Engineering 33 (2024) 178–189

Contents lists available at ScienceDirect

Engineering

journal homepage: www.elsevier.com/locate/eng

Research
Wireless Communications—Article

Low-Cost Federated Broad Learning for Privacy-Preserved Knowledge Sharing in the RIS-Aided Internet of Vehicles

Xiaoming Yuan [a,b], Jiahui Chen [a], Ning Zhang [c], Qiang (John) Ye [d,*], Changle Li [b], Chunsheng Zhu [e], Xuemin Sherman Shen [f]

[a] Qinhuangdao Branch Campus, Northeastern University, Qinhuangdao 066004, China
[b] State Key Laboratory of Integrated Services Networks & the Research Institute of Smart Transportation, Xidian University, Xi'an 710071, China
[c] Department of Electrical and Computer Engineering, University of Windsor, Windsor, ON N9B 3P4, Canada
[d] Department of Electrical and Software Engineering, University of Calgary, Calgary, AB T2N 1N4, Canada
[e] SUSTech Institute of Future Networks, Southern University of Science and Technology, Shenzhen 518055, China
[f] Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada

ARTICLE INFO

Article history:
Received 8 September 2022
Revised 22 January 2023
Accepted 11 April 2023
Available online 14 August 2023

Keywords:
Knowledge sharing
Internet of Vehicles
Federated learning
Broad learning
Reconfigurable intelligent surfaces
Resource allocation

ABSTRACT

High-efficiency and low-cost knowledge sharing can improve the decision-making ability of autonomous vehicles by mining knowledge from the Internet of Vehicles (IoVs). However, it is challenging to ensure high efficiency of local data learning models while preventing privacy leakage in a high mobility environment. In order to protect data privacy and improve data learning efficiency in knowledge sharing, we propose an asynchronous federated broad learning (FBL) framework that integrates broad learning (BL) into federated learning (FL). In FBL, we design a broad fully connected model (BFCM) as a local model for training client data. To enhance the wireless channel quality for knowledge sharing and reduce the communication and computation cost of participating clients, we construct a joint resource allocation and reconfigurable intelligent surface (RIS) configuration optimization framework for FBL. The problem is decoupled into two convex subproblems. Aiming to improve the resource scheduling efficiency in FBL, a double Davidon–Fletcher–Powell (DDFP) algorithm is presented to solve the time slot allocation and RIS configuration problem. Based on the results of resource scheduling, we design a reward-allocation algorithm based on federated incentive learning (FIL) in FBL to compensate clients for their costs. The simulation results show that the proposed FBL framework achieves better performance than the comparison models in terms of efficiency, accuracy, and cost for knowledge sharing in the IoV.

1. Introduction

With the increasing popularity of the Internet of Vehicles (IoVs), various types of data generated for the IoV can be used to support diversified vehicular services (e.g., augmented reality (AR) navigation and traffic control) and customized vehicular operations [1,2]. With onboard communication and computing units, vehicles can process, analyze, and share data with other vehicles to improve the utility of data mining. However, it is easy for raw data sharing to deepen the networking burden and cause data leakage in the IoV. Thus, artificial intelligence (AI) models are used to extract the hidden representation information of the original data and share the data indirectly by sharing the knowledge of the models, which can increase the network bandwidth and improve data security [3]. The knowledge in these data reflects autonomous driving applications (e.g., high-precision map positioning, real-time road condition analysis, etc.), which are inseparable from the learning and sharing of knowledge in the IoV.

Expanding the coverage of knowledge sharing can improve the learning experience and capacity of intelligent vehicles [4]. However, data-driven knowledge sharing also increases the risk of privacy leakage, making it important to ensure the security of client data and reduce the cost of knowledge sharing. Efficient knowledge sharing depends on reliable data transmission to meet the requirements of low delay and high accuracy. Transmission links are vulnerable to interferers and obstacles in pluralized environments. To solve this problem, a reconfigurable intelligent surface (RIS) [5]

* Corresponding author.
E-mail address: qiang.ye@ucalgary.ca (Q. Ye).

https://doi.org/10.1016/j.eng.2023.04.015

technology can be employed. Deploying an RIS on the surface of urban buildings or along the road actively reshapes the wireless environment [6] to establish communication links between source and destination nodes that bypass interference objects in order to improve channel quality [7] and reduce the communication overhead of parameter exchange. With RIS, the communication performance between vehicles and road side units (RSUs) can be improved [8].

Knowledge sharing provide a learning experience based on vehicular behavior data. Sharing data directly can cause privacy leakage, since the raw data of vehicles includes information on travel history and driving habits. Federated learning (FL) [9] is an AI technique that can protect client data privacy while sharing knowledge. An intelligent FL model can be customized through a central server connected to multiple clients, where the clients train local model parameters in a distributed way, and a global model is realized through the aggregation of local model parameters [10,11]. In this way, the original training datasets from clients are not directly sent to third parties, in order to protect data privacy.

Efficient knowledge sharing is easily affected by the mobility of vehicles and the synchronization problem of FL. It is necessary to establish an FL framework with high efficiency and long-term stable benefits for both clients and service servers in order to encourage clients to actively participate in knowledge sharing. Conventional FL frameworks are established based on distributed model training and global aggregation. Local AI model structures (e.g., deep neural networks) can be complex, and their parameter adjustment can be difficult. However, future IoV services require more efficient learning in real-time changing environments. Thus, we employ a novel machine learning paradigm called broad learning (BL) [12], which uses an incremental learning approach to train model parameters instead of using the gradient descent method. Clients adopt the BL model as their local training model to improve the knowledge learning efficiency [13], which is suitable for a large-scale and highly dynamic network.

In this paper, we propose an asynchronous federated BL (FBL) framework to protect data privacy in knowledge sharing. To improve the efficiency of real-time data learning, we design an efficient knowledge learning model—namely, the broad fully connected model (BFCM)—to learn vehicular data knowledge based on BL. To improve resource utilization for FL, we formulate a joint optimization problem of intelligent reflection regulation and resource allocation, which is further decoupled into two subproblems: uplink/downlink transmission rate optimization and phase shift parameter configuration in the RIS. We then present a double Davidon–Fletcher–Powell (DDFP) algorithm as the problem solution to configure the RIS and allocate transmission bandwidth to improve the channel quality between the RSU and vehicles. An RIS-assisted FL (RFL) is proposed to provide rewards to vehicle clients according to their FL contribution in the process of knowledge sharing, based on a federated incentive learning (FIL) algorithm [14]. The contributions of this work are summarized as follows:

(1) **An FL framework design.** We introduce BL and propose an asynchronous FBL framework for knowledge sharing among vehicles, in which an improved BL framework is designed for local data training.

(2) **Joint RIS configuration and resource allocation.** We establish an optimization framework to allocate communication resources and RIS parameters in order to improve the performance of FBL; we also design a DDFP algorithm to solve the optimization problem.

(3) **An FIL algorithm.** To improve resource utilization, we construct RIS-assisted FL based on an FIL algorithm for vehicle–RIS–RSU for knowledge sharing.

The rest of this paper is organized as follows: Section 2 reviews existing works on RFL and BL. Section 3 presents preliminaries on BL, FL, and the building of a cost model in detail. Section 4 presents our proposed FBL framework and BFCM, while the designed RFL and DDFP for FBL are introduced in Section 5. Section 6 presents the simulation results. Section 7 presents concluding remarks and discusses our future work.

2. Related work

2.1. RIS-assisted wireless communications

During knowledge sharing, parameter exchanging requires high-capacity communication channels. In general, RIS can improve the wireless channels between clients and the server during FL [15]. Ni et al. [10] proposed a resource allocation approach for the performance of global aggregation for RIS-assisted channels for FL. Li et al. [16] proposed a distributed algorithm based on FL for configuring all RIS parameters. Wang et al. [17] proposed an RFL approach for over-the-air computation to reduce the mean squared error. Because RIS has the characteristic of intelligent control, when solving a resource allocation problem, the optimization problem is generally decomposed into two subproblems: resource allocation and RIS phase-shift control [18]. Liu et al. [19] greatly improved resource utilization by exploring the available computation resources of vehicles. Most of the works described above decouple a complex resource allocation problem with coupled resource variables into multiple subproblems. Deep learning (DL) or convex optimization techniques are usually used to solve these subproblems.

2.2. FL and BL

As a new machine learning technology, BL has been widely used in text classification, image recognition, and other fields [20]. Peng et al. [21] proposed a fog-assisted BL algorithm for traffic data analysis, which saves bandwidth resources. Guo et al. [22] used broad learning system (BLS) for semi-supervised vehicle type classification. Wei et al. [23] combined BL and reinforcement learning to solve the problem of traffic-light control. These studies show that a BL-based data analysis model in the IoV can efficiently solve the problems of image classification and traffic data analysis in autonomous driving.

FL is an emerging technology to preserve data privacy in distributed machine learning for the IoV [24,25]. Traditional FL paradigms are based on DL models to learn the data generated by the IoV for knowledge sharing. For example, Lu et al. [26] proposed an asynchronous FL scheme for data sharing in the IoV, which can select vehicular nodes according to the quality of learning. Chai et al. [3] proposed a hierarchical knowledge-sharing framework in which an RSU can be used as either a server or a client. However, large-scale mobile data training under the conventional FL architecture cannot meet the requirements of low latency and high efficiency. Thus, some research works have studied the integration of BL with FL for networking [12,27]. However, the question of how to efficiently allocate transmission resources to improve FL performance requires further investigation.

2.3. Incentive mechanisms for FL

During FL, the most important role is the client, so the cost of FL usually comes from the model training of the client [8]. In order to reduce the cost of the system, the main research efforts focus on how to improve the utilization of resources and compress the model [25]. To improve the gain from the client perspective, some researchers add incentive mechanisms to FL—that is, FIL [14]. Le et al. [28] proposed an FIL algorithm based on the auction

approach, which can ensure individual rationality, truthfulness, and efficiency during FL. Liu et al. [29] proposed a privacy-preserving FIL approach that can balance the tradeoff between private data leakage and the model accuracy problem according to the contribution of local model uploading.

3. The FBL framework

The integration of BL and FL can better solve the model training and data privacy leakage problem [13,27]. Considering the channel blocking between vehicles and buildings, RIS is introduced to improve channel quality [30]. Fig. 1 shows the data flow and communication components in FBL for the IoV, in which our asynchronous FBL integrates FL, BL, and RIS. We assume that the set of vehicles is $V = \{1, ..., V\}$, where V is the number of vehicles. The set of local BL model learnable parameters is defined as $\{W_1, ..., W_k\}$, where client vehicle $k \in V$. The benefit is defined as $\{\widehat{u}_1(t), ..., \widehat{u}_k(t)\}$ at time slot t. With the help of an edge computing (EC) server, the RSU can configure the RIS, aggregate parameters, and allocate benefits and resources. To avoid physical interference—such as buildings and other obstacles—interfering with the wireless communication channel, we establish a knowledge sharing channel among the vehicle, RIS, and RSU. The RIS controller can configure the RIS under the EC server and allocate resources to vehicles. The revenues obtained are also allocated by the EC server. Thus, the five main functions of the system model are as follows:

(1) **Local model training.** In each vehicle, we set the BL model as the data learning model. When a user completes the model training, the user will upload the model parameters to the server.

(2) **Parameter aggregation.** The server receives all local models and adopts the given parameter aggregation method to calculate the global model. Finally, it will send the global model to each local client.

(3) **Resource allocation.** Before the next training round, the server can allocate the rewards according to the contribution of each user. In this part, the main purpose of the resource allocation is to allocate the bandwidth during FL.

(4) **RIS configuration.** Because of the high mobility, the RIS parameters should be configured to adapt to the current environment. Thus, we should be able to intelligently control the parameters related to RIS to improve channel performance [31].

(5) **Reward allocation.** During FL, each client needs a certain amount of computation and communication overhead, so we design a reward mechanism based on FIL. This increases the enthusiasm of the client to share knowledge, while giving the client the opportunity to receive benefits.

The detailed processes for knowledge extraction, federated aggregation, and RIS-aided channels for parameter uploading and downloading in FBL are provided below.

3.1. Knowledge extraction from data based on BL

The data collected or generated by vehicles contain valuable driving behavior information and potential driving experience knowledge. Traditional DL models require a considerable amount of training time to extract the information using the gradient descent method. Compared with DL, BL focuses on the increment of features. The generalized BL model includes two layers: the feature mapping layer F and the enhancement node layer E. F mainly uses different weights to map the input data set $\boldsymbol{X}$ into multiple hidden nodes, and the mapped feature $\boldsymbol{Z}$ of the hidden node is the hidden information of $\boldsymbol{X}$. The enhancement node layer mainly enhances the mapped features of the feature mapping layer to further extract the features' correlation of the training set. In generalized BL, it is assumed that the feature mapping layer outputs n feature mappings and each mapping could generate l nodes. The relationship between node i and input data set $\boldsymbol{X}$ is given by the following:

$$\boldsymbol{Z}_i = \phi\left(\boldsymbol{X}\boldsymbol{W}_{e_i} + \beta_{e_i}\right) \tag{1}$$

where $\boldsymbol{W}_{e_i}$ refers to the random generated weights of node i in F, and β_{e_i} refers to the biases of the node i in F. e_i refers to the proper dimension of node i. ϕ is the mapping function in the feature mapping layer. $\boldsymbol{Z}_i$ achieved by Eq. (1) is mapping features of i $(i = 1, 2, 3, ..., n)$ group nodes in F layer. The combination of all output in F is $\boldsymbol{Z}^n = \{\boldsymbol{Z}_1, \boldsymbol{Z}_2, ..., \boldsymbol{Z}_n\}$.

Then, we assume that the number of mapping features in E is m. $\boldsymbol{Z}^n$ undergoes a nonlinear transformation in E to obtain output features. Just like the last layer, $\boldsymbol{H}_j$ $(j = 1, 2, 3, ..., m)$ is the corresponding randomly generated mapping features of node j in E. The relationship between the output of F and the output of E is given by the following:

$$\boldsymbol{H}_j = \xi\left(\boldsymbol{Z}^n\boldsymbol{W}_{h_j} + \beta_{h_j}\right) \tag{2}$$

where $\boldsymbol{W}_{h_j}$ refers to the weights of enhancement node j with h dimensions in E, and β_{h_j} refers to the biases of node j in E. ξ is the

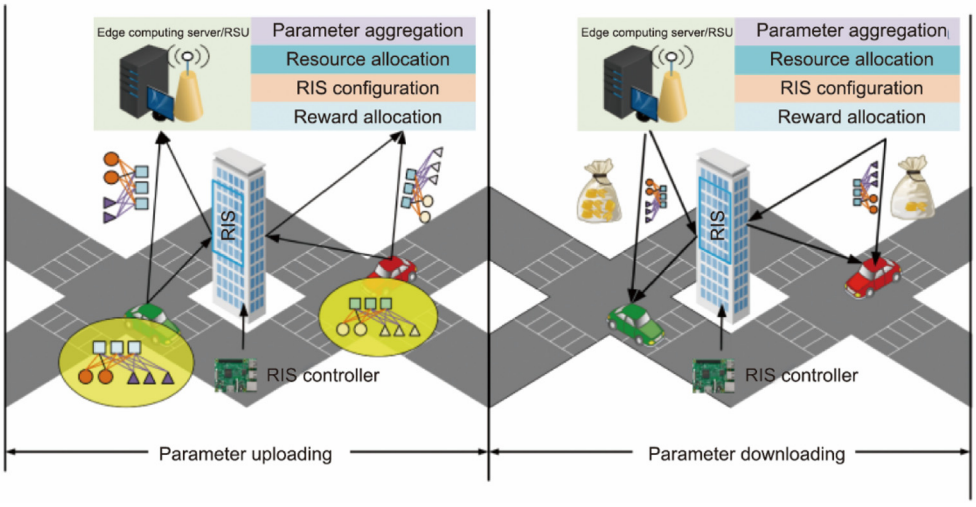

Fig. 1. The proposed asynchronous FBL framework.

mapping function in the enhancement node layer. h_j refers to the proper dimension of node j. i and j could be different according to the complexity of tasks.

We assume that $\boldsymbol{A}$ is $[\boldsymbol{Z}_1, ..., \boldsymbol{Z}_n | \boldsymbol{H}_1, ..., \boldsymbol{H}_m]$, that is the output combination matrix of the feature mapping layer and the enhancement node layer; then, the output $\boldsymbol{Y}$ is obtained as follows:

$$\boldsymbol{Y} = [\boldsymbol{Z}_1, ..., \boldsymbol{Z}_n | \boldsymbol{H}_1, ..., \boldsymbol{H}_m]\boldsymbol{W}^{\mathrm{o}} = \boldsymbol{A}\boldsymbol{W}^{\mathrm{o}} \tag{3}$$

where $\boldsymbol{W}^{\mathrm{o}}$ are the connecting weights for the broad structure that can be updated by incremental learning. The training time is spent in solving $\boldsymbol{W}^{\mathrm{o}}$, and the solution of $\boldsymbol{W}^{\mathrm{o}}$ is more efficient than the gradient descent method [32]. Therefore, deploying BL in FL will greatly improve the efficiency of vehicular task learning.

3.2. The local model

In an FL environment, the global model is set in an EC server, which has powerful computation resources [33]. Some data and tasks to be accessed frequently in the IoV are delegated to the EC server close to clients for execution. At the same time, each provider also has its own model to train data. The parameters of the local model will be shared among clients. In FL, the key is how to aggregate the BL model parameters effectively. In general, the weighted average method is used to aggregate parameters, as follows:

$$\boldsymbol{W}_{\mathrm{g}}(e) = \sum_{k=1}^{V} \frac{\mathrm{size}(D_k)}{\mathrm{size}(D)} \boldsymbol{W}_k(e) \tag{4}$$

where $\boldsymbol{W}_{\mathrm{g}}(e)$ refers to the parameters of the global model at the eth training round in the EC server; $\boldsymbol{W}_k(e)$ refers to the parameters of the BL model in client k; D_k is local raw data of client k while D is the set of local raw data sets including all clients ($D = \{D_1, D_2, \ldots, D_k\}$); size($D_k$) refers to the size of local raw data of client k; and size(D) denotes the size of all raw data.

The purpose of the conventional integration of FL and BL is to learn the weight of the connection between the input and output $\boldsymbol{W}^{\mathrm{o}}$, which refers to the attribution information of the dataset. Each local model uses the aggregated parameter $\boldsymbol{W}_{\mathrm{g}}(e)$ to continue training by means of incremental learning. In the BL model, there are two main forms of incremental learning: adding feature nodes and adding enhancement nodes. Thus, in each epoch training, the parameters are updated by incremental learning. That is:

$$(\boldsymbol{A}')^{+} = \begin{bmatrix} (\boldsymbol{A})^{+} - \boldsymbol{D}\boldsymbol{B}^{\mathrm{T}} \\ \boldsymbol{B}^{\mathrm{T}} \end{bmatrix} \tag{5}$$

where $\boldsymbol{A}'$ is the updated output combination matrix of the feature mapping layer and the enhancement node layer in incremental learning.

$$\boldsymbol{B}^{\mathrm{T}} = \begin{cases} (\boldsymbol{C})^{+}, & \text{if } \boldsymbol{C} \neq 0 \\ \left(1 + \boldsymbol{D}^{\mathrm{T}}\boldsymbol{D}\right)^{-1} \boldsymbol{D}^{\mathrm{T}}(\boldsymbol{A})^{+}, & \text{if } \boldsymbol{C} = 0 \end{cases} \tag{6}$$

and

$$\boldsymbol{C} = \begin{cases} \left[\boldsymbol{Z}_{n+1} | \xi\left(\boldsymbol{Z}_{n+1}\boldsymbol{W}_{h_1} + \beta_{h_1}\right), ..., \xi\left(\boldsymbol{Z}_{n+1}\boldsymbol{W}_{h_m} + \beta_{h_m}\right)\right] - \boldsymbol{A}\boldsymbol{D}, & \text{conditon 1} \\ \xi\left(\boldsymbol{Z}^{n}\boldsymbol{W}_{h_{m+1}} + \beta_{h_{m+1}}\right) - \boldsymbol{A}\boldsymbol{D}, & \text{conditon 2} \end{cases} \tag{7}$$

where condition 1 refers to adding the (n + 1)th feature node and condition 2 refers to adding the (m + 1)th enhancement node, resulting in:

$$\boldsymbol{D} = \begin{cases} (\boldsymbol{A})^{+} \xi\left(\boldsymbol{Z}^{n}\boldsymbol{W}_{h_{m+1}} + \beta_{h_{m+1}}\right) \\ (\boldsymbol{A})^{+}\left[\boldsymbol{Z}_{n+1} | \xi\left(\boldsymbol{Z}_{n+1}\boldsymbol{W}_{h_1} + \beta_{h_1}\right), ..., \xi\left(\boldsymbol{Z}_{n+1}\boldsymbol{W}_{h_m} + \beta_{h_m}\right)\right] \end{cases} \tag{8}$$

where $\boldsymbol{B}$, $\boldsymbol{C}$, and $\boldsymbol{D}$ are temporary variables and $\boldsymbol{Z}^n = [\boldsymbol{Z}_1, ..., \boldsymbol{Z}_n]$. $\boldsymbol{W}_{h_{m+1}}$ and $\beta_{h_{m+1}}$ are random parameters.

Thus, before the next FBL training round, the new weight $\boldsymbol{W}'_{\mathrm{g}}$ is updated as follows:

$$\boldsymbol{W}'_{\mathrm{g}} = \begin{bmatrix} \boldsymbol{W}_{\mathrm{g}} - \boldsymbol{D}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{\mathrm{Y}} \\ \boldsymbol{B}^{\mathrm{T}}\boldsymbol{\mathrm{Y}} \end{bmatrix} \tag{9}$$

where $\boldsymbol{W}'_{\mathrm{g}}$ is a matrix change with respect to $\boldsymbol{W}_{\mathrm{g}}$ while $\boldsymbol{W}_{\mathrm{g}}$ is the global weight. It is related to local weights of all previously received clients and also the clients in the next round aggregation. This incremental learning ensures that the knowledge originally learned by the model will not be forgotten [27].

3.3. RIS-aided parameter uploading and downloading

Client k also has the cost of computation and communication. With the help of RIS, the channel gain can be increased by reducing the obstacle interference [34]. The transmission rate of the uplink $R^{\mathrm{up}}_{k,s}$ between client k and the server is given by the following:

$$R^{\mathrm{up}}_{k,s}(t) = B^{\mathrm{up}}_{k,s}(t)\log_2\left(1 + \frac{p^{\mathrm{up}}_k(t)\left|\boldsymbol{w}^H_k(t)h_k(t)\right|^2}{\sum_{k=1}^{V} p^{\mathrm{up}}_k(t)\left|\boldsymbol{w}^H_k(t)h_k(t)\right|^2 + N_0(t)}\right) \tag{10}$$

where $B^{\mathrm{up}}_{k,s}$ is the uplink bandwidth. $\boldsymbol{w}_k(t)$ is the beamforming weights [18] (i.e. $\boldsymbol{w}_k(t)\boldsymbol{w}^H_k(t) = 1$), $N_0(t)$ is white Gaussian noise. $p^{\mathrm{up}}_k(t)$ is the uplink transmission power between client k and the server. $h_k(t)$ is the total channel gain between vehicle k and the server. The total channel gain between client k and the server is given by the following:

$$h_k(t) = h_{k,s}(t) + \boldsymbol{G}(t)\boldsymbol{\Theta}(t)h_{r,s}(t) \tag{11}$$

where $\boldsymbol{\Theta}(t)$ is the diagonal matrix. $\boldsymbol{\Theta}(t) = \mathrm{diag}\left(\alpha_1 e^{j\theta_1}(t), \alpha_2 e^{j\theta_2}(t), ..., \alpha_N e^{j\theta_N}(t)\right)$, where θ_N and α_N denote the phase shifts and amplitude reflection coefficient of element e (e = 1, 2, ..., ϑ) of RIS, respectively. ϑ is the RIS scale parameter. $\boldsymbol{G}(t)$ is the channel gain matrix between RIS and vehicle k. $h_{r,s}(t)$ is the channel gain between RIS and the server. $h_{k,s}(t)$ is the direct channel gain between vehicle k and the server.

The delay of client k's uploading transmission $L^{\mathrm{up}}_{k,s}$ and downloading transmission $L^{\mathrm{down}}_{k,s}$ can be expressed as follows:

$$L^{\mathrm{up}}_{k,s}(t) = \frac{\mathrm{size}(W_k(t))}{R^{\mathrm{up}}_{k,s}(t)} \tag{12}$$

$$L^{\mathrm{down}}_{k,s}(t) = \frac{\mathrm{size}(W_{\mathrm{g}}(t))}{R^{\mathrm{down}}_{k,s}(t)} \tag{13}$$

where W_k is parameters of the BL model to be uploaded of client k. W_{g} is global parameters of the BL model. size(W_k) refers to the size of W_k and size(W_{g}) refers to the size of W_{g}.

The transmission rate of the downlink $R^{\mathrm{down}}_{k,s}$ is expressed as follows:

$$R^{\mathrm{down}}_{k,s}(t) = B^{\mathrm{down}}_{k,s}(t)\log_2\left(1 + \frac{p^{\mathrm{down}}_k(t)\left|\boldsymbol{w}^H_k(t)h_k(t)\right|^2}{\sum_{k=1}^{V} p^{\mathrm{down}}_k(t)\left|\boldsymbol{w}^H_k(t)h_k(t)\right|^2 + N_0(t)}\right) \tag{14}$$

where $p^{\mathrm{down}}_k(t)$ is the downlink transmission power between client k and the server. $B^{\mathrm{down}}_{k,s}$ is the downlink bandwidth.

However, traditional FBL presents the problems of asynchronous training, resource allocation, and client incentive. Thus, we first improve the local training model and design a novel parameter aggregation approach (Section 4). Then, we formulate

the optimized problem for resource allocation to increase the bandwidth and adaptively adjust the RIS parameters. Finally, we give rewards to clients according to their FL contribution, based on the FIL algorithm (Section 5).

4. Local model training and asynchronous parameter aggregation

In this section, we introduce the process of our proposed local mode, the BFCM. We also introduce our process of federated parameter aggregation.

4.1. The broad fully connected model

At present, some BL models do not consider the difference of weight dimensions, which increases the number of nodes to simply improve the performance of the training process. In a cascade of feature mapping node BLS (CFBLS), random feature extraction is only related to the input data, and the dependency between features is not considered [35]. A CFBLS with pyramid (CFBLS-pyramid) pays too much attention to the correlation between the nodes at each level, ignoring the random characteristics of the input in *F*. With the increase of the number of sublayers, the number of nodes in each sublayer also increases dramatically, resulting in a decrease in the training efficiency in a CFBLS-pyramid. A CFBLS dropout can drop some nodes to prevent overfitting of the model but easily causes a loss-of-feature-information problem [35]. To solve these problems, we propose a fully connected neural-network-like architecture: the BFCM, which is based on the structure of fully connected neural networks in DL.

The main structure of the BFCM is shown in Fig. 2. Compared with a traditional BL model, each group of nodes in the BFCM is fully connected to the nodes in the previous group, just like the fully connected layer in DL. In this model, clients are allowed to customize the number of hidden neurons in each sublevel. That is, the number of sub-nodes in each node can differ, and the number of node groups in each sublevel can also differ. The process of extracting the features of *X* is represented as follows:

$$\boldsymbol{Z}_{i,j}=\begin{cases}\phi\left(\boldsymbol{X}\boldsymbol{W}_{e_{i,j}}+\beta_{e_{i,j}}\right); i=1,\ldots,n;\ \ j=1\\ \phi\left(\boldsymbol{X}\boldsymbol{W}_{e_{i,j}}+\beta_{e_{i,j}}\right); i=1,\ldots,n;\ \ j=2,\ldots,L_i^z\end{cases} \tag{15}$$

where L_i^z indicates that node i has L_i^z sub-nodes. $\boldsymbol{Z}_{i,j}$ is the mapping feature in *F* of node i which has j sub-nodes. In each level, the output of the first nodes group in each node is related to *X*. $\boldsymbol{W}_{e_{i,j}}$ and $\beta_{e_{i,j}}$ are the weights and biases of sub-node j in node i, respectively. $\boldsymbol{W}_{e_{i,j}}$ and $\beta_{e_{i,j}}$ can be different among nodes.

Then, the output of *F* is $\boldsymbol{Z}=\left[\boldsymbol{Z}_{1,1},\ldots,\boldsymbol{Z}_{1,L_1^z},\ldots,\boldsymbol{Z}_{m,1},\boldsymbol{Z}_{m,L_m^z}\right]$. We feed $\boldsymbol{Z}$ into *E* and generate all enhancement nodes $\boldsymbol{H}=\left[\boldsymbol{H}_{1,1},\ldots,\boldsymbol{H}_{1,L_1^h},\ldots,\boldsymbol{H}_{m,1},\ldots,\boldsymbol{H}_{m,L_m^h}\right]$. The group i, sublayer j node is obtained as follows:

$$\boldsymbol{H}_{i,j}=\begin{cases}\xi\left(\boldsymbol{Z}\boldsymbol{W}_{h_{i,j}}+\beta_{h_{i,j}}\right); i=1,\ldots,m;\ j=1\\ \xi\left(H_{i,j-1}W_{h_{i,j}}+\beta_{h_{i,j}}\right); i=1,\ldots,m;\ j=2,\ldots,L_i^h\end{cases} \tag{16}$$

where $\boldsymbol{W}_{h_{i,j}}$ and $\beta_{h_{i,j}}$ are the random parameters in *E*. L_i^h indicates that node i has L_i^h sub-nodes in *E*.

Thus, the relationship between the training input $\boldsymbol{X}$ and output $\boldsymbol{Y}$ is

$$\boldsymbol{Y}=[\boldsymbol{Z}|\boldsymbol{H}]\boldsymbol{W}=\boldsymbol{A}\boldsymbol{W} \tag{17}$$

where the weight $\boldsymbol{W}$ is obtained by $([\boldsymbol{Z}|\boldsymbol{H}])^{+}\boldsymbol{Y}$, and $([\boldsymbol{Z}|\boldsymbol{H}])^{+}$ is the pseudoinverse matrix of $[\boldsymbol{Z}|\boldsymbol{H}]$.

4.2. The asynchronous incremental learning approach

In BL, the model can adopt incremental learning to achieve higher performance. As shown in Fig. 3, based on the original parameters, the model can choose to add nodes in *F* or *E* without reinitializing the parameters, so as to further enhance the learning ability by retaining the knowledge learned before. Hence, clients can choose the number of nodes to increase according to their computing power. Suppose we add a new group to the original N groups or M groups, we will introduce how to add a new group $\boldsymbol{H}_{m+1}$ or $\boldsymbol{Z}_{n+1}$ in two parts, that is adding the $(m+1)$th or $(n+1)$th group.

4.2.1. Asynchronously incremental feature nodes in FL

If one client increases the $(n+1)$th group in the BFCM, the new output $\boldsymbol{Z}_{n+1}$ is given as follows:

$$\boldsymbol{Z}_{n+1,j}=\begin{cases}\phi\left(\boldsymbol{X}\boldsymbol{W}_{e_{i,j}}+\beta_{e_{i,j}}\right); i=n+1;\quad j=1\\ \phi\left(\boldsymbol{Z}_{i,j-1}\boldsymbol{W}_{e_{i,j}}+\beta_{e_{i,j}}\right); i=n+1; j=2,\ldots,L_i^z\end{cases} \tag{18}$$

where $\boldsymbol{W}_{e_{n+1,j}}$ and $\beta_{e_{n+1,j}}$ are the new parameters when $i=n+1$.

We assume that client k adds a total of $\hat{n}^k$ groups. Thus, we define the new feature node sequence of the global model as follows:

$$\boldsymbol{W}_e^{\mathrm{g}}=\mathrm{Concat}_{i=1}^{\max\left\{\hat{n}^k,\ldots,\hat{n}^V\right\}}\mathrm{Concat}_{j=1}^{L_i^z}\boldsymbol{\eta}_{j=1}^{L_i^z}\sum_{k=1,W_{e_{i,j}}^k\in\varnothing}^{V}\boldsymbol{W}_{e_{i,j}}^k \tag{19}$$

$$\beta_e^{\mathrm{g}}=\mathrm{Concat}_{i=1}^{\max\left\{\hat{n}^k,\ldots,\hat{n}^V\right\}}\mathrm{Concat}_{j=1}^{L_i^z}\eta_{j=1}^{L_i^z}\sum_{k=1,\beta_{e_{i,j}}^k\in\varnothing}^{V}\beta_{e_{i,j}}^k \tag{20}$$

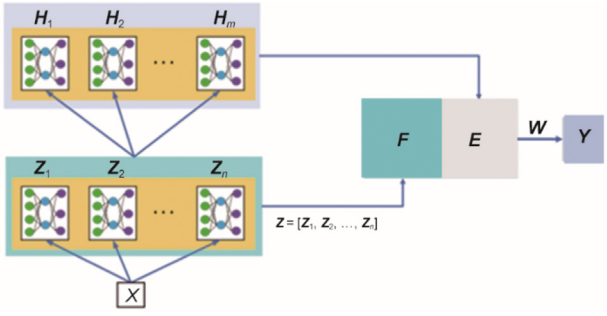

Fig. 2. The structure of the BFCM. ***W***: weight.

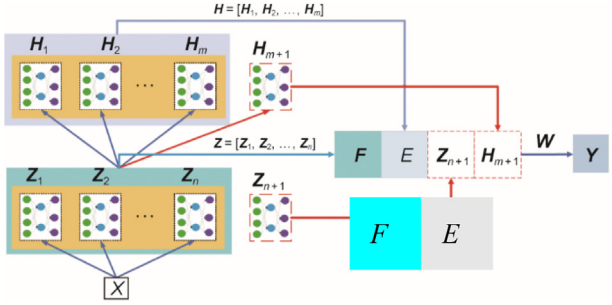

Fig. 3. The incremental learning of the BFCM.

where Concat indicates that the server adds the averaged parameters to sequence $\boldsymbol{W}_e^{\mathrm{g}}$ or β_e^{g}. $\boldsymbol{\eta}_{j=1}^{L_i^z}$ is an aggregation weight related to the proportion of the local training data. Because the number of nodes added by each client is inconsistent, we search the subscript (i, j) of each client's parameter sequence, and the retrieval length is the longest length $\max\{\hat{n}^k, \ldots, \hat{n}^V\}$ among the clients' parameter sequence. If the parameter of the subscript (i, j) is not empty, it can participate in aggregation.

After parameter aggregation, the server sends the parameter sequences β_e^{g} and $\boldsymbol{W}_e^{\mathrm{g}}$ to clients. Thus, the new mapping nodes $\boldsymbol{Z}'$ of the incremental learning are expressed as follows:

$$\boldsymbol{Z}' = \left[\boldsymbol{Z}'_{(n+1),N}, \ldots, \boldsymbol{Z}'_{(n+1),L_{n+1}^z}, \ldots, \boldsymbol{Z}'_{\max\left\{\hat{n}^k,\ldots,\hat{n}^V\right\},1}, \ldots, \boldsymbol{Z}'_{\max\left\{\hat{n}^k,\ldots,\hat{n}^V\right\},L^z_{\max\left\{\hat{n}^k,\ldots,\hat{n}^V\right\}}}\right] \tag{21}$$

Since $\beta_{e_{ij}}^{\mathrm{g}} \in \beta_e^{\mathrm{g}}$ and $\boldsymbol{W}_{e_{ij}}^{\mathrm{g}} \in \boldsymbol{W}_e^{\mathrm{g}}$, Eq. (18) can be rewritten as follows:

$$\boldsymbol{Z}'_{(n+1)j} = \begin{cases} \phi\left(\boldsymbol{X}\boldsymbol{W}_{e_{ij}}^{\mathrm{g}} + \beta_{e_{ij}}^{\mathrm{g}}\right); & i = n+1; \quad j = 1 \\ \phi\left(\boldsymbol{Z}_{ij-1}\boldsymbol{W}_{e_{ij}}^{\mathrm{g}} + \beta_{e_{ij}}^{\mathrm{g}}\right); & i = n+1; j = 2, \ldots, L_i^z \end{cases} \tag{22}$$

Thus, we define the new output of E for incremental nodes as $\boldsymbol{H}' = \left[\boldsymbol{H}'_{1,1}, \ldots, \boldsymbol{H}'_{1,L_1^h}, \ldots, \boldsymbol{H}'_{m,1}, \ldots, \boldsymbol{H}'_{m,L_m^h}\right]$. Each node is obtained by

$$\boldsymbol{H}'_{ij} = \begin{cases} \xi\left(\boldsymbol{Z}'_{(n+1)j}\boldsymbol{W}_{h_j} + \beta_{h_{ij}}\right); & i = 1, \ldots, m; j = 1 \\ \xi\left(\boldsymbol{H}'_{ij-1}\boldsymbol{W}_{h_{ij}} + \beta_{h_{ij}}\right); & i = 1, \ldots, m; j = 2, \ldots, L_i^h \end{cases} \tag{23}$$

The updated output combination matrix of the feature mapping layer and the enhancement node layer in incremental learning $\boldsymbol{A}$' and its pseudoinverse matrix $(\boldsymbol{A}')^+$ are updated refers to Eqs. (5)–(9).

4.2.2. Asynchronously incremental enhancement nodes in FL

If one client increases the $(m+1)$th group in E of the BFCM, the output H_{m+1} is given as follows:

$$\boldsymbol{H}_{(m+1)j} = \begin{cases} \xi\left(\boldsymbol{Z}\boldsymbol{W}_{h_{ij}} + \beta_{h_{ij}}\right); & i = m+1, j = 1 \\ \xi\left(\boldsymbol{H}_{ij-1}\boldsymbol{W}_{h_{ij}} + \beta_{h_{ij}}\right); & i = m+1, j = 2, \ldots, L_i^h \end{cases} \tag{24}$$

We assume that client k increases the total $\hat{m}^k$ groups. Thus, we define the new enhancement node sequence of the global model as follows:

$$\boldsymbol{W}_h^{\mathrm{g}} = \mathrm{Concat}_{i=1}^{\max\left\{\hat{m}^k,\ldots,\hat{m}^V\right\}} \mathrm{Concat}_{j=1}^{L_i^h} \eta_{j=1}^{L_i^h} \sum_{k=1, W_{h_{ij}}^k \in \varnothing}^{V} \boldsymbol{W}_{h_{ij}}^k \tag{25}$$

$$\beta_h^{\mathrm{g}} = \mathrm{Concat}_{i=1}^{\max\left\{\hat{m}^k,\ldots,\hat{m}^V\right\}} \mathrm{Concat}_{j=1}^{L_i^h} \eta_{j=1}^{L_i^h} \sum_{k=1, \beta_{h_{ij}}^k \in \varnothing}^{V} \beta_{h_{ij}}^k \tag{26}$$

where Concat indicates that the server adds the averaged parameters to sequence $\boldsymbol{W}_h^{\mathrm{g}}$ or β_h^{g}. $\eta_{j=1}^{L_i^h}$ is a aggregation weight related to the proportion of the local training data. Because the number of nodes added by each client is inconsistent, we search the subscript (i, j) of each client's parameter sequence, and the retrieval length is the longest length $\max\{\hat{m}^k, \ldots, \hat{m}^V\}$ among the clients' parameter sequence. If the parameter of the subscript (i, j) is not empty, it can participate in aggregation.

Thus, we define the new output of E for incremental nodes as follows: $\boldsymbol{H}' = \left[\boldsymbol{H}'_{1,1}, \ldots, \boldsymbol{H}'_{1,L_1^h}, \ldots, \boldsymbol{H}'_{m,1}, \ldots, \boldsymbol{H}'_{m,L_m^h}\right]$. Each node is obtained by

$$\boldsymbol{H}'_{(m+1)j} = \begin{cases} \xi\left(\boldsymbol{Z}\boldsymbol{W}_{h_{ij}}^{\mathrm{g}} + \beta_{h_{ij}}^{\mathrm{g}}\right); & i = m+1; j = 1 \\ \xi\left(\boldsymbol{H}_{ij-1}\boldsymbol{W}_{h_{ij}}^{\mathrm{g}} + \beta_{h_{ij}}^{\mathrm{g}}\right); & i = m+1; j = 2, \ldots, L_i^h \end{cases} \tag{27}$$

The updated output combination matrix $\boldsymbol{A}'$ and its pseudoinverse matrix $(\boldsymbol{A}')^+$ are updated refers to Eqs. (5)–(9).

4.2.3. Asynchronous aggregation in FBL

Considering that the client may increase the enhancement nodes and feature nodes simultaneously, the client k can send $\left\{\boldsymbol{W}_{h_{ij}}^k, \boldsymbol{W}_{e_{ij}}^k, \beta_{h_{ij}}^k, \beta_{e_{ij}}^k\right\}$, and $\boldsymbol{W}^k$ to participate in FBL. Finally, the server can packet and send the aggregated parameters $\boldsymbol{W}_{\mathrm{g}} = \{\boldsymbol{W}_e^{\mathrm{g}}, \beta_e^{\mathrm{g}}, \boldsymbol{W}_h^{\mathrm{g}}, \beta_h^{\mathrm{g}}, \boldsymbol{W}\}$ to each client. The details are shown in Algorithm 1.

Algorithm 1. The asynchronous FBL algorithm.

Function ClientProcess (k, $\boldsymbol{W}_{\mathrm{g}}$):
  Update the local model parameters by $\boldsymbol{W}_{\mathrm{g}}$
  for each local incremental feature node group i = 1, 2, ..., $\hat{n}^k$ **do**
    Train the local BFCM model by incremental learning according to Eqs. (18)–(22)
  end
  for each local incremental enhancement node group i = 1, 2, ..., $\hat{m}^k$ **do**
    Train the local BFCM model by incremental learning according to Eqs. (24)–(30)
  end
  Return $\left\{\boldsymbol{W}_{h_{ij}}^k, \boldsymbol{W}_{e_{ij}}^k, \beta_{h_{ij}}^k, \beta_{e_{ij}}^k\right\}$ to the server
End Function
Function server process (t):
  Initialize $\boldsymbol{W}_{\mathrm{g}}$
  if t is 1 **then**
    for k = 1, 2, ... **do**
      $\left\{\boldsymbol{W}_{h_{ij}}^k, \boldsymbol{W}_{e_{ij}}^k, \beta_{h_{ij}}^k, \beta_{e_{ij}}^k, \boldsymbol{W}^k\right\} \leftarrow$ **ClientProcess** (k, $\boldsymbol{W}_{\mathrm{g}}$):
    end
  end
  else
    for k = 1, 2, ... **do**
      $\left\{\boldsymbol{W}_{h_{ij}}^k, \boldsymbol{W}_{e_{ij}}^k, \beta_{h_{ij}}^k, \beta_{e_{ij}}^k, \boldsymbol{W}^k\right\} \leftarrow$ **ClientProcess**(k, $\boldsymbol{W}_{\boldsymbol{k}}$):
    end
  end
Compute $\boldsymbol{W}_e^{\mathrm{g}}$ according to Eq. (19)
Compute β_e^{g} according to Eq. (20)
Compute $\boldsymbol{W}_h^{\mathrm{g}}$ according to Eq. (25)
Compute β_h^{g} according to Eq. (26)
End Function

5. Resource and revenue allocation for FBL

In this section, we use the Davidon–Fletcher–Powell (DFP) [36] algorithm to solve the resource allocation problem and RIS configuration problem involved in FIL. Furthermore, we introduce the process of our proposed revenue allocation algorithm, which is based on an FIL algorithm. In FL, the RIS can enhance the channel quality and reduce the communication cost during FIL [14,37].

5.1. Resource allocation for FBL

According to Eq. (28), the purpose of clients is to maximize the individual reward, and clients may not care much about the

network experience. However, for operators, the performance of the network has an important impact on the effect of FBL, so it is necessary to attach great importance to overall network performance. Thus, improving the transmission rate of clients can ensure the maximum quality of service (QoS). The optimized function can be rewritten as two subproblems P1 and P2.

$$\begin{aligned}&\mathrm{P1}: \max \sum R_{k,s}^{\mathrm{up}}(t)\\&\text{s.t.}\\&\mathrm{C1}: \theta_e(t) \in [0, 2\pi], \forall e \in \vartheta\\&\mathrm{C2}: \sum_{k=1}^{V} B_{k,s}^{\mathrm{up}}(t) \le B_{\max}^{\mathrm{up}}(t)\\&\mathrm{C3}: \alpha_e(t) \in [0, 1], \forall e \in \vartheta\end{aligned} \tag{28}$$

where, according to subproblem P1, we should maximize the uplink rate. C1 ensures the phase shift angular range in RIS, while C2 restricts the total transmission rate so that it cannot be greater than $B_{\max}^{\mathrm{up}}(t)$ at time slot t. C3 limits the amplitude reflection coefficient value. $B_{k,s}(t)$, $\theta_e(t)$, and $\alpha_e(t)$ determine the overall transmission rate.

Similarly, according to subproblem P2, we should maximize the downlink rate. In P2, constraints ensure the resource limitation:

$$\begin{aligned}&\mathrm{P2}: \max \sum R_{k,s}^{\mathrm{down}}(t)\\&\text{s.t.}\\&\mathrm{C1}: \theta_e(t) \in [0, 2\pi], \forall e \in \vartheta\\&\mathrm{C2}: \sum_{k=1}^{V} B_{k,s}^{\mathrm{down}}(t) \le B_{\mathrm{down}}^{\max}(t)\\&\mathrm{C3}: \alpha_e(t) \in [0, 1], \forall e \in \vartheta\end{aligned} \tag{29}$$

As shown by the constraints of P1 and P2, P1 and P2 are two non-convex problems. First, we remove the constraint and transform the original P1 problem into a generalized Lagrange function, as follows:

$$\begin{aligned}\min R^{\mathrm{up}}(t) = \min &- \sum_{k=1}^{V} R_{k,s}^{\mathrm{up}}(t) + \sum_{e=1}^{\vartheta} \lambda_e^{\mathrm{a,up}}(\theta_e(t) - 2\pi)\\&+ \sum_{e=1}^{\vartheta} \lambda_e^{\mathrm{b,up}}(-\theta_e(t)) + \upsilon^{\mathrm{a,up}} \sum_{k=1}^{V} B_{k,s}^{\mathrm{up}}(t) - B_{\max}^{\mathrm{up}}(t)\\&+ \upsilon^{\mathrm{b,up}} \sum_{k=1}^{V} -B_{k,s}^{\mathrm{up}}(t) + \sum_{e=1}^{\vartheta} \mu_e^{\mathrm{a,up}}(\alpha_e(t) - 2\pi)\\&+ \sum_{e=1}^{\vartheta} \mu_e^{\mathrm{b,up}}(-\alpha_e(t))\end{aligned} \tag{30}$$

where our goal is to minimize $R^{\mathrm{up}}(t)$. $\lambda_e^{\mathrm{a,up}}$, $\lambda_e^{\mathrm{b,up}}$, $\upsilon^{\mathrm{a,up}}$, $\upsilon^{\mathrm{b,up}}$, $\mu_e^{\mathrm{a,up}}$, and $\mu_e^{\mathrm{b,up}}$ are penalty factors.

Similarly, the downlink rate should be maximized as follows:

$$\begin{aligned}\min R^{\mathrm{down}}(t) = \min &- \sum_{k=1}^{V} R_{k,s}^{\mathrm{down}}(t) + \sum_{e=1}^{\vartheta} \lambda_e^{\mathrm{a,down}}(\theta_e(t) - 2\pi)\\&+ \sum_{e=1}^{\vartheta} \lambda_e^{\mathrm{b,down}}(-\theta_e(t)) + \upsilon^{\mathrm{a,down}} \sum_{k=1}^{V} B_{k,s}^{\mathrm{down}}(t)\\&- B_{\max}^{\mathrm{down}}(t) + \upsilon^{\mathrm{b,down}} \sum_{k=1}^{V} -B_{k,s}^{\mathrm{down}}(t)\\&+ \sum_{e=1}^{\vartheta} \mu_i^{\mathrm{a,down}}(\alpha_e(t) - 2\pi) + \sum_{e=1}^{\vartheta} \mu_e^{\mathrm{b,down}}(-\alpha_e(t))\end{aligned} \tag{31}$$

where $\lambda_e^{\mathrm{a,down}}$, $\lambda_e^{\mathrm{b,down}}$, $\upsilon^{\mathrm{a,down}}$, $\upsilon^{\mathrm{b,down}}$, $\mu_e^{\mathrm{a,down}}$, and $\mu_e^{\mathrm{b,down}}$ are also penalty factors.

To sum up, the optimization goal of the total time slot is

$$\min R = \min \sum_{t=1}^{T} \left(R^{\mathrm{down}}(t) + R^{\mathrm{up}}(t)\right) \tag{32}$$

where T means the total time slot.

Thus, we define the penalty factors list Pen, RIS controlling parameters list I, and networking resource parameters list BW as follows:

$$\begin{aligned}\mathrm{Pen} = \Big\{&\lambda^{\mathrm{b,up}}, \lambda^{\mathrm{a,up}}, \upsilon^{\mathrm{a,up}}, \upsilon^{\mathrm{b,up}}, \mu^{\mathrm{b,up}}, \mu^{\mathrm{a,up}},\\&\lambda^{\mathrm{b,down}}, \lambda^{\mathrm{a,down}}, \upsilon^{\mathrm{a,down}}, \upsilon^{\mathrm{b,down}}, \mu^{\mathrm{b,down}}, \mu^{\mathrm{a,down}}\Big\}\end{aligned}$$

$$I = \{\theta, \alpha\} \tag{33}$$

$$BW = \left\{B^{\mathrm{up}}, B^{\mathrm{down}}\right\}$$

Therefore, our goal is to weigh the performance of FBL against the revenue of clients by determining the values of Pen, I, and B. From this analysis, we decouple the problem into an uplink/downlink problem and an RIS configuration problem. We propose a DDFP algorithm based on the DFP method [36] to solve these two problems. First, we fix the reflector coefficients, then we use a DFP algorithm to solve the optimal transmission bandwidth for parameter upload and parameter download, respectively. Finally, according to the resource optimization, we use a DFP algorithm to solve the optimal RIS coefficient configuration again. The details are shown in Algorithm 2. We assume the maximum iteration to be $r_{\max}$ and the maximum incremental node group to be $\max(\hat{n}_{\max}, \hat{m}_{\max})$, where $\hat{n}_{\max}$ and $\hat{m}_{\max}$ represent the maximum number of feature mapping nodes and feature enhancement nodes added in all clients, respectively. The total time complexity is $O(\max(TV\max(\hat{n}_{\max}, \hat{m}_{\max}), Tr_{\max}))$. In the IoV, the value of V will be large. Thus, $TV\max(\hat{n}_{\max}, \hat{m}_{\max}) \gg Tr_{\max}$, and the total time complexity is approximately equal to $O(TV\max(\hat{n}_{\max}, \hat{m}_{\max}))$. Compared with the DL method for solving the resource allocation problem, the DFP algorithm does not require large-scale trainable parameters, which optimizes to a certain extent a large amount of storage space needed in the implementation of the algorithm. By decoupling complex problems, the difficulty of problem processing is somewhat reduced, and the efficiency of the algorithm is improved.

Algorithm 2. The DDFP algorithm.

Fix I
Initialize j = 0
Randomize Pen, BW
Calculate the Hessian matrix $\boldsymbol{D}(j)$ of $(R^{\mathrm{up}}(t))(\mathrm{Pen}^{\mathrm{up}}, B^{\mathrm{up}})$
While $\|g(j+1)\| <$ **do**; // ◁ optimize the uplink transmission rate, $g(j+1)$ means the gradient for iteration $j+1$
  $\boldsymbol{g}(j) \leftarrow \nabla(R^{\mathrm{up}}(t))(\mathrm{Pen}, BW)$
  $\boldsymbol{x}(j) \leftarrow (R^{\mathrm{up}}(t))(\mathrm{Pen}, BW)$
  Get $\boldsymbol{x}(j+1)$ according to the line search method [1]
  Calculate $g(j+1)$
    $\Delta g = g(j+1) - g(j)$, $\Delta \boldsymbol{x} = \boldsymbol{x}(j+1) - \boldsymbol{x}(j)$
    $\boldsymbol{D}(j+1) = \boldsymbol{D}(j) + \frac{\Delta \boldsymbol{x} \Delta \boldsymbol{x}^{\mathrm{T}}}{\Delta g^{\mathrm{T}} \Delta \boldsymbol{x}} - \frac{\boldsymbol{D}(j) \Delta g \Delta g^{\mathrm{T}} \boldsymbol{D}(j)}{\Delta g^{\mathrm{T}} \boldsymbol{D}(j) \Delta g}$
  End
Initialize j = 0
Randomize Pen, BW
Calculate the Hessian matrix $\boldsymbol{D}(j)$ of
  $\left(R^{\mathrm{down}}(t)\right)\left(\mathrm{Pen}^{\mathrm{down}}, B^{\mathrm{down}}\right)$
While $\|g(j+1)\| <$ **do**; // ◁ optimize the downlink transmission rate
  $g(j) \leftarrow \nabla\left(R^{\mathrm{down}}(t)\right)(\mathrm{Pen}, BW)$
  $\boldsymbol{x}(j) \leftarrow \left(R^{\mathrm{down}}(t)\right)(\mathrm{Pen}, BW)$

Get $\boldsymbol{x}(j+1)$ according to the line search method [1]
Calculate $g(j+1)$
$\Delta g = g(j+1) - g(j),\ \Delta \boldsymbol{x} = \boldsymbol{x}(j+1) - \boldsymbol{x}(j)$
$\boldsymbol{D}(j+1) = \boldsymbol{D}(j) + \frac{\Delta \boldsymbol{x}\Delta \boldsymbol{x}^{\mathrm{T}}}{\Delta g^{\mathrm{T}}\Delta \boldsymbol{x}} - \frac{\boldsymbol{D}(j)\Delta g\Delta g^{\mathrm{T}}\boldsymbol{D}(j)}{\Delta g^{\mathrm{T}}\boldsymbol{D}(j)\Delta g}$
End
If $j\%r == 0$ **then**; // ◁ optimize the RIS
Initialize $j = 0$
Calculate the Hessian matrix $\boldsymbol{D}(j)$ of
$\left(R^{\mathrm{down}}(t)\right)(I) + \left(R^{\mathrm{up}}(t)\right)(I)$
While $\|\boldsymbol{g}(j+1)\| <$ **do**; // ◁ optimize the uplink transmission rate
$g(j) \leftarrow \nabla\left(\left(R^{\mathrm{down}}(t)\right)(I) + \left(R^{\mathrm{up}}(t)\right)(I)\right)$
$\boldsymbol{x}(j) \leftarrow \left(\left(R^{\mathrm{down}}(t)\right)(I) + \left(R^{\mathrm{up}}(t)\right)(I)\right)$
Get $\boldsymbol{x}(j+1)$ according to the line search method [1]
Calculate $g(j+1)$
$\Delta g = g(j+1) - g(j),\ \Delta \boldsymbol{x} = \boldsymbol{x}(j+1) - \boldsymbol{x}(j)$
$\boldsymbol{D}(j+1) = \boldsymbol{D}(j) + \frac{\Delta \boldsymbol{x}\Delta \boldsymbol{x}^{\mathrm{T}}}{\Delta g^{\mathrm{T}}\Delta \boldsymbol{x}} - \frac{\boldsymbol{D}(j)\Delta g\Delta g^{\mathrm{T}}\boldsymbol{D}(j)}{\Delta g^{\mathrm{T}}\boldsymbol{D}(j)\Delta g}$
End

5.2. RFL based on FIL for revenue allocation

Due to the unbalanced FL training and high mobility in the IoV, it is difficult to evaluate the contribution of FL among clients. On the one hand, service providers hope to acquire high-quality knowledge and do not want dirty and useless data from clients. On the other hand, clients consume local computation resources for model training and communication resources for transmitting parameters. Therefore, they must make profits to make up for the expenses caused by knowledge sharing.

According to the RIS-aided channels given in Eqs. (10)–(14), the delay $L_k(t)$ and energy consumption $E_k(t)$ of communication and computation in slot t, respectively, can be determined as follows:

$$L_k(t) = L_{k,s}^{\mathrm{up}}(t) + L_{k,s}^{\mathrm{down}}(t) + \frac{\mathrm{size}(W_k(t))\theta^c}{f_k(t)} + \frac{\mathrm{size}(W_g(t))\theta^c}{f_s(t)} + L_{\mathrm{wait}} \tag{34}$$

$$E_k(t) = a_1 L_{k,s}^{\mathrm{up}}(t) + a_2 L_{k,s}^{\mathrm{down}}(t) + a_3\frac{\mathrm{size}(W_k(t))\theta^c}{f_k(t)} + a_4 \times \frac{\mathrm{size}(W_g(t))\theta^c}{f_s(t)} + a_5 L_{\mathrm{wait}} \tag{35}$$

where a_1–a_5 are the energy factors for energy consumption. We set a_1 and a_2 as 0.05, a_3 and a_4 as 10^{-21}, and a_5 as 0.5. The RSU requires a certain waiting delay to collect all the parameters. Thus, the waiting delay is given as L_{wait}, which represents the time required to collect all the parameters. In an asynchronous learning process, the arrival time of the model parameters to the server is inconsistent. Although the L_{wait} is much less than the delay required for model calculation and communication, we still set it to a constant in the simulation. $f_k(t)$ is the computation frequency of the user k, and $f_s(t)$ is the RSU computation frequency. θ^c means the calculation density.

The total cost c_k of client k is defined as follows:

$$c_k(t) = \xi_{\mathrm{l}} L_k(t) + \xi_{\mathrm{e}} E_k(t) \tag{36}$$

where Eq. (36) indicates that the cost mainly comes from energy consumption and delay. ξ_{l} and ξ_{e} represent the price factors. To sum up, the greater the training time, the greater the cost caused by the client k. This indicates that the costs incurred by clients must be compensated by the server in order to better motivate clients to participate in FL and contribute their knowledge as much as possible.

To encourage clients to participate in FL, the server can give benefits to clients to download high-quality knowledge in the IoV. For each client k at time slot t, the benefit to the client is $u_k(t)$, and the client's contribution to the FL is $q_k(t)$. Driverless processes are mainly based on image classification or real-time road condition analysis for vehicular performance evaluation. Thus, improving the prediction accuracy ACC_k of tasks is key in the IoV. The contribution $q_k(t)$ is related to the prediction accuracy, so we define $q_k(t)$ as follows:

$$q_k(t) = \mathrm{ACC}_k(y_k, \Omega_k(x_k; W_k)) \tag{37}$$

where (x_k, y_k) is the training data of client k under the model parameters W_k. $\Omega_k(\cdot)$ means the local BFCM model.

According to Eqs. (36) and (37), we define the expected marginal revenue function u_k^e of client k as follows:

$$u_k^e(t) = \arctan(q_k(t) - \mathrm{sigmoid}(c_k(t))) + \beta^{\mathrm{u}} \tag{38}$$

where the sigmoid function is primarily adopted to normalize costs to [0, 1]. Since $q_k(t)$ is also in the range of [0, 1], $q_k(t) - \mathrm{sigmoid}(c_k(t))$ is a negative value. The advantage of the arctan function is that the input and output are positively correlated. Thus, we use arctan to correct the value of $q_k(t) - \mathrm{sigmoid}(c_k(t))$ to ensure that the output value is greater than 0, so that the output value is larger, the revenue is greater. β^{u} is the offset we give.

For each client, the FL manager can monitor the client's reward for a long time. We define $Y_k(t)$ as the difference between what the client k has received so far and what the client should have received at time slot t:

$$Y_k(t+1) \triangleq \max\left[Y_k(t) + c_k(t) - \widehat{u}_k(t), 0\right] \tag{39}$$

where the larger $Y_k(t)$ is, the more compensation client k should receive.

Because the training times of clients are asynchronous, clients should receive different benefits. However, because of the limited funds of the server, some clients may have to wait a long time to receive their full benefits. Thus, we define the waiting time cost as $Q_k(t)$. The dynamic update of $Q_k(t)$ can be expressed as follows:

$$Q_k(t+1) \triangleq \max\left[Q_k(t) + \lambda_k(t) - \widehat{u}_k(t), 0\right] \tag{40}$$

where $\lambda_k(t)$ is related to the historical payoff:

$$\lambda_k(t) = \begin{cases} \widehat{c}_k, \text{if } Y_k(t) = 0 \\ 0, \text{otherwise} \end{cases} \tag{41}$$

We calculate the historical cost based on the exponentially weighted averages method as follows:

$$\widehat{c}_k = \sum_{i=1}^{t} \left(\beta_i^{\mathrm{c}}\right)^{i-1}(1-\beta^{\mathrm{c}})^i c_k(i) \tag{42}$$

where β_i^{c} is a hyper-parameter. The closer it is to the current time, the greater the weight that is given to the cost. Also, Eq. (42) can fit the changing trend of client cost.

Thus, the actual revenue received by client k is

$$\widehat{u}_k(t) = \frac{u_k(t)}{\sum_{k=1}^{V} u_k(t)}\mathrm{Inc}(t) \tag{43}$$

where $\mathrm{Inc}(t)$ represents the total budget of the FL manager at time slot t. This equation shows that the actual revenue $\widehat{u}_k(t)$ of client t is related to the revenue of $\mathrm{Inc}(t)$ and other clients; $u_k(t)$ is defined as follows:

$$u_k(t) = \frac{1}{2}\left[\omega_u u_k^e(t) + Y_k(t) + c_k(t) + Q_k(t) + \lambda_k(t)\right] \tag{44}$$

where ω_u is a weight factor, and Eq. (44) shows the expected payoff of client k at time slot t. However, $\widehat{u}_k(t)$ is always less than or equal to $u_k(t)$, and we expect $\widehat{u}_k(t)$ to get close to $u_k(t)$. We define the benefit rate as follows:

$$U_k^{\text{rate}}(t) = \frac{\widehat{u}_k(t)}{u_k(t)} \tag{45}$$

where $U_k^{\text{rate}}(t)$ is adopted to evaluate the benefit of client k.

Algorithm 3. FIL for revenue allocation.

For each slot time t = 1, 2, ..., T **do**
  Call Algorithm 1
  Call Algorithm 2
  For k = 1, 2, ..., V **do**
    Compute $c_k(t)$
    Compute $q_k(t)$
    Compute $u_k(t)$
  End
  For k = 1, 2, ..., V **do**
    $\widehat{u}_k(t) \leftarrow \frac{u_k(t)}{\sum_{k=1}^{V} u_k(t)}\text{Inc}(t)$
    Compute $Y(t+1)$, $Q(t+1)$
End
  Return $\{\widehat{u}_{1,\ldots,}\widehat{u}_k\}$
End

6. Simulation results

In this section, we use datasets to verify and analyze the performance of our proposed algorithms. Then, we analyze the simulation by using different datasets, parameter settings, and comparison models.

6.1. Parameter settings

As shown in Fig. 4, we use Python 3.7 to simulate the vehicle operation and channel status in the Qinhuangdao center area, using an 11th Gen Intel Core i7-11800H@2.30GHz 8 core. The main reference values of the important parameters are provided in Table 1.

Fig. 4. The vehicular scenario of Qinhuangdao.

6.2. Dataset

We use the following datasets:

(1) **Mixed National Institute of Standards and Technology (MNIST).** The MNIST dataset is a handwritten numeral recognition dataset with ten categories (0–9). We set 80% of the dataset as training data and 20% as testing data.

(2) **Car evaluation.** The car evaluation dataset directly relates to the six input attributes of buying, maintenance (maint), doors, persons, luggage_boot (lug_boot), and safety to evaluate the car performance. The car performance has four categories, and the data type is String, so we adopt the HashCode method to encode the original data. The details in this dataset are shown in Table 2.

6.3. Performance comparison

We adopt several models to evaluate the prediction performance of the proposed algorithms, as follows:

(1) **Support vector machine (SVM).** An SVM is a supervised learning algorithm that can be classified nonlinearly via the kernel method. In an SVM, we use a scikit-learn package to search for the best optimized parameters: the kernel function and the penalty.

(2) **Fully connected network (FCN).** An FCN is a network composed of multiple fully connected layers. In an FCN, we set up three layers of fully connected for MNIST, where the number of features output from each layer is 120, 84, and 10, respectively. For car evaluation, we set up two layers of fully connected, and the number of features output from each layer is 10 and 4, respectively.

(3) **BLS.** A traditional BLS includes several feature nodes and enhancement nodes. The details of the parameter settings are provided in Tables 3 and 4.

(4) **CFBLS.** Compared with the BLS, the CFBLS has a cascade structure in feature nodes. The details of the parameter settings are provided in Tables 3 and 4.

(5) **CFBLS-pyramid.** Compared with the CFBLS, the CFBLS-pyramid has a pyramid structure in the feature nodes. The details of the parameter settings are provided in Tables 3 and 4.

(6) **BFCM.** If we set the number of feature mapping layers and feature enhancement layers to 1, BFCM becomes CFBLS. The details of the parameter settings are provided in Tables 3 and 4.

(7) **Federated averaging (FedAVG).** This model adopts the weighted average method to aggregate parameters. In general, the local model adopts a DL model. In this paper, we use long

Table 1
Parameter list.

Parameter	Description	Reference value
$\theta_{\max}$	Calculation density	800–1000 cycle·bit^{-1}
$f_{\max}$	Maximum frequency	0.6×10^9–1.0×10^9 Hz
N_0	Noise power	10^{-13} W
$P_{\max}$	Maximum transmit power	0.6×10^3–1.0×10^3 W
$B_{\max}$	Maximum transmit bandwidth	0.12×10^6–0.18×10^6 Hz
Inc	Total amount of money in the server	2000–3000

Table 2
Description of the car evaluation dataset.

Attribute	Value
Buying	[vhigh (very high), high, med, low]
Maint	[vhigh, high, med (median), low]
Doors	[2, 3, 4, 5, more]
Persons	[2, 4, more]
Lug_boot	[small, med, big]
Safety	[low, med, high]

Table 3
Classification accuracy using the MNIST data set for different parameter settings.

Model	Number of feature nodes or groups	Number of enhancement nodes or groups	Testing accuracy	Training time (s)
SVM	—	—	85.32	—
FCN	—	—	89.21	—
BLS	3000	3000	90.02	36.45
CFBLS	3000	3000	90.33	55.97
CFBLS-pyramid	5500	5000	90.81	85.38
BFCM	1220	1220	91.31	16.69

Table 4
Classification accuracy using the car evaluation dataset for different parameter settings.

Model	Number of feature nodes or groups	Number of enhancement nodes or groups	Testing accuracy	Training time (s)
BLS	500	500	91.08	0.45
CFBLS	500	500	89.53	0.45
CFBLS-pyramid	275	50	91.33	0.03
BFCM	250	250	92.17	0.22

short-term memory (LSTM), a convolutional neural network (CNN), and FCN as local training models to evaluate the performance of FedAVG.

6.4. Performance analysis

Table 3 shows the classification accuracy of the MNIST dataset with different parameter settings. We start with five feature nodes or groups and five enhancement nodes or groups. After incremental learning, the last feature nodes of BLS, CFBLS, CFBLS-pyramid, and BFCM are 3000, 3000, 5500, and 1220, respectively. More specifically, we define each group of nodes in BFCM as 80, 32, and 25, respectively. The last enhancement nodes of BLS, CFBLS, CFBLS-pyramid, and BFCM are 3000, 3000, 5000, and 1220, respectively. Compared with the other models, our proposed model achieves the best average classification accuracy with the least total number of nodes in centralized learning.

Table 4 shows the classification accuracy with the car evaluation dataset under different parameter settings. We start with ten feature nodes or groups and five enhancement nodes or groups. More specifically, we define each group of nodes in BFCM as 10, 5, and 10, respectively. After incremental learning, the last feature nodes of BLS, CFBLS, CFBLS-pyramid, and BFCM are 500, 500, 275, and 250, respectively. The last enhancement nodes of BLS, CFBLS, CFBLS-pyramid, and BFCM are 500, 500, 50, and 250, respectively. The large number of features generated in the middle does not necessarily improve the classification accuracy; on the contrary, it may produce an overfitting phenomenon. However, our model can flexibly set the number of feature nodes or enhancement nodes in each group according to the situation, which improves the generalization of data learning.

The performance of centralized learning is shown in Tables 3 and 4. Centralized learning allows all client data to be uploaded to the server. Mastering all the data in a centralized model is beneficial to the training of model parameters. With an increase in the number of nodes, the mapping features increase, and the classification errors caused by the randomness of the parameters increase.

Figs. 5 and 6 show the performance of FL with different models. Compared with traditional machine learning or DL, BL has advantages in efficiency and prediction accuracy. Our proposed FBL adopts an asynchronous parameter aggregation method, which can effectively solve the problem that the parameters cannot be aggregated directly due to the inconsistency of the nodes between models in an asynchronous BL process. However, it is necessary to train and aggregate parameters efficiently due to the fast movement of vehicles and the limited-service range of RSU. Traditional DL algorithms such as CNN, LSTM, or FCN are used as local models. Due to the long training time or long distance from the service range, RSU does not receive the client parameters, making FedAVG not particularly effective. Because of the lightweight characteristic

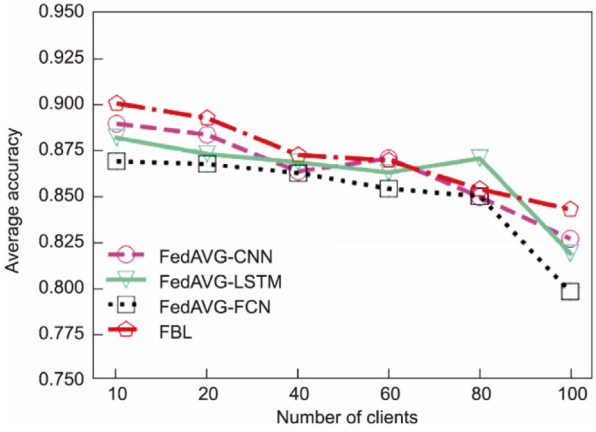

Fig. 5. Average accuracy with the MNIST dataset for different numbers of clients.

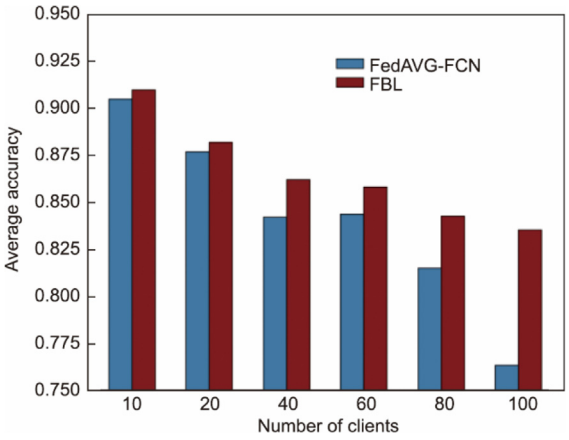

Fig. 6. Average accuracy with the car evaluation dataset for different numbers of clients.

of the local model, FBL can meet the requirement of frequent parameter exchange in the IoV and can improve the learning ability of the global model.

Figs. 7 and 8 show that our proposed FBL can reduce the cost more than the comparison algorithms. However, the increase in training time and clients will lead to a gradual increase in the model training cost. Further analysis shows that the cost of FL usually stems from frequent parameter exchange, so it is necessary to allocate the communication resources reasonably. Increasing clients' rewards is conducive to increasing clients' willingness to participate in FL activities. Figs. 9 and 10 show the performance under distributed learning from different numbers of clients. With the help of RIS, our FBL can allocate resources dynamically according to the environment to improve the utilization of resources and encourage clients to contribute to FL. Our FBL can maintain the yield above 90%, which shows that an efficient local model promotes clients to participate in FL more frequently. More importantly, the reasonable resource allocation and RIS configuration algorithm can enhance the channel quality and improve the efficiency of IoV knowledge sharing.

Fig. 11 shows the influence of a different number of reflective elements on the FBL algorithm. It is clear that an increase in the reflective elements does indeed improve the overall effect of FBL, which is manifested in an increase in the rate of return. More importantly, with the increase in FL clients, the utilization rate of resources increases, making the rate of return close to 100%.

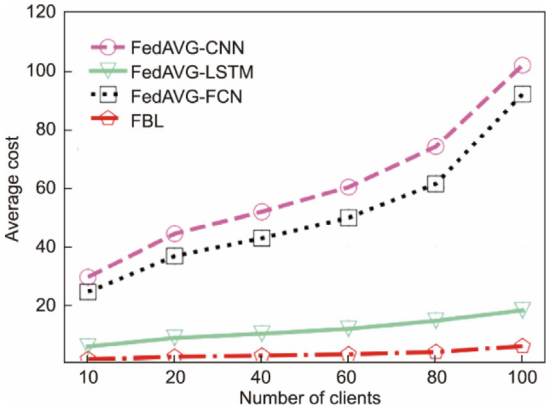

Fig. 7. Average cost for different numbers of clients.

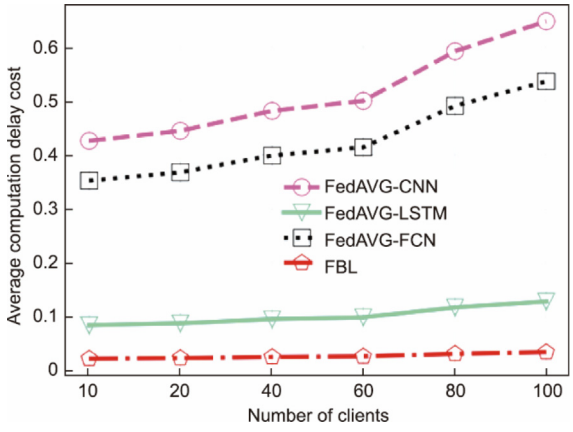

Fig. 8. Average computation delay cost for different numbers of clients.

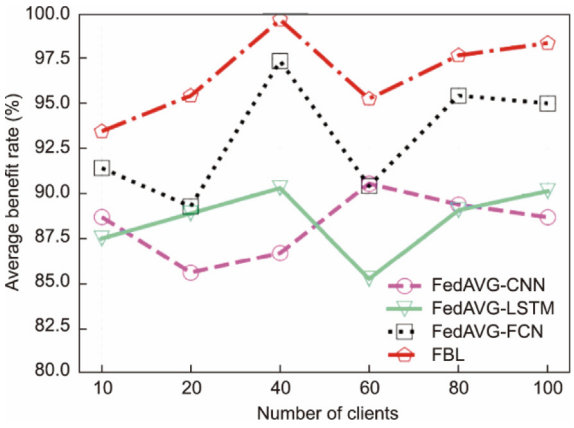

Fig. 9. Average benefit rate with the MNIST dataset for different numbers of clients.

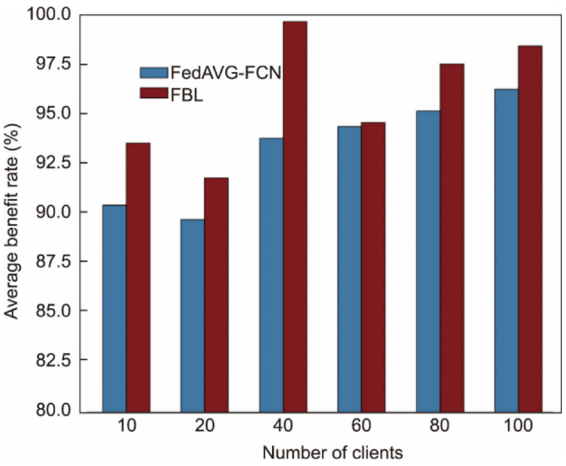

Fig. 10. Average benefit rate with the car evaluation dataset for different numbers of clients.

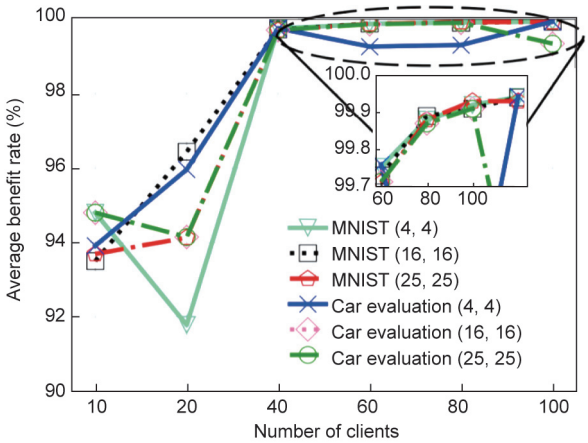

Fig. 11. Average benefit rates for different datasets.

7. Conclusions

In this paper, we proposed an asynchronous FBL framework for knowledge sharing in the IoV. The FBL integrates BL and FL, which not only protects data privacy but also accelerates the training efficiency of local models and improves the accuracy of federated aggregation. To enhance the performance of local training, an improved BFCM was proposed. To solve the resource allocation and RIS controlling problem, we divided it into two convex

subproblems. In order to reduce the communication cost, we used a DDFP algorithm to allocate network bandwidth and configure the RIS between time slots. An RFL algorithm was proposed to enhance the benefits to clients in order to compensate for their costs during communication and computation. The simulation results showed that our algorithm improves the classification accuracy by at least 2.0% on average, and the revenue of clients is improved by roughly 2.5% compared with that from other algorithms. Regarding reward distribution, our algorithm can approach 100% and maximizes users' benefits. Moreover, the efficiency of our algorithm on large-scale data sets is increased by about six times, with many vehicular nodes. For future work, we will investigate the client selection problem for FL.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (62371116 and 62231020), in part by the Science and Technology Project of Hebei Province Education Department (ZD2022164), in part by the Fundamental Research Funds for the Central Universities (N2223031), in part by the Open Research Project of Xidian University (ISN24-08) and Key Laboratory of Cognitive Radio and Information Processing, Ministry of Education (Guilin University of Electronic Technology, China, CRKL210203).

Compliance with ethics guidelines

Xiaoming Yuan, Jiahui Chen, Ning Zhang, Qiang (John) Ye, Changle Li, Chunsheng Zhu, and Xuemin Sherman Shen declare that they have no conflict of interest or financial conflicts to disclose.

References

[1] Niu Z, Shen XS, Zhang Q, Tang Y. Space–air–ground integrated vehicular network for connected and automated vehicles: challenges and solutions. Intell Converg Netw 2020;1(2):142–69.
[2] Xu M, Ng WC, Lim WYB, Kang J, Xiong Z, Niyato D, et al. A full dive into realizing the edge-enabled metaverse: visions, enabling technologies, and challenges. IEEE Commun Surv Tutor 2023;25(1):656–700.
[3] Chai H, Leng S, Chen Y, Zhang K. A hierarchical blockchain-enabled federated learning algorithm for knowledge sharing in Internet of Vehicles. IEEE Trans Intell Transp Syst 2021;22(7):3975–86.
[4] Wang W, Xia F, Nie H, Chen Z, Gong Z, Kong X, et al. Vehicle trajectory clustering based on dynamic representation learning of Internet of Vehicles. IEEE Trans Intell Transp Syst 2021;22(6):3567–76.
[5] Wang Z, Shi Y, Zhou Y, Zhou H, Zhang N. Wireless-powered over-the-air computation in intelligent reflecting surface-aided IoT networks. IEEE Internet Things J 2021;8(3):1585–98.
[6] Di Renzo M, Zappone A, Debbah M, Alouini MS, Yuen C, de Rosny J, et al. Smart radio environments empowered by reconfigurable intelligent surfaces: how it works, state of research, and the road ahead. IEEE J Sel Areas Commun 2020;38 (11):2450–525.
[7] Du H, Wang J, Niyato D, Kang J, Xiong Z, Zhang J, et al. Semantic communications for wireless sensing: RIS-aided encoding and selfsupervised decoding. 2022. arXiv:2211.12727v1.
[8] Yuan X, Chen J, Zhang N, Zhu C, Ye Q, Shen XS. FedTSE: low-cost federated learning for privacy-preserved traffic state estimation in IoV. In: Proceedings of 2022 IEEE Conference on Computer Communications Workshops, 2022 May 2–5, New York City, NY, USA; 2022.
[9] Nguyen DC, Cheng P, Ding M, Lopez-Perez D, Pathirana PN, Li J, et al. Enabling AI in future wireless networks: a data life cycle perspective. IEEE Commun Surv Tutor 2021;23(1):553–95.
[10] Ni W, Liu Y, Eldar YC, Yang Z, Tian H. STAR-RIS integrated nonorthogonal multiple access and over-the air federated learning: framework, analysis, and optimization. IEEE Internet Things J 2022;9(18):17136–56.
[11] Feng J, Zhang W, Pei Q, Wu J, Lin X. Heterogeneous computation and resource allocation for wireless powered federated edge learning systems. IEEE Trans Commun 2022;70(5):3220–33.
[12] Chen CLP, Liu Z. Broad learning system: an effective and efficient incremental learning system without the need for deep architecture. IEEE Trans Neural Netw Learn Syst 2018;29(1):10–24.
[13] Yuan X, Chen J, Zhang N, Fang X, Liu D. A federated bidirectional connection broad learning scheme for secure data sharing in Internet of Vehicles. China Commun 2021;18(7):117–33.
[14] Yu H, Liu Z, Liu Y, Chen T, Cong M, Weng X, et al. A fairness-aware incentive scheme for federated learning. In: Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society; 2020 Feb 7–8; New York City, NY, USA; 2020.
[15] Yang B, Cao X, Huang C, Yuen C, Di Renzo M, Yong LG, et al. Federated spectrum learning for reconfigurable intelligent surfaces-aided wireless edge networks. IEEE Trans Wirel Commun 2022;21(11):9610–26.
[16] Li L, Ma D, Ren H, Wang P, Lin W, Han Z. Towards energy-efficient multiple IRSs: federated learning based configuration optimization. IEEE Trans Green Commun Netw 2022;6(2):755–65.
[17] Wang Z, Qiu J, Zhou Y, Shi Y, Fu L, Chen W, et al. Federated learning via intelligent reflecting surface. IEEE Trans Wirel Commun 2022;21 (2):808–22.
[18] Huang S, Wang S, Wang R, Wen M, Huang K. Reconfigurable intelligent surface assisted mobile edge computing with heterogeneous learning tasks. IEEE Trans Cogn Commun Netw 2021;7(2):369–82.
[19] Liu L, Zhao M, Yu M, Jan MA, Lan D, Taherkordi A. Mobility-aware multi-hop task offloading for autonomous driving in vehicular edge computing and networks. IEEE Trans Intell Transp Syst 2023;24(2):2169–82.
[20] Mao R, Cui R, Chen CLP. Broad learning with reinforcement learning signal feedback: theory and applications. IEEE Trans Neural Netw Learn Syst 2022;33 (7):2952–64.
[21] Peng X, Ota K, Dong M. A broad learningdriven network traffic analysis system based on fog computing paradigm. China Commun 2020;17(2):1–13.
[22] Guo L, Li R, Jiang B. An ensemble broad learning scheme for semisupervised vehicle type classification. IEEE Trans Neural Netw Learn Syst 2021;32 (12):5287–97.
[23] Wei X, Zhao J, Zhou L, Qian Y. Broad reinforcement learning for supporting fast autonomous IoT. IEEE Internet Things J 2020;7(8):7010–20.
[24] Liu Y, Yu JJQ, Kang J, Niyato D, Zhang S. Privacy-preserving traffic flow prediction: a federated learning approach. IEEE Internet Things J 2020;7 (8):7751–63.
[25] Du Z, Wu C, Yoshinaga T, Yau KLA, Ji Y, Li J. Federated learning for vehicular Internet of Things: recent advances and open issues. IEEE Open J Comput Soc 2020;1:45–61.
[26] Lu Y, Huang X, Zhang K, Maharjan S, Zhang Y. Blockchain empowered asynchronous federated learning for secure data sharing in Internet of Vehicles. IEEE Trans Veh Technol 2020;69(4):4298–311.
[27] Le J, Lei X, Mu N, Zhang H, Zeng K, Liao X. Federated continuous learning with broad network architecture. IEEE Trans Cybern 2021;51(8):3874–88.
[28] Le THT, Tran NH, Tun YK, Nguyen MNH, Pandey SR, Han Z, et al. An incentive mechanism for federated learning in wireless cellular networks: an auction approach. IEEE Trans Wirel Commun 2021;20(8):4874–87.
[29] Liu T, Di B, An P, Song L. Privacy-preserving incentive mechanism design for federated cloud–edge learning. IEEE Trans Netw Sci Eng 2021;8 (3):2588–600.
[30] Chu Z, Xiao P, Shojafar M, Mi D, Mao J, Hao W. Intelligent reflecting surface assisted mobile edge computing for Internet of Things. IEEE Wirel Commun Lett 2021;10(3):619–23.
[31] Huang C, Zappone A, Alexandropoulos GC, Debbah M, Yuen C. Reconfigurable intelligent surfaces for energy efficiency in wireless communication. IEEE Trans Wirel Commun 2019;18(8):4157–70.
[32] Gong X, Zhang T, Chen CLP, Liu Z. Research review for broad learning system: algorithms, theory, and applications. IEEE Trans Cybern 2022;52 (9):8922–50.
[33] Ko B, Liu K, Son SH, Park KJ. RSU-assisted adaptive scheduling for vehicle-to-vehicle data sharing in bidirectional road scenarios. IEEE Trans Intell Transp Syst 2021;22(2):977–89.
[34] Mao S, Zhang N, Liu L, Wu J, Dong M, Ota K, et al. Computation rate maximization for intelligent reflecting surface enhanced wireless powered mobile edge computing networks. IEEE Trans Veh Technol 2021;70 (10):10820–31.
[35] Zhang L, Li J, Lu G, Shen P, Bennamoun M, Shah SAA, et al. Analysis and variants of broad learning system. IEEE Trans Syst Man Cybern Syst 2022;52 (1):334–44.
[36] Boyd S, Vandenberghe L, Faybusovich L. Convex optimization. IEEE Trans Autom Control 2006;51(11):1859.
[37] Ng KL, Chen Z, Liu Z, Yu H, Liu Y, Yang Q. A multi-player game for studying federated learning incentive schemes. In: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence; 2020 Jul 22; Yokohama, Japan; 2020.