AiFed: An Adaptive and Integrated Mechanism for Asynchronous Federated Data Mining

Linlin You, *Senior Member, IEEE*, Sheng Liu, Tao Wang, *Member, IEEE*, Bingran Zuo, Yi Chang, *Senior Member, IEEE*, and Chau Yuen, *Fellow, IEEE*

***Abstract*—With the growing concerns on datasecurity and user privacy, a decentralized mechanism is implemented for federated data mining (FDM), which can bridge data silos and collaborate diverse devices in ubiquitous IoT (Internet of Things) systems and services to extract global and shareable knowledge, i.e., encoded in deep neural networks (DDNs). Moreover, compared with FDM in synchronous mode, asynchronous FDM (AFDM) is more suitable to accommodate devices with diversified computing resources and distinguishable working statuses. However, as AFDM is still in its infancy, how to harness heterogeneous resources and biased knowledge of learning participants within the asynchronous context remains to be addressed. Such that, this paper proposes an adaptive and integrated mechanism, named AiFed, in which, a layer-wise optimization of AFDM is implemented based on the integration of two dedicated strategies, i.e., an adaptive local model uploading strategy (ALMU), and an adaptive global model aggregation strategy (AGMA). As shown by the evaluation results, AiFed can outperform five state-of-the-art methods to reduce communication costs by about 61.76% and 56.88%, improve learning accuracy by about 1.66% and 3.05%, and accelerate learning speed by about 22.16% and 37.81% under IID (independent and identically distributed) and Non-IID settings of four standard datasets, respectively.**

***Index Terms*—Asynchronous federated data mining, layer-wise model aggregation, adaptive local model uploading, adaptive global model aggregation.**

I. Introduction

THE ever-increasing level of intelligence and automation in modern systems and services is driven by insightful and decisive knowledge mined from the vast amounts of data sensed by various devices (e.g., personal smartphones, assistive robotics, unmanned vehicles, etc.) [1], [2]. Therefore, how to manage related devices, and in turn, process the data they collect efficiently and effectively becomes crucial in data mining. As enabled by the prevailing Internet of Things (IoT), layered and centralized mechanisms are widely discussed to implement a collaborative cluster, through which, data-hungry models representing comprehensive knowledge shared among users can be trained in central servers and deployed onto edge devices to not only renovate service portfolio but also improve service quality [3], [4].

However, the growing concerns about data security and user privacy may restrict the connectivity of centralized IoT in sharing sensitive data. Such that, more isolated and fragmented data islands are forming at the edge and make centralized data mining tasks unperformable due to the lack of critical information. Moreover, the potential power of layered IoT may not be fully unleashed in centralized solutions as well, since edge devices equipped with plentiful sensing and computing capabilities may sit idle after the collection and transmission of local data, leading to a waste of resources [5], [6]. When facing the intrinsic pitfalls in centralized data mining, a novel approach, named federated data mining (FDM), is in the spotlight to train data-driven models, i.e., deep neural networks (DNNs) [7], [8], collaboratively, and foster the exchange of inter-knowledge in a privacy-preserving manner. As shown in Fig. 1, instead of sharing the raw data, each FDM client can learn and upload individual knowledge separately and simultaneously [9]. After the receival of individual knowledge, the FDM server can aggregate them for global knowledge, and then, distribute it to the clients through the network. Through such an iterative process, complete and precise global knowledge, i.e., encoded in a DNN [10], can be mined and shared to the edges for actual usage.

Due to its advantages in harnessing various computing resources and discovering private knowledge distributed at the edge, FDM has been widely adopted and applied to support related tasks in various data-sensitive domains [11], e.g., user behavior detection for smart home [12], emoji and word prediction for smart keyboard [13], credit risk management for smart finance [14], patient similarity analysis for smart healthcare [15], etc. While considering the mode to orchestrate clients, FDM can be categorized into synchronous FDM (SFDM), whose clients will share the same working pace, and asynchronous FDM (AFDM), whose clients can work separately and simultaneously. Regarding the scalability and elasticity in managing multi-end

Manuscript received 20 March 2023; revised 16 September 2023; accepted 11 November 2023. Date of publication 14 November 2023; date of current version 7 August 2024. The research was supported in part by the National Natural Science Foundation of China under Grant 62002398, in part by the GuangDong Basic and Applied Basic Research Foundation under Grant 2023A1515012895, and in part by the National Research Foundation Singapore under the AI Singapore Programme AISG under Grant AISG2-TC-2023-008-SGKR. Recommended for acceptance by Y. Gao. *(Corresponding author: Chau Yuen.)*

Linlin You, Sheng Liu, and Tao Wang are with the School of Intelligent Systems Engineering, Sun Yat-Sen University, Shenzhen 518107, China (e-mail: youllin@mail.sysu.edu.cn; liush235@mail2.sysu.edu.cn; wangt339@mail.sysu.edu.cn).

Bingran Zuo is with the Rehabilitation Research Institute of Singapore, Nanyang Technological University, Singapore 639798 (e-mail: bingran.zuo@ntu.edu.sg).

Yi Chang is with the School of Artificial Intelligence, Jilin University, Changchun 130012, China (e-mail: yichang@jlu.edu.cn).

Chau Yuen is with the School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore 639798 (e-mail: chau.yuen@ntu.edu.sg).

Digital Object Identifier 10.1109/TKDE.2023.3332770

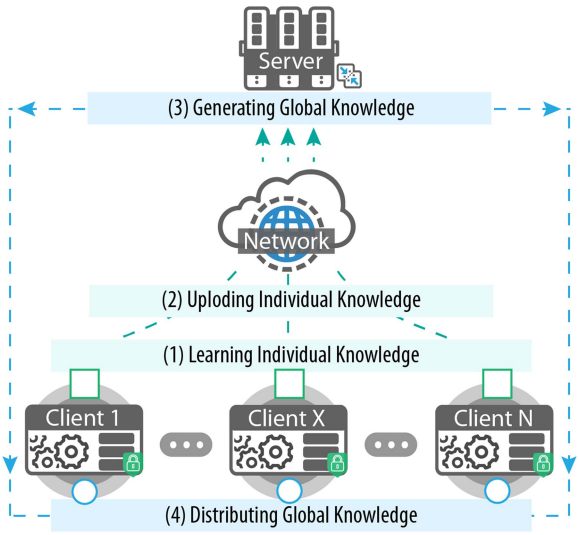

Fig. 1. Schematic diagram of FDM.

resources, AFDM becomes superior to SFDM in supporting ubiquitous IoT systems and services, as it can avoid the impacts of stragglers, which may become the performance bottleneck of FDM with learning accuracy decreased, and learning cost and time increased.

However, AFDM causes two kinds of heterogeneity at the client and the server, respectively. Specifically, the heterogeneity at the client is represented by the imbalanced and distinguishable local resources, i.e., Non-IID (non-independent and identically distributed) data, and rated communication capabilities, e.g., 4G to 6G [16]. It can significantly increase the complexity of learning an unbiased knowledge/model, e.g., in personalized mobility, a consistent and stable update of the travel behavior analysis model can be prevented by the dynamic connectivity of each client changing over time and space [17]. The heterogeneity at the server is shown by the different information staleness and richness of received local models. It can make the model aggregation high-cost and low-performance [18], [19], e.g., due to asynchronism, the created and received time of local models may vary among each other, and a stale model can make the global model overlearnt on redundant knowledge [20], [21].

To tackle the two kinds of heterogeneity, several solutions have been proposed. In general, they can be grouped into two kinds, either 1) focusing on optimizing the client-server interaction to reduce the learning cost or 2) enhancing the model aggregation to improve the learning performance [7], [22]. However, it may be contradictory to apply the two kinds of solutions at the same time, e.g., solutions improving model accuracy may increase the network burden, as more frequent and detailed updates are needed to alleviate the impacts of Non-IID data, and solutions reducing the communication cost may damage the overall training performance, as useful information may be omitted when data packages are reduced to be transmitted through the network.

As a novel solution to resolve such a dilemma, this paper proposes an adaptive and integrated mechanism for AFDM, called AiFed. First, it proposes an adaptive local model uploading strategy (ALMU) at each client and an adaptive global model aggregation strategy (AGMA) at the server to remedy the impacts of the two kinds of heterogeneities, respectively. Moreover, by using ALMU and ALMU jointly, it implements an optimized layer-wise model learning process for AFMD that can reduce learning costs, accelerate learning speed and improve learning accuracy. In general, the main contributions of the paper can be summarized as the followings:

- It optimizes the local model uploading phase in AFDM by ALMU, which can dynamically adjust the uploading frequency of each model layer according to their representational consistencies. As a result, it can not only dramatically reduce communication costs but also avoid overlearning on redundant knowledge;
- It enhances the global model aggregation phase in AFDM by AGMA, which can accurately measure the significance of local models (i.e., intermediate knowledge) according to their information staleness and richness. Such that, it can significantly improve overall performance in terms of learning speed and accuracy;
- By integrating ALMU and AGMA, a novel layer-wise model learning process for AFDM is implemented, which is more adaptive and elastic than the model average function to address the two kinds of heterogeneities.

In addition, according to a holistic evaluation based on IID and Non-IID settings of four common datasets, AiFed can effectively and efficiently support AFDM with 1) an average reduction of communication cost by about 61.76% (IID) and 56.88% (Non-IID), 2) an average increase of learning accuracy by about 1.66% (IID) and 3.05% (Non-IID), and 3) an average boost of training speed by about 22.16% (IID) and 37.81% (Non-IID) against five state-of-the-art baselines (i.e., FedAvg [23], FedProx [24], FedAsync [25], FedConD [26], and PartialNet [27]).

The remainder of this article is organized as follows. Sections II and III introduce AFDM by summarizing related solutions and formulating its optimization objectives, respectively. Next, AiFed is presented and evaluated in Sections IV and V, respectively. Finally, Section VI concludes the work and sketches the future research directions.

II. Related Solutions

As illustrated by Fig. 2, AFDM can loosen the local model uploading and global model aggregation processes, which can run concurrently without the restrictions on having all local models uploaded and received to start the global model aggregation. Such that, the clients and the server in AFDM can interact with each other more freely according to their actual running statuses with a more elastic aggregation process [31].

However, in such an asynchronous context, two kinds of heterogeneity can be identified, namely:

- *Heterogeneity at each client:* The clients at the edge are with limited or rated communication capabilities as well as biased or duplicated local data. Hence, it is neither frugal nor beneficial to upload deviate updates, as sharing them

TABLE I
SUMMARY OF SOLUTIONS ABOUT UPLOADING AND AGGREGATING STRATEGIES

Category	Resolved Issue	Related Work	Proposed Method	Main Contributions
Uploading	Compression of local models	[28]	Quantization	Achieving communication efficiency and model performance improvement
	Reduction of learning frequency	[29]	Periodic model averaging	Obtaining high predictive performance with less communications
	Optimization of uploading mode	[30]	Hybrid blockchain	Improving the running efficiency and training accuracy
		[7]	Layer update optimization	Reducing the communication cost through asynchronous layer updates
Aggregating	Temporal heterogeneity	[31]	Greedy selection	Addressing unstable communication network for an improved performance
		[32]	Age-aware aggregation	Adapting old models to achieve an balanced performance
		[7]	Weighted aggregation	Considering the staleness of local models
	Informative heterogeneity	[33]	Contract theory	Selecting "fine" local parameters to enhance the model aggregation
		[34]	Reinforcement learning	Using feedback of clients to update the aggregation weight
		[35]	Attention mechanism	Solving the imbalance problem to improve the quality of local models

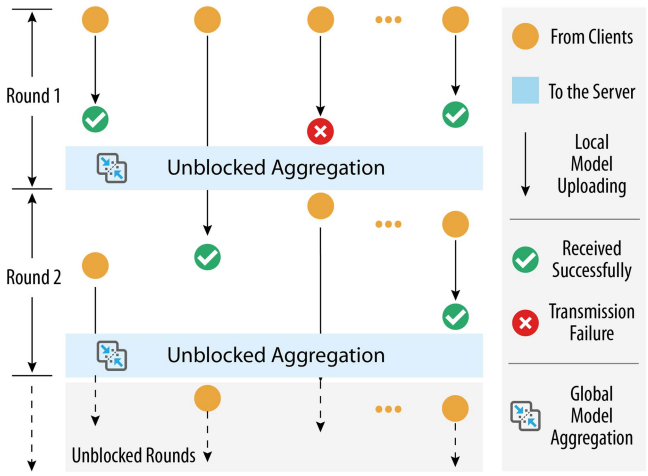

Fig. 2. Schematic diagram of AFDM.

will not only increase the learning cost but also influence the overall performance [17], [20];

- *Heterogeneity at the server:* The server receives local models with various temporal and informative attributes. Hence, without addressing them properly, the global model aggregation may become inefficient and ineffective, failing to achieve optimal performance in terms of learning stability, speed, and accuracy [32], [36].

As summarized in Table I, to address the two heterogeneities, current solutions can be categorized into two groups.

A. Group 1: Model Uploading Strategies

In general, the communication efficiency can be improved in three ways, i.e., the compression of local models, the reduction of learning frequency, and the optimization of updating mode [37]. Specifically, in the first approach, quantization [28], sparsification [38] and related compression methods are proposed to decrease the size of data packages in transmitting local models. As for the second approach, specific methods, such as parameter number reduction [39], periodic aggregation [29], and over-the-air computation [40], are discussed. Even though the first two approaches can effectively and efficiently reduce communication costs in terms of network traffic, they may bring side effects on overall performance.

Hence, under the constraint that useful information shall not be dropped-out during the learning process, the last approach can guarantee that the performance of the model will not be affected. To achieve optimal performance, related updating modes shall be designed by considering the characteristics of learning algorithms, e.g., since shallow layers (which contain general features) are more crucial but with fewer parameters than deep layers (which learn ad hoc features) in DNNs [41], [42], a layer-wise asynchronous model update strategy is designed to optimize the model uploading process by reducing the update frequency of deep layers [43].

B. Group 2: Model Aggregating Strategies

As the clients of AFDM can communicate with the server immediately after completing the local training, the temporal heterogeneity among local models can be reflected by the time attributes, e.g., created time, received time, and aggregated time. Based on the fact that the information importance may decrease over time, known as information staleness, several weighted strategies are studied, e.g., based on FedAvg, a temporally weighted aggregation strategy is implemented to aggregate local models with a weight derived from the difference between their created and received time [7].

Moreover, since the distribution of local data is generally correlated with user behaviors, informative heterogeneity may exist in the local models as well. Based on the observation that the generality and performance of machine learning models depend on the data quality of each client, i.e., information freshness, various filtering strategies are studied, e.g., a selective aggregation strategy to use "fine" local parameters for the model

aggregation [33], and a weighted strategy by incorporating an attention mechanism in optimizing aggregation weights to avoid the imbalance in local models [35].

C. Summary

As illustrated in [7], the joint utilization of the two kinds of strategies can improve the model performance in terms of training accuracy and speed, however, a mechanism, which can uniformly measure the importance of temporal and informative attributes encoded in local models at the server, as well as optimizing the updating frequency of layers of DNNs by utilizing heterogeneous local resources at each client, is still missing. To fill the gap, this paper proposes an adaptive and integrated mechanism AiFed to address the heterogeneity in AFDM through the joint optimization of the local model uploading and global model aggregation processes.

III. Problem Formulation

To achieve an efficient and effective AFDM, this paper aims to address the two kinds of heterogeneity identified at the client and server, respectively, to 1) update local models cost-efficiently, and 2) aggregate received local models for the global model more accurately. To ease the expression, this section, first, formulates the learning procedure of AFDM and then, discusses the two optimization objectives to be addressed by this paper. For the sake of readability, the important notations are listed in Table II.

TABLE II
List of Key Notations Used in the Paper

Notation	Meaning
R	The total number of learning rounds
r	The r^{th} global learning round, $r \in R$
M	The total number of AFDM clients
m	An arbitrary AFDM client, $m \in M$
n	The size of local data
K^r	The total number of local models to be aggregated in r^{th} round, $K^r \subseteq M$
k	A client with its local model received, $k \in K^r$
w^{loc}	The local model to be trained
w^{glo}	The global model to be trained
L	The total layer number of the model
U	The unit communication cost per parameter
l	The l^{th} layer of the model, $l \in L$
θ_l	The parameter of the l^{th} layer
β	The parameter to adjust the model aggregation function
rc_l	The representational consistency of the l^{th} layer
HM	The heterogeneity matrix storing level of information richness and freshness of local models
WV	The weight vector storing adaptive weights
LUM	The layer update matrix storing layer-wise updates from clients
$g(\cdot)$	The function to count number of parameters
$J(\cdot)$	The loss function to train the model
$F(\cdot)$	The function to optimize the local model uploading
$V(\cdot)$	The function to optimize the global model aggregation

A. Learning Procedure of AFDM

The learning procedure can be separated into a local learning phase consisting of local model training and uploading, and a global learning phase including global model aggregation and distribution.

Phase 1. The local learning at each client to update local model: Assume that there are M AFDM clients. Then, for the mth client, its local data and model are marked as d_m and w_m, respectively. Since in every local training, AFDM clients receive the up-to-date global model from the server and use it to train the local model, w_m has two variants before and after the local training, namely 1) the global model received from the server $w_m^{glo^{\hat{r}}}$ (before the training) and 2) the local model updated at the client $w_m^{loc^{\hat{r}}}$ (after the training), where $\hat{r}$ is the global learning round when the received global model is generated.

Accordingly, for client m, its local training can be made according to Formula (1).

$$J_m\left(w_m^{glo^{\hat{r}}}\right) = \frac{1}{n_m}\sum_{i=1}^{n_m} f\left(d_i; w_m^{glo^{\hat{r}}}\right)$$
$$w_m^{loc^{\hat{r}}} = w_m^{glo^{\hat{r}}} - \eta J'_m\left(w_m^{glo^{\hat{r}}}\right), \quad (1)$$

where $J_m(\cdot)$ is the loss learned for client m and $J'_m(\cdot)$ is its derivative; $f(\cdot)$ is the task-oriented loss function shared among clients (i.e., cross-entropy loss for classification, and mean squared error for regression); n_m is the total sample size of local data d_m; and η is the learning rate.

After the local model training, AFDM clients will upload their updated local model $w_m^{loc^{\hat{r}}}$ to the server, separately, and then, related clients will wait for an updated global model from the server to start a new round of local learning or stop the learning when a terminate signal is received.

Phase 2: The global learning at the server to aggregate received local models for the global model: Since the global learning process is unblocked, when a local model $w_m^{loc^{\hat{r}}}$ is received, for consistency, the server marks it as $w_{k,r}^{loc^{\hat{r}_k}}$ to denote that the kth local model is received at the current global learning round r. Note that due to the asynchronization, $\hat{r}$ can be different among clients, therefore, $\hat{r}_k$ is used.

After that, when a pre-defined timer or counter is triggered, the global aggregation in AFDM starts to update the global model w^{glo^r} according to Formula (2), which represents the model-wise aggregation process as shown in Fig. 3(A).

$$w^{glo^r} = \frac{1}{K^r}\sum_{k=1}^{K^r}\left(w_{k,r}^{loc^{\hat{r}}}\right), \quad (2)$$

where K^r is the total number of local models to be aggregated in the current global learning round r. Specifically, since the aggregation is controlled by either a timer or a counter, K^r will not change in each global learning round while a counter is used, and otherwise, it may vary.

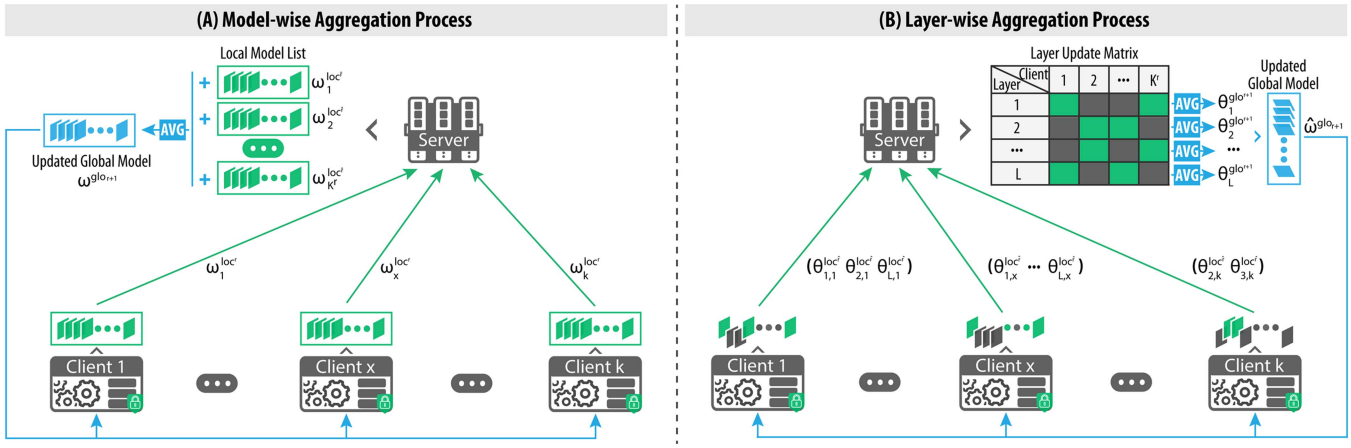

Fig. 3. Two kinds of model aggregation processes. (A) The conventional model-wise aggregation process (defined in Formula (2)), and (B) The novel layer-wise aggregation process (defined in Formula (3)).

Suppose that AI models, i.e., DNNs, consist of L layers, and for a layer l, its corresponding parameter is defined as θ_l, then Formula (2) can be rewritten to Formula (3), which implements a layer-wise model aggregation as shown in Fig. 3(B).

$$\hat{w}^{glo^r} = \left(\theta_1^{glo^r}, \theta_2^{glo^r}, \ldots, \theta_L^{glo^r}\right)$$
$$\theta_l^{glo^r} = avg(LUM_l^r) = \frac{1}{K^r}\sum_{k=1}^{K^r}\theta_{l,k,r}^{loc^{\hat{r}}}, \tag{3}$$

where $\hat{w}^{glo^r}$ is the global model generated based on the layer-wise aggregation function; $\theta_l^{glo^r}$ is the parameter of the lth layer in the rth global learning iteration; and LUM_l^r is the lth row of a layer update matrix (LUM), which, by default, is prefilled by the parameter of layer l of the current global model, and updated by the parameter received from local clients $\theta_{l,k,r}^{loc^{\hat{r}_k}}$. Note that the dimension of LUM is $L \times K^r$, and if clients upload their local model layers adaptively, it may happen that some cells of LUM will keep unchanged.

Regardless of the aggregation function utilized (i.e., model-wise or layer-wise), after the global model is updated, it is distributed to the clients to start another global learning round until a stop condition (e.g., the maximum number of rounds) is reached.

In summary, while comparing the two aggregation functions, the layer-wise aggregation process is more flexible for AFDM to optimize its local model uploading and global model aggregation phases [7], [20], [43]. Hence, to improve the efficiency and effectiveness of AFDM in both local and global learning steps, two optimization objectives of AiFed are defined, respectively.

B. Objective 1 (O.1) to Upload Local Models Cost-Efficiently

Since AFDM clients run with heterogeneous resources, to accommodate their dynamic availabilities, it becomes critical to upload local models with less communication cost. To achieve such an objective, how costly to upload local models shall be analyzed. Accordingly, the local model uploading cost for client m, denoted as B_m, can be calculated according to Formula (4).

$$B_m = g\left(w_m^{loc^{\hat{r}}}\right) \times U = \sum_{l=1}^{L}\left(g(\theta_{l,m})\right) \times U, \tag{4}$$

where $g(\cdot)$ counts the total number of parameters, and U is the unit parameter size (i.e., float 4 bytes or double 8 bytes).

Due to the uncertainty of network status, client availability, etc., B_m that affects data transmitting time and network payload is hard to be estimated. Hence, to reduce the analysis complexity, only the factors L, U and $\theta_{l,m}$ that are directly related to B_m are analyzed. Since L and U are not changeable once the learning is initialized, $\theta_{l,m}$ becomes the only variable affecting B_m. Besides the usage of compression techniques, it is intuitive to save B_m by reducing the number of layers to be uploaded, however, such a reduction may impact the overall model accuracy (as crucial information may lose).

Therefore, the first objective (O.1) of AiFed is to save the learning cost without damaging the learning accuracy, which is defined in Formula (5).

$$(\boldsymbol{O.1}): \quad \min\left(\sum_{l=1}^{L}\left(g(\theta_{l,m}) \times U \times F\left(\theta_{l,m}\right)\right)\right)$$
$$\text{s.t. } J\left(\hat{w}^{glo}\right) \approx J\left(w^{glo}\right), \tag{5}$$

where w^{glo} is the global model generated according to the conventional model-wise aggregation; $\hat{w}^{glo}$ is the global model updated according to the layer-wise aggregation; and to keep the performance of w^{glo} and $\hat{w}^{glo}$ equivalent, $F(\cdot)$ is a boolean function determining whether to upload $\theta_{l,m}$ or not.

C. Objective 2 (O.2) to Update Global Model More Accurately

To achieve optimal performance in model aggregation, it needs to harness the heterogeneity among received local models. By adopting the idea of weighted aggregation [43], layers of the

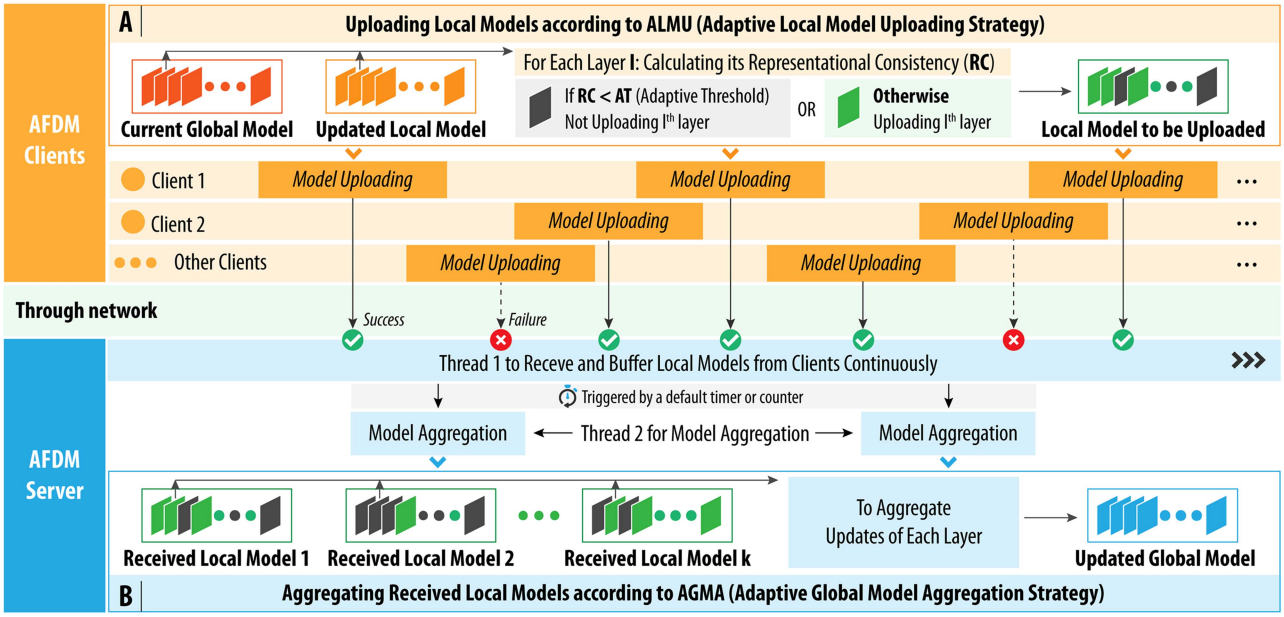

Fig. 4. Schematic diagram of AiFed: Adaptive and integrated mechanism for federated data mining in asynchronous mode. (A) Separated local model uploading process supported by ALMU, and (B) Unblocked global model aggregation supported by AGMA. Note that in the server, $Thread\ 1$ is for the continuous local model receiving, and $Thread\ 2$ is for the global aggregation that is triggered by a default counter or timer.

global model can be updated according to Formula (6).

$$\theta_l^{glo^r} = \frac{1}{K^r} WV^r \times LUM_l^r = \frac{1}{K^r} \sum_{k=1}^{K^r} \left(\beta_k^r \times \theta_{l,k,r}^{loc^{\hat{r}_k}} \right), \quad (6)$$

where β_k^r is a weight factor in the weight vector WV^r to alleviate the heterogeneity among local models in the rth global learning round.

Hence, as the second objective (O.2) of AiFed, a weight function, marked as $V(\cdot)$, is required to generate β_k^r in WV^r that can make the layer-wise aggregated global model $\hat{w}^{glo}$ with smaller loss than the conventional model-wise aggregated model w^{glo} as defined in Formula (7).

$$\begin{aligned} (\boldsymbol{O.2}) &: min \left(J\left(\hat{w}^{glo}\right) - J(w^{glo}) \right), \ J\left(\hat{w}^{glo}\right) \leq J\left(w^{glo}\right) \\ \hat{w}^{glo} &= LUM^r \times WV^r \\ WV^r &= [\beta_1^r, \ \beta_2^r, \ \ldots, \ \beta_{K^r}^r] \\ \beta_k^r &= V\left(w_{k,r}^{loc^{\hat{r}_k}}; \ w_r^{loc} \right), \end{aligned} \quad (7)$$

where w_r^{loc} contains all local models to be aggregated in the rth global learning round, and $V(\cdot)$ finds β_k^r by measuring the importance of $w_{k,r}^{loc^{\hat{r}_k}}$ in w_r^{loc}.

In summary, to achieve $O.1$ and $O.2$, appropriate $F(\cdot)$ and $V(\cdot)$ are required. Hence, AiFed proposes two strategies, i.e., ALMU and AGML, to implement them, respectively.

IV. The Proposed AiFed

As shown in Fig. 4, the proposed AiFed consists of two adaptive strategies, namely 1) an adaptive local model uploading strategy (ALMU) to optimize the local model uploading process by adjusting the uploading frequency of model layers dynamically, and 2) an adaptive global model aggregation strategy (AGMA) to enhance the model aggregation process by updating each layer of the global model adequately.

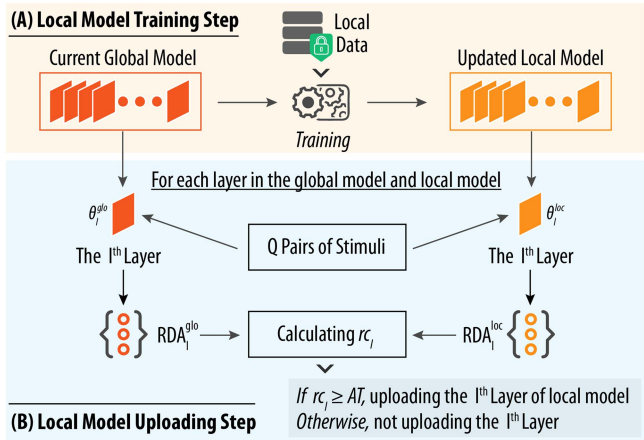

Fig. 5. Workflow of ALMU in each client. (A) Local Model Training Step to learn a new local model according to the current global model, and (B) Local Model Uploading Step to calculate the layer representation consistency (rc_l), which determines whether to upload the corresponding layer.

A. Adaptive Local Model Uploading Strategy (ALMU)

As shown in Fig. 5, ALMU runs at each client with two consecutive steps, namely:

- *Step 1. Local Model Training:* After receiving the global model distributed by the server, the client will use it to

start the local training, in which, the local model is updated based on local data;

- *Step 2. Local Model Uploading:* The layer-wise changes before and after the local training are analyzed. Based on the analysis result, the local model is prepared to be uploaded to the server. In this step, $F(\cdot)$ is implemented as defined in Formula (5) to achieve $O.1$.

In general, $F(\cdot)$ works as a controller to optimize the uploading frequency of each model layer. As a simple strategy, it can be defined as a conditional function, e.g., uploading the shallow and deep layers of DNNs [7] at a fixed rate, i.e., "3 : 1" indicating that when shallow layers are uploaded three times, deep layers are updated once [43]. Compared to shallow layers, deep layers have much more parameters, by applying such a simple strategy, the communication cost can be reduced.

However, the fixed rate may omit the dynamics within the local training, as layers may be updated at different paces. Specifically, in each learning iteration, incremental changes of layers may vary from each other. Hence, ALMU is proposed with an adaptive $F(\cdot)$, which adopts a multivariate analysis technique, called representational similarity analysis [44].

As shown in Fig. 5(B), a simplified representational consistency (RC) [45], [46] of a layer before (θ_l^{glo}) and after (θ_l^{loc}) the local training is calculated by measuring the differences between two representational dissimilarity arrays (RDAs). Note that RDA is a simplified version of the representational dissimilarity matrix (RDM), which is a symmetric matrix ($Q \times Q$) storing ($\frac{Q^2}{2} - Q$) unique pairwise distances between two representations of the lth layer while processing Q pairs of stimuli. Specifically, Q stimuli are small images prepared based on the training dataset [45], and RDA only contains Q elements in ($\frac{Q^2}{2} - Q$) of RDM. Hence, compared to RDM, the computation complexity of RDA can be reduced significantly.

Finally, the adaptive $F(\cdot)$ can be defined as Formula (8).

$$F\left(\theta_l^{glo}, \theta_l^{loc}\right) = \begin{cases} 1, & rc_l \geq AT \\ 0, & \text{otherwise} \end{cases}$$
$$rc_l\left(RDA_l^{glo}, RDA_l^{loc}\right) = \left(\frac{Cov(RDA_l^{glo}, RDA_l^{loc})}{\sigma_{RDA_l^{glo}}\, \sigma_{RDA_l^{loc}}}\right)^2, \tag{8}$$

where rc_l is the representational consistency of layer l, RDA_l^{glo} and RDA_l^{loc} are elements in the corresponding RDAs, AT is an adaptive threshold defined in Formula (9). In general, based on the two coefficients $\alpha_{\hat{r}}$ and α_{acc}, AT can be calculated by giving $\hat{r}$ (the global round when the global model used to update the local model is generated) and δ_{acc} (the accuracy change between the global model and the updated local model).

$$AT = \frac{1}{1 + e^{-(\alpha_{\hat{r}} \times \hat{r} + \alpha_{acc} \times \delta_{acc})}}. \tag{9}$$

In summary, ALMU can be deployed in each AFDM client to optimize the model updating process. Intuitively, ALMU can save communication costs since fewer parameters are transmitted. However, by using RC, ALMU can also filter out redundant knowledge, and in turn, upload more valuable parameters to the server. Hence, in general, ALMU can not only reduce learning costs but also improve overall training accuracy, which is validated in Section V-B.

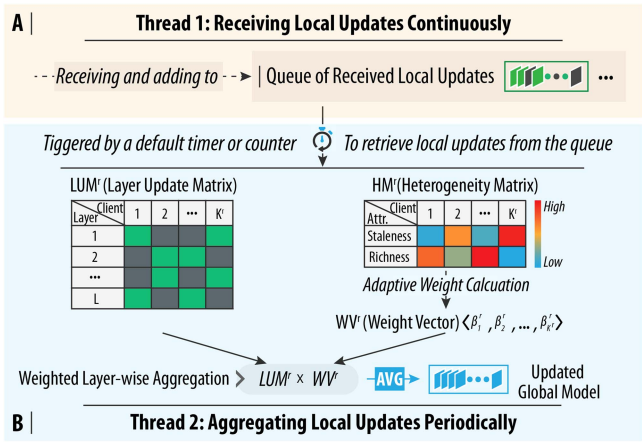

Fig. 6. Workflow of AGMA at the server. (A) The first thread to receive local models from the clients, and save them into a queue continuously. (B) The second thread to retrieve received models from the queue, and aggregate them periodically (i.e., managed by a default timer and counter).

B. Adaptive Global Model Aggregation Strategy (AGMA)

As shown in Fig. 6, AGMA works at the server with two concurrent threads, namely:

- *Thread to receive local models:* It runs continuously to receive local models from asynchronous clients. Once a local model is received successfully, the thread will add it to a queue, which works as a buffer sharing received models to the second thread for global model aggregation.
- *Thread to aggregate local models:* It is triggered by a default timer or counter to first retrieve related local models from the queue. Second, the weight vector (WV) is calculated based on a heterogeneity matrix (HM), which stores the differences among local models. Finally, the global model is updated by applying the weights on LUM as defined in Formula (6).

To achieve the second objective of AiFed, $V(\cdot)$ in Formula (7), which adjusts the adaptive weight β_k^r according to the importance of received local models, shall be defined. Since in each global learning round of AFDM, received parameters may vary in information staleness and richness, HM is used to store these differences, and accordingly, WV (consisting of β_k^r) can be calculated based on HM.

Particularly, for the creation of HM, the level of information staleness and richness shall be properly measured. First, based on the assumption that among received parameters, the latest one shall have the highest weight, the level of information staleness LIS can be measured according to Formula (10).

$$LIS_k^r = \left(\frac{e}{2}\right)^{-(r - \hat{r}_k)}, \tag{10}$$

where r is the current global learning round, and $\hat{r}_k$ is the global learning round when the global model that is used to train the local model in client k is generated. Once LIS for all participants is computed, the first row of HM, denoted as $HM^r(1:K^r)$, can be filled.

Moreover, since local data are, in general, generated randomly and subjectively, local parameters learned from them may make divergent contributions to the global model. Hence, the level of information richness LIR can be calculated according to Formula (11).

$$LIR_k^r = q(D_k^{\hat{r}_k}), \tag{11}$$

where $D_k^{\hat{r}_k}$ is the local data of client k used to update the global model generated in round $\hat{r}_k$, and $q(\cdot)$ can be a function of information entropy (IE) or label number (LN):

- *Function of IE:* It measures the uncertainty of information as defined in Formula (12).

$$IE_k^r(D_k^{\hat{r}_k}) = -\sum_{i=1}^{z}(p_i \times \log_2 p_i), \tag{12}$$

  where p_i is the percentage of the ith class in $D_k^{\hat{r}_k}$; and z is the total number of labels of a dataset.
- *Function of LN:* It measures the count of distinct labels presented in the local data as defined in Formula (13).

$$LN_k^r(D_k^{\hat{r}_k}) = \sum_{i=1}^{z} L_i, \tag{13}$$

  where L_i is a boolean value, representing the presence of the ith label.

Regardless of whether IE or LN is used, after the calculation of LIR for each participant, the second row of HM, denoted as $HM^r(2:K^r)$, can be created. Accordingly, by merging the two rows, HM in the global learning round r can be defined in Formula (14).

$$\begin{aligned} HM^r &= \begin{bmatrix} HM^r(1:K^r) \\ HM^r(2:K^r) \end{bmatrix} \\ &= \begin{bmatrix} LIS_{1,1}^r & \cdots & LIS_{1,K^r}^r \\ LIR_{2,1}^r & \cdots & LIR_{2,K^r}^r \end{bmatrix}. \end{aligned} \tag{14}$$

Finally, based on HM, $V(\cdot)$ implemented in AGMA to generate β_k^r in WV^r can be defined in Formula (15).

$$\begin{aligned} \beta_k^r &= V\left(w_{k,r}^{loc_{\hat{r}_k}}; w_r^{loc}\right) \\ &= \frac{n_k^{\hat{r}_k}}{n_r} \times \frac{LIS_k^r}{LIS^r} \times \frac{LIR_k^r}{LIR^r}, \end{aligned} \tag{15}$$

where $n_k^{\hat{r}_k}$ is the size of local data of client k used to update the global model generated in round $\hat{r}_k$; and n^r, LIS^r and LIR^r are the sum of $n_k^{\hat{r}_k}$, LIS_k^r and LIR_k^r, respectively.

After the calculation of β_k^r, WV^r can be formed and used in Formula (6) to update the global model layer-wisely. In general, LIS and LIR can steer the learning direction of the global model, and in return, improve the learning accuracy. The performance of AGMA is tested in Section V-C.

Algorithm 1: Integration of ALMU and AGMA.

Part 1: Executed in each AFDM client
1: Training the local model
2: **for** each layer $l \in L$ in the updated local model **do**
3: Calulating rc_l and AT
4: **if** $rc_l \geq AT$ **then**
5: Adding θ_l^{loc} to ω^{loc}
6: **end if**
7: **end for**
8: **if** ω_k is not null **then**
9: Uploading ω^{loc}, $\hat{r}$ and data size n to the server
10: **end if**

Part 2: Executed in the AFDM server
1: Thread 1: Receiving local models continuously
2: Thread 2: Aggregating local models periodically
3: **while** the default counter or timer is triggered **do**
4: Increasing the global round to r
5: **for** $k \in K^r$ **do**
6: Calculating β_k^r in WV^r according to Formula (15)
7: **end for**
8: Cloning $\hat{\omega}^{glo^{(r-1)}}$ as $\hat{\omega}^{glo^r}$
9: **for** $\theta_l^{glo^r} \in \hat{\omega}^{glo^r}$ **do**
10: updating $\theta_l^{glo^r}$ according to Formula (6)
11: **end for**
12: **end while**

C. AiFed: The Integration of ALMU and AGMA

In general, ALMU and AGMA can be used in AiFed separately, however, their integrated usage represents the ultimate form of AiFed. As described in Algorithm 1, AiFed consists of two parts, namely:

- *Part 1 in each client:* First, the clients are initialized to train local models in parallel. Second, for each layer in the updated local model, rc_l and AT are calculated and compared. If rc_l is not smaller than AT, θ_l^{loc} is added to a set ω^{loc}. Finally, if ω^{loc} is not empty, it, together with $\hat{r}$ the round when the global model used to train the local model is generated, and n the number of data used in the local training, is uploaded to the server.
- *Part 2 in the server:* There are two parallel threads, namely one to receive the local models from clients continuously, and another one to aggregate received local models periodically. Once the aggregation starts, WV^r is, first, calculated, and then, the weighted layer-wise aggregation starts. Since it may happen that there is no update for some particular layers, a replica of the current global model is used.

Moreover, since the above algorithm runs ALMU and AGMA separately at each client and the server, the time complexity (TC) of AiFed contains two parts, namely:

- *TC of ALMU:* As rc for each layer needs to be calculated based on RDAs, TC for ALMU is $O(L \times Q \times e_{\max})$,

where L is the total number of layers in the model, Q is the number of selected stimuli, and $e_{\max}$ denotes the maximum dimension of the output among the L layers.

- *TC of AGMA:* During the model aggregation, the most complex operation occurs when the adaptive weight is applied to each layer of each received local model. Hence, TC for AGMA is $O(L \times M)$, where L is the total number of layers in the model, and M is the total number of available clients, which is the maximum value of K^r (i.e., the number of learning participants in a learning round).

Finally, as ALMU and AGMA are integrated by AiFed in the asynchronous mode, the overall TC of AiFed is $O(\max((L \times Q \times e_{\max}), (L \times M)))$. In general, $O(L \times Q \times e_{\max})$ of ALMU at each client will be significantly higher than $O(L \times M)$ of AGMA at the server, and it makes AiFed costly for AFDM clients. Hence, a tactic can be utilized during the local training to reduce the TC of ALMU. Specifically, first, the selected Q pairs of stimuli (i.e., small samples under different classes) are selected and marked in the training dataset. Second, when the local training starts, their layer-wise representations are buffered for the initial and last epochs. Finally, based on the buffered representations, RDAs can be created and used to calculate rc for each layer. Such that, the TC of ALMU can be omitted, and the overall TC of AiFed can be reduced to $O(L \times M)$, which only occurs at the server.

In summary, based on the layer-wise learning process, AiFed optimizes and enhances the local model uploading and global model aggregation processes of AFDM according to two dedicated strategies, i.e., ALMU and AGMA. Such that, AiFed can not only reduce client-server communication costs but also improve overall model performance, which is analyzed in Section V-D.

D. Convergence Analysis

Compared to conventional methods, the local uploading and global aggregation processes are optimized by AiFed, which may affect its convergence. As revealed by the research [47], given a proper learning rate and sufficient learning rounds, the layer-wise uploading of FL can converge at a linear speed. Such that, the convergence of the global aggregation mechanism is further analyzed by giving three assumptions on the loss functions $J_m, \forall m \in M$.

Assumption 1 (Smoothness): P J_m is π-*smooth* with $\pi > 0$, i.e., for $\forall w_1, w_2, J_m(w_2) - J_m(w_1) \leq \langle \nabla J_m(w_1), w_2 - w_1 \rangle + \frac{\pi}{2}\|w_2 - w_1\|^2$.

Assumption 2 (Strong Convexity): J_m is μ-*strongly convex* with $\mu \geq 0$, i.e., for $\forall w_1, w_2$, $J_m(w_2) - J_m(w_1) \geq \langle \nabla J_m(w_1), w_2 - w_1 \rangle + \frac{\mu}{2}\|w_2 - w_1\|^2$.

Assumption 3 (Global Optimal): Assume that the learning problem has at least one solution w^*, that minimizes the global loss function $J(w)$, i.e., $\nabla J(w^*) = 0$.

For ease of expression, we define $\alpha = \frac{\sum_{r=1}^{R} K^r}{M \times R}$ as the global participant rate, $\tilde{\beta} = \min_{r,k}\{\beta_k^r\}$ as the minimum aggregation weight, and $\tau_{\max} = \max_{r,k}\{r - \hat{r}_k\}$ as the maximum staleness.

Based on the convergence analysis in [48], the Theorem 1 can be defined as listed below:

Theorem 1: If $\eta < \frac{\mu}{\pi^2}$, after the initial global model is updated for R rounds, the trained model satisfies

$$\mathbb{E}[J(w_R)] - J(w^*) \leq \kappa^R (J(w_0) - J(w^*)) + \delta, \tag{16}$$

where $\kappa = [1 - 2\alpha\eta\tilde{\beta}(\mu - \eta\pi^2)]^{\frac{1}{1+\tau_{\max}}}$ and $\delta = \frac{\eta\pi}{2\tilde{\beta}(\mu - \eta\pi^2)} \sum_{m \in M} \beta_m \|\nabla J_m(w^*)\|^2$.

From Theorem 1, the following insights can be observed:

- κ represents the convergence rate in a round, which decreases when the staleness $\tau_{\max}$ increases;
- δ represents the residual error (i.e., the function can converge to $\delta - neighbourhood$ of the optimal value), which increases when the data heterogeneity level increases (the value of $||J(w^*)||^2$ increases);
- The proposed adaptive uploading and aggregation methods have enhanced $\tilde{\beta}$, thus the residual error δ is reduced compared with traditional methods.

Therefore, theoretically, AiFed can make the model converge, which is also demonstrated by the evaluation results presented in Section V-D.

V. Performance Evaluation

To holistically reveal the performance of AiFed, two kinds of data are prepared based on four standard datasets. Specifically, IID data is created to evaluate the capability of AiFed in tackling the temporal heterogeneity caused by AFDM, and Non-IID data is used to measure how well AiFed is in supporting more realistic AFDM scenarios, in which, both temporal and data heterogeneities coexist. Based on common experimental settings, AiFed is tested in three stages, namely 1) the evaluation of ALMU; 2) the evaluation of AGMA; 3) the evaluation of the integrated ALMU and AGMA, which represents the ultimate form of AiFed.

A. The Common Settings

First, four common AFDM tasks are defined to learn corresponding convolution neural networks (CNNs) with two shallow layers and two deep layers based on four standard datasets, namely:

- Modified National Institute of Standards and Technology (MNIST[1]) dataset and Fashion-MNIST (FMNIST[2]) dataset: They contain 60 thousand training samples and 10 thousand testing samples in 10 labels, and their image size is $28 \times 28 \times 1$. The corresponding CNN model has two convolutional layers (with 64 and 128 filters, respectively) and two fully connected layers (with 256 and 512 neurons, respectively);
- CIFAR-10[3] dataset: It contains 50 thousand training samples and 10 thousand testing samples in 10 labels, and its image size is $32 \times 32 \times 3$. The two convolutional layers of the corresponding model have doubled filter numbers

[1] http://yann.lecun.com/exdb/mnist
[2] https://github.com/zalandoresearch/fashion-mnist
[3] https://www.cs.toronto.edu/kriz/cifar.html

TABLE III
DATA PARTITION SETTING

Dataset	Per Non-IID Data Partition		Per IID Data Partition	
	Samples #	Labels #	Samples #	Labels #
MNIST	1,000-2,000	2-6	1,500	10
FMNIST	1,000-2,000	2-6	1,500	10
CIFAR-10	950-1,550	2-6	1,250	10
GermanTS	650-1,050	4-12	850	43

TABLE IV
HYPERPARAMETERS USED IN THE LEARNING PROCESS

Dataset	Local epochs*	Learning rate	Batch size	Participant fraction
MNIST	2	0.003	48	0.2
FMNIST	2	0.003	48	0.2
CIFAR-10	5	0.001	48	0.2
GermanTS	2	0.001	48	0.2

* Local epoch is the iteration to train the local model in each client.

compared with MNIST's model since CIFAR-10 is a more complicated dataset, and the neuron numbers of fully connected layers are the same as MNIST's model;
- German Traffic Signs (GermanTS[4]) dataset: It contains 34,799 training samples and 12,630 testing samples with 43 labels, and its image size is $32 \times 32 \times 3$. The corresponding model has the same filter numbers as MNIST's model and has half neuron numbers of fully connected layers compared with MNIST's model.

Second, 40 data partitions are created based on the training samples of the four datasets according to the constraints defined in Table III. In general, there are two kinds of settings, i.e., 1) Non-IID setting, which is prepared according to [7] with uneven data size and labels in each data partition, and 2) IID setting, in which, training samples are shuffled randomly and assigned evenly. After these data partitions are prepared, they are randomly and uniquely assigned to 40 clients as their local data, and used to support the learning process that is configured with hyperparameters listed in Table IV, where local epochs denote the number of training iterations a client will execute to update the local model. Note that Non-IID and IID settings share the same hyperparameters.

Third, five state-of-the-art federated learning methods are used as the baselines, namely:
- FedAvg [23]: The most popular synchronous method with an average aggregation strategy ($C = 0.2$ in this paper);
- FedProx [24]: An improved method of FedAvg with a proximal term μ on the local optimization function to address data heterogeneity among clients ($\mu = 1$ in this paper);
- FedAsync [25]: An asynchronous method with a mixed hyper-parameter α in the aggregation function to remedy the impact of stragglers in synchronous mode ($\alpha = 0.5$ and $\mu = 1$ in this paper);
- FedConD [26]: An asynchronous method with a selective communication strategy based on update frequencies to speed up the model convergence ($\gamma = 0.2$ in this paper);
- PartialNet [27]: A synchronous method that only transmits parameters of the biggest dense layer in each round to reduce communication costs for the training of DNNs (to form and update the whole global model, other layers will be transferred every three rounds in this paper).

Finally, as for the evaluation metrics, three common indicators are defined and utilized, namely:
- $\mathbf{cost_a}$ the accumulated communication cost: It is calculated according to Formula (17), where $r_{current}$ is the current global learning round;
- $\mathbf{acc_h}$ the highest accuracy achieved: It is identified according to Formula (18), where TP, TN, FP, FN represent true positives, true negatives, false positives, and false negatives, respectively. Note that the maximum global round is set as 400 when all the baselines are converged;
- $\mathbf{round_t}$ the round first reached the target accuracy: It records the round number that the accuracy of the global model first reaches the pre-defined threshold, which is 90%, 70%, 50%, and 85% for MNIST, FMNIST, CIFAR-10, and GermanTS in Non-IID cases, and 95%, 75%, 60%, and 90% in IID cases. Note that IID cases reduce the learning complexity, and thus, their target accuracies are higher than the Non-IID cases.

$$cost_a = \sum_{r=1}^{r_{current}} (cost^r)$$
$$\text{s.t. } cost^r = \frac{4 \times n^r_{parameter}}{1024 \times 1024} \tag{17}$$
$$acc_h = \max\{acc^t\}, \quad t = 1, 2, \ldots, 400$$
$$\text{s.t. } acc = \frac{TP + TN}{TP + FP + FN + TN}. \tag{18}$$

It is worth noting that all the code and data used in the above-configured experiments are available at the link.[5] In the experiment, 40 clients are virtualized based on a Windows server (with Intel Xeon Gold 5218R CPU 2.10 GHz, 2 NVIDIA GeForce RTX 3090 GPUs, and 128 G RAM) and each virtualized client is randomly assigned with 1.5–4.5 MHz bandwidth, 8–16 GB RAM, and 1.0–2.0 GHz CPU. The variation in computing power is intended to simulate a heterogeneous and asynchronous environment.

B. The Evaluation of ALMU

To evaluate ALMU, stimuli sets for the four datasets shall be defined for the calculation of RC. Specifically, for MNIST, FMNIST, and CIFAR-10, 50 test stimuli are used (10 categories $\times$ 5 stimuli), and for GermanTS, 86 stimuli are used (43 categories $\times$ 2 stimuli). To reduce the burden of computing the conventional representational dissimilarity matrixes (RDMs) (which are symmetric), RDAs are used by randomly sampling 50 pairs

[4]https://bitbucket.org/jadslim/german-traffic-signs

[5]https://github.com/IntelligentSystemsLab/AiFed

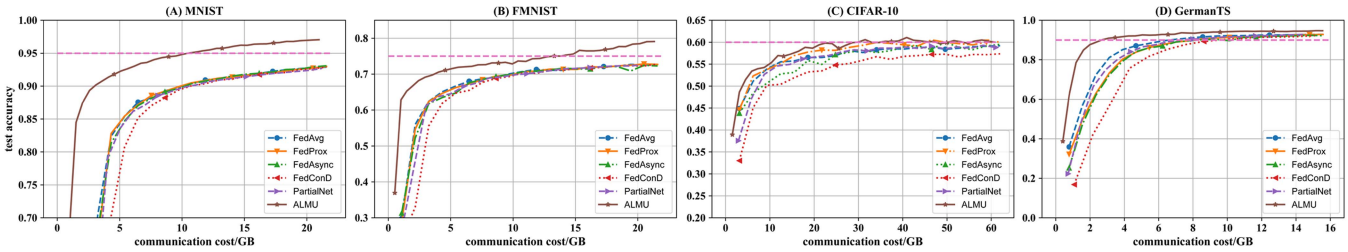

Fig. 7. Accuracy versus cost for ALMU and the five baselines in IID. (A) MNIST, (B) FMNIST, (C) CIFAR-10, and (D) GermanTS.

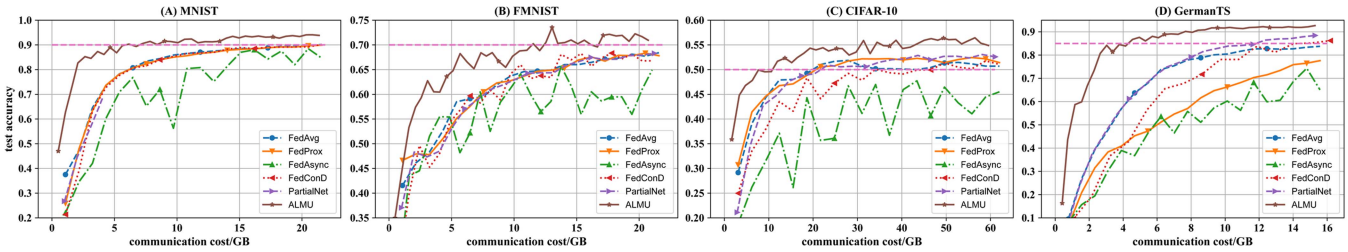

Fig. 8. Accuracy versus cost for ALMU and the five baselines in Non-IID. (A) MNIST, (B) FMNIST, (C) CIFAR-10, and (D) GermanTS.

from either $(50 * 50)/2$ or $(86 * 86)/2$ in RDMs to calculate RC. It is obvious that RDAs are less complex than RDMs (as shown by their differences in the number of stimuli pairs), and hence, the adoption of RDAs can restrict the computation cost added to the local learning step.

Accordingly, the efficiency and effectiveness of ALMU are evaluated. First, as illustrated in Figs. 7 and 8, ALMU can remain at a higher and more stable growth rate per unit cost of communication than the five baselines during the inter-knowledge exchanging process. Particularly, along with the growth of $cost_a$ in all eight cases (four in Non-IID + four in IID), ALMU has a much shaper accuracy growth curve than the five baselines. It shows that ALMU is beneficial for AiFed to be cost-efficient.

Moreover, while comparing the "Uploading" group with the "Baseline" group in Table V, ALMU can also make a bilateral improvement on acc_h and $round_t$, specifically:

- acc_h improves on average by about 1.65% and 2.74% in IID and Non-IID cases, respectively;
- $round_t$ shortens on average by about 17.63% and 33.90% in IID and Non-IID cases, respectively.

The above analyses illustrate that ALMU can resolve the dilemma of saving communication costs and improving learning performance (in terms of accuracy and speed) simultaneously. Notably, its improvements in Non-IID cases are more significant and stable than those in IID cases, illustrating its merits in handling AFDM tasks with heterogeneous local resources that are more likely to occur in real-world situations.

C. The Evaluation of AGMA

Since LIS (level of information staleness) and LIR (level of information richness) can be used separately or jointly, and LIR can be implemented based on IE (information entropy) or LN (label number), in total, there are five AGMA variants, namely LIS, $LIR - IE$ and $LIR - LN$ for independent usage, and $AGMA - IE$, and $AGMA - LN$ for joint usage. Based on the common settings, the five variants are executed and analyzed.

First, according to the results in the "Aggregation" group in Table V, both LIS and the two variants of LIR ($LIR - IE/LN$) outperform most baselines not only in training accuracy but also in training speed regardless of Non-IID and IID settings. As for more complex Non-IID cases, 1) LIS can improve training accuracy by about 1.90%, 2.80%, 0.46%, and 0.65%, and correspondingly, boost training speed by about 51.81%, 70.24%, 12.28%, and 2.76% for MNIST, FMNIST, CIFAR-10, and GermanTS, respectively; and 2) $LIR - IE$ and $LIR - LN$ share a similar performance with acc_h of 93.18%, 71.69%, 53.86%, and 89.65%, and $round_t$ of 160, 242, 56, and 230 for the four datasets, respectively. It shows that the temporal and data heterogeneities in local models can be addressed properly by LIS and LIR.

Moreover, as shown in Tables VI and VII, the accuracy growth of all five AGMA varients keeps ahead of the five baselines in both IID and Non-IID cases, and the joint usage of LIS and LIR, represented by $AGMA - IE/LN$, can inherit and combine their advantages. As a result, compared to the best score of the five baselines, $AGMA - IE/LN$ can have an average acc_h increase of 2.24% in four IID cases, and 2.65% in four Non-IID cases.

In summary, the above analyses show that the joint usage of LIS and LIR is superior to the independent usage, especially in Non-IID settings, and AGMA can tackle the temporal and data heterogeneities of local models through the weighted layer-wise model aggregation process proposed and implemented in AiFed.

TABLE V
EXPERIMENT RESULTS OF THE THREE ANALYSIS GROUPS

Group	Strategy	$\mathbf{acc_h}$ Highest Accuracy Achieved							
		MNIST		FMNIST		CIFAR-10		GermanTS	
		Non-IID	IID	Non-IID	IID	Non-IID	IID	Non-IID	IID
Baseline	FedAvg	92.84%	95.48%	71.87%	75.51%	52.47%	60.45%	86.27%	93.63%
	FedProx	92.69%	95.48%	**72.13%**	75.41%	52.16%	60.53%	86.61%	93.93%
	FedAsync	92.84%	**95.63%**	72.10%	**76.07%**	50.02%	**60.83%**	84.92%	**94.24%**
	FedConD	**93.09%**	95.47%	71.43%	75.87%	53.39%	58.66%	**89.47%**	93.82%
	PartialNet	92.27%	95.12%	70.59%	74.41%	**53.89%**	59.94%	89.29%	93.82%
Uploading	ALMU	94.48%	97.09%	73.53%	79.09%	56.86%	61.07%	93.00%	94.89%
Aggregation	LIS	94.86%	97.27%	74.15%	73.46%	54.14%	60.37%	90.05%	94.11%
	LIR-IE	93.13%	95.56%	71.69%	75.09%	53.49%	**61.40%**	89.65%	93.87%
	LIR-LN	93.18%	95.58%	70.03%	75.49%	53.86%	61.00%	86.71%	93.41%
	AGMA-IE	95.25%	97.29%	74.38%	80.91%	54.03%	61.03%	**90.90%**	**94.41%**
	AGMA-LN	**95.34%**	**97.33%**	**74.80%**	**81.08%**	**55.44%**	60.83%	90.26%	94.20%
Integration	AiFed-IE	**94.84%**	**97.02%**	73.99%	78.71%	**56.18%**	60.72%	**92.41%**	**94.80%**
	AiFed-LN	94.79%	96.97%	**74.15%**	**78.95%**	55.76%	**61.32%**	91.98%	93.95%

Group	Strategy	$\mathbf{round_t}$ Round When Target Accuracy First Achieved							
		MNIST		FMNIST		CIFAR-10		GermanTS	
		Non-IID	IID	Non-IID	IID	Non-IID	IID	Non-IID	IID
Baseline	FedAvg	194	340	260	375	63	270	264	**84**
	FedProx	203	343	252	371	**57**	149	322	104
	FedAsync	210	**325**	**243**	**331**	380	**140**	-	97
	FedConD	**193**	351	**243**	393	84	-	**181**	87
	PartialNet	219	380	337	-	67	-	205	100
Uploading	ALMU	114	197	154	239	48	162	93	68
Aggregation	LIS	93	167	**75**	-	50	75	176	73
	LIR-IE	160	343	242	353	60	99	230	120
	LIR-LN	165	334	349	355	56	90	336	100
	AGMA-IE	90	**163**	80	198	54	69	184	**57**
	AGMA-LN	**89**	167	109	**177**	**38**	**61**	**165**	61
Integration	AiFed-IE	117	199	150	**233**	**52**	205	**102**	**65**
	AiFed-LN	**92**	**195**	**130**	239	**52**	**145**	111	70

TABLE VI
ACCURACY COMPARISON BETWEEN AGGREGATION STRATEGIES AND BASELINES IN IID SETTING

Strategy	Round for MNIST					Round for FMNIST				
	80	160	240	320	400	80	160	240	320	400
FedAvg	89.28%	92.24%	93.74%	94.83%	95.48%	69.06%	72.11%	73.32%	74.23%	74.97%
FedProx	89.22%	92.05%	93.73%	94.80%	95.38%	69.43%	71.91%	73.52%	74.11%	74.66%
FedAsync	89.14%	92.14%	93.80%	94.88%	95.55%	69.35%	72.14%	73.14%	73.17%	75.64%
FedConD	88.25%	91.98%	93.62%	94.74%	95.42%	68.81%	71.90%	72.51%	73.85%	75.68%
PartialNet	88.56%	91.68%	93.10%	94.32%	95.07%	68.91%	71.81%	73.23%	73.35%	73.87%
LIS	92.19%	94.87%	96.05%	96.78%	97.20%	69.43%	73.03%	74.23%	74.98%	78.40%
LIR-IE	89.10%	92.17%	93.78%	94.87%	95.52%	69.47%	72.39%	73.75%	74.45%	75.05%
LIR-LN	89.10%	92.33%	93.78%	94.87%	95.58%	69.52%	72.33%	73.66%	74.28%	75.49%
AGMA-IE	92.27%	**94.89%**	**96.19%**	**96.82%**	97.24%	**72.29%**	**74.44%**	75.67%	**77.88%**	79.90%
AGMA-LN	**92.29%**	94.82%	96.14%	96.75%	**97.26%**	71.97%	73.91%	**76.21%**	77.60%	**80.61%**

Strategy	Round for CIFAR-10					Round for GermanTS				
	80	160	240	320	400	80	160	240	320	400
FedAvg	57.02%	59.27%	58.84%	59.12%	59.45%	89.55%	92.53%	93.17%	93.30%	93.63%
FedProx	57.66%	58.05%	57.91%	58.30%	58.41%	87.71%	92.03%	92.92%	93.34%	93.58%
FedAsync	57.16%	60.05%	58.20%	58.30%	60.23%	87.03%	92.03%	92.81%	93.20%	93.67%
FedConD	54.79%	57.27%	58.39%	57.46%	58.41%	89.30%	92.29%	92.87%	92.98%	93.66%
PartialNet	56.96%	59.14%	59.35%	59.11%	59.54%	83.38%	88.70%	90.09%	91.35%	92.27%
LIS	58.48%	57.51%	58.45%	58.63%	58.72%	91.08%	92.50%	93.52%	93.68%	93.68%
LIR-IE	58.36%	**60.13%**	58.93%	58.92%	59.66%	87.75%	92.36%	93.30%	93.59%	93.88%
LIR-LN	51.32%	59.50%	59.59%	59.42%	59.56%	87.98%	92.59%	93.29%	93.39%	93.41%
AGMA-IE	59.21%	58.85%	58.82%	59.24%	59.71%	**91.53%**	92.84%	**93.71%**	**94.01%**	93.99%
AGMA-LN	**59.27%**	59.00%	**59.74%**	**59.97%**	**60.33%**	91.15%	**93.28%**	93.29%	93.72%	**94.04%**

TABLE VII
ACCURACY COMPARISON BETWEEN AGGREGATION STRATEGIES AND BASELINES IN NON-IID SETTING

Strategy	Round for MNIST					Round for FMNIST				
	80	160	240	320	400	80	160	240	320	400
FedAvg	84.45%	88.79%	90.76%	91.83%	92.84%	61.96%	67.13%	69.20%	70.86%	71.21%
FedProx	84.00%	88.88%	90.74%	91.77%	92.51%	62.31%	66.73%	68.58%	70.49%	71.50%
FedAsync	71.97%	84.27%	87.38%	88.45%	90.50%	51.21%	58.57%	55.95%	71.47%	67.28%
FedConD	83.96%	89.05%	90.51%	91.49%	92.99%	59.70%	63.40%	68.12%	68.08%	68.69%
PartialNet	83.38%	88.70%	90.09%	91.35%	92.27%	60.95%	67.47%	69.05%	69.43%	70.32%
LIS	87.98%	92.27%	92.54%	94.30%	94.01%	65.52%	67.23%	**72.10%**	72.80%	70.11%
LIR-IE	86.06%	90.03%	91.30%	92.16%	93.00%	61.76%	67.04%	69.27%	70.09%	71.38%
LIR-LN	85.60%	89.68%	91.19%	92.54%	93.14%	62.26%	66.70%	68.32%	68.94%	69.04%
AGMA-IE	**89.16%**	92.39%	**93.79%**	**94.54%**	94.85%	**70.17%**	68.95%	70.82%	**73.19%**	**72.84%**
AGMA-LN	88.82%	**92.55%**	93.32%	**94.54%**	**94.98%**	66.14%	**69.42%**	72.03%	72.16%	72.75%

Strategy	Round for CIFAR-10					Round for GermanTS				
	80	160	240	320	400	80	160	240	320	400
FedAvg	51.57%	51.24%	51.83%	50.53%	51.36%	73.81%	82.75%	85.00%	85.49%	86.07%
FedProx	50.75%	51.41%	51.13%	50.27%	52.00%	51.17%	71.53%	80.85%	84.77%	86.06%
FedAsync	36.10%	46.46%	43.42%	46.18%	42.84%	53.56%	59.92%	68.34%	75.61%	81.69%
FedConD	47.25%	51.36%	51.50%	51.22%	52.56%	68.73%	**83.60%**	87.32%	88.68%	89.47%
PartialNet	50.91%	51.99%	52.75%	52.35%	52.21%	80.38%	85.70%	87.09%	88.35%	89.29%
LIS	51.70%	51.52%	51.97%	53.55%	53.22%	**74.14%**	81.56%	86.84%	88.12%	89.76%
LIR-IE	52.16%	52.07%	52.04%	52.51%	52.84%	67.68%	81.07%	86.05%	87.92%	89.08%
LIR-LN	51.32%	**53.00%**	52.37%	51.71%	52.92%	56.87%	74.48%	81.53%	84.19%	86.64%
AGMA-IE	51.81%	51.99%	53.37%	52.67%	53.00%	73.67%	82.96%	**88.23%**	**89.75%**	**90.78%**
AGMA-LN	**52.24%**	52.02%	**53.58%**	**53.99%**	**53.42%**	72.57%	**83.60%**	87.60%	89.15%	89.93%

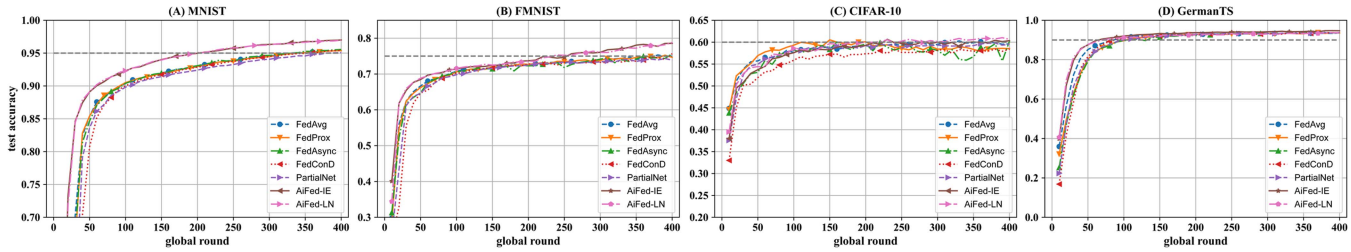

Fig. 9. Accuracy curve of AiFed-IE/LN and the five baselines in IID. (A) MNIST, (B) FMNIST, (C) CIFAR-10, and (D) GermanTS.

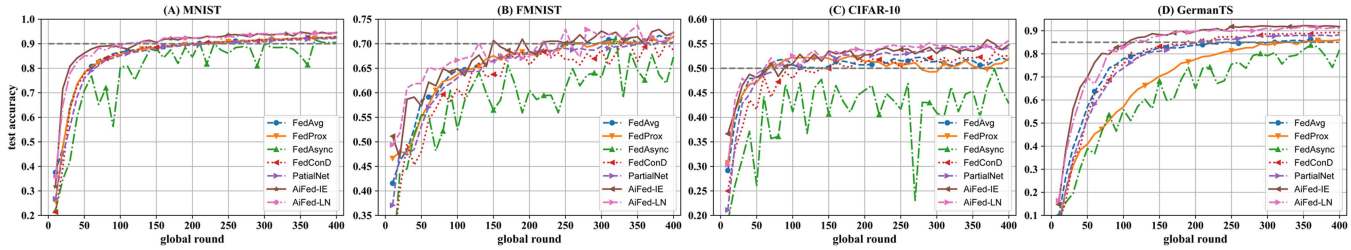

Fig. 10. Accuracy curve of AiFed-IE/LN and the five baselines in Non-IID. (A) MNIST, (B) FMNIST, (C) CIFAR-10, and (D) GermanTS.

D. The Evaluation of the Integrated Mechanism

The optimal performance of AiFed can be analyzed by using ALMU and AGMA jointly. First, according to the results in the "Integration" group of Table V, AiFed outperforms the five baselines in learning accuracy and speed regardless of IID or Non-IID settings in all four datasets, specifically:

- *Learning accuracy and speed in IID cases:* Since IID cases alleviate the data heterogeneity of AFDM, AiFed and the five baselines have tied performance as shown by the accuracy curves in Fig. 9. However, the improvement of AiFed on MNIST and FMNIST is more pronounced than on CIFAR-10 and GermanTS, as a sharp accuracy boost can be observed in Fig. 9(A) and (B). Moreover, compared to the five baselines, AiFed can first reach the target accuracies with an average acceleration of about 22.16%. Such an improvement can also be observed in Fig. 11(A). It shows that AiFed can better tackle the temporal differences in AFDM to accommodate clients with different computation and communication capabilities, and also in various working statuses and availabilities;
- *Learning accuracy and speed in Non-IID cases:* Even though Non-IID cases are more complex than IID cases, the improvement brought by AiFed is more profound. As compared to the five baselines, the learning accuracy increases and the learning speed accelerates by about 3.05% and 37.81% on average, respectively. Moreover, as shown in Figs. 10 and 11(B), such an improvement is consistent across all four datasets, illustrating the high generability of AiFed in supporting not only simple but also complex AFDM tasks. Since in real-world scenarios, Non-IID data sensed from individuals commonly exist in ubiquitous IoT systems and services, AiFed is more suitable to meditate and consolidate duplicated and biased inter-knowledge extracted from AFDM clients.

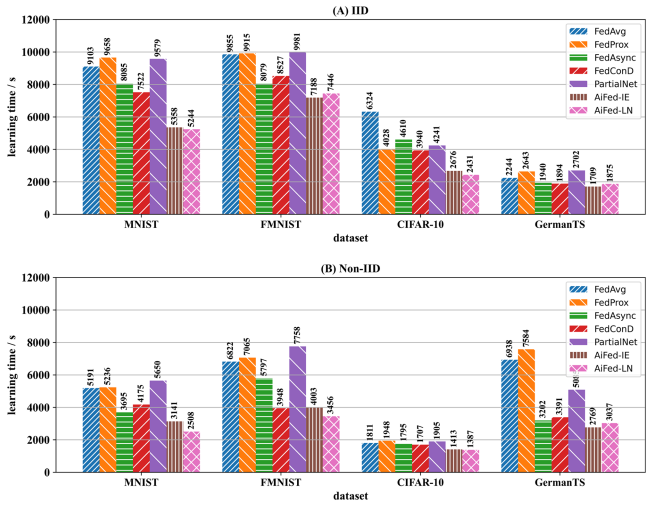

Fig. 11. Learning time comparison when reaching the target accuracy in (A) IID cases and (B) Non-IID cases.

Moreover, as revealed in Fig. 9, AiFed has steady accuracy growth curves and can also converge quickly in IID cases. As for Non-IID cases (Fig. 10), although the curves of AiFed fluctuate slightly compared with IID settings, they can still first reach target accuracies compared to the five baselines and maintain good performances until the end of learning. The results show that AiFed has satisfactory and stable training performance.

Finally, as illustrated in Fig. 12, the communication costs of AiFed and the five baselines when reaching corresponding target accuracies in IID and Non-IID cases are analyzed. It shows

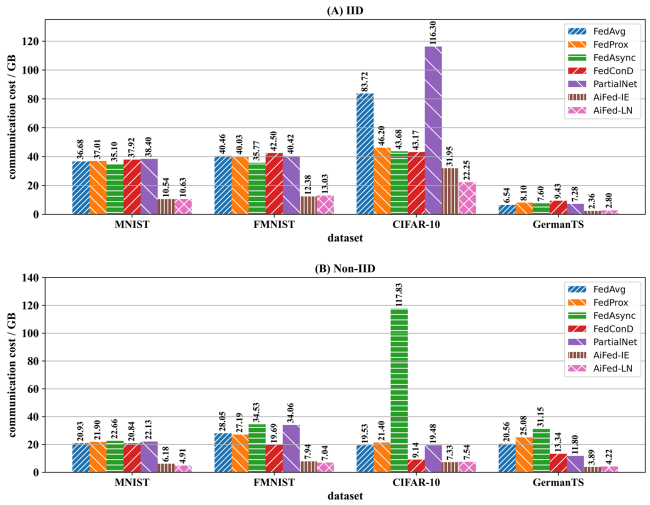

Fig. 12. Communication cost comparison when reaching the target accuracy in (A) IID cases and (B) Non-IID cases.

that AiFed is more cost-efficient than the five baselines with an average cost reduction of 61.76% and 56.88% in IID and Non-IID cases, respectively. Such a significant improvement is achieved through the layer-wise model uploading and aggregation processes proposed and implemented in AiFed, as only selective layers of local models (with high potential to improve the overall performance) will be uploaded and aggregated for a global model. Through the combined effects of layer-wise model uploading and aggregation, AiFed can first reach the target accuracy with the lowest communication cost in all IID and Non-IID cases.

In summary, the proposed mechanism AiFed inherits the advantages of ALMU and AGMA, and compared with the five baselines, it can remarkably improve the overall performance for both IID and Non-IID cases, namely, 1) saving communication costs on average by about 61.76% (IID) and 56.88% (Non-IID), 2) increasing learning accuracy on average by about 1.66% (IID) and 3.05% (Non-IID), and 3) boosting learning speed on average by about 22.16% (IID) and 37.81% (Non-IID). Such a result shows the merits of AiFed in addressing the heterogeneity in AFDM.

VI. Conclusion and Future Work

AFDM starts to be studied to bridge the isolated and fragmented data islands restricted by the growing concerns on data security and user privacy for collaborative knowledge mining. To comprehensively address the heterogeneity at each client (i.e., Non-IID data and uneven computing power) and the server (i.e., differences of information staleness and richness in received local models), this paper proposes an adaptive and integrated mechanism for AFDM, called AiFed. Specifically, first, it defines the layer-wise local model uploading and global model aggregation processes as the optimization foundation. Second, it designs two adaptive strategies, i.e., ALMU to determine the uploading of the model layer based on its representational consistency before and after the local training, and AGMA to aggregate received local model layers based on an adaptive weight measuring the information staleness and richness. Finally, it integrates ALMU and AGMA to support AFDM with overall learning performance (in terms of learning cost, accuracy, and speed) improved.

As compared with five state-of-the-art methods based on four standard datasets in IID and Non-IID settings, first, ALMU can maintain the highest accuracy growth rate per unit cost of communication to be cost-efficient. Second, AGMA can significantly boost the performance of global model aggregation for both IID and Non-IID cases to be task-generic. Finally, AiFed combines and inherits the advantages of ALMU and AGMA that can 1) reduce the communication cost on average by about 61.76% (IID) and 56.88% (Non-IID), 2) increase learning accuracy on average by about 1.66% (IID) and 3.05% (Non-IID), and 3) accelerates learning speed on average by about 22.16% (IID) and 37.81% (Non-IID).

In the future, the connectivity of AFDM clusters will be studied to further optimize client-server communication by creating a hierarchical and robust network with high speed but low traffic. Second, on top of the adaptive weight, a sensitivity indicator of temporal and informative attributes will be investigated to steer the model aggregation direction and stabilize the accuracy growth for each learning round. Finally, in the case of knowledge scarcity caused by either lack of local data or unwillingness to participate, small-sample approaches, i.e., federated meta-learning, will be explored to mine transferable and customizable knowledge, and in the meanwhile, related incentive mechanisms will be designed to ensure the fairness and attractiveness of AFDM.

References

[1] L. You, J. He, W. Wang, and M. Cai, "Autonomous transportation systems and services enabled by the next-generation network," *IEEE Netw.*, vol. 36, no. 3, pp. 66–72, May/Jun. 2022.

[2] B. Yang, X. Cao, C. Yuen, and L. Qian, "Offloading optimization in edge computing for deep-learning-enabled target tracking by internet of UAVs," *IEEE Internet Things J.*, vol. 8, no. 12, pp. 9878–9893, Jun. 2021.

[3] L. You, B. Tunçer, R. Zhu, H. Xing, and C. Yuen, "A synergetic orchestration of objects, data, and services to enable smart cities," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10496–10507, Dec. 2019.

[4] L. You, B. Tunçer, and H. Xing, "Harnessing multi-source data about public sentiments and activities for informed design," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 2, pp. 343–356, Feb. 2019.

[5] P. Zhou, K. Wang, L. Guo, S. Gong, and B. Zheng, "A privacy-preserving distributed contextual federated online learning framework with Big Data support in social recommender systems," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 3, pp. 824–838, Mar. 2021.

[6] L. Zhang, T. Zhu, P. Xiong, W. Zhou, and P. Yu, "A robust game-theoretical federated learning framework with joint differential privacy," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 4, pp. 3333–3346, Apr. 2023.

[7] Y. Chen, X. Sun, and Y. Jin, "Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 10, pp. 4229–4238, Oct. 2020.

[8] L. Wang et al., "Towards understanding learning representations: To what extent do different neural networks learn the same representation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 9607–9616.

[9] L. You, S. Liu, B. Zuo, C. Yuen, D. Niyato, and H. Poor, "Federated and asynchronized learning for autonomous and intelligent things," *IEEE Netw.*, early access, Oct. 09, 2023, doi: 10.1109/MNET.2023.3321519.

[10] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[11] R. S. Antunes, C. André da Costa, A. Küderle, I. A. Yari, and B. Eskofier, "Federated learning for healthcare: Systematic review and architecture proposal," *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 4, pp. 1–23, 2022.
[12] T. Yu et al., "Learning context-aware policies from multiple smart homes via federated multi-task learning," in *Proc. IEEE/ACM 5th Int. Conf. Internet-of-Things Des. Implementation*, 2020, pp. 104–115.
[13] S. Ramaswamy, R. Mathews, K. Rao, and F. Beaufays, "Federated learning for emoji prediction in a mobile keyboard," 2019, *arXiv:1906.04329*.
[14] W. Yang, Y. Zhang, K. Ye, L. Li, and C.-Z. Xu, "FFD: A federated learning based method for credit card fraud detection," in *Proc. Int. Conf. Big Data*, Springer, 2019, pp. 18–32.
[15] J. Xu, Z. Xu, P. Walker, and F. Wang, "Federated patient hashing," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 6486–6493.
[16] B. Yang et al., "A joint energy and latency framework for transfer learning over 5G industrial edge networks," *IEEE Trans. Ind. Informat.*, vol. 18, no. 1, pp. 531–541, Jan. 2022.
[17] L. You, J. He, J. Zhao, and J. Xie, "A federated mixed logit model for personal mobility service in autonomous transportation systems," *Systems*, vol. 10, no. 4, pp. 1–20, 2022.
[18] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, "Asynchronous online federated learning for edge devices with non-IID data," in *Proc. IEEE Int. Conf. Big Data*, 2020, pp. 15–24.
[19] C. Qiao, K. N. Brown, F. Zhang, and Z. Tian, "Federated adaptive asynchronous clustering algorithm for wireless mesh networks," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 3, pp. 2610–2627, Mar. 2023.
[20] 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 Things J.*, vol. 9, no. 23, pp. 24199–24211, Dec. 2022.
[21] S. Liu, H. Qu, Q. Chen, W. Jian, R. Liu, and L. You, "AFMeta: Asynchronous federated meta-learning with temporally weighted aggregation," in *Proc. IEEE Smartworld, Ubiquitous Intell. Comput., Scalable Comput. Commun., Digit. Twin, Privacy Comput., Metaverse Auton. Trusted Veh.*, 2022, pp. 641–648.
[22] S. Chen, C. Shen, L. Zhang, and Y. Tang, "Dynamic aggregation for heterogeneous quantization in federated learning," *IEEE Trans. Wireless Commun.*, vol. 20, no. 10, pp. 6804–6819, Oct. 2021.
[23] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, PMLR, 2017, pp. 1273–1282.
[24] A. K. Sahu, T. Li, M. Sanjabi, M. Zaheer, A. Talwalkar, and V. Smith, "On the convergence of federated optimization in heterogeneous networks," 2018, *arXiv:1812.06127*.
[25] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," 2019, *arXiv:1903.03934*.
[26] Y. Chen, Z. Chai, Y. Cheng, and H. Rangwala, "Asynchronous federated learning for sensor data with concept drift," in *Proc. IEEE Int. Conf. Big Data*, 2021, pp. 4822–4831.
[27] G. Paragliola and A. Coronato, "Definition of a novel federated learning approach to reduce communication costs," *Expert Syst. Appl.*, vol. 189, 2022, Art. no. 116109.
[28] N. Shlezinger, M. Chen, Y. C. Eldar, H. V. Poor, and S. Cui, "Federated learning with quantization constraints," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 8851–8855.
[29] M. Kamp et al., "Efficient decentralized deep learning by dynamic model averaging," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discov. Databases*, Springer, 2018, pp. 393–409.
[30] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Blockchain empowered asynchronous federated learning for secure data sharing in internet of vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4298–4311, Apr. 2020.
[31] Z. Chen, W. Liao, K. Hua, C. Lu, and W. Yu, "Towards asynchronous federated learning for heterogeneous edge-powered Internet of Things," *Digit. Commun. Netw.*, vol. 7, pp. 317–326, 2021.
[32] C.-H. Hu, Z. Chen, and E. G. Larsson, "Device scheduling and update aggregation policies for asynchronous federated learning," in *Proc. IEEE 22nd Int. Workshop Signal Process. Adv. Wireless Commun.*, 2021, pp. 281–285.
[33] D. Ye, R. Yu, M. Pan, and Z. Han, "Federated learning in vehicular edge computing: A selective model aggregation approach," *IEEE Access*, vol. 8, pp. 23920–23935, 2020.
[34] J. Pang, Y. Huang, Z. Xie, Q. Han, and Z. Cai, "Realizing the heterogeneity: A self-organized federated learning framework for IoT," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3088–3098, Mar. 2021.
[35] X. Wang, R. Li, C. Wang, X. Li, T. Taleb, and V. C. Leung, "Attention-weighted federated deep reinforcement learning for device-to-device assisted heterogeneous collaborative edge caching," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 154–169, Jan. 2021.
[36] S. S. Diwangkara and A. I. Kistijantoro, "Study of data imbalance and asynchronous aggregation algorithm on federated learning system," in *Proc. Int. Conf. Inf. Technol. Syst. Innov.*, 2020, pp. 276–281.
[37] O. A. Wahab, A. Mourad, H. Otrok, and T. Taleb, "Federated machine learning: Survey, multi-level classification, desirable criteria and future directions in communication and networking systems," *IEEE Commun. Surv. Tuts.*, vol. 23, no. 2, pp. 1342–1397, Second Quarter 2021.
[38] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-iid data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3400–3413, Sep. 2020.
[39] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2019, pp. 7252–7261.
[40] M. M. Amiri and D. Gündüz, "Machine learning at the wireless edge: Distributed stochastic gradient descent over-the-air," *IEEE Trans. Signal Process.*, vol. 68, pp. 2155–2169, 2020.
[41] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 3320–3328.
[42] A. Krizhevsky, "One weird trick for parallelizing convolutional neural networks," 2014, *arXiv:1404.5997*.
[43] S. Liu, Q. Chen, and L. You, "Fed2A: Federated learning mechanism in asynchronous and adaptive modes," *Electronics*, vol. 11, no. 9, pp. 1–17, 2022.
[44] M. Raghu, J. Gilmer, J. Yosinski, and J. Sohl-Dickstein, "SVCCA: Singular vector canonical correlation analysis for deep learning dynamics and interpretability," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6078–6087.
[45] J. Mehrer, C. J. Spoerer, N. Kriegeskorte, and T. C. Kietzmann, "Individual differences among deep neural network models," *Nature Commun.*, vol. 11, no. 1, pp. 1–12, 2020.
[46] N. Kriegeskorte, M. Mur, and P. A. Bandettini, "Representational similarity analysis-connecting the branches of systems neuroscience," *Front. Syst. Neurosci.*, vol. 2, pp. 1–28, 2008.
[47] S. Lee, T. Zhang, and A. S. Avestimehr, "Layer-wise adaptive model aggregation for scalable federated learning," in *Proc. AAAI Conf. Artif. Intell.*, 2023, pp. 8491–8499.
[48] Q. Ma, Y. Xu, H. Xu, Z. Jiang, L. Huang, and H. Huang, "FedSA: A semi-asynchronous federated learning mechanism in heterogeneous edge computing," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3654–3672, Dec. 2021.

Linlin You (Senior Member, IEEE) received the PhD degree in computer science from the University of Pavia, in 2015. He is an associate professor with the School of Intelligent Systems Engineering, Sun Yat-sen University, and also a research affiliate with the Intelligent Transportation System Lab, Massachusetts Institute of Technology. He was a senior postdoc with the Singapore-MIT Alliance for Research and Technology and a research fellow with the Architecture and Sustainable Design Pillar of Singapore University of Technology and Design. He published more than 60 journal and conference papers in the research fields of smart cities, service orchestration, multi-source data fusion, machine learning, and federated learning.

Sheng Liu received the BEng degree from the School of Intelligent Systems Engineering, Sun Yat-sen University, China, in 2021, where he is currently working toward the master's degree. His research interests include artificial intelligence, federated learning, machine learning, and their applications in smart cities and intelligent transportation systems.

Tao Wang (Member, IEEE) received the bachelor's degree from Northwestern Polytechnical University, the master's degree from Beihang University, and the PhD degree in control science and engineering from the Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University. He is currently an associate professor with the School of Intelligent Systems Engineering, Sun Yat-Sen University. He was a researcher with the China Academy of Launch Vehicle Technology. His research interests include intelligent sensing algorithms, cooperative control of multi-domain unmanned systems, and cognitive and bionic intelligence.

Bingran Zuo received the BSc degree from Zhejiang University, in 1993, and the PhD degree from Shanghai Jiao Tong University, in 1998. He is the deputy director (Technology) of Rehabilitation Research Institute of Singapore (RRIS), Nanyang Technological University. He was the program manager of Future Urban Mobility (FM) with the Singapore-MIT Alliance for Research and Technology Centre (SMART). His research interests include robotics and automation, advanced manufacturing technologies, and data-driven robotics with human-in-the-loop.

Yi Chang (Senior Member, IEEE) is the dean of the School of Artificial Intelligence, Jilin University, Changchun, China. He was elected as a Chinese National distinguished professor, in 2017 and an ACM distinguished scientist, in 2018. Before joining academia, he was the technical vice president with Huawei Research America, and the research director with Yahoo Labs. He is the author of two books and more than 100 papers in top conferences or journals. His research interests include information retrieval, data mining, machine learning, natural language processing, and artificial intelligence. He won the Best Paper Award on KDD'2016 and WSDM'2016. He has served as a Conference General Chair for WSDM'2018, SIGIR'2020 and WWW'2025.

Chau Yuen (Fellow, IEEE) received the BEng and PhD degrees from Nanyang Technological University, Singapore, in 2000 and 2004, respectively. He was a post-doctoral fellow with Lucent Technologies Bell Labs, Murray Hill, in 2005, and a visiting assistant professor with The Hong Kong Polytechnic University, in 2008. From 2006 to 2010, he was with the Institute for Infocomm Research, Singapore. From 2010 to 2023, he was an associate professor with the Engineering Product Development Pillar, Singapore University of Technology and Design. Since 2023, he has been with the School of Electrical and Electronic Engineering, Nanyang Technological University. He was a recipient of the Lee Kuan Yew Gold Medal, the Institution of Electrical Engineers Book Prize, the Institute of Engineering of Singapore Gold Medal, the Merck Sharp and Dohme Gold Medal, and twice a recipient of the Hewlett Packard Prize. He received IEEE Asia Pacific Outstanding Young Researcher Award, in 2012 and IEEE VTS Singapore Chapter Outstanding Service Award, in 2019. He currently serves as an editor for *IEEE Transactions on Vehicular Technology*, *IEEE System Journal*, and *IEEE Transactions on Network Science and Engineering*, where he was awarded as *IEEE Transactions on Network Science and Engineering* Excellent Editor Award and Top associate editor for *IEEE Transactions on Vehicular Technology* from 2009 to 2015. He also served as the guest editor for several special issues, including *IEEE Journal on Selected Areas in Communications*, *IEEE Wireless Communications Magazine*, *IEEE Communications Magazine*, *IEEE Vehicular Technology Magazine*, *IEEE Transactions on Cognitive Communications and Networking*, and *Elsevier Applied Energy*. He is a distinguished lecturer of IEEE Vehicular Technology Society and also a Highly Cited Researcher by Clarivate Web of Science. He has 3 US patents and published more than 500 research papers at international journals or conferences.