Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

FedMcon: an adaptive aggregation method for federated learning via meta controller*&

Tao SHEN[1], Zexi LI[1], Ziyu ZHAO[1], Didi ZHU[1],
Zheqi LV[1], Kun KUANG[1], Shengyu ZHANG[‡2], Chao WU[‡3,4], Fei WU[‡1]

[1]*College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China*
[2]*School of Software Technology, Zhejiang University, Hangzhou 310027, China*
[3]*School of Public Affairs, Zhejiang University, Hangzhou 310027, China*
[4]*Academy of Social Governance, Zhejiang University, Hangzhou 310027, China*
E-mail: tao.shen@zju.edu.cn; zexi.li@zju.edu.cn; ziyuzhao.cs@zju.edu.cn; didi_zhu@zju.edu.cn;
zheqilv@zju.edu.cn; kunkuang@zju.edu.cn; sy_zhang@zju.edu.cn; chao.wu@zju.edu.cn; wufei@zju.edu.cn
Received June 20, 2024; Revision accepted Dec. 15, 2024; Crosschecked Aug. 2, 2025

Abstract: Federated learning (FL) emerged as a novel machine learning setting that enables collaboratively training deep models on decentralized clients with privacy constraints. In the vanilla federated averaging algorithm (FedAvg), the global model is generated by the weighted linear combination of local models, and the weights are proportional to the local data sizes. This methodology, however, encounters challenges when facing heterogeneous and unknown client data distributions, often leading to discrepancies from the intended global objective. The linear combination-based aggregation often fails to address the varied dynamics presented by diverse scenarios, settings, and data distributions inherent in FL, resulting in hindered convergence and compromised generalization. In this paper, we present a new aggregation method, FedMcon, within a framework of meta-learning for FL. We introduce a learnable controller trained on a small proxy dataset and served as an aggregator to learn how to adaptively aggregate heterogeneous local models into a better global model toward the desired objective. The experimental results indicate that the proposed method is effective on extremely non-independent and identically distributed data and it can simultaneously reach 19 times communication speedup in a single FL setting.

Key words: Federated learning; Meta-learning; Adaptive aggregation
https://doi.org/10.1631/FITEE.2400530 **CLC number:** TP39

1 Introduction

In the field of machine learning, federated learning (FL) has been identified as a promising method for preserving privacy (McMahan et al., 2023). FL operates by collaboratively training a shared model among several decentralized clients, without requiring these clients to share their private data, thereby providing a basic level of privacy protection. However, as FL moves from theoretical frameworks to real-world deployments, it faces significant challenges that fundamentally impact its performance and applicability. The heterogeneous nature of client data, varying network conditions, and dynamic

‡ Corresponding authors
* Project supported by the National Key Research and Development Program of China (No. 2021ZD0110505), the Zhejiang Provincial Key Research and Development Project, China (No. 2023C01043), the National Natural Science Foundation of China (No. 62402429), the Key Research and Development Program of Zhejiang Province, China (No. 2024C03270), the ZJU Kunpeng & Ascend Center of Excellence, and the Ningbo Yongjiang Talent Introduction Programme, China (No. 2023A-397-G)
& A preliminary version was presented at the 6th ACM International Conference on Multimedia in Asia Workshops, Auckland, New Zealand, December 3–6, 2024
ORCID: Tao SHEN, https://orcid.org/0000-0003-0819-9782; Shengyu ZHANG, https://orcid.org/0000-0002-0030-8289; Chao WU, https://orcid.org/0000-0003-0885-6869; Fei WU, https://orcid.org/0000-0003-2139-8807

client behaviors create a complex optimization landscape that traditional approaches struggle to navigate effectively.

In the field of FL, a key challenge arises from the presence of heterogeneous data (McMahan et al., 2023), often referred to as the non-independent and identically distributed (non-IID) problem. This is because data are generated and retained across various clients, leading to significant variations in data distribution from one client to another. Traditional FL methods, which rely on fixed aggregation rules, often fail to adapt to these changing conditions, resulting in suboptimal convergence and reduced model quality. The complex interaction between client diversity and model optimization creates scenarios in which static aggregation strategies cannot effectively guide the learning process toward optimal solutions. Such diversity in the data landscape across clients may lead to significant convergence and performance degradation in the implementation of the federated averaging algorithm (FedAvg). To overcome the difficulties posed by non-IID data, various federated optimization techniques have been proposed. Karimireddy et al. (2021b) investigated the issue of client drift in cross-silo FL, where client updates are diverse and the resulting average model parameters differ from those obtained through centralized training. Yao et al. (2020) studied the objective inconsistency problem in cross-device FL, where biased client selection in each round causes the update of the aggregated model to deviate from the true global objective, since the updates only reflect the objectives of the selected clients rather than all clients. These different forms of issues will result in aggregation bias, characterized by a discrepancy from the intended optimization objective, leading to slow convergence and suboptimal results. Despite efforts to address these problems in previous research, it is difficult for an FL method to adapt across any FL contexts including various scenarios, settings, and data distributions (Huang et al., 2021; Karimireddy et al., 2021a). It is evident that an FL strategy might excel in one context but fail in another (Li QB et al., 2021b). The performance of FL methods is also greatly influenced by factors such as the number of local epochs and the number of participating clients, making it challenging to develop an adaptive FL method that can effectively accommodate diverse FL contexts.

To address these challenges, we propose a fundamentally different approach that moves beyond static aggregation rules to a learnable, adaptive framework. Unlike existing methods that rely on fixed aggregation strategies or heuristic adjustments, our approach treats the aggregation process itself as a learnable component that can dynamically adapt to different FL contexts. This represents a paradigm shift from traditional FL methods, as it enables the system to automatically learn and adjust its aggregation behavior based on the global optimization landscape. Inspired by the idea of learning an optimizer for improving optimization presented by Andrychowicz et al. (2016) and Bertinetto et al. (2016), our goal is to explore the possibility of an FL method that can adapt to any FL contexts. To achieve this goal, in this paper, we propose FedMcon, by incorporating meta-learning. The aggregator in this framework is viewed as a meta-controller that helps in learning how to adaptively aggregate the clients' model parameters. To this end, the learning objective of the meta-controller is to aggregate the model parameters from different clients into a proxy model (acting as a learner) that performs well on a proxy dataset. We propose an adaptive approach for debiasing aggregation, where the aggregator acts as a controller to calibrate the optimization direction of FL. This is accomplished by taking the clients' model parameters as the input of the aggregator, followed by adding a control variate for each client and averaging the parameters to produce the proxy model as output. By training the aggregator on the proxy dataset, the meta-controller can capture the "meta-knowledge" that refers to the ability to correct clients' models in a global view and thereby debiasing the aggregation for FL. Thus, the aggregator can guide the optimization process towards the global optimum, resulting in a more precise and efficient aggregation for client models, as illustrated in Fig. 1. Through comprehensive experiments, we demonstrate the effectiveness of our learning-based method across various scenarios of cross-silo FL and cross-device FL, under various settings of different levels of non-IID data, numbers of local epochs in cross-silo FL, and numbers of participating clients in each round of cross-device FL, on three different datasets. The results indicate the ability of our method to adapt to a wide range of FL scenarios, settings, and data distributions.

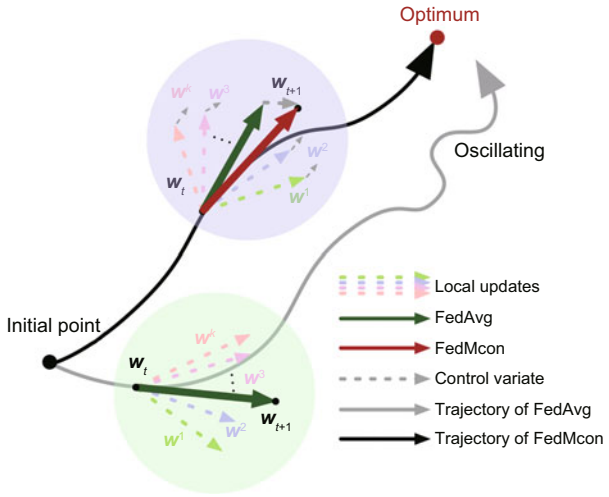

Fig. 1 Analysis of FedAvg and FedMcon under client drift and objective inconsistency. FedAvg may be affected by both issues, leading to oscillation around the global optimum. On the contrary, the FedMcon approach overcomes these limitations by introducing a control variate for each client that guides the optimization process towards the global optimum, thus resulting in a more precise and efficient aggregation for client models

We summarize the primary contributions of this paper as follows:

We present a novel learning-based optimization framework, FedMcon, that uses meta-learning, allowing it to be flexible and adaptive to different FL training scenarios, settings, and data distributions. We also introduce a method for debiasing aggregation by using the aggregator as a controller, which controls the optimization direction of FL by incorporating a control variate for each client's model parameters. Empirical evidence shows that the FedMcon framework effectively outperforms other methods for non-IID data in different FL scenarios, settings, and data distributions.

This paper is an extended version of our previous work published in the ACM International Conference on Multimedia in Asia Workshops (Shen et al., 2024). Compared to the workshop version, this journal paper provides extensions, including (1) extensive experimental evaluation on additional datasets and more comprehensive baseline comparisons, (2) deeper analysis of the aggregation bias problem and its solutions, (3) thorough ablation studies examining the impact of various components and hyperparameters, and (4) enhanced discussion of the relationship between meta-learning and federated optimization. These extensions demonstrate the robustness and general applicability of our proposed method across diverse federated learning scenarios.

2 Related works

2.1 FL with non-IID data

The canonical FedAvg is to train a global model in a distributed manner. The difference between FL and distributed learning (usually refers to distributed training in a data center) is whether the data of clients are fixed locally and cannot be accessed by others. This feature brings the safety of data privacy, but leads to the non-IID and unbalanced data distribution that makes the training process harder. The difficulty of training non-IID data is the accuracy reduction. Due to the non-IIDness, the fact of accuracy reduction can be understood in terms of weight divergence, which results in non-negligible deviation from correct updates at the stage of averaging. We also propose a data-sharing strategy by creating a small globally shared subset of data. This strategy can effectively improve the accuracy, and for privacy safety, the shared data can be extracted with distillation, or generated by a generative adversarial network (GAN). Many theoretical studies have been conducted on FedAvg by focusing on convergence analysis and relaxing the assumptions in the non-IID setting. However, these strategies cannot achieve comparable performance as in the IID setting.

The efficacy of FL is often affected by heterogeneous data distributed across multiple clients. This can reduce accuracy, as Zhao et al. (2022) demonstrated, attributing the phenomenon to weight divergence. To address this issue, Li T et al. (2020) proposed the use of FedProx, a proximal term designed to mitigate the heterogeneity in the data. Li QB et al. (2021a) presented comprehensive strategies for partitioning the non-IID data, which typically pose challenges for FL. Wang et al. (2020) provided insights into the number of local update epochs and introduced normalized averaging to address the objective inconsistency. Li XX et al. (2021) addressed the issue of feature shift non-IID problem in FL by proposing the use of local batch normalization to mitigate the shift before models are averaged.

Recent developments in FL have introduced

novel architectures and frameworks to address these challenges. Chen DS et al. (2023) proposed an elastic aggregation framework that adaptively adjusts the aggregation weights based on client model similarities, effectively handling client drift in non-IID settings. Similarly, Yan et al. (2023) provided new insights into client drift by analyzing it from a logit perspective, demonstrating that logit-level inconsistencies significantly affect model performance. They also proposed techniques to mitigate these inconsistencies through logit calibration. Additionally, federated split learning has emerged as a promising approach for handling sequential data in complex network architectures, particularly in satellite–terrestrial integrated networks (Jiang WW et al., 2024). This approach effectively manages the unique challenges of space–ground communications while maintaining data privacy. Practical deployment frameworks such as kubeFlower (Parra-Ullauri et al., 2024) have also demonstrated significant advances in scaling FL systems, providing robust solutions for real-world implementations.

2.2 Meta-learning

Meta-learning is a branch of machine learning that aims to improve the performance of learning algorithms. The goal of meta-learning is to tackle the "learning to learn" problem (Pramling, 1990). This approach has demonstrated its effectiveness in various domains, including reinforcement learning (Xu et al., 2018), few-shot learning (Nichol et al., 2018), and image classification (Ravi and Larochelle, 2016). Andrychowicz et al. (2016) proposed using deep neural networks to train a meta-learner by means of an optimizer–optimizee setup. The update rule was learned rather than hand-designed, and the components were iteratively learned through gradient descent. Additionally, Ravi and Larochelle (2016) proposed using a long short-term memory (LSTM) meta-learner to learn the optimization procedure for few-shot image classification. A model-agnostic meta-learning (MAML) method introduced by Finn et al. (2017) does not impose any constraints on the architecture of the learner. Reptile was derived from MAML, which simplifies the learning process by conducting first-order gradient updates on the meta-learner (Nichol et al., 2018).

2.3 Federated meta-learning

Meta-learning has several important applications in FL, including fast adaptation, continual learning, personalization, robustness, and efficiency in computation and communication. Jiang YH et al. (2023) highlighted the similarities between the MAML approach of fast, gradient-based adaptation to heterogeneous task distributions and the goal of personalization in FL. They observed that conventional federated averaging can be interpreted as a meta-learning algorithm. Li CL et al. (2021) proposed Meta-HAR to train a shared network for individual users, resulting in robust and personalized learning. Fallah et al. (2020) studied a personalized variant of FL, which seeks to find an initial shared model that can easily adapt to the local dataset of current or new users through one or a few steps of gradient descent. Lin YJ et al. (2020) designed a framework for rating prediction in mobile environments, incorporating a meta recommender module to generate private item embeddings and a rating prediction model based on collaboration. Other studies also use meta-learning in FL; Shamsian et al. (2021) proposed learning a central hypernetwork to generate personalized models, and Yao et al. (2020) presented FedMeta using a proxy dataset for unbiased model aggregation through meta update on the server. However, these methods have the risk of overfitting on the proxy dataset.

3 Methodology

The objective of conventional FL is to develop a shared global model based on decentralized data. As the data cannot be centralized in a server due to privacy concerns, they are stored on multiple devices. One common FL approach, FedAvg, aggregates local model updates through a weighted average approach, represented by $\boldsymbol{w}^{\mathrm{global}} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} \boldsymbol{w}^k$. Here, $\boldsymbol{w}^{\mathrm{global}}$ denotes the global model parameters maintained by the server, $\boldsymbol{w}^k$ represents the local model parameters of the k^{th} client, n_k denotes the number of local data samples held by the k^{th} client, $n = \sum_{k=1}^{K} n_k$ is the total number of data samples across all clients, and K is the total number of participating clients. However, FedAvg encounters a significant decrease in accuracy in the non-IID scenario, where the data distribution $\mathcal{P}(x, y)$ of the overall dataset differs across clients,

i.e., $\mathcal{P}_k(x,y) \neq \mathcal{P}_j(x,y)$ for clients k and j. This paper investigates the non-IID problem in the context of FedAvg and presents a novel meta-learning-based framework to address it.

3.1 Typical FL setup

In FedAvg, the aim is to learn a single shared global model through decentralized data to minimize the global objective function $f(\boldsymbol{w}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{w})$. This objective is the sum of the loss function over all private data $\mathcal{D}_{\text{private}}$, generated by distinct distributions $\mathcal{P}_k(x,y)$ from K clients. The combination of these decentralized private data makes up the training dataset for FL. To minimize the global objective, FedAvg first copies the global model parameters $\boldsymbol{w}_t^k \in \mathbb{R}^d$ to a set of candidate clients. Each candidate then performs a local update by optimizing its local objective through gradient descent for a specified number of epochs.

$$\begin{aligned} F_k(\boldsymbol{w}_t^k) &= \frac{1}{n_k}\sum_{i\in\mathcal{P}_k} f_i(\boldsymbol{w}_t^k), \\ \boldsymbol{w}_t^k &\leftarrow \boldsymbol{w}_t^k - \eta\nabla F_k\left(\boldsymbol{w}_t^k, \mathcal{D}_{\text{private}}^k\right), \end{aligned} \tag{1}$$

where $F_k(\boldsymbol{w}_t^k)$ is the local objective of the k^{th} client, n_k is the number of local samples, η is the local learning rate, and $\nabla F_k\left(\boldsymbol{w}_t^k, \mathcal{D}_{\text{private}}^k\right) \in \mathbb{R}^d$ is the gradient vector. Following the local updates, clients transmit their local model parameters $\boldsymbol{w}_t^k$ to the server and then combine these parameters through weighted averaging:

$$\boldsymbol{w}_{t+1}^{\text{global}} \leftarrow \sum_{k=1}^{K}\frac{n_k}{n}\boldsymbol{w}_t^k, \tag{2}$$

where $\boldsymbol{w}_{t+1}^{\text{global}}$ is the parameter of the global model. This training process is repeated until the global model reaches convergence, allowing the shared global model to be trained collaboratively without revealing private data.

However, when the data distribution $\mathcal{P}_k(x,y)$ of the k^{th} client differs from those of other clients or the overall data distribution, the expected value of the local objective $F_k(\boldsymbol{w})$ may not align with the global objective $f(\boldsymbol{w})$. This mismatch between local and global objectives is especially pronounced in non-IID settings, where $\mathcal{P}_k(x,y) \neq \mathcal{P}_j(x,y)$ or $\mathcal{P}_k(x,y) \neq \mathcal{P}_{\text{overall}}(x,y)$. This deviation, referred to as aggregation bias, can result in slow convergence and suboptimal outcomes when implementing the FedAvg algorithm, as it mismatches with the intended optimization goal. While previous studies have attempted to address these issues, their applicability is limited to specific cases and settings (Yao et al., 2020; Huang et al., 2021; Karimireddy et al., 2021a, 2021b). Thus, it is still challenging to develop an adaptive FL method that can effectively accommodate diverse scenarios, settings, and data distributions. In contrast to traditional FL approaches, which typically rely on fixed aggregation rules to combine client updates, our proposed framework, FedMcon, introduces a novel meta-learning-based approach that dynamically adjusts the aggregation strategy based on the current state of the system and historical client behaviors. Unlike traditional FL methods that use fixed aggregation rules, FedMcon's learnable meta-controller enables more precise control by actively debiasing the aggregation process through a unique control variate mechanism. Furthermore, FedMcon incorporates a novel feedback loop through proxy data, allowing the system to continuously refine its aggregation strategy based on the observed performance. This adaptive approach fundamentally differs from existing methods in three key aspects: (1) dynamic aggregation strategy adjustment, (2) active debiasing through control variates, and (3) continuous refinement through proxy data feedback.

3.2 Learning-based FL framework

To address the aforementioned challenge, we introduce a novel learning-based optimization framework for FL called FedMcon. This framework is motivated by the concept of learning an optimizer for optimization improvement as proposed by Andrychowicz et al. (2016) and Bertinetto et al. (2016). Our objective is to investigate the feasibility of an FL method that is capable of adapting to various FL training tasks, each with its own distinct scenarios, configurations, and data characteristics. To achieve this goal, we use meta-learning, a technique known for its ability to enhance the efficiency of the learning process.

3.2.1 Meta-learning

Unlike conventional machine learning approaches, the objective of meta-learning is not merely to train a model to perform well on a single

task, but to enable the model to learn to learn, thus making the learning process more efficient. This is accomplished by training a meta-learner that operates at a higher level of abstraction than traditional learners. The meta-learner exploits the information obtained from multiple lower-level learners, each of which is trained on specific tasks, to enhance the overall learning process. Unlike traditional machine learning which uses a training set and a test set, a meta-learner is trained by exploring the optimization direction on a support set and updating it on a query set. This innovative learning mechanism enables the meta-learner to acquire higher-level knowledge and thus facilitates the learning process for lower-level learners.

3.2.2 Learning-based optimization

Taking inspiration from the aforementioned approach, we integrate this concept into the training of FL to enhance its performance. In our approach, we view the aggregator as the meta-learner and replace the averaging operator in FedAvg with a deep learning model, denoted by

$$\boldsymbol{\theta}_{\text{learner}} = \text{aggr}(\boldsymbol{\Theta}, \boldsymbol{\phi}_{\text{meta-controller}}), \tag{3}$$

where aggr is the aggregation function. The meta-learner model is parameterized by $\boldsymbol{\phi}_{\text{meta-controller}}$ that takes the features from clients, represented by $\boldsymbol{\Theta}$ as input. The features could be the clients' model parameters or any other relevant information. The meta-learner then outputs the learner's model parameters, denoted by $\boldsymbol{\theta}_{\text{learner}}$. In our meta-learning-based FL framework, each client's private dataset $\mathcal{D}^k_{\text{private}}$ is used as the support set for local adaptation, while the server maintains a proxy dataset $\mathcal{D}_{\text{proxy}}$ that acts as the query set for evaluating and updating the meta-learner. To adapt to a specific FL task, we can train the aggregator over a query set $\mathcal{D}_{\text{query}}$ on the server. The meta-learner can be trained at any point during the FL process, such as at each communication round, and can be optimized using the following objective function:

$$\min_{\boldsymbol{\phi}_{\text{meta-controller}}} \text{loss}(\boldsymbol{\theta}_{\text{learner}}, \mathcal{D}_{\text{query}}), \tag{4}$$

where $\boldsymbol{\theta}_{\text{learner}} = \text{aggr}(\boldsymbol{\Theta}, \boldsymbol{\phi}_{\text{meta-controller}})$. This framework is depicted in the left part of Fig. 2. The advantage of this framework is the flexibility to design a custom loss function based on the specific FL task, using the task-oriented proxy dataset. For instance, a loss function can be designed for robust aggregation (filtering out malicious or corrupted clients), improving generalization (global model), or boosting personalization (local models). The objective of this study is to address the issue of aggregation bias and enhance the learning process in FL, making it adaptable to various scenarios, settings, and data distributions.

3.3 Feedback control for aggregation bias

In this subsection, we apply the FedMcon framework specifically to address the problem of aggregation bias and demonstrate how it can be adaptive to various scenarios, settings, and data distributions.

3.3.1 FedAvg lacking control

The concept of control theory has inspired the perspective of FL being viewed as a dynamic system (Haddad and Chellaboina, 2008). In this perspective, the model parameters $\boldsymbol{w}_t$ are treated as the system states. The FedAvg approach in FL operates as a feedforward process without any control. The presence of the non-IID data makes it uncertain if the aggregated model goes toward the intended update direction. The non-IID data and the lack of control can result in unstable and potentially oscillating model updates. The dynamic process of FL in Eqs. (1) and (2) can be written as follows (in fact, the number of samples of the k^{th} client n_k is usually unknown to the server, and thus we set $\frac{n_k}{n}$ as $\frac{1}{K}$):

$$\boldsymbol{w}_{t+1} = g(\boldsymbol{w}_t) = \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{w}_t^k = \frac{1}{K}\sum_{k=1}^{K}(\boldsymbol{w}_t - \Delta \boldsymbol{w}_t^k), \tag{5}$$

where $\Delta \boldsymbol{w}_t^k$ represents the change in the local update, obtained by minimizing $F_k(\boldsymbol{w}_t^k)$ over several epochs, and $g(\boldsymbol{w}_t)$ denotes the state transfer function of the model parameters $\boldsymbol{w}_t$, which defines the trajectory of the model parameters. However, the presence of the non-IID data can result in the objective function $F_k(\boldsymbol{w}_t^k)$ being a poor approximation to the global objective $f(\boldsymbol{w}_t)$, leading to aggregation bias (client drift or objective inconsistency). To address this issue, our proposed solution involves controlling the local updates towards the optimum of the global objective by intervening in the trajectory of FL via adding a control variate $\boldsymbol{u}_t^k$ for each client. The state transfer function $g_{\text{c}}(\boldsymbol{w}_t)$ incorporating the

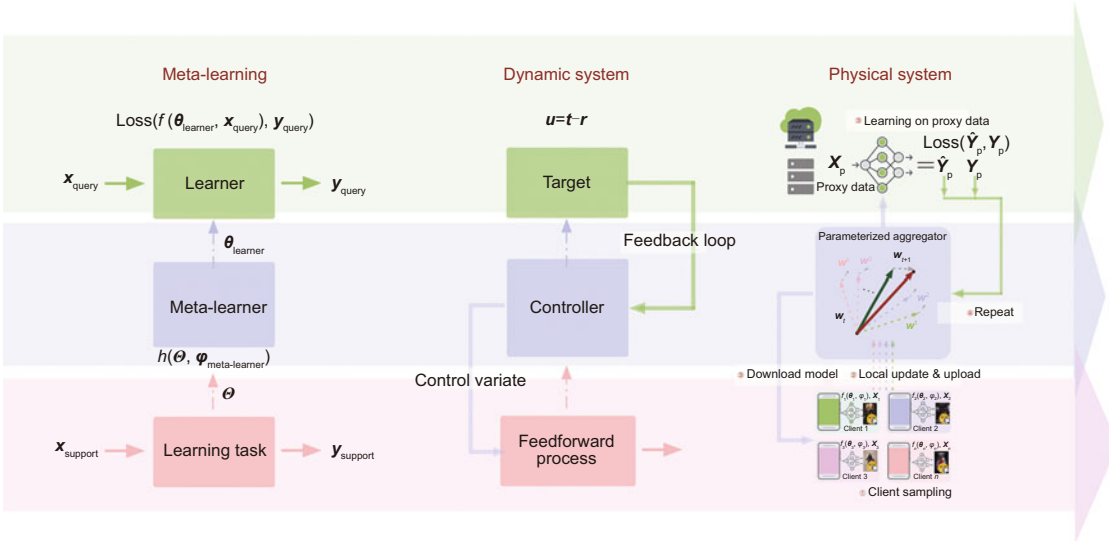

Fig. 2 Relations between meta-learning, dynamic system, and FedMcon. Left: the structure of the general learning-based framework that incorporates meta-learning. The private data act as the support set, while the proxy data act as the query set. The role of the meta-learner is to expedite the learning process. Middle: the process of FL is depicted as a dynamic evolution of the weights w_t, which is difficult to control due to the non-IID nature of the data. The proxy data serve as the target in the control loop, providing feedback to the controller to guide the trajectory of w_t towards the global optimum. Right: the pipeline of the FedMcon includes local updates, model aggregation, and the aggregator training. In the figure, elements with the same color have the same role viewed from different perspectives. The learner serves as the target in the dynamic system and as the proxy model in FL. The meta-learner acts as the controller in the dynamic system and as the aggregator in FL. The learning task corresponds to the feedforward process in the dynamic system and the clients' local updates in FL. References to color refer to the online version of this figure

control variate is formulated as follows:

$$\boldsymbol{w}_{t+1} = g_c(\boldsymbol{w}_t) = \frac{1}{K}\sum_{k=1}^{K}(\boldsymbol{w}_t - \Delta\boldsymbol{w}_t^k(1-\boldsymbol{u}_t^k)), \quad (6)$$

where the control variate $\boldsymbol{u}_t^k$ is introduced element-wise to each $\Delta\boldsymbol{w}_t^k$. The challenge lies in determining an effective control variate that directs the model parameters $\boldsymbol{w}_t$ towards the global optimum. The proposed FedMcon framework addresses this issue by using a meta-learning technique to obtain feedback from the evaluation of proxy data. As the proxy data are assumed to have a similar distribution to the overall dataset, the control variate $(1-\boldsymbol{u}_t^k)$ derived from the proxy data can potentially calibrate the model update towards the intended direction.

3.3.2 Learning to aggregate with feedback

To achieve this, the control variate $\boldsymbol{u}_t^k$ is defined as a function of $\boldsymbol{w}_t$, $\Delta\boldsymbol{w}_t^k$, and a set of parameters $\boldsymbol{\phi}$: $\boldsymbol{u}_t^k = h(\boldsymbol{w}_t, \Delta\boldsymbol{w}_t^k, \boldsymbol{\phi})$. Unlike traditional control methods that use fixed rules, our meta-controller h is implemented as a neural network that learns to adapt its behavior based on both the current state of the global model $\boldsymbol{w}_t$ and the local updates $\Delta\boldsymbol{w}_t^k$ from clients. The adaptive nature of our controller enables several key capabilities. First, it can dynamically adjust to varying degrees of non-IID data distributions, ensuring robust performance across different data scenarios. Second, it learns optimal aggregation strategies for different model architectures, making it versatile across various deep learning tasks. Third, it automatically balances between local optimization and global consensus, preventing both model drift and premature convergence. The controller is trained on proxy data to get feedback, and the resulting control variate $\boldsymbol{u}_t^k$ can help guide the model updates towards the intended direction, which is illustrated in the middle part of Fig. 2. To accomplish this, we integrate the controller and the averaging operator into the aggregator, resulting in an updated aggregation function:

$$\begin{aligned}&\mathrm{aggr}(\boldsymbol{w}_t, \Delta\mathcal{W}_t, \boldsymbol{\phi})\\ =&\frac{1}{K}\sum_{k=1}^{K}(\boldsymbol{w}_t - \Delta\boldsymbol{w}_t^k(1-h(\boldsymbol{w}_t, \Delta\boldsymbol{w}_t^k, \boldsymbol{\phi}))),\end{aligned} \quad (7)$$

where $\Delta\mathcal{W}_t = \{\Delta \boldsymbol{w}_t^k\}$ is the set of local updates from selected clients at the t^{th} round and serves as the input feature for the aggregator. Finally, the dynamic equation with aggregation function (6) is formulated as follows:

$$\boldsymbol{w}_{t+1} = \text{aggr}(\boldsymbol{w}_t, \Delta\mathcal{W}_t, \boldsymbol{\phi}). \tag{8}$$

The implementation of an effective aggregator within a meta-learning framework is the core idea behind FedMcon. In this framework, the aggregator acts as a "meta-learner," using a set of proxy data as the query set. On the contrary, each client in FL can be considered a "learner" whose private data serve as the support set. The aggregated model that is trained on the proxy data is referred to as the proxy model. The optimization objective of the proxy model on the proxy data is designed to align with the global objective, as the assumption is that the better the performance of the proxy model on the proxy data, the better the aggregator becomes. Hence, the aggregator can be optimized by the objective function (4) as follows:

$$\min_{\boldsymbol{\phi}} f(\boldsymbol{w}_{\text{proxy}}, \mathcal{D}_{\text{proxy}}), \tag{9}$$

where $\boldsymbol{w}_{\text{proxy}} = \text{aggr}(\boldsymbol{w}_t, \Delta\mathcal{W}_t, \boldsymbol{\phi})$. The parameters $\boldsymbol{\phi}$ are trained on the proxy data, allowing the aggregator to learn how to effectively aggregate model parameters by providing control variates for each client's model update, thereby reducing bias in the aggregation process. The pipeline of FedMcon is depicted in Fig. 2.

3.3.3 Structure of the aggregator model

As outlined in Eq. (8), the aggregator takes model updates as the input and produces the proxy model as the output. The aggregator has two modules, $\boldsymbol{\phi}_{\text{g}}$ and $\boldsymbol{\phi}_{\text{c}}$, one for the state of the global model $\boldsymbol{w}_t$, and the other for the model updates from clients $\boldsymbol{w}_t^k$. For model parameters $\boldsymbol{w}$ with dimension d, the ideal dimension of $\boldsymbol{\phi}_{\text{g}}$ (or $\boldsymbol{\phi}_{\text{c}}$) would be $d \times d$. However, for deep learning models, the dimension of the model can be quite large, leading to a dimension explosion problem ($\boldsymbol{\phi} \in \mathbb{R}^{d\times d}$ if $\boldsymbol{w} \in \mathbb{R}^d$). To address this issue, a bottleneck architecture is employed, mapping the parameters to a lower-dimensional space (e.g., p) and restoring the output to the original dimension. For example, the input layers of $\boldsymbol{\phi}_{\text{g}}$ and $\boldsymbol{\phi}_{\text{c}}$ are $\boldsymbol{\phi}_{\text{g}_{\text{in}}}, \boldsymbol{\phi}_{\text{c}_{\text{in}}} \in \mathbb{R}^{d\times p}$, mapping the model parameters and updates into a hidden space, respectively. The hidden spaces are concatenated and followed by an output layer $\boldsymbol{\phi}_{\text{out}} \in \mathbb{R}^{2p\times d}$. As a result, the dimension of $\boldsymbol{\phi} = \{\boldsymbol{\phi}_{\text{g}}, \boldsymbol{\phi}_{\text{c}}\}$ is $4 \times d \times p \times m$ for $\boldsymbol{w} \in \mathbb{R}^d$. The control variate $\boldsymbol{u}_t^k$ is finally formulated with the original dimension by adding it to each client's model update. To reduce the size of the parameters $\boldsymbol{\phi}$ of the aggregator, the setting $p = \log_2 d$ is applied. The control variate can then be formulated as follows:

$$\boldsymbol{u}_t^k = (\boldsymbol{w}_t, \Delta\boldsymbol{w}_t^k, \boldsymbol{\phi}) = \boldsymbol{\phi}_{\text{out}}(\boldsymbol{\phi}_{\text{g}_{\text{in}}}(\boldsymbol{w}_t), \boldsymbol{\phi}_{\text{c}_{\text{in}}}(\Delta\boldsymbol{w}_t^k)). \tag{10}$$

3.3.4 Pipeline of FedMcon

The training process of FedMcon consists of two parallel optimization processes: the traditional FL process and the meta-learning process for the aggregator. Specifically, it involves the following key steps: (1) the server randomly selects a subset of participating clients $\mathcal{K}_t$ based on a predefined sampling rate; (2) the selected clients perform local model optimization on their private datasets for multiple epochs and compute their model updates $\Delta\mathcal{W}_t$; (3) in the meantime, the server optimizes the aggregator parameters $\boldsymbol{\phi}$ using the proxy dataset to improve the controlled aggregation strategy; (4) the server applies the learned control variates through the aggregator to combine client updates into a new global model following Eq. (8); (5) this process iterates until convergence or reaching the maximum number of communication rounds. This whole pipeline is detailed in Algorithm 1. The algorithm requires global model $\boldsymbol{w}_t$, proxy dataset $\mathcal{D}_{\text{proxy}}$ on the server, clients indexed by k with local models $\boldsymbol{w}_t^k$ and private datasets $\mathcal{D}_{\text{private}}^k$, local learning rate η_{l}, the number of local epochs E_{l}, the number of epochs for training aggregator E_{g}, and the total number of communication rounds T.

4 Experiments

4.1 Setup

4.1.1 Datasets and models

In this study, we evaluate the performance of FedMcon and several state-of-the-art FL methods on both recommendation and computer vision datasets. The recommendation dataset used is the MovieLens 1M dataset available at

Algorithm 1 FedMcon: a meta-learning-based federated learning framework with controlled model aggregation

Server executions:

Initialize the global model $\boldsymbol{w}_0$ and the aggregator parameters $\boldsymbol{\phi}$
for each round $t = 0, 1, \ldots, T$ **do**
  Randomly sample a set of candidate clients $\mathcal{K}$ with sampling rate C
  Execute in the meantime:
  (a) $\Delta\mathcal{W}_t \leftarrow \text{ClientUpdate}(\boldsymbol{w}_t, \mathcal{K})$
  (b) Optimize the aggregator parameters $\boldsymbol{\phi}$ using proxy data $\mathcal{D}_{\text{proxy}}$ for E_{g} epochs
  Aggregate updates using the controlled aggregation: $\boldsymbol{w}_{t+1} = \text{aggr}(\boldsymbol{w}_t, \Delta\mathcal{W}_t, \boldsymbol{\phi})$
  Broadcast $\boldsymbol{w}_{t+1}$ to the selected clients
end for

ClientUpdate:

for each client $k \in \mathcal{K}$ **in parallel do**
  Receive the global model: $\boldsymbol{w}_t^k \leftarrow \boldsymbol{w}_t$
  Initialize the local model with the global parameters
  for each local epoch $e = 1, 2, \ldots, E_{\text{l}}$ **do**
    for each batch $b \in \mathcal{D}_{\text{private}}^k$ **do**
      Local update: $\boldsymbol{w}_t^k \leftarrow \boldsymbol{w}_t^k - \eta_{\text{l}} \nabla F_k\left(\boldsymbol{w}_t^k, b\right)$
    end for
  end for
  Perform the model update: $\Delta\boldsymbol{w}_t^k \leftarrow \boldsymbol{w}_t - \boldsymbol{w}_t^k$
  Upload $\Delta\boldsymbol{w}_t^k$ to the server
end for
Return $\Delta\mathcal{W}_t = \{\Delta\boldsymbol{w}_t^k\}_{k\in\mathcal{K}}$

https://grouplens.org/datasets/movielens/ (Harper and Konstan, 2015), containing 1 000 209 ratings provided by 6040 unidentifiable users on 3706 movies. The click-through rate (CTR) task is performed using the popular DIN model (Zhou et al., 2018). The evaluation is carried out using the widely adopted leave-one-out protocol (Muhammad et al., 2020) where for each user, its latest interaction is held out as the test set and the rest data are used as the training set. The user feedback is binarized. For each positive instance, four negative instances are sampled in the training set (i.e., a negative-to-positive ratio of 4:1), and 99 negative instances are sampled in the test set (i.e., a negative-to-positive ratio of 99:1), with the number of positive instances as the baseline. The computer vision dataset used is the FEMNIST dataset available at https://github.com/TalwalkarLab/leaf/tree/master/data/femnist (Caldas et al., 2019), which contains 62 classes of 28×28 pixel images of handwritten digits, lowercase and uppercase letters contributed by 3400 users. The dataset includes 671 585 training examples and 77 483 test samples. It is performed using the lightweight LeNet5 model (LeCun et al., 1998). Another computer vision dataset is the CIFAR-10 dataset available at https://www.cs.toronto.edu/~kriz/cifar.html (Krizhevsky and Hinton, 2009), which consists of 10 classes of 32×32 pixel images. The dataset includes 50 000 training examples and 10 000 test samples. It is performed using ResNet18 by replacing batch norm with group norm (Reddi et al., 2021). For methods requiring a proxy dataset (FedMcon, FedMeta, and FedDF), we consistently use a disjoint subset comprising 1% of the total FL training data (e.g., for CIFAR-10, the proxy size is 500 as the training size is 50 000).

4.1.2 FL settings

For the MovieLens dataset, which has a naturally non-IID distribution, it is split into 6040 clients based on the feature of user_id. The FEMNIST dataset is split into 3400 clients and the label distribution skew is simulated using the Dirichlet distribution with the hyperparameter α controlling the degree of non-IIDness (Lin T et al., 2020; Yao et al., 2020). The CIFAR-10 dataset is split into 10 clients using the Dirichlet distribution for the cross-silo FL scenario. For FL training, $T = 200$ communication rounds are set for MovieLens and CIFAR-10, and $T = 1500$ for FEMNIST. For the cross-device FL scenario of MovieLens, 10% clients are sampled per round. For FEMNIST, 10/20/30 clients are sampled. Each client trains $E_{\text{l}} = 1$ epoch locally. For the recommendation task, the Adam optimizer is used, while for the computer vision task, stochastic gradient descent (SGD) is employed as the local optimizer, both with a local learning rate of $\eta_{\text{l}} = 0.01$. For methods with server-side optimization, the Adam optimizer is used with a global learning rate of $\eta_{\text{g}} = 0.01$. The detailed hyperparameter settings for all methods are provided in the Appendix.

4.1.3 Baselines

The proposed FedMcon framework is compared with several state-of-the-art FL methods, including: (1) vanilla FL method FedAvg (McMahan et al., 2023); (2) client-side FL method FedProx (Li T et al., 2020); (3) server-side FL method without proxy data FedAvgM (Hsu et al., 2019); (4) server-side FL

method with proxy data FedDF (Lin T et al., 2020); (5) server-side federated meta-learning method with proxy data FedMeta (Yao et al., 2020) and another two FL methods for vision tasks FedCSD (Yan et al., 2023) and Elastic (Chen DS et al., 2023).

It is important to note that FedMcon is orthogonal to other meta-learning-based FL methods which are designed for model initialization (Chen F et al., 2019), or for personalization (Fallah et al., 2020; Shamsian et al., 2021); as a result, they are not included in comparisons.

4.1.4 Evaluation metrics

In the experiments, for the CTR task, the model performance is evaluated using the following metrics: area under the curve (AUC), hit rate (HR), and normalized discounted cumulative gain (NDCG).

$$\mathrm{AUC}=\frac{\sum_{x_0\in D_{\mathrm{T}}}\sum_{x_1\in D_{\mathrm{F}}}\mathbf{1}\left[f\left(x_1\right)<f\left(x_0\right)\right]}{\left|D_{\mathrm{T}}\right|\left|D_{\mathrm{F}}\right|},$$

$$\mathrm{HR@}K=\frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\mathbf{1}\left(R_{u,g_u}\leq K\right),$$

$$\mathrm{NDCG@}K=\sum_{u\in\mathcal{U}}\frac{1}{|\mathcal{U}|}\frac{2^{\mathbf{1}(R_{u,g_u}\leq K)}-1}{\log_2\left(\mathbf{1}\left(R_{u,g_u}\leq K\right)+1\right)},$$

where $\mathcal{U}$ is the set of users, $\mathbf{1}$ is the indicator function, R_{u,g_u} is the rank generated by the model for the ground truth item g_u, f is the model being evaluated, and D_{T} and D_{F} are the positive and negative sample sets in the test data, respectively. For the image classification task, the model performance is measured by the widely used top-1 accuracy metric.

4.1.5 Implementation

The experiments are run on a deep learning server equipped with an NVIDIA Tesla RTX8000 GPU using PyTorch. The FL environment including clients is simulated to evaluate the performance of the proposed FedMcon framework.

4.2 Analysis

4.2.1 Performance on MovieLens

We used the inherently non-IID MovieLens 1M dataset to evaluate FedMcon for real-world industrial applications. The CTR task samples are particularly non-IID, as each user has a unique profile, including user ID, age, gender, and so on, and each user has a limited number of movie ratings, resulting in only a small number of updated embedding tables in the model. Table 1 compares the performance of several FL algorithms on MovieLens 1M. The results show that the FedMcon algorithm dominantly outperforms the other algorithms in all five metrics, achieving the highest scores. Table 2 compares the number of communication rounds needed by the FL algorithms to reach 90% of their averaged performance when the number of local epochs is fixed to 1. The number of communication rounds is a measure of the efficiency of the FL algorithms. The results show that FedMcon is consistently faster than the other algorithms, requiring a smaller number of communication rounds to reach the desired level of performance. As shown in Fig. 3, for methods that use proxy datasets, FedDF and FedMeta performed poorer than FedMcon. FedDF, with its logistic regression model that only has one output, has limited performance improvement from ensemble distillation on the proxy dataset. Additionally, FedMeta may suffer from overfitting and oscillations on the proxy data. Although the number of communication rounds for FedMcon is the same as that for FedMeta (the fastest among the other algorithms), FedMcon is a highly efficient FL algorithm. In contrast, FedMcon outperforms other methods because it is capable of handling objective inconsistencies, where different objectives may arise at different numbers of communication rounds, even around the global optimum. Furthermore, FedMcon exhibits fast and steady convergence and reaches a better optimum compared to other FL methods.

Table 1 Metrics on the MovieLens 1M dataset

Method	AUC	HR@5	HR@10	NDCG@5	NDCG@10
FedAvg	0.7482	0.2916	0.4290	0.1901	0.2346
FedAvgM	0.7501	0.2909	0.4293	0.1932	0.2364
FedProx	0.7459	0.2924	0.4298	0.1914	0.2358
FedDF	0.7053	0.2553	0.3623	0.1701	0.2046
FedMeta	0.7651	0.2930	0.4429	0.1919	0.2404
FedMcon	**0.8117**	**0.3058**	**0.4517**	**0.2032**	**0.2503**

The best results are in bold

4.2.2 Visualizing objective inconsistency

Our experiments were performed on the FEMNIST dataset, which is designed to provide non-IID data. To quantify the degree of non-IIDness, we set the Dirichlet hyperparameter α to 1, 0.1, and

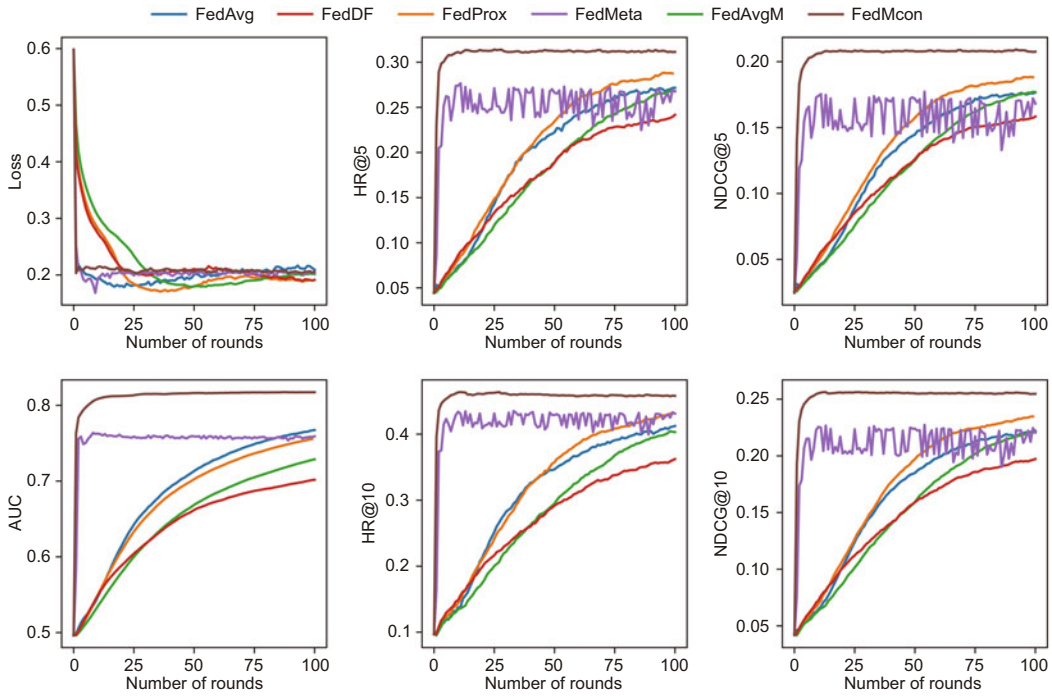

Fig. 3 Learning curves of FedMcon and baselines on the MovieLens 1M dataset

Table 2 Number of communication rounds to reach 90% of the averaged metrics on the MovieLens 1M dataset

Method	AUC	HR	NDCG
FedAvg	37 (1×)	63 (1×)	71 (1×)
FedAvgM	48 (0.8×)	75 (0.8×)	77 (0.9×)
FedProx	38 (1×)	57 (1.1×)	51 (1.4×)
FedDF	49 (0.7×)	–	184 (1×)
FedMeta	**2** (19×)	**2** (32×)	3 (24×)
FedMcon	**2** (19×)	**2** (32×)	**2** (36×)

The best results are in bold. The value in parentheses indicates the speedup ratio. "–" indicates that the target was not reached by the end of training

0.01. Fig. 4 presents the results of the non-IIDness analysis by visualizing the distribution of the 10-digit labels among the 62 classes over the first 5 communication rounds. We selected 10 clients from a pool of 100 clients. These 10 clients were used to demonstrate the changing distribution of label classes over the first 5 communication rounds. It is observed that as the non-IIDness of the data increases, the distribution of labels among the clients becomes more diverse. This diversity is shown through the blocks of color in the histogram, with each block representing the amount of data belonging to a specific label. Within each subfigure, the variation in color indicates the degree of client drift, as the length of the bar representing each label becomes more diverse. Additionally, within each column, the length of the bars illustrates the objective inconsistency, as the distribution of labels becomes increasingly inconsistent between different communication rounds. This is a clear indication that as the non-IIDness of the data increases, the challenge of training an adaptive and accurate model becomes more pronounced.

4.2.3 Performance on FEMNIST

For the cross-device FL scenario, we conducted an experiment on the FEMNIST dataset to evaluate the performance of FedMcon. Table 3 shows the results of each method when α is set to 1, 0.1, and 0.01. The number of clients in FEMNIST was 3400. The top-1 accuracy was reported for different numbers of sampled clients, varying in $\{10, 20, 30\}$. The rightmost column in each row of the table presents the relative improvement over the baseline (FedAvg with $|\mathcal{K}| = 10, \alpha = 1/0.1/0.01$) for each method. Table 4 presents the number of communication rounds to reach 60% test accuracy for each method. Under different levels of non-IIDness and with varying numbers of participating clients in each round, the results reveal that FedMcon consistently outperforms the compared FL methods. Despite the increasing

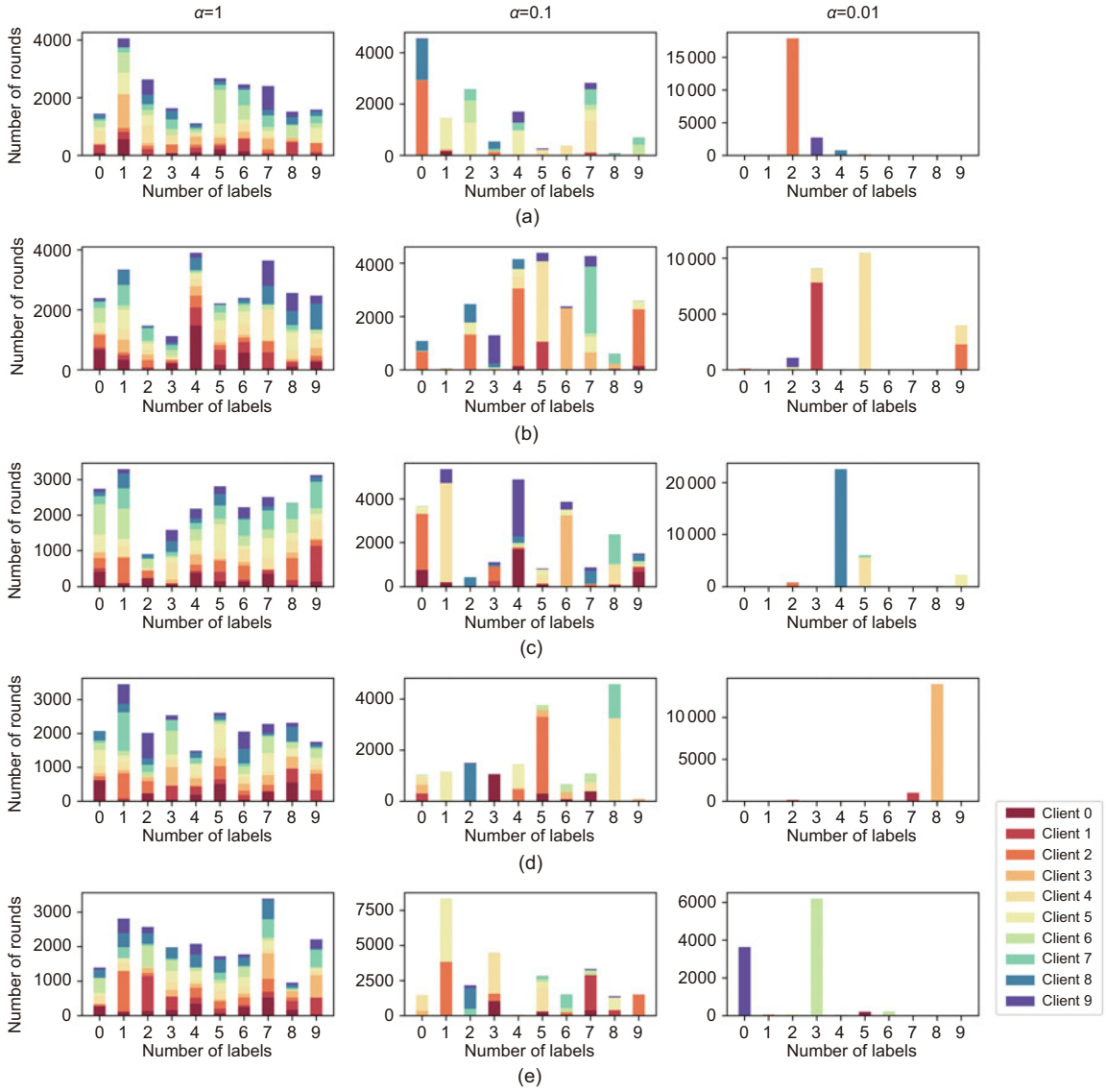

Fig. 4 Visualizing objective inconsistency on the FEMNIST dataset with different numbers of initial training rounds: (a) 1; (b) 2; (c) 3; (d) 4; (e) 5

degree of non-IIDness, FedMcon demonstrates little performance degradation. This is due to the well-trained aggregator in FedMcon, which provides a global view to calibrate the model parameters, resulting in good performance even in extremely non-IID conditions.

4.2.4 Performance on CIFAR-10

We compared the performance of several FL algorithms on the CIFAR-10 dataset to simulate the cross-silo settings. For CIFAR-10, the number of clients was 10 and full participation of the clients was achieved in each round. The top-1 accuracy of each algorithm is shown in Table 5, and the number of communication rounds to reach 60% test accuracy is shown in Table 6. In Table 5, the top-1 accuracy of each algorithm is reported for three different values of α (1, 0.1, and 0.01). The results show that FedMcon consistently outperforms the other algorithms, and it reaches the largest performance gain at $\alpha = 0.01$ (the most non-IID setting). In Table 6, FedMcon is shown to be the fastest algorithm, requiring a significantly smaller number of communication rounds to reach the target accuracy compared to the other methods, also with the most dominant improvement at $\alpha = 0.01$. Overall, the results suggest

Table 3 Top-1 accuracies on the FEMNIST dataset with a varying number of sampled clients

Method	$\|\mathcal{K}\|$	Top-1 accuracy					
		$\alpha=1$		$\alpha=0.1$		$\alpha=0.01$	
FedAvg	10	0.8206	(1.00×)	0.8207	(1.00×)	0.7439	(1.00×)
	20	0.8393	(1.02×)	0.8303	(1.01×)	0.7807	(1.05×)
	30	0.8398	(1.02×)	0.8347	(1.01×)	0.7955	(1.08×)
FedAvgM	10	0.8350	(1.02×)	0.8208	(1.00×)	0.7531	(1.01×)
	20	0.8390	(1.02×)	0.8303	(1.01×)	0.7824	(1.05×)
	30	0.8401	(1.02×)	0.8349	(1.01×)	0.7968	(1.08×)
FedProx	10	0.8352	(1.02×)	0.7838	(0.95×)	0.7332	(0.99×)
	20	0.8334	(1.02×)	0.8077	(0.98×)	0.7743	(1.04×)
	30	0.8371	(1.02×)	0.8178	(1.00×)	0.7905	(1.08×)
FedDF	10	0.5860	(0.71×)	0.6960	(0.85×)	0.3164	(0.42×)
	20	0.6096	(0.72×)	0.6550	(0.88×)	0.6096	(0.81×)
	30	0.5909	(0.74×)	0.7054	(0.85×)	0.1939	(0.27×)
FedMeta	10	0.7852	(0.96×)	0.7606	(0.93×)	0.5905	(0.81×)
	20	0.7947	(0.97×)	0.7687	(0.94×)	0.6844	(0.92×)
	30	0.7894	(0.96×)	0.7786	(0.95×)	0.7096	(0.96×)
FedCSD	10	0.7262	(0.89×)	0.6913	(0.84×)	0.4442	(0.60×)
	20	0.7367	(0.90×)	0.7038	(0.86×)	0.5822	(0.78×)
	30	0.7411	(0.90×)	0.7085	(0.86×)	0.6057	(0.81×)
Elastic	10	0.7162	(0.87×)	0.6759	(0.82×)	0.5121	(0.69×)
	20	0.7740	(0.94×)	0.7246	(0.88×)	0.5333	(0.72×)
	30	0.7962	(0.97×)	0.7495	(0.91×)	0.5928	(0.80×)
FedMcon	10	**0.8366**	(1.02×)	**0.8220**	(1.00×)	**0.7854**	(1.07×)
	20	**0.8399**	(1.02×)	**0.8322**	(1.01×)	**0.8031**	(1.08×)
	30	**0.8403**	(1.02×)	**0.8360**	(1.02×)	**0.8101**	(1.09×)

The best results are in bold. The number of local epochs is fixed to 1. The value in parentheses indicates the speedup ratio

Table 4 Number of communication rounds to reach 60% test accuracy on the FEMNIST dataset

Method	$\|\mathcal{K}\|$	Number of communication rounds					
		$\alpha=1$		$\alpha=0.1$		$\alpha=0.01$	
FedAvg	10	13	(1.0×)	29	(1.0×)	84	(1.0×)
	20	11	(1.2×)	25	(1.2×)	81	(1.0×)
	30	10	(1.3×)	23	(1.3×)	76	(1.1×)
FedAvgM	10	20	(0.7×)	25	(1.2×)	85	(1.0×)
	20	17	(2.0×)	23	(1.3×)	81	(1.0×)
	30	15	(0.9×)	22	(1.3×)	79	(1.1×)
FedProx	10	13	(1.0×)	29	(1.0×)	82	(1.0×)
	20	11	(2.0×)	25	(1.2×)	79	(1.1×)
	30	10	(1.3×)	24	(1.2×)	74	(1.1×)
FedDF	10	49	(0.3×)	50	(0.6×)	–	–
	20	42	(0.3×)	46	(0.6×)	156	(0.5×)
	30	40	(0.3×)	40	(0.7×)	–	–
FedMeta	10	9	(1.4×)	20	(1.5×)	75	(1.1×)
	20	8	(1.6×)	18	(1.6×)	70	(1.2×)
	30	7	(1.9×)	16	(1.8×)	66	(1.3×)
FedCSD	10	25	(0.5×)	45	(0.6×)	120	(0.7×)
	20	22	(0.5×)	40	(0.6×)	110	(0.7×)
	30	20	(0.5×)	35	(0.7×)	95	(0.8×)
Elastic	10	30	(0.4×)	55	(0.5×)	150	(0.6×)
	20	28	(0.4×)	50	(0.5×)	140	(0.6×)
	30	25	(0.4×)	45	(0.5×)	130	(0.6×)
FedMcon	10	**7**	(1.9×)	**12**	(2.4×)	**18**	(4.7×)
	20	**5**	(2.6×)	**11**	(2.6×)	**16**	(5.3×)
	30	**4**	(3.3×)	**8**	(3.6×)	**12**	(7.0×)

The best results are in bold. The value in parentheses indicates the speedup ratio. "–" indicates that the target was not reached by the end of training

Table 5 Top-1 accuracies on the CIFAR-10 dataset

Method	Top-1 accuracy					
	$\alpha=1$		$\alpha=0.1$		$\alpha=0.01$	
FedAvg	0.7698	(1.00×)	0.6858	(1.00×)	0.6035	(1.00×)
FedAvgM	0.7589	(0.98×)	0.6803	(1.00×)	0.6330	(1.05×)
FedProx	0.7584	(0.98×)	0.6741	(0.99×)	0.6024	(1.00×)
FedDF	0.6584	(0.86×)	0.5906	(0.87×)	0.4866	(0.80×)
FedMeta	0.7636	(1.00×)	0.6667	(0.97×)	0.4686	(0.77×)
FedCSD	0.7245	(0.94×)	0.6524	(0.95×)	0.5832	(0.97×)
Elastic	0.7102	(0.92×)	0.6412	(0.93×)	0.5721	(0.95×)
FedMcon	**0.7765**	(1.01×)	**0.7061**	(1.05×)	**0.6824**	(1.13×)

The best results are in bold. The number of local epochs is fixed to 1. The value in parentheses indicates the speedup ratio

that FedMcon is a highly competitive FL algorithm, outperforming other methods in terms of both accuracy and speed, especially in high levels of data heterogeneity.

5 Discussions

5.1 Advantages of learning-based FedMcon

The FedMcon approach stands out for its exceptional versatility compared with other methods, thanks to its ability to adapt to various FL scenarios, configurations, and data distributions. This adaptability allows the control variate to adjust to the specific demands of the tasks. Additionally, factors such as the learning rate, the number of local epochs, and the number of clients can significantly impact the FL performance, particularly for images, texts, recommendations, or graphical data. FedMcon's learning-based framework allows for meta-learning on proxy data to effectively aggregate models, making it flexible and efficient for various FL tasks.

5.2 Difference of using proxy dataset

We compare FedMcon to FedDF and FedMeta, both of which use a proxy dataset to improve the FL performance. However, these approaches differ, with FedDF using ensemble distillation and FedMeta using meta updates. A key limitation of these methods is that they directly update model parameters

Table 6 Number of communication rounds to reach 60% test accuracy on the CIFAR-10 dataset

Method	Number of communication rounds					
	$\alpha = 1$		$\alpha = 0.1$		$\alpha = 0.01$	
FedAvg	31	(1.0×)	45	(1.0×)	61	(1.0×)
FedAvgM	34	(0.9×)	48	(0.6×)	73	(0.8×)
FedProx	35	(0.9×)	49	(0.6×)	149	(0.4×)
FedDF	58	(0.5×)	143	(0.2×)	174	(0.4×)
FedMeta	29	(1.1×)	70	(0.4×)	186	(0.3×)
FedCSD	42	(0.7×)	85	(0.5×)	165	(0.4×)
Elastic	45	(0.7×)	92	(0.5×)	172	(0.4×)
FedMcon	**10**	(3.0×)	**16**	(2.0×)	**21**	(3.0×)

The best results are in bold. The value in parentheses indicates the speedup ratio

using the proxy dataset, which has several drawbacks. First, direct parameter updates on proxy data can easily lead to overfitting, as the model may memorize specific patterns in this small dataset rather than learning generalizable features. Second, the effectiveness becomes highly sensitive to hyperparameter choices like learning rates and batch sizes, requiring careful tuning for each scenario. In contrast, FedMcon takes a fundamentally different approach by using the proxy dataset to learn an aggregative bias that indirectly guides the FL process. This indirect approach has several advantages: (1) Since the proxy data only influence the high-level aggregation strategy rather than directly affect model parameters, the risk of overfitting is significantly reduced; (2) The meta-controller can learn robust aggregation rules that generalize well across different client distributions; (3) The indirect guidance allows the model to maintain its focus on the actual training data while benefiting from the proxy dataset's task-relevant information. These characteristics make FedMcon inherently more resistant to overfitting while remaining hyperparameter-free, safer, robust, and efficient.

5.3 Limitations

Our proposed FedMcon uses a proxy dataset on the server, which may have limitations in some federated learning scenarios. However, it is practical for the server to have a task-oriented proxy dataset, especially for industrial recommender systems. Before FL training, the server should know the desired training task and it can hold a dataset that reveals the task. This dataset can validate the global model's accuracy and some data of it can be used as the proxy dataset in our method. The proxy dataset serves as a task-oriented guide to help the global model converge to the desired objective, regardless of clients' data distributions. In our work, we showcase the proxy dataset as a generalization and efficiency indicator, but it may also be useful for filtering out corrupted or malicious clients, achieving distributional robustness, domain adaptation, and more. Besides, we hold loose requirements on the scale of the proxy dataset, and it is validated in the experiments that the proxy dataset is small (1% of the training data).

6 Conclusions

We present FedMcon, a novel learning-based optimization framework that fundamentally advances federated learning through an adaptive meta-controller approach. Our work makes three key technical contribution: First, we introduce a learnable meta-controller that transforms the aggregation process from a static, rule-based procedure to a dynamic, context-aware system that automatically adapts to client behaviors and optimization landscapes. Second, our control variate mechanism provides an innovative solution for debiasing client updates by considering both immediate model changes and temporal patterns, significantly improving convergence stability. Third, our proxy dataset approach enables efficient meta-learning while maintaining privacy constraints, demonstrating strong practical viability. Our comprehensive experiments show that FedMcon consistently outperforms existing methods across various FL scenarios, achieving up to 19 times communication efficiency gains. Looking ahead, promising research directions include extending the framework to asynchronous FL settings, incorporating privacy-preserving mechanisms into the meta-controller architecture, and developing theoretical convergence guarantees for different data distribution scenarios. These future directions, combined with our current contributions, establish a strong foundation for advancing federated learning in real-world applications.

Contributors

Tao SHEN, Zexi LI, Ziyu ZHAO, Didi ZHU, Zheqi LV, and Kun KUANG designed the research. Shengyu ZHANG,

Chao WU, and Fei WU supervised the study. Tao SHEN, Zexi LI, Ziyu ZHAO, and Didi ZHU drafted the paper. Zheqi LV, Shengyu ZHANG, Chao WU, and Fei WU helped organize and refine the paper. All the authors revised and finalized the paper.

Conflict of interest

Fei WU is an executive associate editor-in-chief of *Frontiers of Information Technology & Electronic Engineering*, and he was not involved with the peer review process of this paper. All the authors declare that they have no conflict of interest.

Data availability

The data that support the findings of this study are available from the corresponding authors upon reasonable request.

References

Andrychowicz M, Denil M, Gomez S, et al., 2016. Learning to learn by gradient descent by gradient descent. https://doi.org/10.48550/arXiv.1606.04474

Bertinetto L, Henriques JF, Valmadre J, et al., 2016. Learning feed-forward one-shot learners. https://doi.org/10.48550/arXiv.1606.05233

Caldas S, Duddu SMK, Wu P, et al., 2019. LEAF: a benchmark for federated settings. https://doi.org/10.48550/arXiv.1812.01097

Chen DS, Hu J, Tan VJ, et al., 2023. Elastic aggregation for federated optimization. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition, p.12187-12197. https://doi.org/10.1109/CVPR52729.2023.01173

Chen F, Dong ZH, Li ZG, et al., 2019. Federated meta-learning for recommendation. https://doi.org/10.48550/arXiv.1802.07876

Fallah A, Mokhtari A, Ozdaglar A, 2020. Personalized federated learning with theoretical guarantees: a model-agnostic meta-learning approach. Proc 34th Int Conf on Neural Information Processing Systems, p.3557-3568.

Finn C, Abbeel P, Levine S, 2017. Model-agnostic meta-learning for fast adaptation of deep networks. https://doi.org/10.48550/arXiv.1703.03400

Haddad WM, Chellaboina V, 2008. Nonlinear Dynamical Systems and Control: a Lyapunov-Based Approach. Princeton University Press, Princeton, USA. https://doi.org/10.2307/j.ctvcm4hws

Harper FM, Konstan JA, 2015. The MovieLens datasets: history and context. *ACM Trans Interact Intell Syst*, 5(4):19. https://doi.org/10.1145/2827872

Hsu TMH, Qi H, Brown M, 2019. Measuring the effects of non-identical data distribution for federated visual classification. https://doi.org/10.48550/arXiv.1909.06335

Huang YT, Chu LY, Zhou ZR, et al., 2021. Personalized cross-silo federated learning on non-IID data. Proc 35th AAAI Conf on Artificial Intelligence, p.7865-7873. https://doi.org/10.1609/aaai.v35i9.16960

Jiang WW, Han HY, Zhang Y, et al., 2024. Federated split learning for sequential data in satellite–terrestrial integrated networks. *Inform Fusion*, 103:102141. https://doi.org/10.1016/j.inffus.2023.102141

Jiang YH, Konečný J, Rush K, et al., 2023. Improving federated learning personalization via model agnostic meta-learning. https://doi.org/10.48550/arXiv.1909.12488

Karimireddy SP, Jaggi M, Kale S, et al., 2021a. Breaking the centralized barrier for cross-device federated learning. Proc 35th Int Conf on Neural Information Processing Systems, p.28663-28676.

Karimireddy SP, Kale S, Mohri M, et al., 2021b. SCAFFOLD: stochastic controlled averaging for federated learning. https://doi.org/10.48550/arXiv.1910.06378

Krizhevsky A, Hinton G, 2009. Learning Multiple Layers of Features from Tiny Images. Toronto, ON, Canada.

LeCun Y, Bottou L, Bengio Y, et al., 1998. Gradient-based learning applied to document recognition. *Proc IEEE*, 86(11):2278-2324. https://doi.org/10.1109/5.726791

Li CL, Niu D, Jiang B, et al., 2021. Meta-HAR: federated representation learning for human activity recognition. Proc Web Conf, p.912-922. https://doi.org/10.1145/3442381.3450006

Li QB, Diao YQ, Chen Q, et al., 2021a. Federated learning on non-IID data silos: an experimental study. https://doi.org/10.48550/arXiv.2102.02079

Li QB, He BS, Song D, 2021b. Model-contrastive federated learning. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition, p.10708-10717. https://doi.org/10.1109/CVPR46437.2021.01057

Li T, Sahu AK, Zaheer M, et al., 2020. Federated optimization in heterogeneous networks. https://doi.org/10.48550/arXiv.1812.06127

Li XX, Jiang MR, Zhang XF, et al., 2021. FedBN: federated learning on non-IID features via local batch normalization. https://doi.org/10.48550/arXiv.2102.07623

Lin T, Kong LJ, Stich SU, et al., 2020. Ensemble distillation for robust model fusion in federated learning. Proc 34th Int Conf on Neural Information Processing Systems, p.2351-2363.

Lin YJ, Ren PJ, Chen ZM, et al., 2020. Meta matrix factorization for federated rating predictions. Proc 43rd Int ACM SIGIR Conf on Research and Development in Information Retrieval, p.981-990. https://doi.org/10.1145/3397271.3401081

McMahan HB, Moore E, Ramage D, et al., 2023. Communication-efficient learning of deep networks from decentralized data. https://doi.org/10.48550/arXiv.1602.05629

Muhammad K, Wang QQ, O'Reilly-Morgan D, et al., 2020. FedFast: going beyond average for faster training of federated recommender systems. Proc 26th ACM SIGKDD Int Conf on Knowledge Discovery and Data Mining, p.1234-1242. https://doi.org/10.1145/3394486.3403176

Nichol A, Achiam J, Schulman J, 2018. On first-order meta-learning algorithms. https://doi.org/10.48550/arXiv.1803.02999

Parra-Ullauri JM, Madhukumar H, Nicolaescu AC, et al., 2024. kubeFlower: a privacy-preserving framework for Kubernetes-based federated learning in cloud-edge environments. *Future Gener Comput Syst*, 157:558-572. https://doi.org/10.1016/j.future.2024.03.041

Pramling I, 1990. Learning to Learn: a Study of Swedish Preschool Children. Springer, New York, USA. https://doi.org/10.1007/978-1-4612-3318-3

Ravi S, Larochelle H, 2016. Optimization as a model for few-shot learning. 5th Int Conf on Learning Representations.

Reddi S, Charles Z, Zaheer M, et al., 2021. Adaptive federated optimization. https://doi.org/10.48550/arXiv.2003.00295

Shamsian A, Navon A, Fetaya E, et al., 2021. Personalized federated learning using hypernetworks. Proc 38th Int Conf on Machine Learning, p.9489-9502.

Shen T, Li ZX, Zhao ZY, et al., 2024. An adaptive aggregation method for federated learning via meta controller. Proc 6th ACM Int Conf on Multimedia in Asia Workshops, Article 20. https://doi.org/10.1145/3700410.3702124

Wang JY, Liu QH, Liang H, et al., 2020. Tackling the objective inconsistency problem in heterogeneous federated optimization. https://doi.org/10.48550/arXiv.2007.07481

Xu ZW, van Hasselt H, Silver D, 2018. Meta-gradient reinforcement learning. https://doi.org/10.48550/arXiv.1805.09801

Yan YL, Feng CM, Ye M, et al., 2023. Rethinking client drift in federated learning: a logit perspective. https://doi.org/10.48550/arXiv.2308.10162

Yao X, Huang TC, Zhang RX, et al., 2020. Federated learning with unbiased gradient aggregation and controllable meta updating. https://doi.org/10.48550/arXiv.1910.08234

Zhao Y, Li M, Lai LZ, et al., 2022. Federated learning with non-IID data. https://doi.org/10.48550/arXiv.1806.00582

Zhou GR, Zhu XQ, Song CR, et al., 2018. Deep interest network for click-through rate prediction. Proc 24th ACM SIGKDD Int Conf on Knowledge Discovery and Data Mining, p.1059-1068. https://doi.org/10.1145/3219819.3219823

Appendix: Hyperparameter settings

Table A1 Hyperparameter settings for different FL methods

Method	Hyperparameter	FEMNIST	CIFAR-10	MovieLens 1M
FedAvg	Local learning rate (η_l)	0.01	0.01	0.01
FedAvgM	Local learning rate (η_l)	0.01	0.01	0.01
	Global learning rate (η_g)	1	1	1
	Momentum (β_1)	0.9	0.9	0.9
FedProx	Local learning rate (η_l)	0.01	0.01	0.01
	Proximal term (μ)	1	1	0.01
FedDF	Local learning rate (η_l)	0.01	0.01	0.01
	Global learning rate (η_g)	0.0001	0.01	0.01
	Proxy ratio	0.01	0.01	0.01
	Server batch size	1000	20	20
	Number of server epochs	1	5	5
FedMeta	Local learning rate (η_l)	0.01	0.01	0.01
	Global learning rate (η_g)	0.0001	0.01	0.01
	Proxy ratio	0.01	0.01	0.01
	Server batch size	1000	20	20
	Number of server epochs	1	1	1
FedCSD	Local learning rate (η_l)	0.01	0.01	–
	Global learning rate (η_g)	1	1	–
Elastic	Local learning rate (η_l)	0.01	0.01	–
	Global learning rate (η_g)	1	1	–
FedMcon	Local learning rate (η_l)	0.01	0.01	0.01
	Global learning rate (η_g)	0.01	0.01	0.01
	Proxy ratio	0.01	0.01	0.01
	Server batch size	500	20	1000
	Number of server epochs	1	1	5