Asynchronous Federated Clustering with Unknown Number of Clusters

Appendix

Yunfan Zhang[1], Yiqun Zhang[1*], 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]Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen, China
[4]Department of Computer Science, Hong Kong Baptist University, Hong Kong SAR, China
{3121008002, yqzhang}@gdut.edu.cn, luyang@xmu.edu.cn, limengke@gml.ac.cn, {csxchen, ymc}@comp.hkbu.edu.hk

1 More Details and Results of the Experiments

This section mainly shows detailed experiment settings and other extended experiments to verify our claims and the validity of the proposed method.

1.1 Dataset Description

This subsection details the synthetic and real-world datasets used in our experiments.

Synthetic datasets: Two 2-D synthetic datasets are generated as follows: (1) SD1: $\mu_1 = [1,1]^T$, $\mu_2 = [1,5]^T$, $\mu_3 = [5,5]^T$, $\mu_4 = [5,1]^T$, $\sum = [0.1,0;0.1,0]$, the number of data objects in each cluster are $G_1 = 1500$, $G_2 = 500$, $G_3 = 200$, $G_4 = 100$; (2) SD2: $\mu_1 = [10,1]^T$, $\mu_2 = [1,10]^T$, $\mu_3 = [1,1]^T$, $\mu_4 = [9,11]^T$, $\mu_5 = [11,10]^T$, $\sum_1 = [0.9,0;0,0.9]$, $\sum_2 = [1-0.7;-0.7,1]$, $\sum_3 = [10.3;0.3,1]$, $\sum_4 = [0.30;0,0.3]$, $\sum_5 = [0.30;0,0.3]$, the number of data objects in each cluster are $G_1 = 500$, $G_2 = 1000$, $G3 = 1000$, $G_4 = 200$, $G_5 = 200$. The visualization of two 2-D synthetic datasets are shown in Fig. 1 in our Appendix.

Real-world datasets: To evaluate AFCL in multiple dimensions, i.e., in different numbers of objects, attributes, and clusters, eleven public datasets are included in our experiments. Specifically, public datasets commonly used include Iris (IR), Seeds (SE), Cancer (CC), Segment (SG), Parkinson (PA), and Transfusion (TF). Other public datasets are Avila(AL), Abalone (AB), Accent (AC), Live (LI), and Audit (AU), which vary in objects n, attributes d, and true clusters k^*.

1.2 Evaluation Indices

We use two metrics to evaluate the clustering performance of our method: the Silhouette Coefficient index (SC) (Rousseeuw 1987) and the Calinski-Harabasz index (CH) (Caliński and Harabasz 1974). SC is a conventional and popular internal index, which simultaneously indicates the compactness of clusters and the dispersion among clusters, with its values in [-1,1]. SC is computed accordingly

$$\text{SC} = \frac{1}{n}\sum_{i=1}^{n}\frac{a_i - b_i}{\max\{a_i, b_i\}}, \tag{1}$$

where a_i is the average nearest inter-cluster distance of $\mathbf{x}_i$, and b_i is the distance between $\mathbf{x}_i$ and the cluster it belongs.

CH computes the ratio of the average inter-seed distance to the average object-seed distance within clusters, with values ranging from (0,+∞). CH is computed by

$$\text{CH} = \frac{S_B/(k-1)}{S_W/(n-k)}, \tag{2}$$

where S_B is the sum of the square distances between cluster centers and the overall data center, and S_W is the sum of the square distance between the objects in each cluster and the center of the cluster.

For both these two indices, a higher value indicates a better clustering performance. Supposing the clustering result is not coherent with the natural data structure, the performance of SC and CH will decrease due to the imbalance between intra-cluster tightness and inter-cluster separation.

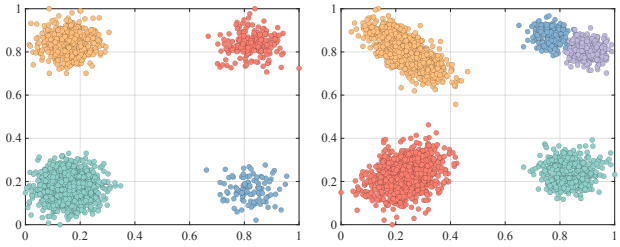

Fig. 1. Visualization of SD1 and SD2.

1.3 Significance Test

We also conducted BD tests to the results in 'Ave. Rank' rows of Tables 3 and 4 in our full paper, and visualize the test results based on the CD interval in Fig. 2 in our Appendix. The lengths of the CD, when comparing seven approaches on thirteen datasets, are 1.9357 and 1.7567 with $\alpha = 0.05$ and 0.1, respectively. It can be seen that AFCL

*Coresponding Author

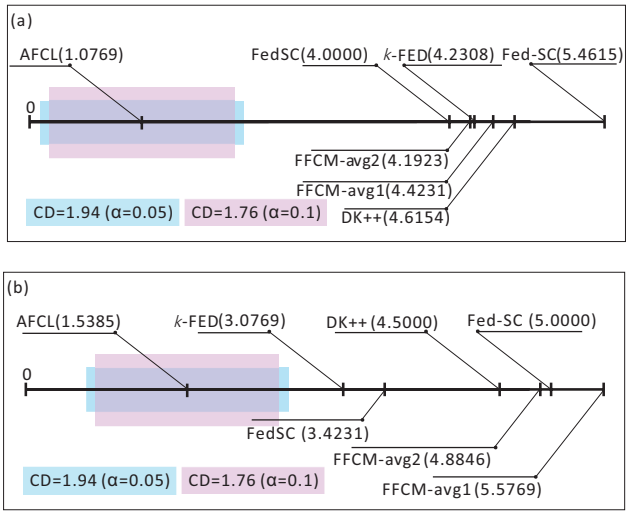

Fig. 2. BD test (a) and (b) based on the average ranks in Tables 3 and 4 in our full paper. Methods ranked outside the CD intervals are believed to perform significantly differently from AFCL.

significantly outperforms most of its counterparts, including three state-of-the-art methods, which indicate the competitiveness of AFCL in asynchronous conditions.

1.4 Ablation Study

We conduct ablation studies by using the discriminative SC index to validate the effectiveness of the core server Competitive and Cooperative of Seeds (CCS) process. The version of AFCL called AFCL-W is formed by replacing the CCS with the simple cluster seed aggregation of FFCM (Stallmann and Wilbik 2022). Specifically, the global seed is computed as a weighted average of the seeds from clients, where the weight assigned to each client's seed is proportional to the number of objects it represents.

It can be observed from Table 1 in our Appendix that AFCL outperforms AFCL-W in most cases in terms of the SC index. This intuitively illustrates the effectiveness of the proposed CCS. This is because CCS can sufficiently let the seed points interact with the neighboring ones. In this way, the detailed local distribution information can be iteratively propagated across different seed points to let them collaboratively eliminate redundant ones and sketch the global cluster distributions in the asynchronous scenarios.

1.5 Clustering Performance of Using the Number of Clusters Learned by AFCL

We found that in some experimental results, the final k', when AFCL convergence, is not equal to the 'true' clusters number k^*. To further demonstrate the versatility and effectiveness of AFCL, we replace k^* by k' that was learned by AFCL in some datasets for our counterparts, forming FFCM-avg1+ours, FFCM-avg2+ours, DK++ +ours, Fed-SC + ours, and FedSC + ours. All counterparts participate in the comparison using k^* and k' as the cluster numbers. The results, evaluated by the SC index, are shown in Fig. 3 in our Appendix.

Table 1: Clustering performance of AFCL and AFCL-W.

Dataset	AFCL	AFCL-W
SD1	**0.9714±0.00**	0.7248±0.27
SD2	**0.8571±0.00**	0.8501±0.03
SE	**0.5033±0.09**	0.3741±0.20
IR	0.6386±0.06	**0.6269±0.07**
AL	**0.6138±0.38**	0.4389±0.25
AB	**0.5005±0.11**	-0.0429±0.05
CC	0.5916±0.04	**0.6467±0.01**
AC	**0.4851±0.26**	0.2395±0.09
SG	**0.6086±0.12**	0.3550±0.13
LI	0.8967±0.01	**0.9012±0.03**
PA	**0.4903±0.11**	0.4763±0.10
AU	**0.5191±0.07**	0.4232±0.22
TF	0.6813±0.03	**0.8237±0.00**

It can be observed that with the utilization of our k', the performance of most methods exhibits improvement to varying degrees, while AFCL consistently outperforms them. This indicates that although each original label in some datasets may have a meaningful interpretation, making some labels become one cluster according to their similar meaning and features can make the cluster more compact. AFCL can effectively extract the data information from each client without prior knowledge of the 'true' cluster number while facilitating the optimal global cluster number for better clustering performance. Note that since classes are manually labeled, the data distribution labeled by the class may be not as compact as the cluster.

1.6 Clustering with Different Pre-specified Number of Cluster

To assess the clustering performance for different numbers of clusters k and to determine when k can be of a similar order of magnitude as n without compromising clustering performance, we set k as $\{n/64, n/32, n/16, n/8, n/4, n/2\}$ for clustering four commonly used real-world datasets. We evaluate the results based on the SC index and utilize k-FED with uniformly k^* as our benchmark to demonstrate our performance intuitively.

As shown in Fig. 4 in our Appendix, it is evident that the clustering performance of most datasets is suboptimal when $k \geq n/16$, which indicates that the excessive seeds may be difficult to merge with other seeds. However, the results consistently surpass the benchmark when $k < n/16$. This confirms that AFCL can perform effectively without having prior knowledge of the true number of clusters by setting k to be the same order of magnitude as n. Meanwhile, we suggest that k should be set relatively small to explore the global cluster distribution if the k^* is pre-known.

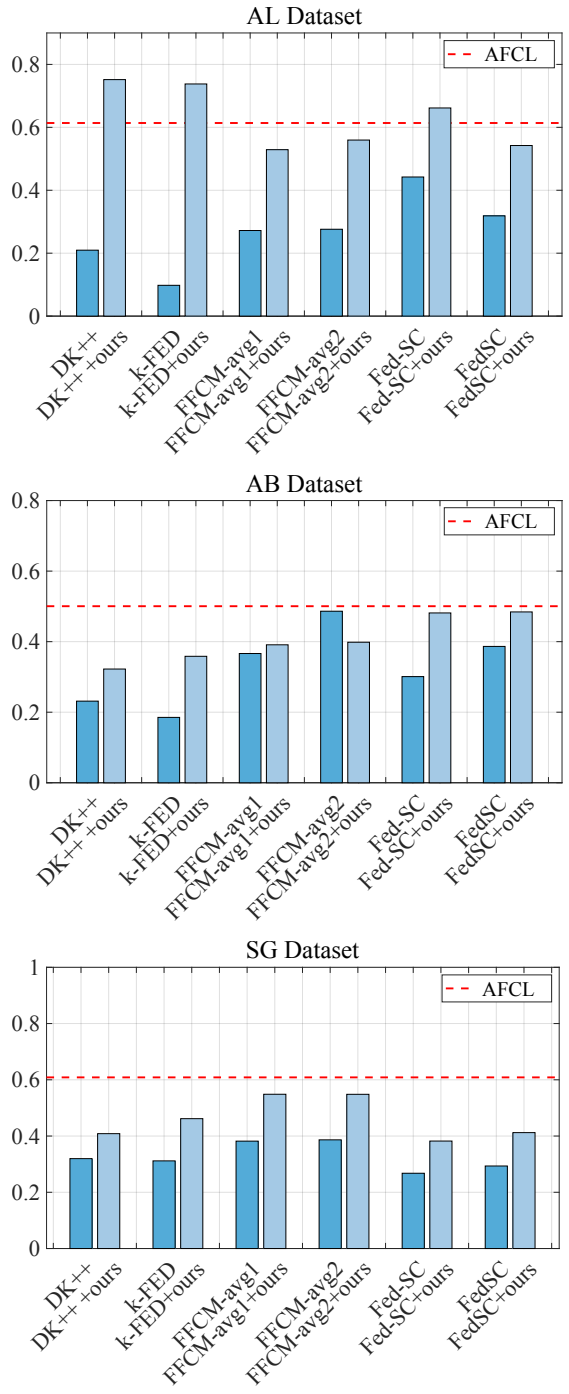

Fig. 3. Clustering performance of different methods evaluated by SC index using the 'true' number of clusters k^* (marked in dark blue) and the clusters number k' learned by AFCL (marked in light blue). The red dashed line indicates the performance of AFCL.

1.7 Efficiency Evaluation

To validate the efficiency of AFCL, its execution time is compared with the six conventional and state-of-the-art

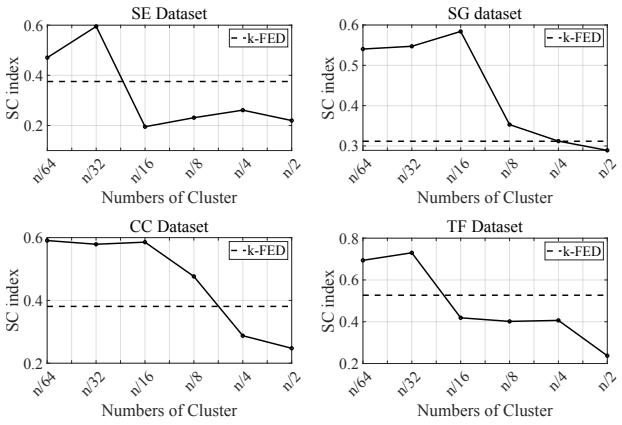

Fig. 4. Clustering performance of AFCL evaluated by SC index using a different number of clusters k. The black dashed line uniformly indicates the performance of k-FED using the true number of cluster k^*.

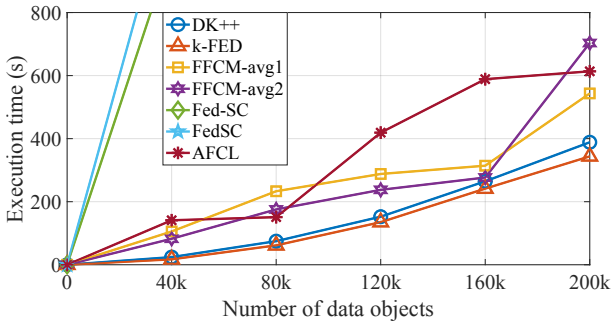

Fig. 5. Comparison of execution time w.r.t. n.

counterparts including DK++, k-FED, FFCM-avg1, FFCM-avg2, Fed-SC, and FedSC, on a large synthetic dataset. By increasing the number of data objects n and the number of attributes d, two groups of comparison are demonstrated in Fig. 5 and Fig. 6 in our Appendix, respectively. For the results with increasing n in Fig. 5, d is fixed at 2, and n is increased by a step-size of 40k until n = 200k. It can be seen that with the linear increase of n, the execution time of AFCL increases almost linearly after n = 80k. For Fig. 6, n is fixed at 2k, the number of attributes d is increased by a step-size of 0.5k until d = 2.5k. It can be seen that the increase in execution time is almost linear w.r.t. the increase in d for AFCL. The above observations conform to the time complexity analysis in Theorem 1. In short, the computational complexity of AFCL is linear to both n and d, which indicates its scalability for large-scale datasets.

2 Proof of Time complexity

Theorem 1. *The overall time complexity of AFCL is* $\mathcal{O}(kn^{\{g\}}dp + n^{\{g\}}k^2d)$ *for each iteration.*

Proof. The initialization of the global k seeds for the server and the local k seeds for each of the p clients by using the k-means++ algorithm takes $\mathcal{O}(kn^{\{g\}}d + kn^{\{g\}}dp)$. Then we

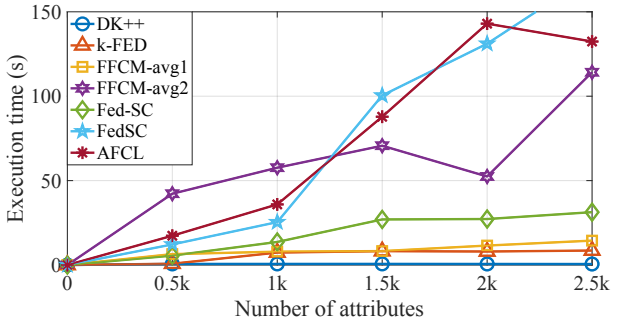

Fig. 6. Comparison of execution time w.r.t. d.

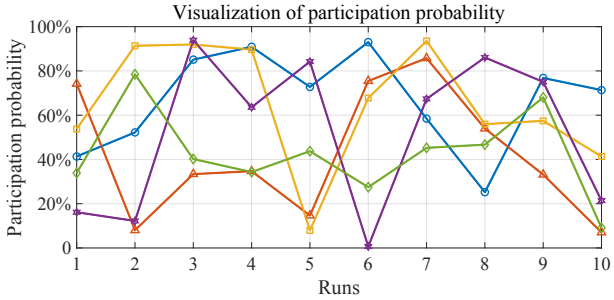

Fig. 7. Visualization of participation probability, where different color indicates different clients.

analyze the client-side and server-side time complexity, respectively.

For the client-side update accumulation, as the initialization operations are executed in constant time, the overall time complexity of the algorithm is primarily dominated by the nested loop structure. For the first nested FOR loop, client g possesses $n^{\{g\}}$ samples and k seeds. Furthermore, there is a summation operation over all k seeds in Eq. (2), leading to a time complexity of $\mathcal{O}(n^{\{g\}} \times k^2 \times d)$ for this loop. In the second FOR loop, all samples from the k seeds are involved, as both Eq. (6) and (7) require traversing all samples to which the seeds belong. The entire loop is equivalent to traversing all $n^{\{g\}}$ samples, resulting in a time complexity of $\mathcal{O}(2k \times n^{\{g\}})$. Consequently, the overall time complexity of client-side process is $\mathcal{O}(n^{\{g\}}k^2d + 2kn^{\{g\}})$.

In the process of Server-side seed interaction, Data from all p clients are processed during the aggregation phase in the worst-case scenario, resulting in a time complexity of $\mathcal{O}(2p)$. The nested FOR loop has k seeds, and Eq. (11) necessitates traversing $k-1$ seeds. The number of iterations in the second FOR loop depends on the result of Eq. (12). Considering the worst-case scenario, it needs to iterate $k-1$ times. Hence, the time complexity is $\mathcal{O}(k \times (k-1) \times d)$. Thus, the overall time complexity of the entire algorithm can be simplified to $\mathcal{O}(dk^2 + dk)$.

Hence, the overall time complexity of AFCL can be simplified into $\mathcal{O}(kn^{\{g\}}dp + n^{\{g\}}k^2d)$ for each iteration. □

3 Discussion of Experimental Settings

This section mainly discusses some important experimental settings in our federated clustering task.

3.1 Number of Clusters

The relationship between the true number of global and local clusters, i.e., k^*, and $k^{\{g\}*}$, is $k^{\{g\}*} \leq \sqrt{k^*}$.

Remark. *In this paper, we say a federated network with sufficient data is heterogeneous if $k^{\{g\}*} \leq \sqrt{k^*}$. The smaller $k^{\{g\}*}$ is, the more heterogeneity exists in this network. Therefore, in this scenario, the number of local seeds in our proposed method satisfied $k \geq k^* \geq (k^{\{g\}})^2$ in each client.*

3.2 Number of Clients

Though we set the number of clients $p = 5$ in our experiments, which is not a large number of clients to participate, the increasing number of clients does not make the federated clustering issue more challenging under complex distributions. Instead, the distribution among different clients can share more similarities if the number of clients is excessive. It is like the data that is randomly chosen from a few hospitals in a certain region is more heterogeneous than the data collected from all hospitals in this region, as patients usually visit various hospitals. Therefore, it cannot adequately verify our experimental settings. Instead of improving the efficiency of operating with large-scale clients, our focus is to adaptively cluster the heterogeneous distribution among clients.

3.3 Participation Probability

To intuitively demonstrate our challenging experimental settings, we visualize the participation probability of each client in the previous 10 runs in the SE dataset. As shown in Fig. 7 in our Appendix, the difference between the maximum and the minimum participation probability is large. Specifically, in nearly every run, there are usually clients with a participation probability of less than 40%. Additionally, in five out of the ten runs, some clients participate in less than 20%.

3.4 Parameter Settings

In our experiments, we set some hyperparameters including η, δ, , Δ, and k for implementing the proposed AFCL. Due to the distinctive characteristics of real-world datasets, η as the learning rate of the learning process is uniformly set to $1e{-}3{\times}d$. Therefore, the sum of attribute weight is uniformly equal to $1e-3\times d$, i.e., $\sum_{j=1}^{d} w_a^j = 1e-3\times d$. Though we randomly set k from the range $[k^*, 2k^*]$ in our clustering performance evaluation experiment, k also can be easily set as $k \leq n/32$ if the true number of cluster k^* is unknown. We uniformly set $\Delta = 1$ and $\delta = 8$ in our experiments.

As δ controls our attribute weight, now we analyze what values we can choose for δ.

If $\delta = 0$, w_a^j uniformly equals to η.

If $\delta = 1$, w_a^j equals to η for the largest value of $\| \mathbf{R}_i^j \|^2$. Therefore, the clustering is made by the selection of one

variable, which is unsuitable for real-world clustering problems.

If $0 < \delta < 1$, the larger $\| \mathbf{R}_i^j \|^2$, the larger w_a^j will be, which is against the attribute weighting principle.

If $\delta > 1$, the larger $\| \mathbf{R}_i^j \|^2$, the smaller w_a^j will be, the effect of j-th attribute with larger $\| \mathbf{R}_i^j \|^2$ is reduced.

If $\delta < 0$, the larger $\| \mathbf{R}_i^j \|^2$, the smaller w_a^j will be. However, w_a^j becomes smaller and has less weighting to the variable in the distance calculation because of δ.

Therefore, we can choose $\delta > 1$ or $\delta < 0$ in our proposed method.

References

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

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.

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