A Federated Bidirectional Connection Broad Learning Scheme for Secure Data Sharing in Internet of Vehicles

Xiaoming Yuan[1], Jiahui Chen[1], Ning Zhang[2], Xiaojie Fang[3], Didi Liu[4,*]

[1] Qinhuangdao Branch Campus, Northeastern University, Qinhuangdao 066004, China
[2] Department of Electrical and Computer Engineering, University of Windsor, Windsor, ON N9B 3P4, Canada
[3] Department of Electronics and Information Engineering, Harbin Institute of Technology, Harbin 150006, China
[4] College of Electronic Engineering, Guangxi Normal University, Guilin 541004, China
[*] The corresponding author, email: ldd866@gxnu.edu.cn

Abstract: Data sharing in Internet of Vehicles (IoV) makes it possible to provide personalized services for users by service providers in Intelligent Transportation Systems (ITS). As IoV is a multi-user mobile scenario, the reliability and efficiency of data sharing need to be further enhanced. Federated learning allows the server to exchange parameters without obtaining private data from clients so that the privacy is protected. Broad learning system is a novel artificial intelligence technology that can improve training efficiency of data set. Thus, we propose a federated bidirectional connection broad learning scheme (FeBBLS) to solve the data sharing issues. Firstly, we adopt the bidirectional connection broad learning system (BiBLS) model to train data set in vehicular nodes. The server aggregates the collected parameters of BiBLS from vehicular nodes through the federated broad learning system (FedBLS) algorithm. Moreover, we propose a clustering FedBLS algorithm to offload the data sharing into clusters for improving the aggregation capability of the model. Some simulation results show our scheme can improve the efficiency and prediction accuracy of data sharing and protect the privacy of data sharing.

Keywords: federated learning; broad learning system; deep learning; Internet of Vehicles; data privacy

Received: Oct. 15, 2020
Revised: Dec. 15, 2020
Editor: Celimuge Wu

I. INTRODUCTION

In the 5G era and beyond, low-latency and high-efficiency networking make it possible to share data in mining to improve user experience or system performance [1]. Artificial intelligence technology is used widely in data sharing. However, data mining and sharing will lead to data privacy issues because the collected data belongs to the private data of customers, which usually includes the habits of driving experience, and can cause privacy leakage. Therefore, efficient solutions are needed to solve the problem of privacy protection in data sharing in Internet of Vehicles (IoV).

Wang et al. [2] proposed a data-sharing scheme based on semi-supervised learning in federated learning environment. Xu et al. [3] proposed a data sharing framework of double-masking protocol that enables clients to verify the operation of cloud servers. Wei et al. [4] proposed a novel framework to solve data sharing problem for privacy between two clients. With the increasing number of vehicles in the IoV, the efficiency of applying these works to IoV will become relatively low.

Federated learning [5] is a novel distributed learning technology widely used to solve the privacy problem of data sharing in IoV. Lu et al. [6] proposed a blockchain empowered asynchronous algorithm based on federated learning. Ye et al. [7] proposed a selective model aggregation approach in order to overcome asymmetry problem in federated learning. All

these works are based on deep learning methods which involve complex data training. Compared with deep learning method, broad learning system (BLS) is a novel efficient and reliable network technology which does not need complex and deep network [8]. Although the BLS structure is relatively simple, it can handle many problems such as image processing [9], text classification [10, 11]. Thus, federated learning combined with BLS has great potential to improve the data sharing in IoV.

However, the IoV environment is a complete dynamic system [12, 13]. The mobility of vehicles and unreliable inter-vehicle communication. Firstly, the efficiency of models in federated learning environment should be considered. At present, data sharing is based on deep learning models in federated learning which are time consuming to wait local vehicular clients to upload parameters. Secondly, we should guarantee the reliability of shared data. When providers share unqualified data such as few slot data and redundant data, it is necessary to improve the utilization of these data as much as possible. Finally, BLS is an efficient technology, but not much work has been reported on data sharing in IoV by using BLS.

In this paper, we propose a federated bidirectional connection broad learning scheme (FeBBLS) to solve data sharing problem in IoV. In FeBBLS, we propose the bidirectional connection broad learning system model (BiBLS) to train data set in local vehicular nodes. In order to aggregate the parameters collected from vehicle nodes by server, we propose federated broad learning system (FedBLS) aggregation algorithm. Because there may exist unbalance data phenomenon in some vehicle nodes, we use transfer learning and unsupervised clustering approach to optimize FedBLS algorithm. In a nutshell, the contributions of this work can be summarized as follows:

- Firstly, we propose the federated bidirectional connection broad learning scheme (FeBBLS) to solve data sharing in IoV. In local vehicular nodes, we adopt BiBLS to train data set in order to improve the efficiency of communication for server.
- Secondly, the proposed FedBLS algorithm enables servers to aggregate parameters uploaded from vehicle nodes in IoV environment in real-time. The server downloads the parameters and uses the weighted average method according to the size of the data set in the vehicle for parameters aggregation.
- Thirdly, we adopt transfer learning to optimize FedBLS algorithm, namely TF-FedBLS algorithm, when vehicle nodes have unbalanced data phenomenon. We add the dense layer after BiBLS and adopt gradient-descent method to fine-tune the parameters.
- Finally, we use an unsupervised clustering approach to enhance the ability of FedBLS. We carry out unsupervised clustering according to the distance between vehicles and road side unit (RSU), then we use TF-FedBLS algorithm to deal with the data sharing of vehicle-to-vehicle (V2V), and finally we use FedBLS algorithm to conduct the data sharing of vehicle-to-RSU (V2R).

The rest of the paper is organized as follows. Related works of BLS and federated learning are reviewed in Section II. In Section III, the federated bidirectional connection broad learning scheme is proposed to solve the data sharing in IoV. In Section IV, simulation results are provided to evaluate the performance of the proposed scheme. The conclusions of the paper and future work are given in Section V.

II. PRELIMINARIES

2.1 Related Work

With the development of BLS, there are some novel BLS structures that can modify or extend the traditional structure in order to improve the performance. Chen et al. [9] proposed cascade BLS to extract the features of data set. Zhang et al. [14] analyzed the shortcoming of traditional BLS, extended the structure of the traditional model and increased its functions. In order to prevent the BLS model from over fitting during training, the authors in [14] adopted the method which randomly discards the node groups of the enhancement layer. Chu et al. [11] adopted $L2$ penalty on the parameters in order to prevent over fitting. Han et al. [15] proposed structured manifold BLS (SM-BLS) to solve chaotic time series prediction problem. Liu et al. [16] proposed a modified BLS model based on K-means algorithm to supervise learning.

In addition to considering the efficiency of the model, the network layer should also pay attention to the efficiency in order to shorten response time. Be-

cause the IoV is a mobile and complex environment, some computations with large tasks can be offloaded to multi-access edge computing (MEC). Typically, it can be offloaded to cloud computing servers [17] or fog computing [18] servers to improve computing efficiency. Rihan et al. [19] proposed an AI-enabled vehicular network architecture based on fog computing which can achieve low-latency communications in vehicle-to-everything (V2X). Quan et al. [20] proposed a scheme to vehicular content sharing based on edge computing. Wu et al. [21] proposed a routing scheme to solve multi-access vehicular problem in edge computing. Zhan et al. [22] used deep reinforcement learning (DRL) to solve computation offloading scheduling problem. DRL based on 5G can optimize the mobile network slicing [23]. In [24], the authors proposed a deep learning method based on software defined network (SDN) to control the network traffic. Liu et al. [25] proposed a scheme that used DRL to optimize resource allocation in vehicle edge computing (VEC). Quan et al. [26] proposed a framework to fuse the traffic data such as traffic flow data and information flow data for path planning based on edge computing.

In IoV, content-centric data dissemination takes place between vehicles in V2V. The content-centric networking (CCN) [27, 28] enables centric data dissemination without knowing the IP address. Gulati et al. [29] used convolutional neural networks (CNN) to identify the ideal vehicle pairs in the vehicular network. Dai et al. [30] proposed a vehicular edge computing network framework based on DRL and blockchain to cache content which can protect the data privacy. In [31], the authors adopted spectral clustering method [32] to cluster vehicular nodes. Effective clustering of vehicle nodes is beneficial to the cooperative work of vehicles in the same cluster in federated environment.

As for the privacy of data sharing in federated learning, most of current works focused on data security and parameters aggregation effectiveness. Xu et al. [3] proposed a federated learning framework of double-masking protocol that enables clients to verify the operation of cloud servers. Lyu et al. [33] proposed a fairness-based federated learning framework to ensure the reliability and exclusiveness of the model while contributing to the server. Synchronous federated learning [34, 5] is a new distributed learning to solve parameters aggregation problem, while ensuring the privacy of client data, these local clients train the model, and then parameters are sent to the server. In order to prevent the server from waiting for a long time to upload parameters from clients, Chen et al. [35] proposed a temporally weighted asynchronous aggregation algorithm in order to reduce the communication cost. Zhu et al. [36] adopted multi-objective evolutionary algorithm to optimize synchronous federated learning.

Federated learning is often used in conjunction with other technologies for data sharing is widely used in IoV to preserve privacy. The authors in [6] proposed blockchain empowered federated learning scheme in data sharing for data security. The combination of federated learning and IoV enable the IoV environment to have better efficiency, and better privacy, shorter response time and stronger utility [13]. The authors in [37] proposed a novel scheme for traffic flow prediction in vehicular networks based on federated learning. Zhang et al. [31] adopted DRL to process resource allocation and mode selection in federated environment. Chai et al. [38] proposed a scheme to share knowledge based on federated learning in IoV for preserving privacy. To improve the efficiency of federated learning, we propose a federated bidirectional connection broad learning scheme.

2.2 Definition of Broad Learning System

The BLS is a novel model which does not need the deep neural network layers. In BLS, gradient descent is not required for parameters training, which greatly improves the efficiency of the model.

Figure 1 shows the structure of the traditional BLS, the BLS includes mapped feature nodes layer and enhancement nodes layer. The mapped feature nodes layer includes n group nodes. Each group is connected to the input X by the formula $Z_i = \phi(XW_{e_i} + \beta_{e_i}), i = 1, 2, ..., n$, where W_{e_i} and β_{e_i} are the random weights and biases. ϕ represents the activation function.

All of the feature nodes can be represented as $Z^n = [Z_1, Z_2, ..., Z_n]$. Then, the output of mapped feature nodes layer is connected to the enhancement nodes layer. In the enhancement nodes layer, it includes m group nodes. The value of each group of enhancement nodes is expressed as the formula: $H_j =$

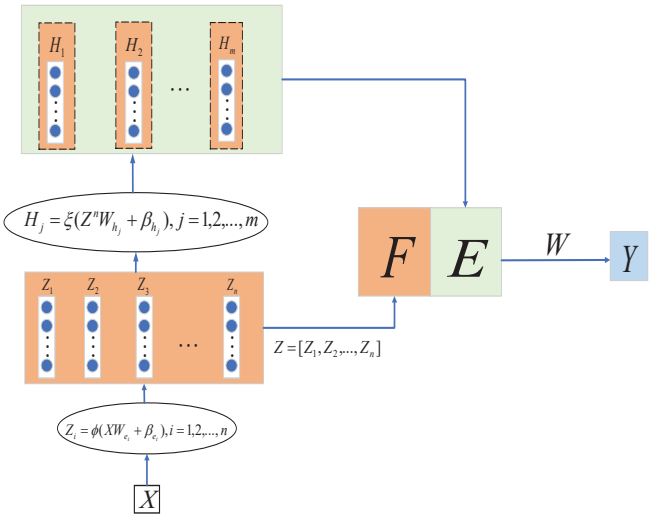


Figure 1. *The BLS Structure.*

$\xi(Z^n W_{h_m} + \beta_{h_m}), j = 1, 2, ..., m$, where W_{h_m} and β_{h_m} are the random weights and biases with the proper dimensions. ξ represents the activation function.

Therefore, the generalized BLS model can be expressed as the formula $Y = [Z_1, ..., Z_n|H_1, ..., H_m]W^m$, and $Y = [Z^n|H^m]W^m$, where $W^m = [Z^n|H^m]^+Y$ is the weight that needs to be required, and Y is the output of data set. We can use the ridge regression algorithm [39] to get the value of W^m by the next formula:

$$W^m = (\lambda I + A^T A)^{-1} A^T Y. \tag{1}$$

Finally, we have:

$$A^+ = \lim_{\lambda \longrightarrow 0} (\lambda I + A^T A)^{-1} A^T, \tag{2}$$

where $A = [Z^n|H^m]$ denotes the expanded input matrix and A^+ is the pseudo-inverse of A, $(\cdot)^+$ means the pseudo-inverse operation, and λ denotes the regularization coefficient.

When defining the model, initializing the parameters of BLS, such as W_{e_i}, β_{e_i}, W_{h_m} and β_{h_m}. $X \in \mathbb{R}^{N \times \mathbb{F}}$ means the X has N samples, each sample has $\mathbb{F}$ features. Thus, $W_{e_i} \in \mathbb{R}^{\mathbb{F} \times \mathbb{B}}$ and $Z_i \in \mathbb{R}^{N \times \mathbb{B}}$, where $\mathbb{B}$ is the parameters we need to initialize. In the same way, $H_i \in \mathbb{R}^{N \times \mathbb{B}}$ can be obtained from $Z_i \in \mathbb{R}^{N \times \mathbb{B}}$ and $W_{h_m} \in \mathbb{R}^{\mathbb{B} \times \mathbb{B}}$.

Therefore, if we have training data set X^{train} and it corresponds to output Y^{train}, we can use BLS to get the W^m. Then, we can use the test data set X^{test} by the (1) and (2) to get A, and use the formula $Y^{predict} = AW^m$ to get the predicted output $Y^{predict}$.

2.3 Definition of FederatedAveraging

The FederatedAveraging (FedAVG) algorithm is widely used in federated learning. In the following, we briefly introduced the FedAVG algorithm. In the federated learning environment, the deep learning model adopts the gradient descent to update parameters.

Algorithm 1. *FedAVG.*

1: **function** SERVEREXECUTION
2: initialize ω_0;
3: send the initial parameters to the clients;
4: **for** each round $t = 1, 2, ...$ **do**
5: $m \leftarrow max(C \cdot K, 1)$;
6: $S_t \leftarrow$ (random set of m clients);
7: **for** each client k $\in S$ in parallel **do**
8: $W_{t+1}^k \leftarrow$ ClientUpdate(k, W_t);
9: **end for**
10: $W_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} (W_{t+1}^k)$;
11: **end for**
12: **end function**
13: **function** CLIENTUPDATE(k,ω)
14: $\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size B);
15: **for** each local epoch i from 1 to E **do**
16: **for** batch $b \in \mathcal{B}$ **do**
17: $\omega \leftarrow \omega - \eta \nabla l(w; b)$;
18: **end for**
19: **end for**
20: return ω to server;
21: **end function**

The Algorithm 1 includes two components $ServerExecution$ and $ClientUpdate$. In the subfunction $ServerExecution$, line 2 initializes ω_0 for deep learning model. Line 3 indicates that the server will do multiple rounds of communication. Line 4 indicates that the server randomly selects m devices in each round of communication. For formula $m \leftarrow max(C \cdot K, 1)$, C is the constant of $[0, 1]$, which is mainly used to adjust the number of devices participating in the parameters of server update in each round. Lines 5-7 indicate that the server selects S_t devices from m devices for synchronous communication to update server parameters and device parameters. Line 8 shows the global W_{t+1}^k is updated by weighted average method, the weighted coefficient is related to the total data size n and the size of the data set n_k owned by each client.

In the subfunction $ClientUpdate$, line 14 divides the device data $\mathcal{P}_k$ into samples of size $\mathcal{B}$. Lines 15-19 use the gradient descent to conduct multiple rounds of training for the samples and update the parameters by minimizing loss function $l(w;b)$. Line 20 sends the final parameters to the server.

III. FEDERATED BIDIRECTIONAL CONNECTION BROAD LEARNING SCHEME (FEBBLS)

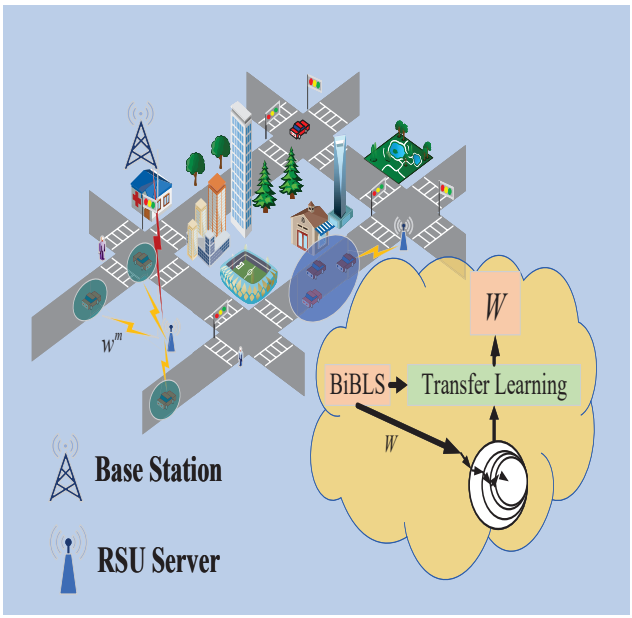


Figure 2. *Federated bidirectional connection broad learning scheme.*

The Figure 2 shows that our proposed scheme. The vehicle and RSU use long term evolution (LTE) technology for communication. RSU preprocesses the received data and sends it to server. The server will initialize the parameters according to the BiBLS model structure defined by (1), and then send it to local vehicle nodes via RSU. Some nodes are not disconnected from the server, the unsupervised clustering method is used to unload the data sharing to the cluster head and the nodes within the cluster for data sharing. In each cluster, these vehicular nodes can fine-tune the parameters obtained by BiBLS based on transfer learning to solve unbalanced data phenomenon and enhance the performance of the local model. The cluster can share parameters with RSU based on FedBLS algorithm.

3.1 Bidirectional Connection Broad Learning System Model (BiBLS)

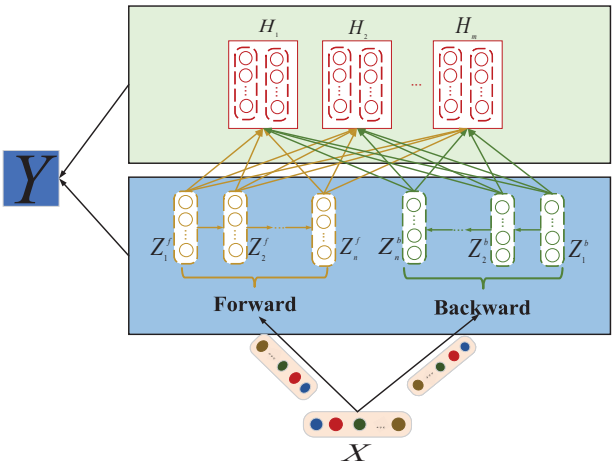


Figure 3. *The BiBLS Structure.*

In order to capture the hierarchical features in both directions, we propose the BiBLS. As seen in Figure 3, the forward mapped feature nodes layer and the backward mapped feature nodes layer all have n node groups. We divide X into forward input X_f and backward input X_b. We feed X_f into the forward mapping layer to obtain $Z_i^f, i = 1, 2, ..., n$, and then feed X_b backward into the backward mapping layer to obtain $Z_i^b, i = 1, 2, ..., n$. The each group of forward mapped feature nodes layer by the following formula:

$$Z_i^f = \begin{cases} \phi(XW_{e_i}^f + \beta_{e_i}^f), i = 1, \\ \phi(Z_{i-1}^f W_{e_i}^f + \beta_{e_i}^f), i = 2, 3, ..., n \end{cases}, \quad (3)$$

where $W_{e_i}^f$ and $\beta_{e_i}^f$ are random parameters with the proper dimensions. ϕ represents the activation function.

The each group of backward mapped feature nodes layer by the following formula:

$$Z_i^b = \begin{cases} \phi(XW_{e_i}^b + \beta_{e_i}^b), i = 1, \\ \phi(Z_{i-1}^b W_{e_i}^b + \beta_{e_i}^b), i = 2, 3, ..., n \end{cases}, \quad (4)$$

where $W_{e_i}^b$ and $\beta_{e_i}^b$ are random parameters with the proper dimensions. ϕ represents the activation function.

Thus, we define $Z_f^n = [Z_1^f, Z_2^f, ..., Z_n^f]$ and $Z_b^n =$

$[Z_1^b, Z_2^b, ..., Z_n^b]$. So we feed Z_f^n and Z_b^n into the enhancement nodes layer to get the forward and reverse hidden states respectively $H_i^f, i = 1, 2, ..., m$ and $H_i^b, i = 1, 2, ..., m$. The H_i^f and H_i^b are updated as follows:

$$\begin{aligned} H_i^f &= \xi(Z_f^n W_{h_m}^f + \beta_{h_m}^f), \\ H_i^b &= \xi(Z_b^n W_{h_m}^b + \beta_{h_m}^b), \end{aligned} \tag{5}$$

where $W_{h_m}^f$, $\beta_{h_m}^f$, $W_{h_m}^b$ and $\beta_{h_m}^b$ are random parameters with the proper dimensions. ξ represents the activation function.

Our model can be summarized as $Y = [Z^n|H^m]W^m$, and Z^n, H^m are obtained by follows:

$$\begin{aligned} Z^n &= \alpha_z^f Z_f^n + \alpha_z^b Z_b^n, \\ H^m &= \alpha_h^f H_f^m + \alpha_h^b H_b^m, \end{aligned} \tag{6}$$

where α_z^f and α_z^b are adjustable coefficients of Z_f^n and Z_b^n. The different values of α_z^b and α_z^b indicate the different importance of Z_f^n and Z_b^n to Z^n. α_h^f and α_h^b are adjustable coefficients of H_f^m and H_b^m. The different values of α_h^f and α_h^b indicate the different importance of H_f^m and H_b^m to H^m.

The traditional BLS model can do incremental learning. Thus, BiBLS uses some efficient ways to increase the enhancement nodes, mapped feature nodes and the training data, without retraining all parameters.

3.1.1 Increment of Additional Enhancement Nodes

Suppose that adding the $(m+1)$th enhancement node groups to the model. Denote $A = [Z^n|H^m]$, such that the input matrix A is updated by:

$$\begin{aligned} A = [A|\alpha_h^f \xi(Z_f^n W_{h_{m+1}}^f + \beta_{h_{m+1}}^f) \\ + \alpha_h^b \xi(Z_b^n W_{h_{m+1}}^b + \beta_{h_{m+1}}^b)], \end{aligned} \tag{7}$$

where $W_{h_{m+1}}^f$, $\beta_{h_{m+1}}^f$, $W_{h_{m+1}}^b$ and $\beta_{h_{m+1}}^b$ are randomly generated, so the pseudoinverse of A could be updated as:

$$(A)^+ = \begin{bmatrix} (A)^+ - DB^T \\ B^T \end{bmatrix}, \tag{8}$$

Algorithm 2. *BiBLS: increment of $m+1$ Additional Enhancement Node Groups.*

Input: training samples X;
Output: W;

1: Set the forward feature mapping group Z_f^n by (3);
2: Set the backward feature mapping group Z_b^n by (4);
3: **for** $j = 1$ to m **do**
4: Random $W_{h_j}^f$, $\beta_{h_j}^f$;
5: Calculate $H_j^f = \xi(Z_f^n W_{h_j}^f + \beta_{h_j}^f)$;
6: Random $W_{h_j}^b$, $\beta_{h_j}^b$;
7: Calculate $H_j^b = \xi(Z_b^n W_{h_j}^b + \beta_{h_j}^b)$;
8: **end for**
9: Set the enhancement nodes group H^m by (6);
10: Set A and calculate $(A)^+$ with (2);
11: **while** *The number of enhancement nodes increased is less than the set number* **do**
12: Random $W_{h_{m+1}}^f$, $\beta_{h_{m+1}}^f$;
13: Calculate $H_{m+1}^f = [\xi(Z_f^n W_{h_{m+1}}^f + \beta_{h_{m+1}}^f)]$;
14: Random $W_{h_{m+1}}^b$, $\beta_{h_{m+1}}^b$;
15: Calculate $H_{m+1}^b = [\xi(Z_b^n W_{h_{m+1}}^b + \beta_{h_{m+1}}^b)]$;
16: Set A by (7);
17: Update $(A)^+$ and W^m by (8), (9), (10);
18: Set $W = W^m$;
19: **end while**

where $D = (A)^+(\alpha_h^f \xi(Z_f^n W_{h_{m+1}}^f + \beta_{h_{m+1}}^f) + \alpha_h^b \xi(Z_b^n W_{h_{m+1}}^b + \beta_{h_{m+1}}^b))$, which represents the intermediate variable.

$$B^T = \begin{cases} (1+D^T D)^{-1} B^T (A)^+, C = 0, \\ (C)^+, C \neq 0, \end{cases} \tag{9}$$

and $C = \alpha_h^f \xi(Z_f^n W_{h_{m+1}}^f + \beta_{h_{m+1}}^f) + \alpha_h^b \xi(Z_b^n W_{h_{m+1}}^b + \beta_{h_{m+1}}^b) - A^m D$, which represents the intermediate variable.

Finally, the new connection weights W^m could be updated by:

$$W^m = \begin{bmatrix} W^m - DB^T \\ B^T \end{bmatrix}. \tag{10}$$

Algorithm 3. *BiBLS: increment of $n+1$ Mapped Feature Node Groups.*

Input: training samples X;
Output: W;

Set the forward feature mapping group Z_f^n by (3);
Set the backward feature mapping group Z_b^n by (4);
for $j = 1$ to m **do**
    Random $W_{h_j}^f$, $\beta_{h_j}^f$;
    Calculate $H_j^f = \xi(Z_f^n W_{h_j}^f + \beta_{h_j}^f)$;
    Random $W_{h_j}^b$, $\beta_{h_j}^b$;
    Calculate $H_j^b = \xi(Z_b^n W_{h_j}^b + \beta_{h_j}^b)$;
end for
Set the enhancement nodes group H^m by (6);
Set A and calculate $(A)^+$ with (2);
while *The number of feature nodes increased is less than the set number* **do**
    Random $W_{e_{n+1}}^f$, $\beta_{e_{n+1}}^f$;
    Random $W_{e_{n+1}}^b$, $\beta_{e_{n+1}}^b$;
    Calculate Z_f^{n+1} and Z_b^{n+1} by (11);
    Random $W_{ex_i}^f$, $\beta_{ex_i}^f, i = 1, ..., m$;
    Random $W_{ex_i}^b$, $\beta_{ex_i}^b, i = 1, ..., m$;
    Set $H_{ex_m}^f$ and $H_{ex_m}^b$ by (12);
    Update A;
    Update A^+ and W^m by (13), (14), (15);
    Set $W = W^m$;
end while

3.1.2 Increment of Mapped Feature Nodes

Simply, if we want to add the $(n+1)$th feature mapping node groups, we denote as:

$$\begin{aligned} Z_f^{n+1} &= \phi(Z_f^n W_{e_{n+1}}^f + \beta_{e_{n+1}}^f), \\ Z_b^{n+1} &= \phi(Z_b^n W_{e_{n+1}}^b + \beta_{e_{n+1}}^b). \end{aligned} \tag{11}$$

The corresponding bidirectional status of enhancement nodes are randomly generated as follows:

$$\begin{aligned} H_{ex_m}^f &= [\xi(Z_f^{n+1} W_{ex_1}^f + \beta_{ex_1}^f), ..., \\ &\quad \xi(Z_f^{n+1} W_{ex_m}^f + \beta_{ex_m}^f)], \\ H_{ex_m}^b &= [\xi(Z_b^{n+1} W_{ex_1}^b + \beta_{ex_1}^b), ..., \\ &\quad \xi(Z_b^{n+1} W_{ex_m}^b + \beta_{ex_m}^b)], \end{aligned} \tag{12}$$

where $W_{ex_i}^f$, $\beta_{ex_i}^f$, $W_{ex_i}^f$ and $\beta_{ex_i}^f$ are randomly generated. Denote $A_{n+1} = [A|\alpha_z^f Z_f^{n+1} + \alpha_z^b Z_b^{n+1}|\alpha_h^f H_{ex_m}^f + \alpha_b^f H_{ex_m}^b]$, which is the upgrade of new mapped features and the corresponding enhancement nodes. The relatively upgraded pseudo-inverse matrix can be achieved as follows:

$$A^+ = \begin{bmatrix} (A^+ - DB^T \\ B^T \end{bmatrix}, \tag{13}$$

where $D^T = A^+[\alpha_z^f Z_f^{n+1} + \alpha_z^b Z_b^{n+1}|\alpha_h^f H_{ex_m}^f + \alpha_b^f H_{ex_m}^b]$, which represents the intermediate variable.

$$B^T = \begin{cases} (1+D^T D)^{-1}(A_n^m)^+ D, C = 0, \\ (C)^+, C \neq 0, \end{cases} \tag{14}$$

and $C = [\alpha_z^f Z_f^{n+1} + \alpha_z^b Z_b^{n+1}|\alpha_h^f H_{ex_m}^f + \alpha_b^f H_{ex_m}^b] - A_n^m D$, which represents the intermediate variable.

Finally, the new weights can be defined as:

$$W^{m+1} = \begin{bmatrix} W^m - DB^T \\ B^T \end{bmatrix}. \tag{15}$$

3.1.3 Increment of new training data

If we need to add the new training data into the model, X_a represents the input matrix of the new data. The initial model includes n groups mapped feature nodes and m groups enhancement nodes. The A_n^m is the expanded input matrix. The extended input matrix of the new additional training data is defined as follows:

$$\begin{aligned} A_x = [&\phi(X_a W_{e_1} + \beta_{e_1}), .., \phi(X_a W_{e_n} + \beta_{e_n})| \\ &\xi(Z_x^n W_{h_1} + \beta_{h_1}), ..., \xi(Z_x^n W_{h_m} + \beta_{h_m})], \end{aligned} \tag{16}$$

where $Z_x^n = [\phi(X_a W_{e_1} + \beta_{e_1}), .., \phi(X_a W_{e_n} + \beta_{e_n})]$, and Z_x^n is the mapped matrix of the X_a. W_{e_i}, β_{e_i}, W_{h_j}, and β_{h_j} are random parameters. Thus, the new extended input matrix $(A_n^m)_x$ can be calculated as:

$$(A_n^m)_x = \begin{bmatrix} A_n^m \\ (A_x)^T \end{bmatrix}. \tag{17}$$

The pseudoinverse of A_n^m could be updated as follows:

$$((A_n^m)_x)^+ = [(A_n^m)^+ - BD^T|B], \tag{18}$$

where $D^T = (A_x)^T (A_n^m)^+$, which represents the intermediate variable.

$$B^T = \begin{cases} (1+D^T D)^{-1}(A_n^m)^+ D, C = 0, \\ (C)^+, C \neq 0, \end{cases} \tag{19}$$

and $C = (A_x)^T - D^T A_n^m$, which represents the intermediate variable.

Finally, the new connection weights could be expressed as:

$$W_n^m = W_n^m + (Y_a^T - (A_x)^T W_n^m)B, \tag{20}$$

where Y_a is the output matrix of the new training data.

Algorithm 4. *BiBLS:Increment of new training data X_a.*

Input: training samples X;
Output: W;

1: Set the forward feature mapping group Z_f^n by (3);
2: Set the backward feature mapping group Z_b^n by (4);
3: **for** $j = 1$ to m **do**
4: Random $W_{h_j}^f$, $\beta_{h_j}^f$;
5: Calculate $H_j^f = \xi(Z_f^n W_{h_j}^f + \beta_{h_j}^f)$;
6: Random $W_{h_j}^b$, $\beta_{h_j}^b$;
7: Calculate $H_j^b = \xi(Z_b^n W_{h_j}^b + \beta_{h_j}^b)$;
8: **end for**
9: Set the enhancement nodes group H^m by (6);
10: Set A^m and calculate $(A^m)^+$ with (2);
11: Calculate A_x by (16), update A_n^m by (17);
12: Update $(A_n^m)^+$ and W_n^m by (18), (19), (20);
13: Set $W = W^{m+1}$;

Algorithm 2 and Algorithm 3 show that when the BiBLS could not achieve the desired effect with the given number of node groups, we can increase the number of node groups in order to update the parameters that need to be uploaded to the server of the model, which greatly improves the efficiency of the model. Algorithm 4 shows the BiBLS model can be fed the training data efficiently. In federated learning, with the client's permission, more data can be used to participate in the training of the model, which can improve the training effect of the model while keeping the number of model parameters unchanged.

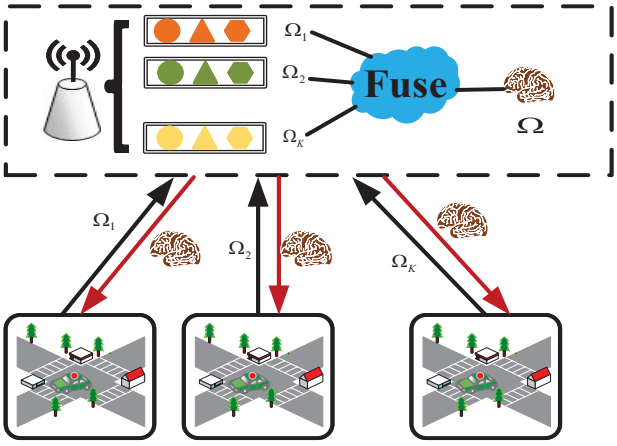


Figure 4. *The FedBLS for data sharing.*

3.2 Federated Broad Learning System (FedBLS) Algorithm for V2R

Local participating nodes rely on a large number of computing resources to update the model. Traditional FedAVG algorithm is the limited network resource that bottlenecks of aggregated local parameters from mobile nodes in IoV. In order to reduce the delay of data sharing in IoV, we use BiBLS and federated learning to achieve accurate and timely data sharing without privacy disclosure. The description of FedBLS is shown in Figure 4, and the main steps of the algorithm are as follows:

1) **Model initialization**. Vehicles and RSU use LTE technology for communication. RSU acts as a server for data exchange with vehicle nodes which act as clients. It initializes the parameters of BiBLS, and sends model Ω to mobile nodes. The local BiBLS of node is defined as Ω_i.

2) **Participate Nodes Selection**. In each communication, the RSU exchanges data with mobile nodes that meet certain conditions. Then, it randomly selects m clients from a certain proportion C of the total clients K. These vehicle nodes participate communication in parallel.

3) **Update parameters**. The vehicle node calculates W based on the local data set and sends it to the server. Without increment learning of BiBLS, they just upload the trained parameters to the server.

4) **Model Aggregation**. When training the model, there is no need for batch size segmentation of the data set. Thus, the server gets the global parameters W_g for each round of communication to aggregate

Algorithm 5. *FedBLS algorithm for data sharing in IoV.*

Input: Vehicle participate nodes $\mathbb{V} = \{V_1, V_2, ..., V_k\}$;
Output: Parameters $\mathbf{W}$;

1: Initialize w_{x_i},β_{x_i},w_{h_i},β_{h_i}, α_z^f, α_z^b, α_h^f, α_h^b;
2: $S \leftarrow$ (random set of $\mathbb{V}$ clients);
3: **for** each round t **do**
4: Send the initial parameters to all $V_s \in S$;
5: **for** each V_s in $\in S$ in parallel **do**
6: Calculate W_s according to the model Ω_s in this round by local data set;
7: **if** Iter!=0 **then**
8: **for** i=0:Iter **do**
9: Use Algorithm 2 and Algorithm 3 to update W_s;
10: **end for**
11: **else if** New training data set X_a is needed to add **then**
12: Use Algorithm 4 update W_s;
13: **end if**
14: return W_s to server;
15: **end for**
16: $W_g = \sum_{V_s \in S}^{S} \frac{n_s}{n}(W_s)$;
17: **end for**

the parameters of local vehicle nodes by the formula $W_g = \sum_{V_s \in S}^{S} \frac{n_s}{n}(W_s)$ in tth round.

The FedBLS does not make use of the gradient-descent method to update the model. If there is no increment learning of the local model in the data sharing, the node and RSU can update the global model with only one communication round. If the local training is not satisfactory or a new data set is added to the local node, incremental learning is required. Algorithm 5 shows the FedBLS algorithm with $Iter$ rounds incremental learning. In Algorithm 5, lines 6-14 show that the node uses BiBLS for incremental learning.

3.3 FedBLS with Transfer Learning for V2V

BiBLS improves the training effect of the model by increasing the node group of the internal structure. Although multiple groups of node can be added in mapping feature nodes layer or enhancement nodes layer when the learning of BiBLS is not ideal, parameters' size increases with the increasing of nodes, and the solution time of the W also increases. Thus, Gao et al, in [40] proposed a end-to-end BLS structure based on deep learning. However, when the samples of the node that can be trained are relatively less, parameters directly trained by BLS model may not achieve the expected effect.

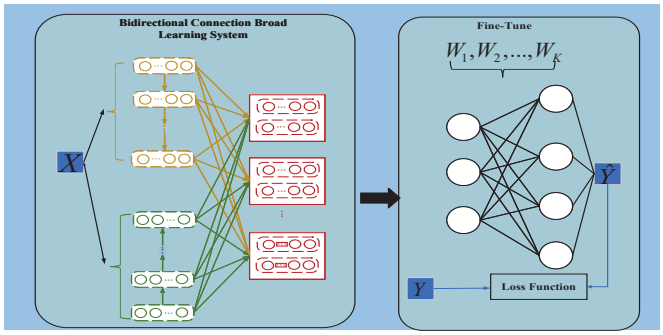


Figure 5. *FedBLS with Transfer Learning for V2V.*

The IoV has inter-vehicle communications. Thus, a vehicular node can share data with other nodes. In inter-vehicle communications, there are differences in the data between each node. For example, when a vehicular node has few samples, or when the client only contains samples of a certain label, the trained model parameters have no generalization. In order to solve these issues, we use transfer learning [41, 42] technology to improve the generalization ability of models. From the Figure 5, we use transfer learning combined FedBLS, dubbed (TF-FedBLS) algorithm to apply in the IoV in order to enhance the ability for aggregation. In inter-vehicle communications, a node uses a weighted average aggregation method to aggregate parameters from other nodes. However, there may be bad parameters from other nodes, so we add a neural network layer after the model to fine-tune W with freezing other parameters of the model. From the Figure 5, the first stage is that the vehicle use the local data set X to train the local BiBLS model. Through the mapped feature nodes layer and enhancement nodes layer, we can get the Z^n and H^m by the (3)- (6). In the second stage, we feed the W into the neural networks.

First, we initialize the neural networks to fine-tune model parameters. The full connection (FC) layer is defined in the TF-FedBLS. The weights and biases of the FC is set to W and zero, respectively. The purpose of this is to be able to update the model parameters using gradient-descent method. The updating process is described as Algorithm 6. From the Algorithm 6, the $ClientUpdate$ of TF-FedBLS is the same as the FedBLS. In $ServerExecution$, line 10 defines the

Algorithm 6. *TF-FedBLS.*

1: **function** SERVEREXECUTION
2: Initialize w_{x_i},β_{x_i},w_{h_i},β_{h_i}, $\alpha_z^f, \alpha_z^b, \alpha_h^f, \alpha_h^b$ of BiBLS model;
3: Send the initial parameters to the clients;
4: $m \leftarrow max(C \cdot K, 1)$;
5: $S \leftarrow$ (random set of m clients);
6: **for** each client k $\in S$ in parallel **do**
7: $W_k \leftarrow$ ClientUpdate(k);
8: **end for**
9: $W \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} W_k$;
10: Set $LinearNet().weight = W$;
11: $\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size B);
12: **for** each epoch i from 1 to E **do**
13: **for** batch $b \in \mathcal{B}$ **do**
14: Update W by gradient-descent method;
15: **end for**
16: **end for**
17: **end function**
18: **function** CLIENTUPDATE(k)
19: Calculate W_k by (1);
20: return W_k to server;
21: **end function**

one layer neural network $LinearNet$ and resets the weights. Lines 12-14 use gradient-descent method to update parameters.

3.4 Clustering Federated Broad Learning Data Sharing Algorithm in IoV

There is a distance requirement for the connection between RSU and the vehicle networking node. The ith vehicle v_i communicates with the RSU beyond d_0, and then it communicates with the nearest vehicle node. If it is within the communication distance range with the RSU, it will directly communicate with the RSU.

The vehicle connection topology graph $\mathcal{G}(V, E)$. Denotes the set of nodes is V, and E denotes the set of weights in inter-vehicle. We define the d_{ij} as the distance between node v_i and node v_j. We use FINCH Algorithm [43] which is an efficient unsupervised clustering algorithm to aggregate the vehicular nodes by the next formula:

$$A(i,j) = \begin{cases} 1, if \quad j = k_i \quad or \quad k_j = i \quad or \quad k_i = k_j, \\ 0, otherwise, \end{cases} \tag{21}$$

where k_i represents the nearest neighbor of the ith node, k_j represents the nearest neighbor of the jth node, and A is the adjacency matrix of IoV networks topology graph.

From the (21), the value in the adjacency matrix is one as long as the following:

1)$j = k_i$: It means the nearest neighbor representing the ith node is the jth node.

2)$k_j = i$: It means that the node i is the nearest neighbor to node j.

3)$k_i = k_j$: It means that the nearest neighbor of the ith node and the nearest neighbor of the jth node are the same.

Figure 6 shows the relationship between TF-FedBLS and FedBLS. In the inter-vehicle communications, the vehicular nodes will use TF-FedBLS algorithm to share parameters. The vehicular nodes will share parameters based on FedBLS. Algorithm 7 lists the process of cluster-based FedBLS algorithm. Assuming that nodes are divided into K categories. For each class, the node closest to the RSU is selected as the cluster head $C_k^h, k = 1, 2, .., K$, and then the RSU collects the uploading parameters by these cluster heads for federation aggregation. Each cluster head aggregates other vehicular nodes in the same cluster by $W_i = \sum_{k=1}^{K} n_k W_k / \sum_{k=1}^{K} n_k$, where n_k is the size of data set in the kth local node, W_k is the parameters which the kth node needs to share, and W_i means the parameters of the ith cluster head.

IV. SIMULATION RESULTS AND ANALYSES

4.1 Simulation Design

We use MNIST data set to evaluate our model, randomly allocate 80% of the training data and assign it to local devices as privacy data, and assign 20% to the server for test data. Each device selects 90% of the data to train models, 10% for test data. We use *OMNET++* combined *VEINS* to randomly generate the number of vehicle nodes on the map based on Simulation of Urban MObility (SUMO) [44]. The

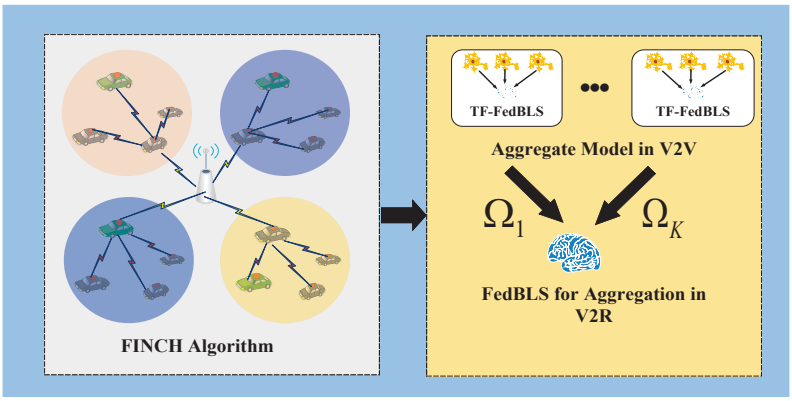


Figure 6. *Clustering FedBLS in IoV for Data Sharing.*

Algorithm 7. *Clustering FedBLS (cFedBLS) algorithm.*

Initialize K and get the graph $\mathcal{G}(V, E)$;
Calculate A by (21);
Calculate the cluster sets $\{C_1, C_2, .., C_K\}$ by FINCH Algorithm [43];
for $k = 1, 2, ..., K$ **do**
    Take the node closest to the RSU as the C_k^h in C_k;
end for
for each clustering center C_k^h **do**
    for each node v in category C_k^h **do**
        Execute TF-FedBLS algorithm;
    end for
    Execute FedBLS algorithm;
end for
The server downloads the parameters and update the BiBlS model;
The server sends the model to the clustering of each category;
The client returns W to server;

simulation scenario plotted by *OpenStreetMap*. The network layer parameters involved in IoV are shown in Table 1. Through the simulation of the IoV environment, we will obtain the connection status of V2X at each episode of IoV. All algorithms are running in $Pytorch$, and our stimulation environment is Intel(R) Core(TM) i7-5500U CPU, and 8GB RAM.

Table 1. *System parameters.*

Parameters	Value
Maximum communication distance	2000 m
Bit rate	6 Mbps
Thermal noise	-110 dBm
The number of RSUs	1

In this paper, the simulation parameters of models are set as follows:

FedAVG The architecture of the CNN used for the MNIST recognition tasks has two convolution layers (the first one has 6 output channels and the other has 16 output channels) and each convolution layer needs 2×2 max-pooling layer. Then, we add three full connected layers with Relu activation function, which have 120,84 neurons, respectively.

FedBLS The enhancement nodes layer includes 10 group nodes and the mapped features nodes layer includes 10 group nodes. In increment learning of BiBLS, we add 5 and 2 groups for each step of mapped features node groups and enhancement node groups.

TF-FedBLS The number of mapped feature node groups and enhanced node groups are the same as the FedBLS algorithm. In TF-FedBLS algorithm, we set one neural network to fine-tune the parameters of BiBLS.

4.2 Results And Analysis

We evaluate the performance of BiBLS under different adjustment coefficients. The different adjustment coefficients of mapped feature nodes layer and enhancement nodes layer means that we assign different attention to them. From the Figure 7, when the adjustment coefficients are $\alpha_z^f = 1, \alpha_z^b = 0, \alpha_h^f = 0.5, \alpha_h^b = 0.5$, the better effect of BiBLS can be obtained for 10-30 vehicles. However, when the adjustment coeffi-

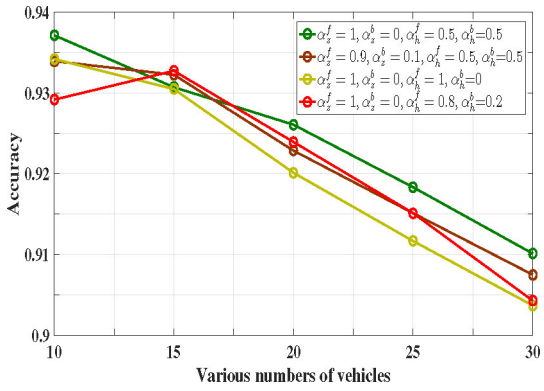


Figure 7. *The performance of BiBLS under different adjustment coefficients.*

cients are $\alpha_z^f = 1, \alpha_z^b = 0, \alpha_h^f = 0.8, \alpha_h^b = 0.2$, the best effect of BiBLS can be obtained for 15 vehicles. Particularly, when the adjustment coefficients are $\alpha_z^f = 1, \alpha_z^b = 0, \alpha_h^f = 1, \alpha_h^b = 0$, BiBLS structure becomes the traditional BLS structure. Thus, we believe that the bidirectional mechanism has a catalytic effect in BLS. In some cases, assigning different attention to hidden states can make the model work better.

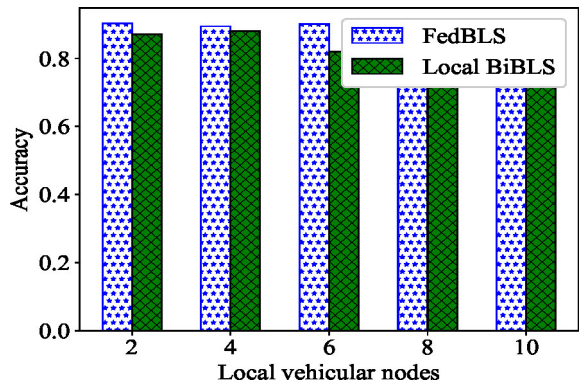


Figure 8. *The prediction effect of local BiBLS and FedBLS in the different local vehicles.*

Meanwhile, we also compare FedBLS with BiBLS. We use the BiBLS model for training in the vehicle node and validate the test data of the server. From the Figure 8, we can see the number of vehicle clients increases, the average accuracy decreases significantly. When we use FedBLS algorithm for parameters sharing, the accuracy varies steadily which proves that our algorithm works well in federated parameter sharing. With the increasing of local clients, the average accuracy is decreasing, because the local test data set is less. If we don't share the parameters, the loss of precision will be more serious. We use FedBLS to share parameters, the average accuracy is less affected by the number of clients, because each local client can have global parameters which means the predictive power of the model is enhanced.

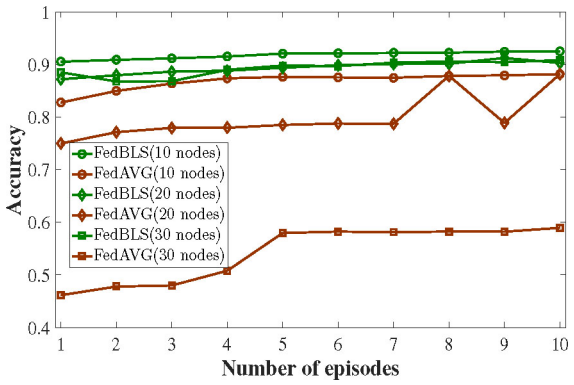


Figure 9. *The performance of increment learning under the FedBLS different for the number of vehicles.*

We compare our FedBLS algorithm with FedAVG. In federated learning, the training of models is generally based on CNN, but the disadvantage of CNN is that the model training takes a long time. BiBLS updates parameters to achieve a certain effect by increasing the number of node groups, instead of updating the parameters through gradient descent. From the Figure 9, when we use the FedBLS to test the global data set of the server, we can see the model converges quickly and with the increasing of episodes, the prediction effect of the global model has little change, and the effect is better than that of FedAVG algorithm with different nodes.

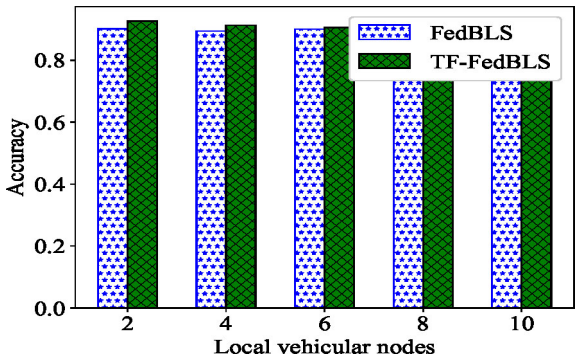


Figure 10. *The prediction effect of local TF-FedBLS and FedBLS in the different local vehicles.*

In V2V, vehicular nodes use TF-FedBLS to share data. Thus, we compare FedBLS with TF-FedBLS. We also use the BiBLS model for training in the vehicle node and validated the test data of the server. From the Figure 10, with the increasing of clients, TF-FedBLS algorithm is always better than FedBLS. This indicates that TF-FedBLS algorithm can improve the average accuracy of the global model in server to a certain extent when some vehicle nodes have sample imbalance issues. At the same time, distributed computing can also improve the efficiency of our algorithm.

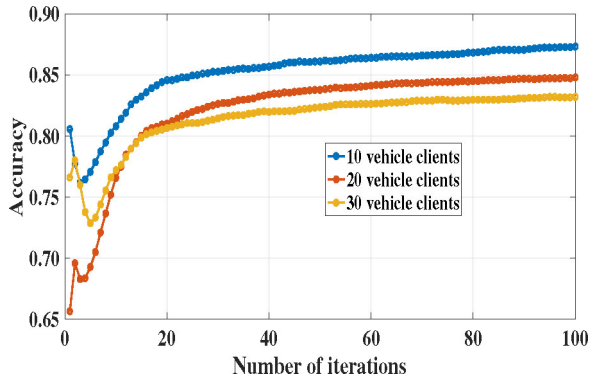


Figure 11. *The performance of TF-FedBLS with various numbers of data clients.*

In some cases, the data volume of the local client may be fewer, the model may be less than ideal after only one round of training, or the server may require the client to upload parameters which has a certain accuracy based on the needs of the IoV environment. Thus, we evaluate the performance of FedBLS to combine with deep learning. From the Figure 11, with the increasing of the number of training iterations, the prediction accuracy of TF-FedBLS algorithm first decreases and then increases, and finally improves significantly compared with the original prediction accuracy. However, with the increase in the number of IoV clients, although the average prediction accuracy of the BiBLS model is relatively low with out iterative training. After the certain number of iterations, it can be seen that the accuracy has been greatly improved.

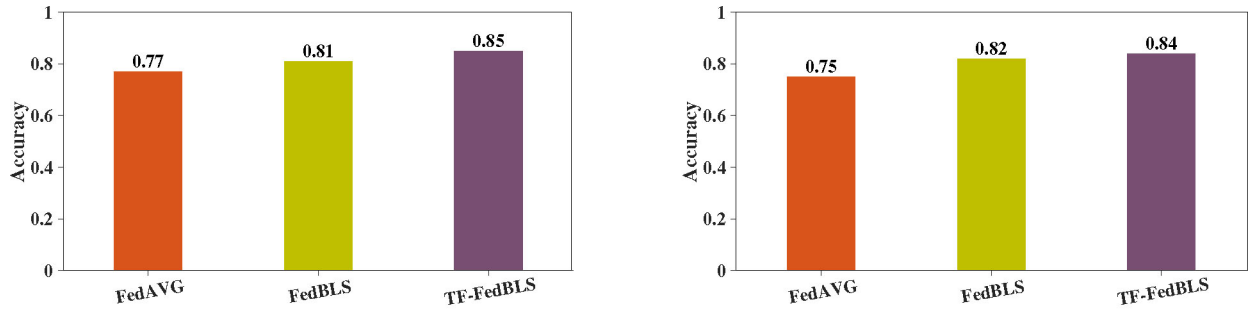
From the Figure 12-13, when some samples are fewer, the performance of FedBLS and TF-FedBLS of various vehicular nodes are better than the FedAVG. Compared with FedAVG algorithm, our TF-FedBLS algorithm has significant improvement in few-shot learning. Compared with FedBLS, TF-FedBLS has certain improvement in parameter aggregation effect. Especially when there are 10 nodes, the improvement effect is obvious. The main reason is that when some nodes have very little data to train the model, the effect trained by BiBLS model alone may be poor, because there is under fitting phenomenon, and sending the parameters to the server for aggregation may make the effect of the model lower on the server.

From the Figure 14-15, we evaluate the FedBLS algorithm based on clustering. The FedBLS is used to make comparison algorithm. The performance of FedBLS with clustering of various vehicular nodes are better than the FedBLS. If FedBLS is used only, some vehicle nodes do not share data with the server while moving, so the average accuracy results of BiBLS model lag far behind FedBLS with cluster. When some nodes are not disconnected from the server, the unsupervised clustering method is used to unload the data sharing to the cluster head and the nodes within the cluster for data sharing. Thus, FedBLS with cluster can use the transfer learning to enhance the effectiveness of data sharing in V2V. This approach in various of nodes can enhance the effectiveness of local BiBLS model.

Table 2. *The parameters size and efficiency of baseline models.*

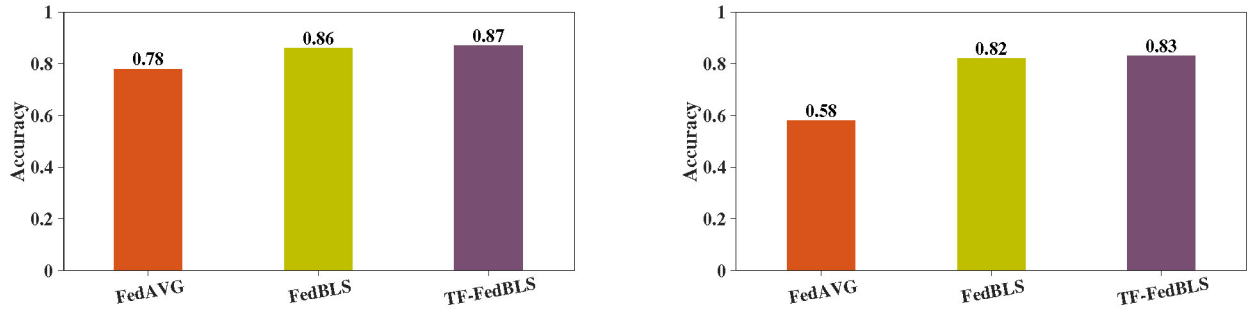
Models	Training Time	Parameters Size
FedAVG	264.82s	0.02M
FedBLS	**66.19s**	**0.005M**

We also compare the performance of FedBLS and FedAVG in training time, the size of model parameters and communication time. The local clients use CNN to train model based on FedAVG. From the Table 2, it takes a lot of time to train CNN 100 times in CPU. Due to the deep structure of CNN, there are many parameters, which makes the size of model large. Therefore, without considering the communication delay and packet loss rate in an ideal situation, it takes a lot of time to communicate. Since the IoV is a real-time and mobile environment, it takes lots of time to train the model, and it is likely that uploading parameters to the server are not real-time. The FedBLS algorithm only uploads W, so the size of the parameters is small and the communication time is negligible.



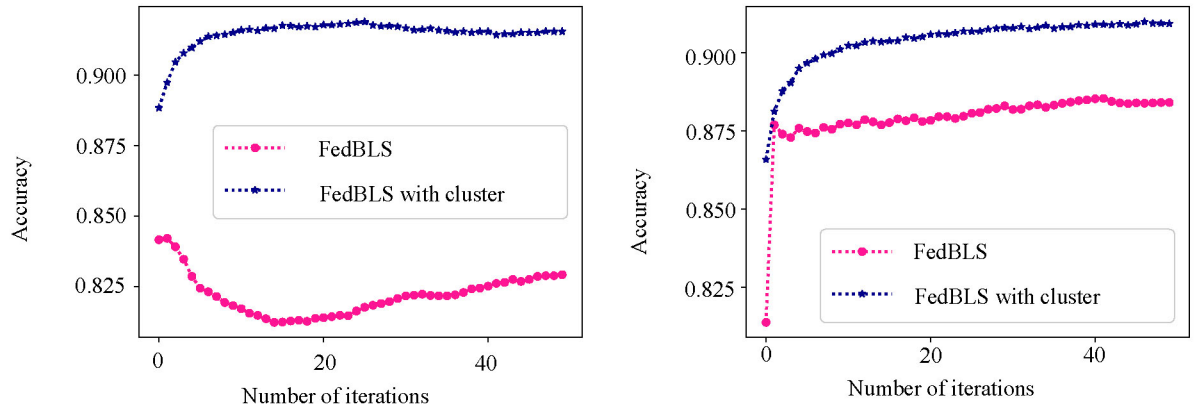
(a) *10 vehicular nodes.* **(b)** *15 vehicular nodes.*

Figure 12. *The global accuracy results of FedBLS, TF-FedBLS and FedAVG for 10 and 15 vehicular nodes.*



(a) *20 vehicular nodes.* **(b)** *25 vehicular nodes.*

Figure 13. *The global accuracy results of FedBLS, TF-FedBLS and FedAVG for 20 and 25 vehicular nodes.*

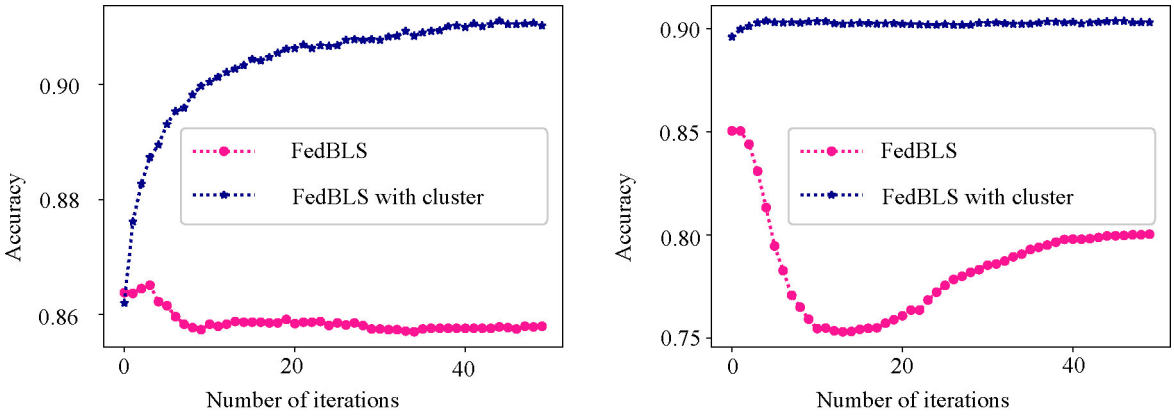


(a) *10 vehicular nodes.* **(b)** *15 vehicular nodes.*

Figure 14. *The global accuracy results of FedBLS and FedBLS with clustering for 10 and 15 vehicular nodes.*

V. CONCLUSION

In this paper, we have proposed the federated bi-directional connection broad learning scheme for data sharing in IoV. Vehicular nodes use BiBLS to train local data set in IoV. The server uses FedBLS algorithm to aggregate the parameters of vehicular nodes. Moreover, we use FedBLS algorithm based on transfer learning to solve unbalanced data problems. Compared with the benchmark algorithms, our simulation results show that our proposed scheme has improved the average accuracy of prediction on the server by



(a) *20 vehicular nodes.* **(b)** *25 vehicular nodes.*

Figure 15. *The global accuracy results of FedBLS and FedBLS with clustering for 20 and 30 vehicular nodes.*

about 5%, and the efficiency has improved by more than 3 times, while greatly reducing the size of parameters sharing between the server and vehicle nodes. In the future, we will continue to discuss the communication efficiency of our scheme at the network layer. At the same time, we will continue to study that our scheme can be dynamically adjusted according to the environment through deep reinforcement learning.

ACKNOWLEDGEMENT

This work was supported by the National Natural Science Foundation of China under Grant No.61901099, 61972076, 61973069 and 62061006, the Natural Science Foundation of Hebei Province under Grant No.F2020501037, and the Natural Science Foundation of Guangxi Province under Grant No.2018JJA170167.

References

[1] Y. Li, H. He, *et al.*, "Deep reinforcement learning-based energy management for a series hybrid electric vehicle enabled by history cumulative trip information," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, 2019, pp. 7416–7430.

[2] T. Wang, Z. Cao, *et al.*, "Privacy-enhanced data collection based on deep learning for internet of vehicles," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 10, 2020, pp. 6663–6672.

[3] G. Xu, H. Li, *et al.*, "Verifynet: Secure and verifiable federated learning," *IEEE Transactions on Information Forensics and Security*, vol. 15, 2019, pp. 911–926.

[4] K. Wei, J. Li, *et al.*, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, 2020, pp. 3454–3469.

[5] B. McMahan, E. Moore, *et al.*, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, 2017, pp. 1273–1282.

[6] Y. Lu, X. Huang, *et al.*, "Blockchain empowered asynchronous federated learning for secure data sharing in internet of vehicles," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, 2020, pp. 4298–4311.

[7] D. Ye, R. Yu, *et al.*, "Federated learning in vehicular edge computing: A selective model aggregation approach," *IEEE Access*, vol. 8, 2020, pp. 23 920–23 935.

[8] C. P. Chen and Z. Liu, "Broad learning system: An effective and efficient incremental learning system without the need for deep architecture," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 1, 2017, pp. 10–24.

[9] C. L. P. Chen, Z. Liu, *et al.*, "Universal approximation capability of broad learning system and its structural variations," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 4, 2019, pp. 1191–1204.

[10] J. Du, C. Vong, *et al.*, "Novel efficient rnn and lstm-like architectures: Recurrent and gated broad learning systems and their applications for text classification," *IEEE Transactions on Cybernetics*, 2020, pp. 1–12.

[11] F. Chu, T. Liang, *et al.*, "Weighted broad learning system and its application in nonlinear industrial process modeling," *IEEE Transactions on Neural Networks and Learning Systems*, 2019.

[12] L. Xu, X. Yu, *et al.*, "Physical layer security performance of mobile vehicular networks," *Mobile Networks and Applications*, vol. 25, no. 2, 2020, pp. 643–649.

[13] Z. Du, C. Wu, *et al.*, "Federated learning for vehicular internet of things: Recent advances and open issues," *IEEE Open Journal of the Computer Society*, vol. 1, 2020, pp. 45–61.

[14] L. Zhang, J. Li, *et al.*, "Analysis and variants of broad learning system," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. PP, no. 99, 2020, pp. 1–11.

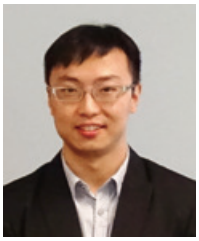
[15] M. Han, S. Feng, *et al.*, "Structured manifold broad learn-

ing system: A manifold perspective for large-scale chaotic time series analysis and prediction," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 9, 2018, pp. 1809–1821.
[16] Z. Liu, J. Zhou, *et al.*, "Broad learning system: Feature extraction based on k-means clustering algorithm," in *2017 4th International Conference on Information, Cybernetics and Computational Social Systems (ICCSS)*. IEEE, 2017, pp. 683–687.
[17] H. Wang, T. Liu, *et al.*, "Architectural design alternatives based on cloud/edge/fog computing for connected vehicles," *IEEE Communications Surveys Tutorials*, 2020, pp. 1–1.
[18] P. Habibi, M. Farhoudi, *et al.*, "Fog computing: A comprehensive architectural survey," *IEEE Access*, vol. 8, 2020, pp. 69 105–69 133.
[19] M. Rihan, M. Elwekeil, *et al.*, "Deep-vfog: When artificial intelligence meets fog computing in v2x," *IEEE Systems Journal*, 2020.
[20] Q. Yuan, H. Zhou, *et al.*, "Toward efficient content delivery for automated driving services: An edge computing solution," *IEEE Network*, vol. 32, no. 1, 2018, pp. 80–86.
[21] C. Wu, Z. Liu, *et al.*, "Collaborative learning of communication routes in edge-enabled multi-access vehicular environment," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 4, 2020, pp. 1155–1165.
[22] W. Zhan, C. Luo, *et al.*, "Deep reinforcement learning-based offloading scheduling for vehicular edge computing," *IEEE Internet of Things Journal*, 2020.
[23] Z. Xiong, Y. Zhang, *et al.*, "Deep reinforcement learning for mobile 5g and beyond: Fundamentals, applications, and challenges," *IEEE Vehicular Technology Magazine*, vol. 14, no. 2, 2019, pp. 44–52.
[24] A. Jindal, G. S. Aujla, *et al.*, "Sedative: Sdn-enabled deep learning architecture for network traffic control in vehicular cyber-physical systems," *IEEE Network*, vol. 32, no. 6, 2018, pp. 66–73.
[25] Y. Liu, H. Yu, *et al.*, "Deep reinforcement learning for offloading and resource allocation in vehicle edge computing and networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, 2019, pp. 11 158–11 168.
[26] Q. Yuan, J. Li, *et al.*, "Cross-domain resource orchestration for the edge-computing-enabled smart road," *IEEE Network*, vol. 34, no. 5, 2020, pp. 60–67.
[27] X. Jiang, J. Bi, *et al.*, "A survey on information-centric networking: Rationales, designs and debates," *China Communications*, vol. 12, no. 7, 2015, pp. 1–12.
[28] C. Fang, H. Yao, *et al.*, "A survey of mobile information-centric networking: Research issues and challenges," *IEEE Communications Surveys Tutorials*, vol. 20, no. 3, 2018, pp. 2353–2371.
[29] A. Gulati, G. S. Aujla, *et al.*, "Deep learning-based content centric data dissemination scheme for internet of vehicles," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–6.
[30] Y. Dai, D. Xu, *et al.*, "Deep reinforcement learning and permissioned blockchain for content caching in vehicular edge computing and networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, 2020, pp. 4312–4324.
[31] X. Zhang, M. Peng, *et al.*, "Deep reinforcement learning based mode selection and resource allocation for cellular v2x communications," *IEEE Internet of Things Journal*, vol. 7, no. 7, 2020, pp. 6380–6391.
[32] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, 2007, pp. 395–416.
[33] L. Lyu, J. Yu, *et al.*, "Towards fair and privacy-preserving federated deep models," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 11, 2020, pp. 2524–2541.
[34] J. Konečnỳ, H. B. McMahan, *et al.*, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.
[35] Y. Chen, X. Sun, *et al.*, "Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation," *IEEE Transactions on Neural Networks and Learning Systems*, 2019.
[36] H. Zhu and Y. Jin, "Multi-objective evolutionary federated learning," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 4, 2019, pp. 1310–1322.
[37] Y. Qi, M. S. Hossain, *et al.*, "Privacy-preserving blockchain-based federated learning for traffic flow prediction," *Future Generation Computer Systems*, vol. 117, 2020, pp. 328–337.
[38] H. Chai, S. Leng, *et al.*, "A hierarchical blockchain-enabled federated learning algorithm for knowledge sharing in internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, 2020, pp. 1–12.
[39] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, 1970, pp. 55–67.
[40] S. Gao, G. Guo, *et al.*, "An end-to-end broad learning system for event-based object classification," *IEEE Access*, vol. 8, 2020, pp. 45 974–45 984.
[41] M. Wang and W. Deng, "Deep visual domain adaptation: A survey," *Neurocomputing*, vol. 312, 2018, pp. 135–153.
[42] T. V. Phan, S. Sultana, *et al.*, "q-transfer: A novel framework for efficient deep transfer learning in networking," in *2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*. IEEE, 2020, pp. 146–151.
[43] S. Sarfraz, V. Sharma, *et al.*, "Efficient parameter-free clustering using first neighbor relations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8934–8943.
[44] M. Behrisch, D. Krajzewicz, *et al.*, "Simulation of urban mobility," in *2nd NEARCTIS workshop 2009*, 2014.

Biographies

Xiaoming Yuan received her B.E. degree in Electronics and Information Engineering from Henan Polytechnic University, China, in 2012. She received the Ph.D. degree in Communication and Information System from Xidian University, China, in 2018. She is now an assistant professor of Qinhuangdao Branch Campus, Northeastern University, China, from 2018. Her research interests include cloud/edge computing, medium access control and performance analysis for wireless body area networks and the Internet of Things.

Jiahui Chen received B.E. degree in Internet of Things Engineering from Lanzhou University of Technology, Lanzhou, China. He is currently pursuing master's degree of Computer Science and Technology since September 2020 at Qinhuangdao Branch Campus, Northeastern University, Shenyang, China. His research interests include cloud/edge computing and Internet of Vehicles.



Ning Zhang is an Associate Professor at University of Windsor, Canada. He received the Ph.D degree from University of Waterloo, Canada, in 2015. After that, he was a postdoctoral research fellow at University of Waterloo and University of Toronto, Canada, respectively. His current research interests include next generation mobile networks, physical layer security, machine learning, and mobile edge computing.

Xiaojie Fang received his B.Sc., M.Sc. and Ph.D. degrees from Department of Electronics and Information Engineering, Harbin Institute of Technology in 2010, 2012, and 2018 respectively. During 2015-2016, he was a visiting scholar in the Broadband Communications Research Group (BBCR), University of Waterloo, Canada. He is currently a postdoc research fellow and an assistant professor at the Department of Electronics and Information Technology, Harbin Institute of Technology. His research interests include physical layer security, coding and modulation theory.

Didi Liu received her MS degree in communication and information system from Guilin University of Electronic Technology, china, in 2006 and her Ph.D. degree in communication and information system from Xidian University, China, in 2018. Since 2014, she has been an associate professor in Guangxi Normal University. Her research interests include stochastic network optimization and smart grid.