MSCFL: Model Structure-Aware Clustered Federated Learning for System Heterogeneity and Data Drift

Yang Xu[1], Xiaowei Wu[1], Zifeng Xu[1], Cheng Zhang[1*], Ju Ren[2], Yaoxue Zhang[2]

[1]College of Cyber Science and Technology, Hunan University, Changsha, China
[2]Department of Computer Science and Technology, BNRist, Tsinghua University, Beijing, China
{xuyangcs, wxw68, xuzifeng, zhangchengcs}@hnu.edu.cn, {renju, zhangyx}@tsinghua.edu.cn

Abstract

Federated Learning (FL) faces significant challenges arising from both data and system heterogeneity. While Clustered Federated Learning (CFL) mitigates data heterogeneity by grouping clients with similar data distributions, it remains vulnerable to system heterogeneity, which can slow convergence due to performance disparities among clients. Moreover, data drift may degrade clustering accuracy and training efficiency over time. In this work, we propose a Model Structure-aware Clustered Federated Learning (MSCFL) framework that simultaneously addresses the issues of data heterogeneity, system heterogeneity, and data drift. MSCFL incorporates model pruning (MP) into the CFL framework to enhance training efficiency under system heterogeneity. To enable this integration, we address the key challenge of performing effective clustering based on heterogeneous, pruned local models with varying structures. To this end, we design a model structure-based similarity computation algorithm to integrate CFL with MP. To effectively address data drift, we propose a dynamic cluster migration strategy that efficiently monitors model structures via Hamming Distance and triggers re-clustering only when necessary. Extensive experimental results show that MSCFL improves the accuracy and convergence speed of cluster models, outperforming traditional CFL in various settings.

Code — https://github.com/TaiyakiOffical/MSCFL

Introduction

Federated learning (FL) is a privacy-preserving distributed machine learning technique (McMahan et al. 2017; Nguyen et al. 2022; Bao et al. 2025; Liu et al. 2025). In the vanilla FL setup, a central server coordinates the collaboration of multiple clients to train a machine learning model, with each client contributing its model parameters rather than private data. However, a significant challenge in FL is handling the heterogeneous data from different clients. The non-independent and identically distributed (Non-IID) data on each client can cause the optimization paths of their local models to diverge. Directly aggregating these local models may result in the global model failing to converge (Li et al. 2022). Clustered Federated Learning (CFL) (Li, Chen, and Teng 2024; Guo, Tang, and Lin 2025; Zhang et al. 2025) addresses the aforementioned issue by grouping clients based on the similarity of their data distributions and training a personalized cluster model for each cluster (Briggs, Fan, and Andras 2020; Sattler, Müller, and Samek 2021; Long et al. 2023). CFL not only reduces the variance in the global model training process, but also produces personalized models that are better tailored to the data distributions of clients within a cluster.

Despite the advantages of CFL in addressing data heterogeneity, it still faces challenges in coping with *system heterogeneity*, which arises from the diverse computational and communication capabilities of participating clients (Zhou et al. 2024b; Zhang et al. 2024; Liu et al. 2024; Cheng et al. 2025). This issue primarily stems from CFL's synchronous training paradigm, where the server must wait for updates from all clients before proceeding with clustering and aggregation. Consequently, clients with lower performance can delay global updates, leading to increased training latency. Recent studies in FL (Zhou et al. 2024a; Wang et al. 2024) have employed model pruning (MP) techniques (Cheng, Zhang, and Shi 2024; Kim and Yoo 2025) to generate adaptive smaller local models for resource-constrained clients. This approach aims to balance training overhead across heterogeneous devices and mitigate round-level latency. However, the model heterogeneity introduced by MP fundamentally disrupts the clustering mechanism in CFL. Most existing CFL frameworks rely on measuring similarity between local models to group clients into clusters. When local models differ in structure due to pruning, such similarity metrics become invalid, making it impossible to perform meaningful clustering (Long et al. 2023). As a result, CFL fails to properly identify clients with similar data distributions, causing the framework to degenerate into a conventional single-model FL, thereby losing its ability to learn personalized models (Ma et al. 2023).

Furthermore, similar to other FL paradigms, CFL also suffers from *data drift* (Duan et al. 2022), where the distribution of features or labels within clients' private datasets changes over time, leading to degraded model performance. Although several approaches have been proposed to mitigate data drift (Casado et al. 2022; Panchal et al. 2023), integrating them into CFL remains challenging. Most existing CFL frameworks assume relatively stable data distribu-

*Cheng Zhang is the corresponding author.

tions, where each client's cluster identity is consistent across rounds. However, data drift can alter a client's underlying data characteristics, effectively changing its cluster affiliation. If such clients remain in their original clusters, the corresponding cluster models may converge slowly or even fail to capture the new data distribution, resulting in suboptimal accuracy. To handle this issue, some CFL methods (Jothimurugesan et al. 2023; Ghosh et al. 2022) perform re-clustering strategy in every communication round to adaptively update client groups. Nevertheless, this process introduces substantial computational and communication overhead. Although prior work (Duan et al. 2022) has explored efficient client migration mechanisms for CFL, these techniques mainly target label drift and remain ineffective in addressing feature drift, where feature distributions evolve over time.

In this paper, we propose an efficient Model Structure-based Clustered Federated Learning (MSCFL) framework, which is compatible with a wide range of MP (Li et al. 2017; Liu et al. 2017; Vahidian, Morafah, and Lin 2021) and clustering algorithms (Briggs, Fan, and Andras 2020; Sattler, Müller, and Samek 2021; Long et al. 2023; Duan et al. 2022; Liu et al. 2024). MSCFL is designed to address the key challenges of data heterogeneity, system heterogeneity, and data drift in FL. Specifically, MSCFL leverages CFL and MP to jointly mitigate data and system heterogeneity. To address the incompatibility issue that arises when integrating MP into CFL, we propose a model structure-based similarity computation algorithm that measures client similarity based on the neural structures retained or pruned with higher priority during MP. These structures are considered to capture the most representative data characteristics of each client, thereby enabling accurate and efficient clustering even under model heterogeneity. To further address data drift, we introduce a dynamic cluster migration strategy that re-clusters clients adaptively based on changes in their model structures. The central server monitors model structure consistency by calculating the Hamming distance among client models in each round. When a significant deviation is detected, clients are reassigned to appropriate clusters to maintain alignment with their evolving data distributions. Our main contributions can be summarized as follows:

- We propose MSCFL, an efficient model structure-based clustered FL framework that ensures compatibility between MP and CFL, while addressing the challenges of data heterogeneity, system heterogeneity and data drift.
- We design a model structure-based similarity computation algorithm that effectively resolves the issue of inaccurate clustering caused by model heterogeneity.
- We propose a cluster migration strategy that detects data drift by leveraging the Hamming Distance of successive masks, which then rectifies the cluster results to achieve better model performance.
- Experiments on four benchmark datasets demonstrate that MSCFL achieves higher model accuracy while assigning clients to clusters more quickly and accurately than traditional CFL methods.

Related Work

Clustered Federated Learning

CFL is a collaborative machine learning method that groups clients into separate clusters to train personalized cluster models, solving the issue of data heterogeneity in FL. The clustering algorithm of CFL consists of two main components: similarity computation algorithm and classification algorithm. FL+HC (Briggs, Fan, and Andras 2020) utilizes a hierarchical classification algorithm to categorize clients according to the L_2 distance of model updates, whereas One-Shot (Ghosh et al. 2019) opts for the K-Means clustering algorithm for client separation. FeSEM (Long et al. 2023) introduces the stochastic expectation maximization classification algorithm to cluster clients based on the L_2 distance of local models, achieving better clustering results. FlexCFL (Duan et al. 2022) is pioneering in addressing the issue of data drift by leveraging wasserstein distance of the training data of all clients. In addition, FlexCFL utilizes the K-Means++ clustering algorithm to cluster clients based on euclidean distance of decomposed cosine similarity of model updates. In IFCA (Ghosh et al. 2022), clients select the one with the lowest loss on their dataset from all the cluster models in each round and train the local model on it. In general, the aforementioned studies seek to infer data distribution similarities indirectly through the training process. In contrast, PACFL (Vahidian et al. 2023) extracts distinctive feature vectors from client's datasets that can be utilized for clustering purposes. HCFL+ (Guo, Tang, and Lin 2025) addresses key challenges in current methods, such as inconsistent intra-client clustering weights and lack of adaptive clustering for soft clustering paradigms. Although CFL has strengths in managing data heterogeneity, it encounters difficulties when addressing system heterogeneity.

Model Pruning

MP is an effective strategy to tackle system heterogeneity in FL by strategically removing the least significant model structures, which in turn speeds up both the training and communication phases. In existing studies of MP, they employ different methods to evaluate the significance of a unit within the model such as the extent of neuron activation (Wang et al. 2024), L_1-norm of filters (Li et al. 2017), the scaling factor of Batch Normalization (BN) layer (Liu et al. 2017), the group lasso value (Li et al. 2021), the distance from the geometric median of filters of same layer (He et al. 2019), the rank of feature map (Lin et al. 2020), or the importance indicators introduced in training (Lin et al. 2019). Furthermore, some studies focus on determining the optimal model structures to retain in a communication round to enhance the performance of the global model. For instance, FjORD (Horváth et al. 2021) and HeteroFL (Diao, Ding, and Tarokh 2021) create static submodels extracted from the global model. In contrast, Federated Dropout (Caldas et al. 2018) opts for a random submodel extraction approach, while FedRolex (Alam et al. 2022) implements a rolling mechanism for submodel extraction. SVS (Kim and Yoo 2025) minimizes disparities between singular values of pruned weights, improving fine-tuning performance.

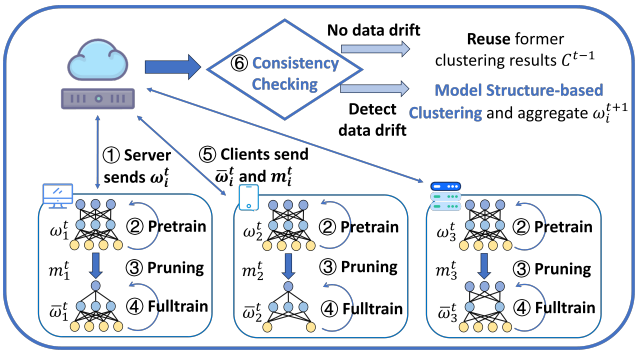


Figure 1: System overview of MSCFL.

In this paper, we combine CFL and MP to tackle the issue of data heterogeneity and system heterogeneity. Nevertheless, the majority of the previously mentioned MP are incompatible with traditional CFL. This incompatibility can cause CFL to malfunction. Moreover, CFL also struggles to cope with data drift, which may result in performance degradation. As a result, we propose a similarity computation method to achieve such compatibility, combined with a cluster migration strategy that can be seamlessly integrated into these existing methods with just minimal adjustments to overcome the aforementioned problems.

Method

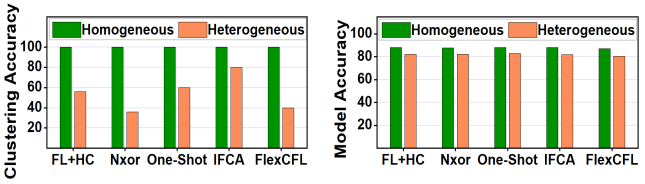
We begin with the system model, followed by the motivation that discusses two trivial and infeasible solutions. Next, we present the details of our proposed approach.

System Model

In our work, we assume that there are N clients, each with different computational and communication capabilities. The data distribution across the clients is Non-IID and evolves over time. The clients can be grouped into K distinct clusters, a classification determined by their underlying data distributions, which are reflected in their pruned model structures. The workflow is shown in Figure 1. A central server assigns clients to clusters and maintains a dedicated cluster model for each. Before local training, each client prunes the complete cluster model using MP according to its capability. After training, clients upload their locally updated models to the server. The server then checks for data drift and reassigns clients to clusters if necessary. The core problem addressed in this work is how to train FL models under the conditions of client data heterogeneity, system heterogeneity, and data drift.

Motivation

Here, we first identify the incompatibility between CFL and MP. Next, we propose two intuitive methods to address this issue. Although these methods partially mitigate the incompatibility, they still fail to produce accurate clustering results. Nevertheless, they provide a foundation and groundwork for our subsequent core algorithm.



(a) Clustering accuracy (b) Cluster model accuracy

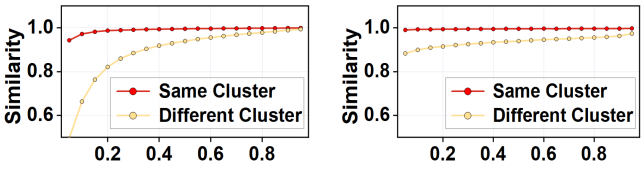
Figure 2: Impact of system heterogeneity on clustering accuracy and model accuracy when combining CFL with MP.

(1) *MP causes model heterogeneity, making it incompatible with CFL.* A natural approach to addressing both data and system heterogeneity is to combine CFL and MP. By using the CFL algorithm to train cluster models, we can avoid the difficulty of training a generalized model on heterogeneous data, while dynamically adjusting the clustering to handle data drift. Moreover, MP is employed to generate models that align with each client's training capacity, addressing the problem of system heterogeneity. However, most of the existing CFL methods are not compatible with heterogeneously pruned models. To validate this idea, we evaluate the effectiveness of combining existing CFL approaches with MP, including FL+HC (Briggs, Fan, and Andras 2020), FlexCFL (Duan et al. 2022), One-Shot (Ghosh et al. 2019), and IFCA (Ghosh et al. 2022). We apply a hybrid pruning strategy to prune local models. To accommodate MP, the clustering algorithms in FL+HC, FlexCFL, and One-Shot are adapted to compute similarity based on the shared portions of the retained structures, rather than the full client models. Specifically, the similarity Sim between two models can be calculated as follows:

$$Sim = ||(m_i \wedge m_j) \cdot (\tilde{\omega}_i - \tilde{\omega}_j)||_2 \quad (1)$$

where m_i and m_j is the corresponding model structure mask of client i and j, $\tilde{\omega}_i$ is the pruned local model of client i. We use a Dirichlet distribution as 1 for data setting, model pruning rates sampled from a normal distribution $\mathcal{N}(0.4, 0.25)$. Figure 2 shows the clustering accuracy and cluster model accuracy of training a CNN on the CIFAR-10 dataset. In a system homogeneous settings, where all clients exhibit similar performance, the pruned models are also comparable, resulting in good baseline accuracy. However, in a heterogeneous system, clustering accuracy and model performance significantly decline. This is because the pruned local models of clients often lack sufficient commonality, making it difficult to generate accurate clustering results. In this case, considering only the similarity of the retained model parts proves insufficient for accurate clustering, leading to a decline in cluster model accuracy.

(2) *Simple structure-based similarity is insufficient for effective clustering clients.* The second intuitive solution involves clustering based on the model structure while considering the similarity between both the retained and pruned parts of the model. Our intuition is that model pruning techniques eliminate the least important parts of the model for each client, and the selection of these parts is influenced by the client's data distribution. Therefore, we can enhance



(a) Pruning ratio (b) Retaining ratio

Figure 3: Model structure similarity between clients under varying pruning ratio and retaining ratio.

clustering algorithms by incorporating information from the pruned parts of the models to improve client clustering.

We design a simple method to validate our intuition. Specifically, we use the NXOR value between masks that represent the model structure as the similarity measure. The details of the similarity calculation method are as follows:

$$Sim = \frac{\sum(m_i \odot m_j)}{Size(m_{i,j})} \tag{2}$$

where $Size(m)$ represents the length of mask m. To evaluate its performance, we utilize the Nxor in conjunction with a hierarchical classification algorithm for client clustering. However, as detailed in Figure 2, it still fails to operate properly when the system is heterogeneous. This finally leads to worse model performance.

This result highlights a key limitation: current structure-based similarity metrics treat all pruned or retained components equally, overlooking their varying importance. To illustrate this, we analyze the model structure similarity focusing separately on pruned and retained parts using a CNN model on CIFAR-10. Figure 3(a) presents similarity computed only on the pruned parts, while Figure 3(b) shows similarity based on the retained parts. When the pruning ratio is low, the similarity calculated from the pruned parts clearly differentiates clients with heterogeneous data distributions, indicating these components' sensitivity to data heterogeneity. However, as the pruning ratio increases, the key structural points that capture client-specific data distribution, which are usually those pruned with higher priority, become diluted among many less important pruned components. This dilution reduces the discriminative power of the similarity metric computed solely on the pruned parts, causing client differences to become less distinguishable at higher pruning ratios (e.g., 0.8). Therefore, effective clustering should be designed to consider the pruning priority of model parameters, relying primarily on the components that best reflect client data distribution characteristics.

Detailed Design of MSCFL

The proposed MSCFL method has two key components: (1) model structure-based clustering, which addresses the compatibility issue between model personalization and federated learning, and (2) a cluster migration strategy, which enhances clustering efficiency under data drift.

Algorithm 1: MSCFL

Input: Total number of communication rounds T, client number N, local training epochs E_{after}, epochs before pruning E_{pre}, pruning ratios $PR = \{pr_1, pr_2, ..., pr_N\}$, initial model ω^0 and consistency threshold τ
Output: Finial models $\{\tilde{\omega}_1^T, \tilde{\omega}_2^T, ..., \tilde{\omega}_n^T\}$

1: Initialize the same initial models ω^0 for each client
2: **while** $t < T$ **do**
3: **Client process:**
4: **for** each client $i = 1, 2, ..., N$ **do**
5: Receive ω_i^t from the server
6: $\omega_i^t \leftarrow$ UPDATE(ω_i^t, E_{pre})
7: $\tilde{\omega}_i^t, m_i^t \leftarrow$ Any MP technique
8: $\tilde{\omega}_i^t \leftarrow$ UPDATE($\tilde{\omega}_i^t$, E_{after})
9: Upload $\tilde{\omega}_i^t$ and m_i^t to the server
10: **end for**
11: **Server process:**
12: $\tilde{W}^t, M^t \leftarrow$ Receive $\tilde{\omega}_i^t$ and m_i^t from each client i
13: **if** ConsistencyChecking(M^{t-2}, M^{t-1}, M^t, τ) is false **then**
14: $C^t \leftarrow$ Algorithm2(M^t, PR)
15: **else**
16: $C^t \leftarrow C^{t-1}$
17: **end if**
18: **for** each cluster c_j^t in C^t **do**
19: $\omega_j^{t+1} \leftarrow$ Compute the average of $\tilde{\omega}_j^t$ from each client in c_j^t, considering only the intersection of retained parameters
20: Send ω_j^{t+1} to each client in c_j^t
21: **end for**
22: Start the next round and set $t \leftarrow t + 1$
23: **end while**
24: Return $\{\tilde{\omega}_1^T, \tilde{\omega}_2^T, ..., \tilde{\omega}_n^T\}$
25: **function** UPDATE(ω, e) ▷ Local model training
26: **for** each local epoch $i = 1, 2, ..., e$ **do**
27: **for** each bath $b \in \beta$ **do**
28: $\omega \leftarrow \omega - \eta * \nabla\mathbb{L}(\omega, b)$
29: **end for**
30: **end for**
31: Return ω
32: **end function**

As summarized in Algorithm 1, the workflow of MSCFL is similar to the vanilla FL methods, with the addition of client model pruning and server clustering operations. Specifically, in the initialization phase, the server initializes the cluster model for each cluster and sets the hyperparameters. In each round of iterative training, the server first assigns the complete cluster model to each client (line 1). Then, the client pre-trains a few epochs of the cluster model and prunes it to an appropriate size using model pruning techniques, which generates a model mask to represent the pruned model structure (lines 6-7). After the pruning process, the client proceeds with subsequent training rounds and uploads the local model and mask to the server (lines 8-9). The server detects client data drift based on the model migration strategy and determines whether re-clustering is needed.

If re-clustering is required, the server computes the client's new cluster identity using the model structure-based clustering algorithm (lines 12-14). Otherwise, the server continues using the previous clustering result to avoid additional workload caused by frequent re-clustering (line 16). Finally, the server aggregates the models that fall into the same cluster on an averaged basis and sends the generated cluster models to each respective client (lines 19-20).

Model Structure-based Clustering To realize efficient clustering based on heterogeneous local models, we propose a similarity computing algorithm (called in line 14 of Algorithm 1), which takes the model structure masks M^t and clients' pruning ratios PR as inputs and returns the clustering result C^t. Algorithm 2 gives the details of the similarity computing algorithm.

Our main idea is to identify the parts of each client's model most susceptible to data distribution changes in order to compute similarity, while ignoring the other parts. In this way, we can avoid the additional overhead that might be introduced by calculating weights. Initially, we initialize the Sim as zeros and calculate the respective retained rations RR based on PR (lines 1-2). Subsequently, to identify the most significant structures that can be used to determine the similarity between models, we consider the lower of the two values, either pr or rr, as the most significant structures (line 5). If pr is the lower value, then the pruned structures are considered to be unique, we utilize the COUNTTARGET$(m_i, m_j, pr, 0)$ function to assess the similarity between these personalized structures; otherwise, we rely on the COUNTTARGET$(m_i, m_j, rr, 1)$ function for this purpose (lines 5-11). Finally, we utilize any clustering method that uses Sim as input in our framework (line 14). It is worth noting that we utilize HC (Briggs, Fan, and Andras 2020) as the default one in our experiments.

The rationality of Algorithm 2 is based on the phenomenon that when the pruning ratios are at the extremes, either very high (such as 90%) or very low (such as 10%), clients with identical data traits often retain or prune the same model structures as detailed in Figure 3. In other words, the most significant structures are those that are either retained or pruned with higher priority in MP. As a result, the minimum of these structures are used to identify models that are similar to one another in the context of CFL. Additionally, the function mentioned above only involves binary operations in nature, and as a result, our performance in terms of speed surpasses that of other comparative baselines.

Cluster Migration Strategy To further accelerate training efficiency and reduce the clustering burden on the server, we have designed a model migration strategy (called in Line 13 of Algorithm 2) to monitor client data drift and perform re-clustering only when there is a significant change in data distribution. This strategy takes the model masks M^{t-2}, M^{t-1} and M^t, along with a consistency threshold τ, as inputs. It then produces a boolean output that indicates whether M^{t-1} and M^t are consistent.

To be specific, we calculate the Hamming Distance between consecutive masks m_i^{t-2} and m_i^{t-1}, as well as m_i^{t-1} and m_i^t, represented by H_i^{t-1} and H_i^t respectively. If the value of H_i^t surpasses the product of H_i^{t-1} and a predefined threshold τ, it is inferred that a change in the data distribution has occurred. In this case, the algorithm returns false, signaling that the server needs to re-cluster the clients. Otherwise, the data distribution change of the client is acceptable, and the server does not cluster it to reduce the computing overhead. Additionally, the computational resources required to compute the bit-wise Hamming Distance are significantly lower compared to those needed for a clustering algorithm. Consequently, we consider it a more efficient method for detecting the occurrence of data drift.

Algorithm 2: Model Structure-based Clustering

Input: Model masks M^t and pruning ratios PR
Output: Clustering result C^t

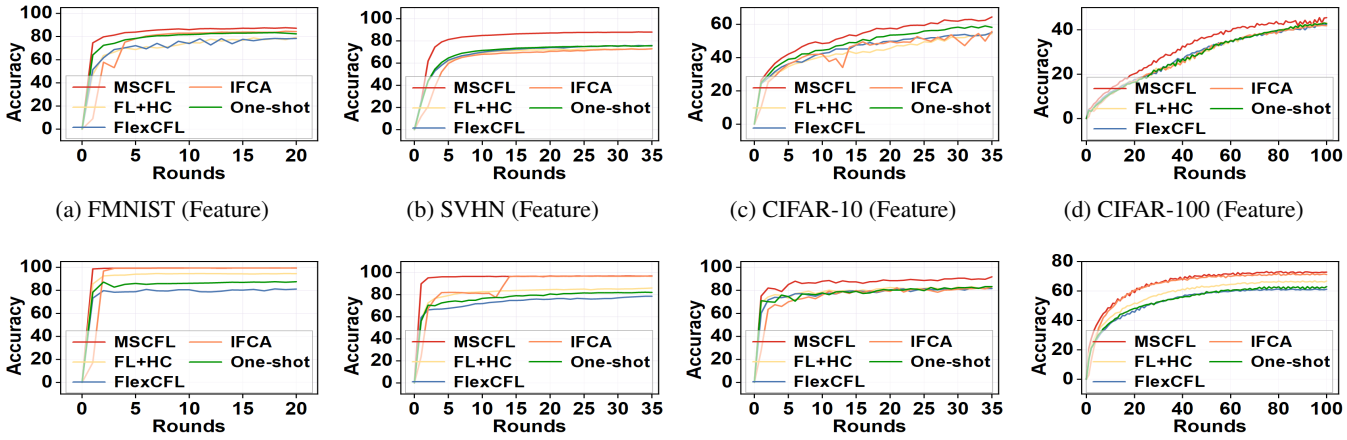
```
1: Initialize the similarity matrix Sim as zeros
2: Set RR ← 1 − PR
3: for each m_i, pr_i, rr_i in M^t, PR, RR do
4:     for each m_j, pr_j, rr_j in M^t, PR, RR do
5:         if the minimum of [pr_i, pr_j, rr_i, rr_j] is pr then
6:             Calculate similarity of pruned structures:
7:             Sim_{i,j} ← COUNTTARGET(m_i, m_j, pr, 0)
8:         else
9:             Calculate similarity of retained structures:
10:            Sim_{i,j} ← COUNTTARGET(m_i, m_j, rr, 1)
11:        end if
12:    end for
13: end for
14: Return C^t ← HC (Briggs, Fan, and Andras 2020)
15: function COUNTTARGET(m_i, m_j, r, target)
16:    cnt ← 0
17:    for each k = 0, 1, ..., len(m) do
18:        if m_i[k] == m_j[k] and m_i[k] == target then
19:            cnt ← cnt + 1
20:        end if
21:    end for
22:    Return cnt/(r * len(m))
23: end function
```

Experiments

In this section, we first present the experimental setup and then the empirical studies of MSCFL on the accuracy and training efficiency of the cluster models. Our experiments were conducted on a server equipped with an RTX 3090 GPU, a Ryzen 7 5800X CPU, and 32GB of memory.

Experimental Setup

Datasets and Models. Our method is tested on four different datasets: FMNIST, SVHN, CIFAR-10, and CIFAR-100. We assess these datasets using customized CNN models. For FMNIST dataset, the model consists of two convolutional layers with 20 and 40 channels respectively, followed by batch normalization, max-pooling, and two fully connected layers with 50 and 10 output logits. As for CIFAR-10, SVHN, the models all have convolutional layers with 32 and 64 channels. For CIFAR-100, we have a more complicated model with 64, 128, 256, and 512 channels.



(a) FMNIST (Feature) (b) SVHN (Feature) (c) CIFAR-10 (Feature) (d) CIFAR-100 (Feature)

(e) FMNIST (Label) (f) SVHN (Label) (g) CIFAR-10 (Label) (h) CIFAR-100 (Label)

Figure 4: Performance of the baselines and MSCFL under feature (TOP) and label (BOTTOM) Non-IID settings.

Baselines. We compare our proposed method with four different CFL methods: **One-Shot** (Ghosh et al. 2019) utilizes K-Means clustering on model updates to group clients; **FL+HC** (Briggs, Fan, and Andras 2020) applies hierarchical clustering to model updates for client grouping; **IFCA** (Ghosh et al. 2022) sends all models to clients, which select the one with the lowest loss; **FlexCFL** (Duan et al. 2022) breaks down the model updates by using truncated SVD and computes the similarity based on the L_2-norm of the cosine similarity. Furthermore, to validate the effect of the proposed cluster migration strategy, we make **MSCFL-α** the version of MSCFL without the cluster migration strategy.

Heterogeneous Settings. To simulate Non-IID data, we consider two scenarios: the label Non-IID and the feature Non-IID. For the label Non-IID, data is divided into 5 groups, each assigned two unique labels based on a Non-IID rate $\beta = 1$ by default (Wang et al. 2020). A smaller β means groups are more likely to receive labels other than the specified ones. For the feature Non-IID, the dataset is split into two: one part remains unchanged, and the other has images rotated by 180 degrees. In both cases, clients are evenly distributed across groups, and data is randomly sampled from each. The pruning ratio pr is strictly set within the range of 0.2 to 0.7, following a gaussian distribution with a mean of $\mu = 0.4$ and a variance of $\sigma^2 = 0.25$ by default to introduce system heterogeneity.

Experimental Results

Model Performance Comparison. We conduct our experiments over a span of 20 to 100 communication rounds, recording the peak accuracy achieved in an environment characterized by data heterogeneity and system heterogeneity. As illustrated in Table 1, MSCFL exhibits a clear advantage and sees its peak increase when compared with baselines, rising by up to 20.61% and 23.47% in feature Non-IID and label Non-IID respectively. The visual data presented in Figure 4 also clearly demonstrates that MSCFL exhibits a faster convergence speed and attains a more advanced level of performance when compared to the baselines. It is

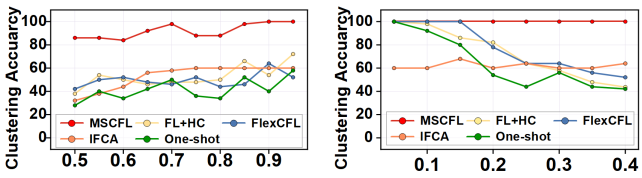
Non-IID Settings	Method	FMNIST	SVHN	CIFAR10	CIFAR100
Feature Non-IID	**MSCFL**	**87.60**	**87.96**	**64.38**	**45.51**
	One-shot	82.39	75.61	58.10	43.06
	FL+HC	79.33	73.04	54.29	43.29
	FlexCFL	78.34	75.53	55.05	42.57
	IFCA	84.24	72.93	55.70	42.01
Label Non-IID	**MSCFL**	99.37	**96.94**	**91.84**	**72.98**
	One-shot	87.53	82.07	83.18	62.98
	FL+HC	94.50	86.01	83.01	66.94
	FlexCFL	81.17	78.51	81.60	61.32
	IFCA	**99.45**	96.64	83.10	71.85

Table 1: Peak accuracy of the baselines and MSCFL under feature Non-IID and label Non-IID.

worth noting that, even when clustering incorrectly, IFCA can sometimes attain similar accuracy to MSCFL, particularly in scenarios with label Non-IID data. This is attributed to the fact that in IFCA, clients consistently select the cluster model with the lowest loss, which in turn facilitates the training of a more generalized model. However, in cases where a single generalized model fails to meet the personalized requirements, the accuracy of IFCA drops immediately. This suggests that MSCFL is capable of fostering the development of more personalized models, a consequence of accurate clustering, which in turn leads to elevated test accuracy.

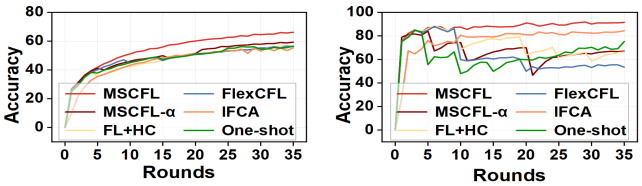
Clustering Accuracy Comparison. According to (Ma et al. 2023), it is evident that inaccurate clustering can result in a scenario called cluster collapse. This underscores the significance of the initial clustering outcomes, as they set the foundation for all subsequent training rounds. Given its pivotal role, we choose to record the clustering accuracy only after the first communication round.

From the data depicted in Figure 5(a), it is apparent that the clustering accuracy for all methods, including MSCFL, tends to rise as the rate of data heterogeneity increases. This



(a) Data heterogeneity rates (b) System heterogeneity rates

Figure 5: Clustering accuracy of baselines and MSCFL under different data heterogeneity rates β and system heterogeneity rates σ^2 based on CIFAR-10.



(a) Feature distribution drift (b) Label distribution drift

Figure 6: Performance of the baselines and MSCFL under feature (LEFT) and label (RIGHT) distribution drift taking place in an interval of 5 rounds of communication based on CIFAR-10. MSCFL-α is the version of MSCFL without cluster migration strategy.

trend can be attributed to the fact that a lower value of β implies that the data held by a client is more generic, encompassing a broader spectrum of knowledge. Conversely, a higher β value indicates that the data is more specific to the client, containing unique characteristics that are more distinctly reflected in the model structure. These distinct features serve as a foundation for more effective clustering processes. Nevertheless, in contrast to MSCFL, the baseline methods continue to face challenges in achieving accurate clustering outcomes, a difficulty attributed to the model heterogeneity.

Furthermore, we explore how the rate of system heterogeneity, as indicated by the pruning ratio, impacts clustering accuracy. By examining Figure 5(b), it becomes clear that as system heterogeneity increases, the clustering accuracy of the baseline methods decreases. This decline is due to the fact that more heterogeneous models have fewer commonalities. In contrast, MSCFL employs masks to compute similarity, effectively circumventing the challenges associated with high heterogeneity. As a result, MSCFL not only achieves the highest clustering accuracy, stabilizing around 100%, but also maintains this level of accuracy even under conditions of significant system heterogeneity.

Effect of Cluster Migration Strategy. In order to evaluate the impact of feature drift and label drift on baselines and MSCFL, we track the progression of model accuracy over 35 rounds of communication with τ set to 1.1, as illustrated in Figure 6. During this process, drifts in the feature or label distribution occur at intervals of 5 rounds. Our observations indicate that drifts in the feature distribution have a relatively small impact on the accuracy of the baseline models, whereas changes in the label distribution result in a significant and abrupt decline in accuracy. This is attributed to the fact that erroneous clustering during data drifts leads to clients developing local models that are not well-suited to their specific data characteristics. It is worth noting that the performance of MSCFL-α degrades like other baselines, thereby demonstrating the effectiveness of our proposed cluster migration strategy. Furthermore, the unpredictable nature of data drifts in practical scenarios also makes it challenging to determine the optimal conclusion point for training sessions. Nevertheless, MSCFL demonstrates remarkable stability in the face of such data drift, due to its ability to promptly detect and respond to changes in the model structure.

Time (s)	FMNIST	SVHN	CIFAR10	CIFAR100
MSCFL	0.12	1.08	1.05	10.18
MSCFL-α	**0.10**	**0.72**	**0.71**	**3.58**
One-Shot	0.13	1.13	1.13	18.47
FL+HC	0.51	6.76	6.61	37.89
FlexCFL	0.23	2.82	2.79	30.50
IFCA	6.64	19.72	20.33	82.22

Table 2: Time cost of the clustering algorithm of baselines and MSCFL under data drift.

Efficiency Analysis. To evaluate the clustering efficiency of MSCFL, we record the total time taken to cluster 250 clients over 20 rounds for FMNIST, 35 rounds for SVHN and CIFAR-10, and 100 rounds for CIFAR-100. As shown in Table 2, our findings indicate that MSCFL outperforms baselines, achieving improvements of 7.69% to 98.19% in the FMNIST dataset, 4.42% to 94.52% in the SVHN dataset, 7.08% to 94.84% in the CIFAR-10 dataset, and 44.88% to 87.62% in the CIFAR-100 dataset. It is worth noting that MSCFL incurs an efficiency reduction of up to 64.83% on CIFAR-100 due to its cluster migration strategy. Despite a decline in efficiency, the trade-off is deemed to be worthwhile, as MSCFL effectively handles data drift and maintains superior overall time cost compared to baselines. The improved performance of MSCFL is due to its similarity computation algorithm and cluster migration strategy, both of which rely solely on binary operations.

Conclusion

In this paper, we introduce a novel CFL framework, termed MSCFL, compatible with the MP technique, to tackle data and system heterogeneity. Furthermore, we also design an efficient cluster migration strategy that re-clusters clients according to the consistency of masks to overcome data drift. The empirical findings demonstrate that our method not only achieves superior model accuracy but also incurs a diminished expenditure of time throughout the clustering procedure. In the future, we aspire to find a more sensitive way to identify data drift. As the Hamming Distance utilized for consistency checking is heavily dependent on the specified threshold, it could be susceptible to more frequent data drift.

Acknowledgements

We would like to heartfully thank the anonymous reviewers for their constructive comments. This work is supported in part by the National Natural Science Foundation of China (No. 62522208, 62272154, 62472163), the Hunan Provincial Natural Science Foundation of China (No. 2024JJ5096), the science and technology innovation Program of Hunan Province (No. 2023RC3125), the Science & Technology talents lifting project of Hunan Province (No. 2023TJ–N23), the Training Program for Excellent Young Innovators of Changsha (No. kq2209008), the Hunan Provincial Department of Education Excellent Youth Project (No. 24B0039).

References

Alam, S.; Liu, L.; Yan, M.; and Zhang, M. 2022. FedRolex: Model-Heterogeneous Federated Learning with Rolling Sub-Model Extraction. In *Conference on Neural Information Processing Systems*, volume 35, 29677–29690.

Bao, H.; Chen, P.; Sun, Y.; and Li, Z. 2025. EFSkip: A New Error Feedback with Linear Speedup for Compressed Federated Learning with Arbitrary Data Heterogeneity. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(15): 15489–15497.

Briggs, C.; Fan, Z.; and Andras, P. 2020. Federated learning with hierarchical clustering of local updates to improve training on non-IID data. In *IEEE International Joint Conference on Neural Networks*, 1–9.

Caldas, S.; Konečný, J.; McMahan, H. B.; and Talwalkar, A. 2018. Expanding the Reach of Federated Learning by Reducing Client Resource Requirements. *CoRR*, abs/1812.07210.

Casado, F. E.; Lema, D.; Criado, M. F.; Iglesias, R.; Regueiro, C. V.; and Barro, S. 2022. Concept drift detection and adaptation for federated and continual learning. *Springer Multimedia Tools and Applications*, 1–23.

Cheng, H.; Zhang, M.; and Shi, J. Q. 2024. A Survey on Deep Neural Network Pruning: Taxonomy, Comparison, Analysis, and Recommendations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12): 10558–10578.

Cheng, Y.; Zhang, W.; Zhang, Z.; Kang, J.; Xu, Q.; Wang, S.; and Niyato, D. 2025. SnapCFL: A Pre-Clustering-Based Clustered Federated Learning Framework for Data and System Heterogeneities. *IEEE Transactions on Mobile Computing*, 24(6): 5214–5228.

Diao, E.; Ding, J.; and Tarokh, V. 2021. HeteroFL: Computation and Communication Efficient Federated Learning for Heterogeneous Clients. In *International Conference on Learning Representations*.

Duan, M.; Liu, D.; Ji, X.; Wu, Y.; Liang, L.; Chen, X.; Tan, Y.; and Ren, A. 2022. Flexible Clustered Federated Learning for Client-Level Data Distribution Shift. *IEEE Transactions on Parallel and Distributed Systems*, 33(11): 2661–2674.

Ghosh, A.; Chung, J.; Yin, D.; and Ramchandran, K. 2022. An Efficient Framework for Clustered Federated Learning. *IEEE Transactions on Information Theory*, 68(12): 8076–8091.

Ghosh, A.; Hong, J.; Yin, D.; and Ramchandran, K. 2019. Robust Federated Learning in a Heterogeneous Environment. *CoRR*, abs/1906.06629.

Guo, Y.; Tang, X.; and Lin, T. 2025. Enhancing Clustered Federated Learning: Integration of Strategies and Improved Methodologies. In *The Thirteenth International Conference on Learning Representations*.

He, Y.; Liu, P.; Wang, Z.; Hu, Z.; and Yang, Y. 2019. Filter Pruning via Geometric Median for Deep Convolutional Neural Networks Acceleration. In *IEEE/CVF Computer Vision and Pattern Recognition Conference*, 4340–4349.

Horváth, S.; Laskaridis, S.; Almeida, M.; Leontiadis, I.; Venieris, S.; and Lane, N. 2021. FjORD: Fair and Accurate Federated Learning under heterogeneous targets with Ordered Dropout. In *Conference on Neural Information Processing Systems*, volume 34, 12876–12889.

Jothimurugesan, E.; Hsieh, K.; Wang, J.; Joshi, G.; and Gibbons, P. B. 2023. Federated Learning under Distributed Concept Drift. In *JMLR International Conference on Artificial Intelligence and Statistics*, volume 206, 5834–5853.

Kim, H.; and Yoo, J. 2025. Singular Value Scaling: Efficient Generative Model Compression via Pruned Weights Refinement. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(17): 17859–17867.

Li, A.; Sun, J.; Li, P.; Pu, Y.; Li, H.; and Chen, Y. 2021. Hermes: an efficient federated learning framework for heterogeneous mobile clients. In *ACM International Conference on Mobile Computing and Networking*, 420–437.

Li, H.; Kadav, A.; Durdanovic, I.; Samet, H.; and Graf, H. P. 2017. Pruning Filters for Efficient ConvNets. In *International Conference on Learning Representations*.

Li, J.; Chen, T.; and Teng, S. 2024. A comprehensive survey on client selection strategies in federated learning. *Elsevier Computer Networks*, 251: 110663.

Li, Q.; Diao, Y.; Chen, Q.; and He, B. 2022. Federated Learning on Non-IID Data Silos: An Experimental Study. In *IEEE International Conference on Data Engineering*, 965–978.

Lin, M.; Ji, R.; Wang, Y.; Zhang, Y.; Zhang, B.; Tian, Y.; and Shao, L. 2020. HRank: Filter Pruning Using High-Rank Feature Map. In *IEEE/CVF Computer Vision and Pattern Recognition Conference*, 1529–1538.

Lin, S.; Ji, R.; Yan, C.; Zhang, B.; Cao, L.; Ye, Q.; Huang, F.; and Doermann, D. 2019. Towards Optimal Structured CNN Pruning via Generative Adversarial Learning. In *IEEE/CVF Computer Vision and Pattern Recognition Conference*, 2790–2799.

Liu, B.; Ma, Y.; Zhou, Z.; Shi, Y.; Li, S.; and Tong, Y. 2024. CASA: Clustered Federated Learning with Asynchronous Clients. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, 1851–1862.

Liu, Q.; Sun, S.; Liang, Y.; Liu, M.; and Xue, J. 2025. Personalized Federated Learning for Spatio-Temporal Forecasting: A Dual Semantic Alignment-Based Contrastive Approach. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(11): 12192–12200.

Liu, Z.; Li, J.; Shen, Z.; Huang, G.; Yan, S.; and Zhang, C. 2017. Learning Efficient Convolutional Networks through Network Slimming. In *IEEE International Conference on Computer Vision*, 2755–2763.

Long, G.; Xie, M.; Shen, T.; Zhou, T.; Wang, X.; and Jiang, J. 2023. Multi-center federated learning: clients clustering for better personalization. In *Springer World Wide Web*, volume 26, 481–500.

Ma, J.; Zhou, T.; Long, G.; Jiang, J.; and Zhang, C. 2023. Structured Federated Learning through Clustered Additive Modeling. In *Conference on Neural Information Processing Systems*, volume 36, 43097–43107.

McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and Arcas, B. A. y. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *JMLR International Conference on Artificial Intelligence and Statistics*, volume 54, 1273–1282.

Nguyen, D. C.; Pham, Q.-V.; Pathirana, P. N.; Ding, M.; Seneviratne, A.; Lin, Z.; Dobre, O.; and Hwang, W.-J. 2022. Federated Learning for Smart Healthcare: A Survey. *ACM Computing Surveys*, 55(3).

Panchal, K.; Choudhary, S.; Mitra, S.; Mukherjee, K.; Sarkhel, S.; Mitra, S.; and Guan, H. 2023. Flash: Concept Drift Adaptation in Federated Learning. In *ACM International Conference on Machine Learning*, volume 202, 26931–26962.

Sattler, F.; Müller, K.-R.; and Samek, W. 2021. Clustered Federated Learning: Model-Agnostic Distributed Multitask Optimization Under Privacy Constraints. *IEEE Transactions on Neural Networks and Learning Systems*, 32(8): 3710–3722.

Vahidian, S.; Morafah, M.; and Lin, B. 2021. Personalized Federated Learning by Structured and Unstructured Pruning under Data Heterogeneity. In *IEEE International Conference on Distributed Computing Systems Workshops*, 27–34.

Vahidian, S.; Morafah, M.; Wang, W.; Kungurtsev, V.; Chen, C.; Shah, M.; and Lin, B. 2023. Efficient Distribution Similarity Identification in Clustered Federated Learning via Principal Angles between Client Data Subspaces. In *AAAI Conference on Artificial Intelligence*, volume 37, 10043–10052.

Wang, H.; Jia, Y.; Zhang, M.; Hu, Q.; Ren, H.; Sun, P.; Wen, Y.; and Zhang, T. 2024. FedDSE: Distribution-aware Submodel Extraction for Federated Learning over Resource-constrained Devices. In *ACM International World Wide Web Conference*, 2902–2913.

Wang, H.; Kaplan, Z.; Niu, D.; and Li, B. 2020. Optimizing federated learning on non-iid data with reinforcement learning. In *IEEE International Conference on Computer Communications*, 1698–1707.

Zhang, C.; Xu, Y.; Tan, J.; An, J.; and Jin, W. 2025. MingledPie: A Cluster Mingling Approach for Mitigating Preference Profiling in CFL. In *Network and Distributed System Security Symposium*.

Zhang, C.; Xu, Y.; Wu, X.; Wang, E.; Jiang, H.; and Zhang, Y. 2024. A Semi-Asynchronous Decentralized Federated Learning Framework via Tree-Graph Blockchain. In *IEEE International Conference on Computer Communications*, 1121–1130.

Zhou, G.; Li, Q.; Liu, Y.; Zhao, Y.; Tan, Q.; Yao, S.; and Xu, K. 2024a. FedPAGE: Pruning Adaptively Toward Global Efficiency of Heterogeneous Federated Learning. *IEEE/ACM Transactions on Networking*, 32(3): 1873–1887.

Zhou, Y.; Pang, X.; Wang, Z.; Hu, J.; Sun, P.; and Ren, K. 2024b. Towards Efficient Asynchronous Federated Learning in Heterogeneous Edge Environments. In *IEEE International Conference on Computer Communications*, 2448–2457.