Received 18 July 2022, accepted 26 July 2022, date of publication 29 July 2022, date of current version 26 August 2022.

*Digital Object Identifier 10.1109/ACCESS.2022.3195039*

RESEARCH ARTICLE

# Heterogeneous Defect Prediction Based on Federated Reinforcement Learning via Gradient Clustering

**AILI WANG[1], (Member, IEEE), YINGHUI ZHAO[1], GUODONG LI[1], JUN ZHANG[2], HAIBIN WU[1], AND YUJI IWAHORI[3]**

[1]Heilongjiang Province Key Laboratory of Laser Spectroscopy Technology and Application, Harbin University of Science and Technology, Harbin 150080, China
[2]China Energy Taishan Power, Taishan 529200, China
[3]Department of Computer Science, Chubu University, Kasugai, Aichi 487-8501, Japan

Corresponding author: Guodong Li (ligd7126@hrbust.edu.cn)

This work was supported by the Reserved Leaders of Heilongjiang Provincial Leading Talent Echelon, in 2021.

**ABSTRACT** Heterogeneous defect prediction (HDP) refers to using heterogeneous data collected by other projects to build a defect prediction model to predict the software defects in a project. Traditional methods usually involve the measurement of the source project and the target project. However, due to the limitations of laws and regulations, these original data are not easy to obtain, which forms a data island. As a new machine learning paradigm, federated learning (FL) has great advantages in training heterogeneous data and data island. In order to solve the data island and data heterogeneity of HDP, we propose a novel Federated Reinforcement Learning via Gradient Clustering (FRLGC) method in this paper. Firstly, the parameters of the global model are transferred to each dueling deep Q network (dueling DQN) model and each client uses private data to train the dueling model which combines experience replay to increase data efficiency in limited datasets. Secondly, gaussian differential privacy is used to encrypt the model parameters to ensure the privacy and security of the model. Finally, we cluster the clients according to their locally encrypted model parameters and use weighted average to aggregate to create a new global model locally and globally. Experiments on nine projects in three public databases (Relink, NASA and AEEEM) show that FRLGC is superior to the relevant HDP solutions.

**INDEX TERMS** Data island, federated reinforcement learning, experience replay, Gaussian differential privacy.

## I. INTRODUCTION

With the continued growth of the functionalities and requirements of the software products, their size and complexity are also increasing, developing good quality software is an expensive task. Software defect detection (SDP) is useful for software engineer to pay attention to defective modules and minimize the occurrence of software defect to minimize the software development cost.

Most SDP methods use the collected historical from a project as training data to build prediction models and then predict defects of new software modules in the same project through prediction models. These SDP methods are called as within-project defect prediction (WPDP) [1], [2], [3]. However, it is difficult to predict new projects for WPDP. Cross-project defect prediction (CPDP) can solve the lack of historical defect datasets, which establishes a learning model on defect datasets of other projects [4], [5], [6].

Unfortunately, the prediction effect of most CPDP is not satisfactory [7], because most previous CPDP methods were based on the assumption that the same feature sets are shared across projects, which is not realistic and it's challenging to collect the same feature sets when projects are developed in different programming languages. When the metrics of source and target projects are totally different, it becomes a heterogeneous problem and conventional

The associate editor coordinating the review of this manuscript and approving it for publication was Chengpeng Hao.

CPDP approaches cannot be applied to heterogeneous scenario directly. In recent years, many heterogeneous defect prediction (HDP) models have been proposed. Gong *et al.* proposed conditional domain adversarial adaptation (CDAA) for HDP which takes advantage of label information to effectively map source project to target project and improves the predictive performance [8]. In order to realize the improvement of generalization ability for the defect prediction model, Shen *et al.* constructed a semi-supervised software defect prediction method based on sampling and integration for the problem of class imbalance in software defect data and the incomplete classification of data sets [9]. Wu *et al.* set up an intermediate domain from the target domain to the source domain to break the distribution gap and narrow the difference between the source domain and the target domain [10]. Jin. proposed to implement domain adaptation (DA) by using the kernel twin support vector machines (KTSVMs), which can match the distributions of training data for different projects [11]. Because of the redundancy and nonlinearity characteristics of the source and target data, Li *et al.* proposed a landmark selection-based kernelized discriminant subspace alignment (LSKDSA) approach, which can reduce the discrepancy of the data distributions between the target and source projects [12]. Tong *et al.* proposed to combine kernel spectral embedding, transfer learning, and ensemble learning to find the potential common feature of source datasets and target datasets [13]. Most HDP models realize defect prediction through domain adaptation or searching the latest common feature space between source datasets and target datasets. However, all of the above approaches involve the measurement of the source project and the target project, which will expose the privacy of original data and is illegal. In fact, most defective datasets are scattered in different organizations and are not easy to obtain due to the limitations of many laws and regulations, which forms a data island and it will be challenging in the future.

Google first proposed the concept of federated learning (FL) in 2016, which is a powerful framework that can solve data island problem while protecting users' privacy. FL is a promising collaboration paradigm which enables all clients to train their models with local datasets and jointly build a global model by transferring models' parameters to the server. The whole process will not expose data privacy, and clients have complete autonomy over local data. Wang *et al.* proposed Federated Transfer Learning via Knowledge Distillation (FTLKD) approach which improves the performance of heterogeneous defect prediction models compared to the popular methods nowadays [14]. In order to protect data privacy, Bai and Fan. proposed a homomorphic encryption-based privacy enhancement mechanism which has effect on membership inference attacks [15]. Poor network connection and limited computing resources make it very slow or infeasible to train a deep neural network (DNN) according to the FL pattern. Xu *et al.* proposed weight quantization, structured pruning and selective updating to accelerate the FL training process [16]. Su *et al.* used FL to preserve energy data and enable energy data owners (EDOs) to cooperatively train a shared AI model without revealing the local energy data [17]. Ye *et al.* proposed edge federated learning (EdgeFed), which can train a deep learning model from decentralized data on modern mobile devices [18]. However, building a global model of high-quality is challenging when the feature space is small and the training data is limited of each client for FL.

Adam *et al.* thought a promising approach is experience replay (ER) in reinforcement learning (RL). Experience replay is that obtained data during the learning process are stored and repeatedly presented to the underlying RL algorithm, which can increase data efficiency in limited datasets [19]. Federated reinforcement learning (FRL) is that each client builds their private models by using the RL algorithm in the FL framework. The goal of FRL is to improve efficiency of training or policy quality by interacting with information with privacy protection. FRL can solve the data island and improve data efficiency simultaneously. Zhuo *et al.* proposed a novel deep reinforcement learning framework, which builds a DQN for each client to build high-quality policies when the training data is limited and the feature space of states is small [21]. Lee and Choi proposed a novel FRL approach, in which convergence of the agents' optimal policies is faster [22]. Hu *et al.* proposed a general FRL framework, which takes reward shaping as the information shared between different clients to improve the training speed and policy quality of each client [23]. When all edge computing devices jointly build a global model, the heterogeneous private datasets of a large number of devices will cause deterioration of model quality in the training process. EK *et al.* proposed a novel aggregation algorithm, called FedDist (Federated Distance), which can identify the differences between specific neurons and modify their model structure [24]. Pang *et al.* proposed an intelligent central server which has the ability to identify heterogeneity and helps guide most clients to achieve better performance [25]. In order to address the local clients' data distributions diverge, Sattler *et al.* group the client population into clusters which have jointly trainable data distribution by using the geometric properties of the FL loss surface [26]. Huang *et al.* proposed a federated learning framework (FedDSR) and a similarity aggregation algorithm to improve the quality of the model [27].

In this paper, we propose a federated reinforcement frame via gradient clustering, called FRLGC. The basic idea behind FRLGC is to address the problem of HDP by utilizing the deep reinforcement learning model in a federated learning framework and using similarity knowledge of client gradients to improve the quality of the model. The contributions of this paper can be summarized as follows:

1. We suggest federated reinforcement learning for HDP. Federated reinforcement learning allows multiple clients to train their private models and build a global model without exposing their private training data to solve the problem of data island. In the case of limited datasets, experience replay is used to improve data efficiency.

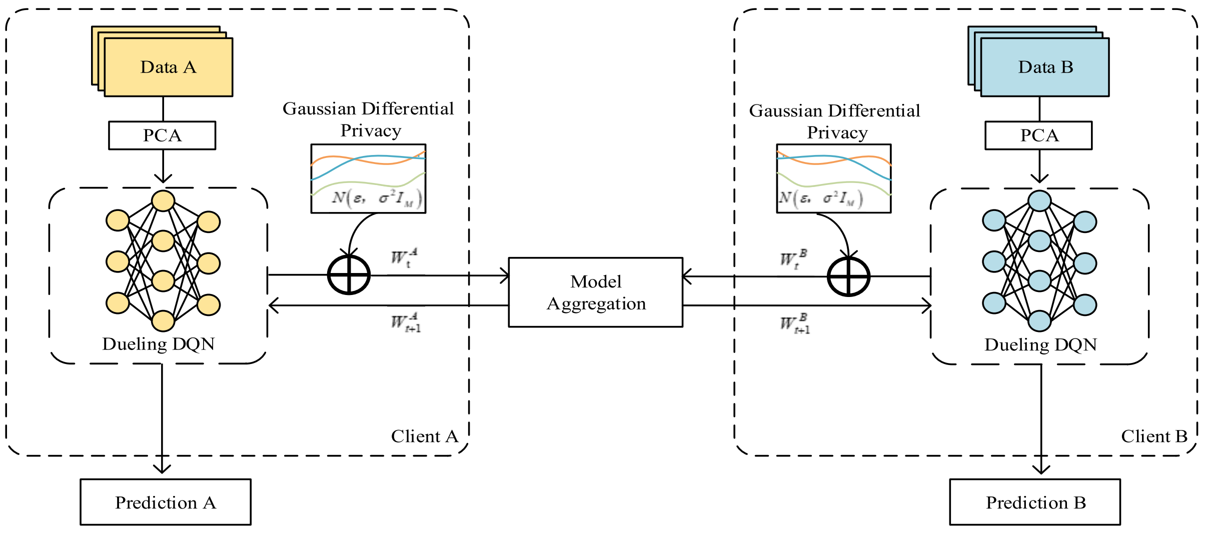


**FIGURE 1.** The overall framework of FRLGC with two clients.

2. We separate the original output into the value branch and the advantage branch and then combine the two branches of fully connected layers into one output, which leads to better policy evaluation in the presence of many similar-valued predictions through making the last module of the network implement the forward mapping.

3. Similarity knowledge of client to guide FL aggregation. Clients with similar gradients are aggregation locally by using average weights and followed by the global aggregation to ensure better coverage while reducing differences.

4. We prove the correctness of FRLGC and evaluate its performance through experimental results, which can significantly improve prediction performance.

The remainder of this paper is organized as follows. Section II introduces the overall framework of FRLGC and shows the details of steps in FRLGC. The experimental setup, including experimental data, evaluation measure and analysis of the experiments results are shown in Section III. Section IV finally concludes the paper.

## II. METHOD

Fig. 1 shows the overall method framework of FRLGC proposed in this paper. The overall approach is carried out under the framework of federal learning. In order to eliminate data redundancy and align the data of all clients, we first use principal components analysis (PCA) to reduce the dimension of data in the data preprocessing. Then training the data of all clients locally by dueling DQN. After the training, the model parameters of each local model will be encrypted through gaussian differential privacy. At the process of model aggregation, we only select some of the clients to participate in the federation aggregation process restricted by poor network connection and limited computing resources. The selected clients are clustered through K-means. Each cluster will conduct local aggregation first, and then conduct global aggregation on the central server to form a global model. Finally, the global model will be broadcast to each client. Each client mainly has four stages: data preprocessing, local training, data encryption and model aggregation. When the private model of each client converges or reacheing the maximum communication round, the update stops.

### A. DATA PREPROCESSING

The number of data sample features owned by each client is different, and the data features may have redundancy. Therefore, in the data preprocessing stage, we use PCA to reduce the dimension of the data.

PCA uses the variance of projection data to represent the information size of original data. So the purpose of PCA is to get the larger the variance of projection data which is to find an adaptation matrix $\boldsymbol{A} \in R^{d\times k}$ to maximize the following problems:

$$\max_{\boldsymbol{A}^T\boldsymbol{A}=\boldsymbol{I}} \operatorname{tr}\left(\boldsymbol{A}^T\boldsymbol{XHX}^T\boldsymbol{A}\right) \tag{1}$$

$R^{d\times k}$ represents the real number space of $d \times k, k$ is a parameter less than $d$. $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_a\} \in R^{d\times a}$ is matrix of input dataset, $\boldsymbol{x}_i$ represents the $i$-th sample of dataset, $1 \leq i \leq a$, $R^{d\times a}$ represents a real space, $a$ is the total amount of test data and training data, $d$ is the dimension of each sample, $\boldsymbol{H} = \boldsymbol{I} - \frac{\boldsymbol{Q}}{a}\boldsymbol{Q}^T$ represents the central matrix and $\boldsymbol{Q}$ represents a full 1 matrix of size $a \times a$, $\boldsymbol{I}$ is the identity matrix whose size is $a \times a$.

$\frac{1}{a}\boldsymbol{XHX}^T$ is another representation of the covariance matrix and $\boldsymbol{XHX}^T\boldsymbol{A} = \boldsymbol{A\Phi}$, $\boldsymbol{\Phi} = \operatorname{diag}(\phi_1, \ldots, \phi_k) \in R^{k\times k}$, $R^{k\times k}$ is a real number space, and $\phi_1, \ldots, \phi_k$ is the first $k$ largest eigenvalues, $\boldsymbol{\Phi}$ is the matrix constructed by $\phi_1, \ldots, \phi_k$ as the diagonal element, and other elements except the diagonal are 0, then the optimal low dimensional feature representation is $\boldsymbol{Z} : \boldsymbol{Z} = [z_1, \ldots, z_a] = \boldsymbol{A}^T\boldsymbol{X}$.

## B. DUELING DQN

In FRL, building a policy of high-quality is challenging when the feature space of states is small and the training data of each client is limited. The FRL aims to promote training efficiency or improve policy quality through information interaction with privacy protection.

Mnih *et al.* [20] proposed Deep Q-Network (DQN), which can train AI agents to play an Atari game better than human players. The DQN trains a Q-network by estimating the value function of the actions for a given state. Even though DQN improves the accuracy and efficiency of existing Q-network in a great scale, it suffers from an overestimation problem that causes inaccurate results. Wang *et al.* [29] improved DQN by presenting dueling DQN, which trained the Q-network faster than the DQN.

In local training stage of FRL, each client trains local data by dueling DQN of reinforcement learning technique for private model construction. Dueling DQN repeatedly presented the stored data in the training process to the policy network of dueling DQN, which can improve the utilization efficiency of limited datasets. And at the output layer of dueling DQN, which combines the value branch and the advantage branch into one output, which can avoid estimating the redundant and low-valued predictions and lead to a better policy evaluation through making the last module of the network implement the forward mapping in the case of that existing presence of many similar-valued predictions.

Suppose there are $N$ available devices for a federated learning job. During each round, each device uses dueling DQN to build a private model.

Dueling DQN usually consists of a dynamic environment and an agent that interacts with the environment. It is a process in which the agent interacts with the environment and keeps learning from the environment to maximize its expectation of the rewards.

In HDP domain, we define states, actions and rewards as follows.

**States**: Let the state be represented by $S_t = \left(s_t^{(1)}, s_t^{(2)}, \ldots, s_t^{(N)}\right)$, where $s_t^{(1)}, \ldots, s_t^{(N)}$ are data characteristics of the N agents, respectively.

**Actions**: There are two actions for each agent: {select, neglect}, selecting a state at this time indicates that it is a defective sample and neglecting indicates the state is flawless. Select and neglect correspond to 1 and 0 respectively. The state is the data characteristic.

**Rewards**: Let the instant reward be $r_t$, where $r_t$ determined by data set defect rate $d_i, d_i \subseteq [0, 1]$ and as defined as follows:

$$r_a = \begin{cases} +1, & \text{if the action is correct} \\ -1, & \text{if the action is wrong} \end{cases} \tag{2}$$

$$r_t = r_a d_i \tag{3}$$

The goal of each agent is to maximize the expectation of the cumulative discounted reward as far as possible during the training process and the expectation is given by $R_t = \sum_{t=1}^{T} \gamma^{t-1} r_t$, where $\gamma \subseteq (0, 1]$ is a factor discounting future rewards.

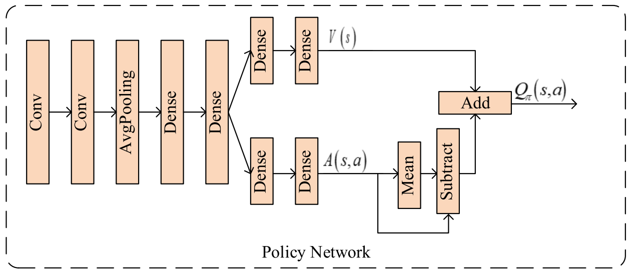


**FIGURE 2.** Structure of neural network of policy network.

For an agent behaving according to a stochastic policy $\pi$, the $Q$ function of the state-action pair $(s, a)$ and the state-value $V^{\pi}$ (s) are defined as follows:

$$Q^{\pi}(s, a) = E\left[R_t \mid s_t = s, a_t = a, \pi\right] \tag{4}$$

$$V^{\pi}(s) = E_{a \sim \pi(s)}\left[Q^{\pi}(s, a)\right] \tag{5}$$

We define the advantage function, which relates to the value and $Q$ function:

$$A^{\pi}(s, a) = Q^{\pi}(s, a) - V^{\pi}(s) \tag{6}$$

From the (6) and state-value $V^{\pi}(s)$, it follows:

$$E_{a \sim \pi(s)}\left[A^{\pi}(s, a)\right] = 0 \tag{7}$$

$$Q^{\pi}(s, a) = A^{\pi}(s, a) + V^{\pi}(s) \tag{8}$$

Equation (8) is unidentifiable in the sense that given $Q$ we cannot recover $V$ and $A$ uniquely. This lack of identifiability is mirrored by poor practical performance when this equation is used directly.

To address this issue of identifiability, we can force the advantage function estimator to have zero advantage at the chosen action. That is, we let the last module of the network implement the forward mapping.

$$Q(s, a) = V(s) + \left[A(s, a) - \frac{1}{|A|} \sum_{a'} A\left(s, a'\right)\right] \tag{9}$$

Fig. 2 shows the structure of the neural network of the policy network. The lower layers of the dueling DQN are two convolution layers, one average pool layer and two fully-connected layers. However, we use two sequences of fully connected layers instead of using a fully connected layer sequence as the output which can lead to a better policy evaluation through making the last module of the network implement the forward mapping in the case of that existing presence of many similar-valued predictions. The construction of these streams enables them to provide a separating estimation of the value and advantage functions. Finally, the two sequences are combined into an output function. The output of the network is a set of Q-value.

Fig. 3 shows the overall framework of dueling DQN.

Data, data labels and reward functions constitute a dynamic environment. The policy network uses two neural networks

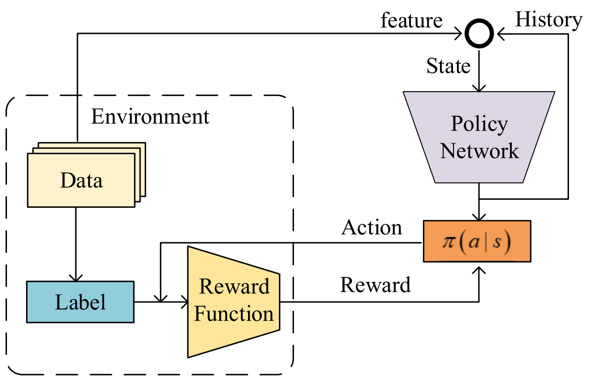


**FIGURE 3.** **Overall framework of dueling DQN.**

with the same structure (a primary network and a target network) to predict. When the data features are input into the policy network, the primary network outputs the action $a_t = \arg\max Q(s, a)$. The history includes the action, the reward and the data features. When the history is the input policy network, the primary network is for selecting an action and the target network is for generating a Q-value for the action. During training, dueling DQN employs backpropagation. For the backpropagation, we computed the error between the estimated value and the target value. The target value, which is recorded during training process, which is related to parameter $s$, $a$ and $r$, and the estimated value comes from the primary network. $Q(s, a)$ is the result of the $Q$ function, which is the estimation of the final reward. Because $Q(s, a)$ is the estimation of the final state, we estimate the reward from the current reward $r_t$ by adding the maximum reward $\gamma \max_{a'} Q(s', a')$. Therefore, the loss function is written as:

$$\text{loss} = \left[r_t + \gamma \max_{a'} Q\left(s', a'\right) - Q(s, a)\right]^2 \tag{10}$$

Dueling DQN algorithm separates the target network and the primary network for loss function to remove the correlations between the $Q$ and the target value.

The weight of the target network is fixed, and only the weight of the primary network will be updated regularly.

### C. GAUSSIAN DIFFERENTIAL PRIVACY

FRL achieves privacy-preserving by keeping the local datasets on user devices and only sharing the local updates with the server. However, it has been proven to be insufficient for maintaining data privacy as the parameters can reveal insights into the datasets that have been used for training. Consequently, FRL can only be incorporated with honest participating clients and extended for secure and privacy-preserving settings, extra measures should be considered.

In the data encryption stage, we use the gaussian differential privacy method [28]. The idea of gaussian differential privacy is that when the adversary tries to query individual information from the database, which will confuse. The adversary cannot distinguish the sensitivity of individual level from the query results.

Gaussian differential privacy has its unique advantages. It has no cumbersome encryption and decryption process, which can save computing resources and has higher data processing efficiency compared to the traditional cryptography method. For the FRL system, gaussian difference privacy can confuse the model parameter information by adding gaussian noise to the model parameter information, so as to avoid other curious clients or central server inferring the exchanged information about training data or the model updating parameters by repeatedly receiving the model parameters in the joint training process. In addition, gaussian difference privacy can provide a clearer boundary of privacy loss by adding the average and variance of noise.

For two datasets with only one different data sample $D$ and $D'$, and the functions of arbitrary fields $f : D \to R^M$ and a randomized mechanism $\mathrm{M} : R^M \to O$, $\mathrm{M} \circ f$ achieves gaussian differential privacy for any output subset $S \subseteq O$:

$$\Pr[\mathrm{M}(f(D)) \subseteq S] \le e^{\varepsilon} \cdot \Pr\left[\mathrm{M}\left(f\left(D'\right)\right) \subseteq S\right] + \delta \tag{11}$$

where the smaller $\varepsilon$ represents stronger level of privacy protection, and $\delta \subseteq [0, 1]$ represents the probability of breaking the gaussian differential privacy. Properly add gaussian random noise to obfuscate $f(\cdot)$.

$$\mathrm{M}(f(D)) = f(D) + z, z \subseteq N\left(\varepsilon, \sigma^2 I_M\right) \tag{12}$$

where $I_M$ is the identity matrix, and $N\left(\varepsilon, \sigma^2 I_M\right)$ is the multivariate gaussian noise with mean $\varepsilon$ and variance $\sigma^2$, the required noise variance is:

$$\sigma^2 = 2(\Delta f)^2 \ln(1.25/\delta)/\varepsilon^2 \tag{13}$$

where $\Delta f$ is global sensitivity of function $f$, and $\|\cdot\|$ denotes the 2-norm.

$$\Delta f \triangleq \max_{D,D'} \left\|f(D) - f\left(D'\right)\right\| \tag{14}$$

### D. MODEL AG GREGATI ON

The models locally trained on heterogeneous data can be significantly different from each other. Aggregating these divergent models can slow down convergence and substantially reduce the model performance. Effective aggregation of models is essential to create a common global model for all clients. By analyzing inter clients' relations, we can determine the extent to which one client is universal and contribute to this aggregation. We use similarity knowledge of clients to guide model aggregation in model aggregation stage. Model aggregation is divided into two steps, which are local aggregation and global aggregation. Clients with similar gradients are clustered together, local aggregation models are weighted average within the cluster for local aggregation. And then using global aggregation to ensure better coverage while reducing differences.

Fig. 4 shows the structure of model aggregation. Firstly, clustering randomly selected clients according to their local gradients without sharing privacy by using K-means. Secondly, selected clients update their model weights. Then,

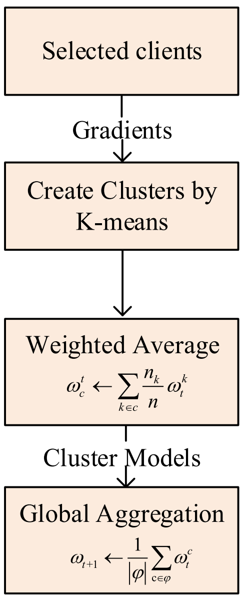

Selected clients
Gradients
Create Clusters by K-means
Weighted Average
$\omega_c^t \leftarrow \sum_{k\in c} \frac{n_k}{n} \omega_t^k$
Cluster Models
Global Aggregation
$\omega_{t+1} \leftarrow \frac{1}{|\varphi|} \sum_{c\in\varphi} \omega_t^c$

**FIGURE 4.** **Structure of model aggregation.**

in each cluster, local aggregation is to form a specialized model by using weighted average. For a given cluster $c$, at round $t$, the weighted average is formed as:

$$\omega_c^t \leftarrow \sum_{k\in c} \frac{n_k}{n} \omega_t^k \tag{15}$$

where $n_k$ is the size of selected clients, and $n$ is the total number of clients.

Finally, combining specialized models to build a new global model. At round $t+1$, the new global model is formed as:

$$\omega_{t+1} \leftarrow \frac{1}{|\varphi|} \sum_{\mathrm{c}\in\varphi} \omega_t^c \tag{16}$$

where $\varphi$ is the federated cluster space.

In a heterogeneous environment, increasing the coverage of clients will contribute to the universality of the global model, while gradients' similarity can help to identify and reduce the potentially harmful impact of different clients on global aggregation.

We propose FRLGC as Algorithm 1. More specifically, each epoch includes five procedures: (1) Line 2 is broadcasting to all agents; (2) Lines 3-11 are clustering clients with similar gradients for local aggregation; (3) Lines 12-13 are the global aggregation to ensure better coverage while reducing differences; (4) Lines 14-17 are the acting procedure by $\varepsilon-$exploration, where agents of clients choose whether to act randomly or follow the actor net strategies; (5) Lines 18-22 are the experience replay procedure of the networks which can increase data efficiency in case of limited datasets.

## III. EXPERIMENT RESULTS AND ANALYSIS

### A. EXPERIMENT DATASET DESCRIPTION

In this part, we evaluate the performance of the FRLGC algorithm based on three imbalanced datasets: NASA [30], AEEEM [31] and Relink [32], which are in the field of software defect prediction. Table 1 lists the details of the databases used in the experiments.

**Algorithm 1** FRLGC Algorithm

```
Input: $\omega_0$ initial global model, agents, n_clusters, max epoch MAX_EPOCH, datasets for each agent $s$
Output: prediction labels for each agent: L1~L6
Method
1.  for epoch t = 0 to MAX_EPOCH do
2.      Broadcast $\omega_t$ to all agents
3.      Select $\varphi$ agents where $\varphi \subseteq$ agents
4.      $\phi \leftarrow$ ClusterGradients $(\omega, \varphi)$
5.      for $c \subseteq \phi$ do
6.          for $k \subseteq c$ do
7.              $\omega_t^k \leftarrow$ update $\omega_t$ using SGD
8.          end for
9.          $\omega_c^t \leftarrow$ ClusterAggregation $\left(\omega_t^1, \omega_t^2, \ldots, \omega_t^{|c|}\right)$
10.         $\omega_t^c = \sum_{k\in c} \frac{n_k}{n} \omega_t^k$
11.     end for
12.     $\omega_{t+1} \leftarrow$ GlobalAggregation $\left(\omega_t^1, \omega_t^2, \ldots, \omega_t^{|\phi|}\right)$
13.     $\omega_{t+1} \leftarrow \frac{1}{|\phi|} \sum_{c\in\phi} \omega_t^c$
14.     for each agent $k \in s$ do
15.         $a_k = \varepsilon -$ explorationProcedure $(s_k)$
16.         Interact with environment and obtain reward $r(k)$
17.         add $\{s(k), a(k), r(k)\}$ into $B$;
18.         if $B[k] \geq B$ then
19.             ExperienceReplay($B[k]$)
20.         end if
21.     end for
22. end for
```

In addition, we note that when developers perform corrective maintenance work, they usually consider the severity of defects and the importance of modules. However, in three datasets, there is no information about defect severity or module importance. Moreover, in the current SDP research, there is no clear explanation about the incorporation between the defect severity and the importance of software modules in the evaluation process. Therefore, in this paper, we do not consider the severity of defects or the importance of modules.

To evaluate the prediction performance, the area under the receiver operating characteristic curve (AUC) and G-mean are used in this article. AUC is unaffected by class imbalance as well as being independent from the cutoff probability (prediction threshold) that is used to decide whether an instance should be classified as positive or negative. The value range of AUC is 0~1 and the higher AUC represents a better prediction performance. G-mean can also be used to evaluate the model performance of imbalanced dataset, which comprehensively considers both minority and majority classes. When the classification accuracy of the minority class and the majority class is closer, we can get the best G-mean.

**TABLE 1.** Details description of the datasets.

| Database | Project | Description of Project | Metrics | Instance | Defects | Defective (%) |
|---|---|---|---|---|---|---|
| | PC3 | Flight Software of Each Orbiting Satellite | 37 | 1077 | 134 | 12.44 |
| NASA | PC4 | Flight Software of Each Orbiting Satellite | 37 | 1458 | 178 | 12.21 |
| | MW1 | A Zero Gravity Experiment on Combustion | 37 | 253 | 27 | 10.67 |
| | EQ | OSGi Framework | 61 | 324 | 129 | 39.81 |
| AEEEM | JDT | IDE Development | 61 | 997 | 206 | 20.66 |
| | LC | Text Search Engine Library | 61 | 691 | 64 | 9.26 |
| | Apache | Open Source software system | 26 | 194 | 98 | 50.52 |
| Relink | Safe | Open Intents Library | 26 | 56 | 22 | 39.29 |
| | Zxing | Barcode Image-processing Library | 26 | 399 | 118 | 29.57 |

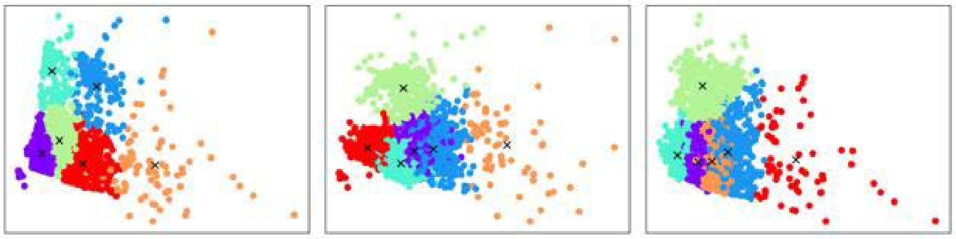


**FIGURE 5.** Clustering of clients. (a) AEEEM and Relink, (b) NASA and Relink, (c) NASA and AEEEM.

The experiment was programmed with Pycharm. We run all experiments on NVIDIA GTX 1070Ti. Two projects belong to NASA, AEEEM and Relink are selected in turn as the data set of each client in three groups of experiments.

## B. ANALYSIS OF CLUSTERING OF CLIENTS

We perform a cluster analysis by using datasets of all clients to explore the impact of the relationship of clients on final prediction performance.

The evaluation criterion of good cluster is having both high intra-cluster similarity and low inter-cluster distance.

For the intra-cluster similarity, the dataset $z$ is clustered by K-means algorithm, and the centroid of each cluster $C_i$ is expressed as $c_i$. The intra-cluster similarity of $z$ in the $k$ clusters is written as:

$$d_{\text{intra-cluster}} = \frac{1}{n}\sum_{i=1}^{k}\sum_{z\in C_i}\|c_i - z\|^2 \tag{17}$$

The inter-cluster distance represents the distance between the centroids $c_i$ and $c_j$ of the two clusters. So, the inter-cluster distance of $k$ clusters is expressed by the sum of all the cluster-cluster distances as

$$d_{\text{inter-cluster}} = \sum_{i=1}^{k-1}\sum_{j=i+1}^{k}\|c_i - c_j\|^2 \tag{18}$$

Fig.5 presents a two-dimensional mapping of the clustering in which different clusters of all clients are displayed in different colors. It is apparent that AEEEM and Relink have higher intra-cluster similarity (density) compared to the other datasets, which possibly explains the significant yet greater performance. All datasets lack a great inter-cluster distance (separability). Here the analysis of clustering shows that we can use the similarities during the learning process of clients, and can capture similarity by comparing their gradients to realize the aggregation process based on gradient similarity.

## C. ANALYSIS OF NUMBER OF CLIENTS

In order to test the impact of randomly selected participants on the FRL system, AEEEM and Relink are selected as the prediction results of the data set for analysis. TABLE 2 summarizes the relationship between average AUC and G-mean and time and number clients. 2, 3, 4, 5 and 6 clients are selected to participate in each round of communication process of FRL. When the number of clients is 4, the average values of AUC and G-mean can reach 0.6571 and 0.5519 respectively, and the time is 370.193s. When the number of clients is 5 and 6, the average values of AUC and G-mean are equivalent to situation of 4 clients, but the time is increased by 16.752s and 129.407s respectively. To consider AUC, G-mean and time together, we choose 4 clients for the experiments.

## D. ANALYSIS OF PREDICTION RESULTS OF CLIENTS IN COMMUNICATION ROUNDS

Fig. 6, Fig. 7, Fig. 8 show the AEEEM and Relink, NASA and AEEEM, NASA and Relink as the datasets of each client, and the relationship between AUC and G-mean and communication rounds, respectively. Communication rounds refers to the number of times the server and each client

**TABLE 2.** Result of the number of clients when AEEEM and Relink as datasets.

| Project \ number | Index | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| EQ | AUC | 0.5417 | 0.5892 | 0.6680 | 0.6703 | 0.6541 |
| | G-mean | 0.4907 | 0.5131 | 0.5497 | 0.5488 | 0.5482 |
| JDT | AUC | 0.5163 | 0.5932 | 0.7181 | 0.6954 | 0.7051 |
| | G-mean | 0.4805 | 0.5157 | 0.5693 | 0.5608 | 0.5642 |
| LC | AUC | 0.5428 | 0.6483 | 0.6935 | 0.6855 | 0.6962 |
| | G-mean | 0.4917 | 0.5383 | 0.5608 | 0.5584 | 0.5616 |
| Apache | AUC | 0.5812 | 0.6012 | 0.6071 | 0.6518 | 0.6371 |
| | G-mean | 0.5174 | 0.5254 | 0.5432 | 0.5550 | 0.5532 |
| Safe | AUC | 0.7022 | 0.7045 | 0.6524 | 0.6424 | 0.7424 |
| | G-mean | 0.5774 | 0.5641 | 0.5603 | 0.5803 | 0.5803 |
| Zxing | AUC | 0.5794 | 0.5754 | 0.6032 | 0.6310 | 0.6212 |
| | G-mean | 0.5106 | 0.5118 | 0.5283 | 0.5389 | 0.5462 |
| Average | AUC | 0.5772 | 0.6186 | 0.6571 | 0.6627 | 0.6760 |
| | G-mean | 0.5114 | 0.5281 | 0.5519 | 0.5570 | 0.5590 |
| | Time(s) | 185.503 | 265.252 | 370.193 | 386.945 | 499.600 |

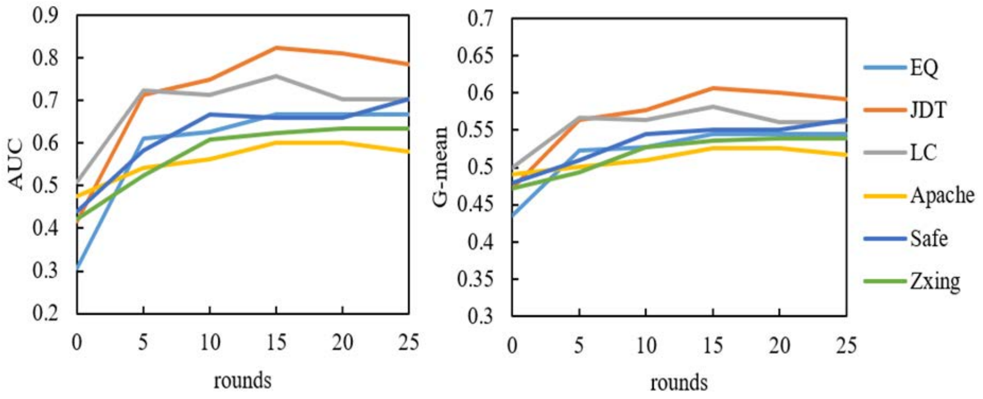


**FIGURE 6.** Prediction results of AEEEM and Relink as datasets. (a) AUC, (b) G-mean.

communicate with each other. It can be found most clients show stable convergence with increasing rounds. AUC and G-mean all increase in varying degrees in the process of continuous communication between the client and the central server for most clients.

In Fig. 6, AUC and G-mean increased by 0.25 and 0.08 on average compared with those before they did not participate in the communication. At 0<round<5 or 10, the growth rate is fast, and at round>20, most lines of clients tend to be stable. In Fig. 7, AUC and G-mean increased by 0.16 and 0.04 on average compared with those before they did not participate in the communication. At 0<round<5, the growth rate of most clients is fast, and at round>25, most lines of clients tend to be stable. In Fig. 8, the result with NASA and AEEEM is less stable compared to other experiments, which is possibly owing to poor performance of clustering. On the whole, AUC and G-mean increased by 0.27 and 0.21 on average compared with those before they did not participate in the communication.

## *E. EFFECTIVENESS OF OUR METHOD*

In order to evaluate the effectiveness of our method, we compared it with three classical non federated learning methods CCA+ [33], KCAA+ [34], KSETE [35] and two federated learning methods, FedAvg [37] and FTLKD [14]. The classical methods of non-federated learning use LR as the classifier, six datasets as source data, and each dataset tested as target data. The federated learning method sets each data set to six clients for testing. Taking AUC and G-mean as the measurement indicators of each project, the best effect is expressed in bold font.

In Table 3, AUC and G-mean are 0.6032 ∼ 0.7181 and 0.5283 ∼ 0.5693, respectively, with average values of 0.6571 and 0.5519, respectively. Compared with CCA+, KCAA+, KSETE, FedAvg and FTLKD, the average value increased by (14.06%, 5.97%), (9.67%, 5.03%), (10.06%, 4.28%), (15.6%, 6.54%), (9.29%, 4.36%) respectively. The FRLGC has better prediction results for JDT, while AUC and G-mean values on Zxing are lower. It is better than the comparison method in JDT, Safe and Apache and performance.

In Table 4, AUC and G-mean are 0.4769 ∼ 0.75 and 0.4659 ∼ 0.5774 respectively, with average values of 0.5993 and 0.51187 respectively. Compared with CCA+, KCAA+, KSETE, FedAvg and FTLKD, the average value increased by (6.83%, 1.1%), (7.1%, 0.93%), (9%, 1.47%), (8.79%, 1.8%), (4.54%, 0.85%) respectively. The FRLGC has

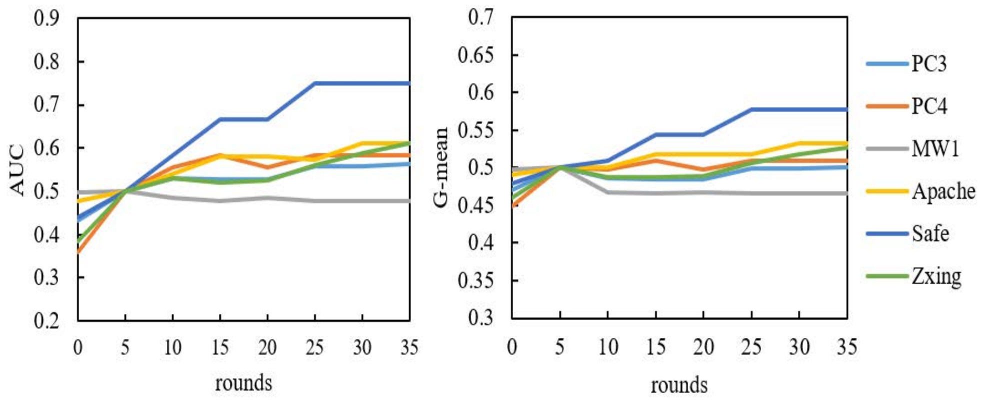


**FIGURE 7.** Prediction results of NASA and Relink as datasets. (a) AUC, (b) G-mean.

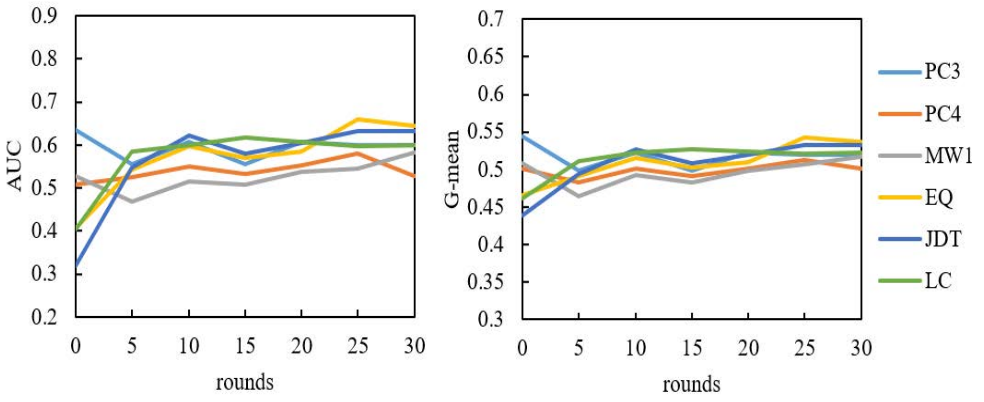


**FIGURE 8.** Prediction results of NASA and AEEEM as datasets. (a) AUC, (b) G-mean.

**TABLE 3.** Result of the comparison methods when AEEEM and Relink as the datasets.

| Project | Index | CCA+ | KCAA+ | KSETE | FedAvg | FTLKD | FRLGC |
|---|---|---|---|---|---|---|---|
| EQ | AUC | 0.5389 | 0.5907 | 0.5729 | 0.4894 | **0.6880** | 0.6680 |
| | G-mean | 0.4438 | 0.5357 | 0.5334 | 0.4957 | **0.5560** | 0.5497 |
| JDT | AUC | 0.5092 | 0.5478 | 0.5570 | 0.5087 | 0.5119 | **0.7181** |
| | G-mean | 0.5035 | 0.4206 | 0.5064 | 0.4893 | 0.4907 | **0.5693** |
| LC | AUC | 0.5444 | 0.5725 | 0.5384 | 0.5126 | **0.7000** | 0.6935 |
| | G-mean | 0.5137 | 0.5176 | 0.5043 | 0.5002 | 0.5578 | **0.5608** |
| Apache | AUC | 0.5275 | 0.5524 | 0.4916 | 0.4849 | 0.4688 | **0.6071** |
| | G-mean | 0.5075 | 0.4959 | 0.4606 | 0.4948 | 0.4640 | **0.5432** |
| Safe | AUC | 0.4761 | 0.5976 | 0.5665 | 0.5361 | 0.4000 | **0.6524** |
| | G-mean | 0.4891 | 0.5395 | 0.4972 | 0.5113 | 0.4472 | **0.5603** |
| Zxing | AUC | 0.5029 | 0.5011 | 0.6127 | 0.4750 | **0.6167** | 0.6032 |
| | G-mean | 0.4956 | 0.5001 | 0.5529 | 0.4877 | **0.5340** | 0.5283 |
| Average | AUC | 0.5165 | 0.5604 | 0.5565 | 0.5011 | 0.5642 | **0.6571** |
| | G-mean | 0.4922 | 0.5016 | 0.5091 | 0.4865 | 0.5083 | **0.5519** |

better prediction results for Safe, while AUC and G-mean values on PC4 are lower. It is better than the comparison method in PC3, Safe, Apache and Zxing performance.

In Table 5, AUC and G-mean are 0.553 ~ 0.6447 and 0.5014 ~ 0.5365 respectively, with average values of 0.602 and 0.5216 respectively. Compared with CCA+, KCAA+, KSETE, FedAvg and FTLKD, the average value increased by (10.56%,2.7%), (17.64%, 4.89%), (6.11%, 1.46%), (10.65%, 2.83%), (3.47%, 0.41%) respectively. The FRLGC has better prediction results for EQ, while AUC and G-mean values on PC are lower. It is better than the comparison method in EQ, JDT and LC performance.

Table 3, Table 4 and Table 5 show that each client performs best when AEEEM and Relink are used as datasets.

**TABLE 4.** Result of the comparison methods when NASA and Relink as the datasets.

| Project | Index | CCA+ | KCAA+ | KSETE | FedAvg | FTLKD | FRLGC |
|---|---|---|---|---|---|---|---|
| PC3 | AUC | 0.5112 | 0.4850 | 0.5054 | 0.5002 | 0.5196 | **0.5625** |
| | G-mean | 0.4991 | 0.4994 | 0.5040 | 0.4982 | 0.5027 | **0.5000** |
| PC4 | AUC | **0.5859** | 0.5224 | 0.4170 | 0.5060 | 0.4783 | 0.5834 |
| | G-mean | **0.5199** | 0.4882 | 0.4651 | 0.5006 | 0.4663 | 0.5092 |
| MW1 | AUC | 0.4998 | 0.5474 | 0.5511 | 0.5805 | **0.7464** | 0.4769 |
| | G-mean | 0.5005 | 0.5104 | 0.5209 | 0.5241 | **0.5815** | 0.4659 |
| Apache | AUC | 0.5107 | 0.5732 | 0.4772 | 0.5051 | 0.5625 | **0.6118** |
| | G-mean | 0.5019 | 0.5333 | 0.4904 | 0.5017 | 0.5103 | **0.5326** |
| Safe | AUC | 0.5231 | 0.5572 | 0.5230 | 0.4813 | 0.5000 | **0.7500** |
| | G-mean | 0.5065 | 0.5228 | 0.5070 | 0.4869 | 0.5000 | **0.5774** |
| Zxing | AUC | 0.5550 | 0.4843 | 0.5832 | 0.4954 | 0.5167 | **0.6111** |
| | G-mean | 0.5184 | 0.5022 | 0.5369 | 0.4928 | 0.5004 | **0.5273** |
| Average | AUC | 0.5310 | 0.5283 | 0.5095 | 0.5114 | 0.5539 | **0.5993** |
| | G-mean | 0.5077 | 0.5094 | 0.5041 | 0.5007 | 0.5102 | **0.5178** |

**TABLE 5.** Result of the comparison methods when NASA and AEEEM as the datasets.

| Project | Index | CCA+ | KCAA+ | KSETE | FedAvg | FTLKD | FRLGC |
|---|---|---|---|---|---|---|---|
| PC3 | AUC | 0.4148 | 0.4875 | 0.5172 | 0.4641 | **0.7333** | 0.5989 |
| | G-mean | 0.4588 | 0.4891 | 0.5119 | 0.4821 | **0.6086** | 0.5201 |
| PC4 | AUC | 0.5176 | 0.3294 | 0.6051 | 0.4830 | **0.6179** | 0.5530 |
| | G-mean | 0.4843 | 0.4558 | 0.5380 | 0.4943 | **0.5524** | 0.5014 |
| MW1 | AUC | 0.4449 | 0.4882 | **0.6150** | 0.4432 | 0.5514 | 0.5825 |
| | G-mean | 0.4726 | 0.4983 | **0.5277** | 0.4797 | 0.4989 | 0.5173 |
| EQ | AUC | 0.5782 | 0.3795 | 0.3167 | 0.5421 | 0.6062 | **0.6447** |
| | G-mean | 0.5331 | 0.4444 | 0.4121 | 0.5065 | 0.5278 | **0.5365** |
| JDT | AUC | 0.6104 | 0.3126 | 0.6623 | 0.5308 | 0.4167 | **0.6326** |
| | G-mean | 0.5525 | 0.4372 | 0.5480 | 0.5035 | 0.4513 | **0.5321** |
| LC | AUC | 0.4125 | 0.5563 | 0.5294 | 0.5102 | 0.4783 | **0.6004** |
| | G-mean | 0.4660 | 0.5116 | 0.5046 | 0.4935 | 0.4663 | **0.5222** |
| Average | AUC | 0.4964 | 0.4256 | 0.5410 | 0.4956 | 0.5673 | **0.6020** |
| | G-mean | 0.4946 | 0.4727 | 0.5071 | 0.4933 | 0.5176 | **0.5216** |

The reason of best performance is that AEEEM and Relink have higher intra-cluster similarity (density) compared to the other datasets, which is good at using similarity knowledge of client to guide FL aggregation.

In general, the prediction effect of the FRLGC is better than CCA+, KCAA+, KSETE, FedAvg and FTLKD, indicating that the FRLGC has better prediction performance.

## IV. CONCLUSION

In this paper, we propose FRLGC for heterogeneous defect prediction, which can solve the problems of data island and heterogeneity and increase the data efficiency in the case of limited datasets. Each client jointly constructs a global model by communicating without exposing privacy, which can improve the performance of private models. Firstly, the parameters of the global model are transferred to each dueling DQN model and each client use private data to train the dueling DQN model which combines experience replay to increase data efficiency in limited datasets. Secondly, gaussian differential privacy is used to encrypt the model parameters to ensure the privacy and security of the model. Finally, we cluster the selected clients according to their locally encrypted model parameters and use weighted average to aggregate locally and aggregate globally to create a new global model and solve data heterogeneity problems. The experimental results show that FRLGC has good convergence and a better prediction performance than the existing HDP solutions.

In the future, we will expand the experiment to more defect datasets to test the generalization ability of our method. We are considering that FRLGC needs multiple rounds of communication for each client to realize the convergence of private model. Therefore, more consideration should be given to the communication costs. Therefore, we must create a new algorithm in FRLGC to reduce communication costs.

## REFERENCES

[1] W. Li, Z. Huang, and Q. Li, "Three-way decisions based software defect prediction," *Knowl.-Based Syst.*, vol. 91, pp. 263–274, Jan. 2016.

[2] P. Singh, N. R. Pal, S. Verma, and O. P. Vyas, "Fuzzy rule-based approach for software fault prediction," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 47, no. 5, pp. 826–837, May 2017.

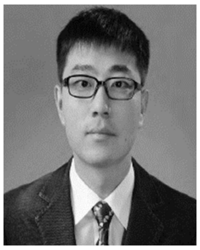
[3] Z. Xu, J. Liu, X. Luo, Z. Yang, Y. Zhang, P. Yuan, Y. Tang, and T. Zhang, "Software defect prediction based on kernel PCA and weighted extreme learning machine," *Inf. Softw. Technol.*, vol. 106, pp. 182–200, Feb. 2019.

[4] S. Herbold, A. Trautsch, and J. Grabowski, "A comparative study to benchmark cross-project defect prediction approaches," *IEEE Trans. Softw. Eng.*, vol. 44, no. 9, pp. 811–833, Sep. 2018.

[5] D. Chen, X. Chen, H. Li, J. Xie, and Y. Mu, "DeepCPDP: Deep learning based cross-project defect prediction," *IEEE Access*, vol. 7, pp. 184832–184848, 2019.
[6] C. Ni, X. Xia, D. Lo, X. Chen, and Q. Gu, "Revisiting supervised and unsupervised methods for effort-aware cross-project defect prediction," *IEEE Trans. Softw. Eng.*, vol. 48, no. 3, pp. 786–802, Mar. 2022.
[7] T. Zimmermann, N. Nagappan, H. Gall, E. Giger, and B. Murphy, "Cross-project defect prediction: A large scale experiment on data vs. domain vs. process," in *Proc. 7th Joint Meeting Eur. Softw. Eng. Conf. ACM SIGSOFT Symp. Found. Softw. Eng. Eur. Softw. Eng. Conf. Found. Softw. Eng. Symp. (ESEC/FSE)*, Aug. 2009, pp. 91–100.
[8] L. Gong, S. Jiang, and L. Jiang, "Conditional domain adversarial adaptation for heterogeneous defect prediction," *IEEE Access*, vol. 8, pp. 150738–150749, 2020.
[9] P. Shen, X. Ding, X. Mu, and J. Xu, "A software defect prediction method based on sampling and integration," in *Proc. SCSET*. Shanghai, China, Oct. 2021, doi: 10.1088/1742-6596/1732/1/012002.
[10] J. Wu, Y. Wu, M. Zhou, and X. Jiang, "SLA+: Narrowing the difference between data sets in heterogenous cross-project defection prediction," in *Proc. IEEE 44th Annu. Comput., Softw., Appl. Conf. (COMPSAC)*, Jul. 2020, pp. 1229–1234.
[11] C. Jin, "Cross-project software defect prediction based on domain adaptation learning and optimization," *Expert Syst. Appl.*, vol. 171, Jun. 2021, Art. no. 114637.
[12] Z. Li, J. Niu, X.-Y. Jing, W. Yu, and C. Qi, "Cross-project defect prediction via landmark selection-based kernelized discriminant subspace alignment," *IEEE Trans. Rel.*, vol. 70, no. 3, pp. 996–1013, Sep. 2021.
[13] H. Tong, B. Liu, and S. Wang, "Kernel spectral embedding transfer ensemble for heterogeneous defect prediction," *IEEE Trans. Softw. Eng.*, vol. 47, no. 9, pp. 1886–1906, Sep. 2021.
[14] A. Wang, Y. Zhang, and Y. Yan, "Heterogeneous defect prediction based on federated transfer learning via knowledge distillation," *IEEE Access*, vol. 9, pp. 29530–29540, 2021.
[15] Y. Bai and M. Fan, "A method to improve the privacy and security for federated learning," in *Proc. IEEE 6th Int. Conf. Comput. Commun. Syst. (ICCCS)*, Apr. 2021, pp. 704–708.
[16] W. Xu, W. Fang, Y. Ding, M. Zou, and N. Xiong, "Accelerating federated learning for IoT in big data analytics with pruning, quantization and selective updating," *IEEE Access*, vol. 9, pp. 38457–38466, 2021.
[17] Z. Su, Y. Wang, T. H. Luan, N. Zhang, F. Li, T. Chen, and H. Cao, "Secure and efficient federated learning for smart grid with edge-cloud collaboration," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1333–1344, Feb. 2022.
[18] Y. Ye, S. Li, F. Liu, Y. Tang, and W. Hu, "EdgeFed: Optimized federated learning based on edge computing," *IEEE Access*, vol. 8, pp. 209191–209198, 2020.
[19] S. Adam, L. Busoniu, and R. Babuska, "Experience replay for real-time reinforcement learning control," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 2, pp. 201–212, Mar. 2012.
[20] V. Mnih, K. Kavukcuoglu, D. Silver, A. Rusu, J. Veness, M. Bellemare, A. Graves, M. Riedmiller, A. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[21] H. H. Zhuo, W. Feng, Y. Lin, Q. Xu, and Q. Yang, "Federated deep reinforcement learning," 2019, *arXiv:1901.08277*.
[22] S. Lee and D.-H. Choi, "Federated reinforcement learning for energy management of multiple smart Homes with distributed energy resources," *IEEE Trans. Ind. Informat.*, vol. 18, no. 1, pp. 488–497, Jan. 2022.
[23] Y. Hu, Y. Hua, W. Liu, and J. Zhu, "Reward shaping based federated reinforcement learning," *IEEE Access*, vol. 9, pp. 67259–67267, 2021.
[24] S. Ek, F. Portet, P. Lalanda, and G. Vega, "A federated learning aggregation algorithm for pervasive computing: Evaluation and comparison," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. (PerCom)*, Mar. 2021, pp. 1–10.
[25] J. Pang, Y. Huang, Z. Xie, Q. Han, and Z. Cai, "Realizing the heterogeneity: A self-organized federated learning framework for IoT," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3088–3098, Mar. 2021.
[26] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 8, pp. 3710–3722, Aug. 2021.
[27] W. Huang, J. Liu, T. Li, T. Huang, S. Ji, and J. Wan, "FedDSR: Daily schedule recommendation in a federated deep reinforcement learning framework," *IEEE Trans. Knowl. Data Eng.*, vol. 9, pp. 2169–3536, 2021.
[28] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, nos. 3–4, pp. 211–407, 2014.
[29] Z. Wang, T. Schaul, M. Hessel, H. van Hasselt, M. Lanctot, and N. de Freitas, "Dueling network architectures for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1995–2003.
[30] M. Shepperd, Q. Song, Z. Sun, and C. Mair, "Data quality: Some comments on the NASA software defect datasets," *IEEE Trans. Softw. Eng.*, vol. 39, no. 9, pp. 1208–1215, Sep. 2013.
[31] M. D'Ambros, M. Lanza, and R. Robbes, "An extensive comparison of bug prediction approaches," in *Proc. 7th IEEE Work. Conf. Mining Softw. Repositories (MSR)*, May 2010, pp. 31–41.
[32] R. Wu, H. Zhang, S. Kim, and S.-C. Cheung, "ReLink: Recovering links between bugs and changes," in *Proc. 19th ACM SIGSOFT Symp. 13th Eur. Conf. Found. Softw. Eng. (SIGSOFT/FSE)*, 2011, pp. 15–25.
[33] X. Jing, F. Wu, X. Dong, F. Qi, and B. Xu, "Heterogeneous cross-company defect prediction by unified metric representation and CCA-based transfer learning," in *Proc. 10th Joint Meeting Found. Softw. Eng.*, Aug. 2015, pp. 496–507.
[34] Y. Ma, S. Zhu, Y. Chen, and J. Li, "Kernel CCA based transfer learning for software defect prediction," *IEICE Trans. Inf. Syst.*, vol. E100.D, no. 8, pp. 1903–1906, 2017.
[35] H. Tong, B. Liu, and S. Wang, "Kernel spectral embedding transfer ensemble for heterogeneous defect prediction," *IEEE Trans. Softw. Eng.*, vol. 47, no. 9, pp. 1886–1906, Sep. 2021.
[36] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," 2016, *arXiv:1602.05629*.

**AILI WANG** (Member, IEEE) was born in Tianjin, China, in 1979. She received the B.S., M.S., and Ph.D. degrees in information and signal processing from the Harbin Institute of Technology, Harbin, China, in 2002, 2004, and 2008, respectively. She joined the Harbin University of Science and Technology as an Assistant, in 2004, and she became an Associate Professor and a Master Tutor of the Department of Communication Engineering, in 2010. She was a Visiting Professor to do search of 3D polyp reconstruction with the Computer Science Laboratory, Chubu University, Japan, in 2014. She is the author of two books, more than 100 articles which are published on IEEE conferences and journals (EI indexed or SCI indexed). Her research interests include image super resolution, image fusion, object tracking, software engineering, and reinforcement learning and federated learning. She is the Chairperson of the 11th EAI International on Wireless and Satellites (WISATS) and the 7th EAI International Conference on Green Energy and Networking (GreeNets).

**YINGHUI ZHAO** received the bachelor's degree from the College of Electronics Information Engineering, Shenyang Aerospace University, in 2020. She is currently pursuing the master's degree with the College of Communication and Information System, Harbin University of Science and Technology. Her research interests include software engineering and reinforcement learning and federated learning.



**GUODONG LI** received the B.Sc. and M.Sc. degrees from Harbin Engineering University, Harbin, China, in 2001 and 2004, respectively, and the Ph.D. degree from the Harbin Institute of Technology, Harbin, in 2016.

He is currently a Lecturer with the School of Measurement-Control Technology and Communications Engineering, Harbin University of Science and Technology, Harbin. His current research interests include complementary coded CDMA, machine learning for communications and multicarrier systems, software engineering, and reinforcement learning and federated learning.

**JUN ZHANG** was born in Heilongjiang, China, in 1983. He received the B.S. degree in civil engineering from the Kunming University of Science and Technology, in 2005. He is currently working at China Taishan Energy Power for 17 years. His research interests include steam temperature control and fuel combustion and flue gas purification based on deep learning and federated learning.

**HAIBIN WU** was born in Harbin, China, in 1977. He received the B.S. and M.S. degrees from the Harbin Institute of Technology, Harbin, in 2000 and 2002, respectively, and the Ph.D. degree in measuring and testing technologies and instruments from the Harbin University of Science and Technology, Harbin, in 2008. From 2009 to 2012, he held a Postdoctoral Researcher with the Key Laboratory of Underwater Robot, Harbin Engineering University. From 2014 to 2015, he was a Visiting Scholar with the Robot Perception and Action Laboratory, University of South Florida. Since 2012, he has been a Professor with the Instrument Science and Technology Discipline, Harbin University of Science and Technology. He is the author of three books, more than 40 articles, and more than 20 inventions. His research interests include robotic vision, visual measuring and image processing, medical virtual reality, and photoelectric testing. He was the Director of the Precision Machinery Branch, China Instrumentation Society, and the Director of the Visual Inspection Committee and Chinese Graphic and Image Society. He serves as an Editorial Board Member for *Chinese Journal of Liquid Crystals and Displays* and an Associate Editor-in-Chief for *Journal of Harbin University of Science and Technology*.

**YUJI IWAHORI** was born in Japan, in 1959. He received the B.S. degree from the Nagoya Institute of Technology, in 1983, and the M.S. and Ph.D. degrees from the Tokyo Institute of Technology, in 1985 and 1988, respectively. He joined the Nagoya Institute of Technology, in 1988. He was a Professor with the Nagoya Institute of Technology, in 2002. He has been a Visiting Researcher with UBC, since 1991. He has been with Chubu University, as a Professor, since 2004. He has also been a Research Collaborator with IIT Guwahati, since 2010. Since 2010, he has been engaged in research of endoscopic polyps with Aichi Medical University. He was also engaged in biomedical engineering with the Nagoya University of Technology. He has been with Chulalongkorn University, since 2014. He has published 91 high-level journal articles and 156 international conference papers. He held 149 lectures. His research interests include computer vision, image recognition, deep learning and neural networks, pattern recognition and pattern classification, bioinformatics, biomedical imaging and artificial intelligence applications, and research of medical images. He is a member of the committee of KES, KES IDT, the IEEE/ACIS, ICIS, CSII, BIBE, the IEEE SITIS, HI-BI-BI, *Patterns*, *Signals*, ACIVS, and other international conferences. He serves as a Peer-Reviewer for KES journals, such as the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, *Multimedia Tools and Applications* (MTAP) (Springer), and IEEJ. He serves as an Editorial Board Member for *The Open Bioinformatics Journal*, *Journal of Robotics and Mechanical Engineering Research*, and the *International of Journal of Computer Science and Engineering*.