Research on Aggregation of Federated Model for Software Defect Prediction Based on Dynamic Selection

Huiling Song[1,2], Yong Li[1,*], Wenjing Zhang[1], and Ying Liu[1]
[1]College of Computer Science and Technology, Xinjiang Normal University, Urumqi, Xinjiang, China
[2]Xinjiang Electronics Research Institute, Urumqi, Xinjiang, China
Huiling20@163.com, liyong@live.com, 1633171055@qq.com, 846532344@qq.com
*corresponding author

Abstract—With the continuous expansion of the application scope of computer software, the standards of software quality of various organizations are becoming more and more strict. As a testing method, software defect prediction can effectively improve software quality. Software defect prediction is to predict the potential defect modules in advance at the early stage of project development, so as to improve the software quality. The software defect prediction under Federal learning realized the data sharing of all project parties, but the problem of low-quality data holders was not considered in the aggregation phase. Therefore, the dynamically selected software defect prediction federated model aggregation method (DS-SDP). Firstly, a threshold is set on the server side based on the public dataset. To prevent malicious attacks, participants add Gaussian noise to the parameters after completing local training and transmit them to the server. Then, during the aggregation process, the server uses a dynamic selection method to filter participants who do not meet the threshold criteria, and weights the participants who meet the criteria. Finally, construct a convolutional neural network model. Experiments based on NASA software defect prediction dataset show that compared with the traditional federated aggregation method, the DS-SDP method proposed in this paper improves the model performance, greatly reduces the impact of low-quality data on the model, and also enhances the robustness of the model.

Keywords-Software defect prediction; Federated learning; Low quality data; Dynamic aggregation

1. INTRODUCTION

Defects form in the early development process of software and can be restored through later repairs. Early testing of software defects through technical means is an effective way to ensure software reliability. Software defect prediction technology is to build a prediction model based on the historical data accumulated in the software development process to predict whether the target software module has defects, defect severity or defect number distribution [1][2][3][4]. Software defect prediction hopes to identify potential defective program modules in the project in advance at the early stage of project development, and allocate sufficient test resources to such program modules to ensure that adequate code review or unit testing can be carried out, ultimately achieving the purpose of improving the quality of software products [5]. Therefore, combining machine learning and deep learning methods to predict whether there are defects in the software and ensure software quality under controllable factors, it not only reduces the cost of later detection and repair, but also avoids unnecessary losses caused by software failures.

In the data-driven era, data is the foundation for building models, and high-quality data can maximize its value. The better the data quality, the better the performance of the model. Conversely, low quality data cannot guarantee model performance. In the field of software defects, the combination of federated learning and software defect prediction has enabled data sharing between project. Federated learning is a special machine learning method that breaks traditional training methods and enables training of global models without exposing user data. This method can avoid centralized management of user data and reduce communication overhead caused by transmitting large amounts of data. Federated learning has been widely applied in various fields such as smartphones, IoT devices, and healthcare. This method can effectively protect user privacy while ensuring the accuracy and reliability of the model. However, in the traditional federated learning aggregation process, the server directly aggregates the passed parameters, without checking the data quality of project participants, while low-quality data directly affects the final model performance. To address this issue, this article proposes an Aggregation of Dynamically Selected Federated Models for Software Defect Prediction (DS-SDP). This method filters project data parties that do not meet the standards during the server aggregation stage, and aggregates the model parameters of project participants that meet the data standards. This article uses publicly available NASA datasets for experiments and compares them with traditional federated aggregation methods. The results show that the method proposed in this article can effectively address the negative impact of low-quality clients in model training, not only reducing unnecessary communication costs, but also improving model performance. Has a positive impact on model robustness.

Section 1 of this paper introduces the important role of data quality in the model, and proposes corresponding solutions for low-quality data. Section 2 introduces the background and research work related to software defect prediction and federated learning. Section 3 introduces the aggregation method of dynamically selected model parameters and the aggregation method of dynamically selected software defect

2767-6684/23/$31.00 ©2023 IEEE
DOI 10.1109/DSA59317.2023.00037

prediction federated model. The fourth section is based on the NASA dataset, with convolutional neural networks as the basic model. Through experimental analysis and comparison, the effectiveness of the proposed method is demonstrated. Finally, a summary and outlook are provided for this article.

2. Related Work

2.1 Research on Software Defect Prediction

Software defect prediction mining appropriate historical data from the database as the basic data for testing new development projects, followed by measuring and defect labeling the mined historical data, and finally constructing a model based on machine learning or deep learning methods to achieve rapid testing of software modules by developers. From a data perspective, software defect prediction includes intra project software defect prediction [6] and cross project software defect prediction [7].

Chang [8] proposed AdaBoost's ensemble learning method, which uses SVM algorithm as the base classifier. Experimental results show that this method can effectively improve model performance. Liu [9] proposed a cost sensitive learning based software defect prediction method based on convolutional neural networks, which has the best model performance compared to traditional methods. Bennin [10] proposed a new efficient synthesis oversampling method of software defect data set based on chromosome genetic theory, which has good effect on solving class imbalance. Pan [11] proposed a typical principal component analysis method, whose core content is to minimize the data distribution distance between source and target projects. However, this method is sensitive in the data preprocessing stage and affects the performance of cross project models. Subsequently, Nam [12] proposed the TCA+method based on TCA, which selects appropriate normalization options based on given cross project predictions. Ni Chao [13] proposed a feature and instance based cross project defect prediction method (FeCTrA). This method combines features with instance migration, and the FeCTrA method improves cross project model performance by 23% compared to the TCA+method.

While improving software defect prediction performance, people are increasingly paying attention to data privacy issues. Peters [14] first proposed the CLIFF+MORPH method to achieve privatization of shared data across projects. But this is based on a small amount of data, only considering the privatization of unilateral data without considering multi-party data. Faced with such problems, Peters [15] added buffer pools based on their previous work, allowing data holders to incrementally add data to the private cache they pass between them based on the existing content in the private cache. Li [16] designed a dual fuzzy algorithm based on sparse representation and applied it to HDP for software defect prediction in heterogeneous scenarios. Zhang [17] achieved prediction of heterogeneous software defects in federated scenarios.

2.2 Federated Learning Research

In recent years, privacy breaches have emerged one after another, bringing significant negative impacts on political, economic, and social life. In response to such issues, domestic and foreign laws and regulations on privacy protection are also constantly improving. Typical examples include the "Cybersecurity Law of the People's Republic of China" implemented in 2017 and the "General Data Protection Regulations" implemented in the European Union in 2018. These documents have made clear provisions for data collection and processing. Data also possesses the attributes of assets [18]. All data owners have begun to value data and are unwilling to share it, which directly leads to a fragmented state of data. Data plays a crucial role in the field of artificial intelligence, and without it, model training cannot be carried out. Faced with this major obstacle, federated learning technology has emerged.

The core idea of federated learning is that data remains unchanged and models move. The model can be trained and achieve the expected results without leaving the data locally. Feng [19] proposed a secure federated aggregation method from the perspective of attackers, effectively improving the reliability of federated learning. WANG [20] proposed using knowledge distillation to achieve training for each participant and update private models. Xu Chenyang [21] calculated sample similarity based on boundary extension local sensitive hashing and conducted local training. Chen [22] proposed that in non independent distribution scenarios, medical personalization can be achieved by learning the similarity between clients through batch processing of statistical information in the normalization layer, while preserving the features of the clients. Wu [23] proposed a federated learning scheme combining adaptive gradient descent strategy and differential privacy mechanism to avoid overfitting of private models. Fang [24] studied the federated learning method of homomorphic encryption and proposed an improved Paillier algorithm, which effectively improved the training speed. Xin [25] proposed a private generative adversarial network, which can also be applied in non independent distribution scenarios.

To sum up, previous studies only considered the privacy problems in software defect prediction technology, ignoring the impact of low-quality data on the model. Therefore, this article proposes a dynamically selected federated model aggregation method for software defect prediction. On the one hand, differential privacy mechanism is introduced to prevent malicious attacks, and on the other hand, low-quality data is filtered to improve the accuracy of the model, which is of great significance to the research of software defect prediction.

3. Dynamic Selection Based Federated Model Aggregation Algorithm for Software Defect Prediction

3.1 DS-MA Algorithm

In the practical application of software defect prediction, the data volume of each project data holder is often different. Distributed learning is the process of collecting all data and broadcasting it equally to each site. Unlike distributed learning, the data volume of each participant in federated learning scenarios is completely different. To solve the problem of data volume, the federated learning server aggregation stage adopts the federated average algorithm. In the existing research, FedAvg is widely used in the deep neural network model [26], which can be used to solve the non convex loss function in the model. Let's assume there are K clients, where n represents the data volume of the client and w represents the parameters in the neural network model, $f_i(w)$ represents the predicted loss value of w on the i-th training sample, p_k represents the index set of the k-th client's data points, and t represents the global training for the t-th round. Formulas (1)、 (2) and (3) show that under the same distribution of data, it has a good effect on the value of non convex loss function. Formula (4) is a federal average algorithm formula, and literature [27] proves that the model has good performance under different data distribution states.

$$f_i(w) = \varsigma(x_i, y_i, w) \tag{1}$$

$$F_k(w) = \frac{1}{n_k} \sum_{i \in p_k} f_i(w) \tag{2}$$

$$f(w) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(w) \tag{3}$$

$$\overline{w}_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k \tag{4}$$

The FedAvg algorithm performs well on the premise of high-quality data, but the quality of parameter data in the actual scene is unknown, which may lead to damage to the model performance. To solve this problem, a dynamic selection model aggregation algorithm (DS-MA) is proposed.

In the parameter transfer stage, the parameters transferred by the participants include local model parameters and loss function values. The loss function is an evaluation method of sample prediction effect. The larger the loss value, the less ideal the prediction effect. Filter participants with low data quality by setting thresholds. Specifically, the server selects a data set with good performance as the public data set for pre training, gets the initial global model parameters and loss function values, and then broadcasts the global parameters to the participants. After the participants finish local training, they get private model parameters and loss values. After the loss values are passed to the server, the server judges whether the participants meet the training standards according to the threshold value [28]. When the loss value of the participant exceeds the threshold, they will no longer participate in the next round of training. Formula (5) is the cross entropy loss function of the two classifications. Figure 1 shows the aggregation process of a dynamically selected model. DS_ MA aggregation algorithm updates pseudocode as shown in algorithm 1.

$$L = -[y \log \hat{y} + (1-y) \log(1-\hat{y})] \tag{5}$$

L is the cross entropy loss function, y represents the true value, $\hat{y}$ represents the predicted value. The cross entropy loss function can measure the difference between the real result and the predicted result. The smaller the L value, the closer the predicted result is to the real result. On the contrary, the prediction effect is not ideal.

Algorithm 1 DS-MA aggregation algorithm

Input: Global training rounds *N*, pubic data, local model parameters w_i , *N* clients C={1,2... n}, The ith customer's loss function $Loss_i$

Output: aggregated model parameters $\overline{w}_{t+1}$

1. Calculate global initial parameters w^0 、 threshold loss
2. for epoch t in range(N) do
// Parameter aggregation is performed for each round
If $loss_i^t > loss_t$ // Filter participant information
i not in C={1,2…n}
else

$\overline{w}_{t+1} \leftarrow \sum_{k=1}^{k} \frac{n_k}{n} w_t^k$ // weighted mean

3. t=t+1
4. $\overline{w}_{t+1} \in C$
6. print($\overline{w}_{t+1}$)

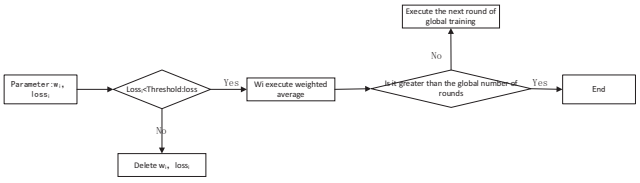


Figure 1. Model aggregation flowchart for dynamic selection

3.2 DS-SDP Algorithm

Combining software defect prediction with federated learning technology can achieve model training in privacy preserving scenarios. Firstly, the feature selection method is used to map the features of the project side datasets to the same space as the common dataset, server training public dataset to obtain global parameters w^0 , And send it to the client end (steps 1-3). Then the server randomly selects the client for training, and the selected project party conducts local training to obtain the

loss function value and model parameters. Due to the inability to ensure that every project party is honest during the process of establishing federated learning, there may be semi honest or malicious project parties attempting to obtain sensitive data from other project parties through model parameter information [29]. Therefore, in order to avoid malicious attacks on model parameters during their transmission to the server, which may result in the leakage of sensitive data from participating parties, Gaussian noise is introduced before the model parameters are transmitted to the client to enhance privacy protection (steps 4-5). Finally, the server aggregates the passed parameter information based on dynamic selection method and determines whether to stop training (steps 6-7). DP-SDP pseudocode is shown in algorithm 1. Figure 2 shows the framework of a cross project software defect prediction method based on federated learning.

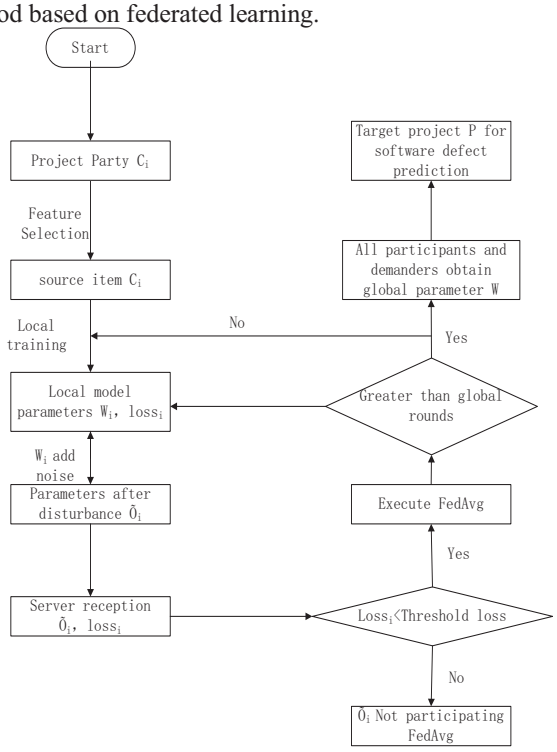


Figure 2. A Framework for Cross Project Software Defect Prediction Based on Federated Learning

The Gaussian mechanism satisfies L2 global sensitivity(ϵ,δ) Differential privacy has certain advantages in the process of federated learning. It calculates the noise value in the function based on parameter information. In the joint modeling of static software defect data, when L2 is larger, it is necessary to add larger noise to make function M meet the privacy budget. The noise generated by the Gaussian mechanism added by each project party is generated through the random number algorithm of the Numpy library, which conforms to the Gaussian distribution. Formulas 3-6 define the sensitivity function f for D1 and D2. Formulas (7) represent the result of adding noise to the Gaussian mechanism [30].

$$\triangle f_{sen} = \max_{D_1, D_2} \| f(D_1) - f(D_2) \| \quad (6)$$

$$M(d) \triangleq f(d) + N(0, s_f^2 \cdot \sigma) \quad (7)$$

$N(0, s_f^2 \cdot \sigma)$ represents the noise generated by the Gaussian mechanism, The standard deviation is $\delta = s_f^2 \cdot \sigma$, s_f is the sensitivity adjustment operator. When the privacy budget is smaller, the data availability is lower, and on the contrary, the degree of privacy protection is higher. For any $\delta \in (0,1)$, wherein $\sigma > \frac{\sqrt{2In\frac{1.25}{\delta}\triangle f}}{\varepsilon}$, Under static software defect prediction, there is noiseN (0, σ^2) that satisfies (ϵ,δ)-Differential privacy.

4. EXPERIMENT

4.1 Datasets and evaluation indicators

This article selects the NASA Aerospace Data MDP dataset for the experiment. The dataset includes 12 items. The NASA dataset has metrics ranging from 22 to 40; The project languages include C, C++, and Java; The number of project samples varies from 155 to 16792; The defect prediction rate for each project ranges from 0.41 to 48. The metric elements of each project are a set of features extracted using LOC, Halstead, and McCabe methods. The MDP dataset is widely used in the field of software defect prediction, making it easy to compare with other algorithms. This paper studies the binary classification of software defect prediction. The software defect prediction evaluation index is calculated according to the sample prediction rate in the confusion matrix. Table 4-1 is the confusion matrix.

The software defect prediction evaluation index is calculated according to the sample prediction rate in the confusion matrix. Table 1 is the confusion matrix.

Table 1. Confusion matrix table

	Forecast results	
	Positive example	***negative example***
Positive example	TP	FN
negative example	FP	TN

This article uses AUC and PD as performance indicators for software defect prediction and evaluation models. Formulas (8) and (9) define AUC and PD.

$$AUC = \int_0^1 PDd(PF) \quad (8)$$

$$PD = \frac{TP}{TP + FN} \quad (9)$$

Compared with other evaluation indicators, AUC focuses more on sorting results and is the main evaluation indicator for binary classification problems. The prediction rate PD, also known as the recall rate, represents the measure of correctly predicting defective modules in defective modules. The larger the PD value, the stronger the model's ability to predict defective modules.

4.2 Experimental Setup

To make the experiment comparable, this article uses PC5 as a public dataset and 10 different software projects as local participants to participate in model training. Based on the CNN global model, federated simulation is conducted. The experiment is divided into two parts:

(1) Compared with traditional aggregation methods, verify the effectiveness of the DS-SDP method.

Traditional aggregation methods assume that aggregation of model parameters sent by all participants can achieve good results on the premise that all participants have high-quality data. However, in real life, there may be inefficient model parameters that affect model training. In the model parameter aggregation stage, a dynamic selection method is used to select participants who meet the threshold and continue with the next round of training. If they do not meet the threshold, the iteration is stopped, which effectively prevents unnecessary communication and model loss to a certain extent.

The experiment was divided into eleven groups, with CM1, JM, KC1, KC3, MC1, MC2, MW1, PC1, PC2, PC3, and PC4 as target projects, and the remaining ten project datasets as individual participants. The number of clients varies, with Epoch set to 10, ε Is 0.3, is 0.02. Using AUC as the evaluation indicator, take the average AUC of each project, and compare the dynamically selected model aggregation method with the traditional federated average aggregation method to verify the effectiveness of the DS-SDP method.

(2) The impact of different client numbers on the DS-SDP method.

Due to the varying quality of data owned by different project data holders, other collaborators and third-party servers are unable to determine the data quality of the project data holders in scenarios where each data holder is unwilling to share data. Therefore, different numbers of participants are set up to analyze how the model performs in the context of dynamically selected model aggregation.

The experiment was divided into four groups of clients: 3, 5, 7, and 9. The experiment adopts the Control variates to change the number of participants and other conditions remain unchanged. Analyze the model performance and convergence under different client counts.

4.3 Experimental results

(1) Compare the DS-SDP method with the traditional federated learning cross project software defect prediction method to verify the effectiveness of the DS-SDP model. Figure 2 shows the AUC results of the DS-SDP method and the federated learning cross project software defect prediction method under the same number of communication rounds. Table 2 shows the AUC results of the two methods. The horizontal coordinates 1 to 11 represent CM1, JM, KC1, KC3, MC1, MC2, MW1, PC1, PC2, PC3, and PC4, respectively. The vertical axis represents the AUC value.

Table 2. AUC values

	CM1	***JM1***	***KC1***	***KC3***	***MC1***	***MC2***	***MW1***	***PC1***	***PC2***	***PC3***	***PC4***
FedAvg	0.702	0.693	0.691	0.656	0.669	0.656	0.725	0.667	0.655	0.715	0.681
DS_MA	0.796	0.744	0.717	0.721	0.79	0.662	0.796	0.817	0.721	0.768	0.767

Table 3. AUC values

number	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	10
clients3	0.690	0.704	0.707	0.706	0.711	0.711	0.711	0.711	0.711	0.711
clients5	0.715	0.714	0.720	0.720	0.720	0.720	0.720	0.725	0.725	0.725
clients7	0.719	0.719	0.719	0.719	0.719	0.723	0.723	0.723	0.723	0.723
clients9	0.720	0.720	0.720	0.720	0.720	0.725	0.725	0.725	0.725	0.730

Table 4. Recall values

number	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	10
clients3	0.397	0.397	0.408	0.502	0.509	0.519	0.538	0.55	0.631	0.665
clients5	0.426	0.434	0.453	0.501	0.530	0.537	0.602	0.666	0.667	0.687
clients7	0.402	0.414	0.463	0.485	0.523	0.549	0.556	0.667	0.674	0.696
clients9	0.431	0.458	0.470	0.491	0.494	0.556	0.626	0.639	0.647	0.681

Table 5. Loss Values

number	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	10
clients3	0.782	0.642	0.532	0.435	0.382	0.349	0.331	0.320	0.310	0.306
clients5	0.756	0.608	0.480	0.401	0.354	0.332	0.315	0.300	0.290	0.287
clients7	0.776	0.623	0.512	0.413	0.362	0.337	0.319	0.296	0.290	0.283
clients9	0.688	0.568	0.479	0.387	0.332	0.309	0.298	0.286	0.284	0.282

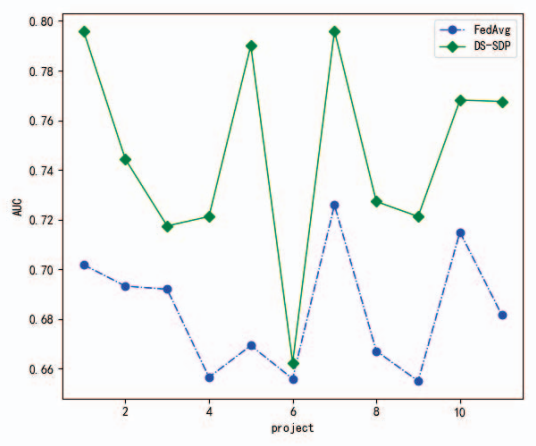
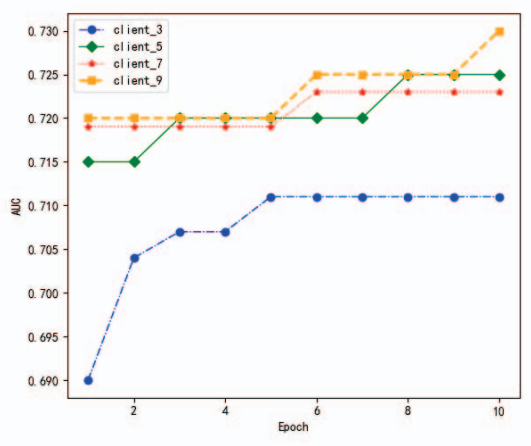


Figure 3. line chart Chart of AUC Results.

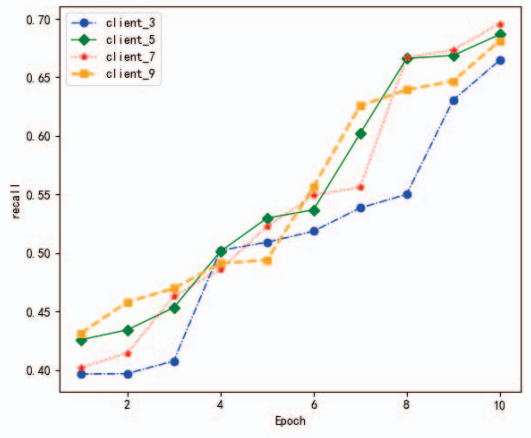
From Figure 3, it can be observed that the dynamic selection aggregation method overall performs better than traditional aggregation methods in model performance. Averaging the AUC values of 11 projects resulted in a traditional model aggregation method with an AUC value of 0.683 and a dynamically selected AUC value of 0.754. It can be observed that DS-SDP. The model performance of the MA method has improved by 0.071 overall. It is proved that the model is effective, and the dynamic selection method effectively reduces the impact of low-quality data holders on the model, greatly enhancing the robustness of the model.

(2) The impact of different number of clients on the model. In the process of federated learning, the effectiveness of the model varies when the number of participants is different. Tables 3、4 and 5 represent the experimental results of the performance and convergence of different client numbers on the model as the number of communication rounds increases. In order to display the results more intuitively, the line chart 3 is used to show the performance of the model with different number of clients. Among them, the horizontal axis represents the number of communication rounds, and the vertical axis represents AUC, recall, and convergence, respectively.

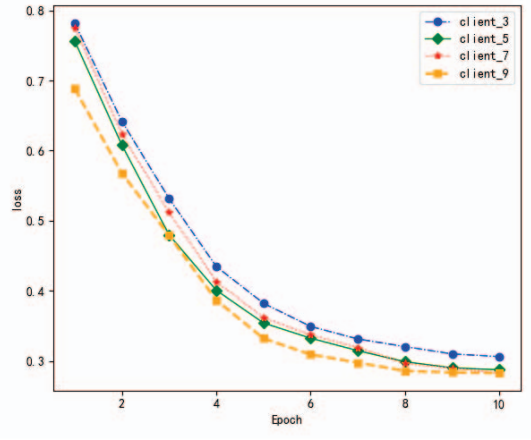
From Figure 4, it can be seen that as the number of communication rounds increases, the overall performance of the model also increases. When the number of communication rounds reaches 10, the model tends to converge. The more clients participate in training, the better the model performance. When the number of clients is 3, the value of the loss function is the largest. The performance of AUC and recall rate is relatively poor compared with other models. When the number of clients is 9, the performance of the model is the best. When the number of clients is 5, the final model effect is better than when the number of clients is 7. This is in line with the fact that there may be many low-quality data scenarios in practical applications. Moreover, compared to traditional aggregation methods, the convergence speed of dynamic aggregation model aggregation methods has also increased. Overall, the more clients involved, the better the performance of the model.



(a) AUC line chart



(b) recall line chart



(c) loss line chart

Figure 4. line chart Chart of Experimental Results

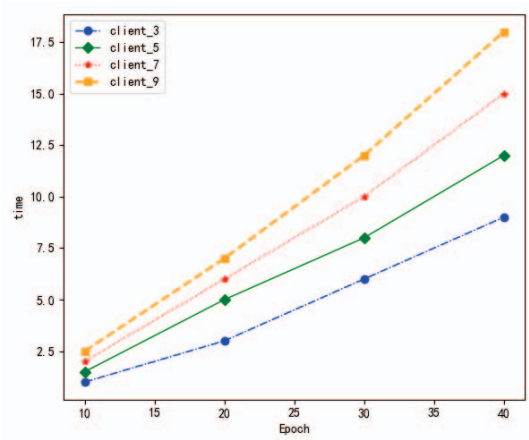


Figure 5. line chart Chart of Experimental Results

Figure 5 shows the relationship between the number of communication rounds and communication time under different numbers of clients. The vertical axis represents time in minutes. From the experimental results, it can be seen that as the number of communication rounds increases, the communication time also increases in a proportional manner, and the more clients there are, the longer the communication time. Overall, the more clients there are, the better the model performance is, but the communication time is also relatively long.

5. Conclusion

In the information age, software plays a major role in social, economic, political, and other aspects. The development of software has brought great convenience to the people while improving work efficiency. Good software can ensure the normal operation of the work. When software quality problems occur, to a large extent, users will suffer from various losses. Software defect prediction technology can test whether there are defects in software, which is convenient for developers to correct in time and improve software quality.

This article addresses the issue of low quality participants affecting model performance in traditional federated learning aggregation methods. From the perspective of optimizing aggregation during the model parameter aggregation stage, a dynamically selected software defect prediction federated model aggregation method is proposed. Experiments were conducted on 12 project datasets, and the overall performance of the DS-SDP method proposed in this paper was improved by 0.071 compared to the traditional federated aggregation method FedAvg. Moreover, the experimental hypothesis is in line with the real scenario. Analysis and comparison were conducted on different client numbers, and it was found that the more clients there are, the better the model performance, but the communication cost will also increase. Overall, this method increases the robustness of the model while also improving its performance. In the future, research can be conducted through the following aspects:

(1) This article uses NASA datasets for experiments, which have limitations in terms of data volume and features. In future research, construct defect datasets from practical software applications, validate the methods proposed in this article, and make improvements.

(2) In practical scenarios, different users have different requirements for privacy protection and data availability, and federated learning provides unified privacy protection for all clients, which cannot meet personal needs. In the future, research will be conducted on how to allocate privacy budgets to achieve personalized privacy requirements for project parties.

Acknowledgment

This work is sponsored by the National Natural Science Foundation of China (62241209), the Natural Science Foundation of Xinjiang Uygur Autonomous Region (2022D01A225) and the Xinjiang Key Research and Development Program (2022B01007-1).

References

[1] Arya, A., & Malik, S. K. (2023). Software Fault Prediction using K-Mean-Based Machine Learning Approach. International Journal of Performability Engineering, 19(2).

[2] Rao, K. E., Rao, G. A., & Rao, P. S. (2022). A Weighted Ada-Boosting Approach for Software Defect Prediction using Characterized Code Features Associated with Software Quality. International Journal of Performability Engineering, 18(11).

[3] Wong W E, Horgan J R, Syring M, et al. Applying design metrics to predict fault‐proneness: a case study on a large‐scale software system[J]. Software: Practice and Experience, 2000, 30(14): 1587-1608.

[4] Li Y, Wong W E, Lee S Y, et al. Using tri-relation networks for effective software fault-proneness prediction[J]. IEEE Access, 2019, 7: 63066-63080.

[5] Chen X, Wang L P, etc. Review of cross project software defect prediction methods [J]. Journal of Computer Science, 2018,41 (01): 254-274.

[6] Li Y, Huang Z q, Wang Y, et al. A review of data-driven software defect prediction research [J] Journal of Electronic Science, 2017, 45 (4): 7

[7] Tang S, Huang S , et al. A Novel Cross-Project Software Defect Prediction Algorithm Based on Transfer Learning[J]. Journal of Tsinghua University: Natural Science Edition (English Edition), 2022, 27 (1): 17

[8] Chang C . Research about software defect priority prediction model based on AdaBoost-SVM algorithm. Nanjing : Nanjing University of Posts and Telecommunications,2013 (in Chinese）.

[9] Liu S h. Research on Software Defect Prediction Method Based on Cost Sensitive Learning [D]. Jilin University,2022.DOI:10.27162/d.cnki.gjlin.2022.006726.

[10] Bennin K E , Keung J , Phannachitta P , et al. MAHAKIL: Diversity Based Oversampling Approach to Alleviate the Class Imbalance Issue in Software Defect Prediction[J]. IEEE Transactions on Software Engineering, 2018:1-1.

[11] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," IEEE Transactions on Neural Networks,vol. 22, Feb. 2011.

[12] Nam J, Pan S J, Kim S. Transfer defect learning[C]//2013 35th international conference on software engineering (ICSE). IEEE, 2013: 382-391.

[13] Ni C, Chen X, et al.Cross project defect prediction method based on feature transfer and instance transfer [J] Journal of Software, 2019, 30 (5): 22.

[14] Peters F , Menzies T , Gong L , et al. Balancing Privacy and Utility in Cross-Company Defect Prediction[J]. IEEE Transactions on Software Engineering, 2013, 39(8):1054-1068.

[15] Peters F , Menzies T , Layman L . LACE2: Better Privacy-Preserving Data Sharing for Cross Project Defect Prediction[C]// IEEE/ACM IEEE International Conference on Software Engineering. IEEE, 2015.

[16] Li Z , Jing X Y , Zhu X , et al. On the Multiple Sources and Privacy Preservation Issues for Heterogeneous Defect Prediction[J]. IEEE Transactions on Software Engineering, 2017:1-1.

[17] Zhang Y, Research on Heterogeneous Software Defect Prediction Method Based on transfer learning [D]. Harbin University of Technology, 2021. DOI: 10.27063/d.cnki.ghlgu.2021.000726.

[18] Yang Q, Liu Y, Chen,etal. Federal Learning Practice [M] Beijing: Electronic Industry Press, 2021:1-323

[19] Feng J, Research on Secure Aggregation Methods for Privacy Data in Federated Learning [D]. Guangzhou University, 2022. DOI: 10.27040/d.cnki.ggzdu.2022.000884

[20] Wang A,et al. Heterogeneous Defect Prediction Based on Federated Transfer Learning via Knowledge Distillation[J]. IEEE Access, 2021, PP(99):1-1.

[21] Xu C Y, Ge L N, etal. Federated learning method based on differential privacy protection knowledge transfer [J/OL]. Computer Application Research: 1-9 [2023-04-04]. DOI: 10.19734/j.issn.1001-3695.2022.12.0813.

[22] Chen Y,Lu W , Wang J , et al. Federated Learning with Adaptive Batchnorm for Personalized Healthcare[J]. arXiv e-prints, 2021.

[23] Wu X , Zhang Y , Shi M , et al. An adaptive federated learning scheme with differential privacy preserving[J]. Future Generation Computer Systems, 2022, 127:362-372.

[24] Fang H, Qian Q. Privacy preserving machine learning with homomorphic encryption and federated learning[J]. Future Internet, 2021, 13(4): 94.

[25] Xin B, Geng Y, Hu T, et al. Federated synthetic data generation with differential privacy[J]. Neurocomputing, 2022, 468: 1-10.

[26] Luo C Y, Chen X B, etal. Federated ensemble algorithm based on deep learning [J]. Journal of Applied Science, 2022,40 (03): 493-510.

[27] SU H, CHEN H. Experiments on parallel training of deep neural network using model averaging[A/OL]. arXiv.org (2018-07-01). https://arxiv.org/abs/1507.01239.

[28] Andrew G, Thakkar O, McMahan B, et al. Differentially private learning with adaptive clipping[J]. Advances in Neural Information Processing Systems, 2021, 34: 17455-17466.

[29] Deng X, Ye W, Xie R, etal. A Review of Source Code Defect Detection Based on Deep Learning [J]. Journal of Software Science, 2023,34 (2): 625-654.

[30] Muneeb Ul Hassan, Mubashir Husain Rehmani, Jinjun Chen. Differential privacy in blockchain technology: A futuristic approach[J]. Journal of Parallel and Distributed Computing, 2020:50-74. DOI:10.1016/j.jpdc.2020.06.003.