A Triple-Step Asynchronous Federated Learning Mechanism for Client Activation, Interaction Optimization, and Aggregation Enhancement

Linlin You, *Member, IEEE*, Sheng Liu, Yi Chang, *Senior Member, IEEE*, and Chau Yuen, *Fellow, IEEE*

***Abstract*—Federated Learning in asynchronous mode (AFL) is attracting much attention from both industry and academia to build intelligent cores for various Internet of Things (IoT) systems and services by harnessing sensitive data and idle computing resources dispersed at massive IoT devices in a privacy-preserving and interaction-unblocking manner. Since AFL is still in its infancy, it encounters three challenges that need to be resolved jointly, namely: 1) how to rationally utilize AFL clients, whose local data grow gradually, to avoid over-learning issues; 2) how to properly manage the client-server interaction with both communication cost reduced and model performance improved; and finally, 3) how to effectively and efficiently aggregate heterogeneous parameters received at the server to build a global model. To fill the gap, this article proposes a triple-step asynchronous federated learning mechanism (TrisaFed), which can: 1) activate clients with rich information according to an informative client activating strategy (ICA); 2) optimize the client–server interaction by a multiphase layer updating strategy (MLU); and 3) enhance the model aggregation function by a temporal weight fading strategy (TWF), and an informative weight enhancing strategy (IWE). Moreover, based on four standard data sets, TrisaFed is evaluated. As shown by the result, compared with four state-of-the-art baselines, TrisaFed can not only dramatically reduce the communication cost by over 80% but also can significantly improve the learning performance in terms of model accuracy and training speed by over 8% and 70%, respectively.**

***Index Terms*—Aggregation enhancement, asynchronous federated learning (AFL), client activation, federated learning (FL), interaction optimization.**

I. Introduction

With the rapid development of the Internet of Things (IoT), a versatile network can be created by connecting substantial devices in various domains, e.g., transportation [1], healthcare [2], governance [3], environment [4], etc., to support the sensing of diverse data, the mining of valuable insights and, in turn, the lifting of service quality [5], [6]. Along with this trend, the data processing paradigm starts shifting from centralized data integration to decentralized parameter aggregation, since more data silos restricted by data security and user privacy are emerging, especially at IoT edges, preventing the sharing and exchanging of sensitive data [7]. As a result, the centralized learning approach may become inefficient and ineffective in harnessing these silos to build the intelligent core of IoT.

As a novel solution, federated learning (FL) is proposed to train a global model by utilizing local data and computing resources of learning participants in a collaborative and privacy-preserving way [8], and starts to be used in many fields to support domain-specific tasks, e.g., in smart home to learn user behaviors [9]; in personal application to assist keyboard inputting [10]; in smart manufacturing to detect product defects [11]; in open banking to perform fraud detection [12]; in digital healthcare to predict hospitalization rate [13], etc. Moreover, FL can also be categorized into two forms according to its working mode, namely: 1) the synchronous FL (SFL), which requires all participants to have the same pace and 2) the asynchronous FL (AFL), which enables its participants to work separately [14], [15]. While comparing the two forms, AFL is more critical and suitable for ubiquitous IoT systems and services, as it is more efficient and effective to support devices, whose capability and availability may change over time and space [16].

In general, compared with SFL, AFL is still in its infancy, and several challenges are emerging. First, it is challenging to resolve the overlearning issue in the context that local data accumulated vary among learning participants in the 4V characteristics, i.e., volume, velocity, variety, and value [17], [18]. Moreover, the communication resource of an IoT device is shared among various services and may become scarce for AFL. Hence, it is critical to optimize network usage without compromising the overall learning performance [19]. Finally, the importance of locally trained models may vary among each other and shall be adequately measured on the server to support model aggregation efficiently and effectively [20].

Accordingly, several solutions are proposed: 1) to improve model accuracy by measuring the diversity of local resources and the heterogeneity of local models and 2) to save

Manuscript received 7 March 2022; revised 28 May 2022; accepted 1 July 2022. Date of publication 5 July 2022; date of current version 21 November 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62002398; in part by the Singapore Ministry of National Development and the National Research Foundation, Prime Minister's Office through the Cities of Tomorrow (CoT) Research Programme under Award COT-V2-2021-1; and in part by the Collaborative Innovation Center for Transportation of Guangzhou under Grant 202206010056. *(Corresponding author: Chau Yuen.)*

Linlin You and Sheng Liu 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).

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 Engineering Product Development, Singapore University of Technology and Design, Singapore 487372 (e-mail: yuenchau@sutd.edu.sg).

Digital Object Identifier 10.1109/JIOT.2022.3188556

communication cost by reducing the package size and data uploading frequency [19], [21], [22]. However, there is a dilemma in improving the accuracy growth and saving the communication cost [23]. As an initial attempt to solve it, a temporally weighted aggregation strategy and a layerwise model update strategy are designed and used jointly to improve both model accuracy and training time [19]. However, a mechanism that can: 1) tackle the overlearning issue by activating clients with appropriate data; 2) reduce communication cost by optimizing the client–server interaction pattern; and 3) enhance model aggregation by addressing the heterogeneity of local models is still missing.

To fill the gap, this article presents a triple-step AFL mechanism (TrisaFed), which can address the above challenges in three consecutive steps, as follows.

Step 1) *Client Activation:* It selects clients, whose sensing data grow gradually, from an AFL cluster according to an informative client activating strategy (ICA), which enables top-k participants with rich information based on an index sorted by a self-relative entropy (SRE).

Step 2) *Communication Optimization:* It simplifies the client–server interaction in learning deep neural networks (DNNs) according to a multiphase layer updating strategy (MLU), which optimizes the update frequency of shallow and deep layers.

Step 3) *Aggregation Enhancement:* It handles local parameters according to two weighted strategies, namely, a temporal weight fading strategy (TWF), which aggregates local parameters with a fading weight subject to their created time, and an informative weight enhancing strategy (IWE), which lifts local parameters encoded with rich information by weights measured by the label number (LN) or information entropy (IE).

Moreover, compared with conventional AFL studies, TrisaFed has the following strengths.

1) It can support a more realistic scenario, where the local data can be sensed and accumulated gradually along with model training, e.g., new data will continuously appear before data integrity is reached [18].
2) It can achieve the state-of-the-art performance in terms of communication cost, model accuracy, and learning speed against four baselines (i.e., FedAvg [8], FedProx [24], FedAsync [20], and ASO-Fed [18]) as evaluated by using four standard data sets.

The remainder of this article is organized as follows. Section II summarizes the related work and then the proposed mechanism is presented and tested in Sections III and IV, respectively. Finally, Section V concludes the work and sketches the future research directions.

II. Related Work

In general, as shown in Fig. 1, AFL consists of two entities, namely: 1) clients, each of which owns data to train its local model separately and 2) a server, which controls the learning process to build a global model [20]. Moreover, it contains

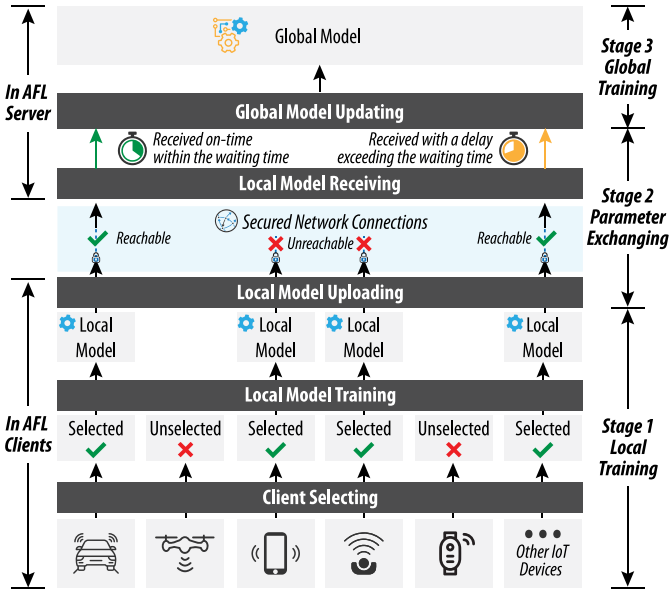

Fig. 1. Overview of AFL.

three working stages, namely: 1) a local training stage, in which a set of potential clients is activated as learning participants to train their local models; 2) a parameter exchanging stage, in which related learning parameters are exchanged between the activated clients and the server through secured connections (based on 5G, Wi-Fi, etc.); and 3) a global training stage, in which the server updates the global model by aggregating currently or previously received, but unused local parameters [15]. Note that in AFL, a communication round starts when the clients are selected to train their local models and ends when predefined conditions are reached, e.g., when the server receives a certain number of updates [20].

In general, while training a global model, AFL may encounter three critical challenges, as follows.

C1) *Client Activation in Local Training:* How to avoid the overlearning issue by selecting learning participants with appropriate data.

C2) *Interaction Optimization in Parameter Exchanging:* How to reduce communication cost by optimizing the client–server interaction.

C3) *Aggregation Enhancement in Global Training:* How to boost model performance by enhancing the aggregation process.

To solve these challenges, several solutions are proposed and, accordingly, summarized in the following sections.

A. Solutions About Client Activation

Since AFL in the ubiquitous IoT is commonly supported by various devices, it is challenging to select an appropriate set of clients to train the global model with the overlearning issue addressed. To tackle such a challenge, several client activation strategies are proposed by considering the diversity of local resources. Specifically, in the beginning, strategies by using a single attribute to measure the diversity are studied, e.g., the one (called TiFL) using training time per round to categorize and select clients with similar performance [25]; and the one

TABLE I
OVERALL EVALUATION OF REVIEWED SOLUTIONS
(● SUPPORTED ○ NOT SUPPORTED)

Related Work	C1*	C2*	C3*
Z. Chai, et al. [25]	●	○	○
I. Mohammed, et al. [26]	●	○	○
F. Lai, et al. [27]	●	○	○
Z. Chen, et al. [28]	●	○	○
S. Abdulrahman, et al. [29]	●	○	○
L.Yu, et al. [30]	●	○	○
N. Agarwal, et al. [31]	○	●	○
M. Kamp, et al. [32]	○	●	○
Y. Chen, et al. [19]	○	●	●
Z. Chai, et al. [22]	○	●	●
N. Chen, et al. [33]	○	○	●
D. Ye, et al. [34]	●	○	●
X. Wang, et al. [35]	●	○	●
This paper (TrisaFed)	●	●	●

* C1, C2, and C3 represent Client Activation, Interaction Optimization, and Aggregation Enhancement respectively.

using test accuracy in an online stateful FL heuristic to choose the best candidate clients [26].

Furthermore, strategies by utilizing more than one criterion to represent the diversity are designed, e.g., a participant selection algorithm called Oort, which proposes a concept of client utility calculated based on local training speed and accuracy growth [27]; an FL client activating framework, which measures local computing and communicating conditions at the same time [28]; a multicriteria-based client selection approach named FedMCCS, which considers not only communication overhead but also imbalanced data distribution [29]; and a resource management algorithm, which estimates energy consumption and communication latency jointly [30].

B. Solutions About Interaction Optimization

Since AFL works in a distributed environment, an efficient and effective interaction among clients and collaborators is required to reduce the consumption of limited communication resources at the edges. In general, three approaches, i.e., the update compression, the frequency reduction, and the pattern optimization [19], [22], are widely discussed. Specifically, the first two approaches can save the cost by minimizing the amount of data to be transmitted [31], [32]. However, they may cause side effects on the performance of the global model.

In contrast, the last approach can not only avoid side-effects with useful information preserved but also reduce network traffic with training performance improved, e.g., based on the fact that: 1) the shallow and deep layers of DNNs learn general and *ad hoc* features, respectively, and 2) shallow layers are more crucial, but with fewer parameters than deep layers [36], a layerwise asynchronous model update strategy is designed to optimize the interaction by reducing the update frequency of deep layers [19].

C. Solutions About Aggregation Enhancement

Since the importance of local parameters may vary among each other, various weighting strategies are proposed to address such a heterogeneity for a performance boost during the aggregation of local models, e.g., based on the assumption that up-to-date parameters may be superior, a temporally weighted strategy is designed to merge parameters with a weight derived from the difference between their created and received time [19]; based on the observation that a frequent aggregation on parameters from clients with shorter per-round response latency may make the global model biased, heuristic weights are used to steer and balance the training across clients [22].

Moreover, since parameters learned from local data may vary in the richness of information, some solutions are studied to process them distinctively, e.g.: 1) a selection-driven strategy is implemented to train the global model by choosing "fine" local parameters [34]; 2) an attention-based strategy is designed to reduce model bias by optimizing the aggregation weight [35]; and 3) a mutual information (MI)-driven mechanism is proposed to avoid the overtraining on duplicated data by monitoring the consistency of training direction [33].

D. Summary

The abilities of related solutions to address issues in the three stages are evaluated as shown in Table I. It shows that a mechanism that can address obstacles in the three stages simultaneously is still undiscussed, specifically to: 1) stabilize and boost the learning growth by selecting learning participants with appropriate data; 2) reduce the communication cost but without damaging the model performance by optimizing client–server interactions; and 3) aggregate local parameters by jointly weighing on their temporal and informative attributes to train a global model. To fill the gap, this article presents TrisaFed, which can train DNNs with both the communication cost reduced and model performance improved.

III. TRISAFED: TRIPLE-STEP AFL MECHANISM

As illustrated in Fig. 2, TrisaFed consists of three consecutive steps, namely: 1) the client activation step to select learning participants by measuring information changes according to an ICA; 2) the interaction optimization step to initialize the communication pattern according to an MLU; and 3) the aggregation enhancement step to update the global model according to two weighted strategies, i.e., a TWF and an IWE. For the sake of readability, the abbreviations and notations used in TrisaFed are summarized in Tables II and III, respectively.

A. Client Activation

As shown in Fig. 2(a), this step consists of three phases, namely: 1) a sensing phase to accumulate data in each client; 2) an activating phase to select clients with sufficient new

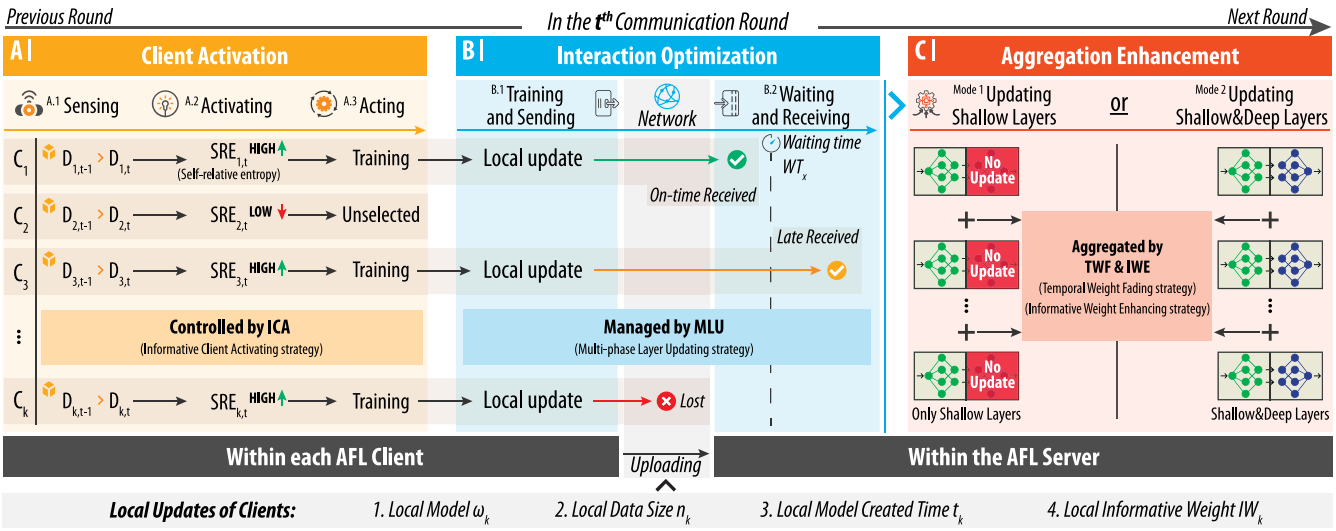

Fig. 2. Overall workflow of TrisaFed in a communication round: a) client activation to select participants according to an ICA; b) interaction optimization to support the client–server communication according to an MLU; and c) aggregation enhancement to aggregate local parameters by using a TWF, or an IWE, or both of them.

TABLE II
ABBREVIATIONS USED IN TRISAFED

Abbreviation	Description
ICA	Informative Client Activating Strategy
MLU	Multi-phase Layer Updating Strategy
TWF	Temporal Weight Fading Strategy
IWE	Informative Weight Enhancing Strategy
SRE	Self-Relative Entropy
CAI	Client Activation Index
TW	Temporal Weight
IW	Informative Weight
IE	Information Entropy
LN	Label Number

TABLE III
NOTATIONS USED IN TRISAFED

Notation	Meaning
k	The k^{th} client
t	The t^{th} communication round
K_0	The total number of available clients
K	The number of participants in a round
l_n	The total number of data labels
$n_{k,t}$	The data size of a client k in the t^{th} round
In ICA	
$D_{k,t}$	The dataset of a client k in the t^{th} round
$p_{i,t}$	The percentage of the i^{th} class in $D_{k,t}$
$\hat{n_t}$	The total data size of all K_0 clients in the t^{th} round
k_0	The top k_0 clients in the normalized CAI
α	The activating fraction variable
In MLU	
M	The total communication rounds
m	The total rounds in a MLU phase
n	The last n rounds in a MLU phase to update deep layers
d	The parameter size of deep layers
In TWF and IWE	
n_t	The total size of data used in the t^{th} round
t_k	The timestamp when the model of a client k is received
r_k	The round when the model of a client k is generated
ω^s	The model parameter of shallow layers
ω^d	The model parameter of deep layers
ω_k	The local model of the k^{th} client
ω_t	The global model of the t^{th} round

information; and 3) an acting phase to start the local training in selected clients. It is worth noting that if a client is not selected, it will continue to sense data and wait for the activation in future rounds.

To support the selection of clients, an ICA is designed by using a SRE as the activation criteria. Specifically, SRE is designed based on relative entropy or Kullback–Leibler Divergence to measure the self-information change as defined in (1), where $D_{k,t-1}$ and $D_{k,t}$ are the two data sets of a client k in the tth and $(t-1)$th round; $p_{i,t}$ and $p_{i,t-1}$ are the percentage of the ith class in $D_{k,t}$ and $D_{k,t-1}$, respectively; l_n is the total number of labels; and c is a constant (default value 1) introduced to avoid the generation of the infinite value

$$\mathrm{SRE}_{k,t}\left(D_{k,t}||D_{k,t-1}\right)=\sum_{i=1}^{l_n}\left(p_{i,t}\times\frac{\log_2\left(p_{i,t}+c\right)}{\log_2\left(p_{i,t-1}+c\right)}\right). \quad (1)$$

In general, a higher SRE indicates that the current data set is more different from the previous one. Hence, the use of clients with a higher SRE can help avoid overlearning issues, e.g., caused by tragglers (which may lead the model to be trained on similar data sets repeatedly). Accordingly, ICA(α) is designed to select participants based on (2), where K_0 denotes the total

number of available clients; $\overline{\text{CAI}}_{k,t}$ is the normalized client activation index of the kth client in the tth round; $\text{top}k_0(\cdot)$ selects the Top$-k_0$ clients from the set $\{\overline{\text{CAI}}_{k,t} : k \in K_0\}$; $n_{k,t}$ and $\hat{n}_t$ are the data size of the kth client, and all K_0 clients in the tth round, respectively; $\text{CAI}_{k,t}$ denotes the original index of the k_{th} client measured by $\text{SRE}_{k,t}$, and CAI_t is its sum; and finally, α is a hyperparameter, called the activating fraction, and $k_0 = \alpha \times K_0$

$$ICA(\alpha) = \text{Top}k_0\left(\left\{\overline{\text{CAI}}_{k,t} : k \in K_0\right\}, a\right)$$
$$\begin{cases} \overline{\text{CAI}}_{k,t} = \frac{n_{k,t}}{\hat{n}_t} \times \frac{\text{CAI}_{k,t}}{\text{CAI}_t} \\ \text{CAI}_t = \sum_{k=1}^{K_0} \text{CAI}_{k,t} \\ \text{CAI}_{k,t} = e^{\text{SRE}_{k,t}}. \end{cases} \tag{2}$$

In summary, by adjusting α, the mechanism can activate a set of clients with sufficient new information to avoid the overlearning issue. As for the performance of ICA, it is tested in Section IV-B.

B. Interaction Optimization

In the asynchronous environment, as shown in Fig. 2(b), the sending and receiving behaviors of AFL clients and the server are unbounded. Therefore, there are three possible conditions, namely: 1) the unknown failure, in which data packages are lost (no contributions of corresponding clients can be made to the global model); 2) the on-time success, in which local parameters are received within a default waiting time in the tth communication round, and will be used in the current round for model aggregation; and 3) the delayed success, in which, the parameters are received after the default waiting time and will be used in the nearest round.

To reduce the consumption of local resources by optimizing the client–server interaction, an MLU is introduced, which differentiates the update frequency of shallow and deep layers of DNNs. Specifically, MLU, denoted as $MLU(M, m, n)$, divides the total M communication rounds into (M/m) phases. Moreover, within each phase, deep layers only update in the last n rounds to learn *ad hoc* features periodically, while shallow layers update in all m rounds to learn general features consistently.

Besides the saving on computation resources by training models with fewer layers to be updated, MLU can reduce the communication cost of $([M \times n \times d]/m)$, where d is the parameter size of deep layers. Moreover, fringe benefits can be gained by configuring m and n as evaluated in Section IV-C.

C. Aggregation Enhancement

As shown in Fig. 2(c), to work with $MLU(M, m, n)$, there are two kinds of aggregation modes, namely: 1) the solo update of shallow layers when the current round falls into 1 to $m-n$ rounds of an MLU phase and 2) the joint update of shallow and deep layers when the current round belongs to the last n rounds of an MLU phase.

Moreover, regardless of the aggregation modes, the AFL server will process two kinds of local parameters in a communication round, namely: 1) the one generated in previous communication rounds and unused until now and 2) the one received in the current round. Accordingly, two weighted strategies, namely, a TWF and an IWE, are designed and implemented to aggregate local parameters.

1) TWF: It is defined by (3), where K is the total number of success participants, whose parameters are to be aggregated in the tth communication round; n_t is the data size of all clients; $\overline{TW}_{k,t}$ is the normalized temporal weight of the kth client in the tth round; $TW_{k,t}$ is the original weight, whose sum is TW_t; and r_k denotes the round when the parameter of the kth client is generated

$$\overline{TW}_{k,t} = \frac{TW_{k,t}}{TW_t}$$
$$\begin{cases} TW_{k,t} = \frac{n_{k,t}}{n_t} \times \frac{f(t,k)}{f(t)} \\ TW_t = \sum_{k=1}^{K} TW_{k,t} \\ f(t,k) = \left(\frac{e}{2}\right)^{-(t-r_k)} \\ f(t) = \sum_{k=1}^{K} f(t,k). \end{cases} \tag{3}$$

2) IWE: It is defined by (4), where $IE(D_{k,t})$ is the IE of a local data set $D_{k,t}$; p_i is the percentage of the ith class in $D_{k,t}$; $LN(D_{k,t})$ calculate the total LN in $D_{k,t}$; L_i, which is a boolean value, stands for the presence of the ith label; $\overline{IW}_{k,t}$ is the normalized informative weight of the kth client in the tth round; $IW_{k,t}$ is the original weight, which is measured by either $IE(D_{k,t})$ or $LN(D_{k,t})$, and IW_t is its sum

$$\overline{IW}_{k,t} = \frac{n_{k,t}}{n_t} \times \frac{IW_{k,t}}{IW_t}$$
$$\begin{cases} IW_t = \sum_{k=1}^{K} IW_{k,t} \\ IW_{k,t} = \begin{cases} IE(D_{k,t}), & \text{if IE is used} \\ LN(D_{k,t}), & \text{otherwise} \end{cases} \\ IE(D_{k,t}) = -\sum_{i=1}^{l_n} (p_i \times \log_2 p_i) \\ LN(D_{k,t}) = \sum_{i=1}^{l_n} L_i. \end{cases} \tag{4}$$

In summary, TWF can aggregate parameters of clients based on a normalized weight, whose value decreases according to the temporal difference between the generated and received time of the parameter. Moreover, IWE with LN or IE can lift the weight of related parameters according to their richness of information. Finally, the usage of TWF and IWE-LN/IE can significantly enhance the learning performance, which is analyzed in Section IV-D.

D. Integration of the Four Strategies

TrisaFed consists of the four proposed strategies, namely: 1) ICA in client activation; 2) MLU in interaction optimization; and 3) TWF; and 4) IWE in aggregation enhancement. It is worth noting that TrisaFed can use the four strategies: 1) individually to improve the corresponding step or 2) jointly to achieve the maximum savings in resource consumption and improvements in learning a federated model.

While using them jointly, ICA and MLU are two initialization strategies, namely: 1) $ICA(\alpha)$ defines the activating proportion of AFL clients in each communication round and 2) $MLU(M, m, n)$ configures the update frequency of shallow and deep layers. Moreover, TWF and IWE with IE or LN are two runtime strategies to aggregate local parameters with a

Algorithm 1 TrisaFed at the Clients

Initialization: $ICA(\alpha)$ and $MLU(M, m, n)$ are defined.
a. Activating clients according to $ICA(\alpha)$ in each communication round;
b. In total M rounds, there are $\frac{M}{m}$ phases;
In each phase:
1) Shallow layers are updated in all m rounds;
2) Deep layers are updated at the last n rounds.

for each client $k \in K_0$ in parallel **do**
  ICA_k is calculated
  $n_{k,t} \leftarrow |D_{k,t}|$ ▷ get the data size of $D_{k,t}$
end for
Transmiting ICA_k and $n_{k,t}$ to the server
The local training starts if selected, otherwise break
IW_k is calculated by $IE(D_{k,t})$ or $LN(D_{k,t})$
$r_k \leftarrow$ *the timestamp of current round*
Local model ω_k is trained by using local data
Transmiting local parameters consisting of ω_k, IW_k and r_k to the server

Algorithm 2 TrisaFed at the Server

Receiving ICA_k and $n_{k,t}$ in an AFL round t
Activating $top - k_0$ clients ▷ $k_0 = \alpha \times K_0$
Receiving local parameters and starting the aggregation after a default waiting time
Calculating TW and IW according to TWF and IWE
Calculating the normalized weight $\overline{TW\_IW}_{k,t}$
if $1 \le t\%m \le m - n$ **then** ▷ Aggregation mode 1
  Updating the global model $\omega_{t+1} \leftarrow \sum_{k=1}^{K}(\overline{TW\_IW}_{k,t} \times \omega_k^s) + \omega_t^d$
  ▷ ω^s shallow layers, and ω^d deep layers
else if $m - n < t\%m \le m$ **then** ▷ Aggregation mode 2
  Updating the global model $\omega_{t+1} \leftarrow \sum_{k=1}^{K}(\overline{TW\_IW}_{k,t} \times \omega_k)$
end if

normalized weight $\overline{TW\_IW}_{k,t}$ as defined in the following:

$$\overline{TW\_IW}_{k,t} = \frac{n_{k,t}}{n_t} \times \frac{TW_{k,t} \times IW_{k,t}}{TW_t \times IW_t}. \tag{5}$$

Moreover, TrisaFed needs to be deployed at both the client and server sides, as follows.

1) *TrisaFed at the Clients:* As shown in Algorithm 1, first, all AFL clients are initialized by $ICA(\alpha)$ and $MLU(M, m, n)$. Note that the hyperparameters in terms of α, M, m, and n are configured according to the training command received from the server. Second, according to ICA, clients are activated to start the local training concurrently. Finally, in each activated client, local parameters are learned and transmitted to the server.
2) *TrisaFed at the Server:* As shown in Algorithm 2, first, the server will activate $top - k_0$ clients, and in the meanwhile, wait for local updates from the clients. Second, when the aggregation is triggered, i.e., by a default timer, the normalized weight $\overline{TW\_IW}_{k,t}$ is calculated. Finally, by using the normalized weight, the global model is updated in two modes to merge only the shallow layers or both the shallow and deep layers.

As the above algorithms run in parallel, it can simplify the computational complexity (CC) of ICA, MLU, TWF, and IWE in a communication round, specifically:

1) ***O*(n_{max}) *for ICA:*** since each client can scan its local data simultaneously, the CC of ICA is reduced to $O(n_{\max})$, where $n_{\max}$ is the maximum size of local data among all available AFL clients;
2) ***O*(1) *for MLU:*** the application of MLU requires a simple conditional check to determine whether to upload deep layers, therefore, its CC is $O(1)$;
3) ***O*(K) *for TWF:*** since the calculation of temporal weights happens at the server side, and it needs to scan all local parameters to be processed, the CC of TWF is $O(K)$, where K is the number of participants in a round;
4) ***O*(max(n_{max}, K)) *for IWE:*** IWE calculates the informative weight by scanning not only the local data on the client side but also the parameters to be aggregated on the server side, hence, its CC is $O(\max(n_{\max}, K))$.

Finally, as the TrisaFed implements a sequential integration of ICA, MLU, TWF, and IWE, the overall CC of TrisaFed is $O(\max(n_{\max}, K))$ (which is linear). Particularly, when decomposing it into the client and server: 1) the CC of TrisaFed on the client side is $O(n_{\max})$, which is lower than the CC of training DNNs (i.e., which can be polynomial) and 2) the CC of TrisaFed on the server side is $O(K)$, which is similar to the conventional aggregation function (merging all the local updates received in a round). Even though the application of TrisaFed needs additional local parameters to be calculated and transmitted, it consumes relatively low computation and communication resources. In turn, it is cost efficient for AFL.

In summary, TrisaFed can address the issue of overlearning, optimize the interaction between the server and clients, and enhance the performance of the global model by using four dedicated strategies. As for the overall performance of TrisaFed, it is analyzed in Section IV-E.

IV. Performance Evaluation

The performance of the proposed mechanism is evaluated in four groups, namely: 1) "Activation" group, in which the configuration of α in *ICA* is analyzed; 2) "Interaction" group, in which the effects of m and n of *MLU* are tested; 3) "Aggregation" group, in which TWF and IWE with *IE* or *LN* are compared; and finally, 4) "Integration" group, in which the ultimate performance of TrisaFed is revealed.

A. Common Settings

First, four standard data sets, i.e., Fashion-MNIST[1] (FMNIST), Large-scale CelebFaces Attributes Data set[2] (CelebA), CIFAR-10,[3] and German Traffic Signs Data set[4] (GermanTS), are used to train a model with two CNN layers

[1]https://github.com/zalandoresearch/fashion-mnist
[2]http://mmlab.i.e.,cuhk.edu.hk/projects/CelebA.html
[3]https://www.cs.toronto.edu/ kriz/cifar.html
[4]https://bitbucket.org/jadslim/german-traffic-signs

TABLE IV
SUMMARY OF DATA SETS AND MODELS TO BE TRAINED

Term	FM-NIST	CelebA	CIFAR-10	GermanTS
Task complexity	Low	Low	High	High
Number of clients	60	60	20	20
Total training samples	60,000	60,000	50,000	34,799
Total testing samples	10,000	10,000	10,000	12,630
Number of classes	10	2	10	43
Sample size	$28 \times 28 \times 1$	$84 \times 84 \times 3$	$32 \times 32 \times 3$	$32 \times 32 \times 3$
First CNN layer	$32 \times 5 \times 5 \times 1$	$32 \times 5 \times 5 \times 1$	$128 \times 5 \times 5 \times 1$	$32 \times 5 \times 5 \times 1$
Second CNN layer	$64 \times 5 \times 5 \times 1$	$64 \times 5 \times 5 \times 1$	$256 \times 5 \times 5 \times 1$	$64 \times 5 \times 5 \times 1$
Fully connected layer	256	256	256	256
Local epoch	2	2	5	2
Batch size	48	48	48	48
Learning rate	0.003	0.003	0.003	0.003

TABLE V
PARAMETERS USED TO ASSIGN DATA DYNAMICALLY

Variable	Description	Value
A_{min}	Minimum amount of data	200
A_{max}	Maximum amount of data	2,000
L_{min}	Minimum number of labels per partition	2/1/2/3*
L_{max}	Maximum number of labels per partition	6/2/6/12*
I_{min}	Minimum proportion of initial data	5%
I_{max}	Maximum proportion of initial data	15%
$R1_{min}$	Rate 1: Lower bound of new data per round	3%
$R1_{max}$	Rate 1: Upper bound of new data per round	5%
$R2_{min}$	Rate 2: Lower bound of new data per round	0.1%
$R2_{max}$	Rate 2: Upper bound of new data per round	0.2%

* Since the four common datasets have different number of labels, the values of L_{min} and L_{max} are set to be different with an order of FMNIST/CelebA /CIFAR-10/GermanTS.

TABLE VI
ESTIMATED UNIT COST OF COMMUNICATION

Cost type	Parameter size	Estimated cost
$cost_{model}$ for FMNIST	1,693,322	6.45 MB
$cost_{deep}$ for FMNIST	1,641,226	6.26 MB
$cost_{model}$ for CelebA	23,712,926	90.45 MB
$cost_{deep}$ for CelebA	23,659,266	90.25 MB
$cost_{model}$ for CIFAR-10	10,269,194	39.17 MB
$cost_{deep}$ for CIFAR-10	9,440,010	36.01 MB
$cost_{model}$ for GermanTS	2,424,299	9.24 MB
$cost_{deep}$ for GermanTS	2,370,603	9.04 MB

and one fully connected layer. Table IV summarizes the four data sets and their models to be trained. Note that the model has the same structure adopted from [19].

Second, to mimic how the data are sensed gradually by various IoT devices in the real world, a dynamic data assignment process is designed according to the parameters defined in Table V. Specifically, data from the training sets of FMNIST, CelebA, CIFAR10, and GermanTS are assigned to the clients with global constraints on data volume and LN as represented by A and L. Under these constraints, the local data set of each client grows with: 1) an initial proportion I of the standard data set and 2) a dynamic rate, which is either $R1$ or $R2$. Note that the parameters are designed according to [18] and [19], and testing data sets of the four standard data sets are untouched and used only for performance testing.

Third, as for the performance baseline, it uses four state-of-the-art methods, i.e., FedAvg [8], FedProx [24], FedAsync [20] and ASO-Fed [18].

Finally, as for the evaluation metrics, three measurements are used, as follows.

M1) *Accuracy at the Final Communication Round:* It is calculated according to (6), where TP, TN, FP, and FN represent true positives, true negatives, false positives, and false negatives, respectively.

M2) *Round Number When the Global Model First Reaches the Target Accuracy:* The target accuracies are 70.00%, 85.00%, 45.00%, and 85.00% for FMNIST, CelebA, CIFAR-10, and GermanTS, respectively. They are defined as the mean of the four baselines [27].

M3) *Total Communication Cost When the Target Accuracy Is Reached:* It is calculated according to (7) with estimated unit costs listed in Table VI

$$\text{acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} \tag{6}$$

$$\text{cost}_{\text{total}} = \sum_{t=1}^{T_{\text{target}}} \left(\text{cost}^t_{\text{shallow}} + \text{cost}^t_{\text{deep}} \right) \tag{7}$$

$$\text{s.t.:} \begin{cases} \text{cost}_{\text{model}} = \frac{4 \times \text{model}_{\text{parameter\_size}}}{1024 \times 1024} \\ \text{cost}_{\text{deep}} = \frac{4 \times \text{deep}_{\text{parameter\_size}}}{1024 \times 1024} \\ \text{cost}_{\text{shallow}} = \text{cost}_{\text{model}} - \text{cost}_{\text{deep}}. \end{cases} \tag{8}$$

B. Client Activation

ICA utilizes the hyperparameter α to determine the maximum number of clients to be activated in a communication round. In general, a small α can reduce the overall resource consumption of an AFL cluster, but may have side effects on the global model. To analyze the potential influences, ten standardized ICA variants with $\alpha = 0.1$ to 1.0 are created and tested. It is worth noting that $ICA(1.0)$ means all clients in the AFL cluster will be activated to train the global model.

As shown in Table VII, the results in the Activation Group indicate that compared to the four best baselines (which are FedProx for FMNIST, ASO-Fed for CelebA, FedProx for CIFAR-10, and FedAvg for GermanTS, respectively), the use of ICAs can, surprisingly, benefit the overall performance with a maximum improvement in accuracy by about 7.43%, and learning speed by about 50.59%. It shows that the activation of clients with richer information in each communication round can tackle the overlearning issue effectively and efficiently.

Moreover, as shown in Fig. 3 the heatmap of accuracy growth, two distinctive patterns can be detected, as follows.

TABLE VII
SUMMARY OF EVALUATION RESULTS

Group	Strategy	Accuracy at 600^{th} round				Target accuracy reached at x^{th} round			
		FMNIST	CelebA	CIFAR-10	GermanTS	FMNIST	CelebA	CIFAR-10	GermanTS
Baseline	$FedAvg$	70.83%	85.06%	47.89%	**88.48%**	359	588	374	**207**
	$FedProx$	**71.35%**	85.08%	**48.13%**	88.10%	**280**	562	336	476
	$FedAsync$	68.87%	84.64%	47.64%	85.48%	-	-	422	487
	$ASO-Fed$	69.11%	**85.22%**	46.92%	83.14%	303	**510**	**306**	-
$Activation$	$ICA(0.1)$	**76.65%**	86.62%	27.10%	43.00%	**204**	259	-	-
	$ICA(0.2)$	74.20%	86.26%	43.06%	57.70%	254	259	-	-
	$ICA(0.3)$	72.72%	**86.80%**	47.76%	68.86%	282	**252**	227	-
	$ICA(0.4)$	72.82%	84.94%	47.04%	76.34%	315	403	302	-
	$ICA(0.5)$	73.10%	85.56%	48.46%	83.55%	261	381	253	-
	$ICA(0.6)$	72.41%	85.44%	**49.51%**	87.88%	264	443	**222**	153
	$ICA(0.7)$	71.53%	85.56%	48.63%	89.40%	269	440	296	151
	$ICA(0.8)$	71.49%	85.54%	48.55%	90.76%	284	448	285	150
	$ICA(0.9)$	70.95%	85.36%	47.93%	**91.72%**	309	485	303	**148**
$Interaction$	$MLU(600,2,1)$	70.52%	85.18%	48.18%	92.12%	351	512	351	177
	$MLU(600,3,1)$	**71.25%**	**85.60%**	**49.43%**	92.45%	**328**	**506**	**321**	199
	$MLU(600,4,1)$	71.00%	85.30%	48.01%	91.30%	333	576	366	197
	$MLU(600,5,1)$	70.98%	84.86%	49.04%	92.35%	343	-	333	171
	$MLU(600,6,1)$	70.76%	85.10%	47.96%	**92.55%**	350	559	335	**170**
	$MLU(600,7,1)$	70.61%	84.84%	48.37%	92.44%	360	-	336	197
	$MLU(600,8,1)$	70.82%	85.04%	48.63%	91.88%	354	563	348	196
	$MLU(600,9,1)$	70.51%	84.86%	47.62%	91.87%	377	-	360	183
	$MLU(600,10,1)$	70.50%	84.22%	48.06%	91.55%	377	-	363	203
	$MLU(600,15,5)$	70.67%	84.66%	48.61%	92.10%	392	-	340	192
$Aggregation$	TWF	**73.11%**	86.74%	49.96%	**92.87%**	**187**	**220**	**214**	**133**
	$IWE-IE$	72.12%	**87.34%**	49.60%	92.19%	209	280	289	135
	$IWE-LN$	71.89%	86.24%	**50.02%**	92.27%	204	473	278	149
$Integration$	$TrisaFed(IE)*$	**77.34%**	**88.20%**	50.08%	**94.13%**	103	**94**	151	**117**
	$TrisaFed(LN)*$	77.20%	87.76%	**50.46%**	93.30%	**83**	149	**134**	**117**

* In "Integration" Group, two variants of TrisaFed, namely TrisaFed(LN) and TrisaFed(IE) are evaluated, and they are only different on the method (using LN or IE) to calcuate the informative weight. Moreover, both TrisaFed(LN) and TrisaFed(IE) use 1) $ICA(0.5)$ and $MLU(600,3,1)$ for FMNIST; 2) $ICA(0.5)$ and $MLU(600,3,1)$ for CelebA; 3) $ICA(0.6)$ and $MLU(600,3,1)$ for CIFAR-10; and 4) $ICA(0.9)$ and $MLU(600,6,1)$ for GermanTS respectively. These defualt settings of ICA and MLU are made according to the evaluation results in "Activation" and "Interaction" groups, which are analyzed in Section IV-B and Section IV-C respectively.

1) *Pattern 1 (More Stable Around the Middle for Simple Tasks):* As shown in Fig. 3(a) and (b), in the first half of communication rounds represented by Areas 1–3, ICAs in Areas 2 and 3 gain a higher accuracy growth than the ones in Area 1 for FMNIST, but for CelebA, ICAs in Area 1 is better than Areas 2 and 3. While comparing Areas 4–6 with Areas 1–3, their performances change diagonally, as Area 4 in FMNIST and Area 6 in CelebA

TABLE VIII
ACCURACY COMPARISON BETWEEN $MLU(600, 3, 1)$ AND $MLU(600, 15, 5)$

Round	FMNIST		CelebA		CIFAR-10		GermanTS	
	MLU1*↑	MLU2*	MLU1↑	MLU2	MLU1↑	MLU2	MLU1↑	MLU2
100	**58.87%**	56.86%	**64.00%**	62.12%	**35.18%**	33.45%	**77.23%**	74.30%
200	**68.29%**	66.89%	**73.66%**	72.08%	**41.58%**	40.05%	**87.18%**	86.40%
300	**69.80%**	69.08%	**78.56%**	78.30%	**44.84%**	43.23%	**89.50%**	89.07%
400	**70.44%**	70.24%	**82.72%**	81.44%	**46.92%**	45.53%	**91.45%**	90.19%
500	**70.86%**	70.00%	**83.96%**	83.64%	**48.71%**	47.76%	**91.90%**	90.08%
600	**71.25%**	70.67%	**85.60%**	84.66%	**49.34%**	48.61%	**92.54%**	92.10%

* MLU1 and MLU2 denote $MLU(600, 3, 1)$ and $MLU(600, 15, 5)$ respectively.

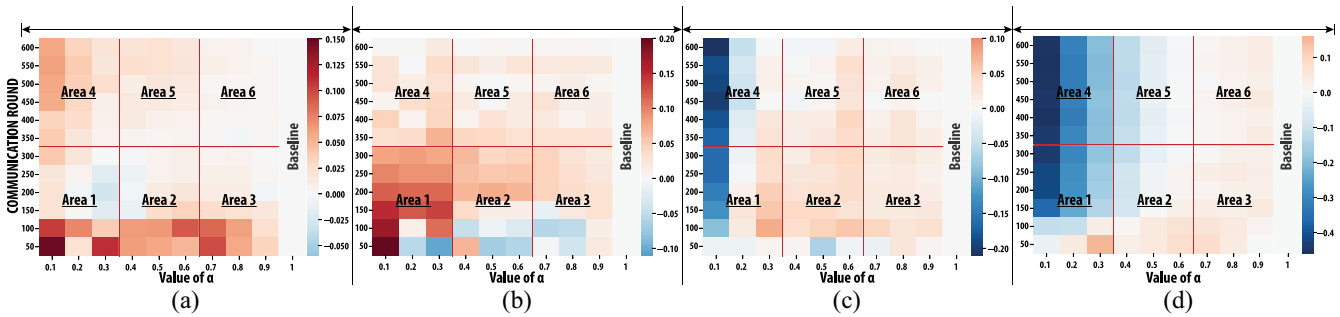

Fig. 3. Heatmap of accuracy growth of ICA variants in (a) FMNIST, (b) CelebA, (c) CIFAR-10, and (d) GermanTS.

overtake the rest areas. This phenomenon indicates that ICAs with α in the medium range from 0.4 to 0.6 can maintain more stable and consistent accuracy growth to support simple learning tasks.

2) *Pattern 2 (More Stable Above the Middle for Complex Tasks):* As shown in Fig. 3(c) and (d), most ICAs can gain a consistent accuracy growth, except ICAs with α close to 0 (reflected by Areas 1 and 4), whose performances become worse than the baseline along with the increase of communication rounds. It indicates that a larger α (which can activate more clients to provide sufficient data) is more suitable for complex tasks (as they need more data samples to train an accurate model).

Finally, since $ICA(0.5)$, $ICA(0.6)$, and $ICA(0.9)$ can not only achieve the best performance but also remain a stable accuracy growth for FMNIST and CelebA, CIFAR-10, and GermanTS, respectively, they are used as the default settings of ICAs in the Integration group.

C. Interaction Optimization

Since MLU can reduce the communication cost in the case that m is bigger than n, ten normalized MLU variants with $M = 600$, $m = 1$ *to* 10, and $n = 1$, and an additional MLU with $m = 15$ and $n = 5$ are created to analyze how m and n may influence the performance. Note that $MLU(600, 1, 1)$ is the case that no optimization is made.

As a result, two observations can be made as follows.

1) *Observation About m (It Is Beneficial to Update Shallow and Deep Layers Separately):* According to the results listed in Table VII Interaction Group, MLUs can outperform the baselines in CIFAR-10 (with $m = 3$), CelebA (with $m = 3$), and GermanTS (with $m = 6$), respectively, and for FMNIST, MLU with $m = 3$ can also catch up with the best baseline FedProx, and surpass the rest three baselines. It shows that MLU can, indeed, improve the model performance by separating the update of shallow and deep layers.

2) *Observation About n (It Is Helpful to Update Shallow and Deep Layers Actively):* As illustrated in Table VIII, MLU(600, 3, 1) and MLU(600, 15, 5) are compared. Even though they have the same client–server interaction ratio of (m/n) (it means they have the same communication cost), $MLU(600, 3, 1)$ can always stay ahead of MLU(600, 15, 5) in all cases during the learning. It indicates that the update of deep layers needs to be active and less delayed with shallow layers to maximize the potential benefits of MLU.

Finally, among MLU variants, for FMNIST, CelebA, and CIFAR-10, $MLU(600, 3, 1)$ can not only achieve the highest accuracy of 71.25%, 85.60% and 49.43% but also first reach the target accuracy at the 328th, 506th, and 321st round, and for GermanTS, $MLU(600, 6, 1)$ can also achieve the best performance in both training accuracy (92.55%) and speed (at 170th round). Accordingly, they are used as the default setting of MLUs in the Integration group.

D. Aggregation Enhancement

In general, according to the evaluation results summarized in Table VII Aggregation Group, when comparing the performance of TWF, IWE-IE, and IWE-LN with the four baselines, two improvements can be observed as follows.

1) *Model Accuracy:* It improves about: a) 2.47% against the best baseline FedProx (71.35%) in FMNIST; b) 2.49% against the best baseline ASO-Fed (85.22%) in CelebA;

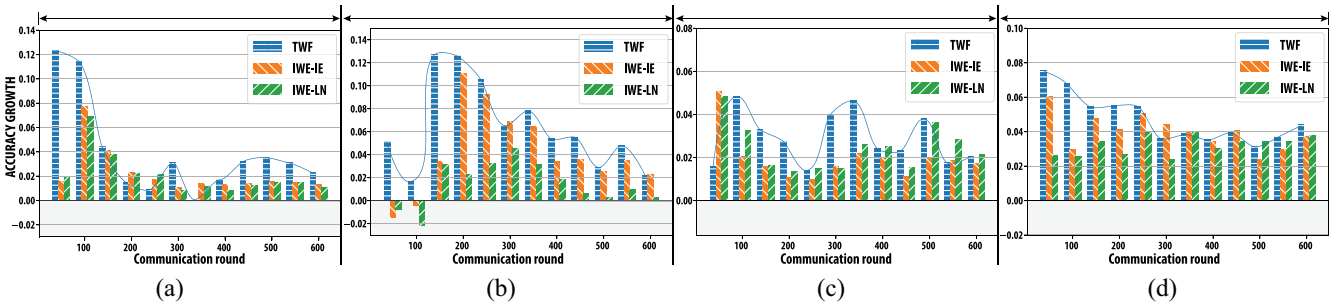

Fig. 4. Accuracy growth of TWF, IWE-IE, and IWE-LN in (a) FMNIST, (b) CelebA, (c) CIFAR-10, and (d) GermanTS.

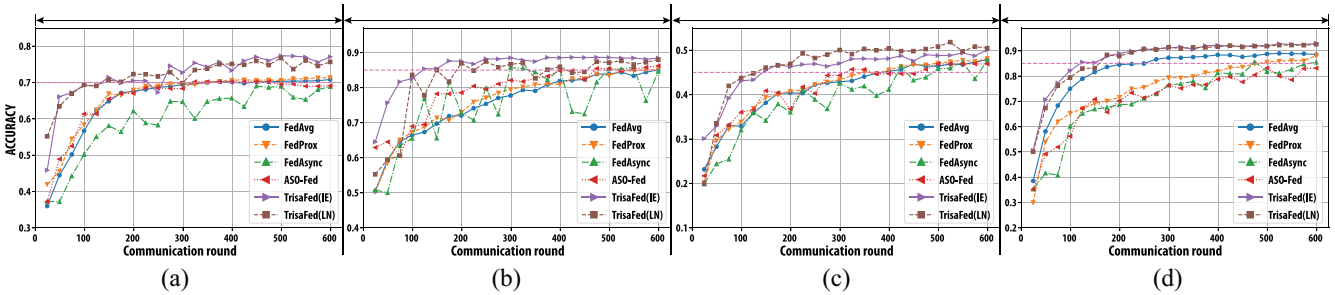

Fig. 5. Performance comparison in (a) FMNIST, (b) CelebA, (c) CIFAR-10, and (d) GermanTS.

c) 3.93% against the best baseline FedProx (48.13%) in CIFAR-10; and finally, d) 4.96% against the best baseline FedAvg (88.48%) in GermanTS.

2) *Training Speed:* It accelerates about: a) 33.21% against the best baseline FedProx in FMNIST; b) 56.86% against the best baseline ASO-Fed in CelebA; c) 30.07% against the best baseline ASO-Fed in CIFAR-10; and finally, d) 35.75% against the best baseline FedAvg in GermanTS.

Moreover, their accuracy growth (using FedAvg as the baseline) is also analyzed as shown in Fig. 4. In general: 1) both TWF and IWE-IE/LN can consistently improve the model accuracy and 2) TWF performs better than IWE-IE/LN in all four cases (as indicated by the blue line in Fig. 4). Besides that, a decrease of two IWEs can be found in Fig. 4(b). It is because: 1) CelebA has binary labels; 2) the local data of clients grows gradually; and 3) IWE may give more informative weights to a few clients with massive but similar data at the early stage. However, when more data are assigned to the clients, the accuracy growth of IWE can catch up and overcome the baseline.

Finally, according to the above analyses, the proper usage of temporal and informative attributes during the model aggregation can improve both model accuracy and training speed.

E. Full Integration

The optimal performance of TrisaFed is tested by integrating the four strategies and using them simultaneously. Since IW can be calculated according to IE or LN, there are two TrisaFed variants, denoted as TrisaFed(IE) and TrisaFed(LN).

In general, as summarized in Table VII Integration group, both TrisaFed(IE) and TrisaFed(LN) can outperform the best baseline in each data set with: 1) model accuracy improved by about 8.40% and 8.20%, respectively, and 2) training speed accelerated by about 81.57% and 70.78%, respectively.

Moreover, the capability of TrisaFed is further analyzed in three aspects as follows.

1) *Ability to Improve Training Performance*: As illustrated in Fig. 5, the two TrisaFed variants share a similar curve, which not only grows sharper but also positions higher than the rest four lines in each data set. The same result can also be observed in Table IX. Such an observation highlights the strength of TrisaFed that as an integrated mechanism of AFL, it can train high-performance models rapidly and steadily.
2) *Ability to Support Growing Data Sets:* As demonstrated in Fig. 6, with the growth of data, TrisaFed is ahead of the four baselines in all the data sets. It implies that TrisaFed can better support the real-world scenario, in which IoT devices can collaborate with the server to train a global model based on their local data sensed and accumulated gradually and continuously.
3) *Ability to Reduce Communication Cost:* As shown in Fig. 7, compared with the four baselines, TrisaFed can dramatically reduce the communication cost by about 89.77% for FMNIST, 93.90% for CelebA, 83.26% for CIFAR-10, and 89.84% for GermanTS, respectively. It underlines another strength of TrisaFed that it can optimize the client–server interaction efficiently and effectively.

In summary, by integrating the four proposed strategies, i.e., ICA, MLU, TWF, and IWE, TrisaFed can: 1) dramatically improve training performance in terms of accuracy and speed and 2) significantly reduce the consumption of limited communication resources of IoT devices.

TABLE IX
ACCURACY COMPARISON BETWEEN TRISAFED AND BASELINES

Round	FMNIST						CelebA					
	FedAvg	FedProx	FedAsync	ASO-Fed	TrisaFed (IE)	TrisaFed (LN)	FedAvg	FedProx	FedAsync	ASO-Fed	TrisaFed (IE)	TrisaFed (LN)
100	56.71%	58.40%	50.21%	61.34%	69.16%	**69.38%**	66.44%	67.32%	65.52%	68.96%	82.84%	**83.64%**
200	67.61%	67.95%	62.16%	67.03%	70.51%	**72.32%**	72.04%	72.96%	72.74%	78.58%	**87.50%**	86.98%
300	69.46%	69.67%	64.70%	68.33%	**72.74%**	69.82%	77.76%	79.50%	85.70%	82.10%	**88.48%**	86.68%
400	70.14%	70.64%	65.69%	70.21%	73.32%	**75.11%**	81.88%	80.96%	85.34%	84.86%	**88.52%**	85.96%
500	70.36%	70.60%	68.96%	69.24%	**77.36%**	76.76%	83.90%	83.46%	85.36%	85.40%	**88.60%**	87.16%
600	70.83%	71.35%	68.87%	69.11%	**77.34%**	77.20%	85.06%	85.08%	84.64%	85.22%	**88.20%**	87.76%

Round	CIFAR-10						GermanTS					
	FedAvg	FedProx	FedAsync	ASO-Fed	TrisaFed (IE)	TrisaFed (LN)	FedAvg	FedProx	FedAsync	ASO-Fed	TrisaFed (IE)	TrisaFed (LN)
100	32.91%	33.85%	32.01%	36.04%	43.10%	**43.75%**	74.98%	65.34%	60.13%	56.23%	**82.16%**	79.45%
200	40.39%	40.76%	36.05%	36.81%	46.25%	**46.93%**	84.55%	71.64%	68.72%	70.22%	**88.94%**	87.89%
300	42.90%	43.36%	42.57%	44.34%	47.03%	**50.05%**	87.27%	79.43%	76.41%	76.35%	91.09%	**91.44%**
400	45.26%	45.60%	41.17%	44.70%	48.12%	**50.40%**	88.39%	81.85%	81.28%	78.78%	**92.19%**	91.65%
500	46.42%	47.11%	45.82%	46.92%	48.81%	**50.82%**	88.77%	85.54%	81.76%	83.22%	**91.92%**	91.86%
600	47.89%	48.13%	47.64%	46.92%	50.08%	**50.46%**	88.48%	88.10%	85.48%	83.14%	**94.13%**	93.30%

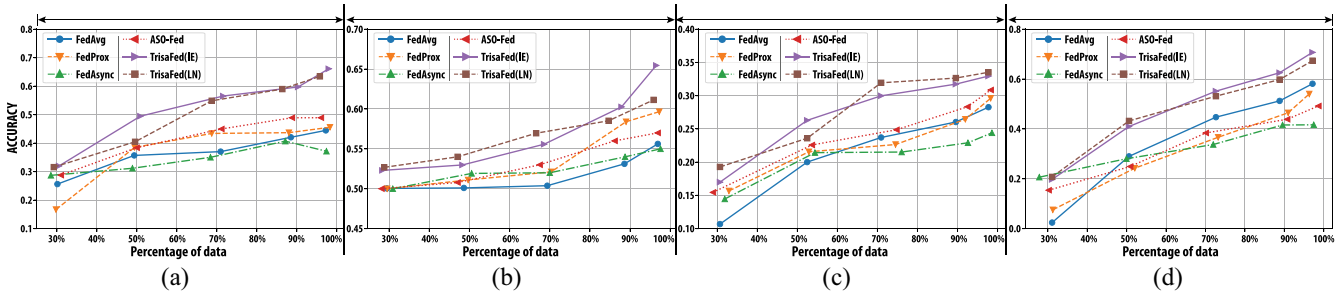

Fig. 6. Accuracy comparison while training data increases in (a) FMNIST, (b) CelebA, (c) CIFAR-10, and (d) GermanTS.

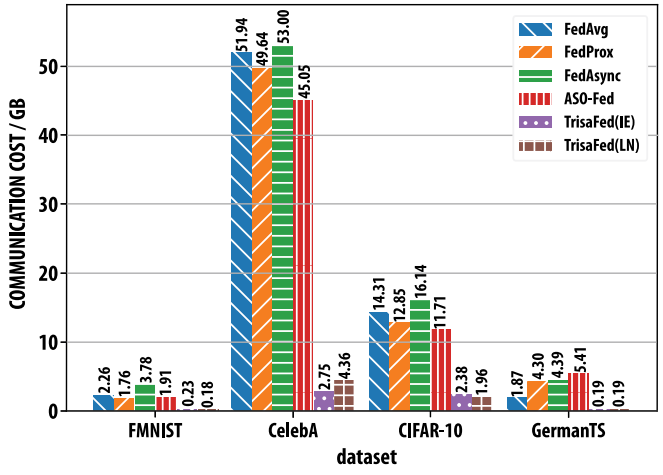

Fig. 7. Comparison of communication cost in the four data sets.

F. Discussion

From the above analyses, the pros and cons of the four strategies can be summarized as follows.

1) *Pros and Cons of ICA:* In general, ICA can significantly remedy the overlearning issue , especially in the scenario that the local data of each client are sensed and accumulated gradually. However, it still needs prior knowledge to configure α. Such that, a heuristic algorithm can be studied to search for an optimal α according to runtime feedback, i.e., the accuracy improvement.
2) *Pros and Cons of MLU:* By adjusting the uploading frequency of deep layers of DNNs, MLU can not only dramatically reduce the communication cost but also slightly improve the model accuracy. However, MLU uploads shallow layers consistently without distinguishing their contributions to the global model. Hence, a method to analyze layerwise changes can be investigated to further optimize the client–server interaction.
3) *Pros and Cons of TWF and IWE:* Both TWF and IWE can remarkably improve the overall performance of AFL in terms of training speed and accuracy. However, their accuracy growth may tremble during the learning because of the temporal and informative uncertainty. Therefore, an adaptive weight can be designed to implement a more efficient and effective aggregation function.

As TrisaFed integrates the four strategies, it inherits their advantages in activating clients with sufficient new information as learning participants to remedy the overlearning issue, optimizing the uploading frequency of model layers to reduce communication cost, and enhancing the aggregation function

to improve the overall performance of the global model. As a result, TrisaFed can support AFL comprehensively and systematically with outstanding performance in terms of communication efficiency, learning speed, training stability, and model accuracy.

V. Conclusion and Future work

To tackle emerging challenges of AFL in client activation, interaction optimization, and aggregation enhancement to efficiently and effectively train an intelligent core of IoT in an unblocking and privacy-preserving way, this article proposes the triple-step AFL mechanism, called TrisaFed, which consists of: 1) ICA to activate AFL participants with rich information in each communication round; 2) MLU to reduce communication cost by adjusting the update frequency of shallow and deep layers of DNNs; and 3) TWF and IWE to enhance model aggregation by utilizing the temporal and informative attributes of local parameters, respectively.

As shown by the evaluation results, first, ICA can improve the model accuracy and learning speed together with the over-learning issue addressed by using an appropriate α. Second, by adjusting m and n, MLU can not only reduce the consumption of local resources but also improve the overall performance to learn DNNs. Third, both TWF and IWE can improve the overall performance in terms of training speed and model accuracy. Finally, while comparing with the four state-of-the-art baselines (i.e., FedAvg, FedProx, FedAsync, and ASO-Fed), TrisaFed (using ICA, MLU, TWF, and IWE jointly) can: 1) achieve the highest accuracy in the four data sets (i.e., FMNIST, CelebA, CIFAR-10, and GermanTS) with a maximum improvement of about 8%; 2) reach the target accuracy about five times faster; and 3) reduce communication cost by about 90%.

In the future, instead of configuring ICA and MLU manually, a dynamic optimizer will be designed and implemented to automatically select α, m, and n in runtime based on a heuristic algorithm, which can search for a local optimum in a space defined by constraints, e.g., the overall training cost, the minimum model accuracy, etc. Moreover, the data heterogeneity among AFL clients will be analyzed to create a unified data quality indicator, through which more accurate informative attributes can be generated and utilized to support client activation and model aggregation. Finally, a collaboration mechanism that manages massive IoT devices according to an adaptive and robust network structure will be investigated to further improve the efficiency and effectiveness of AFL.

Acknowledgment

Any opinion, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the Singapore Ministry of National Development and National Research Foundation, Prime Minister's Office, Singapore.

References

[1] F. Zhu, Y. Lv, Y. Chen, X. Wang, G. Xiong, and F.-Y. Wang, "Parallel transportation systems: Toward IoT-enabled smart urban traffic control and management," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 10, pp. 4063–4071, Oct. 2020.

[2] J. Venkatesh, B. Aksanli, C. S. Chan, A. S. Akyurek, and T. S. Rosing, "Modular and personalized smart health application design in a smart city environment," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 614–623, Apr. 2018.

[3] Y. Bai, Q. Hu, S.-H. Seo, K. Kang, and J. J. Lee, "Public participation consortium blockchain for smart city governance," *IEEE Internet Things J.*, vol. 9, no. 3, pp. 2094–2108, Feb. 2022.

[4] A. Javed, A. Malhi, T. Kinnunen, and K. Främling, "Scalable IoT platform for heterogeneous devices in smart environments," *IEEE Access*, vol. 8, pp. 211973–211985, 2020.

[5] 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.

[6] C. Perera and A. V. Vasilakos, "A knowledge-based resource discovery for Internet of Things," *Knowl. Based Syst.*, vol. 109, pp. 122–136, Oct. 2016.

[7] Y. Li, H. Li, G. Xu, T. Xiang, X. Huang, and R. Lu, "Toward secure and privacy-preserving distributed deep learning in fog-cloud computing," *IEEE Internet Things J.*, vol. 7, no. 12, pp. 11460–11472, Dec. 2020.

[8] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Stat.*, 2017, pp. 1273–1282.

[9] 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 Things Design Implement. (IoTDI)*, 2020, pp. 104–115.

[10] S. Ramaswamy, R. Mathews, K. Rao, and F. Beaufays, "Federated learning for emoji prediction in a mobile keyboard," 2019, *arXiv:1906.04329*.

[11] X. Han, H. Yu, and H. Gu, "Visual inspection with federated learning," in *Proc. Int. Conf. Image Anal. Recognit.*, 2019, pp. 52–64.

[12] G. Long, Y. Tan, J. Jiang, and C. Zhang, "Federated learning for open banking," in *Federated Learning: Privacy and Incentive*, Q. Yang, L. Fan, and H. Yu, Eds. Cham, Switzerland: Springer Int., 2020, pp. 240–254.

[13] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. C. Paschalidis, and W. Shi, "Federated learning of predictive models from federated electronic health records," *Int. J. Med. Inf.*, vol. 112, pp. 59–67, Apr. 2018.

[14] K. Bonawitz *et al.*, "Towards federated learning at scale: System design," in *Proc. Mach. Learn. Syst.*, vol. 1, 2019, pp. 374–388.

[15] P. Kairouz *et al.*, "Advances and open problems in federated learning," 2019, *arXiv:1912.04977*.

[16] T. Zhang, A. Song, X. Dong, Y. Shen, and J. Ma, "Privacy-preserving asynchronous grouped federated learning for IoT," *IEEE Internet Things J.*, vol. 9, no. 7, pp. 5511–5523, Apr. 2022.

[17] S. S. Diwangkara and A. I. Kistijantoro, "Study of data imbalance and asynchronous aggregation algorithm on federated learning system," in *Proc. IEEE Int. Conf. Inf. Technol. Syst. Innov. (ICITSI)*, 2020, pp. 276–281.

[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 (Big Data)*, 2020, pp. 15–24.

[19] 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.

[20] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," in *Proc. OPT 12th Annu. Workshop Optim. Mach. Learn.*, 2020, pp. 1–9.

[21] Q. Xia, W. Ye, Z. Tao, J. Wu, and Q. Li, "A survey of federated learning for edge computing: Research problems and solutions," *High-Confidence Comput.*, vol. 1, no. 1, 2021, Art. no. 100008.

[22] Z. Chai, Y. Chen, A. Anwar, L. Zhao, Y. Cheng, and H. Rangwala, "FedAT: A high-performance and communication-efficient federated learning system with asynchronous tiers," in *Proc. Int. Conf. High Perform. Comput. Netw. Storage Anal.*, 2021, pp. 1–16.

[23] 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. Surveys Tuts.*, vol. 23, no. 2, pp. 1342–1397, 2nd Quart., 2021.

[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] Z. Chai *et al.*, "TiFl: A tier-based federated learning system," in *Proc. 29th Int. Symp. High Perform. Parallel Distrib. Comput.*, 2020, pp. 125–136.
[26] I. Mohammed *et al.*, "Budgeted online selection of candidate IoT clients to participate in federated learning," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5938–5952, Apr. 2021.
[27] F. Lai, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "Oort: Efficient federated learning via guided participant selection," in *Proc. 15th USENIX Symp. Oper. Syst. Design Implement. (OSDI)*, Jul. 2021, pp. 19–35.
[28] 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, no. 3, pp. 317–326, Aug. 2021.
[29] S. AbdulRahman, H. Tout, A. Mourad, and C. Talhi, "FedMCCS: Multicriteria client selection model for optimal IoT federated learning," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4723–4735, Mar. 2021.
[30] L. Yu, R. Albelaihi, X. Sun, N. Ansari, and M. Devetsikiotis, "Jointly Optimizing client selection and resource management in wireless federated learning for Internet of Things," *IEEE Internet Things J.*, vol. 9, no. 6, pp. 4385–4395, Mar. 2022.
[31] N. Agarwal, A. T. Suresh, F. Yu, S. Kumar, and H. B. McMahan, "cpSGD: Communication-efficient and differentially-private distributed SGD," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 7575–7586.
[32] M. Kamp *et al.*, "Efficient decentralized deep learning by dynamic model averaging," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Disc. Databases*, 2018, pp. 393–409.
[33] N. Chen, Y. Li, X. Liu, and Z. Zhang, "A mutual information based federated learning framework for edge computing networks," *Comput. Commun.*, vol. 176, pp. 23–30, Aug. 2021.
[34] 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.
[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] 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.*, vol. 2, 2014, pp. 3320–3328.

Linlin You (Member, IEEE) received the Ph.D. degree in computer science from The University of Pavia, Pavia, Italy, in 2015.

He is an Associate Professor with the School of Intelligent Systems Engineering, Sun Yat-sen University, Guangzhou, China, and also a Research Affiliate with the Intelligent Transportation System Lab, Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, where he was a Senior Postdoctoral Fellow with the Singapore-MIT Alliance for Research and Technology and a Research Fellow with the Architecture and Sustainable Design Pillar, Singapore University of Technology and Design, Singapore. He published more than 40 journal and conference papers in the research fields of smart cities, service orchestration, multisource data fusion, machine learning, and federated learning.

Sheng Liu received the B.Sc. degree from the School of Intelligent Systems Engineering, Sun Yat-sen University, Guangzhou, China, in 2021, where he is currently pursuing the master's degree.

His research interests include federated learning, machine learning, and their applications in smart cities, and intelligent transportation systems.

Yi Chang (Senior Member, IEEE) received the B.Sc. degree from Jilin University, Changchun, China, the double M.Sc. degrees from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, and Carnegie Mellon University, Pittsburgh, PA, USA, respectively, and the Ph.D. degree from the University of Southern California, Los Angeles, CA, USA.

He is the Dean of the School of Artificial Intelligence, Jilin University. His research interests include information retrieval, data mining, machine learning, natural language processing, and artificial intelligence.

Dr. Chang is an Associate Editor of the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, and he served as one of the conference General Chairs for ACM WSDM'2018 and ACM SIGIR'2020. He is an ACM Distinguished Scientist.

Chau Yuen (Fellow, IEEE) received the B.Eng. and Ph.D. degrees from Nanyang Technological University, Singapore, in 2000 and 2004, respectively.

He was a Postdoctoral Fellow with Lucent Technologies Bell Labs, Murray Hill, NJ, USA, in 2005. He was with the Institute for Infocomm Research, Singapore, from 2006 to 2010. He has been with the Singapore University of Technology and Design, Singapore, since 2010.

Dr. Yuen received the IEEE Marconi Prize Paper Award in *Wireless Communications* and the EURASIP Best Paper Award for the *Journal on Wireless Communications and Networking* in 2021, the IEEE Asia–Pacific Outstanding Young Researcher Award in 2012, and the IEEE VTS Singapore Chapter Outstanding Service Award in 2019. He serves as an Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS and IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY. He also served as the Guest Editor for several special issues, including the 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 *Applied Energy* (Elsevier).