^K s_i^k$
    if $r\%T_g = 0$ then
        $\{\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i, \boldsymbol{w}_i\}_{i=1}^{N_c} \leftarrow \texttt{FedPop-G}$
          $(\{\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i, \boldsymbol{w}_i, s_i\}_{i=1}^{N_c}, N_c)$
return $\{\boldsymbol{w}_i\}_{i=1}^{N_c}$

Function FedPop-L($\boldsymbol{\beta}, \boldsymbol{s}, K$)
    $\boldsymbol{P}_b \leftarrow \{k : s^k \geq \frac{\rho-1}{\rho}\text{-quantile}(\{s^k\})\}$
    $\boldsymbol{P}_t \leftarrow \{k : s^k \leq \frac{1}{\rho}\text{-quantile}(\{s^k\})\}$
    for $k_b \in \boldsymbol{P}_b$ do
        Sample $k_t$ from $\boldsymbol{P}_t$.
        Delete $\boldsymbol{\beta}^{k_b}$.
        $\boldsymbol{\beta}^{k_b} \leftarrow \texttt{Evo}(\boldsymbol{\beta}^{k_t})$
    return $\boldsymbol{\beta}$

Function FedPop-G($\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{w}, \boldsymbol{s}, N_c$)
    $\boldsymbol{Q}_b \leftarrow \{i : s_i \geq \frac{\rho-1}{\rho}\text{-quantile}(\{s_i\}\}$
    $\boldsymbol{Q}_t \leftarrow \{i : s_i \leq \frac{1}{\rho}\text{-quantile}(\{s_i\})\}$
    for $i_b \in \boldsymbol{Q}_b$ do
        Sample $i_t$ from $\boldsymbol{Q}_t$.
        Delete $\boldsymbol{\alpha}_{i_b}, \boldsymbol{\beta}_{i_b}, \boldsymbol{w}_{i_b}$.
        $\boldsymbol{\alpha}_{i_b}, \boldsymbol{\beta}_{i_b}^0 \leftarrow \texttt{Evo}(\boldsymbol{\alpha}_{i_t}, \boldsymbol{\beta}_{i_t}^0)$
        $\boldsymbol{w}_{i_b} \leftarrow \boldsymbol{w}_{i_t}$
    return $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{w}$
```

conducted with an interval of T_g communication rounds. Hereby, the list of scores s_i over T_g rounds are recorded and their weighted sum with a power-law weight decay (γ_g) is utilized as the final measurement:

$$s_i = \frac{\sum_{r=1}^{T_g} \gamma_g^{T_g - r} \cdot s_i^{(r)}}{\sum_{r=1}^{T_g} \gamma_g^{T_g - r}}. \tag{5}$$

The tuning procedure starts by sorting the HP-configurations according to their validation scores. Afterwards, 2 subsets, i.e., $\boldsymbol{Q}_b$ and $\boldsymbol{Q}_t$, are constructed, representing the indices of the bottom and top $\frac{1}{\rho}$-quantile of the HP-configurations, respectively. Finally, the HP-configurations with indices in $\boldsymbol{Q}_b$ will be replaced by the perturbed version of the HP-configurations with indices in $\boldsymbol{Q}_t$. Specifically, $\boldsymbol{\alpha}_{i_b}, \boldsymbol{\beta}_{i_b}^0$ are replaced by the perturbed version of $\boldsymbol{\alpha}_{i_t}, \boldsymbol{\beta}_{i_t}^0$ via Evo (Equation 3), the model weight in i_b-*th* HP-configuration ($\boldsymbol{w}_{i_b}$) are replaced by the i_t-*th* ($\boldsymbol{w}_{i_t}$).

FedPop-L for Intra-configuration Tuning: To further explore the local neighborhood of $\boldsymbol{\beta}_i^0$ for client local update in a fine-grained manner, we apply FedPop-L inside each tuning process. Hereby, we provide an informative assessment of $\boldsymbol{\beta}_i^0$ and its local neighborhood to enhance the robustness of HP-configuration. For simplicity, we omit i in the following notations. We consider the base HP-vector $\boldsymbol{\beta}^0$ as the perturbation center and restrict the perturbated HP-vector to lie inside a Δ-ball of it, i.e., $||\boldsymbol{\beta}^k - \boldsymbol{\beta}^0||_2 \leq \Delta$. At each communication round, $\boldsymbol{\beta}^k$ will be assigned to Loc of the k_{th} active client, the validation loss of the optimized model $\boldsymbol{w}^k$ will be recorded as the score s^k for HP-vector $\boldsymbol{\beta}^k$. Afterwards, $\{\boldsymbol{\beta}^k\}_{k=1}^K$ will be sorted according to the validation scores and separated into 2 subsets, containing the indices of the bottom ($\boldsymbol{P}_b$) and the top ($\boldsymbol{P}_t$) $\frac{1}{\rho}$-quantile of the $\boldsymbol{\beta}$, respectively. Finally, the HP-vectors $\boldsymbol{\beta}^{k_t}$ with indices in $\boldsymbol{P}_t$ will be perturbed to replace the HP-vectors $\boldsymbol{\beta}^{k_b}$ with indices in $\boldsymbol{P}_b$ via Evo.

Solutions to Challenges: (**C1**) FedPop does not require Bayesian Optimization (Zhou et al. 2023) or gradient-based hyperparameter optimization (Khodak et al. 2021), which saves the communication and computation costs. Besides, FedPop utilizes an *online* evolutionary method (Evo) to update the hyperparameters, i.e., not "training-after-tuning" but "tuning-while-training", which eliminates the need for "retraining" after finding a promising HP-configuration. Note that all procedures in FedPop can be conducted in a parallel and asynchronous manner. (**C2**) FedPop-G is conducted every T_g communication rounds to mitigate the noise depicted in the validation scores of HP-configurations. Besides, FedPop-L dynamically searches and evaluates the local neighborhood of $\boldsymbol{\beta}$, providing a more informative guidance for the client local HP optimization. Consequently, by enhancing the robustness of $\boldsymbol{\beta}$ to HP perturbation, we aim at improving its robustness against client data Non-IIDness.

Experiments and Analyses

We conduct an extensive empirical analysis to investigate the proposed method and its viability. Firstly, we compare FedPop with the SOTA and other baseline methods on three common FL benchmarks following (Khodak et al. 2021). Subsequently, we validate our approach by tuning hyperparameters for complex real-world cross-silo FL settings. Besides, we conduct an ablation study on FedPop to demonstrate the importance of its components. Moreover,

Tuning Wrapper	Tuning Algorithm	CIFAR-10			FEMNIST		Shakespeare	
		IID	NIID ($Dir_{1.0}$)	NIID ($Dir_{0.5}$)	IID	NIID	IID	NIID
RS	None	69.04 ±7.38 (65.28 ±5.83)	63.47 ±3.14 (60.51 ±8.03)	62.88 ±8.13 (60.65 ±7.37)	82.86 ±1.24 (83.76 ±3.56)	79.06 ±5.59 (83.09 ±2.64)	33.76 ±11.27 (31.19 ±10.18)	32.67 ±12.27 (31.32 ±9.92)
	FedEx	67.91 ±7.15 (64.21 ±7.84)	64.34 ±5.28 (62.97 ±7.27)	63.22 ±7.13 (61.92 ±8.06)	82.84 ±0.80 (82.57 ±3.25)	82.14 ±1.60 (84.03 ±2.48)	42.68 ±7.24 (41.22 ±6.34)	44.28 ±8.78 (46.69 ±7.39)
	FedPop	**71.18** ±4.68 (**68.01** ±3.42)	**68.25** ±5.03 (**65.74** ±3.97)	**67.01** ±4.98 (**65.24** ±3.97)	**84.33** ±1.41 (**85.99** ±1.62)	**83.21** ±2.08 (**85.48** ±1.48)	**44.30** ±3.37 (**44.46** ±3.53)	**47.28** ±3.47 (**50.25** ±3.87)
SHA	None	78.57 ±2.39 (75.93 ±4.96)	70.37 ±5.03 (67.83 ±4.41)	68.65 ±4.68 (65.58 ±8.10)	83.81 ±0.45 (85.52 ±1.63)	80.62 ±2.88 (87.64 ±0.64)	52.23 ±2.54 (49.06 ±5.98)	51.68 ±0.95 (48.83 ±3.12)
	FedEx	79.83 ±2.59 (77.04 ±1.45)	72.02 ±4.91 (70.81 ±4.65)	69.69 ±7.03 (67.02 ±7.65)	81.19 ±3.24 (85.69 ±1.91)	82.76 ±0.54 (86.79 ±2.89)	51.79 ±1.25 (51.89 ±1.30)	51.26 ±2.73 (51.01 ±3.36)
	FedPop	**81.47** ±1.24 (**78.96** ±0.87)	**76.42** ±3.04 (**75.03** ±2.56)	**74.88** ±2.06 (**72.41** ±1.87)	**84.33** ±0.57 (**86.84** ±0.98)	**83.26** ±0.86 (**88.33** ±0.79)	**53.48** ±0.57 (**52.66** ±1.91)	**53.07** ±0.97 (**52.79** ±0.36)

Table 2: Evaluation results of different hyperparameter tuning algorithms on three benchmark datasets. We report the *global* and locally *finetuned* (in the brackets) model performance with mean ±std from 5-trial runs. The best results are marked in **bold**.

we present convergence analysis of FedPop and its promising scalability by training ResNets from scratch on *full-sized* ImageNet-1K with Non-IID label distribution. Finally, we demonstrate the applicability of FedPop when combined with different federated optimization methods.

Benchmark Experiments

Datasets Description We conduct experiments on three benchmark datasets on both vision and language tasks: (1) *CIFAR-10* (Krizhevsky, Hinton et al. 2009), which is an image classification dataset containing 10 categories of real-world objects. (2) *FEMNIST* (Caldas et al. 2018), which includes gray-scale images of hand-written digits and English letters, producing a 62-way classification task. (3) *shakespeare* (Caldas et al. 2018) is a next-character prediction task and comprises sentences from Shakespeare's Dialogues.

We investigate 2 different partitions of the datasets: (1) For i.i.d (*IID*) setting, we randomly shuffle the dataset and evenly distribute the data to each client. (2) For non-i.i.d (*NIID*) settings, we follow (Khodak et al. 2021; Caldas et al. 2018) and assume each client contains data from a specific writer in FEMNIST, or it represents an actor in Shakespeare. For CIFAR-10 dataset, we follow prior arts (Zhu, Hong, and Zhou 2021; Lin et al. 2020) to model Non-IID label distributions using Dirichlet distribution Dir_x, in which a smaller x indicates higher data heterogeneity. We set the communication budget (R_t, R_c) to $(4000, 800)$ for CIFAR-10 and shakespeare, while $(2000, 200)$ for FEMNIST following (Khodak et al. 2021; Caldas et al. 2018). Besides, We adopt 500 clients for CIFAR-10, 3550 clients for FEMNIST, and 1129 clients for Shakespeare. For the coefficients used in FedPop, we set the initial perturbation intensity ϵ^0 to 0.1, the initial resampling probability p_{re}^0 to 0.1, and the quantile coefficient ρ to 3. The perturbation interval T_g for FedPop-G is set to $0.1R_c$. Following (Khodak et al. 2021), we define $\boldsymbol{\alpha} \in \mathbb{R}^3$ and $\boldsymbol{\beta} \in \mathbb{R}^7$, i.e., we tune learning rate, scheduler, and momentum for server-side aggregation (Agg), and learning rate, scheduler, momentum, weight-decay, the number of local epochs, batch-size, and dropout rate for local clients updates (Loc), respectively. More details about the HP search space, dataset descriptions, and model architectures are provided in Appendix.

Results and Discussion In Table 2, we report the testing accuracy achieved by the final model after performing hyperparameter tuning with different algorithms on three benchmarks. Hereby, we report the results of the *global* model, which is the server model $\boldsymbol{w}$ after the execution of the final communication round, and the *finetuned* model (in the brackets), which is the final global model finetuned on clients local data via Loc($\boldsymbol{\beta}^0, \boldsymbol{w}, T^k$). We observe that FedPop, combined with either RS or SHA as a wrapper, outperforms all the competitors on all benchmarks. For IID settings, the global model tuned on CIFAR-10 with FedPop, with RS or SHA as a wrapper, outperforms the baseline by 2.14% and 2.90%, respectively. Likewise, FedPop yields the highest average accuracy on FEMNIST and Shakespeare. For Non-IID settings, FedPop achieves a significant improvement of 3.85% and 4.79% on average compared with FedEx in CIFAR-10, when combined with RS and SHA, respectively. Moreover, we find that the performance improvement of the finetuned model using FedPop surpasses the other baselines. Additionally, we observe that during the tuning procedures, certain trials in the baselines and FedEx fail to converge. We attribute this to their predefined and fixed hyperparameter search spaces and values, resulting in higher sensitivity to the hyperparameter initialization that could not be mitigated during the tuning process. This phenomenon is observed via their larger accuracy deviation compared with FedPop, which further highlights the tuning stability of FedPop.

Validation on Real-World Cross-Silo FL Systems

As described in Section , previous hyperparameter tuning algorithms focused on small-scale benchmarks and simple model architectures. To indicate the effectiveness of FedPop on real-world FL applications, we further conduct experiments on three large-scale benchmarks: (1) PACS (Li et al. 2017), which includes images that belong to 7 classes from 4 domains Art-Painting, Cartoon, Photo, and Sketch.

Tuning Algorithm	PACS	OfficeHome	DomainNet
SHA	68.71 ±7.38 (76.53 ±12.54)	38.65 ±14.82 (57.64 ±12.21)	71.41 ±6.56 (79.41 ±11.81)
FedEx	73.47 ±3.06 (80.61 ±5.68)	42.99 ±8.72 (58.40 ±10.77)	71.68 ±6.13 (78.96 ±10.71)
FedPop	**75.17** ±1.18 (**85.37** ±2.12)	**45.71** ±7.64 (**62.76** ±7.38)	**73.59** ±3.58 (**81.78** ±3.14)

Table 3: Evaluation results on three real-world cross-silo FL benchmarks with feature space distribution shifts.

(2) OfficeHome (Venkateswara et al. 2017), which contains 65 different real-world objects in 4 styles: Art, Clipart, Product, and Real. (3) DomainNet (Peng et al. 2019), which is collected under 6 different data sources: Clipart, Infograph, Painting, Quickdraw, Real, and Sketch. All images are reshaped with larger sizes, i.e., 224x224. Following the setting proposed by (Li et al. 2021; Chen et al. 2023), we apply cross-silo (Li et al. 2020) FL settings and assume each client contains data from one of the sources (domains), but there exist feature distributions shift across different clients (feature space NIID (Li et al. 2021)). We use a more complex network architecture, i.e., ResNet-18, as the classification backbone. We set the tuning budget (R_t, R_c) to $(1000, 200)$. More details about the settings are provided in Appendix.

In Table 3, we report the evaluation results of the target model after tuning by `SHA` or its combination with `FedEx` or `FedPop`. We highlight the performance improvements achieved by the proposed method compared with the competitors, where `FedPop` surpasses the others up to 2.72% and indicates smaller accuracy deviations. These results indicate the effectiveness of `FedPop` on real-world FL scenarios with a smaller number of clients, large-scale private datasets, and more complex network architectures.

Ablation Study

To illustrate the importance of different `FedPop` components, we conduct an ablation study on CIFAR-10 benchmark considering *IID* and *NIID* settings. The results are shown in Table 4. We first notice that applying only one population-based tuning algorithm, i.e., either `FedPop-L` or `FedPoP-G`, already leads to distinct performance improvements on the baselines, especially when the client's data are *Non-IID*. Moreover, employing both functions together significantly improves the tuning results, which demonstrates their complementarity.

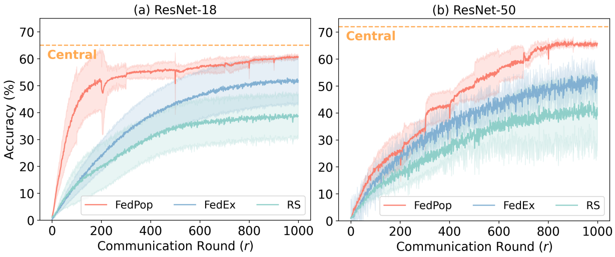


Figure 5: Convergence analysis of different tuning algorithms on full-sized *Non-IID* ImageNet-1k.

Tuning Algorithm	CIFAR-10		
	IID	NIID $(Dir_{1.0})$	NIID $(Dir_{0.5})$
RS	69.04 ±7.38	63.47 ±3.14	62.88 ±8.13
FedPop-G	70.61 ±3.21	66.81 ±3.24	65.63 ±4.67
FedPop-L	70.03 ±2.13	67.50 ±2.06	64.24 ±5.96
FedPop	**71.18** ±4.68	**68.25** ±5.03	**67.01** ±4.98

Table 4: Ablation study for different components in `FedPop` on CIFAR-10 benchmark.

Analysis on Full-sized NIID ImageNet-1k

To further demonstrate the scalability of `FedPop`, we display the convergence analysis of `FedPop` on *full-sized* ImageNet-1K, where we distribute the data among 100 clients with Non-IID label distributions using Dirichlet distribution $Dir_{1.0}$. Hereby, we set $(R_t, R_c) = (5000, 1000)$ and report the average local testing results of the active clients after communication round r. We provide more details about the experimental setup in Appendix.

As shown in Figure 5, we discover that `FedPop` already outperforms the others from the initial phase, indicating its promising convergence rate. Besides, we also observe a reduced performance variation in `FedPop`, which further substantiates the benefits of evolutionary updates in stabilizing the overall tuning procedure. Most importantly, `FedPop` achieves comparable results with centralized training of the networks, indicating its scalability to tuning HPs for large-scale FL applications.

Conclusion and Outlooks

IIn this study, we introduce a novel population-based algorithm, `FedPop`, designed for hyperparameter tuning in distributed federated learning (FL) systems. Unlike conventional ”training-after-tuning” approaches, `FedPop` adopts a ”tuning-while-training” paradigm, making it uniquely suited for FL applications. The algorithm leverages evolutionary updates to optimize hyperparameters based on the performance of population members at both the client and server levels. Its global component `FedPop-G`, is applicable for tuning hyperparameters used in both server aggregation and client local updates. For a more detailed tuning of hyperparameters specific to client updates, we apply the fine-grained component, `FedPop-L`. Empirical results demonstrate that `FedPop-G` achieves state-of-the-art performance across three widely used FL benchmarks, handling both IID and non-IID data distributions. Furthermore, its promising performance on real-world FL tasks with feature distribution shifts underscores its effectiveness for complex applications. Finally, experiments on large-scale FL systems, including full-sized non-IID ImageNet-1K, validate its scalability and practical utility for real-world scenarios.

References

Caldas, S.; Duddu, S. M. K.; Wu, P.; Li, T.; Konečnỳ, J.; McMahan, H. B.; Smith, V.; and Talwalkar, A. 2018. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*.

Chen, H.; Frikha, A.; Krompass, D.; Gu, J.; and Tresp, V. 2023. FRAug: Tackling federated learning with Non-IID features via representation augmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 4849–4859.

Dai, Z.; Low, B. K. H.; and Jaillet, P. 2020. Federated Bayesian optimization via Thompson sampling. *Advances in Neural Information Processing Systems*, 33: 9687–9699.

Dai, Z.; Low, B. K. H.; and Jaillet, P. 2021. Differentially private federated Bayesian optimization with distributed exploration. *Advances in Neural Information Processing Systems*, 34: 9125–9139.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Garg, A.; Saha, A. K.; and Dutta, D. 2020. Direct federated neural architecture search. *arXiv preprint arXiv:2010.06223*.

He, C.; Annavaram, M.; and Avestimehr, S. 2020. Towards non-iid and invisible data with fednas: federated deep learning via neural architecture search. *arXiv preprint arXiv:2004.08546*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hutter, F.; Kotthoff, L.; and Vanschoren, J. 2019. *Automated machine learning: methods, systems, challenges*. Springer Nature.

Jaderberg, M.; Dalibard, V.; Osindero, S.; Czarnecki, W. M.; Donahue, J.; Razavi, A.; Vinyals, O.; Green, T.; Dunning, I.; Simonyan, K.; et al. 2017. Population based training of neural networks. *arXiv preprint arXiv:1711.09846*.

Kairouz, P.; McMahan, H. B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A. N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. 2021. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2): 1–210.

Khan, S.; Rizwan, A.; Khan, A. N.; Ali, M.; Ahmed, R.; and Kim, D. H. 2023. A multi-perspective revisit to the optimization methods of Neural Architecture Search and Hyper-parameter optimization for non-federated and federated learning environments. *Computers and Electrical Engineering*, 110: 108867.

Khodak, M.; Li, T.; Li, L.; Balcan, M.-F.; Smith, V.; and Talwalkar, A. 2020. Weight-Sharing for Hyperparameter Optimization in Federated Learning. In *Int. Workshop on Federated Learning for User Privacy and Data Confidentiality in Conjunction with ICML*, volume 2020.

Khodak, M.; Tu, R.; Li, T.; Li, L.; Balcan, M.-F. F.; Smith, V.; and Talwalkar, A. 2021. Federated hyperparameter tuning: Challenges, baselines, and connections to weight-sharing. *Advances in Neural Information Processing Systems*, 34: 19184–19197.

Koskela, A.; and Honkela, A. 2018. Learning rate adaptation for federated and differentially private learning. *arXiv preprint arXiv:1809.03832*.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Li, D.; Yang, Y.; Song, Y.-Z.; and Hospedales, T. M. 2017. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, 5542–5550.

Li, T.; Sahu, A. K.; Talwalkar, A.; and Smith, V. 2020. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3): 50–60.

Li, X.; Jiang, M.; Zhang, X.; Kamp, M.; and Dou, Q. 2021. Fedbn: Federated learning on non-iid features via local batch normalization. *arXiv preprint arXiv:2102.07623*.

Liang, J.; Meyerson, E.; Hodjat, B.; Fink, D.; Mutch, K.; and Miikkulainen, R. 2019. Evolutionary neural automl for deep learning. In *Proceedings of the Genetic and Evolutionary Computation Conference*, 401–409.

Lin, T.; Kong, L.; Stich, S. U.; and Jaggi, M. 2020. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33: 2351–2363.

Liu, H.; Simonyan, K.; Vinyals, O.; Fernando, C.; and Kavukcuoglu, K. 2017. Hierarchical representations for efficient architecture search. *arXiv preprint arXiv:1711.00436*.

Maumela, T.; Nelwamondo, F.; and Marwala, T. 2022. Population based training and federated learning frameworks for hyperparameter optimisation and ML unfairness using Ulimisana Optimisation Algorithm. *Information Sciences*, 612: 132–150.

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

Mlodozeniec, B.; Reisser, M.; and Louizos, C. 2023. Hyperparameter Optimization through Neural Network Partitioning. *arXiv preprint arXiv:2304.14766*.

Mostafa, H. 2019. Robust federated learning through representation matching and adaptive hyper-parameters. *arXiv preprint arXiv:1912.13075*.

Parker-Holder, J.; Nguyen, V.; and Roberts, S. J. 2020. Provably efficient online hyperparameter optimization with population-based bandits. *Advances in Neural Information Processing Systems*, 33: 17200–17211.

Peng, X.; Bai, Q.; Xia, X.; Huang, Z.; Saenko, K.; and Wang, B. 2019. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, 1406–1415.

Real, E.; Liang, C.; So, D.; and Le, Q. 2020. Automl-zero: Evolving machine learning algorithms from scratch. In *International Conference on Machine Learning*, 8007–8019. PMLR.

Real, E.; Moore, S.; Selle, A.; Saxena, S.; Suematsu, Y. L.; Tan, J.; Le, Q. V.; and Kurakin, A. 2017. Large-scale evolution of image classifiers. In *International Conference on Machine Learning*, 2902–2911. PMLR.

Reddi, S.; Charles, Z.; Zaheer, M.; Garrett, Z.; Rush, K.; Konečnỳ, J.; Kumar, S.; and McMahan, H. B. 2020. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*.

Seng, J.; Prasad, P.; Dhami, D. S.; and Kersting, K. 2022. HANF: Hyperparameter And Neural Architecture Search in Federated Learning. *arXiv preprint arXiv:2206.12342*.

Snoek, J.; Larochelle, H.; and Adams, R. P. 2012. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25.

Tarzanagh, D. A.; Li, M.; Thrampoulidis, C.; and Oymak, S. 2022. FedNest: Federated bilevel, minimax, and compositional optimization. In *International Conference on Machine Learning*, 21146–21179. PMLR.

Telikani, A.; Tahmassebi, A.; Banzhaf, W.; and Gandomi, A. H. 2021. Evolutionary machine learning: A survey. *ACM Computing Surveys (CSUR)*, 54(8): 1–35.

Venkateswara, H.; Eusebio, J.; Chakraborty, S.; and Panchanathan, S. 2017. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5018–5027.

Wang, S.; Tuor, T.; Salonidis, T.; Leung, K. K.; Makaya, C.; He, T.; and Chan, K. 2019. Adaptive federated learning in resource constrained edge computing systems. *IEEE journal on selected areas in communications*, 37(6): 1205–1221.

Xu, M.; Zhao, Y.; Bian, K.; Huang, G.; Mei, Q.; and Liu, X. 2020. Federated neural architecture search. *arXiv preprint arXiv:2002.06352*.

Zhang, H.; Zhang, M.; Liu, X.; Mohapatra, P.; and DeLucia, M. 2022. Fedtune: Automatic tuning of federated learning hyper-parameters from system perspective. In *MILCOM 2022-2022 IEEE Military Communications Conference (MILCOM)*, 478–483. IEEE.

Zhou, Y.; Ram, P.; Salonidis, T.; Baracaldo, N.; Samulowitz, H.; and Ludwig, H. 2023. Single-shot general hyper-parameter optimization for federated learning. In *The Eleventh International Conference on Learning Representations*.

Zhu, Z.; Hong, J.; and Zhou, J. 2021. Data-free knowledge distillation for heterogeneous federated learning. In *International Conference on Machine Learning*, 12878–12889. PMLR.

Notation Summary

In Table 1, we provide the detailed notations and their explanations used in the main paper.

FedPop Details

For completeness, we present the pseudo code of FedPop wrapped with SHA in Algorithm 1. We set the number of active clients at each communication round to 10. We execute 3 times elimination and set the number of initial HP-configurations to 27 for SHA following (Khodak et al. 2021). The power-law decay used in the computation of the weighted sum for the validation scores list $\{s_i\}$ is described as following:

$$s = \frac{\sum_{r=1}^{R} 0.9^{R-r} \cdot s_i^r}{\sum_{r=1}^{R} 0.9^{R-r}}, \tag{1}$$

where r is index of score in the score list, R is the length of the validation score list.

The number of tried α and β

In this section, we provide the computation of the numbers of tried $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ shown in Figure 2 of the main paper. Specifically, we set $N_c = 5$ for RS and $N_c = 27$ for SHA, where each tuning process is assigned with one HP-configuration, i.e., one $\boldsymbol{\alpha}$ and one $\boldsymbol{\beta}$. For FedEx wrapped with RS, we follow the settings provided in the original paper and assign each tuning process one HP-configuration and 27 additional $\boldsymbol{\beta}$, which leads to in total $(27 \times 5 =)135$ tried $\boldsymbol{\beta}$. For FedPop wrapped with RS, we provide the computation of the numbers in the following:

$$\begin{aligned} \#\text{ of tried } \boldsymbol{\alpha} &= N_c + \frac{1}{\rho} N_c \cdot \frac{R_c}{T_g}, \\ \#\text{ of tried } \boldsymbol{\beta} &= N_c \cdot K + T_g \frac{K}{\rho}. \end{aligned} \tag{2}$$

Following the experimental settings described in the main paper, we observe that FedPop experiments more HP-vectors compared with other methods.

Annealing process of ϵ and p_{re}

In this section, we describe the cosine annealing process for the values of perturbation intensity ϵ and resampling probability p_{re} described in the main paper. For ϵ, we apply

$$\epsilon = \begin{cases} \frac{\epsilon_0}{2} \cdot (1 + cos(\pi \frac{r}{r_0})), & r < r_0 \\ 0, & r \geq r_0 \end{cases} \tag{3}$$

where ϵ_0 is set to 0.1 for all experiments. For ϵ used in FedPop-L, we set $r_0 = 0.2T_g$. Specifically, we stop the local search of β after the first $0.2T_g$ communication rounds of a newly initialized (perturbed) HP-configuration to save local computation costs at each client. We observe that this early-stopping of FedPop-L leads to comparable results as executing FedPop-L for all rounds. Therefore, we apply this strategy to save local computational costs without huge performance decrease. For ϵ used in FedPop-G, we set $r_0 = R_c$, i.e., FedPop-G will be executed throughout the overall tuning process.

For p_{re}, we apply

$$p_{re} = \frac{p_{re}^0}{2} \cdot (1 + cos(\pi \frac{r}{R_c})) \tag{4}$$

where p_{re}^0 is set to 0.1 for all experiments.

Local search space for β_i^k

In this section, we describe the process of the selection criterion of local search space for $\boldsymbol{\beta}_i^k$ in the i-th HP-configuration. Following previous work (Khodak et al. 2021), we sample $\boldsymbol{\beta}_i^k$ inside a Δ-ball centered by $\boldsymbol{\beta}_i^0$. Specifically, for hyperparameters sampled from discrete uniform distribution, e.g., $epoch\_num$, we define the search space as its neighboring discrete values, i.e., $\{x_{j-1}, x_j, x_{j+1}\}$, where j is the index of the current value. For hyperparameters sampled from continuous uniform distribution, e.g., $learning\_rate$, we define the search space as $[x_j - 0.2(b - a), x_j + 0.2(b-a)]$, where a and b are the upper- and lower-bound of the original distribution.

Experimental Details

Visualization of Benchmark Datasets

In this section, we show example images in different domains from the adopted benchmark datasets, i.e., PACS (Figure 5a), OfficeHome (Figure 5b), and DomainNet (Figure 5c). We can see that there exists strong appearance variation and distribution shifts across different domains, e.g., in PACS and DomainNet there exists both photo-like realistic pictures (*Photo*) and highly abstract human sketches (*Sketch*). Therefore, by assigning data from one of the domains to each client, we are able to simulate the experimental setting with features distribution shift in FL.

ImangeNet-1k Experimental Setup

In this section, we provide more details for our analysis on ImageNet-1K (Deng et al. 2009) dataset. We first split the original training set into training and validation set with a ratio of 9:1 for our experiment and use the original validation set as the testing set since the original test set is unlabelled. Afterwards, we split the training and validation set using the Dirichlet distribution with coefficient of 1.0. Here, we split the data into 100 subsets and assigning each subset to one client, leading to 100 clients joining the FL. We set the active clients per communication round as 10 and use the same hyperparameter search space as other datasets. For the centralized training, we adopt the hyperparameters used in the PyTorch repository *https://github.com/pytorch/examples/tree/main/imagenet*.

Hyperparameter Search Space

For all optimization, we use stochastic gradient descent (SGD) optimizer. We sample all hyperparameters from Uniform distribution (U), where $U\{...\}$ indicates discrete distribution and $U[a,b]$ indicates continuous distribution. The

Table 1: Summary of notations used in the main paper.

Method	Parameter	Explanation
Hyperparameter Tuning System	Agg	Server aggregation function
	Loc	Client local update function
	$\boldsymbol{\alpha}, \boldsymbol{\beta}$	HP-vector used in Agg, Loc
	$Concat(\boldsymbol{\alpha}, \boldsymbol{\beta})$	HP-configuration
	H_a, H_b	Distribution for HP-vector $\boldsymbol{\alpha}, \boldsymbol{\beta}$
	$\boldsymbol{w}$	Model weight
	C	# of total clients
	K, k	# of active clients, index of client
	T^k, V^k, E^k	Training, validation and testing subset
	R_t	Total tuning budget
	N_c	Number of initial HP-configurations
	R_c	Tuning budget for each HP-configuration
SHA	μ	Quantile for elimination
	$\{R_0, R_1, ..., R_E\}$	Communication round indices for elimination
FedPop	s_i^k	Validation score for k_{th} active client in i_{th} tuning process
	ϵ	Perturbation intensity in Evo
	p_{re}	Probability for resampling a hyperparameter from its original distribution
	ρ	Quantile for Evo
	Q_t, Q_b	Indices of HP-configurations within the top, bottom-performing quantile
	P_t, P_b	Indices of active clients within the top, bottom-performing quantile
	T_g	Perturbation interval for FedPop-G

Algorithm 1: FedPop wrapped with SHA.

```
Input: Number of active clients per round $K$, number of HP-configurations $N_c$, maximum communication budget for each
  HP-configuration $R_c$, perturbation interval for FedPop-G $T_g$, model weight $\boldsymbol{w}$, $N_c$ server HP-vectors $\boldsymbol{\alpha} = \{\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_{N_c}\}$, $N_c$
  client HP-vectors $\boldsymbol{\beta} = \{\boldsymbol{\beta}_1^0, ..., \boldsymbol{\beta}_{N_c}^0\}$, elimination rate $\eta_{sha} \in \mathcal{N}$, elimination rounds $\{R_0 = 0, R_1, ..., R_E\}$
Copy the model weights $\boldsymbol{w}_i \leftarrow \boldsymbol{w}$ for all $N_c$ tuning processes.
for elim. step $t \leftarrow 1$ to $E$ do
    for comm. round $r \leftarrow R_{t-1}$ to $R_t$ do
        for $i \leftarrow 1$ to $N_c$ do
            // in parallel
            if $len(\boldsymbol{\beta}_i) == 1$ then
                Randomly sample $\{\boldsymbol{\beta}_i^k\}_{k=1}^K$ inside $\Delta$-ball of $\boldsymbol{\beta}_i^0$.
            for Client $k \leftarrow 1$ to $K$ do
                // in parallel
                $\boldsymbol{w}_i^k \leftarrow$ Loc$(\boldsymbol{\beta}_i^k, \boldsymbol{w}_i, T^k)$
                $s_i^k \leftarrow$ Val$(\boldsymbol{w}_i^k, V^k)$
            $\boldsymbol{\beta}_i \leftarrow$ FedPop-L $(\boldsymbol{\beta}_i, \{s_i^k\}_{k=1}^K, K)$
            $\boldsymbol{w}_i \leftarrow$ Agg$(\boldsymbol{\alpha}_i, \boldsymbol{w}_i, \{\boldsymbol{w}_i^k\}_{k=1}^K)$
            $s_i \leftarrow \frac{1}{K} \sum_{k=1}^K s_i^k$
        if $r \% T_g = 0$ then
            $\{\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i, \boldsymbol{w}_i\}_{i=1}^{N_c} \leftarrow$ FedPop-G $(\{\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i, \boldsymbol{w}_i, s_i\}_{i=1}^{N_c}, N_c)$
    $\{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{w}\} \leftarrow \{\{\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i, \boldsymbol{w}_i\} : s_i \leq \frac{1}{\eta_{sha}}$-quantile$(\{s_i\}_{i=1}^{N_c})\}$
    $N_c \leftarrow \frac{N_c}{\eta_{sha}}$
return $\boldsymbol{w}$
```

HP-distributions used for server HP-vector ($\boldsymbol{\alpha}$) is listed in the following:

$$\begin{aligned} log_{10}\text{lr}: &\quad U[-1, 1] \\ \text{momentum}: &\quad U[0, 0.9] \\ log_{10}(1-\gamma): &\quad U[-4, -2] \end{aligned}$$

where γ is the multiplicative factor of lr decay. The HP-distributions used for client local HP-vector ($\boldsymbol{\beta}$) is listed in the following:

$$\begin{aligned} log_{10}\text{lr}: &\quad U[-4, 0] \\ \text{momentum}: &\quad U[0, 1.0] \\ log_{10}(\lambda): &\quad U[-5, -1] \\ \text{epoch}: &\quad U\{1, .., 5\} \\ log_{2}(\text{batch}): &\quad U\{3, .., 7\} \\ \text{dropout}: &\quad U[0, 0.5] \end{aligned}$$

where λ is the weight decay for SGD optimizer. All experiments are executed in the GPU GeForce GTX TITAN X with 12GB memory.

For the experiments of tuning HPs for FedProx (Li et al. 2020), we additionally add a hyperparameter in $\boldsymbol{\beta}$ with the initial distribution $U[0, 1.0]$ to control the strength for the regularization proximal term. For SCAFFOLD (Karimireddy et al. 2020), we add additional learning rate and scheduler, i.e., $log_{10}(lr\_control) : U[-1, 1]$ and $log_{10}(1 - \gamma\_control) : U[-4, -2]$ in $\boldsymbol{\alpha}$ for the optimization of control variants.

Model Architecture and Dataset Details

In this section, we provide details about the model architecture used for different benchmark datasets. We adopt 500 clients for CIFAR-10, 3550 clients for FEMNIST, and 1129 clients for Shakespeare datasets. The settings we adopt are following previous work (Caldas et al. 2018; Khodak et al. 2021).

Following (Khodak et al. 2021), we use a 6-layer CNN with its details listed in Table 2, 3, and 4, for CIFAR-10, FEMNIST, and shakespeare dataset, respectively. For the convolutional layer (Conv2D), we list parameters with the sequence of input and output dimensions, kernel size, stride, and padding. For the max-pooling layer (MaxPool2D), we list kernel and stride. For the dropout layer (Dropout), we list dropout probability (hyperparameter in hyp-vector $\boldsymbol{\beta}$). For the fully-connected layer (FC), we list input and output dimensions. For the Batch Normalization layer (BN), we list the channel dimension. For the embedding layer (Embedding), we list the number of embedding and embedding dimension. For the LSTM layer (LSTM), we list the input dimension and hidden dimension.

Table 2: Model architecture for CIFAR-10.

Layer	Details

1	Conv2D(3, 32, 3, 1, 1)
	ReLU(), MaxPool2D(2, 2)

2	Conv2D(32, 64, 3, 1, 1)
	ReLU(), MaxPool2D(2, 2)

3	Conv2D(64, 64, 3, 1, 1)
	ReLU(), MaxPool2D(2, 2)

4	Dropout(p)

5	FC(1024, 64)
	ReLU()

6	FC(64, 10)

Table 3: Model architecture for FEMNIST.

Layer	Details

1	Conv2D(3, 32, 3, 1, 1)
	ReLU(), MaxPool2D(2, 2)

2	Conv2D(32, 64, 3, 1, 1)
	ReLU(), MaxPool2D(2, 2)

3	Conv2D(64, 64, 3, 1, 1)
	ReLU(), MaxPool2D(2, 2)

4	Dropout(p)

5	FC(9216, 1024)
	ReLU()

6	Dropout(p)
6	FC(1024, 62)

Table 4: Model architecture for shakespeare.

Layer	Details
1	Embedding(95, 8)
2	LSTM(8, 256)
3	FC(256, 10)

For the classification models on OfficeHome, PACS and DomainNet datasets, we use the widely adopted the backbone ResNet18 (He et al. 2016) and change the output dimension of the last fully-connected layer (FC) to match the number of categories in the dataset.

Additional Results

Comparison under Different System Designs

In this section, we analyze the tuning methods under different system designs. Hereby, we demonstrate the effectiveness of FedPop with different tuning budgets. To adapt the tuning process according to different R_t, we consider 2 possibilities of resource allocations: (1) Varying the number of tuning processes N_c from $\{5, 10, 15, 20\}$ and fixing the per process tuning budget R_c to 800 rounds (200 for FEMNIST). (2) Varying R_c and fixing N_c to 5 (10 for FEMNIST). Here, we select R_c from $\{200, 400, 800, 1600\}$ ($\{100, 200, 300, 400\}$ for FEMNIST).

We report the results in Figure 1. First, we observe that FedPop outperforms both FedEx and the baseline RS in all experimental setups, indicating its robustness against different system designs. Also, we observe that a larger communication budget per process R_c leads to better tuning results, while initializing more tuning processes (larger N_c) does not lead to obvious performance improvement. This re-

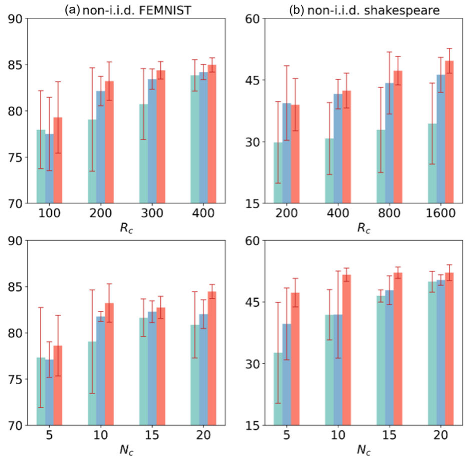


Figure 1: Evaluation results with varying the communication budget for each configuration R_c (top), and varying the number of tuning processes N_c (bottom).

veals the importance of having a sufficient tuning budget for each configuration.

Ablation Study on Score Computation

In Table 5, we provide an ablation study on the computation methods for the score list $\{s^i\}$. Hereby, we compare 3 different methods: (1) average (*avg*) which takes the average of all scores in the list: $s = \frac{\sum_{r=1}^{R} s_i^r}{R}$, (2) last score (*last*) which takes the last score in the list: $s = s_i^R$, and (3) power-lay decay which is described in Equation 1.

We observe that simply taking the average of the scores in the list leads to worse results leads to worse results. We assume this is due to the fact that the scores increase with higher rate at the initial stage. Using power-law decay generally leads to the most promising results among all the 3 methods.

Perturbation Intensity in `FedPop`

In this section, we analyze the impact of initial perturbation intensity used in `FedPop`, i.e., ϵ_0. Hereby, we select ϵ_0 from $\{10^{-2}, 10^{-1.5}, 10^{-1}, 10^{-0.5}, 10^{0}\}$. As shown in Figure 2, we observe that using smaller values of ϵ_0 leads to stable performance and smaller accuracy variations, where `FedPop` always outperforms the baseline `SHA`.

Table 5: Ablation study for different score computation methods in `FedPop`.

Computation Method	CIFAR-10		
	IID	Non-IID ($Dir_{1.0}$)	Non-IID ($Dir_{0.5}$)
avg	78.97 ±2.14 (75.84 ±1.41)	75.68 ±3.51 (73.72 ±1.68)	74.41 ±2.08 (72.54 ±1.99)
last	80.03 ±1.05 (77.12 ±2.47)	**77.58** ±1.68 (**75.46** ±0.56)	73.76 ±1.45 (71.88 ±0.99)
power-law	**81.47** ±1.24 (**78.96** ±0.87)	76.42 ±3.04 (75.03 ±2.56)	**74.88** ±2.06 (**72.41** ±1.87)

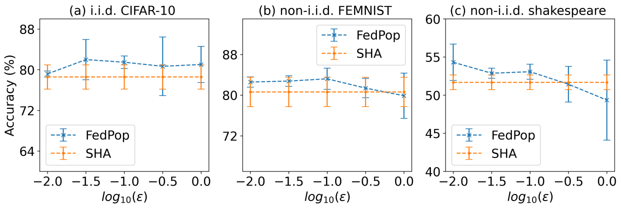


Figure 2: Effects analysis of initial perturbation intensity ϵ_0 in `FedPop`.

Perturbation Interval T_g for `FedPop-G`

In this section, we provide the results of `FedPop` with different choices of T_g for the perturbation interval of `FedPop-G`. We conduct the experiments on `FedPop` and `SHA` on i.i.d. CIFAR-10 and non-i.i.d. CIFAR-10. We select T_g from $\{5\%, 10\%, 15\%, 20\%\}$ of R_c (total communication budget of each HP-configuration). From the box plot in Figure 3, we observe that applying only limited numbers of `FedPop-G` already leads to promising results. Most importantly, `FedPop`, executing `FedPop-G` with different frequency, always outperforms the baseline method, indicating its promising performance.

Visualization of Hyperparameter Evolution

In Figure 4, we demonstrate the evolution of the client local hyperparameters, i.e., learning rate and momentum, in `FedPop-L`. We observe that with the help of our `Evo` function, both hyperparameter explores multiple possible values and eventually converges to an intermediate value. Note that these search steps are also executed when the local HP-vectors $\boldsymbol{\beta}$ are reinitialized via `FedPop-G` throughout the overall tuning process.

References

Caldas, S.; Duddu, S. M. K.; Wu, P.; Li, T.; Konečnỳ, J.; McMahan, H. B.; Smith, V.; and Talwalkar, A. 2018. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

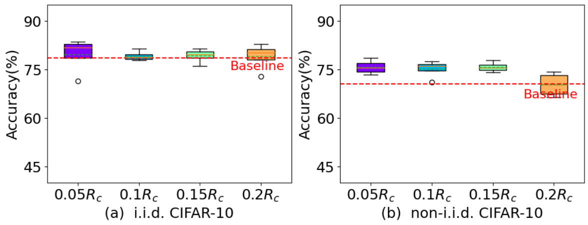


Figure 3: Effects analysis for T_g (evolutionary update frequency for `FedPop-G`) on CIFAR-10.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

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

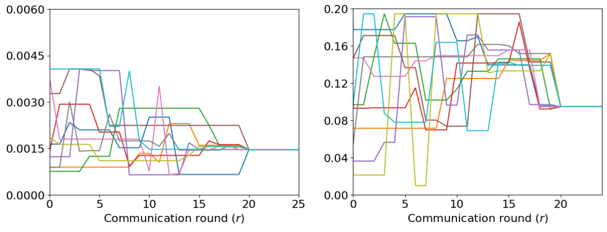
Khodak, M.; Tu, R.; Li, T.; Li, L.; Balcan, M.-F. F.; Smith, V.; and Talwalkar, A. 2021. Federated hyperparameter tuning: Challenges, baselines, and connections to weight-sharing. *Advances in Neural Information Processing Systems*, 34: 19184–19197.

Li, D.; Yang, Y.; Song, Y.-Z.; and Hospedales, T. M. 2017. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, 5542–5550.

Li, T.; Sahu, A. K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; and Smith, V. 2020. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2: 429–450.

Peng, X.; Bai, Q.; Xia, X.; Huang, Z.; Saenko, K.; and Wang, B. 2019. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, 1406–1415.

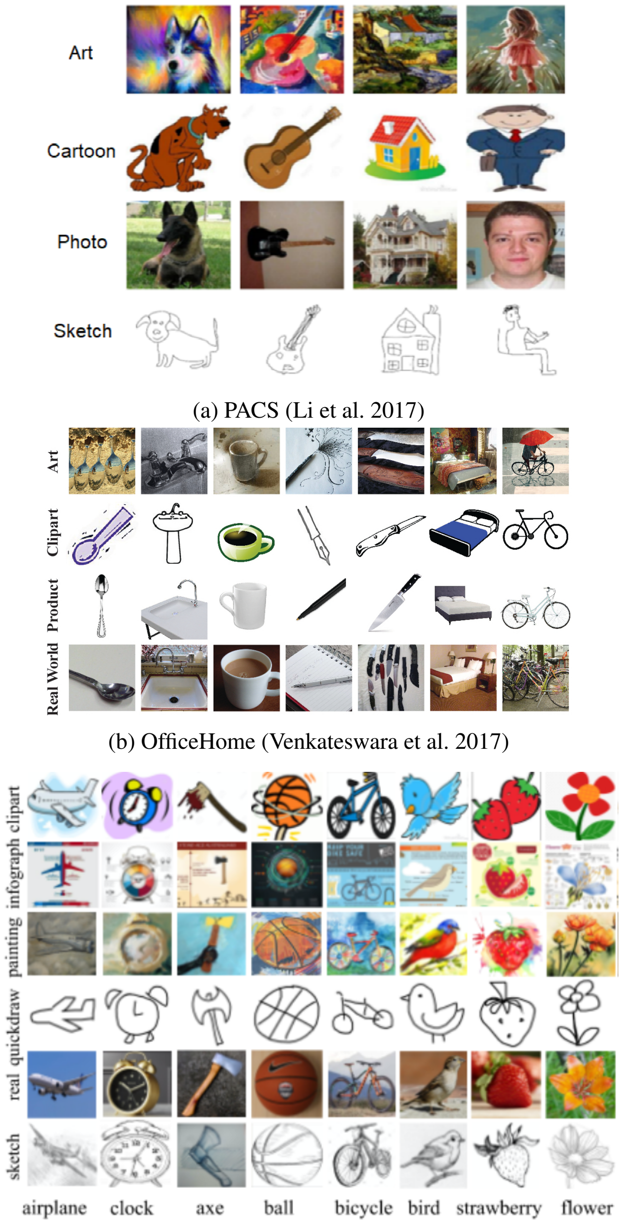
Venkateswara, H.; Eusebio, J.; Chakraborty, S.; and Panchanathan, S. 2017. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5018–5027.



(a) Learning rate

(b) Momentum

Figure 4: Visualization of HP evolution for the client local update.



(a) PACS (Li et al. 2017)

(b) OfficeHome (Venkateswara et al. 2017)

(c) DomainNet (Peng et al. 2019)

Figure 5: Example images from the selected cross-silo FL benchmark datasets with non-IID features. *Best viewed in color.*

Chapter 5

FedBiP: Heterogeneous One-Shot Federated Learning with Personalized Latent Diffusion Models

This chapter contains the publication

> **Haokun Chen**, Hang Li, Yao Zhang, Gengyuan Zhang, Jinhe Bi, Philip Torr, Jindong Gu, Denis Krompass, Volker Tresp. FedBiP: Heterogeneous One-Shot Federated Learning with Personalized Latent Diffusion Models.

FedBiP: Heterogeneous One-Shot Federated Learning with Personalized Latent Diffusion Models

Haokun Chen[1,2*] **Hang Li**[1,2] **Yao Zhang**[1,4] **Gengyuan Zhang**[1,4] **Jinhe Bi**[1]
Philip Torr[3] **Jindong Gu**[3*] **Denis Krompass**[2] **Volker Tresp**[1,4]
[1] Ludwig Maximilian University of Munich [2] Siemens Technology
[3] University of Oxford [4] Munich Center for Machine Learning

Abstract

One-Shot Federated Learning (OSFL), a special decentralized machine learning paradigm, has recently gained significant attention. OSFL requires only a single round of client data or model upload, which reduces communication costs and mitigates privacy threats compared to traditional FL. Despite these promising prospects, existing methods face challenges due to client data heterogeneity and limited data quantity when applied to real-world OSFL systems. Recently, Latent Diffusion Models (LDM) have shown remarkable advancements in synthesizing high-quality images through pretraining on large-scale datasets, thereby presenting a potential solution to overcome these issues. However, directly applying pretrained LDM to heterogeneous OSFL results in significant distribution shifts in synthetic data, leading to performance degradation in classification models trained on such data. This issue is particularly pronounced in rare domains, such as medical imaging, which are underrepresented in LDM's pretraining data. To address this challenge, we propose Federated Bi-Level Personalization (FedBiP), which personalizes the pretrained LDM at both instance-level and concept-level. Hereby, FedBiP synthesizes images following the client's local data distribution without compromising the privacy regulations. FedBiP is also the first approach to simultaneously address feature space heterogeneity and client data scarcity in OSFL. Our method is validated through extensive experiments on three OSFL benchmarks with feature space heterogeneity, as well as on challenging medical and satellite image datasets with label heterogeneity. The results demonstrate the effectiveness of FedBiP, which substantially outperforms other OSFL methods.

1 Introduction

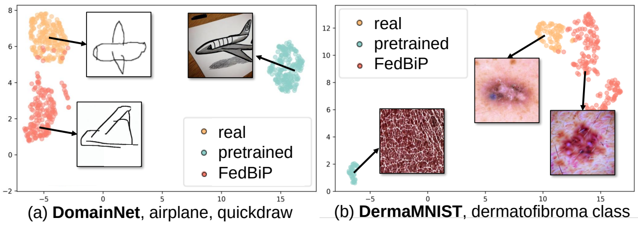
Federated Learning (FL) (McMahan et al., 2017) is a decentralized machine learning paradigm, in which multiple clients collaboratively train neural networks without centralizing their local data. However, traditional FL frameworks require frequent communication between a server and clients to transmit model weights, which would lead to significant communication overheads (Kairouz et al., 2021). Additionally, such frequent communication increases system susceptibility to privacy threats, as transmitted data can be intercepted by attackers who may then execute membership inference attacks (Lyu et al., 2020). In contrast, a special variant of FL, One-Shot Federated Learning (OSFL) (Guha et al., 2019), serves as a promising solution. OSFL requires only single-round server-client communication, thereby enhancing communication efficiency and significantly reducing the risk of interception by malicious attackers. Therefore, we focus on OSFL given its promising properties.

Despite these promising prospects, existing methods for OSFL face significant challenges when applied to real-world scenarios. Previous works (Guha et al., 2019; Li et al., 2020) require additional public datasets, presenting challenges in privacy-critical domains such as medical data (Liu et al., 2021), where acquiring data that conforms to client-specific distributions is often impractical. Alternatively, they can involve the transmission of entire model weights (Zhang et al., 2022) or local

[*] Corresponding to *haokun.chen@siemens.com* and *jindong.gu@outlook.com*

training data (Zhou et al., 2020), which are inefficient and increase the risk of privacy leakage. Moreover, these approaches overlook the issue of feature space heterogeneity, wherein the data features across different clients exhibit non-identically distributed properties. This presents an important and prevalent challenge as emphasized in (Li et al., 2021; Chen et al., 2023). Another vital challenge in (One-Shot) FL is the limited quantity of data available from clients (McMahan et al., 2017). This problem is particularly notable in specialized domains, such as medical or satellite imaging (So et al., 2022) where data collection is time-consuming and costly.

Data augmentation constitutes a promising strategy to address these challenges in traditional FL (Zhu et al., 2021; Li et al., 2022) by optimizing an auxiliary generative model. However, its reliance on multiple communication rounds makes it unsuitable for OSFL. Recently, diffusion models (Ho et al., 2020), particularly Latent Diffusion Model (LDM) (Rombach et al., 2022), have gained significant attention due to their capability to synthesize high-quality images after being pretrained on large-scale datasets. They are proven effective in various tasks, including training data augmentation (Yuan et al., 2023; Azizi et al., 2023) and addressing feature shift problems (Niemeijer et al., 2024; Gong et al., 2023) under centralized settings. However, directly applying a pretrained LDM for specialized domains presents challenges. As demonstrated in Figure 1, there is a noticeable feature distributional shift and visual discrepancy between real and synthetic data. This mismatch could lead to performance degradation when incorporating such synthetic data into the training process, especially in heterogeneous OSFL settings, where each client possesses data with varying distributions.



(a) **DomainNet**, airplane, quickdraw (b) **DermaMNIST**, dermatofibroma class

Figure 1: Feature map visualization of original client images (*real*), synthetic images by prompted pretrained LDM (*pretrained*), and our method (*FedBiP*) on two datasets. FedBiP effectively mitigates the strong distribution shifts between pretrained LDM and client local data.

Therefore, in this paper, we propose Federated Bi-Level Personalization (FedBiP), a framework designed to adapt pretrained LDM for synthesizing high-quality training data that adheres to client-specific data distributions in OSFL. FedBiP incorporates personalization of the pretrained LDM at both instance and concept levels. Specifically, instance-level personalization focuses on adapting the pretrained LDM to generate high-fidelity samples that closely align with each client's local data while preserving data privacy. Concurrently, concept-level personalization integrates category and domain-specific concepts from different clients to enhance data generation diversity at the central server. This bi-level personalization approach improves the performance of classification models trained on the synthesized data. Our contributions can be summarized as follows:

- We propose a novel method FedBiP to incorporate pretrained Latent Diffusion Model (LDM) for heterogeneous OSFL, marking the first OSFL framework to tackle feature space heterogeneity via personalizing LDM.
- We conduct comprehensive experiments on three OSFL benchmarks with feature space heterogeneity, in which FedBiP achieves state-of-the-art results.
- We validate the maturity and scalability of FedBiP on real-world medical and satellite image datasets with label space heterogeneity, and further demonstrate its promising capability in preserving client privacy.

2 Related Works

2.1 One-Shot Federated Learning

A variety of efforts have been made to address One-Shot Federated Learning (OSFL), primarily from two complementary perspectives: one focuses on model aggregation through techniques such as model prediction averaging (Guha et al., 2019), majority voting (Li et al., 2020), conformal prediction method (Humbert et al., 2023), loss surface adaptation (Su et al., 2023), or Bayesian methods (Yurochkin et al., 2019; Chen & Chao, 2020; Hasan et al., 2024). These approaches may not fully exploit the underlying knowledge across different client data distributions. Another aims to transmit training data instead of model weights: data distribution (Kasturi et al., 2020; Beitollahi et al., 2024; Shin et al., 2020), Generative Adversarial Networks (GANs) (Goodfellow et al., 2020; Zhang et al., 2022; Kasturi & Hota, 2023; Kang et al., 2023; Dai et al., 2024), or distilled dataset (Zhou et al., 2020; Song et al., 2023) are optimized and transmitted to the central server for subsequent model training. Given the success of diffusion models (Rombach et al., 2022), (Zhang et al., 2023; Yang et al., 2024b) suggests transmitting image captions to reproduce training data at the server, while (Yang et al., 2024a) focuses on one-shot semi-supervised FL. However, these approaches are either inefficient or pose risks of client information leakage. In contrast, `FedBiP` functions as an OSFL algorithm, offering enhanced efficiency and robust privacy-preserving capabilities.

2.2 Diffusion Models For Image Synthesis

Diffusion models (Ho et al., 2020), especially Latent Diffusion Model (LDM) (Rombach et al., 2022), have attracted significant attention due to their capability to generate high-resolution natural images. They have demonstrated effectiveness in various applications, including image stylization (Guo et al., 2023; Meng et al., 2021; Kawar et al., 2023) and training data generation (Yuan et al., 2023; Sarıyıldız et al., 2023; Azizi et al., 2023). We refer readers to (Croitoru et al., 2023; Yang et al., 2023b) for a comprehensive overview of recent progress on diffusion models. Pretrained LDM has been adopted to address client data scarcity in OSFL (Zhang et al., 2023; Yang et al., 2024b). However, these methods often overlook the feature distribution shift between the LDM pretraining dataset and the clients' local data. This challenge is particularly pronounced in complex domains such as medical and satellite imaging. To address this issue, we propose `FedBiP`, which personalizes the pretrained LDM to synthesize data that is aligned with the clients' data distributions.

3 Preliminaries

3.1 Heterogeneous One-shot Federated Learning

In this section, we introduce our problem setting, i.e., heterogeneous One-Shot Federated Learning (OSFL). Following (Zhang et al., 2023), we focus on image classification tasks with the goal of optimizing a C-way classification model ϕ utilizing the client local data, where $C \in \mathbb{N}$ denotes the number of categories. We assume there are $K \in \mathbb{N}$ clients joining the collaborative training. Each client k owns its private dataset D^k containing $N^k \in \mathbb{N}$ (image, label) pairs: $\{x_i^k, y_i^k\}_{i=1}^{N^k}$. Only one-shot data upload from the clients to the central server is allowed.

As described in (Kairouz et al., 2021), OSFL with data heterogeneity is characterized by distribution shifts in local datasets: $P_{\mathcal{XY}}^{k_1} \neq P_{\mathcal{XY}}^{k_2}$ with $k_1 \neq k_2$, where $P_{\mathcal{XY}}^{k}$ defines the joint distribution of input space $\mathcal{X}$ and label space $\mathcal{Y}$ on D^k. Data heterogeneity can be decomposed into two types: (1) *label space* heterogeneity, where $P_{\mathcal{Y}}$ varies across clients, while $P_{\mathcal{X}|\mathcal{Y}}$ remains the same, and (2) *feature space* heterogeneity, where $P_{\mathcal{X}}$ or $P_{\mathcal{X}|\mathcal{Y}}$ varies across clients, while $P_{\mathcal{Y}|\mathcal{X}}$ or $P_{\mathcal{Y}}$ remains the same.

3.2 Latent Diffusion Model Pipeline

In this section, we introduce the training and inference pipelines for Latent Diffusion Model (LDM). We provide a schematic illustration in Figure 2. Given an image $x \in \mathbb{R}^{H \times W \times 3}$, the encoder $\mathcal{E}$ encodes x into a latent representation $z(0) = \mathcal{E}(x)$, where $z(0) \in \mathbb{R}^{h \times w \times c}$. Besides, the decoder $\mathcal{D}$ reconstructs the image from the latent, giving $\tilde{x} = \mathcal{D}(z(0)) = \mathcal{D}(\mathcal{E}(x))$. The forward diffusion and denoising processes occur in the latent representation space, as described below.

In the forward diffusion of LDM training, random noise $\epsilon \sim \mathcal{N}(0, I)$ is added to $z(0)$, producing

$$z(t) = \delta(t, z(0)) = \sqrt{\alpha_t} z(0) + \sqrt{1 - \alpha_t} \epsilon, \tag{1}$$

where $t \sim \text{Uniform}(\{1, ..., T\})$ is the timestep controlling the noise scheduler α_t. A larger t corresponds to greater noise intensity. In the denoising process, a UNet ϵ_θ is applied to denoise $z(t)$, yielding $\tilde{z}(0)$ for image reconstruction. To further condition LDM generation on textual inputs P, a feature extractor τ_θ is used to encode the prompts into intermediate representations for ϵ_θ. By sampling different values of ϵ and t, ϵ_θ can be optimized via the following loss function:

$$L_{LDM} = \mathbb{E}_{z(0),P,\epsilon,t}\left[||\epsilon - \epsilon_\theta(\delta(t, z(0)), t, \tau_\theta(P))||_2^2\right] \quad (2)$$

In the inference stage, latent representation $z(T)$ will be sampled directly from $\mathcal{N}(0, I)$, and multiple denoising steps are executed to obtain $\tilde{z}(0)$. The image is then decoded via $\tilde{x} = \mathcal{D}(\tilde{z}(0))$.

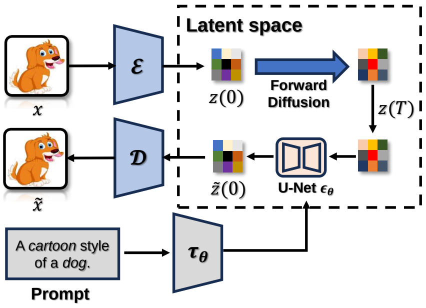


Figure 2: Schematic illustration of the Latent Diffusion Model pipeline with textual prompt conditioning.

4 Methodology

4.1 Motivational Case Study

To substantiate the necessity of the proposed method, we present an empirical analysis to address the following research question: *Can pretrained Latent Diffusion Model (LDM) generate images that are infrequently represented in the pretraining dataset using solely textual conditioning?* Specifically, we adopt two datasets, namely DomainNet (Peng et al., 2019) and DermaMNIST (Yang et al., 2023a), which contain images indicating different styles and images from challenging medical domains, respectively. We prompt LDM with *"A quickdraw style of an airplane."* to generate airplane images in quickdraw style for DomainNet dataset, and *"A dermatoscopic image of a dermatofibroma, a type of pigmented skin lesions."* for DermaMNIST. We synthesize 100 images for each setting and adopt a pretrained ResNet-18 (He et al., 2016) to acquire the feature embeddings of real and synthetic images. Finally, we visualize them using UMAP (McInnes et al., 2018).

As shown in Figure 1, we observe markedly different visual characteristics between synthetic and real images. Specifically, for DomainNet, there exist significant discrepancies between the "quickdraw" concept demonstration in the original dataset and the pretrained LDM. For DermaMNIST, the pretrained LDM is only able to perceive the general concepts of "dermatoscopic" and "skin lesion", failing to capture category-specific information. This further highlights the difficulties in reproducing medical domain data via LDM. Additionally, there is a substantial gap in the extracted feature embeddings between real and synthetic images. Most importantly, despite the high visual quality of the synthetic images, they may not contribute to the final performance of the classification model. As demonstrated by our experimental results (Table 5.3), directly applying such prompts to generate images for server-side training sometimes yields worse results than baseline methods. Therefore, it is essential to design a more sophisticated method to effectively personalize the pretrained LDM to the specific domains of client local datasets. These observations motivate our proposed method `FedBiP`, which mitigates the distribution shifts between pretrained LDM and the client local data. We introduce `FedBiP` in the following.

4.2 Proposed Method

A schematic overview of the proposed method is provided in Figure 3. Additionally, the pseudocode of the proposed method is presented in Algorithm 1. We begin by introducing the bi-level personalization in the local update of k^{th} client, omitting the subscript k for simplicity in the following description.

4.2.1 Instance-level Personalization

While the traditional Latent Diffusion Model (LDM) employs a Gaussian distribution to initialize the latent vector $z(T) \sim \mathcal{N}(0, I)$, we directly compute $z(T)$ from the local training set D^k of each client. Specifically, we leverage the VAE encoder $\mathcal{E}$ from pretrained LDM to obtain $z_i(T)$ for each specific real sample x_i. We first extract the low-dimensional latent representation by feeding the

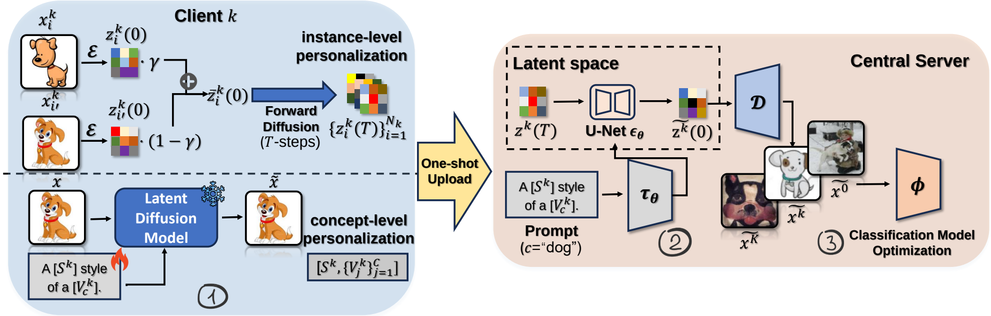


Figure 3: Schematic illustration of Federated Bi-Level Personalization (FedBiP). (①) Each client executes bi-level personalization and obtains latent vectors $z^k(T)$ and concept vectors S^k, V^k. (②) The central server integrates the vectors into the generation process of the pretrained Latent Diffusion Model θ. (③) The classification model ϕ is optimized using synthetic images.

training image into VAE encoder: $z_i(0) \leftarrow \mathcal{E}(x_i)$. We implement additional measures to enhance client privacy. First, we interpolate $z_i(0)$ with another latent representation, $z_{i'}(0)$, from the same class, thereby reducing the risk of exact sample reconstruction. Second, we add T-steps of random noise to obtain $z_i(T)$, which corresponds to the maximum noise intensity in LDM. A comprehensive privacy analysis is provided in Section 5.5 and 5.6. The overall process can be formalized as

$$z_i(T) \leftarrow \delta(T, \gamma z_i(0) + (1-\gamma) z_{i'}(0)), s.t., i \neq i', y_i = y_{i'}, \tag{3}$$

where $\gamma \sim \mathcal{N}(0.5, 0.1^2)$ and clipped to $[0, 1]$. After the computation, we store $z_i(T)$ and its corresponding ground truth label y_i for all training images in the k^{th} client as the instance-level personalization. We emphasize that this level of personalization does not require any additional optimization, making the process computationally efficient.

4.2.2 Concept-level Personalization

Solely applying instance-level personalization results in reduced diversity in image generation. To mitigate this limitation, we enhance personalization by incorporating domain and category concepts into the LDM generation process. Specifically, ”domain” denotes the feature distribution within a client’s local dataset, such as an image style in the DomainNet dataset. To avoid the costly finetuning of the LDM weights θ, we finetune only the textual guidance. Specifically, we randomly initialize the domain concept vector $S \in \mathbb{R}^{n_s \times d_w}$ and category concept vector $V \in \mathbb{R}^{C \times n_v \times d_w}$, where n_s and n_v are the number of tokens for domain concept and category concept, respectively, and d_w is the token embedding dimension of the textual conditioning model τ_θ. Subsequently, specific tokens in the textual template P are substituted with the concept vectors S and V_y corresponding to a specific category y. For instance, this could result in textual prompts like *”A* $[S]$ *style of a* $[V_y]$*”* for DomainNet dataset. Following this, τ_θ encodes these modified prompts, transforming the textual embeddings into intermediate representation for the denoising UNet ϵ_θ.

To jointly optimize both concept vectors S and V_y, we adopt the following objective function:

$$L_g = \mathbb{E}_{\mathcal{E}(x(0)), y, \epsilon \sim \mathcal{N}(0,1), t} \left[||\epsilon - \epsilon_\theta(z(t), t, \tau_\theta(S, V_y))||_2^2 \right], \tag{4}$$

where timestep t is sampled from Uniform$(\{1, ..., T\})$.

After the local optimization of each client, the latent vectors $\{z_i(T), y_i\}_{i=1}^{N^k}$, along with the optimized concept vectors S, V, are uploaded to the central server. To further increase the generation diversity, we introduce a small perturbation to the domain concept vector S. Specifically, we define $\hat{S} = S + \eta$ with $\eta \sim \mathcal{N}(0, \sigma_\eta)$, where σ_η controls the perturbation intensity. The central server then integrates these vectors into the same pretrained LDM and generates synthetic images with

$$\tilde{x}_i = \mathcal{D}(\epsilon_\theta(z_i(T), T, \tau_\theta(\hat{S}, V_{y_i}))). \tag{5}$$

The data sample $(\tilde{x}_i, y_i)$ is appended to the synthetic set D_{syn}. It is crucial to note that FedBiP performs image generation asynchronously, eliminating the need to wait for all clients to complete their local processes. Once the server receives the vectors uploaded from all clients and completes the image generation, we proceed to optimize the target classification model ϕ with the objective:

$$L_{cls} = L_{CE}(\phi(\tilde{x}), y). \tag{6}$$

Algorithm 1 Training process of `FedBiP`

```
ServerUpdate
Initialize Latent Diffusion Model with pretrained weights $\theta$, classification model $\phi$, synthetic dataset $D_{syn} \leftarrow \varnothing$
for client $k$ = 1 to $K$ do {in parallel}
    $k^{th}$ client execute $ClientUpdate(k)$ and upload $\{z_i^k(T), y_i^k\}_{i=1}^{N_k}, \{V_j^k\}_{j=1}^{C}, S^k$
    for $i$ = 1 to $N_k$ do
        $e \leftarrow \tau_\theta$(”A [$S^k$] style of a [$V_{y_i^k}^k$]”)
        $\tilde{z}(0) \leftarrow \epsilon_\theta(z_i^k(T), t, e)$, $\tilde{x} \leftarrow \mathcal{D}(\tilde{z}(0))$
        $D_{syn}.append([\tilde{x}, y_i^k])$
Optimize $\phi$ using $D_{syn}$ (Equation 6)

ClientUpdate($k$)
Initialize Latent Diffusion Model with pretrained weights $\theta$, randomly initialize $\{V_j^k\}_{j=1}^{C}, S^k$.
for $i$ = 1 to $N^k$ do
    Randomly sample an image $x_{i'}^k$ with $i \neq i', y_i = y_{i'}$
    $\bar{z}(0) \leftarrow \gamma\mathcal{E}(x_i^k) + (1-\gamma)\mathcal{E}(x_{i'}^k)$
    $z_i^k(T) \leftarrow \delta(T, \bar{z}(0))$
for local step $st$ = 1 to $N_{step}$ do
    Sample one mini-batch $\{x_b^k, y_b^k\}$ from $D^k$, timestep $t$
    $e \leftarrow \boldsymbol{\tau}$({”A [$S^k$] style of a [$V_{y_b^k}^k$]”})
    Optimize $S^k, \{V_j^k\}_{j=1}^{C}$ (Equation 4)
```

5 Experiments and Analyses

We conduct extensive empirical analyses to investigate the proposed method. Firstly, we compare `FedBiP` with other baseline methods on three One-Shot Federated Learning (OSFL) benchmarks with feature space heterogeneity. Next, we evaluate `FedBiP` using a medical dataset and a satellite image dataset adapted for OSFL setting with label space heterogeneity, illustrating its effectiveness under challenging real-world scenarios. Finally, we perform an ablation study on `FedBiP` and further analyze its promising privacy-preserving capability.

5.1 Benchmark Experiments

Datasets Description: We adapt three common image classification benchmarks with feature distribution shift for our OSFL setting: (1) *DomainNet* (Peng et al., 2019), which contains six domains: Clipart (C), Infograph (I), Painting (P), Quickdraw (Q), Real (R), and Sketch (S). We select 10 categories following (Zhang et al., 2023). (2) *PACS* (Li et al., 2017), which includes images that belong to 7 classes from four domains: Art (A), Cartoon (C), Photo (P), and Sketch (S). (3) *OfficeHome* (Venkateswara et al., 2017) comprises images of daily objects from four domains: Art (A), Clipart (C), Product (P), and Real (R). Each client is assigned a specific domain. To simulate local data scarcity described in previous sections, we adopt 16-shot per class (8-shot for OfficeHome) for each client, following previous works (Li et al., 2021; Chen et al., 2023).

Baseline Methods: We compare `FedBiP` with several baseline methods, including *FedAvg* and *Central*, i.e., aggregating the training data from all clients. We note that *Central* is an oracle method as it infringes on privacy requirements, while *FedAvg* requires multi-round communication and is not applicable to OSFL. Besides, we validate concurrent generation-based methods for OSFL: (1) *FedD3* (Song et al., 2023), where distilled instances from the clients are uploaded. (2) *DENSE* (Zhang et al., 2022), where client local models are uploaded and distilled into one model using synthetic images. (3) *FedDEO* (Yang et al., 2024b), where the optimized category descriptions are uploaded and guide pretrained diffusion models. (4) *FGL* (Zhang et al., 2023), where captions of client local images, extracted by BLIP-2 (Li et al., 2023), are uploaded and guide pretrained LDM.

Implementation Details: We adopt the HuggingFace open-sourced ”CompVis/stable-diffusion-v1-4” as the pretrained Latent Diffusion Model, and use ResNet-18 pretrained on ImageNet (Deng et al., 2009) as the initialization for the classification model. We investigate three variants of `FedBiP`,

Table 1: Evaluation results of different methods on three OSFL benchmarks with feature space heterogeneity. We report the mean±std classification accuracy from 3 runs with different seeds. The best and second-best results are marked with **bold** and underline, respectively.

Dataset		FedAvg	Central (*oracle*)	FedD3	DENSE	FedDEO	FGL	**FedBiP-S**	**FedBiP-M**	**FedBiP-L**
DomainNet	C	73.12 ±1.54	73.63 ±0.91	61.21 ±1.46	63.84 ±2.51	72.33 ±1.26	67.71 ±3.15	68.07 ±0.96	74.01 ±1.67	**77.52** ±0.67
	I	59.85 ±1.51	61.76 ±0.94	50.39 ±1.64	52.87 ±0.38	57.39 ±0.84	59.83 ±1.55	54.06 ±2.56	58.42 ±2.05	**60.94** ±2.08
	P	63.77 ±1.12	69.18 ±1.74	60.50 ±1.09	62.07 ±0.97	63.17 ±1.05	**68.56** ±2.51	58.24 ±0.22	63.01 ±2.25	65.20 ±0.78
	Q	16.26 ±2.60	72.83 ±0.82	28.25 ±3.11	29.92 ±1.62	37.86 ±2.47	19.83 ±2.99	51.09 ±2.05	49.64 ±5.05	**51.85** ±3.24
	R	87.90 ±0.09	87.86 ±0.24	79.15 ±1.44	81.69 ±1.14	81.51 ±1.03	**87.09** ±0.88	80.44 ±1.38	82.20 ±0.67	83.16 ±0.60
	S	68.07 ±4.67	75.28 ±0.96	58.07 ±1.35	59.20 ±2.12	62.86 ±1.61	67.15 ±3.97	57.17 ±1.59	61.92 ±1.35	**68.24** ±0.78
	Avg	61.49 ±0.58	73.42 ±0.53	56.26 ±0.74	58.26 ±1.33	62.52 ±1.56	61.69 ±1.56	61.51 ±0.62	64.86 ±0.49	**67.82** ±0.56
PACS	A	52.68 ±3.22	53.06 ±0.53	42.42 ±1.81	44.64 ±0.14	49.89 ±0.91	**55.04** ±1.79	43.01 ±1.80	50.15 ±1.86	53.26 ±2.54
	C	68.27 ±4.22	71.43 ±1.61	60.47 ±2.46	63.10 ±1.47	68.31 ±1.41	69.94 ±1.43	64.58 ±3.23	67.71 ±0.93	**70.90** ±2.97
	P	86.31 ±1.03	81.55 ±6.16	72.08 ±2.25	74.70 ±0.81	71.96 ±0.56	**76.47** ±0.68	70.24 ±2.73	73.07 ±1.80	74.85 ±1.36
	S	31.25 ±9.94	63.24 ±3.35	30.40 ±1.99	31.40 ±2.06	48.95 ±1.34	41.82 ±6.26	48.66 ±4.26	50.30 ±2.20	**51.70** ±1.69
	Avg	59.63 ±3.13	67.32 ±2.36	51.34 ±2.51	53.46 ±1.62	59.78 ±1.07	60.82 ±1.90	56.62 ±1.23	60.30 ±0.42	**62.67** ±0.45
Office Home	A	54.48 ±1.60	58.68 ±1.72	50.71 ±1.30	52.37 ±0.96	49.37 ±2.06	48.48 ±3.18	39.80 ±0.88	45.06 ±0.75	**55.41** ±0.55
	C	47.63 ±1.08	51.09 ±1.17	44.06 ±0.86	46.24 ±1.74	42.92 ±0.81	36.58 ±2.36	36.79 ±1.15	40.86 ±0.80	**48.62** ±0.42
	P	73.94 ±1.27	77.79 ±0.83	71.09 ±1.69	73.76 ±2.07	73.81 ±0.46	59.38 ±0.66	69.20 ±1.17	73.23 ±0.69	**76.63** ±0.20
	R	63.94 ±0.56	69.97 ±0.63	60.25 ±0.88	61.86 ±1.45	61.77 ±0.51	62.08 ±2.37	56.57 ±1.01	61.94 ±1.32	**65.43** ±0.96
	Avg	60.00 ±0.88	64.38 ±1.06	56.52 ±1.07	58.55 ±1.35	56.96 ±1.71	51.63 ±1.71	50.59 ±0.70	55.27 ±0.73	**61.52** ±0.39

namely ”S”, ”M”, and ”L”, which corresponds to generating 2×, 5×, 10× the number of images in the original client local dataset, respectively. Note that synthesizing more images does not affect the client's local optimization costs. We optimize the concept vectors for 50 epochs at each client. For *FGL*, 3500 samples per class per domain are generated. For *FedDEO*, the total number of synthetic images is identical to `FedBiP-L` for a fair comparison. Further details about training hyperparameters are provided in the Appendix.

Results and Analyses: We report the validation results in Table 1, where we observe `FedBiP-L` outperforms all baseline methods in average performance, indicating an average performance improvement of up to 5.96%. Notably, `FedBiP-S` achieves comparable performance to *FGL* by generating only 16 images for DomainNet per class and domain, while *FGL* requires 3500 images. This further highlights the efficiency of our proposed method. Additionally, `FedBiP` excels in challenging domains, such as Quickdraw (Q) of DomainNet and Sketch (S) of PACS, showcasing its effectiveness in generating images that are rare in the Latent Diffusion Model (LDM) pretraining dataset. However, `FedBiP` slightly underperforms in certain domains, e.g., Real (R) in DomainNet. We attribute this to the overlap between these domains and the LDM pretraining dataset, where adapting LDM with the client local datasets reduces its generation diversity. Nevertheless, `FedBiP` narrows the gap between the generation-based methods and oracle `Central` method.

5.2 Validation on Medical and Satellite Image Datasets

To illustrate the effectiveness of `FedBiP` on challenging real-world applications, we adopt a medical dataset, *DermaMNIST* (Yang et al., 2023a), comprising dermatoscopic images of 7 types of skin lesion, and a satellite image dataset, UC Merced Land Use Dataset (*UCM*) (Yang & Newsam, 2010), which includes satellite images representing 21 different land use categories. We assume there are 5 research institutions (clients) participating in the collaborative training. To construct local datasets for each client in OSFL, we employ the Dirichlet distribution Dir_β to model label space

Table 2: Evaluation results of different methods on real-world medical and satellite OSFL benchmarks with varying levels of label space heterogeneity. The best results are marked with **bold**.

Dataset	Split	FedAvg	Central (*oracle*)	FedD3	DENSE	FedDEO	FGL	**FedBiP-S**	**FedBiP-M**	**FedBiP-L**
UCM	IID	63.82 ±0.67	68.44 ±0.52	59.37 ±1.24	64.08 ±0.95	63.15 ±0.86	52.65 ±1.74	61.58 ±0.76	63.74 ±0.47	**65.59** ±1.01
	$Dir_{0.5}$	62.96 ±1.41	68.44 ±0.52	56.86 ±0.81	61.41 ±1.51	61.04 ±0.34	52.65 ±1.74	61.02 ±1.03	62.37 ±0.84	**64.41** ±0.88
	$Dir_{0.01}$	57.47 ±1.76	68.44 ±0.52	50.24 ±0.49	54.16 ±0.77	55.81 ±1.05	52.65 ±1.74	54.48 ±1.24	56.19 ±0.65	**59.84** ±0.47
Derma MNIST	IID	53.47 ±1.49	60.08 ±0.98	50.26 ±0.67	52.91 ±0.34	54.29 ±1.12	40.82 ±2.56	53.84 ±1.52	54.91 ±0.71	**56.10** ±1.34
	$Dir_{0.5}$	51.98 ±0.52	60.08 ±0.98	49.52 ±1.46	50.83 ±0.61	52.61 ±0.84	40.82 ±2.56	51.47 ±1.32	53.26 ±0.84	**55.03** ±1.02
	$Dir_{0.01}$	43.99 ±2.07	60.08 ±0.98	40.25 ±1.91	41.08 ±2.30	42.14 ±0.96	40.82 ±2.56	45.32 ±0.91	46.71 ±1.31	**48.15** ±1.67

heterogeneity, in which a smaller β indicates higher data heterogeneity. Following (Zhou et al., 2022), we use the textual template ”*A dermatoscopic image of a [CLS], a type of pigmented skin lesions.*” and ”*A centered satellite photo of [CLS].*” for DermaMNIST and UCM, respectively.

In Table 2, we report the validation results of different methods on real-world OSFL benchmarks with varying levels of label space heterogeneity. We observe that FedBiP-L consistently outperforms all baseline methods across all settings, with an average performance increase of up to 4.16% over *FedAvg*. Furthermore, we notice that the most lightweight version, FedBiP-S, surpasses the method with pretrained LDM, *FGL*, by a substantial margin. This demonstrates the importance of our LDM personalization schema, particularly in scenarios involving significant feature distribution shifts compared to the pretraining dataset of LDM.

5.3 Ablation Study

Table 3: Ablation study for different components of FedBiP on three benchmarks.

Instance	**Concept**		Domain Net	PACS	Office Home
$z(T)$	$\hat{S}$	V_c			
FedAvg (*multi-round*)			61.49	59.63	60.00
			60.22	58.90	53.23
		✓	61.71	59.15	55.81
	✓		63.96	60.08	56.32
✓			66.08	61.83	59.35
✓	✓ (no perturb.)	✓	67.09	62.78	60.84
✓	✓	✓	67.65	62.67	61.52

To illustrate the importance of different FedBiP components, we conduct an ablation study on three OSFL benchmark datasets. The results are shown in Table 5.3. First, we observe that simply prompting LDM with *”A [STY] style of a [CLS]”* and synthesizing images at central server is ineffective. Next, we notice that optimizing only the category concept vector V_c leads to only minimal performance improvements. We hypothesize that this is because the categories in these benchmarks are general objects, such as ”person” or ”clock”, which are already well-captured by LDM during pretraining. In contrast, optimizing the domain concept vector S produces visible performance gain. This can be attributed to the mismatch between the textual representation of domain concepts and LDM’s pretraining. For example, as described in Motivation section (Figure 1), ”Quickdraw” in DomainNet encompasses images characterized by very simple lines, while LDM tends to generate images with finer details. Furthermore, applying instance-level personalization with $z(T)$ yields a performance boost, highlighting the importance of fine-grained personalization in improving LDM. Finally, combining both levels of personalization further improves the results, which demonstrates their complementarity.

5.4 Scalability Analysis of FedBiP

To show the scalability of FedBiP under various application scenarios, we validate FedBiP in systems with varying client numbers and analyze the effects of synthetic image quantity.

Varying Number of Clients: We split each domain of the DomainNet dataset into 5 subsets, ensuring that each subset contains 16 samples per category to simulate the local data scarcity described in previous sections. Each subset is then assigned to a specific client. In our experiments, we select 1 to 5 clients from each domain, resulting in a total of 6 to 30 clients participating in federated learning.

The validation results are presented in Figure 4. We observe that the performance of the baseline method *FedAvg* remains unchanged with the addition of more clients to FL. In contrast, the validation performance of FedBiP consistently increases, narrowing the gap between distributed opti-

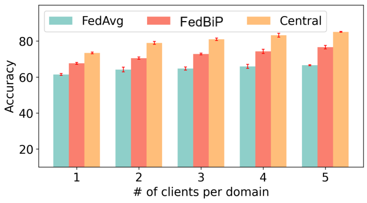
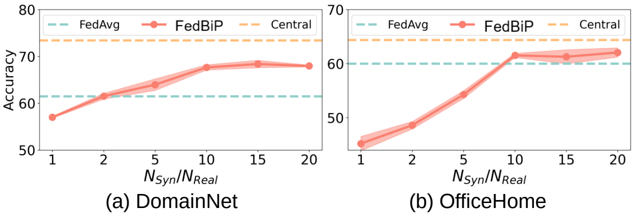


Figure 4: Validation results of FedBiP with varying number of clients on DomainNet.



(a) DomainNet

(b) OfficeHome

Figure 5: Validation results of FedBiP with synthesizing different numbers of images at central server.

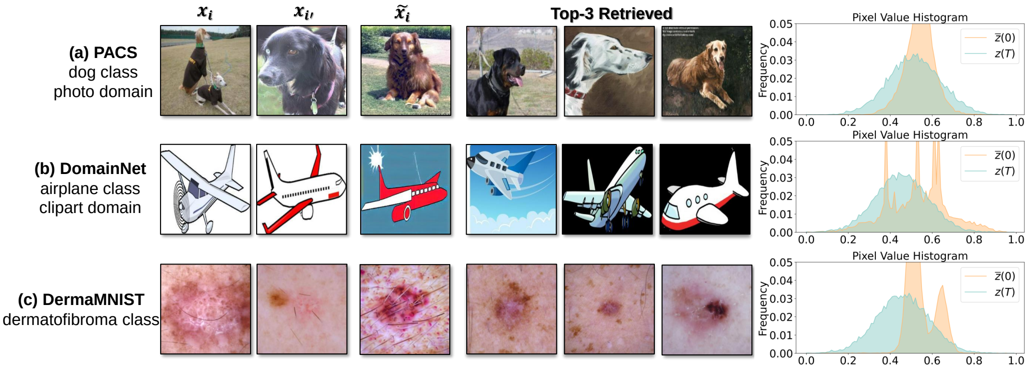


Figure 6: FedBiP privacy analysis: (1) **Visual**: The reproduced images are notably dissimilar to the original images x_i and $x_{i'}$. Besides, the retrieved images exhibit visual discrepancies compared to synthetic $\tilde{x}_i$. (2) **Statistical**: The pixel value histogram of $z(T)$ resembles a standard Gaussian distribution more closely compared to $\overline{z}(0)$, making it hard to extract private information from $z(T)$.

mization and *Central* optimization. Furthermore, FedBiP outperforms *FedAvg* by 9.51% when the largest number of clients join FL, further indicating its scalability for real-world complex federated systems with more clients.

Varying Number of Synthetic Images: We synthesize varying quantities of images for each category and domain, scaling from $1\times$ to $20\times$ the size of the original client local dataset. The results for the DomainNet and OfficeHome benchmarks are presented in Figure 5. Our analysis reveals that increasing the number of synthetic images enhances the performance of the target classification model, significantly outperforming the baseline method (*FedAvg*) by up to 6.47%. Furthermore, we observe that synthesizing images at $10\times$ the original dataset size emerges as the most effective approach, when considering the trade-off between generation time and final performance. This finding is consistent with the design principles of FedMLA-L.

5.5 Privacy Analysis

In this section, we present a comprehensive privacy analysis of FedBiP, encompassing both qualitative and quantitative evaluations, as illustrated in Figure 6.

Visual discrepancy between synthetic and real images: We visualize both synthetic image $\tilde{x}_i$, and its corresponding real images, i.e., x_i, $x_{i'}$. Besides, we use the pretrained ResNet-18 to extract the feature map of $\tilde{x}_i$ and retrieve the top-3 real images which indicate the largest cosine similarities in the feature space. We observe differences in both background (e.g., textual and color) and foreground (e.g., the exact object shape, position, and pose) between real and synthetic images. These visual discrepancies indicate that the synthetic images do not closely resemble any individual real images, thereby reducing the risk of revealing sensitive information about the original client data.

Pixel Value Histogram Analysis: To further analyze FedBiP from a statistical perspective, we provide histograms of both $\overline{z}(0)$ (the interpolated latent vectors of input images) and the corresponding $z(T)$ ($\overline{z}(0)$ with T-steps of random noise added). We observe that $z(T)$ closely resembles a standard Gaussian distribution, which contains less information about the original input images compared to $\overline{z}(0)$. This indicates that transmitting the noised $z(T)$ is more private than $\overline{z}(0)$, and would not significantly compromise privacy regulations. Additionally, we notice that $\overline{z}(0)$ could be further replaced with the average latent vectors of all samples from a specific class, i.e., categorical prototypes (Tan et al., 2022). This substitution might further protect client privacy and is appropriate for applications with stringent privacy requirements. We leave this for future work.

Membership Inference Attack (MIA) Analysis: Finally, we analyze the resilience of FedBiP against MIA. Following (Yeom et al., 2018; Salem et al., 2018), we compute the average loss and entropy of the final model on both training member and non-member data, and report the difference between the two averages. A smaller difference corresponds to better membership privacy preservation. From the MIA Analysis in Table 4, we can observe that FedBiP demonstrates superior resilience against MIA.

Table 4: Membership Inference Attack (MIA) analysis on different benchmarks. A lower metric corresponds to better MIA privacy.

Dataset	MIA Metric	FedAvg	**FedBiP**
DomainNet	Entropy ↓	0.1311	0.0186 ↓**85.8%**
	Loss ↓	0.5976	0.1611 ↓**73.0%**
DermaMNIST	Entropy ↓	0.0897	0.0551 ↓**38.6%**
	Loss ↓	0.5860	0.4127 ↓**29.6%**
PACS	Entropy ↓	0.1635	0.0338 ↓**79.3%**
	Loss ↓	0.4459	0.1244 ↓**72.1%**

5.6 Visualization with Varying γ

In this section, we visualize the synthetic image $\tilde{x}_i$ using different interpolation coefficients γ for DomainNet benchmark. Specifically, we compute the interpolated latent vector $\overline{z}_i(0)$ using $\gamma z_i(0) + (1-\gamma)z_{i'}(0)$. As shown in Figure 7, we observe that the synthetic images exhibit distinct visual characteristics compared to the real images, even when γ is set to 0.0 or 1.0, corresponding to the direct use of latent vectors from the original images. We attribute these differences to the sampling process involved in the denoising phase of Latent Diffusion Model. Additionally, applying γ values near 0.5 offers the most effective privacy protection. Most importantly, varying γ produces diverse images, which enhances generation diversity and is beneficial for training the classification model. Therefore, we use a Gaussian distribution $\mathcal{N}(0.5, 0.1^2)$ to sample γ in FedBiP.

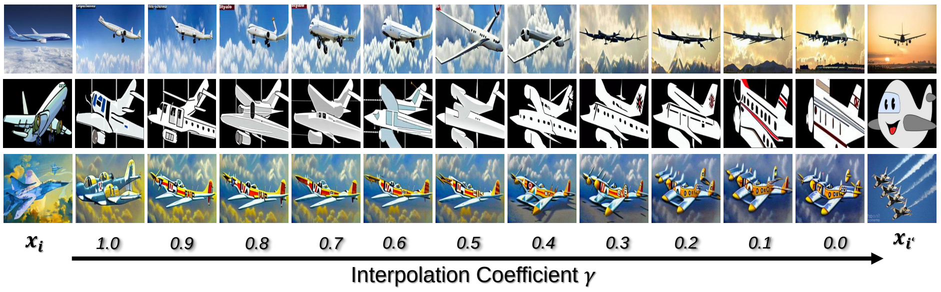
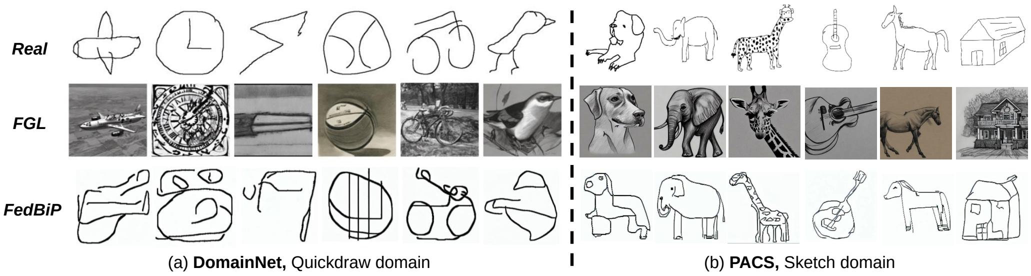


Figure 7: Synthetic images generated with varying γ for latent embedding interpolation.

5.7 Visualization for Challenging Domains

In this section, we present the synthetic images generated for the challenging domains, i.e., Quickdraw (DomainNet) and Sketch (PACS), as shown in Figure 8. Our observations indicate that FedBiP achieves superior generation quality by more accurately adhering to the original distribution of clients' local data compared to the diffusion-based method FGL (Zhang et al., 2023). This visualization further highlights the effectiveness of our bi-level personalization approach.



(a) **DomainNet,** Quickdraw domain

(b) **PACS,** Sketch domain

Figure 8: Comparison of synthetic images for challenging domains.

6 Conclusion

In this work, we propose the first framework to address feature space heterogeneity in One-Shot Federated Learning (OSFL) using generative foundation models, specifically Latent Diffusion Model (LDM). The proposed method, named `FedBiP`, personalizes the pretrained LDM at both instance-level and concept-level. This design enables LDM to synthesize images that adhere to the local data distribution of each client, exhibiting significant deviations compared to its pretraining dataset. The experimental results indicate its effectiveness under OSFL systems with both feature and label space heterogeneity, surpassing the baseline and multiple concurrent methods. Additional experiments with medical or satellite images demonstrate its maturity for challenging real-world applications. Moreover, additional analysis highlights its promising scalability and privacy-preserving capability.

References

Shekoofeh Azizi, Simon Kornblith, Chitwan Saharia, Mohammad Norouzi, and David J Fleet. Synthetic data from diffusion models improves imagenet classification. *arXiv preprint arXiv:2304.08466*, 2023.

Mahdi Beitollahi, Alex Bie, Sobhan Hemati, Leo Maxime Brunswic, Xu Li, Xi Chen, and Guojun Zhang. Parametric feature transfer: One-shot federated learning with foundation models. *arXiv preprint arXiv:2402.01862*, 2024.

Haokun Chen, Ahmed Frikha, Denis Krompass, Jindong Gu, and Volker Tresp. Fraug: Tackling federated learning with non-iid features via representation augmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4849–4859, 2023.

Hong-You Chen and Wei-Lun Chao. Fedbe: Making bayesian model ensemble applicable to federated learning. *arXiv preprint arXiv:2009.01974*, 2020.

Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. Diffusion models in vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(9): 10850–10869, 2023.

Rong Dai, Yonggang Zhang, Ang Li, Tongliang Liu, Xun Yang, and Bo Han. Enhancing one-shot federated learning through data and ensemble co-boosting. *arXiv preprint arXiv:2402.15070*, 2024.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Rui Gong, Martin Danelljan, Han Sun, Julio Delgado Mangas, and Luc Van Gool. Prompting diffusion representations for cross-domain semantic segmentation. *arXiv preprint arXiv:2307.02138*, 2023.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Neel Guha, Ameet Talwalkar, and Virginia Smith. One-shot federated learning. *arXiv preprint arXiv:1902.11175*, 2019.

Yuwei Guo, Ceyuan Yang, Anyi Rao, Yaohui Wang, Yu Qiao, Dahua Lin, and Bo Dai. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. *arXiv preprint arXiv:2307.04725*, 2023.

Mohsin Hasan, Guojun Zhang, Kaiyang Guo, Xi Chen, and Pascal Poupart. Calibrated one round federated learning with bayesian inference in the predictive space. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 12313–12321, 2024.

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

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Pierre Humbert, Batiste Le Bars, Aurélien Bellet, and Sylvain Arlot. One-shot federated conformal prediction. In *International Conference on Machine Learning*, pp. 14153–14177. PMLR, 2023.

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

Myeongkyun Kang, Philip Chikontwe, Soopil Kim, Kyong Hwan Jin, Ehsan Adeli, Kilian M Pohl, and Sang Hyun Park. One-shot federated learning on medical data using knowledge distillation with image synthesis and client model adaptation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 521–531. Springer, 2023.

Anirudh Kasturi and Chittaranjan Hota. Osgan: One-shot distributed learning using generative adversarial networks. *The Journal of Supercomputing*, 79(12):13620–13640, 2023.

Anirudh Kasturi, Anish Reddy Ellore, and Chittaranjan Hota. Fusion learning: A one shot federated learning. In *Computational Science–ICCS 2020: 20th International Conference, Amsterdam, The Netherlands, June 3–5, 2020, Proceedings, Part III 20*, pp. 424–436. Springer, 2020.

Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6007–6017, 2023.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pp. 5542–5550, 2017.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pp. 19730–19742. PMLR, 2023.

Qinbin Li, Bingsheng He, and Dawn Song. Practical one-shot federated learning for cross-silo setting. *arXiv preprint arXiv:2010.01017*, 2020.

Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization. *arXiv preprint arXiv:2102.07623*, 2021.

Zijian Li, Jiawei Shao, Yuyi Mao, Jessie Hui Wang, and Jun Zhang. Federated learning with gan-based data synthesis for non-iid clients. In *International Workshop on Trustworthy Federated Learning*, pp. 17–32. Springer, 2022.

Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. *arXiv preprint arXiv:2202.09778*, 2022.

Quande Liu, Cheng Chen, Jing Qin, Qi Dou, and Pheng-Ann Heng. Feddg: Federated domain generalization on medical image segmentation via episodic learning in continuous frequency space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1013–1023, 2021.

Lingjuan Lyu, Han Yu, and Qiang Yang. Threats to federated learning: A survey. *arXiv preprint arXiv:2003.02133*, 2020.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

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

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

Joshua Niemeijer, Manuel Schwonberg, Jan-Aike Termöhlen, Nico M Schmidt, and Tim Fingscheidt. Generalization by adaptation: Diffusion-based domain extension for domain-generalized semantic segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2830–2840, 2024.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1406–1415, 2019.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *arXiv preprint arXiv:1806.01246*, 2018.

Mert Bülent Sarıyıldız, Karteek Alahari, Diane Larlus, and Yannis Kalantidis. Fake it till you make it: Learning transferable representations from synthetic imagenet clones. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8011–8021, 2023.

MyungJae Shin, Chihoon Hwang, Joongheon Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Xor mixup: Privacy-preserving data augmentation for one-shot federated learning. *arXiv preprint arXiv:2006.05148*, 2020.

Jinhyun So, Kevin Hsieh, Behnaz Arzani, Shadi Noghabi, Salman Avestimehr, and Ranveer Chandra. Fedspace: An efficient federated learning framework at satellites and ground stations. *arXiv preprint arXiv:2202.01267*, 2022.

Rui Song, Dai Liu, Dave Zhenyu Chen, Andreas Festag, Carsten Trinitis, Martin Schulz, and Alois Knoll. Federated learning via decentralized dataset distillation in resource-constrained edge environments. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–10. IEEE, 2023.

Shangchao Su, Bin Li, and Xiangyang Xue. One-shot federated learning without server-side training. *Neural Networks*, 164:203–215, 2023.

Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. Fedproto: Federated prototype learning across heterogeneous clients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8432–8440, 2022.

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5018–5027, 2017.

Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. Medmnist v2-a large-scale lightweight benchmark for 2d and 3d biomedical image classification. *Scientific Data*, 10(1):41, 2023a.

Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 56(4):1–39, 2023b.

Mingzhao Yang, Shangchao Su, Bin Li, and Xiangyang Xue. Exploring one-shot semi-supervised federated learning with pre-trained diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 16325–16333, 2024a.

Mingzhao Yang, Shangchao Su, Bin Li, and Xiangyang Xue. Feddeo: Description-enhanced one-shot federated learning with diffusion models. In *ACM Multimedia*, 2024b.

Yi Yang and Shawn Newsam. Bag-of-visual-words and spatial extensions for land-use classification. In *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*, pp. 270–279, 2010.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*, pp. 268–282. IEEE, 2018.

Jianhao Yuan, Jie Zhang, Shuyang Sun, Philip Torr, and Bo Zhao. Real-fake: Effective training data synthesis through distribution matching. *arXiv preprint arXiv:2310.10402*, 2023.

Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. In *International conference on machine learning*, pp. 7252–7261. PMLR, 2019.

Jie Zhang, Chen Chen, Bo Li, Lingjuan Lyu, Shuang Wu, Shouhong Ding, Chunhua Shen, and Chao Wu. Dense: Data-free one-shot federated learning. *Advances in Neural Information Processing Systems*, 35:21414–21428, 2022.

Jie Zhang, Xiaohua Qi, and Bo Zhao. Federated generative learning with foundation models. *arXiv preprint arXiv:2306.16064*, 2023.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022.

Yanlin Zhou, George Pu, Xiyao Ma, Xiaolin Li, and Dapeng Wu. Distilled one-shot federated learning. *arXiv preprint arXiv:2009.07999*, 2020.

Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. In *International conference on machine learning*, pp. 12878–12889. PMLR, 2021.

Table 5: Detailed hyperparameters for each dataset. The highlighted words ([STY]) in the textual prompt will be replaced by the domain concept vectors. The [CLS] will be replaced by the class concept vectors.

Dataset	prompt	n_s	n_c	C	Class Names
Derma MNIST	A dermatoscopic image of a [CLS], a type of pigmented skin lesions.	2	4	10	intraepithelial carcinoma, basal cell carcinoma, benign keratosis, dermatofibroma, melanoma, melanocytic nevi, vascular skin
UCM	A centered satellite photo of [CLS].	3	3	21	agricultural, dense residential, medium residential, sparse residential, parking lot, buildings, harbor, mobile homepark, storage tanks, freeway, intersection, overpass, golf course, baseball diamond, runway, tenniscourt, beach, forest, river, chaparral, airplane
Domain Net	A [STY] of [CLS].	1	1	10	airplane, clock, axe, basketball, bicycle, bird, strawberry, flower, pizza, bracelet
Office Home	A [STY] of [CLS].	1	1	20	Marker, Spoon, Pencil, Speaker, Toys, Fan, Hammer, Notebook, Telephone, Sink, Chair, Fork, Kettle, Bucket, Knives, Monitor, Mop, Oven, Pen, Couch
PACS	A [STY] of [CLS].	1	1	7	dog, elephant, giraffe, guitar, horse, house, person

A Experimental Details

We use 1 NVIDIA RTX A5000 with 24GB RAM to run the experiments. We use PyTorch (Paszke et al., 2019) to implement our algorithm. For the baseline FedAvg, the total communication round is set to 50. For FGL (Zhang et al., 2023), we generate 3500 images per class per domain. For the optimization of the classification model, we use SGD with momentum as the optimizer, where the learning rate is set to 0.01 and the momentum is 0.9. The optimization epoch is set to 50. The training image resolution is set to 512 $\times$ 512 for all datasets.

For FedD3 (Song et al., 2023), we adopt Kernel Inducing Points (KIP) to distill the original dataset into 1 image per class per domain and transmit them to the central server. For DENSE (Zhang et al., 2022), we first finetune the pretrained ResNet-18 (He et al., 2016) at each client and then optimize a Generator to conduct model distillation at central server. The hyperparameters used in these methods are following their original papers. For `FedMLA`, we use Adam optimizer to optimize the concept vectors. The learning rate is set to 0.1 and beta is set to (0.9, 0.999). The total training epochs is set to 30. We adopt the Pseudo Numerical Diffusion Model (PNDM) (Liu et al., 2022) in the Latent Diffusion Model. The perturbation intensity for domain concept vector σ_μ is set to 0.1 for all dataset. More dataset specific hyperparameters are provided in Table 5.

B Synthetic Image Visualization

We provide synthetic images for all benchmarks in the following figures, where we observe that the synthetic images generally follow the distribution and characteristics of the original training datasets at each client. Besides, the visual quality of the generated images, e.g., the detailed features of the objects, is also promising.

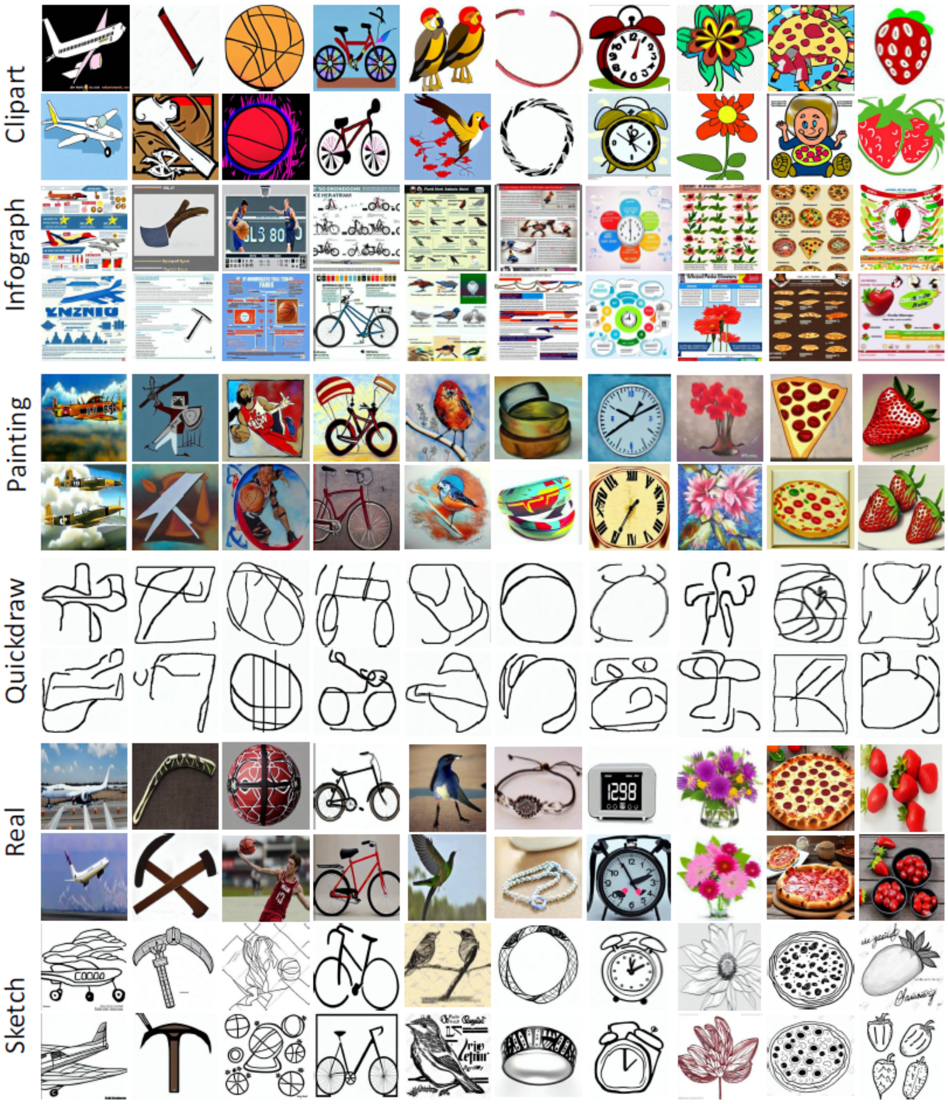


Figure 9: Synthetic Images for DomainNet benchmark.

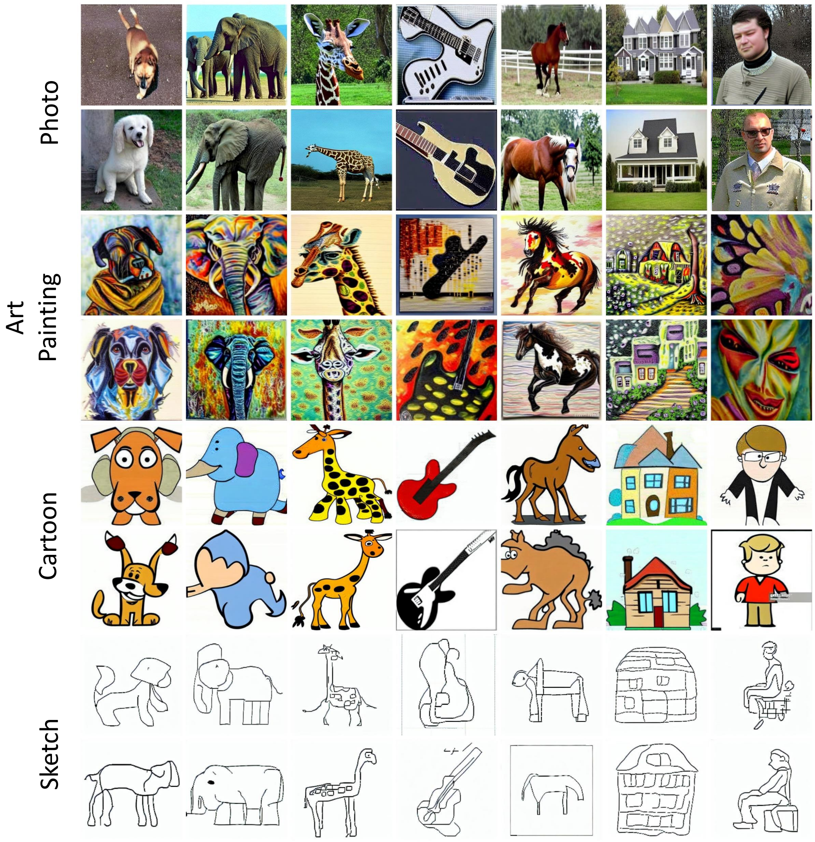


Figure 10: Synthetic Images for PACS benchmark.

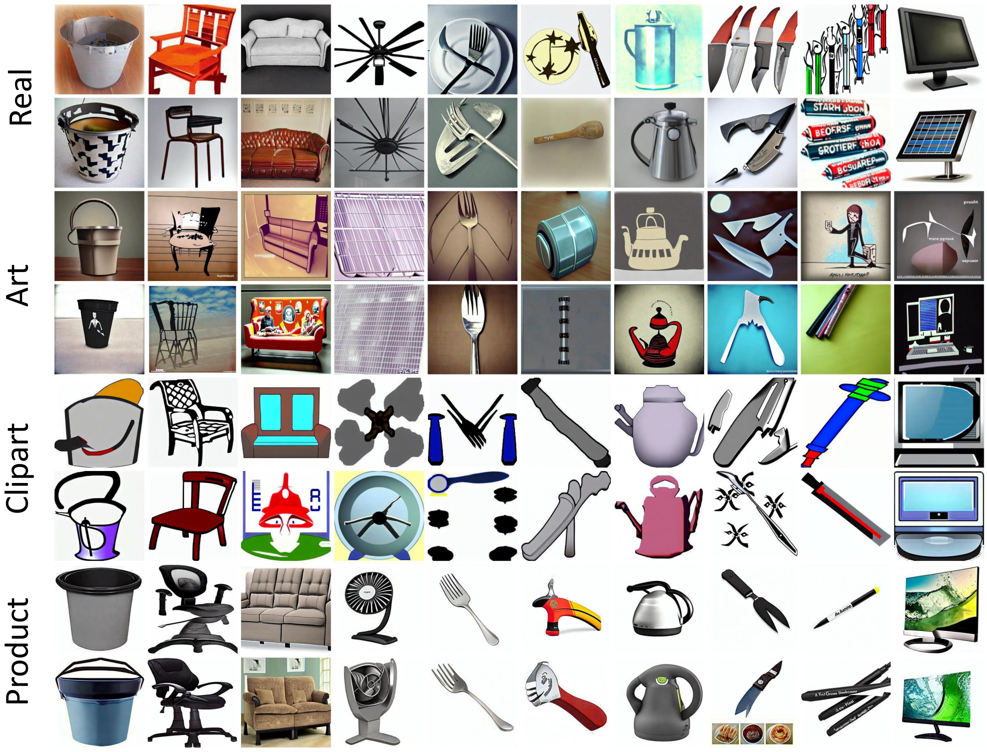


Figure 11: Synthetic Images for OfficeHome benchmark.

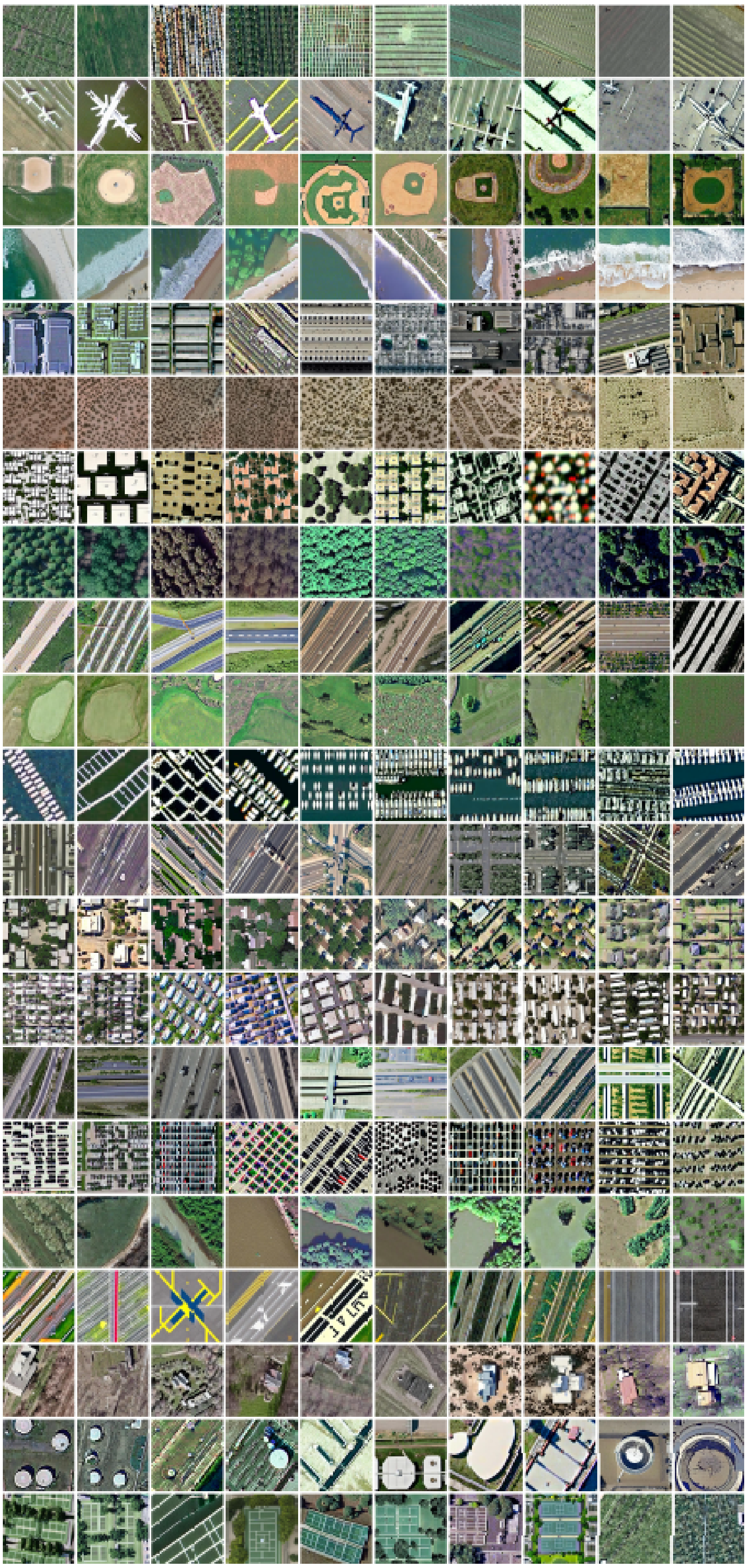


Figure 12: Synthetic Images for UCM benchmark.

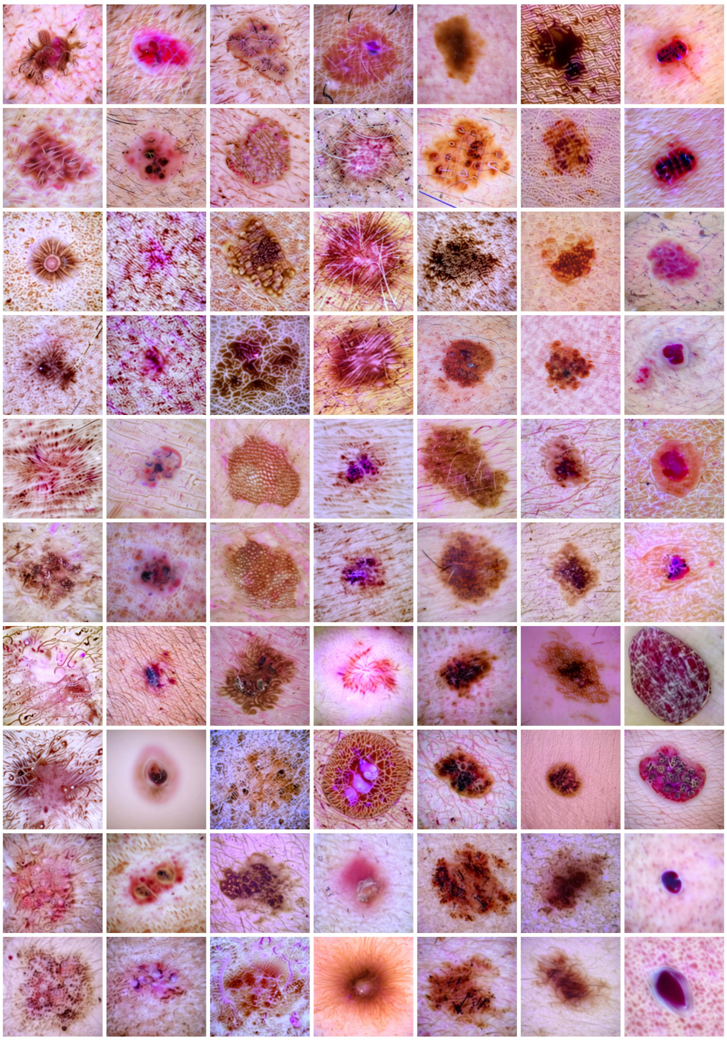


Figure 13: Synthetic Images for DermaMNIST benchmark.

Chapter 6

Conclusion

This thesis investigates the interplay between federated learning (FL) and core challenges arising from heterogeneous, distributed, and resource-constrained data environments. As FL gains traction in privacy-preserving and decentralized machine learning, practical deployment remains limited by issues such as non-identically distributed data across clients, limited communication budgets, and the need for personalized and adaptive modeling. To address these challenges, we develop methods that enhance FL systems across multiple fronts, including feature space alignment, multi-modal model adaptation, hyperparameter optimization, and efficient training under communication constraints.

The first part of this thesis explores the underexamined intersection of FL and feature space heterogeneity problems. In this learning scenario, clients possess data with differing feature distributions, which is a practical challenge arising from variations in data collection environments across clients. To address this, we propose a data augmentation method to augment the feature space of different clients. This is achieved by optimizing multiple generative models tailored to this scenario. We further provide both theoretical and empirical analyses of the proposed method. Experiments conducted on three public benchmarks and a real-world medical dataset demonstrate its effectiveness.

The second part focuses on the intersection of FL and fine-tuning vision-language foundation models. The objective is to fine-tune a vision-language model using distributed datasets while accounting for distribution shifts across modalities (vision and language) among clients. To address this, we adapt the conventional parameter-efficient fine-tuning (PEFT) approach, specifically using adapters, to FL systems. Meanwhile, we design a specialized architecture that effectively retains both client-specific and client-agnostic knowledge within the system. Empirical evaluations demonstrate that our method not only surpasses existing PEFT approaches but also achieves promising communication and computation efficiency.

In the third part, we explore the intersection of FL and Hyperparameter Tuning (HPT). Similar to conventional machine learning algorithms, FL is highly sensitive to hyperparameter selection. However, existing HPT methods designed for centralized ML are inadequate for FL due to the limited computational resources available at clients and the presence of distribution shifts across clients. To address these challenges, we propose a novel HPT method based on population-based evolutionary algorithms to optimize hyperparameters for both client-side and server-side learning processes. Our approach employs an online "tuning-while-training" framework, which enhances computational efficiency and broadens the hyperparameter search space. Through empirical evaluations on standard FL benchmarks and real-world datasets, including the full-scale Non-IID ImageNet1K, we demonstrate that our method significantly surpasses state-of-the-art HPT techniques in FL scenarios.

In the fourth part, we address the problem of One-Shot Federated Learning (OSFL), a paradigm designed to minimize communication costs and privacy risks by requiring only a single round of client data or model upload. Despite its advantages, OSFL is hindered by client data heterogeneity and limited data availability. Latent Diffusion Models (LDMs) provide a promising solution by generating high-quality synthetic images. However, directly applying pretrained LDMs in heterogeneous OSFL scenarios leads to distribution shifts and performance degradation, particularly in specialized domains such as medical imaging. To address these issues, we propose a novel method to personalize pretrained LDMs, enabling the synthesis of images that align with each client's local data distribution while preserving privacy. This represents the first approach to address feature space heterogeneity and data scarcity in OSFL. Extensive experiments on OSFL benchmarks and demanding datasets, including those in medical and satellite imaging, validate the effectiveness of our method, FedBiP, which surpasses existing OSFL techniques.

Bibliography

[1] Sarita Agrawal, Manik Lal Das, and Javier Lopez. Detection of node capture attack in wireless sensor networks. *IEEE Systems Journal*, 13(1):238–247, 2018.

[2] Alham Fikri Aji and Kenneth Heafield. Sparse communication for distributed gradient descent. *arXiv preprint arXiv:1704.05021*, 2017.

[3] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International conference on artificial intelligence and statistics*, pages 2938–2948. PMLR, 2020.

[4] Gilad Baruch, Moran Baruch, and Yoav Goldberg. A little is enough: Circumventing defenses for distributed learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[5] Dan Bogdanov, Sven Laur, and Jan Willemson. Sharemind: A framework for fast privacy-preserving computations. In *Computer Security-ESORICS 2008: 13th European Symposium on Research in Computer Security, Málaga, Spain, October 6-8, 2008. Proceedings 13*, pages 192–206. Springer, 2008.

[6] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017.

[7] Parimala Boobalan, Swarna Priya Ramu, Quoc-Viet Pham, Kapal Dev, Sharnil Pandya, Praveen Kumar Reddy Maddikunta, Thippa Reddy Gadekallu, and Thien Huynh-The. Fusion of federated learning and industrial internet of things: A survey. *Computer Networks*, 212:109048, 2022.

[8] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

[9] Fernando E Casado, Dylan Lema, Marcos F Criado, Roberto Iglesias, Carlos V Regueiro, and Senén Barro. Concept drift detection and adaptation for federated and continual learning. *Multimedia Tools and Applications*, pages 1–23, 2022.

[10] Cheng Chen, Bhavya Kailkhura, Ryan Goldhahn, and Yi Zhou. Certifiably-robust federated adversarial learning via randomized smoothing. In *2021 IEEE 18th international conference on mobile ad hoc and smart systems (MASS)*, pages 173–179. IEEE, 2021.

[11] Mingzhe Chen, Nir Shlezinger, H Vincent Poor, Yonina C Eldar, and Shuguang Cui. Communication-efficient federated learning. *Proceedings of the National Academy of Sciences*, 118(17):e2024789118, 2021.

[12] Mingzhe Chen, Zhaohui Yang, Walid Saad, Changchuan Yin, H Vincent Poor, and Shuguang Cui. A joint learning and communications framework for federated learning over wireless networks. *IEEE transactions on wireless communications*, 20(1):269–283, 2020.

[13] Wenxin Chen, Jinrui Zhang, and Deyu Zhang. Fl-joint: joint aligning features and labels in federated learning for data heterogeneity. *Complex & Intelligent Systems*, 11(1):1–14, 2025.

[14] Yujing Chen, Zheng Chai, Yue Cheng, and Huzefa Rangwala. Asynchronous federated learning for sensor data with concept drift. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 4822–4831. IEEE, 2021.

[15] Wu-Chun Chung, Yan-Hui Lin, and Sih-Han Fang. Fedism: Enhancing data imbalance via shared model in federated learning. *Mathematics*, 11(10):2385, 2023.

[16] Lei Cui, Youyang Qu, Gang Xie, Deze Zeng, Ruidong Li, Shigen Shen, and Shui Yu. Security and privacy-enhanced federated learning for anomaly detection in iot infrastructures. *IEEE Transactions on Industrial Informatics*, 18(5):3492–3500, 2021.

[17] Georgios Damaskinos, El-Mahdi El-Mhamdi, Rachid Guerraoui, Arsany Guirguis, and Sébastien Rouault. Aggregathor: Byzantine machine learning via robust gradient aggregation. *Proceedings of Machine Learning and Systems*, 1:81–106, 2019.

[18] Cynthia Dwork. Differential privacy. In *International colloquium on automata, languages, and programming*, pages 1–12. Springer, 2006.

[19] El-Mahdi El-Mhamdi, Rachid Guerraoui, Arsany Guirguis, Lê Nguyên Hoang, and Sébastien Rouault. Genuinely distributed byzantine machine learning. In *Proceedings of the 39th Symposium on Principles of Distributed Computing*, pages 355–364, 2020.

[20] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local model poisoning attacks to {Byzantine-Robust} federated learning. In *29th USENIX security symposium (USENIX Security 20)*, pages 1605–1622, 2020.

[21] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

[22] Xueluan Gong, Yanjiao Chen, Qian Wang, and Weihan Kong. Backdoor attacks and defenses in federated learning: State-of-the-art, taxonomy, and future directions. *IEEE Wireless Communications*, 30(2):114–121, 2022.

[23] Neel Guha, Ameet Talwalkar, and Virginia Smith. One-shot federated learning. *arXiv preprint arXiv:1902.11175*, 2019.

[24] Yongxin Guo, Xiaoying Tang, and Tao Lin. Fedrc: Tackling diverse distribution shifts challenge in federated learning by robust clustering. *arXiv preprint arXiv:2301.12379*, 2023.

[25] Prajjwal Gupta, Krishna Yadav, Brij B Gupta, Mamoun Alazab, and Thippa Reddy Gadekallu. A novel data poisoning attack in federated learning based on inverted loss function. *Computers & Security*, 130:103270, 2023.

[26] Ali Hatamizadeh, Hongxu Yin, Holger R Roth, Wenqi Li, Jan Kautz, Daguang Xu, and Pavlo Molchanov. Gradvit: Gradient inversion of vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10021–10030, 2022.

[27] Junyuan Hong, Haotao Wang, Zhangyang Wang, and Jiayu Zhou. Federated robustness propagation: sharing adversarial robustness in heterogeneous federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 7893–7901, 2023.

[28] Xixi Huang, Ye Ding, Zoe L Jiang, Shuhan Qi, Xuan Wang, and Qing Liao. Dp-fl: a novel differentially private federated learning framework for the unbalanced data. *World Wide Web*, 23:2529–2545, 2020.

[29] Yangsibo Huang, Samyak Gupta, Zhao Song, Kai Li, and Sanjeev Arora. Evaluating gradient inversion attacks and defenses in federated learning. *Advances in neural information processing systems*, 34:7232–7241, 2021.

[30] Jinwoo Jeon, Kangwook Lee, Sewoong Oh, Jungseul Ok, et al. Gradient inversion with generative image prior. *Advances in neural information processing systems*, 34:29898–29908, 2021.

[31] Ellango Jothimurugesan, Kevin Hsieh, Jianyu Wang, Gauri Joshi, and Phillip B Gibbons. Federated learning under distributed concept drift. In *International Conference on Artificial Intelligence and Statistics*, pages 5834–5853. PMLR, 2023.

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

[33] Georgios Kaissis, Alexander Ziller, Jonathan Passerat-Palmbach, Théo Ryffel, Dmitrii Usynin, Andrew Trask, Ionésio Lima Jr, Jason Mancuso, Friederike Jungmann, Marc-Matthias Steinborn, et al. End-to-end privacy preserving deep learning on multi-institutional medical imaging. *Nature Machine Intelligence*, 3(6):473–484, 2021.

[34] Hyunjun Kim, Yungi Cho, Younghan Lee, Ho Bae, and Yunheung Paek. Exploring clustered federated learning's vulnerability against property inference attack. In *Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses*, pages 236–249, 2023.

[35] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[36] Cody Lewis, Vijay Varadharajan, and Nasimul Noman. Attacks against federated learning defense systems and their mitigation. *Journal of Machine Learning Research*, 24(30):1–50, 2023.

[37] Jichang Li, Guanbin Li, Hui Cheng, Zicheng Liao, and Yizhou Yu. Feddiv: Collaborative noise filtering for federated learning with noisy labels. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 3118–3126, 2024.

[38] Jie Li, Tianqing Zhu, Wei Ren, and Kim-Kwang Raymond. Improve individual fairness in federated learning via adversarial training. *Computers & Security*, 132:103336, 2023.

[39] Qi Li, Zhuotao Liu, Qi Li, and Ke Xu. martfl: Enabling utility-driven data marketplace with a robust and verifiable federated learning architecture. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 1496–1510, 2023.

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

[41] Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization. *arXiv preprint arXiv:2102.07623*, 2021.

[42] Xin-Chun Li and De-Chuan Zhan. Fedrs: Federated learning with restricted softmax for label distribution non-iid data. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 995–1005, 2021.

[43] Xuan Li, Naiyu Wang, Shuai Yuan, and Zhitao Guan. Fedimp: Parameter importance-based model poisoning attack against federated learning system. *Computers & Security*, 144:103936, 2024.

[44] Yanli Li, Zhongliang Guo, Nan Yang, Huaming Chen, Dong Yuan, and Weiping Ding. Threats and defenses in the federated learning life cycle: A comprehensive survey and challenges. *IEEE Transactions on Neural Networks and Learning Systems*, 2025.

[45] Yanli Li, Dong Yuan, Abubakar Sadiq Sani, and Wei Bao. Enhancing federated learning robustness in adversarial environment through clustering non-iid features. *Computers & Security*, 132:103319, 2023.

[46] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. *arXiv preprint arXiv:1712.01887*, 2017.

[47] Or Litany, Haggai Maron, David Acuna, Jan Kautz, Gal Chechik, and Sanja Fidler. Federated learning with heterogeneous architectures using graph hypernetworks. *arXiv preprint arXiv:2201.08459*, 2022.

[48] Jiewen Liu, Bing Chen, Baolu Xue, Mengya Guo, and Yuntao Xu. Piafgnn: Property inference attacks against federated graph neural networks. *Computers, Materials & Continua*, 82(2), 2025.

[49] Ming Liu, Stella Ho, Mengqi Wang, Longxiang Gao, Yuan Jin, and He Zhang. Federated learning meets natural language processing: A survey. *arXiv preprint arXiv:2107.12603*, 2021.

[50] Guodong Long, Yue Tan, Jing Jiang, and Chengqi Zhang. Federated learning for open banking. In *Federated learning: privacy and incentive*, pages 240–254. Springer, 2020.

[51] Raúl López-Blanco, Ricardo S Alonso, Sara Rodríguez-González, Javier Prieto, and Juan M Corchado. Trustworthy artificial intelligence-based federated architecture for symptomatic disease detection. *Neurocomputing*, 579:127415, 2024.

[52] Bing Luo, Xiang Li, Shiqiang Wang, Jianwei Huang, and Leandros Tassiulas. Cost-effective federated learning design. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2021.

[53] Andreas Lutz, Gabriele Steidl, Karsten Müller, and Wojciech Samek. Boosting federated learning with fedentopt: Mitigating label skew by entropy-based client selection. *arXiv preprint arXiv:2411.01240*, 2024.

[54] Jing Ma, Si-Ahmed Naas, Stephan Sigg, and Xixiang Lyu. Privacy-preserving federated learning based on multi-key homomorphic encryption. *International Journal of Intelligent Systems*, 37(9):5880–5901, 2022.

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

[56] Linhao Meng, Yating Wei, Rusheng Pan, Shuyue Zhou, Jianwei Zhang, and Wei Chen. Vadaf: visualization for abnormal client detection and analysis in federated learning. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 11(3-4):1–23, 2021.

[57] Viraaji Mothukuri, Prachi Khare, Reza M Parizi, Seyedamin Pouriyeh, Ali Dehghantanha, and Gautam Srivastava. Federated-learning-based anomaly detection for iot security attacks. *IEEE Internet of Things Journal*, 9(4):2545–2554, 2021.

[58] Yongli Mou, Jiahui Geng, Feng Zhou, Oya Beyan, Chunming Rong, and Stefan Decker. pfedv: Mitigating feature distribution skewness via personalized federated learning with variational distribution constraints. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 283–294. Springer, 2023.

[59] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE symposium on security and privacy (SP)*, pages 739–753. IEEE, 2019.

[60] Dinh C Nguyen, Ming Ding, Pubudu N Pathirana, Aruna Seneviratne, Jun Li, and H Vincent Poor. Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 23(3):1622–1658, 2021.

[61] Thien Duc Nguyen, Phillip Rieger, Roberta De Viti, Huili Chen, Björn B Brandenburg, Hossein Yalame, Helen Möllering, Hossein Fereidooni, Samuel Marchal, Markus Miettinen, et al. {FLAME}: Taming backdoors in federated learning. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 1415–1432, 2022.

[62] Thien Duc Nguyen, Phillip Rieger, Markus Miettinen, and Ahmad-Reza Sadeghi. Poisoning attacks on federated learning-based iot intrusion detection system. In *Proc. Workshop Decentralized IoT Syst. Secur.(DISS)*, volume 79, 2020.

[63] Thuy Dung Nguyen, Tuan A Nguyen, Anh Tran, Khoa D Doan, and Kok-Seng Wong. Iba: Towards irreversible backdoor attacks in federated learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[64] Florian Nuding and Rudolf Mayer. Data poisoning in sequential and parallel federated learning. In *Proceedings of the 2022 ACM on international workshop on security and privacy analytics*, pages 24–34, 2022.

[65] Jaehyoung Park and Hyuk Lim. Privacy-preserving federated learning using homomorphic encryption. *Applied Sciences*, 12(2):734, 2022.

[66] Jiaming Pei, Wenxuan Liu, Jinhai Li, Lukun Wang, and Chao Liu. A review of federated learning methods in heterogeneous scenarios. *IEEE Transactions on Consumer Electronics*, 2024.

[67] Krishna Pillutla, Kshitiz Malik, Abdel-Rahman Mohamed, Mike Rabbat, Maziar Sanjabi, and Lin Xiao. Federated learning with partial model personalization. In *International Conference on Machine Learning*, pages 17716–17758. PMLR, 2022.

[68] Tao Qi, Fangzhao Wu, Lingjuan Lyu, Yongfeng Huang, and Xing Xie. Fedsampling: A better sampling strategy for federated learning. *arXiv preprint arXiv:2306.14245*, 2023.

[69] Yuanhang Qi, M Shamim Hossain, Jiangtian Nie, and Xuandi Li. Privacy-preserving blockchain-based federated learning for traffic flow prediction. *Future Generation Computer Systems*, 117:328–337, 2021.

[70] Han Qin, Guimin Chen, Yuanhe Tian, and Yan Song. Improving federated learning for aspect-based sentiment analysis via topic memories. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pages 3942–3954, 2021.

[71] Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

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

[73] Tao Sheng, Chengchao Shen, Yuan Liu, Yeyu Ou, Zhe Qu, Yixiong Liang, and Jianxin Wang. Modeling global distribution for federated learning with label distribution skew. *Pattern Recognition*, 143:109724, 2023.

[74] Benyuan Sun, Hongxing Huo, Yi Yang, and Bo Bai. Partialfed: Cross-domain personalized federated learning via partial initialization. *Advances in Neural Information Processing Systems*, 34:23309–23320, 2021.

[75] Gan Sun, Yang Cong, Jiahua Dong, Qiang Wang, Lingjuan Lyu, and Ji Liu. Data poisoning attacks on federated machine learning. *IEEE Internet of Things Journal*, 9(13):11365–11375, 2021.

[76] Youbang Sun, Zitao Li, Yaliang Li, and Bolin Ding. Improving lora in privacy-preserving federated learning. *arXiv preprint arXiv:2403.12313*, 2024.

[77] Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*, 2019.

[78] Anshuman Suri, Pallika Kanani, Virendra J Marathe, and Daniel W Peterson. Subject membership inference attacks in federated learning. *arXiv preprint arXiv:2206.03317*, 2022.

[79] Zhen Ling Teo, Liyuan Jin, Nan Liu, Siqi Li, Di Miao, Xiaoman Zhang, Wei Yan Ng, Ting Fang Tan, Deborah Meixuan Lee, Kai Jie Chua, et al. Federated machine learning in healthcare: A systematic review on clinical applications and technical architecture. *Cell Reports Medicine*, 5(2), 2024.

[80] Yuchen Tian, Weizhe Zhang, Andrew Simpson, Yang Liu, and Zoe Lin Jiang. Defending against data poisoning attacks: from distributed learning to federated learning. *The Computer Journal*, 66(3):711–726, 2021.

[81] Vale Tolpegin, Stacey Truex, Mehmet Emre Gursoy, and Ling Liu. Data poisoning attacks against federated learning systems. In *Computer security–ESORICs 2020: 25th European symposium on research in computer security, ESORICs 2020, guildford, UK, September 14–18, 2020, proceedings, part i 25*, pages 480–501. Springer, 2020.

[82] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

[83] Aleksei Triastcyn and Boi Faltings. Federated learning with bayesian differential privacy. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 2587–2596. IEEE, 2019.

[84] Ihsan Ullah, Umair Ul Hassan, and Muhammad Intizar Ali. Multi-level federated learning for industry 4.0-a crowdsourcing approach. *Procedia Computer Science*, 217:423–435, 2023.

[85] Xiujuan Wang, Kangmiao Chen, Keke Wang, Zhengxiang Wang, Kangfeng Zheng, and Jiayue Zhang. Fedkg: A knowledge distillation-based federated graph method for social bot detection. *Sensors*, 24(11):3481, 2024.

[86] Yuwei Wang, Runhan Li, Hao Tan, Xuefeng Jiang, Sheng Sun, Min Liu, Bo Gao, and Zhiyuan Wu. Federated skewed label learning with logits fusion. *arXiv preprint arXiv:2311.08202*, 2023.

[87] Zhibo Wang, Yuting Huang, Mengkai Song, Libing Wu, Feng Xue, and Kui Ren. Poisoning-assisted property inference attack against federated learning. *IEEE Transactions on Dependable and Secure Computing*, 20(4):3328–3340, 2022.

[88] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE transactions on information forensics and security*, 15:3454–3469, 2020.

[89] Kang Wei, Jun Li, Chuan Ma, Ming Ding, Sha Wei, Fan Wu, Guihai Chen, and Thilina Ranbaduge. Vertical federated learning: Challenges, methodologies and experiments. *arXiv preprint arXiv:2202.04309*, 2022.

[90] Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Terngrad: Ternary gradients to reduce communication in distributed deep learning. *Advances in neural information processing systems*, 30, 2017.

[91] Chuhan Wu, Fangzhao Wu, Lingjuan Lyu, Yongfeng Huang, and Xing Xie. Communication-efficient federated learning via knowledge distillation. *Nature communications*, 13(1):2032, 2022.

[92] Tong Xia, Jing Han, Abhirup Ghosh, and Cecilia Mascolo. Cross-device federated learning for mobile health diagnostics: A first study on covid-19 detection. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[93] Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li. Dba: Distributed backdoor attacks against federated learning. In *International conference on learning representations*, 2019.

[94] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Asynchronous federated optimization. *arXiv preprint arXiv:1903.03934*, 2019.

[95] Yi Xu, Changgen Peng, Weijie Tan, Youliang Tian, Minyao Ma, and Kun Niu. Non-interactive verifiable privacy-preserving federated learning. *Future Generation Computer Systems*, 128:365–380, 2022.

[96] Yunlu Yan and Lei Zhu. A simple data augmentation for feature distribution skewed federated learning. *arXiv preprint arXiv:2306.09363*, 2023.

[97] Hongwei Yang, Juncheng Li, Meng Hao, Weizhe Zhang, Hui He, and Arun Kumar Sangaiah. An efficient personalized federated learning approach in heterogeneous environments: a reinforcement learning perspective. *Scientific Reports*, 14(1):28877, 2024.

[98] Andrew C Yao. Protocols for secure computations. In *23rd annual symposium on foundations of computer science (sfcs 1982)*, pages 160–164. IEEE, 1982.

[99] Xin Yao, Chaofeng Huang, and Lifeng Sun. Two-stream federated learning: Reduce the communication costs. In *2018 IEEE Visual Communications and Image Processing (VCIP)*, pages 1–4. IEEE, 2018.

[100] Abbas Yazdinejad, Ali Dehghantanha, Reza M Parizi, Mohammad Hammoudeh, Hadis Karimipour, and Gautam Srivastava. Block hunter: Federated learning for cyber threat hunting in blockchain-based iiot networks. *IEEE Transactions on Industrial Informatics*, 18(11):8356–8366, 2022.

[101] Rui Ye, Zhenyang Ni, Chenxin Xu, Jianyu Wang, Siheng Chen, and Yonina C Eldar. Fedfm: Anchor-based feature matching for data heterogeneity in federated learning. *IEEE Transactions on Signal Processing*, 71:4224–4239, 2023.

[102] Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deepinversion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8715–8724, 2020.

[103] Han Yu, Zelei Liu, Yang Liu, Tianjian Chen, Mingshu Cong, Xi Weng, Dusit Niyato, and Qiang Yang. A fairness-aware incentive scheme for federated learning. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 393–399, 2020.

[104] Chengliang Zhang, Suyi Li, Junzhe Xia, Wei Wang, Feng Yan, and Yang Liu. {BatchCrypt}: Efficient homomorphic encryption for {Cross-Silo} federated learning. In *2020 USENIX annual technical conference (USENIX ATC 20)*, pages 493–506, 2020.

[105] Jie Zhang, Zhiqi Li, Bo Li, Jianghe Xu, Shuang Wu, Shouhong Ding, and Chao Wu. Federated learning with label distribution skew via logits calibration. In *International Conference on Machine Learning*, pages 26311–26329. PMLR, 2022.

[106] Rui Zhang, Song Guo, Junxiao Wang, Xin Xie, and Dacheng Tao. A survey on gradient inversion: Attacks, defenses and future directions. *arXiv preprint arXiv:2206.07284*, 2022.

[107] Shuyao Zhang, Jordan Tay, and Pedro Baiz. The effects of data imbalance under a federated learning approach for credit risk forecasting. *arXiv preprint arXiv:2401.07234*, 2024.

[108] Xianglong Zhang, Anmin Fu, Huaqun Wang, Chunyi Zhou, and Zhenzhu Chen. A privacy-preserving and verifiable federated learning scheme. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2020.

[109] Zhengming Zhang, Ashwinee Panda, Linyue Song, Yaoqing Yang, Michael Mahoney, Prateek Mittal, Ramchandran Kannan, and Joseph Gonzalez. Neurotoxin: Durable backdoors in federated learning. In *International Conference on Machine Learning*, pages 26429–26446. PMLR, 2022.

[110] Yang Zhao, Jun Zhao, Mengmeng Yang, Teng Wang, Ning Wang, Lingjuan Lyu, Dusit Niyato, and Kwok-Yan Lam. Local differential privacy-based federated learning for internet of things. *IEEE Internet of Things Journal*, 8(11):8836–8853, 2020.

[111] Tailin Zhou, Jun Zhang, and Danny HK Tsang. Fedfa: Federated learning with feature anchors to align features and classifiers for heterogeneous data. *IEEE Transactions on Mobile Computing*, 23(6):6731–6742, 2023.

[112] Xingchen Zhou, Ming Xu, Yiming Wu, and Ning Zheng. Deep model poisoning attack on federated learning. *Future Internet*, 13(3):73, 2021.

[113] Yanlin Zhou, George Pu, Xiyao Ma, Xiaolin Li, and Dapeng Wu. Distilled one-shot federated learning. *arXiv preprint arXiv:2009.07999*, 2020.

[114] Gongxi Zhu, Donghao Li, Hanlin Gu, Yuan Yao, Lixin Fan, and Yuxing Han. Fedmia: An effective membership inference attack exploiting" all for one" principle in federated learning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 20643–20653, 2025.

[115] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. In *International conference on machine learning*, pages 12878–12889. PMLR, 2021.