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, Ningbo 315048, China*
[3]*School of Public Affairs, Zhejiang University, Hangzhou 310058, China*
[4]*Academyof Social Governance, Zhejiang University, Hangzhou 310058, China*
E-mail: tao.shen, zexi.li, benzhao.styx, didi_zhu, zheqilv, sy_zhang, kunkuang, chao.wu, wufei@zju.edu.cn

Received June 20, 2024; Revision accepted Dec. 15, 2024; Crosschecked

Abstract: Federated learning (FL) emerged as a novel machine learning setting that enables collaboratively training deep models on decentralized clients with privacy constraints. In vanilla federated learning 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 presents a new aggregation method, FedMcon, in a framework of meta learning for FL. We introduce a learnable controller that trained on a small proxy dataset and served as the 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 in extremely non-i.i.d. data and it can simultaneously reach 8.5% generalization gain and 19 times communication speedup in one FL setting.

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

1 Introduction

In the field of machine learning, federated learning (FL) has been identified as a promising method for preserving privacy McMahan et al. (2017). 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 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., 2017), often referred to as the non-i.i.d. (non-independent and identically distributed) problem.

‡ Corresponding authors
* Project supported by the National Key Research and Development Project of China (No. 2021ZD0110505), the Zhejiang Provincial Key Research and Development Project (No. 2023C01043), the National Natural Science Foundation of China (No. 62402429), the Key Research and Development Program of Zhejiang Province (No. 2024C03270), ZJU Kunpeng&Ascend Center of Excellence, and the Ningbo Yongjiang Talent Introduction Programme (No. 2023A-397-G)

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 federated learning algorithm (FedAvg). To overcome the difficulties posed by non-i.i.d. 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. (2019) studied the objective inconsistency in cross-device FL, where biased client selection leads to updates in each round that are not consistent with the global objective. 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 a 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 a FL strategy might excel in one context but fail in another (Li et al., 2021c). 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 Bertinetto et al. (2016) and Andrychowicz et al. (2016), our goal is to explore the possibility of a 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, considered as a *learner*, that performs well on a proxy dataset. We propose an adaptive approach for debias 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 debias 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 Figure 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-i.i.d data, local epochs in cross-silo FL, and numbers of participating clients in each round of cross-device FL, on four different datasets. The results indicate the ability of our method to adapt to a wide range of FL scenarios, settings, and data distributions.

Contributions We summarize the primary contributions of this paper as follows:

- We present a novel learning-based optimization framework, FedMcon, that utilizes 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 direc-

tion of FL by incorporating a control variate for each client's model parameters.

- Empirical evidence shows that the FedMCONframework effectively outperforms other methods for non-i.i.d data in different FL scenarios, settings, and data distributions.

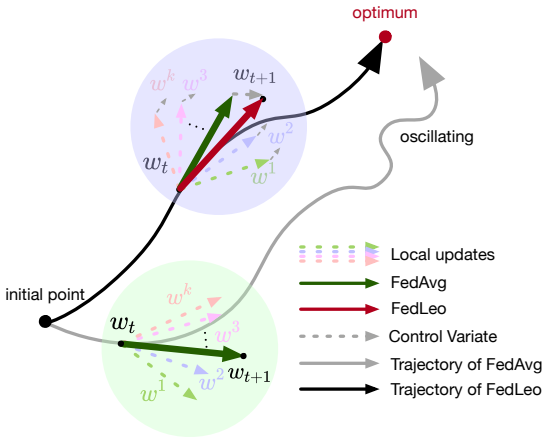

Fig. 1 Analysis of FedAvgand FedMconunder client drift and objective inconsistency. FedAvgmay be affected by both issues, leading to oscillation around the global optimum. On the contrary, the FedMconapproach overcomes these limitations by introducing a control variate for each client that guides the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

2 Related work

2.1 FL with non-i.i.d data.

The canonical FedAvgis 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-i.i.d and unbalanced data distribution that makes the training process harder. The difficulty of training non-i.i.d data is the accuracy reduction. Due to the non-i.i.dness, 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 proposes 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 generative adversarial network (GAN). Many theoretical studies are also conducted on FedAvgby focusing on convergence analysis and relaxing the assumptions in the non-i.i.d 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 lead to a reduction in accuracy as demonstrated by (Zhao et al., 2018), which attributes the phenomenon to weight divergence. To address this issue, Li et al. (2020) proposed the use of FedProx, a proximal term designed to mitigate the heterogeneity in the data. Moreover, Li et al. (2021b) presented comprehensive strategies for partitioning the non-i.i.d data, which typically pose challenges for FL. The work in (Wang et al., 2020) provided insight into the number of local update epochs and introduced a normalized averaging approach to address the objective inconsistency. Finally, Li et al. (2021) addressed the issue of feature shift non-i.i.d 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 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-i.i.d 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 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, 2012). 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). In (Andrychowicz et al., 2016), the authors proposed using deep neural networks to train a meta learner by means of an optimizer-optimizee setup. The update rule is learned rather than hand-designed, and the components are iteratively learned through gradient descent. Additionally, (Ravi and Larochelle, 2016) proposed using an 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 is 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 et al. (2019) highlighted the similarities between the MAML approach of fast, gradient-based adaptation to heterogeneous task distributions and the goal of personalization in FL. They observe that conventional federated averaging can be interpreted as a meta learning algorithm. Li et al. (2021a) proposed Meta-HAR to trains 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 et al. (2020b) 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 methods that use meta learning in FL include (Shamsian et al., 2021), which proposes learning a central hypernetwork to generate personalized models, and (Yao et al., 2019), which presents 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, it is stored on multiple devices. One common FL approach, FedAvg, aggregates local model updates through a weighted average approach, represented by $w^{global} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w^k$. However, FedAvgencounters a significant decrease in accuracy in the non-i.i.d scenario, where $\mathcal{P}(x,y) \sim \mathcal{P}_k(x,y) \neq \mathcal{P}_j$. This paper investigates the non-i.i.d problem in the context of FedAvgand presents a novel meta-learning-based framework to address it.

3.1 Typical FL setup

In Federated Averaging (FedAvg), the aim is to learn a single shared global model through decentralized data to minimize the global objective function $f(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w)$. This objective is the sum of the loss function over all private data $\mathcal{D}_{private}$, 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, FedAvgfirst copies the global model parameters $w_t^k \in \mathbb{R}^d$ to a set of candidate clients. Each candidate then performs a local update by optimizing their local objective through gradient descent for a specified number of epochs.

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

where $F_k(w_t^k)$ is the local objective of the k-th client, n_k is the number of local samples, η is the local learning rate, and $\nabla F_k \left(w_t^k\right) \in \mathbb{R}^d$ is the gradient vector. Following the local updates, clients transmit their local model parameters w_t^k to the server and then combine these parameters through weighted averag-

ing:

$$w_{t+1}^{global} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_t^k, \tag{2}$$

where w_{t+1}^{global} is the parameters 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, the expectation of $F_k(w)$ with respect to the data distribution of the k-th client, $\mathcal{P}_k$, can deviate from $f(w)$ when $\mathcal{P}_k$ is distinct from the data distributions of other clients and the overall distribution, $\mathcal{P}_j \neq \mathcal{P}_{overall}$. This deviation, referred to as aggregation bias, can result in slow convergence and suboptimal outcomes when implementing the FedAvgalgorithm, 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 (Karimireddy et al., 2021b; Yao et al., 2019; Huang et al., 2021; Karimireddy et al., 2021a). 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 optimization direction control by actively debiasing the aggregation process through a unique control variate mechanism. Furthermore, FedMconincorporates a novel feedback loop through proxy data, allowing the system to continuously refine its aggregation strategy based on 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 Bertinetto et al. (2016) and Andrychowicz 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 utilize meta learning, a technique known for its ability to enhance the efficiency of the learning process.

3.2.1 Introducing meta learning

Unlike the conventional machine learning approach, 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 testing 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 FedAvgwith a deep learning model, denoted by

$$\theta_{learner} = aggr(\Theta, \phi_{meta-controller}), \tag{3}$$

where $aggr$ is the aggregation function. The meta-learner model is parameterized by $\phi_{meta-controller}$ that takes the features from clients, represented by Θ 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, which is denoted by $\theta_{learner}$. The private

data on each client, represented by $\mathcal{D}^k_{private}$, functions as the support set in the meta-learning process, while the proxy data on the server, represented by $\mathcal{D}_{proxy}$, serves as the query set. To adapt to a specific FL task, we can train the aggregator over a query set $\mathcal{D}_{query}$ 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:

$$\begin{aligned} &\min_{\phi_{meta-controller}} \quad loss(\theta_{learner}, \mathcal{D}_{query}), \\ &\text{where} \quad \theta_{learner} = aggr(\theta_{learner}, \phi_{meta-controller}). \end{aligned} \tag{4}$$

This framework is depicted in the left part of Figure 2. The advantage of this framework is the flexibility to design a custom loss function based on the specific FL task, utilizing 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 Debias aggregation by feedback control

In this section, we apply the FedMCONframework 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 FedAvgis lack of control

The concept of control theory has inspired the perspective of FL being viewed as a dynamic system (Haddad and Chellaboina, 2011). In this perspective, the model parameters w_t are treated as the system states. The FedAvgapproach in FL operates as a feedforward process without any control. The presence of non-i.i.d data makes it uncertain if the aggregated model goes toward the intended update direction. The non-i.i.d data and the lack of control can result in unstable and potentially oscillating model updates. The dynamic Equations (1) and (2) of FL can be written as follows [1]:

$$w_{t+1} = g(w_t) = \frac{1}{K}\sum_{k=1}^{K} w_t^k = \frac{1}{K}\sum_{k=1}^{K}(w_t - \Delta w_t^k), \tag{5}$$

where Δw_t^k represents the change in the local update, obtained by minimizing $F_k(w_t^k)$ over several epochs, and $g(w_t)$ denotes the state transfer function of the model parameters w_t, which defines the trajectory of the model parameters. However, the presence of non-i.i.d data can result in the objective function $F_k(w_t^k)$ being a poor approximation to the global objective $f(w_t)$, 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 by adding a control variate u_t^k for each client. The state transfer function $g_c(w_t)$ incorporating the control variate is formulated as follows:

$$w_{t+1} = g_c(w_t) = \frac{1}{K}\sum_{k=1}^{K}(w_t - \Delta w_t^k(1 - u_t^k)), \tag{6}$$

where the control variate u_t^k is introduced element-wise to each Δw_t^k. The challenge lies in determining an effective control variate that directs the model parameters w_t towards the global optimum. The proposed FedMConframework addresses this issue by utilizing a learning-based meta-learning technique to obtain feedback from the evaluation of proxy data. As the proxy data are assumed to have a similar distribution as the overall dataset, the control variates $(1 - u_t^k)$ derived from the proxy data can potentially calibrate the model update towards the intended direction.

3.3.2 Learning to aggregate with feedback.

In order to achieve this, the control variate u_t^k is defined as a function of w_t, Δw_t^k, and a set of parameters ϕ: $u_t^k = h(w_t, \Delta w_t^k, \phi)$. 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 w_t and the local updates Δw_t^k from clients. The adaptive nature of our controller enables several key capabilities. First, it can dynamically adjust to varying degrees of non-i.i.d. data distributions, ensuring robust performance across different data scenarios. Second, it learns optimal

[1] In fact, the number of samples of clients n_k is usually unknown to the server, thus we set $\frac{n_k}{n}$ as $\frac{1}{K}$.

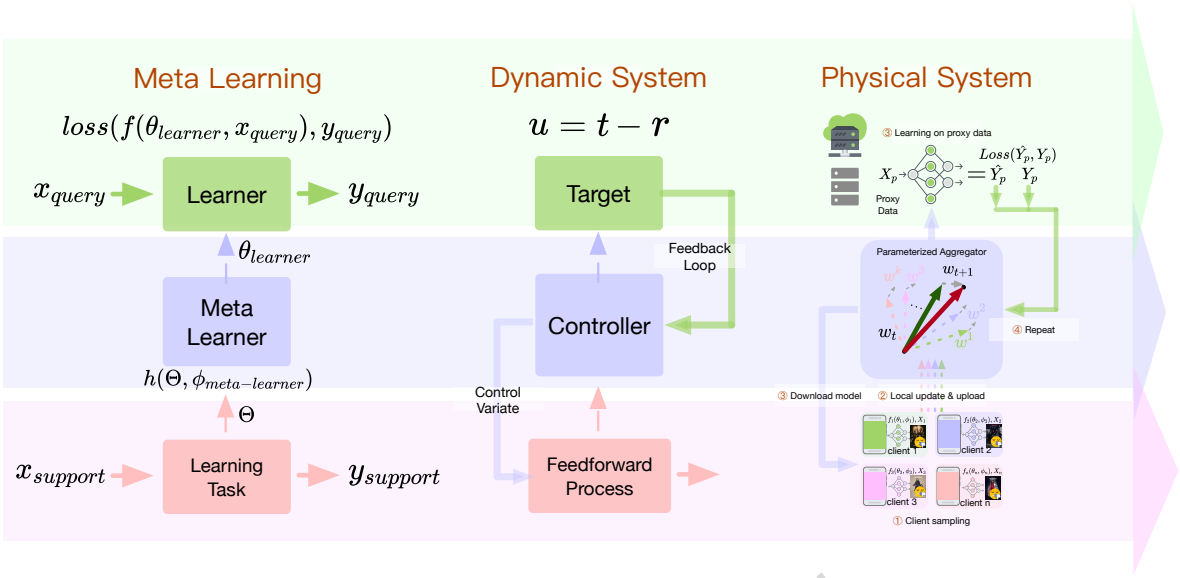

Fig. 2 Relations between meta-learning, dynamic system and FedLeo. Left: The structure of the general learning-based framework that incorporates meta learning. The private data acts as the support set, while the proxy data acts as the query set. The role of the meta-learner is to expedite the learning 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-i.i.d. nature of the data. The proxy data serves as a 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 FedLeoincludes 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.

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 variates u_t^k can help guide the model updates towards the intended direction, which is illustrated in the middle part of Figure 2. To accomplish this, we integrate the controller and the averaging operator into the aggregator, resulting in an updated aggregation function:

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

where $\Delta\mathcal{W}_t = \Delta w_t^k$ 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 in Equation (3) is formulated as follows:

$$w_{t+1} = aggr(w_t, \Delta\mathcal{W}_t, \phi). \tag{8}$$

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 serves 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 as shown in Equation (4) as follows:

$$\min_{\phi} f(w_{proxy}, \mathcal{D}_{proxy}), \quad \text{where} \quad w_{proxy} = aggr(w_t, \Delta\mathcal{W}_t.\phi). \tag{9}$$

The parameters ϕ 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 Figure 2.

3.3.3 Structure of aggregator model

As outlined in Equation (8), the aggregator takes model updates as input and produces the proxy model as output. The aggregator has two modules, ϕ_g and ϕ_c, one for the state of the global model w_t, and the other for the model updates from clients w_t^k. For model parameters w with dimension d, the ideal dimension of ϕ_g (or ϕ_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 ($\phi \in \mathbb{R}^{d\times d}$ if $w \in \mathbb{R}^d$). To address this issue, a bottleneck architecture is employed, mapping the parameters to a low-dimensional space (e.g. p) and restoring the output to the original dimension. For example, the input layers of ϕ_g and ϕ_c are $\phi_{g_{in}}, \phi_{c_{in}} \in \mathcal{R}^{d\times p}$, mapping the model parameters and updates into a hidden space, respectively. The hidden spaces are concatenated and followed by an output layer $\phi_{out} \in \mathcal{R}^{2p\times d}$. As a result, the dimension of $\phi = \phi_g, \phi_c$ is $4 \times d \times p \times m$ for $w \in \mathcal{R}^d$. The control variate 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 ϕ of the aggregator, the setting $p = \log_2(d)$ is applied. The control variate can then be formulated as follows:

$$u_t^k = (w_t, \Delta w_t^k, \phi) = \phi_{out}(\phi_{g_{in}}(w_t), \phi_{c_{in}}(\Delta w_t^k)). \tag{10}$$

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) at the meantime, the server optimizes the aggregator parameters ϕ 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 Equation (8); and 5) this process iterates until convergence or reaching the maximum number of communication rounds. This whole pipeline is detailed in Algorithm 1.

Algorithm 1 FedMcon: A meta-learning based federated learning framework with controlled model aggregation. The algorithm requires: global model w_t, proxy dataset $\mathcal{D}_{proxy}$ on server, clients indexed by k with local models w_t^k and private datasets $\mathcal{D}_{private}^k$, local learning rate η_l, number of local epochs E_l, number of epochs for training aggregator E_g, and total number of communication rounds T.

Server executes:

initialize the global model w_0 and aggregator parameters ϕ
for each round $t = 0, 1, 2, \ldots, T$ **do**
  randomly sample a set of candidate clients $\mathcal{K}$ with sampling rate C
  Execute at the meantime:
  a) $\Delta\mathcal{W}_t \leftarrow$ ClientsUpdate(w_t, $\mathcal{K}$)
  b) optimize aggregator parameters ϕ using proxy data $\mathcal{D}_{proxy}$ for E_g epochs
  aggregate updates using controlled aggregation: $w_{t+1} = aggr(w_t, \Delta\mathcal{W}_t, \phi)$
  broadcast w_{t+1} to selected clients
end for

ClientUpdate:

for each client $k \in \mathcal{K}$ **in parallel do**
  receive global model: $w_t^k \leftarrow w_t$
  initialize local model with global parameters
  for each local epoch $e = 1, 2, \ldots, E_l$ **do**
    for each batch $b \in \mathcal{D}_{private}^k$ **do**
      local update: $w_t^k \leftarrow w_t^k - \eta_l \nabla F_k\left(w_t^k, b\right)$
    end for
  end for
  compute model update: $\Delta w_t^k \leftarrow w_t - w_t^k$
  upload Δw_t^k to server
end for
return $\Delta\mathcal{W}_t = \{\Delta w_t^k\}_{k\in\mathcal{K}}$

4 Experiments

4.1 Setup

4.1.1 Datasets and models

In this study, the performance of FedMCONis evaluated on various state-of-the-art FL methods on both recommendation and computer vision datasets. The recommendation dataset used is the MovieLens 1M dataset[2] Harper and Konstan (2015), containing 1,000,209 ratings provided by 6,040 unidentifiable users on 3,706 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, their latest interaction is held out as the testset and the rest of the data is used as the trainset. The user feedback is binarized and negative instances are sampled 4:1 for training and 99:1 for testing, with the number of positive instances used as the baseline. The computer vision dataset utilized is the FEMNIST dataset[3] Caldas et al. (2018), which consists of 62 classes of 28x28 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[4] Krizhevsky et al. (2009), which consists of 10 classes of 32x32 pixel images. The dataset includes 50,000 training examples and 10,000 test samples. It is performed using the ResNet18 (replacing batch norm with group norm (Reddi et al., 2020)). 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-i.i.d 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-i.i.dness Lin et al. (2020a); Yao et al. (2019). 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 $E_l = 1$ epoch locally. For the recommendation task, the Adam optimizer, and for the computer vision task, SGD is used as the local optimizer, both with a local learning rate of $\eta_l = 0.01$. For methods with server-side optimization, the Adam optimizer is used with a global learning rate of $\eta_g = 0.001$. The detailed hyperparameter settings for all methods are provided in Table 7.

4.1.3 Baselines

The proposed FedMCONframework is compared with several state-of-the-art FL methods, including: (1) Vanilla FL method FedAvg(McMahan et al., 2017); (2) Client-side FL method FedProx(Li 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 et al., 2020a); (5) Server-side federated meta learning method with proxy data FedMeta(Yao et al., 2019) and another two FL methods for vision tasks FedCSD(Yan et al., 2023) and Elastic(Chen et al., 2023). It is important to note that FedMCONis orthogonal to other meta-learning-based FL methods which are designed for model initialization (Chen et al., 2018), or for personalization (Shamsian et al., 2021; Fallah et al., 2020), as a result, they are not included in comparisons.

4.1.4 Evaluation metric

In the experiments, for the CTR (Click-Through Rate) task, the model performance is evaluated using the following metrics: area under curve (AUC), Hit Ratio (HR) and Normalized Discounted Cumulative

[2]https://grouplens.org/datasets/movielens/ (license)
[3]https://github.com/TalwalkarLab/leaf/tree/master/data/femnist
[4]https://www.cs.toronto.edu/ kriz/cifar.html

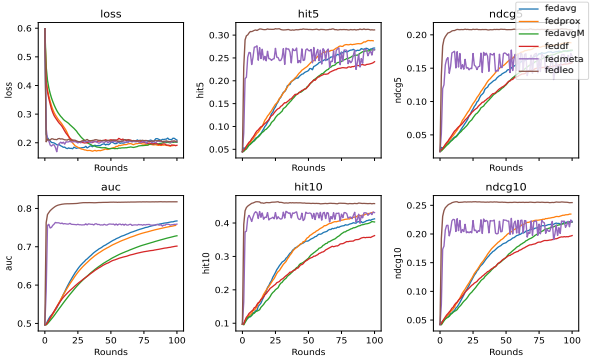

Fig. 3 The learning curves of FedMconand baselines on the MovieLens 1M dataset

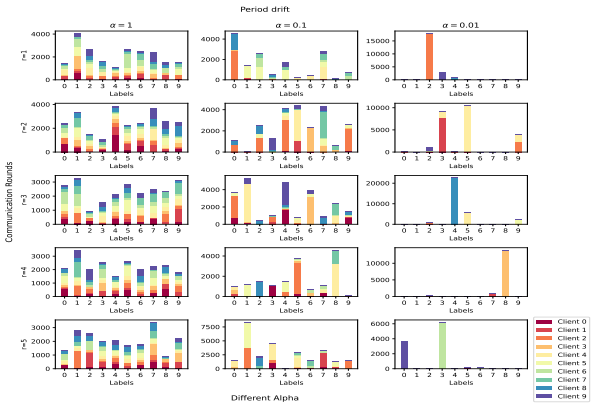

Fig. 4 Visualizing objective inconsistency on FEMNIST

Gain (NDCG).

$$\text{AUC} = \frac{\sum_{x_0 \in D_T} \sum_{x_1 \in D_F} \mathbf{1}\left[f\left(x_1\right) < f\left(x_0\right)\right]}{\left|D_T\right| \left|D_F\right|},$$

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

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

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 testing 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 FedMconframework.

4.2 Analysis

4.2.1 The performance on MovieLens

We utilized the inherently non-i.i.d MovieLens 1M dataset to evaluate FedMconfor ***real-world industrial applications***. The CTR task samples are particularly non-i.i.d, 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 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 communication rounds are a measure of the efficiency of the FL algorithms. The results show that FedMcon is consistently faster than the other algorithms, requiring fewer communication rounds to reach the desired level of performance. As shown in Figure 3, for methods that use proxy datasets, FedDFand FedMetaperformed lower 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, FedMetamay suffer from overfitting and oscillations on the proxy data, although the number of communication rounds for FedMcon is the same as for FedMeta, the fastest among the other algorithms, indicating that FedMcon is a highly efficient FL algorithm. In contrast, FedMconoutperforms other methods because it is capable of handling objective inconsistencies, where different objectives may arise at different communication rounds, even around the global optimum. Furthermore, FedMconexhibits fast and steady convergence and reaches a better optimum compared to other FL methods.

Table 1 Metrics on MovieLens 1M. The number of local epochs is fixed to 1

	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**

Table 2 Communication rounds to reach 90% of averaged metrics on MovieLens

	AUC		HR		NDCG	
FedAvg	37	(1.0×)	63	(1.0×)	71	(1.0×)
FedAvgM	48	(0.8×)	75	(0.8×)	77	(0.9×)
FedProx	38	(1×)	57	(1.1×)	51	(1.4×)
FedDF	49	(0.7×)	–	-	184	(1.0×)
FedMeta	2	(19.0×)	2	(32.0×)	3	(24.0×)
FedMcon	**2**	(19.0×)	**2**	(32.0×)	**2**	(36.0×)

4.2.2 Visualizing objective inconsistency

Our experiments are performed on the FEMNIST dataset, which is designed to provide non-i.i.d data. To quantify the degree of non-i.i.dness, we set the Dirichlet hyperparameter α to 1, 0.1, and 0.01. Figure 4 presents the results of the non-i.i.dness 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-i.i.dness 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-i.i.dness of the data increases, the challenge of training a adaptive and accurate model becomes more pronounced.

Table 3 Top-1 accuracies on FEMNIST with varying the number of sampled clients The number of local epochs is fixed to 1

	$\|\mathcal{K}\|$	$\alpha=1$		$\alpha=0.1$		$\alpha=0.01$	
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×)

4.2.3 The 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, 0.1, and 0.01. The number of clients in FEMNIST is 3400. The Top-1 accuracy is 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 (FedAvgwith $|\mathcal{K}| = 10, \alpha = 1/0.1/0.01$) for each method. Table 4 presents the communication round to reach 60% test accuracy for each method. Under different levels of non-i.i.dness and with varying numbers of participating clients in each round, the results reveal that FedMconconsistently outperforms the compared FL methods. Despite the increasing degree of non-i.i.dness, FedMcondemonstrates 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-i.i.d conditions.

Table 4 Communication rounds to reach 60% test accuracy on FEMNIST.

	$\|\mathcal{K}\|$	$\alpha=1$		$\alpha=0.1$		$\alpha=0.01$	
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×)

4.2.4 The performance on CIFAR-10

We compare the performance of several FL algorithms on the CIFAR-10 dataset to stimulate the ***cross-silo*** settings. For CIFAR-10, the number of clients is 10 and full participation of clients is conducted 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 FedMconconsistently outperforms the other algorithms, and it reaches the largest performance gain at $\alpha = 0.01$ (the most non-i.i.d setting). In Table 6, FedMconis shown to be the fastest algorithm, requiring significantly fewer 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 that FedMconis a highly competitive FL algorithm, outperforming other methods in terms of both accuracy and speed, especially in high levels of data heterogeneity.

Table 5 Top-1 accuracies on CIFAR-10. The number of local epochs is 1.

	$\alpha=1$		$\alpha=0.1$		$\alpha=0.01$	
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×)

Table 6 Communication rounds to reach 60% test accuracy on CIFAR-10.

	$\alpha=1$		$\alpha=0.1$		$\alpha=0.01$	
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×)

5 Discussions

5.1 Advatages of learning-based FedMcon

The FedMconapproach 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, local epochs, number of clients, and models can significantly impact 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 FedMconto FedDFand FedMeta, both of which also use a proxy dataset to improve FL performance. However, their approaches differ, with FedDFutilizing ensemble distillation and FedMetausing meta updates. A key limitation of these methods is that they directly update model parameters 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, FedMCONtakes 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 influences the high-level aggregation strategy rather than directly affecting 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 FedMCONinherently more resistant to overfitting while remaining hyperparameter-free, safer, more robust, and more efficient.

5.3 Limitations

Our proposed FedMCONuses 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 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, it is validated in the experiments that the proxy dataset is small (1% of the training data).

6 Conclusion

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 contributions: 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 FedMCONconsistently outperforms existing methods across various FL scenarios, achieving up to 8.5% accuracy improvements and 19× 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, Kun KUANG designed the research. Shengyu ZHANG, Chao WU, and Fei WU supervised the study. Tao SHEN, Zexi LI, Ziyu ZHAO, Didi ZHU drafted the paper. Zheqi LV, Shengyu ZHANG, Chao WU, and Fei WU helped organize and refine the paper. All authors revised and finalized the paper for submission.

Conflict of interest

All the authors declare that they have no conflict of interest.

References

Andrychowicz M, Denil M, Gomez S, et al., 2016. Learning to learn by gradient descent by gradient descent. *arXiv:160604474 [cs]*, .

Bertinetto L, Henriques JF, Valmadre J, et al., 2016. Learn-

ing feed-forward one-shot learners. *arXiv:160605233 [cs]*, .

Caldas S, Duddu SMK, Wu P, et al., 2018. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:181201097*, .

Chen D, Hu J, Tan VJ, et al., 2023. Elastic aggregation for federated optimization. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, p.12187-12197.

Chen F, Dong Z, Li Z, et al., 2018. Federated Meta-Learning for Recommendation. *CoRR*, abs/1802.07876.

Fallah A, Mokhtari A, Ozdaglar AE, 2020. Personalized Federated Learning with Theoretical Guarantees: A Model-Agnostic Meta-Learning Approach. Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, Virtual.

Finn C, Abbeel P, Levine S, 2017. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *arXiv:170303400 [cs]*, .

, 2011. Nonlinear dynamical systems and control.

Harper FM, Konstan JA, 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1-19.

Hsu TMH, Qi H, Brown M, 2019. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:190906335*, .

Huang Y, Chu L, Zhou Z, et al., 2021. Personalized cross-silo federated learning on non-i.i.d data. Proceedings of the AAAI Conference on Artificial Intelligence, 35:7865-7873.

Jiang W, Han H, Zhang Y, et al., 2024. Federated split learning for sequential data in satellite–terrestrial integrated networks. *Information Fusion*, 103:102141.

Jiang Y, Konečný J, Rush K, et al., 2019. Improving Federated Learning Personalization via Model Agnostic Meta Learning. *CoRR*, abs/1909.12488.

Karimireddy SP, Jaggi M, Kale S, et al., 2021a. Breaking the centralized barrier for cross-device federated learning. *Advances in Neural Information Processing Systems*, 34:28663-28676.

Karimireddy SP, Kale S, Mohri M, et al., 2021b. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning. *arXiv:191006378 [cs, math, stat]*, .

Krizhevsky A, Hinton G, et al., 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. *Proceedings of the IEEE*, 86(11):2278-2324.

Li C, Niu D, Jiang B, et al., 2021a. Meta-HAR: Federated Representation Learning for Human Activity Recognition. WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021, p.912-922.
https://doi.org/10.1145/3442381.3450006

Li Q, Diao Y, Chen Q, et al., 2021b. Federated learning on non-i.i.d data silos: An experimental study. *arXiv preprint arXiv:210202079*, .

Li Q, He B, Song D, 2021c. Model-contrastive federated learning. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, p.10713-10722.

Li T, Sahu AK, Zaheer M, et al., 2020. Federated Optimization in Heterogeneous Networks. *arXiv:181206127 [cs, stat]*, .

Li X, Jiang M, Zhang X, et al., 2021. Fedbn: Federated learning on non-i.i.d features via local batch normalization. *arXiv preprint arXiv:210207623*, .

Lin T, Kong L, Stich SU, et al., 2020a. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33:2351-2363.

Lin Y, Ren P, Chen Z, et al., 2020b. Meta Matrix Factorization for Federated Rating Predictions. Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020, p.981-990.
https://doi.org/10.1145/3397271.3401081

McMahan HB, Moore E, Ramage D, et al., 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. *arXiv:160205629 [cs]*, .

Muhammad K, Wang Q, O'Reilly-Morgan D, et al., 2020. Fedfast: Going beyond average for faster training of federated recommender systems. Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, p.1234-1242.

Nichol A, Achiam J, Schulman J, 2018. On First-Order Meta-Learning Algorithms. *arXiv:180302999 [cs]*, .

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 Generation Computer Systems*, 157:558-572.

Pramling I, 2012. Learning to learn: A study of Swedish preschool children. Springer Science & Business Media.

Ravi S, Larochelle H, 2016. Optimization as a model for few-shot learning. 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings.

Reddi S, Charles Z, Zaheer M, et al., 2020. Adaptive federated optimization. *arXiv preprint arXiv:200300295*, .

Shamsian A, Navon A, Fetaya E, et al., 2021. Personalized federated learning using hypernetworks. International Conference on Machine Learning, p.9489-9502.

Wang J, Liu Q, Liang H, et al., 2020. Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization. *arXiv:200707481 [cs, stat]*, .

Xu Z, van Hasselt H, Silver D, 2018. Meta-Gradient Reinforcement Learning. *arXiv:180509801 [cs, stat]*, .

Yan Y, Feng CM, Ye M, et al., 2023. Rethinking client drift in federated learning: A logit perspective. *arXiv preprint arXiv:230810162*, .

Yao X, Huang T, Zhang RX, et al., 2019. Federated Learning with Unbiased Gradient Aggregation and Controllable Meta Updating. *CoRR*, abs/1910.08234.

Zhao Y, Li M, Lai L, et al., 2018. Federated Learning with Non-IID Data. *arXiv:180600582 [cs, stat]*, .

Zhou G, Zhu X, Song C, et al., 2018. Deep interest network for click-through rate prediction. Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, p.1059-1068.

Table 7 Hyperparameter settings for different FL methods.

Method	Hyperparameters	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
	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
	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
	Server epochs	1	1	5