Asynchronous Federated Clustering with Unknown Number of Clusters

Yunfan Zhang[1], Yiqun Zhang[1,4,*], Yang Lu[2], Mengke Li[3], Xi Chen[4], Yiu-ming Cheung[4]

[1]School of Computer Science and Technology, Guangdong University of Technology, Guangzhou, China
[2]Fujian Key Laboratory of Sensing and Computing for Smart City, School of Informatics, Xiamen University, Xiamen, China
[3]College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China
[4]Department of Computer Science, Hong Kong Baptist University, Hong Kong SAR, China
{yfzhang, yqzhang}@gdut.edu.cn, luyang@xmu.edu.cn, csmengkeli@gmail.com, {csxchen, ymc}@comp.hkbu.edu.hk

Abstract

Federated Clustering (FC) is crucial to mining knowledge from unlabeled non-Independent Identically Distributed (non-IID) data provided by multiple clients while preserving their privacy. Most existing attempts learn cluster distributions at local clients, then securely pass the desensitized information to the server for aggregation. However, some tricky but common FC problems are still relatively unexplored, including the heterogeneity in terms of clients' communication capacity and the unknown number of proper clusters. To further bridge the gap between FC and real application scenarios, this paper first shows that the clients' communication asynchrony and unknown proper cluster numbers are complex coupling problems, and then proposes an Asynchronous Federated Cluster Learning (AFCL) method accordingly. It spreads the excessive number of seed points to clients as a learning medium and coordinates them across clients to form a consensus. To alleviate the distribution imbalance cumulated due to the unforeseen asynchronous uploading from the heterogeneous clients, we also design a balancing mechanism for seeds updating. As a result, the seeds gradually adapt to each other to reveal a proper number of clusters. Extensive experiments demonstrate the efficacy of AFCL.

Code — https://github.com/Yunfan-Zhang/AFCL

Introduction

Federated Learning (FL) is common in implementing distributed machine learning while preserving privacy (Banabilah et al. 2022; Zhang et al. 2021; Zhou et al. 2023). In unsupervised FL tasks, Federated Clustering (FC) that partitions a dataset into compact object clusters demonstrates great potential in mining data concepts and knowledge (Ma et al. 2019; Nelus, Glitza, and Martin 2021; Chung, Lee, and Ramchandran 2022). However, without label guidance, FC faces significant challenges brought by the privacy protection requirements and non-IID of heterogeneous clients.

Most existing methods address FC by first letting clients learn cluster distributions and then passing the privacy-protected cluster knowledge to the server for global aggregation (Ghosh et al. 2020; Kumar, Karthik, and Nair 2020). For example, k-FED (Dennis, Li, and Smith 2021) explores global cluster distributions with a higher security level through one-shot aggregation of the non-IID distributions learned by the clients. Two independent works, F-FCM (Pedrycz 2022) and FFCM (Stallmann and Wilbik 2022), share similar names and principles, and adopt fuzzy-c-means as their local clustering algorithm. Since the fuzzy object-cluster affiliation can more finely reflect the partition information of data objects, the information loss caused by the privacy constraints of FL can be considerably offset.

Notably, most existing works overlook the common problem of client asynchrony[1] attributed to the divergence of clients' communication capabilities. Due to the lack of data

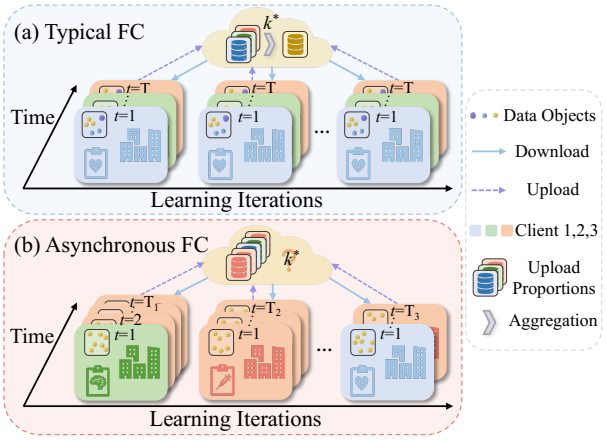


Figure 1: **AFCL (ours) vs. typical FC.** (a) Existing FC approaches typically assume that the clients are synchronous and the optimal k^* is known by both the clients and server. By contrast, (b) AFCL learns under a more realistic scenario that the clients can upload distribution information of completely non-overlapping and non-uniform numbers of clusters with unforeseen and imbalanced frequencies.

[1]The terms "asynchrony" and "asynchronous communication" in this paper indicate the non-uniform participation rates of different clients in each round of communication. Since the goal of FC is to aggregate the distributions of clients at the server, the asynchronous problem addressed in this work is conceptually different from the model update asynchronous problem in supervised FL.

*Coresponding Author

labels, such a problem can severely bias the learning towards the clients that upload their distributions more frequently. Accordingly, a recently proposed federated spectral clustering method (Qiao, Ding, and Fan 2024) copes with the asynchronous scenario by aggregating intermediate variables to construct a global similarity matrix, which is robust to the communication frequency bias. However, it still relies on the naive assumption that the true number of clusters k^* is known in advance, which limits the application domain of FC in many real applications with unforeseen k^*.

Automatic determination of the optimal k^* has been an attractive clustering research topic in recent decades. Silhouette Coefficient (Rousseeuw 1987) determines k^* by considering the intra-compactness and the inter-dispersion of clusters. Density-based clustering (Schubert et al. 2017; Zhang et al. 2025; Peng et al. 2025) performs cluster exploration by considering the distribution connection of objects, and can select k^* according to the quality of cluster partition. More advanced learning-based approaches (Ahalt et al. 1990; Cheung, Y.-m. 2005; Jia, Cheung, and Liu 2014; Cai et al. 2024; Zou et al. 2024) have been proposed to learn k^* by letting seeds compete or cooperate to eliminate redundant low-quality clusters. Recently, significance-based clustering (Hu et al. 2022, 2025) has been proposed using gap statistics to estimate k^*. Nevertheless, their learning process depends on detailed and sufficient data statistics, preventing them from being utilized in FC. Therefore, the absence of k^* brings difficulties to FC due to the lack of clustering guidance, and also incurs additional k^* learning objective, collectively making asynchronous FC a challenging task.

A more detailed comparison of asynchronous FC and typical FC are illustrated in Figure. 1. Unlike typical FC where the clients communicate synchronously with known k^*, our focused asynchronous FC poses the problem of learning clusters and cluster numbers from the uploaded complex distributions caused by the non-uniform communication of clients. More specifically, the difficulties of asynchronous FC lie in the cross-coupled k^* learning and asynchronous uploading of clients. That is, distributions learned at different clients are not associated with a uniform k^*, while the server struggles in the learning of k^* due to the asynchronously uploaded unreliable local distributions.

Therefore, we propose the Asynchronous Federated Cluster Learning (AFCL) method that can learn an optimal number of clusters with the asynchronously communicated clients in a self-adaptive way. It uniformly generates seed points for the clients, and accumulates the distribution information of their surrounding objects indicated by the difference between the seed and objects within each client to capture the clients' own distributions. Then the accumulated information is passed to the server to update the seeds for client-to-seed distribution information fusion. To gain a consensus among the clients, we let the neighboring seeds share their update intensity to achieve seed-to-seed information completion, as the non-IID clients may provide only a partial global distribution. A balancing mechanism has also been developed to evaluate and adjust the update intensity accumulated from different clients, to relieve the potential bias caused by their asynchronous participation. As a result, AFCL can automatically converge redundant neighboring seeds to learn an appropriate number of clusters under the challenging asynchronous federated scenario. Extensive experiments demonstrate the effectiveness of AFCL. The main contributions of this work are summarized into three points:

- We propose a new FC approach to learn a distribution consensus from asynchronously communicated clients without requiring the 'true' number of clusters. It serves to enhance the robustness and universality of FL.
- This paper first considers and attempts to solve a more realistic but challenging non-IID case in FC, i.e., a global cluster could be composed of completely non-overlapping sub-clusters belonging to different clients.
- A balancing mechanism is developed to allocate the contribution of clients with heterogeneous communication modes during the interactive learning on the server, even if they participate in the learning by only one-shot.

Related Work

Federated Clustering

An one-shot FC approach called k-FED (Dennis, Li, and Smith 2021) has been developed to relieve the leakage of information during communication, while FedKKM (Zhou and Wang 2022) has been proposed to improve communication efficiency by using a novel Lanczos algorithm in its distributed matrices. Meanwhile, F-FCM (Pedrycz 2022), and FFCM (Stallmann and Wilbik 2022) utilized fuzzy clustering techniques to enhance privacy protection by only transmitting object-cluster affiliation. Furthermore, A multi-view FC approach (Hu et al. 2023) has been proposed to extend multi-view clustering into the federated scenario by designing a strategy of consensus prototype learning. However, they employ clustering techniques under the assumption that the true cluster numbers are known by the clients and server in advance. Recently, an FC framework VKMC (Huang et al. 2022) has been proposed to improve the vertical FL based on coresets, while a density-based FC method HFDPC (Ding et al. 2023) has been proposed for improving the effectiveness of data partitioning by introducing a similar density chain. Most recently, Federated Subspace Clustering (Fed-SC) (Xie, Wu, and Liao 2023) and Federated Spectral Clustering (FedSC) (Qiao, Ding, and Fan 2024) have been proposed to address the FC of high-dimensional and noisy data, respectively. Although some of the above-mentioned FC methods have considered the non-IID or the asynchrony issues of clients in FC, most of their solutions heavily depend on the availability of the 'true' number of clusters k^*, which hinders their applications in real complex scenarios.

Clustering with Unknown Cluster Number

The more realistic unsupervised or weakly supervised learning has attracted much attention in recent years, especially for some significant application domains (Cheung and Zeng 2009; Wu et al. 2019; Zhang, Cheung, and Tan 2020; Wang et al. 2023). Clustering is a key unsupervised learning technique, where the traditional clustering methods determine the optimal number of clusters k^* manually (Rousseeuw

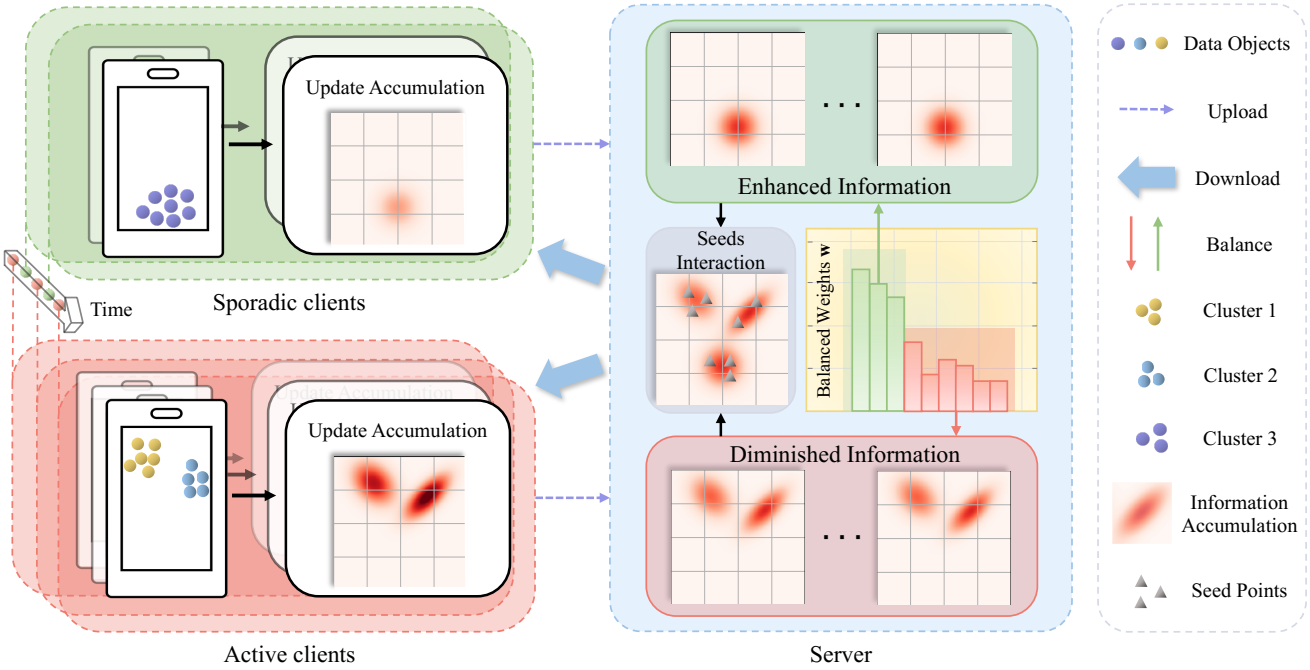


Figure 2: Overview of the proposed AFCL framework. Initialized seed points accumulate update intensity from different clients independently, then the server balances the update information to facilitate appropriate seeds interaction for fusing the clients' distributions. The heat map represents the intensity of update information of seeds accumulated from asynchronous clients.

1987; Thorndike 1953). To realize automated k^* selection, density-based clustering (Ester et al. 1996; Zhang et al. 2025; Peng et al. 2025) have been proposed to automatically determine k^* at a "knee point" during their cluster exploration. Recently, more advanced learning-based approaches (Cheung 2004; Cheung and Jia 2013; Cai et al. 2024) introduce a cooperative or competitive mechanism to excessive cluster centers. They simultaneously ensure the comprehensive representation of object distributions and make the elimination of redundant cluster centers learnable, thus achieving satisfactory clustering performance. Most recently, significance-based approaches (Hu et al. 2022, 2025) have been proposed to rigorously judge the significance of cluster distributions under the current k. Nevertheless, all the above-mentioned solutions require detailed statistics of the entire dataset, which hamper their applicability in FC.

Proposed Method

In this section, we first define the asynchronous FC problem, and then present the proposed AFCL algorithm composed of two key technical components: 1) CSUA: Client-Side Update Accumulation, and 2) SSSI: Server-Side Seeds Interaction. Frequently used notations are summarized in Table 1, and the overview of AFCL is shown in Figure. 2.

Problem Formulation

Assuming a federated network with p clients dividing the entire dataset $\mathbf{X}$ into the corresponding subsets $\{\mathbf{X}^{\{1\}}, \mathbf{X}^{\{2\}}, ..., \mathbf{X}^{\{g\}}, ..., \mathbf{X}^{\{p\}}\}$, where the subset of g-th client $\mathbf{X}^{\{g\}}$ has $n^{\{g\}}$ objects $\{\mathbf{x}_1^{\{g\}}, \mathbf{x}_2^{\{g\}}, ..., \mathbf{x}_{n^{\{g\}}}^{\{g\}}\}$ with $\sum_{g=1}^{p} n^{\{g\}} = n$, and each object $\mathbf{x}_i^{\{g\}} = [x_{i,1}^{\{g\}}, x_{i,2}^{\{g\}}, ..., x_{i,d}^{\{g\}}]^\top$ is a d-dimensional vector. We use a matrix $\mathbf{Q} \in \mathbb{R}^{n \times k}$ to indicate the object-cluster affiliation, and the conventional clustering objective is to minimize the overall intra-cluster dissimilarity between the objects and the cluster center (also called seed point or seed interchangeably hereinafter), which can be written as:

$$Z(\mathbf{Q}, \mathbf{M}) = \sum_{i=1}^{n} \sum_{l=1}^{k} q_{i,l} \Phi(\mathbf{x}_i, \mathbf{m}_l), \tag{1}$$

where $\Phi(\mathbf{x}_i, \mathbf{m}_l)$ denotes the Euclidean distance between i-th object $\mathbf{x}_i$ and l-th seed $\mathbf{m}_l$. All the k seeds can be organized as a matrix $\mathbf{M} \in \mathbb{R}^{d \times k}$ and $q_{i,l}$ is the (i,l)-th entry of $\mathbf{Q}$ satisfying $\sum_{l=1}^{k} q_{i,l} = 1$ and $q_{i,l} \in \{0, 1\}$. For federated

Notations	Explanations
$\mathbf{X}$	Global dataset
$\mathbf{X}^{\{g\}}$	Dataset of g-th client
$\mathbf{M}^{\{g\}}$	Seed points of g-th client
$\mathbf{m}_l$	l-th global seed point
$\mathbf{Q}^{\{g\}}$	Object-cluster affiliation matrix corresponding to g-th client
$\mathbf{B}^{\{g\}}$	Cluster center set of g-th client
$\mathbf{R}_l^{\{g\}}$	Update intensity of l-th seed on g-th client
k^*	The 'true' global cluster number of $\mathbf{X}$

Table 1: Summary of notations.

clustering, each g-th client can perform the above clustering locally on $\mathbf{X}^{\{g\}}$, and the ultimate goal is to minimize the objective function at the server with the entire dataset $\mathbf{X}$ and the consensus seed points $\mathbf{M}$ learned across all the clients.

CSUA: Client-Side Update Accumulation

To protect the privacy of clients while transmitting their distribution information to the server for global clustering, some intermediate quantities, e.g., update intensity of seeds, clustering center, and intra-cluster dissimilarity will be extracted by performing local clustering on each client and then uploaded to the server for seeds interaction.

For each object $\mathbf{x}_i^{\{g\}}$ in g-th client, its belonging cluster is determined by:

$$q_{i,l} = \begin{cases} 1, & \text{if } l = \arg\min_r \gamma_r \parallel \mathbf{x}_i^{\{g\}} - \mathbf{m}_r^{\{g\}} \parallel^2 \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

where $\mathbf{m}_r^{\{g\}}$ is the r-th seed in $\mathbf{M}^{\{g\}}$, and $\gamma_r^{\{g\}}$ is the weight of $\mathbf{m}_r^{\{g\}}$ computed by:

$$\gamma_r^{\{g\}} = \frac{s_r^{\{g\}}}{\sum_{l=1}^{k} s_l^{\{g\}}}. \tag{3}$$

Here, $s_r^{\{g\}}$ denotes the winning time of $\mathbf{m}_r^{\{g\}}$ within a single iteration, which is updated by:

$$s_c^{\{g\}} = s_c^{\{g\}} + 1, \tag{4}$$

for each $q_{i,c} = 1$, $i \in \{1, 2, ..., n^{\{g\}}\}$.

To facilitate the interaction of seeds across clients, we also compute the update intensity of seeds locally, but suspend the update until they are uploaded to the server. For the winner seed $\mathbf{m}_c^{\{g\}}$, its update intensity $\mathbf{r}_{c,i}^{\{g\}}$ contributed by the object $\mathbf{x}_i^{\{g\}}$ is computed by:

$$\mathbf{r}_{c,i}^{\{g\}} = \eta(\mathbf{x}_i^{\{g\}} - \mathbf{m}_c^{\{g\}}), \tag{5}$$

where $\mathbf{r}_{c,i}^{\{g\}} \in \mathbf{R}_c^{\{g\}}$ and η is the learning rate. By computing the update intensity of k seeds provided by all the $n^{\{g\}}$ objects, we obtain $R^{\{g\}} = \{\mathbf{R}_1^{\{g\}}, \mathbf{R}_2^{\{g\}}, ..., \mathbf{R}_k^{\{g\}}\}$ to upload to the server.

Remark 1. ***Privacy Protection w.r.t.*** $R^{\{g\}}$***:*** *AFCL uploads $R^{\{g\}}$s to the server to facilitate the interaction among seeds for the fusion of clients' information. Since the update intensity of a r-th seed provided by each of the objects treating it as a winner (i.e., $\mathbf{R}_r^{\{g\}}$) will be uploaded, the risk of objects recovery and privacy leakage will be increased. Therefore, existing privacy-preserving techniques such as holomorphic encryption (Acar et al. 2018) and differential privacy (Wei et al. 2020; Li et al. 2023) can be incorporated in some scenarios with high privacy protection requirements. Specifically, these techniques can be utilized to perturb the update intensity provided by each object in $R^{\{g\}}$ while ensuring that the radius of cooperative seeds selection in Eq.* (13) *and the overall update of seeds in Eq.* (14) *unchanged. Note that this paper focuses on more robust FC, rather than improving its privacy protection level.*

To judge convergence at the server, we also compute the centers $\mathbf{B}^{\{g\}} = \{\mathbf{b}_1^{\{g\}}, \mathbf{b}_2^{\{g\}}, ..., \mathbf{b}_k^{\{g\}}\}$ of the k clusters partitioned by the seeds by:

$$\mathbf{b}_r^{\{g\}} = \frac{1}{o_r^{\{g\}}} \sum_{i=1}^{n^{\{g\}}} q_{i,r} \mathbf{x}_i^{\{g\}}, \tag{6}$$

where $o_r^{\{g\}}$ is the number of objects in the r-th cluster corresponding to $\mathbf{m}_r^{\{g\}}$. Based on $\mathbf{B}^{\{g\}}$, the contribution of the r-th cluster to the global objective Z can be computed by:

$$z_r^{\{g\}} = \sum_{i=1}^{n^{\{g\}}} q_{i,r} \parallel \mathbf{b}_r^{\{g\}} - \mathbf{x}_i^{\{g\}} \parallel^2, \tag{7}$$

and the contributions from all the seeds in g-th client can be collectively denoted as $\mathbf{z}^{\{g\}} = [z_1^{\{g\}}, z_2^{\{g\}}, ..., z_k^{\{g\}}]^\top$.

Remark 2. ***Necessity of Intermediate*** $\mathbf{B}^{\{g\}}$ ***and*** $\mathbf{z}^{\{g\}}$***:*** *Since the trajectories of seeds $\mathbf{M}$ is usually complex as shown in Figure. 3(d), directly calculating the objective function based on seeds may lead to constant fluctuations in the objective function value. Therefore, intermediate values $\mathbf{B}^{\{g\}}$ and $\mathbf{z}^{\{g\}}$ computed locally on each client are relatively stable and can timely reflect the goodness of current seeds in terms of each client, which are helpful to obtain a smooth and more approximate overall objective function value Z.*

SSSI: Server-Side Seeds Interaction

During the learning, communication frequencies of each client should be recorded as $\mathbf{\Theta} = [\theta^{\{1\}}, \theta^{\{2\}}, .., \theta^{\{p\}}]^\top$. Accordingly, weights $\mathbf{w} = [w^{\{1\}}, w^{\{2\}}, ..., w^{\{p\}}]^\top$ for balancing the seeds update bias caused by the asynchronous communication should be maintained. For g-th participating client, its balance weight is computed by:

$$w^{\{g\}} = \frac{\xi}{\xi + \frac{\theta^{\{g\}}}{\sum_{j=1}^{p} \theta^{\{j\}}}}, \tag{8}$$

where ξ is a hyper-parameter controlling the sensitivity of balance weight w.r.t. the communication frequencies $\mathbf{\Theta}$. Intuitively, a larger ξ makes the balance weight less sensitive to the frequency, and a client with a higher frequency will have a lower weight to weaken its contribution.

Upon receiving the uploading from clients, the server initially aggregates the cluster centers and their objective contributions with the balance weights by:

$$\mathbf{b}_r = \sum_{j=1}^{\bar{p}} \frac{w^{\{j\}} o_r^{\{j\}} \mathbf{b}_r^{\{j\}}}{\sum_{j=1}^{\bar{p}} o_r^{\{j\}}}, \quad z_r = \sum_{j=1}^{\bar{p}} \frac{w^{\{j\}} o_r^{\{j\}} z_r^{\{j\}}}{\sum_{j=1}^{\bar{p}} o_r^{\{j\}}}, \tag{9}$$

where $\bar{p}$ denotes the number of participating clients. $\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_k\}$ and $\mathbf{z} = [z_1, z_2, ..., z_k]^\top$ represent the aggregated centers and contributions to the objective Z from the server perspective. Accordingly, an approximation of the objective on the server can be derived from Eq. (1) as:

$$Z(\mathbf{B}, \mathbf{z}) = \frac{1}{k} \sum_{l=1}^{k} \max_{r \neq l} \left(\frac{z_l + z_r}{\parallel \mathbf{b}_l - \mathbf{b}_r \parallel^2} \right), \tag{10}$$

which is in the form of the DBI index (Ros, Riad, and Guillaume 2023) that simultaneously reflects cluster compactness (numerator) and clusters' dispersion (denominator).

To ensure that the seeds learned from data can consensually minimize Z, a cooperative set:

$$C_r = \{\mathbf{m}_l \mid\mid \mathbf{m}_r - \mathbf{m}_l \parallel^2 \leq \parallel \mathbf{m}_r - \mathbf{x}_i^{\{g\}} \parallel^2\} \tag{11}$$

is identified for each seed $\mathbf{m}_r$, and we let all the seeds in C_r collectively receive the updates from the samples as:

$$\mathbf{m}_u = \mathbf{m}_u + \eta(\mathbf{x}_i^{\{g\}} - \mathbf{m}_u), \tag{12}$$

where $\mathbf{m}_u \in C_r$. However, since original samples are unavailable at the server due to privacy constraints, we compute Eqs. (11) and (12) alternatively based on the update intensity $\mathbf{r}_{r,i}^{\{g\}}$ that has been uploaded to the server as:

$$C_r = \{\mathbf{m}_l \mid\mid \mathbf{m}_r - \mathbf{m}_l \parallel^2 \leq \parallel \frac{w^{\{g\}}\mathbf{r}_{r,i}^{\{g\}}}{\eta} \parallel^2\} \tag{13}$$

and

$$\mathbf{m}_u = \mathbf{m}_u + w^{\{g\}}\mathbf{r}_{r,i}^{\{g\}} + w^{\{g\}}\eta(\mathbf{m}_r - \mathbf{m}_u) \tag{14}$$

respectively, where $w^{\{g\}}$ is used to mitigate the bias of seeds update caused by the asynchronous uploading. Intuitively, a smaller $w^{\{g\}}$ corresponds to a client that uploads frequently. Thus its neighbor identifying radius $\parallel w^{\{g\}}\mathbf{r}_{r,i}^{\{g\}}/\eta \parallel^2$ in Eq. (13) will be smaller to avoid over-update of the seeds in C_r. In Eq. (14), $\mathbf{r}_{r,i}^{\{g\}}$ from Eq. (5) is already with the learning rate η, and thus its latter two terms are with the same coefficients $w^{\{g\}}\eta$, equivalent to update $\mathbf{m}_u$ by a small step towards the data object that yields the update intensity $\mathbf{r}_{r,i}^{\{g\}}$.

Remark 3. ***Interaction of Seeds:*** *According to Eqs.* (13) *and* (14)*, all seeds in C_r will move closer to the cluster distribution represented by $\mathbf{m}_r$ according to the distribution information accumulated on different clients, which facilitates the distribution completion across different clients.*

Overall AFCL Algorithm

In the AFCL algorithm, after the global initialization of the seed points and local update accumulation on all the clients, sufficient interaction among the clients through the server is iteratively performed until the convergence of Z. In this process (summarized as Algorithm 1), the client-side (indicated by pink color) mainly implements cluster distribution learning, and the server-side (indicated by blue color) is responsible for privacy-protected distribution information fusion. For each learning iteration of the AFCL algorithm, the time complexity is $\mathcal{O}(kn^{\{g\}}dp + n^{\{g\}}k^2d)$, which is efficient compared to the state-of-the-art FC methods. A more detailed analysis can be found in the Appendix.

Appendix — https://github.com/Yunfan-Zhang/AFCL

Algorithm 1: AFCL: Asynchronous Federated Clustering.

Input: $\mathbf{X}^{\{1\}}, \mathbf{X}^{\{2\}}, ..., \mathbf{X}^{\{p\}}$, k, ξ, η.

for all clients in parallel **do**
  Initialize k seeds using k-means++;
  Transfer initialized seeds to the server;
end for
Initialize k global seeds $\mathbf{M}$ using k-means++ according to the seeds received from all clients;
repeat
  Transfer global seeds $\mathbf{M}$ to participating clients;
  for g**-th participating client** in parallel **do**
    Update $\theta^{\{g\}} = \theta^{\{g\}} + 1$;
    Compute $\mathbf{Q}^{\{g\}}, R^{\{g\}}, \mathbf{B}^{\{g\}}, \mathbf{z}^{\{g\}}$, by Eqs. (2), (5), (6), and (7), respectively;
    Upload $\mathbf{Q}^{\{g\}}, R^{\{g\}}, \mathbf{B}^{\{g\}}, \mathbf{z}^{\{g\}}$ to the server;
  end for
  Compute $\mathbf{M}$, Z using Eqs. (14) and (10);
  Update $\mathbf{w}$ by Eq. (8);
until Convergence

Output: $\mathbf{M}$, $\mathbf{Q}$.

No.	Datasets	Abbrev.	n	d	k^*
1	Synthetic Dataset 1	SD1	2300	2	4
2	Synthetic Dataset 2	SD2	2900	2	5
3	Seeds	SE	210	7	3
4	Iris	IR	150	4	3
5	Avila	AL	10430	10	12
6	Abalone	AB	4177	7	29
7	Breast Cancer	CC	569	30	2
8	Accent	AC	329	12	6
9	Segment	SG	2100	19	7
10	Live	LI	7051	9	2
11	Parkinson	PA	197	22	2
12	Audit	AU	776	24	2
13	Transfusion	TF	748	4	2

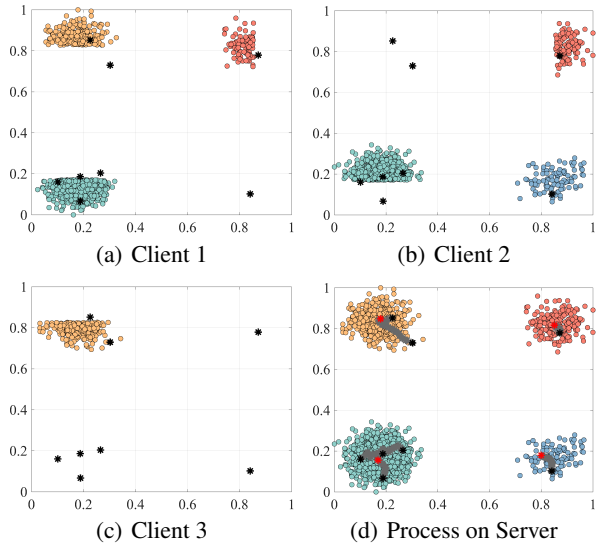
Table 2: Statistics of experimental datasets.

Experiments

Experimental Setup

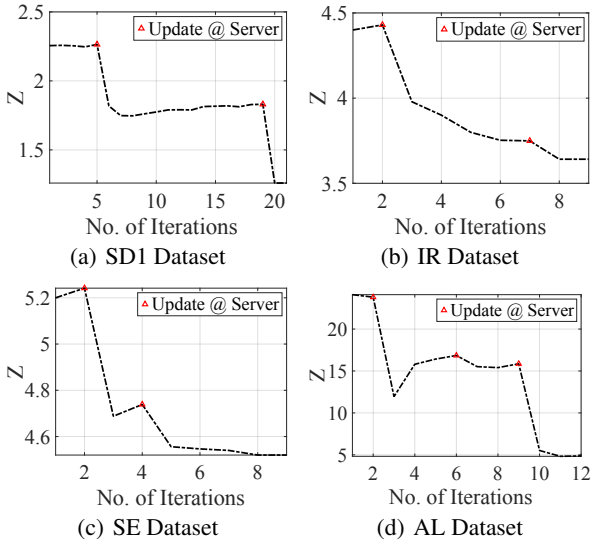
Six experiments have been conducted: (1) Visualization: Intuitive demonstration of the learning process of the proposed AFCL; (2) Convergence Evaluation: Objective function values at each learning iteration are recorded to illustrate the convergence and efficiency of AFCL; (3) Clustering Performance Evaluation: We compare AFCL with the conventional and state-of-the-art counterparts to demonstrate the superiority of AFCL; Due to space limitation, the remainder three experiments, i.e., significance test, ablation study, execution time evaluation, and more detailed experimental settings, are provided in the online appendix.

Six counterparts are compared: DK++ (Bahmani et al. 2012) is a conventional distributed learning approach that conforms to the settings of FL. Five state-of-the-art methods, i.e., the iterative learning approaches FFCM-avg1, FFCM-avg2 (Stallmann and Wilbik 2022), and FedSC (Qiao, Ding,



(a) Client 1 (b) Client 2 (c) Client 3 (d) Process on Server

Figure 3: Seed points and their trajectories on the server during the learning of AFCL. Black and red dots indicate the initial and final positions of the seed points, respectively.



(a) SD1 Dataset (b) IR Dataset (c) SE Dataset (d) AL Dataset

Figure 4: Values of the AFCL objective function on (a) SD1, (b) IR, (c) SE, and (d) AL datasets. Red triangles mark the iterations that the server update starts.

and Fan 2024), and the one-shot learning approaches k-FED (Dennis, Li, and Smith 2021) and Fed-SC (Xie, Wu, and Liao 2023), are compared. All their hyper-parameters (if any) are set according to the corresponding source papers.

13 datasets, including two Gaussian spherical synthetic datasets, and 11 public real datasets collected from the UCI machine learning repository (Asuncion and Newman 2007), are utilized for the experiments, and their statistics are shown in Table 2. All the real datasets are pre-processed by omitting the objects with missing values and normalized.

Three validity indices including the Silhouette Coefficient index (SC) (Rousseeuw 1987), Calinski-Harabasz index (CH) (Caliński and Harabasz 1974), and Bonferroni–Dunn (BD) test (Demšar 2006) are chosen for performance evaluation. Values of SC and CH are in the intervals [-1,1] and (0,$+\infty$), respectively, with a higher value indicating a better clustering performance. These two internal indices are insensitive to the number of clusters, thus facilitating a fair comparison of AFCL that learns its own k and the counterparts with a pre-set k^* as shown in Table 2. BD test is conducted on the performance ranks of the counterparts to validate if AFCL performs significantly better.

Visualization

To intuitively validate the effectiveness of AFCL, we split SD1 into three subsets for creating three extremely non-IID clients as shown in Figure. 3(a) - (c). Figure. 3(d) shows the global data distributions and update trajectories of the seeds on the server. It can be observed that even though the three clients have completely non-overlapping distributions, AFCL can still appropriately learn a set of seeds to represent the global cluster distributions. The trajectories also demonstrate that the seeds updating mechanism of AFCL can effectively facilitate interaction among the seeds with imbalanced update information uploaded by the asynchronous clients. After a certain number of learning iterations, redundant seeds are homogenized, i.e., overlapped at the center of several prominent clusters. This intuitively demonstrates the autonomous cluster number selection ability of AFCL.

To demonstrate the convergence efficiency of AFCL, we plot its objective function values on four datasets in Figure. 4. It can be observed that AFCL converges quickly with around 10 iterations in most cases. Moreover, the objective function always experiences a steep decline after the server updates, confirming that the designed seeds interaction mechanism is highly effective. It is also noteworthy that, since only limited statistics are permitted to be communicated between clients and server, a strict gradient descent cannot be guaranteed and thus the convergence curve in Figure. 4 is not monotonically decreasing. Such an effect is rational for FC because the clustering objective can be viewed as heterogeneous at different clients and server.

Clustering Performance Evaluation

The clustering performance of AFCL and the existing FC approaches are compared under the non-IID and asynchronous scenarios. For each dataset, we implement clustering by using each compared method by 20 times and report the average performance. For each of the 20 trails, we first implement k-means with $k = 5$, to divide the whole dataset into five clients to simulate the extreme non-overlapping distributions of non-IID clients. Then to simulate the asynchronous participation of clients, we randomly set each client with a different participation probability for controlling their upload in each iteration during the learning. As AFCL does not require k^*, the initial number of seed points k is randomly selected from the range $[k^*, 2k^*]$. The clustering performance in terms of SC and CH obtained un-

Dataset	DK++	k-FED	FFCM-avg1	FFCM-avg2	Fed-SC	FedSC	AFCL
SD1	0.5986±0.18	0.8494±0.00	0.5063±0.02	0.5036±0.02	0.8261±0.09	0.8539±0.00	**0.9714±0.00**
SD2	0.6127±0.11	0.7699±0.00	0.4773±0.03	0.4679±0.03	0.6005±0.14	0.7477±0.00	**0.8571±0.00**
SE	0.3229±0.00	0.3754±0.04	0.4323±0.05	0.4323±0.05	0.3619±0.00	0.3774±0.00	**0.5033±0.09**
IR	0.4955±0.01	0.4818±0.02	0.5672±0.02	0.6119±0.21	0.5384±0.04	0.5719±0.00	**0.6386±0.06**
AL	0.2096±0.02	0.0979±0.03	0.2721±0.09	0.2761±0.07	0.4422±0.05	0.3187±0.00	**0.6138±0.38**
AB	0.2314±0.02	0.1853±0.01	0.3664±0.34	0.4863±0.26	0.3009±0.03	0.3865±0.06	**0.5005±0.11**
CC	0.3778±0.00	0.3809±0.02	0.2873±0.05	0.3051±0.04	0.3672±0.04	0.3747±0.00	**0.5916±0.04**
AC	0.1831±0.02	0.0992±0.01	0.2656±0.13	0.2358±0.13	0.1376±0.02	0.1922±0.00	**0.4851±0.26**
SG	0.3197±0.02	0.3117±0.00	0.3819±0.11	0.3865±0.10	0.2677±0.04	0.2935±0.00	**0.6086±0.12**
LI	0.8241±0.00	0.8256±0.01	0.4805±0.16	0.4239±0.23	0.7980±0.10	0.8252±0.00	**0.8967±0.01**
PA	0.2763±0.00	0.4517±0.03	0.4419±0.15	0.4419±0.15	0.2443±0.00	**0.5138±0.00**	0.4903±0.11
AU	0.4046±0.05	0.3601±0.04	0.1440±0.08	0.2239±0.02	0.3411±0.07	0.4296±0.00	**0.5191±0.07**
TF	0.4935±0.00	0.5269±0.01	0.6402±0.16	0.6402±0.16	0.5172±0.25	0.5390±0.00	**0.6813±0.03**
Ave. Rank	4.6154	4.2308	4.4231	4.1923	5.4615	4.0000	**1.0769**

Table 3: Clustering performance evaluated by SC on all the 13 datasets.

Dataset	DK++	k-FED	FFCM-avg1	FFCM-avg2	Fed-SC	FedSC	AFCL
SD1	13933.3423	18933.7121	3136.2342	3140.0656	18901.0798	18958.6529	**19482.8610**
SD2	13187.1700	16936.0195	1462.2227	1474.5426	1575.4188	16913.4103	**17200.3823**
SE	192.6124	**251.1952**	83.3635	83.3635	193.6615	206.0449	230.9555
IR	**315.2151**	232.9987	65.6174	66.3026	211.5583	232.9386	310.7035
AL	1524.2576	1559.4321	1607.4565	1809.4353	2260.6359	1524.2576	**4880.6220**
AB	3756.1887	3791.0247	3821.2626	3890.4190	3072.4473	3244.3141	**5906.3378**
CC	202.4573	**290.0258**	104.0924	107.4325	222.6533	231.4785	231.5775
AC	112.3225	80.9425	82.4193	77.3857	73.2801	99.9817	**140.7156**
SG	973.7952	1121.9421	765.2036	1278.2297	781.12178	1057.1972	**1541.6809**
LI	5013.7432	5526.4145	1050.4084	806.5331	4075.7079	3608.0879	**6090.9982**
PA	40.2442	68.3886	79.5121	79.5121	56.2801	**83.3207**	78.1141
AU	304.9842	251.1567	65.7637	107.1965	201.6834	**433.3965**	306.7529
TF	393.1126	540.1820	436.6372	436.6372	420.2884	518.4788	**651.6017**
Ave. Rank	4.5000	3.0769	5.5769	4.8846	5.0000	3.4231	**1.5385**

Table 4: Clustering performance evaluated by CH on all the 13 datasets.

der the above settings is shown in Tables 3 and 4. The best and the second-best results are highlighted using boldface and underline, respectively. The 'Ave. Rank' rows report the average rank of different approaches across all datasets.

It can be observed that AFCL outperforms all its counterparts in general, indicating its superiority in asynchronous FC. Specifically, AFCL performs the best on almost all datasets w.r.t. the SC index, except on the PA dataset where AFCL still performs the second best. This is because AFCL can effectively minimize the intra-cluster dissimilarity and maximize the inter-cluster dispersion to search for the global optimal seeds. For the CH index, although AFCL performs the second-best on some datasets, i.e., SE, IR, CC, and AU, the best counterparts differ on these datasets while AFCL remains competitive in most cases. More specifically, for the cases where AFCL does not perform the best, the performance gap between AFCL and the best-performing counterpart is usually tiny, which demonstrates the effectiveness and robustness of AFCL on different datasets.

Concluding Remarks

This paper proposes a new FC approach called AFCL for mining global cluster distributions upon heterogeneous data distributions of asynchronously communicated clients. It advances FC to a more challenging but realistic scenario, i.e., clients can participate in the client-to-server uploading asynchronously, and all the clients and server can be extremely non-IID without knowing the 'true' number of clusters. AFCL achieves this by adopting a client-to-seed information fusion framework, which lets the seed points cooperate on the server to complete the non-IID distributions of clients and automatically learns to eliminate redundant seeds as well. A balance mechanism is also designed to relieve the non-uniform of the update information uploaded by the asynchronously participated clients. As a result, AFCL can effectively outline the global cluster distributions upon the seeds learned by the aggregated update intensity received from clients, even if the communication is extremely asynchronous and the distributions of clients are completely divergent. Comprehensive experiments have illustrated the efficacy of AFCL. Despite the superiority of AFCL, there are still some noteworthy potential limitations. That is, we assume FC on pure numerical data and the number of clients is relatively small. The next promising avenue would be the FC of datasets comprising both numerical and categorical attributes distributed on a large number of clients.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (NSFC) under grants: 62476063, 62102097, 62376233, and 62306181, the NSFC/Research Grants Council (RGC) Joint Research Scheme under grant: N_HKBU214/21, the Natural Science Foundation of Guangdong Province under grants: 2024A1313010039, 2024A1515010163, and 2023A1515012855, the Natural Science Foundation of Fujian Province under grant: 2024J09001, the General Research Fund of RGC under grants: 12201321, 12202622, and 12201323, the RGC Senior Research Fellow Scheme under grant: SRFS2324-2S02, the Shenzhen Science and Technology Program under grant: RCBS20231211090659101, the National Key Laboratory of Radar Signal Processing under grant: JKW202403, and the Xiaomi Young Talents Program.

References

Acar, A.; Aksu, H.; Uluagac, A. S.; and Conti, M. 2018. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys*, 51(4): 1–35.

Ahalt, S. C.; Krishnamurthy, A. K.; Chen, P.; and Melton, D. E. 1990. Competitive learning algorithms for vector quantization. *Neural Networks*, 3(3): 277–290.

Asuncion, A.; and Newman, D. 2007. UCI Machine Learning Repository.

Bahmani, B.; Moseley, B.; Vattani, A.; Kumar, R.; and Vassilvitskii, S. 2012. Scalable k-means+. *In 2021 Very Large Data Base Endowment*, (7): 1–12.

Banabilah, S.; Aloqaily, M.; Alsayed, E.; Malik, N.; and Jararweh, Y. 2022. Federated learning review: Fundamentals, enabling technologies, and future applications. *Information Processing & Management*, 59(6): 103061.

Cai, S.; Zhang, Y.; Luo, X.; Cheung, Y.-m.; Jia, H.; and Liu, P. 2024. Robust categorical data clustering guided by multi-granular competitive learning. In *2024 International Conference on Distributed Computing Systems*, 288–299.

Caliński, T.; and Harabasz, J. 1974. A dendrite method for cluster analysis. *Communications in Statistics-Theory and Methods*, 3(1): 1–27.

Cheung, Y.-m. 2004. A competitive and cooperative learning approach to robust data clustering. In *Neural Networks and Computational Intelligence*, 131–136.

Cheung, Y.-m.; and Jia, H. 2013. Categorical-and-numerical-attribute data clustering based on a unified similarity metric without knowing cluster number. *Pattern Recognition*, 46(8): 2228–2238.

Cheung, Y.-m.; and Zeng, H. 2009. Local kernel regression score for selecting features of high-dimensional data. *IEEE Transactions on Knowledge and Data Engineering*, 21(12): 1798–1802.

Cheung, Y.-m. 2005. On rival penalization controlled competitive learning for clustering with automatic cluster number selection. *IEEE Transactions on Knowledge and Data Engineering*, 17(11): 1583–1588.

Chung, J.; Lee, K.; and Ramchandran, K. 2022. Federated unsupervised clustering with generative models. In *2022 International Workshop on Trustable, Verifiable and Auditable Federated Learning*, 1–9.

Demšar, J. 2006. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7: 1–30.

Dennis, D. K.; Li, T.; and Smith, V. 2021. Heterogeneity for the win: One-shot federated clustering. In *2021 International Conference on Machine Learning*, 2611–2620.

Ding, S.; Li, C.; Xu, X.; Guo, L.; Ding, L.; and Wu, X. 2023. Horizontal federated density peaks clustering. *IEEE Transactions on Neural Networks and Learning Systems*, 1–10.

Ester, M.; Kriegel, H.-P.; Sander, J.; and Xu, X. 1996. Density-based spatial clustering of applications with noise. In *1996 International Conference on Knowledge Discovery and Data Mining*, 6, 1–5.

Ghosh, A.; Chung, J.; Yin, D.; and Ramchandran, K. 2020. An efficient framework for clustered federated learning. In *2020 Advances in Neural Information Processing Systems*, 19586–19597.

Hu, L.; Jiang, M.; Liu, X.; and He, Z. 2025. Significance-based decision tree for interpretable categorical data clustering. *Information Sciences*, 690: 121588.

Hu, L.; Jiang, M.; Liu, Y.; and He, Z. 2022. Significance-based categorical data clustering. *arXiv preprint arXiv:2211.03956*, 1–17.

Hu, X.; Qin, J.; Shen, Y.; Pedrycz, W.; Liu, X.; and Liu, J. 2023. An efficient federated multi-view fuzzy c-means clustering method. *IEEE Transactions on Fuzzy Systems*, 1886–1899.

Huang, L.; Li, Z.; Sun, J.; and Zhao, H. 2022. Coresets for vertical federated learning: Regularized linear regression and k-means clustering. In *2022 Advances in Neural Information Processing Systems*, 29566–29581.

Jia, H.; Cheung, Y.-m.; and Liu, J. 2014. Cooperative and penalized competitive learning with application to kernel-based clustering. *Pattern Recognition*, 47(9): 3060–3069.

Kumar, H. H.; Karthik, V.; and Nair, M. K. 2020. Federated k-means clustering: A novel edge ai based approach for privacy preservation. In *2020 International Conference on Cloud Computing in Emerging Markets*, 52–56.

Li, Y.; Wang, S.; Chi, C.-Y.; and Quek, T. Q. 2023. Differentially private federated clustering over non-IID data. *IEEE Internet of Things Journal*, 6705–6721.

Ma, J.; Zhang, Q.; Lou, J.; Ho, J. C.; Xiong, L.; and Jiang, X. 2019. Privacy-preserving tensor factorization for collaborative health data analysis. In *2019 International Conference on Information and Knowledge Management*, 1291–1300.

Nelus, A.; Glitza, R.; and Martin, R. 2021. Unsupervised clustered federated learning in complex multi-source acoustic environments. In *2021 European Signal Processing Conference*, 1115–1119.

Pedrycz, W. 2022. Federated FCM: Clustering under privacy requirements. *IEEE Transactions on Fuzzy Systems*, 30(8): 3384–3388.

Peng, M.; Wu, Y.; Zhang, Y.; Lu, Y.; Li, M.; and Cheung, Y.-m. 2025. Weighted density for the win: Accurate subspace density clustering. In *2025 International Conference on Acoustics, Speech and Signal Processing*, 1–5.

Qiao, D.; Ding, C.; and Fan, J. 2024. Federated spectral clustering via secure similarity reconstruction. In *2024 Advances in Neural Information Processing Systems*, volume 36, 58520–58555.

Ros, F.; Riad, R.; and Guillaume, S. 2023. PDBI: A partitioning Davies-Bouldin index for clustering evaluation. *Neurocomputing*, 528: 178–199.

Rousseeuw, P. J. 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20: 53–65.

Schubert, E.; Sander, J.; Ester, M.; Kriegel, H. P.; and Xu, X. 2017. DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN. *ACM Transactions on Database Systems*, 42(3): 1–21.

Stallmann, M.; and Wilbik, A. 2022. Towards federated clustering: A federated fuzzy c-means algorithm (FFCM). arXiv:2201.07316.

Thorndike, R. L. 1953. Who belongs in the family? *Psychometrika*, 18(4): 267–276.

Wang, B.; Yang, Y.; Wu, J.; Qi, G.-j.; and Lei, Z. 2023. Self-similarity driven scale-invariant learning for weakly supervised person search. In *2023 International Conference on Computer Vision*, 1813–1822.

Wei, K.; Li, J.; Ding, M.; Ma, C.; Yang, H. H.; Farokhi, F.; Jin, S.; Quek, T. Q.; and Poor, H. V. 2020. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15: 3454–3469.

Wu, J.; Yang, Y.; Liu, H.; Liao, S.; Lei, Z.; and Li, S. Z. 2019. Unsupervised graph association for person re-identification. In *2019 International Conference on Computer Vision*, 8321–8330.

Xie, S.; Wu, Y.; and Liao, e. a., Kewen. 2023. Fed-SC: One-shot federated subspace clustering over high-dimensional data. In *2023 International Conference on Data Engineering*, 2905–2918.

Zhang, C.; Xie, Y.; Bai, H.; Yu, B.; Li, W.; and Gao, Y. 2021. A survey on federated learning. *Knowledge-Based Systems*, 216: 106775.

Zhang, Y.; Cheung, Y.-m.; and Tan, K. C. 2020. A unified entropy-based distance metric for ordinal-and-nominal-attribute data clustering. *IEEE Transactions on Neural Networks and Learning Systems*, 31(1): 39–52.

Zhang, Y.; Zou, R.; Zhang, Y.; Zhang, Y.; Cheung, Y.-m.; and Li, K. 2025. Adaptive micro partition and hierarchical merging for accurate mixed data clustering. *Complex & Intelligent Systems*, 11: 1–14.

Zhou, W.; Li, P.; Han, Z.; Lu, X.; Li, J.; Ren, Z.; and Liu, Z. 2023. Privacy-preserving federated learning via disentanglement. In *2023 International Conference on Information and Knowledge Management*, 3606–3615.

Zhou, X.; and Wang, X. 2022. Memory and communication efficient federated kernel k-means. *IEEE Transactions on Neural Networks and Learning Systems*, 7114–7125.

Zou, R.; Zhang, Y.; Zhang, Y.; Lu, Y.; Li, M.; and Cheung, Y.-m. 2024. Federated clustering with unknown number of clusters. In *2024 International Conference on Data-driven Optimization of Complex Systems*, 671–677.