Towards Efficient Asynchronous Federated Learning in Heterogeneous Edge Environments

Yajie Zhou[†,♯], Xiaoyi Pang[†,♯], Zhibo Wang[†,♯,*], Jiahui Hu[†,♯], Peng Sun[≀] and Kui Ren[†,♯]
[†]The State Key Laboratory of Blockchain and Data Security, Zhejiang University, Hangzhou, China
[♯]School of Cyber Science and Technology, Zhejiang University, Hangzhou, China
[≀]College of Computer Science and Electronic Engineering, Hunan University, Changsha, China
Email: {yajiezhou, zhibowang, jiahuihu, kuiren}@zju.edu.cn, xypang@whu.edu.cn, psun@hnu.edu.cn

***Abstract*—Federated learning (FL) is widely used in edge environments as a privacy-preserving collaborative learning paradigm. However, edge devices often have heterogeneous computation capabilities and data distributions, hampering the efficiency of co-training. Existing works develop staleness-aware semi-asynchronous FL that reduces the contribution of slow devices to the global model to mitigate their negative impacts. But this makes data on slow devices unable to be fully leveraged in global model updating, exacerbating the effects of data heterogeneity. In this paper, to cope with both system and data heterogeneity, we propose a clustering and two-stage aggregation-based Efficient Asynchronous Federated Learning (EAFL) framework, which can achieve better learning performance with higher efficiency in heterogeneous edge environments. In EAFL, we first propose a gradient similarity-based dynamic clustering mechanism to cluster devices with similar system and data characteristics together dynamically during the training process. Then, we develop a novel two-stage aggregation strategy consisting of staleness-aware semi-asynchronous intra-cluster aggregation and data size-aware synchronous inter-cluster aggregation to efficiently and comprehensively aggregate training updates across heterogeneous clusters. With that, the negative impacts of slow devices and Non-IID data can be simultaneously alleviated, thus achieving efficient collaborative learning. Extensive experiments demonstrate that EAFL is superior to state-of-the-art methods.**

***Index Terms*—Asynchronous Federated Learning, System Heterogeneity, Data Heterogeneity, Clustering**

I. Introduction

Federated learning (FL) is a privacy-preserving collaborative learning paradigm that can train a joint model across devices (called clients) without direct access to their local data, which has been widely adopted in edge environments [1]–[6]. Each client uploads local updates to the server after completing local training, and the server aggregates the updates from all participating clients to update a new global model. However, edge environments are usually highly heterogeneous, i.e., edge devices have different computing capabilities (called system heterogeneity) and local data is Not Identically and Independently Distributed (Non-IID) (called data heterogeneity), posing challenges of efficiency and utility to FL.

Specifically, system heterogeneity results in different local training time, which means fast clients have to wait for slow clients, thus greatly reducing the learning efficiency. Recently, semi-Asynchronous Federated Learning (AFL) [7]–[11] has been proposed to deal with the system heterogeneity and improves efficiency by aggregating clients' local updates in a semi-asynchronous manner. At each iteration, the server aggregates local updates from the fastest K clients and broadcasts the updated global model to them, which significantly decreases the waiting time of each iteration. Considering that clients may update based on stale global models that are updated in different iterations while high-stale updates have an adverse effect on the global model, most AFL algorithms [9]–[11] adopt staleness-aware weighted aggregation to reduce the contribution of high-stale updates. Such algorithms called staleness-aware AFL (SAFL) can further mitigate the effect of slow clients and improve learning performance. As for the data heterogeneity in edge environments, it leads to drift in the local model [12], [13], i.e., the update directions of different local models would be inconsistent. Aggregating such heterogeneous updates would inevitably degrade the accuracy of the global model, which has been validated in many existing works [14], [15]. To solve the data heterogeneity issue, most existing works [16]–[20] try to leverage the overall data and enhance the role played by global information during the FL training process, thus the global model can learn comprehensively and achieve better performance on various data distributions.

Although SAFL, the mainstream AFL scheme, improves training efficiency, it violates the principle of solving the Non-IID problem, thus SAFL suffers a degradation of model utility under data heterogeneity. Firstly, in SAFL, fast-trained clients participate in global aggregation more frequently than slow ones. Therefore, local data on slow clients would be underused by the global model. Secondly, since slower clients are more likely to update based on more stale global models, they would be assigned lower weights during the global aggregation, which further prevents them from fully participating in global model updating and reduces the contribution of their local data to the global model, exacerbating the negative impact of data heterogeneity. Some methods [8], [20] have been proposed to improve the performance of AFL in Non-IID scenarios, but they can not ensure that all kinds of data distributions whether distributed across fast or slow clients can be fully learned by the global model, thus the global model would still be biased. That is, existing works cannot tackle system heterogeneity and data heterogeneity well at the same time. Thus, it is necessary

[*] Zhibo Wang is the corresponding author

to design a more efficient FL algorithm to cope with both data and system heterogeneity to achieve better training efficiency and utility.

In this paper, our goal is to cope with both system heterogeneity and data heterogeneity to achieve efficient federated learning in heterogeneous edge environments. The challenges we face are: *1) how to fully learn from clients with heterogeneous system and data characteristics without being encumbered with slow clients?* Involving all clients in training can effectively leverage the overall data and fully learn the global information, but slow devices would dramatically affect the training efficiency. Having only fast devices involved in training can significantly improve efficiency, but it's hard to learn from all kinds of data distributions. Therefore, it is highly challenging to realize a trade-off between client participation and training efficiency. *2) how to always effectively and comprehensively aggregate clients' local updates to achieve better performance?* When the global aggregation is performed in an asynchronous manner, at each iteration, the contribution to global model of local updates from every client participating in the aggregation changes as its staleness varies. Therefore, it is challenging to aggregate clients' updates in a suitable way during the long training process.

To solve the challenges, we propose a clustering and two-stage aggregation-based Efficient Asynchronous Federated Learning framework, named EAFL, which can achieve efficient collaborative learning in edge environments with both system and data heterogeneity. First, we propose a gradient similarity-based dynamic clustering mechanism to cluster clients with similar system and data characteristics together. The homogeneity of clients in terms of data and system can be reflected by the consistency of their local update directions, which are further embodied in the similarity of gradients. Therefore, we dynamically group clients with similar gradient directions into the same cluster. Second, we design a novel two-stage aggregation strategy to efficiently and comprehensively aggregate various updates across heterogeneous clusters, including staleness-aware semi-asynchronous intra-cluster aggregation and data size-aware synchronous inter-cluster aggregation. Within cluster, the intra-cluster aggregation is semi-asynchronous to mitigate the straggler effect and improve the training efficiency. And to avoid being affected by high-stale clients, we proposed weighted aggregation according to staleness. Inter-cluster aggregation is synchronous to aggregate various updates from heterogeneous clients, thus making full use of global information. And it is also data size-aware to allow the aggregated updates to better represent the entire cluster rather than individual client.

Our main contributions can be summarized as follows:

- We propose an efficient asynchronous federated learning based on clustering and two-stage aggregation, which fills the gap that data heterogeneity and system heterogeneity cannot be well solved at the same time.
- We propose a gradient similarity-based dynamic clustering and a two-stage aggregation strategy with a combination of synchronous and asynchronous manners to ensure that there is always an efficient and comprehensive aggregation of heterogeneous clients' updates.
- Theoretical analysis validates the nice convergence property of EAFL. Besides, we conduct an extensive experimental evaluation of EAFL, which demonstrates that EAFL outperforms existing state-of-the-art schemes in terms of test accuracy, especially in highly heterogeneous edge environments.

II. Related Work

In this section, we introduce the related work of system and data heterogeneity in FL, and clustering-based FL.

System heterogeneity in FL. System heterogeneity leads to different processing time of clients, making clients may train their local models based on global models with different levels of staleness in AFL. Dai et al. [21] found that local updates with high staleness would hamper the efficiency and performance of the global model. To reduce the impact of stale updates, most studies developed staleness-aware weighted aggregation. In [9], [22], temporally weighted aggregations were proposed to aggregate local updates with a fading weight according to the staleness of clients. Wang et al. [23] measured the staleness of updates by the Euclidean distance between the stale model and the current global model. To cope with the staleness problem caused by system heterogeneity, EAFL in this paper also adopts staleness-aware weighted aggregation to alleviate the effect of high-stale updates on global model.

Data heterogeneity in FL. It is undisputed that data heterogeneity has a negative influence on FL. Previous researches have shown that Non-IID data leads to drift in the local model, finally reducing global model performance [12], [13]. To solve the issue, the core idea is to utilize the collective data and amplify the influence of global information throughout the federated learning training procedure. For example, in [16], the global model is warmed up with data shared by clients. Similarly, in [17]–[19], researchers used generators to learn the global data distribution and generate samples to construct a global shared synthetic dataset or augment local data of clients. Wang et al. [20] applied momentum to accumulate historical information since the previous aggregated gradients contain rich global information. Our proposed EAFL also tries to make full use of global information during training.

Clustering-based FL. Clustering is often used in heterogeneous environments to cluster clients with certain homogeneity together to improve efficiency of FL. Lee et al. [24] clustered clients based on geographic location and data distributions to simultaneously maximize accuracy and communication speed. Some methods constructed a multi-center FL framework by grouping clients with similar local training data into the same cluster to avoid global model performance degradation caused by Non-IID data. They measured data similarity through model weights [25] or loss value [26]. After clustering, aggregation takes place first within clusters and then between clusters. Wang et al. [27] adopted synchronous aggregation for intra-cluster updates while adopting asynchronous aggregation for inter-cluster updates to improve training efficiency, which

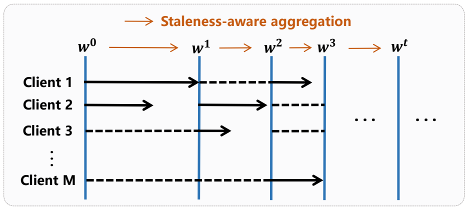


Fig. 1. A schematic of 2-SAFL

cannot address data heterogeneity. Regardless of the clustering criteria, clustering-based FL always groups clients that are similar in a certain feature into the same cluster, inspiring us to deal with system and data heterogeneity.

III. Problem Definition

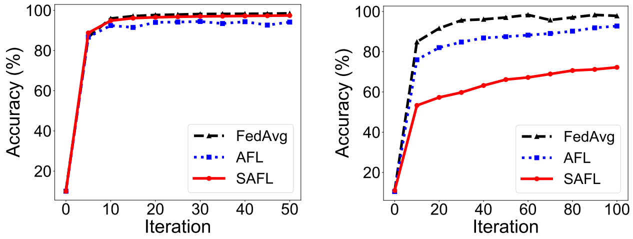
In this section, we first define our system model. Then we formally present the problem to be solved in this paper.

A. System model

In this paper, we consider a FL system in a heterogeneous edge environment, consisting of a server and multiple heterogeneous edge devices (clients). Clients are heterogeneous in terms of data and system. That is, the distribution of clients' local training data is different, and the time required to complete local training for each client is different. Due to the superior performance of the clustering methods in dealing with heterogeneity, to solve the data heterogeneity and system heterogeneity and achieve efficient FL, we adopt clustering in our FL system. To be specific, clients would be divided into several clusters based on a certain criterion. Each cluster would choose a client therein as the cluster head, and other clients interact with the server through the cluster head rather than interact with the server directly. In this case, each client trains its model locally as usual and uploads the local updates to the cluster head, and the cluster head is responsible for 1) aggregating local updates uploaded by clients within cluster, i.e., intra-cluster aggregation, and sending the aggregated updates to clients, 2) receiving global updates from the server and sending them to clients that participate in this intra-cluster aggregation. The server performs inter-cluster aggregation in order to aggregate updates uploaded by cluster heads to update the global model, and then sends the new global back to corresponding cluster heads.

B. Problem Formulation

Suppose there are a server and M heterogeneous clients. Each client $i \in \{1, 2, \cdots, M\}$ has its own training data D_i and $|D_i|$ denotes the client i's local data size. Since clients are heterogeneous in data distribution, $D_1, D_2, \cdots, D_M$ are Non-IID, and they may contain samples belonging to different kinds of labels with different sizes. Besides, due to the system heterogeneity, clients have different operation time, denoted by $E = \{e_1, e_2, \cdots, e_M\}$. We tend to divide these clients into N clusters $X = \{X_1, X_2, \cdots, X_N\}$, where each X_n ($n \in \{1, 2, \cdots, N\}$) is the ID set of part of clients, and $X_1 \cap X_2 \cap \cdots \cap X_N = \varnothing$ and $X_1 \cup X_2 \cup \cdots \cup X_N = \{1, 2, \cdots, M\}$. Each cluster X_n has a cluster head C_n. In this paper, we aim to achieve efficient collaborative learning in heterogeneous edge environments. The problem to be solved is: with the assistance of clusters $\{X_1, X_2, \cdots, X_N\}$, how to realize efficient collaboration between heterogeneous clients with various local training time $\{e_1, e_2, \cdots, e_M\}$ and various local training data $\{D_1, D_2, \cdots, D_M\}$ from different distributions, so as to alleviate the impact of slow clients and uneven data distribution, achieving efficient federated learning and getting a global model w with high performance.



(a) IID setting

(b) Non-IID setting

Fig. 2. The illustration of the impact of Non-IID data on SAFL.

IV. Preliminaries

This section first reviews the operation precess of FL and SAFL, then analyze the drawbacks of SAFL.

A. Federated Learning

FL [28]–[31] is regarded as a privacy-preserving collaborative learning paradigm that is widely used in privacy-sensitive edge environments, which enables edge devices (called clients) to jointly train a global model without directly sharing their local data with the central server. In FL, the training process occurs on clients rather than the centralized server. At each global iteration, each participating client performs local training and uploads local updates to the server, and the server aggregates all received updates in a synchronous manner to update a new global model and broadcast it to all clients for their further updating.

B. Staleness-aware Asynchronous Federated Learning

The efficiency of FL can be significantly degraded by data and system heterogeneity. For example, if a few clients experience delay, the entire FL system's progress can be affected. SAFL [9]–[11] can improve the efficiency of FL. First, it mitigates the impact of slow clients since it enables clients to independently submit their updates without waiting for others. Further, it improves the learning performance by aggregating local updates weighted according to staleness.

In SAFL, the server performs staleness-aware global aggregation to get the new global model once receiving local updates from the fastest K clients, and the updated global model would be broadcasted to only these K clients. Meanwhile, other clients continue their respective local training to prepare for future global aggregations. Note that the staleness of a client measures how stale the global model is in local training, which can be defined as the number of experienced iterations since the client's last participation in global aggregation. Fig.

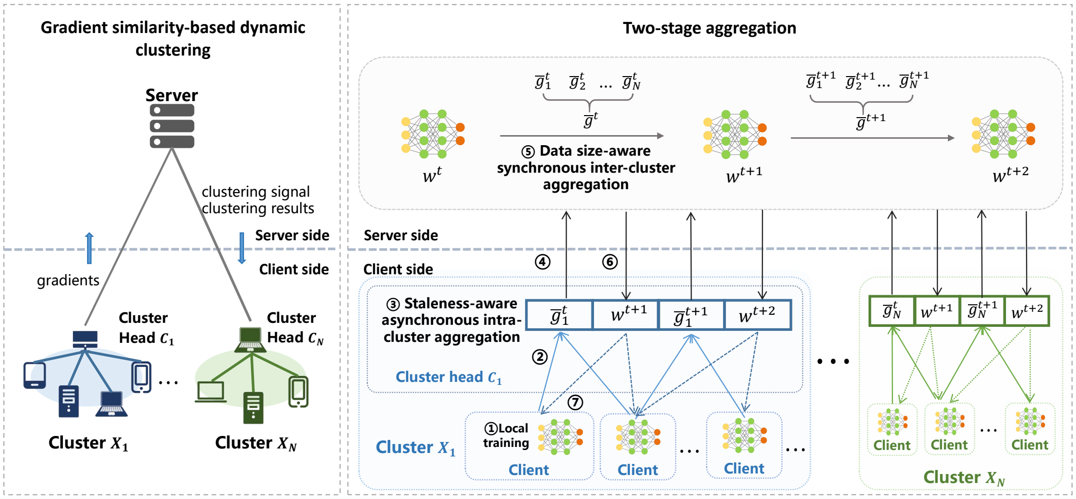


Fig. 3. The overview of EAFL

1 gives an example of SAFL where $K = 2$. Take iteration 2 as an example, Client 2 and Client 3 are the fastest two clients in this iteration. Since their local updates are based on w^1 and w^0, respectively, their staleness are 1 and 2, respectively. So the server aggregates Client 2 and Client 3's local updates with weights inversely proportional to their staleness to obtain the updated global model w^2.

Let $\mathcal{V}^t$ ($|\mathcal{V}^t| = K$) denotes the fastest K clients at iteration t, and the staleness of each client $i \in \mathcal{V}^t$ is τ_i^t, then SAFL updates the global model w^t at iteration t by:

$$w^t = (1-\alpha)w^{t-1} - \alpha \sum_{i\in\mathcal{V}^t} \frac{|D_i| \cdot S(\tau_i^t) \cdot w_i^t}{|D^t|}, \tag{1}$$

where w_i^t is the local model of client i, α is a hyperparameter and $S(\tau_i^t)$ is the staleness-aware weight of client i, which is inversely proportional to client i's staleness τ_i^t, e.g., $S(\tau_i^t) = 1/\tau_i^t$. $|D^t|$ is the total number of local data samples of the K fastest clients at iteration t, and $D^t = \cup_{i\in\mathcal{V}^t} D_i$.

C. Drawbacks of SAFL

SAFL can solve the system heterogeneity and perform well in terms of efficiency and accuracy in an IID setting. However, it will most likely suffer degradation in accuracy in the Non-IID setting. This is because it tends to choose fast clients to participate in the global aggregation. In this circumstance, if the rare and unique data is only held by slow clients, this kind of data can not play a full role in global model training. Even worse, staleness-aware aggregation exacerbates the negative effects of data heterogeneity since it weights the updates based on the staleness of the clients. Specifically, slow clients are usually with high staleness due to their infrequent participation in the global aggregation. As a result, slow clients are assigned with smaller weights and its rare and unique data contributes less to the global model. Therefore, SAFL, the existing mainstream method to improve the efficiency of FL, can not well tackle system heterogeneity and data heterogeneity simultaneously.

We conduct experiments on classification tasks on MNIST [32] to demonstrate the above deduction. We consider an IID setting and a Non-IID setting where each client only has data belonging to 1 label. In these two settings, to illustrate the impact of Non-IIDness on the performance of SAFL, we separately compare SAFL with the typical FL algorithm - FedAvg [33] and the typical synchronous federated learning algorithm [7] without staleness-aware aggregation - AFL. The results are shown in Fig. 2. We can observe that in the IID setting, SAFL improves the performance of AFL, and they both perform about the same as the mostly used FL algorithm – FedAvg. But in the Non-IID setting, both AFL and SAFL have much lower accuracy than FedAvg, and SAFL has a lower accuracy than KAFL. This shows that AFL can not solve the data heterogeneity issue well, and the staleness-based weighted aggregation method in SAFL plays a counterproductive role in the Non-IID setting, which is consistent with our analysis.

V. Mechanism Design

In this paper, we try to fill the gap that data heterogeneity and system heterogeneity cannot be well solved at the same time. To this end, we propose a clustering and two-stage aggregation-based Efficient Asynchronous Federated Learning framework, named EAFL, to achieve efficient learning and get the global model with higher utility in heterogeneous edge environments. In this section, we first give an overview of EAFL, and then introduce the proposed mechanisms in detail.

A. Overview of EAFL

Fig. 3 shows the framework of EAFL, which consists of two main mechanisms: a Gradient similarity-based Dynamic Clustering mechanism (GDC) and a novel two-stage aggregation strategy. The former is performed on the server, which dynamically assigns clients with the same direction of local updates (i.e., gradients) to the same cluster to make clients in each cluster have similar system and data characteristics. The latter contains the Staleness-aware semi-Asynchronous intra-cluster Aggregation (SAA) processed at each cluster and a Data size-aware Synchronous inter-cluster Aggregation (DSA) at the server. With SAA, we can overcome the resistance of slow clients and gradients with high staleness, thus efficiently aggregating local updates from homogeneous clients within each cluster. With DSA, we can comprehensively aggregate updates from heterogeneous clusters to ensure that clients with each kind of characteristic of data and system can fully participate in global aggregation and contribute to the global model. With our designs, the negative impacts of slow devices and uneven data can be simultaneously alleviated, thus achieving efficient collaborative learning and improving the global model performance.

The general process of EAFL is shown in Alg. 1. We respectively introduce two important subjects in training: server and clients. The server has two main functions: 1) Every R rounds, the server needs to send a clustering signal to clients, collects the gradients from all clients, and performs GDC (Line 4-12). 2) Performs DSA, and distributes new model (Line 14-15). If a client is chosen as cluster head, it undertakes the interactions between clients within the cluster and server. It collects updates from the fastest φ ratio clients within the cluster and performs local updating at the same time, performs SAA and then sends the aggregated updates to the server. Besides, it sends the new model to clients participating in the intra-cluster aggregation (Line 27-30). What clients need to do is 1) sending gradients to the server when they receive a clustering signal (Line 19-25) and 2) performing local training and then uploading updates to the cluster head, renewing the local model after receiving a new global model. (Line 32-33).

B. Gradient Similarity-based Dynamic Clustering

We aim to cluster clients with similar system and data characteristics together. Clients with consistent system characteristics tend to have similar update frequencies, resulting in similar staleness. Furthermore, clients with similar staleness have similar initial models, leading to similar directions of local model updates. Thus, clients with homogeneous system characteristics are most likely to have similar local update directions. Besides, clients with consistent data characteristics also have similar directions of local model updates, since their local training data have analogous features. In summary, clients with similar system and data characteristics tend to have similar directions of model updates, which are indicated by the gradient descent direction in training in most machine learning frameworks. Therefore, we cluster clients based on the similarity of gradient directions so that clients in each cluster can be approximately homogeneous in data and system.

We use the cosine distance to measure the gradient direction similarity across clients. The larger the cosine distance is, the more consistent the direction of descent of two gradients is, and the more similar the two clients' characteristics of data and system are. For gradients $\vec{g}$ and $\vec{g'}$ from any two clients, the cosine distance is $cos(\vec{g},\vec{g'}) = \frac{\vec{g}\cdot\vec{g'}}{|\vec{g}|\cdot|\vec{g'}|}$.

After measuring the gradient similarity of the clients via cosine distance, we use the K-Means [34] algorithm to cluster the clients with high gradient similarity into the same cluster. Specifically, M clients $\{1,2,\cdots,M\}$ are divided into N non-overlapping clusters $X=\{X_1,X_2,\cdots,X_N\}$ with the goal to minimize the sum of distances from all clients' gradients to the centroid of the cluster they belong to, i.e. $\sum_{n=1}^{N}\sum_{i\in X_n} distance(g_i, centroid_n)$, where g_i denotes the gradient of client i, and $centroid_n$ is the centroid of X_n. The centroid of a cluster is defined as the mean of all samples in the cluster, i.e. $centroid_n = \frac{1}{|X_n|}\cdot\sum_{i\in X_n} g_i, \forall X_n \in X$. Then the server randomly chooses a cluster head C_i for each cluster X_n $(n\in\{1,2,\cdots,N\})$.

Alg. 1 EAFL Algorithm

```
Input: re-clustering iteration interval $R$, percentage of clients participate in each iteration $\varphi$, maximum iteration $T$, number of clusters $N$
 1: Run Server() thread on the server and Client() thread on each client in parallel
 2: Thread Server():
 3: for $t = 0$ to $T$ do
 4:     if $R \mid t$ then
 5:         if $t = 0$ then
 6:             Send initial model to all clients
 7:         end if
 8:         Send clustering signal to all clients
 9:         Wait until receiving gradients from all clients
10:         Perform GDC to cluster clients into $N$ clusters and randomly select a client as cluster head from each cluster
11:         Send result to all clients
12:     end if
13:     Wait for intra-cluster aggregated updates from all cluster heads
14:     Perform DSA to update the global model
15:     Send current global model to all cluster heads
16: end for
17: Thread Client():
18: repeat
19:     if Receive clustering signal then
20:         if $t = 0$ then
21:             Perform local training to get gradients
22:         end if
23:         Send gradients to server
24:         Wait until receiving clustering result
25:     end if
26:     if This client is cluster head then
27:         Perform local training while waiting for updates from the fastest $\varphi$ ratio clients at the same time
28:         Perform SAA once receiving enough updates
29:         Send the update after aggregation to server and record the ID of clients within cluster participated in this aggregation
30:         Wait until receive global model from server and send new global model to clients participating in this aggregation
31:     else
32:         Perform local training and send updates to cluster head
33:         Wait until receive global model from its cluster head and renew the local model
34:     end if
35: until The whole training process has finished
Output: the final global model
```

In AFL, since the global aggregation is performed in an asynchronous manner, clients with consistent system characteristics do not always have similar update frequencies and staleness. Therefore, it is not absolute that clients with similar staleness have consistent system characteristics - but the probability is very high. We believe that clients with similar gradients have approximate staleness and data distribution, and grouping them into the same cluster can to a large extent ensure that they are isomorphic in data and system characteristics. However, the staleness of each client is dynamically changing during the training process, which exacerbates the uncertainty of the gradient similarity-based clustering. To resolve such uncertainty, we dynamically perform the gradient similarity-based clustering every R global iterations to ensure that clients within each cluster have consistent staleness and data distribution, thus having similar system and data characteristics as much as possible.

C. Two-stage Aggregation

In clustering-based FL, how to aggregate updates within and among clusters greatly affects global learning efficiency. In this paper, we propose a novel two-stage aggregation strategy that combines SAA and DSA to fully learn about global information and reduce the damage to the global model from high-stale client updates, thus improving the efficiency of global learning.

Intra-cluster aggregation. The intra-cluster aggregation takes semi-asynchronous aggregation to improve training efficiency. In this way, within each cluster, fast clients are not required to wait for slow clients, thus mitigating the straggler effect. Meanwhile, intra-cluster aggregation is staleness-aware, which reduces the participation of high-stale gradients to mitigate their negative impact.

Suppose that a client i is participating in intra-cluster aggregation at the current global iteration t, and the last time it participated in intra-cluster aggregation was at global iteration t', i.e., the client is currently updated based on the global model $w^{t'}$ of iteration t'. Then the staleness of client i at current iteration t is computed by $\tau_i^t = t - t'$. Besides, the local updating process of this client can be expressed by:

$$w_i^t = w^{t'} - \eta \cdot \nabla f(w^{t'}, D_i), \tag{2}$$

where D_i and η are the local training dataset and learning rate, respectively. f is the loss function of local training and $\nabla f(w^{t'}, D_i)$ is the local gradient. After local training, the client uploads the gradient to the corresponding cluster head.

At the iteration t, for each cluster X_n $(n \in \{1, 2, \cdots, N\})$, the cluster head C_n aggregates local updates (i.e., gradient) uploaded by the fastest φ ratio clients within the cluster via staleness-aware aggregation. Let V_n^t denotes the ID set of the fastest φ ratio clients, and their staleness set is $\{\tau_i^t\}_{i \in V_n^t}$. During the aggregation, the cluster head C_n assigns a weight p_i^t to each client $i \in V_n^t$ based on its staleness τ_i^t by a inverse proportional function:

$$p_i^t = 1/\tau_i^t. \tag{3}$$

Then staleness-aware semi-asynchronous intra-cluster aggregation processed at each cluster X_n at iteration t can be expressed by:

$$\bar{g}_n^t = \sum_{i \in V_n^t} \frac{|D_i|}{|D_n^t|} \cdot p_i^t \cdot \nabla f(w_i^t, D_i), \tag{4}$$

where $\nabla f(w_i^t, D_i)$ is the local update of client i at iteration t based on w_i^t and D_i, i.e., gradient. $|D_n^t|$ is the total number of data samples of clients that participated in intra-cluster aggregation in cluster X_n, and $|D_n^t| = \sum_{i \in V_n^t} |D_i|$.

Inter-cluster aggregation. We adopt DSA for inter-cluster aggregation, which allows each cluster to participate in training with equal probability, making clients of every data distribution and computation capability participate in training to alleviate the problem that the global model is biased towards fast-training clients. In most federated learning paradigms, the server performs weighted aggregation based on the amount of data from the clients participating in the aggregation. But considering that the gradient uploaded by a particular cluster does not only represent the situation of the clients participating in the intra-cluster aggregation, but also the situation of the whole cluster, the inter-cluster aggregation is performed on the basis of the entire cluster's weighted data volume.

Let $|D_n|$ denote the total number of data samples in each cluster X_n, and $|D| = \sum_{n=1}^{N} |D_n|$ denotes the total number of data samples of all clients. At each iteration, t, the server performs data size-aware synchronous inter-cluster aggregation to aggregate the uploaded aggregated gradients from all cluster heads to update the global model w^t:

$$w^t = w^{t-1} - \eta \cdot \frac{\sum_{n=1}^{N} |D_n| \cdot \bar{g}_n^t}{|D|}, \tag{5}$$

where η is the learning rate.

After that, the server sends the new global model back to all cluster heads, and each cluster head broadcasts the new global model to clients participating in the intra-cluster aggregation of this iteration for their further local updating.

D. Convergence Analysis

To give the convergence analysis, we firstly assume that the loss function is μ-strongly convex and L-smooth.

Assumption 1. *Assume that the loss function f satisfies:*

1) f is μ-strongly convex, where $\mu \geq 0$, i.e. $f(y) - f(x) \geq \nabla \mathbf{f}^\top(x)(y-x) + \frac{\mu}{2}||y-x||^2, \forall x, y$.

2) f is L-smooth, where $L \geq 0$, i.e., $f(y) - f(x) \leq \nabla \mathbf{f}^\top(x)(y-x) + \frac{L}{2}||y-x||^2, \forall x, y$.

Assumption 2. *Assume that there exists at least one x^* to minimize the loss function $f(w)$.*

Many models with convex function such as LR and SVM can satisfy those assumptions above.

Assumption 3. *Assume that the variance of gradients at client i, $i \in 1, 2, \cdots, M$ with dataset D_i is bounded, i.e., $E||\nabla f(w;\xi) - \nabla F(w)||^2 \leq P$ $(P > 0)$, $\forall w \in R^d$ and $\forall \xi \in D_i$.*

Based on the mentioned assumptions, we prove the convergence of EAFL in two steps, including the convergence bound after Q local updates and the bound after T epochs.

Theorem 1. *Assume that the loss function F is μ-strongly convex and L-smooth, and each client executes Q local updates before reporting updates to the cluster head. When the following conditions are satisfied: 1) $\eta < \frac{1}{L}$, 2) $\mu \cdot \eta > 1 - \sqrt[Q]{\frac{1}{4N \cdot M}}$,* *3) $F(w^0) - F(w^*) > \frac{Q\cdot\eta\cdot P}{4(1-\mu\cdot\eta)^Q}$, the convergence bound of the global loss function after T epochs is:*

$$E[F(w^T) - F(w^*)] \leq [2N \cdot M \cdot (1-\mu\cdot\eta)^Q]^T \cdot [F(w^0 - F(w^*)] + \frac{[1-(2N \cdot M \cdot (1-\mu\cdot\eta)^Q)^T]Q\cdot\eta\cdot P}{4(1-\mu\cdot\eta)^Q} \tag{6}$$

where w^0 is the initial model parameter and w^* donates the optimal model to minimize the global loss function F.

Proof. Since some proof procedure can be found in previous work [35], we only propose the differences. In EAFL framework, the clustering is $X = \{X_1, X_2, \cdots, X_N\}$. After client i in an arbitrary cluster performs Q local updates based on the model $w^{T-\tau_i}$ with local dataset D_i to get local model $w_i^{T-\tau_i,Q}$. Obviously, $w_i^{T-\tau_i,0} = w^{T-\tau_i}$. The convergence bound is:

$$\begin{aligned} E[F(w_i^{T-\tau_i,Q}) - F(w^*)] &\leq (1-\mu\cdot\eta)^Q \\ \cdot[F(w_i^{T-\tau_i,0}) - F(w^*)] &+ \frac{Q\cdot\eta\cdot P}{2} \end{aligned} \tag{7}$$

After that, EAFL performs intra-aggregation and inter-aggregation in order based on Eq. (4) and Eq. (5) above. With the help of Eq. (7), as a result, the convergence bound of EAFL after T epoch is:

$$\begin{aligned} &E[F(w^T) - F(w^*)] \\ &\leq [2N \cdot M \cdot (1-\mu\cdot\eta)^Q] \cdot [F(w^{T-1}) - F(w^*)] \\ &+ \frac{N \cdot M \cdot Q \cdot \eta \cdot P}{2} \end{aligned} \tag{8}$$

Based on Eq. (8), we get the convergence bound after T epochs as shown Theorem 1.

■

VI. Evaluation

In this section, we evaluate the performance of EAFL on two commonly used datasets to validate its effectiveness.

A. Setup

Datasets. The experiments are conducted over two image datasets commonly used in FL experiments, i.e., MNIST [32] and CIFAR-10 [36]. For each dataset, we follow the previous works [37], [38] to simulate two kinds of Non-IID distribution. The first type (T1) allocates ϵ-proportion of the examples in an IID fashion and allocates the rest $(1-\epsilon)$-proportion in a sort-and-partition fashion. In the second type (T2), each client is only assigned data samples from a fixed L_{num} kinds of labels. ϵ and L_{num} indicate the degree of Non-IID. The smaller the value of ϵ or L_{num}, the higher the Non-IID degree.

Training settings. We train Convolutional Neural Networks (CNN). In experiments, there is 1 server and 100 clients in FL system. For MNIST, we deployed a CNN model containing two convolutional layers, both with ReLU activation layers. We use Stochastic Gradient Descent (SGD) as optimizer with learning rate $\eta = 0.005$ and the maximum iteration is $T = 1600$. We set the number of clusters $N = 5$, the re-clustering round interval $R = 100$. We adopt $\epsilon = 0.04$ in (T1) and $L_{num} = 1$ in (T2). In each iteration, clients with the percentage of $\varphi = 0.1$ participate. For CIFAR-10, we used the classical CNN model, LeNet [39]. We use SGD as optimizer with $\eta = 0.0005$, and $T = 10000$. We set $N = 15$ and $R = 1000$. We adopt $\epsilon = 0.04$ in (T1) and $L_{num} = 1$ in (T2). In each iteration of training, we allocate $\varphi = 0.2$.

Metrics. In experiments, we measure global model utility by the prediction accuracy on the test set.

Baselines. We compared EAFL with 3 state-of-the-art AFL algorithms focused on data or system heterogeneity.

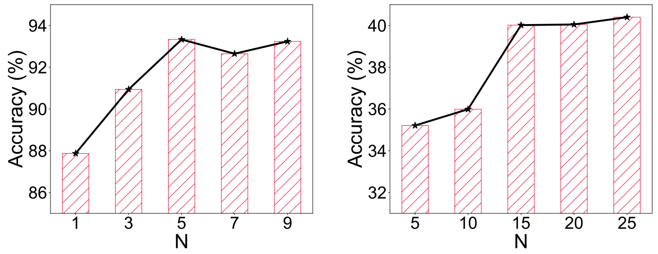
- TWAFL [9] is a SAFL algorithm, which computes the staleness-aware weight for each client i that participates in iteration j of global aggregation by: $S(\tau_i^t) = (\frac{e}{2})^{-\tau_i^t}$.
- GSGM [20] is an AFL algorithm that aims to deal with the data heterogeneity. It blocks fast clients to enforce them to wait for slow clients so that all clients with heterogeneous data distributions have chances to fully participate in training.
- WKAFL [8] is an AFL algorithm that aims to deal with the data heterogeneity. It first performs the staleness-aware aggregation among the fastest K local updates to estimate the unbiased global update, then it re-assigns weights to these local updates according to their distance to the unbiased global update. In this way, the contribution of local updates that have high staleness but have a similar update direction to the unbiased global update to the global model can be improved. Thus, the global model has a chance to learn more comprehensive information.

B. Evaluation on Parameters

In this section, we evaluate the effect of the number of clusters N and the re-clustering round interval R on EAFL. Both of them affect the trade-off between model utility and efficiency. We test the effect of a parameter by changing the value of the parameter while fixing the others.

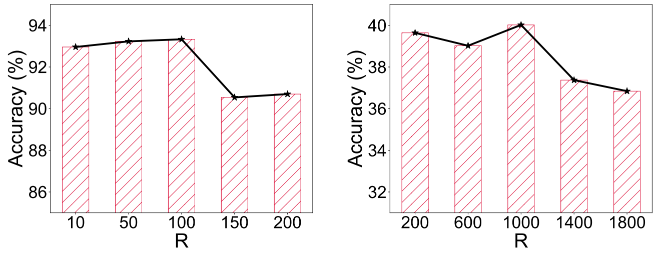
Effect of the number of clusters N. Fig. 4 shows the accuracy of the global model when N changes. We can observe that for both MNIST and CIFAR-10, the global model accuracy first increases and then almost settles down with the increase of N. The reason is as follows. When N is small, the number of clients in each cluster would be large, and the degree of heterogeneity within each cluster would be high. In this case, it is hard to ensure that the global model comprehensively learns information from clients with each kind of characteristic of data and system. When N increases to a certain value, clients within each cluster can be essentially homogeneous, thus achieving relatively stable global model performance. Considering that the more clusters, the more communication resources are needed to support the interactions between cluster heads and the server, we set $N = 5$ for MNIST and $N = 15$ for CIFAR-10 in the subsequent experiments.

Effect of the re-clustering round interval R. Fig. 5 shows the accuracy of the global model when R varies.



(a) MNIST (b) CIFAR-10

Fig. 4. Evaluation on parameter N.



(a) MNIST (b) CIFAR-10

Fig. 5. Evaluation on parameter R.

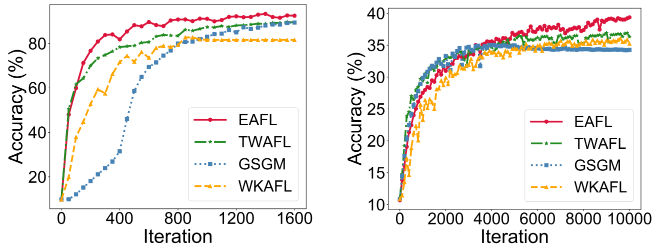
The experimental result indicates that for both MNIST and CIFAR-10, with the decrease of R, global model accuracy first increases and then remains essentially constant. The reason is as follow. We separate clients with similar staleness into the same cluster because clients with similar staleness are more likely to have similar system characteristics. As the semi-asynchronous aggregation within cluster proceeds, staleness of clients changes. If R is too large, re-clustering is not timely, thus clients within the same cluster do not have similar system characteristics. When R decreases to a certain value, the frequency of re-clustering can catch up with changes in intra-cluster staleness, so the global model performance stabilizes. Considering that if re-clustering is too frequent, it will affect the training efficiency, $R = 100$ for MNIST and $R = 1000$ for CIFAR-10 are chosen in the subsequent experiments.

C. Performance Evaluation

In this section, we first demonstrate the effect of each mechanism of the proposed EAFL, and then compare EAFL with baseline methods.

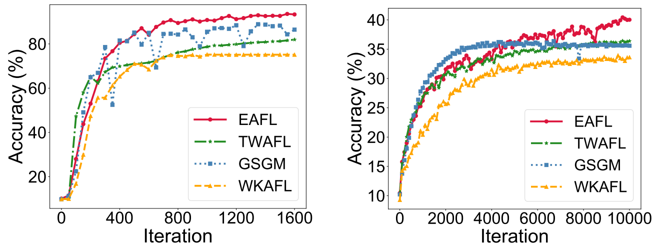
Effect of the proposed mechanisms. To verify the effectiveness of mechanisms consist of GDC, SAA and DSA, we conduct experiments of EAFL with and without GDC, SAA, and DSA, respectively. Among them, EAFL without GDC performs the clustering only once at the beginning of the training process. EAFL without SAA performs intra-cluster aggregation via averaging rather than staleness-aware weighted aggregation. EAFL without DSA aggregates updates uploaded by cluster heads in a synchronous manner rather than an asynchronous manner. The results are shown in Table. I.

We can see that when GDC is not performed, the global model accuracy decreases from 93.33% to 86.69% for MNIST dataset and decreases from 39.37% to 34.99% for CIFAR-10 dataset. Such results demonstrate the necessity of dynamic clustering and the effectiveness of the gradient similarity-based dynamic clustering mechanism.



(a) MNIST (b) CIFAR-10

Fig. 6. Performance comparison of EAFL and baseline methods under Non-IID setting T1. The Non-IID degree parameter $\epsilon = 0.04$.



(a) MNIST (b) CIFAR-10

Fig. 7. Performance comparison of EAFL and baseline methods under Non-IID setting T2. The Non-IID degree parameter $L_{num} = 1$.

TABLE I
THE RESULTS OF ABLATION EXPERIMENTS.

Dataset	Global Model Accuracy (%)			
	EAFL	EAFL w/o GDC	EAFL w/o SAA	EAFL w/o DSA
MNIST	93.33	86.69	67.02	71.67
CIFAR-10	39.37	34.99	34.74	30.78

It is shown that not performing SAA results in an accuracy drop of nearly 27% for the MNIST dataset and nearly 5% for the CIFAR-10 dataset. This is because staleness-aware aggregation in each cluster can help EAFL mitigate the impact of slow clients and their stale updates. Thus, we draw a conclusion that the adoption of the staleness-aware semi-asynchronous intra-cluster aggregation mechanism does improve the learning performance of EAFL.

We can also observe that DSA improves the global model accuracy for both MNIST and CIFAR-10 datasets. It proves that the data size-aware synchronous inter-cluster aggregation mechanism is useful in improving the learning performance of EAFL. It is certainly true since synchronous aggregation among clusters enables clients with each kind of data and system characteristics to fully participate in the global aggregation, thus making the global model learn comprehensively to achieve better performance.

Performance comparison under Non-IID settings. We compare the performance of EAFL with baseline methods (i.e., TWAFL, GSGM, and WKAFL) under two Non-IID settings with MNIST and CIFAR-10 datasets, and the results are shown in Fig. 6 and Fig. 7. We can find that under these two Non-IID settings, our proposed EAFL achieves better global model accuracy, which demonstrates the effectiveness of EAFL. This

TABLE II

Performance comparison when the degree of non-IID (T2) changes. The best accuracy is highlighted in bold.

Dataset	L_{num}	Global Model Accuracy (%)			
		EAFL	TWAFL	GSGM	WKAFL
MNIST	1	**93.33**	81.95	86.45	75.13
	5	**94.58**	91.67	91.23	92.88
	10	94.68	**95.66**	94.61	93.54
CIFAR10	1	**40.02**	36.39	35.59	33.63
	5	**39.27**	36.94	36.45	34.70
	10	**39.53**	38.19	38.56	35.48

TABLE III

Performance comparison when the percentage of clients participating in global aggregation changes. The best accuracy is highlighted in bold.

Dataset	φ	Global Model Accuracy (%)			
		EAFL	TWAFL	GSGM	WKAFL
MNIST	0.05	**92.78**	55.07	83.11	71.23
	0.10	**93.33**	81.95	86.45	75.13
	0.20	**93.58**	92.14	93.30	88.48
CIFAR10	0.10	**38.70**	31.46	33.89	29.20
	0.20	**40.02**	36.39	35.59	33.63
	0.30	**39.04**	37.47	33.86	34.74

is because our clustering and two-stage aggregation-based method can ensure that clients with all kinds of data and system characteristics efficiently and fully participate in each iteration of the training of the global model. However, TWAFL only proposes weighted aggregation to solve the staleness problem caused by system heterogeneity but does not consider data heterogeneity, GSGM can only increase the opportunity for each data distribution to participate in training but does not adequately address the problem of staleness, WKAFL can not guarantee that clients with slow training but unique data distribution are fully engaged in training.

Performance comparison when the Non-IID degree changes. We compare EAFL with baseline methods under Non-IID setting (T2) with $L_{num} = 1, 5, 10$ respectively, and the results are shown in Table. II. We can observe that as L_{num} increases, the accuracy of all three baseline methods increases since with more labeled data available on each client, local model drift becomes minor, and clients update in a more consistent direction, i.e. local updates can better represent global information. EAFL maintains a high and stable accuracy on both MNIST and CIFAR-10 across different L_{num} settings, while the baseline methods degrade significantly when L_{num} is small, which demonstrates that EAFL is less affected by data heterogeneity and has robustness. EAFL outperforms the baseline methods at different levels of Non-IID settings because we learn the global information more comprehensively by leveraging clients with different data and system characteristics during training while the baseline methods do not learn comprehensively. We note that our accuracy is not as good as TWAFL when $L_{num} = 10$. This is because in this setting, each client has data with all 10 labels, and it is difficult to have a situation where the clients train slowly but their data is unique, since the data heterogeneity is low at this point. EAFL needs to ensure that the slow-training clients are also fully involved in the training, which inevitably suffers from staleness. In contrast, TWAFL can get better model utility by selecting only fast-training clients.

Performance comparison when the percentage of participated clients changes. We compare the performance of EAFL with baseline methods when the percentage of participated clients in each global iteration changes, and the results are shown in Table. III. We set three different φ, for MNIST is $0.05, 0.10, 0.20$ and for CIFAR-10 is $0.10, 0.20, 0.30$. As φ increases, the accuracy of all three baseline methods increases generally. This is because 1) previous work [8] has shown that the minimum average of staleness approximately equals $\frac{1}{2\varphi}$, which means staleness of clients consequently decreases so that the global model is less negatively affected by the high-stale updates and 2) with more clients participating in each iteration of training, server can learn more global information comprehensively. EAFL maintains a high and stable accuracy on both MNIST and CIFAR-10 with three different φ settings, while the baseline methods suffer significant degradation when φ is small. This represents that EAFL is less affected by the percentage of participated clients and has robustness. EAFL outperforms the baseline methods when the percentage of participated clients changes since staleness-aware weighted aggregation is taken within clusters to avoid being affected by high-stale updates while leveraging various updates.

VII. Conclusion

In this paper, we proposed a clustering and two-stage aggregation-based Efficient Asynchronous Federated Learning (EAFL) framework, to cope with system and data heterogeneity and improve global model utility in heterogeneous edge environments. We put forward a gradient similarity-based dynamic clustering mechanism to cluster edge devices with similar system and data characteristics together dynamically during the training process so as to adaptively leverage various updates. Meanwhile, we designed a novel two-stage aggregation consisting of staleness-aware semi-asynchronous intra-cluster aggregation and data size-aware synchronous inter-cluster aggregation, to fully utilize the global information without being encumbered with slow devices. The experiment results demonstrated that EAFL outperforms the existing state-of-the-art methods and can be a valuable solution for efficient collaborative training in heterogeneous edge environments.

VIII. Acknowledgments

This work was supported in part by National Key R&D Program of China (Grant No. 2021ZD0112803), National Natural Science Foundation of China (Grants No. 62122066, U20A20182, 61872274, 62102337), Key R&D Program of Zhejiang (Grant No. 2022C01018) and Natural Science Foundation of Hunan Province, China (No. 2023JJ40174).

References

[1] S. Singh, S. Rathore, O. Alfarraj, A. Tolba, and B. Yoon, “A framework for privacy-preservation of iot healthcare data using federated learning and blockchain technology,” *Future Generation Computer Systems*, vol. 129, pp. 380–388, 2022.

[2] P. Sun, H. Che, Z. Wang, Y. Wang, T. Wang, L. Wu, and H. Shao, “Pain-fl: Personalized privacy-preserving incentive for federated learning,” *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3805–3820, 2021.

[3] Z. Zhang, L. Wu, C. Ma, J. Li, J. Wang, Q. Wang, and S. Yu, “Lsfl: A lightweight and secure federated learning scheme for edge computing,” *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 365–379, 2022.

[4] J. Hu, Z. Wang, Y. Shen, B. Lin, P. Sun, X. Pang, J. Liu, and K. Ren, “Shield against gradient leakage attacks: Adaptive privacy-preserving federated learning,” *IEEE/ACM Transactions on Networking*, 2023, doi: 10.1109/TNET.2023.3317870.

[5] X. Pang, Z. Wang, Z. He, P. Sun, M. Luo, J. Ren, and K. Ren, “Towards class-balanced privacy preserving heterogeneous model aggregation,” *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 3, pp. 2421–2432, 2023.

[6] Z. Wang, K. Liu, J. Hu, J. Ren, H. Guo, and W. Yuan, “Attrleaks on the edge: Exploiting information leakage from privacy-preserving co-inference,” *Chinese Journal of Electronics*, vol. 32, no. 1, pp. 1–12, 2023.

[7] C. Xie, S. Koyejo, and I. Gupta, “Asynchronous federated optimization,” *arXiv preprint arXiv:1903.03934*, 2019, doi: 10.48550/arXiv.1903.03934.

[8] Z. Zhou, Y. Li, X. Ren, and S. Yang, “Towards efficient and stable k-asynchronous federated learning with unbounded stale gradients on non-iid data,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 12, pp. 3291–3305, 2022.

[9] Y. Chen, X. Sun, and Y. Jin, “Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation,” *IEEE transactions on neural networks and learning systems*, vol. 31, no. 10, pp. 4229–4238, 2019.

[10] G. Shi, L. Li, J. Wang, W. Chen, K. Ye, and C. Xu, “Hysync: Hybrid federated learning with effective synchronization,” in *2020 IEEE 22nd International Conference on High Performance Computing and Communications; (HPCC)*. IEEE, 2020, pp. 628–633.

[11] C. Zhou, H. Tian, H. Zhang, J. Zhang, M. Dong, and J. Jia, “Tea-fed: time-efficient asynchronous federated learning for edge computing,” in *Proceedings of the 18th ACM International Conference on Computing Frontiers*, 2021, pp. 30–37.

[12] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, “Advances and open problems in federated learning,” *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[13] Q. Li, Y. Diao, Q. Chen, and B. He, “Federated learning on non-iid data silos: An experimental study,” in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 2022, pp. 965–978.

[14] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, “Federated optimization in heterogeneous networks,” *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

[15] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, “Federated learning with non-iid data,” *arXiv preprint arXiv:1806.00582*, 2018, doi: 10.48550/arXiv.1806.00582.

[16] N. Yoshida, T. Nishio, M. Morikura, K. Yamamoto, and R. Yonetani, “Hybrid-fl for wireless networks: Cooperative learning mechanism using non-iid data,” in *ICC 2020-2020 IEEE International Conference On Communications (ICC)*. IEEE, 2020, pp. 1–7.

[17] Z. Li, J. Shao, Y. Mao, J. H. Wang, and J. Zhang, “Federated learning with gan-based data synthesis for non-iid clients,” in *International Workshop on Trustworthy Federated Learning*. Springer, 2022, pp. 17–32.

[18] Z. Wang, S. Duan, C. Wu, W. Lin, X. Zha, P. Han, and C. Liu, “Generative data augmentation for non-iid problem in decentralized clinical machine learning,” in *2022 4th International Conference on Data Intelligence and Security (ICDIS)*. IEEE, 2022, pp. 336–343.

[19] N. H. Quyen, P. T. Duy, N. C. Vy, D. T. T. Hien, and V.-H. Pham, “Federated intrusion detection on non-iid data for iiot networks using generative adversarial networks and reinforcement learning,” in *International Conference on Information Security Practice and Experience*. Springer, 2022, pp. 364–381.

[20] H. Wang, R. Li, C. Li, P. Zhou, Y. Li, W. Xu, and S. Guo, “Gradient scheduling with global momentum for asynchronous federated learning in edge environment,” *IEEE Internet of Things Journal*, vol. 9, no. 19, pp. 18 817–18 828, 2022.

[21] W. Dai, Y. Zhou, N. Dong, H. Zhang, and E. P. Xing, “Toward understanding the impact of staleness in distributed machine learning,” *arXiv preprint arXiv:1810.03264*, 2018, doi: 10.48550/arXiv.1810.03264.

[22] L. You, S. Liu, Y. Chang, and C. Yuen, “A triple-step asynchronous federated learning mechanism for client activation, interaction optimization, and aggregation enhancement,” *IEEE Internet of Things Journal*, vol. 9, no. 23, pp. 24 199–24 211, 2022.

[23] Q. Wang, Q. Yang, S. He, Z. Shi, and J. Chen, “Asyncfeded: Asynchronous federated learning with euclidean distance based adaptive weight aggregation,” *arXiv preprint arXiv:2205.13797*, 2022, doi: 10.48550/arXiv.2205.13797.

[24] J.-w. Lee, J. Oh, Y. Shin, J.-G. Lee, and S.-Y. Yoon, “Accurate and fast federated learning via iid and communication-aware grouping,” *arXiv preprint arXiv:2012.04857*, 2020, doi: 10.48550/arXiv.2012.04857.

[25] P. Tian, W. Liao, W. Yu, and E. Blasch, “Wscc: A weight-similarity-based client clustering approach for non-iid federated learning,” *IEEE Internet of Things Journal*, vol. 9, no. 20, pp. 20 243–20 256, 2022.

[26] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, “An efficient framework for clustered federated learning,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 19 586–19 597, 2020.

[27] Z. Wang, H. Xu, J. Liu, H. Huang, C. Qiao, and Y. Zhao, “Resource-efficient federated learning with hierarchical aggregation in edge computing,” in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[28] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, “Federated learning: Strategies for improving communication efficiency,” *arXiv preprint arXiv:1610.05492*, 2016, doi: 10.48550/arXiv.1610.05492.

[29] P. Sun, X. Chen, G. Liao, and J. Huang, “A profit-maximizing model marketplace with differentially private federated learning,” in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 2022, pp. 1439–1448.

[30] B. Zhao, P. Sun, T. Wang, and K. Jiang, “Fedinv: Byzantine-robust federated learning by inversing local model updates,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 8, 2022, pp. 9171–9179.

[31] Z. Wang, W. Yuan, X. Pang, J. Li, and H. Shao, “Towards task-free privacy-preserving data collection,” *China Communications*, vol. 19, no. 7, pp. 310–323, 2022.

[32] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, “Emnist: Extending mnist to handwritten letters,” in *2017 international joint conference on neural networks (IJCNN)*. IEEE, 2017, pp. 2921–2926.

[33] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[34] J. MacQueen *et al.*, “Some methods for classification and analysis of multivariate observations,” in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14. Oakland, CA, USA, 1967, pp. 281–297.

[35] J. Liu, H. Xu, Y. Xu, Z. Ma, Z. Wang, C. Qian, and H. Huang, “Communication-efficient asynchronous federated learning in resource-constrained edge computing,” *Computer Networks*, vol. 199, p. 108429, 2021.

[36] A. Krizhevsky, G. Hinton *et al.*, “Learning multiple layers of features from tiny images,” 2009.

[37] J. Xu, S.-L. Huang, L. Song, and T. Lan, “Byzantine-robust federated learning through collaborative malicious gradient filtering,” in *2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2022, pp. 1223–1235.

[38] H. Zhu, J. Xu, S. Liu, and Y. Jin, “Federated learning on non-iid data: A survey,” *Neurocomputing*, vol. 465, pp. 371–390, 2021.

[39] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.