World Scientific Research Journal Volume 10 Issue 11, 2024
ISSN: 2472-3703 DOI: 10.6911/WSRJ.202411_10(11).0001

Momentum Method and Top-k Optimisation Algorithm in Efficient and Secure Federated Learning Modelling

Shuhao Fan[1], Juntao Zhang[2,*], Yuelin Liu[1]

[1]Department of Mathematics and Applied Mathematics, Nanjing University of Science and Technology, Nanjing, China

[2]Department of Information and Computing Science, Nanjing University of Science and Technology, Nanjing, China

[*] Corresponding author: 1829789048@qq.com

Abstract

Federated learning allows multiple clients to collaboratively train models without sharing data, which is used to protect user privacy. Currently, homomorphic encryption is widely used in federated learning to prevent the leakage of model parameters. However, the large number of ciphertexts generated by homomorphic encryption imposes a huge overhead in communication transmission, which affects the training efficiency. Although various methods have been proposed to reduce communication costs through algorithm optimisation or decentralised training, there is a long way to go to solve the communication problem caused by homomorphic encryption. Yu et al. propose an efficient homomorphic encryption secure federated aggregation framework (ESFL) based on the quantisation of gradient pruning. We firstly simplify the ESFL framework by retaining the batch processing of the gradient of the candidate indexes in it, and the Top-k Gradient Sparsification for model gradient screening, which reduces the number of gradients to be transmitted, and then we apply momentum to refine the user model updates and accelerate convergence. It is shown that after the improvement of the gradient selection algorithm, our proposed method has a high-efficiency improvement while ensuring data security.

Keywords

Federated Learning; Top-k Gradient Sparsification; Batch Processing; Homomorphic Encryption; Momentum Method.

1. INTRODUCTION

With the occurrence of various incidents of data privacy leakage, people are paying more and more attention to data privacy and security, and various regulations have been introduced at home and abroad to protect data privacy and security. However, this makes it difficult for AI technologies relying on big data to obtain large training datasets to train high-quality models, thus hindering their development to a certain extent. As a result, the data that cannot be shared gradually forms a data silo, making it difficult to play a greater role.

To solve the above problems, in 2017 McMahan proposed a meet privacy protection and data security a viable solution called Federated Learning (McMahan 2017). [1] The federated learning technique simply means that in performing machine learning, multiple clients collaborate in training models without sharing data. Each client's local dataset is independent of each other, trains the model locally computes updates, and then sends these updates to a central server for aggregation, thus making it impossible for a malicious attacker to directly

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202411_10(11).0001

access the local dataset, which safeguards the privacy of the dataset to a certain extent.The process of federal learning is shown in Figure 1.

Since the concept of federated learning was introduced, it has rapidly gained extensive academic attention and research, Bonawitz, K. et al. (2017)[2] proposed a secure aggregation protocol in 2017 to ensure the privacy of model updates in federated learning. Gentry, C. (2009)[3] The theory of fully homomorphic encryption using ideal lattices published in 2009 laid the foundation of homomorphic encryption and provided theoretical support for subsequent applications of homomorphic encryption in federated learning. Fang, H. K. and Quan, Q. et al. (2021)[4] proposed a multi-party privacy-preserving machine learning framework called PFMLP, which combines partial homomorphic encryption and federated learning and mainly protects the privacy of the learning party by transmitting the gradient of encryption. However, this research direction faces many threats and challenges, especially the obvious lack of communication efficiency. For this reason, many scholars have proposed different improvement strategies, including optimising the federated learning algorithm, compressing model updates, and adopting a hierarchical training architecture. McMahan et al. [1] proposed the FedAvg algorithm for low-bandwidth environments.

Allows clients to execute SGD locally multiple times and then exchange model updates with a central server, thus reducing the number of communication rounds and achieving the same accuracy of model training. Chengliang Zhang et al. [5] proposed BatchCrypt, a systematic solution for cross-shaft FL, which greatly reduces the encryption and communication overhead caused by homomorphic encryption by encoding a batch of quantised gradients into a long integer and encrypting it at once. Pengchao Han et al.(2020) [6] proposed a fairness-aware applied gradient sparsification method with a new online learning formulation and algorithm to automatically determine the near-optimal communication and computation trade-offs controlled by the gradient sparsity. However, the communication overhead on federated learning based on homomorphic encryption is still very large, to address this problem, this paper proposes a new idea approach.

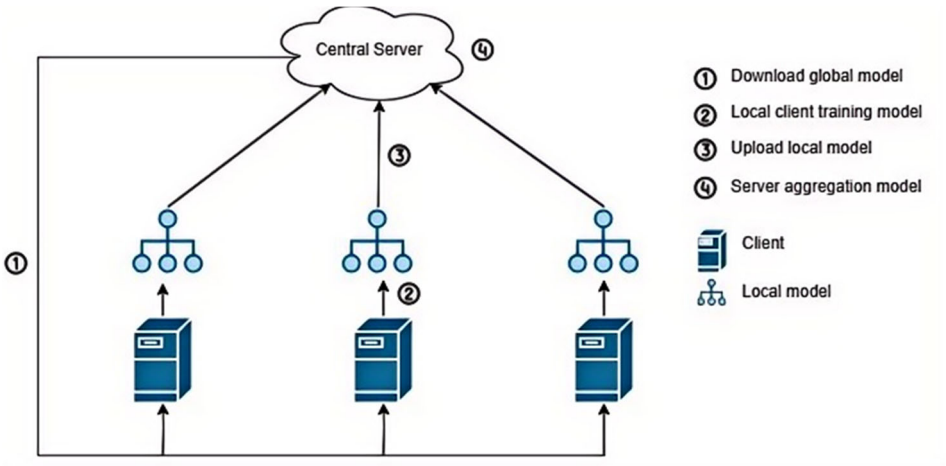

Figure 1. Process of federal learning

The main contributions of this paper are as follows:

We make improvements on the widely used gradient compression technique top-k gradient selection algorithm by using the ideas of number structure and heap sorting to reduce the number of gradients that need to be updated, thus reducing the communication overhead. Specifically, the traditional top-k sparsified gradient aggregation algorithm, which exchanges the largest k values in the gradient computed by each client, but after the aggregation of the

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202411_10(11).0001

array of candidate indexes, due to the inconsistency in the selection of gradient indexes by each client, it will result in the number of gradients to be uploaded in the end after the aggregation to be greater than K. Especially when the number of clients increases, and the increase in the volume of data leads to the increase in the inconsistency in the indexes selected. Especially when the number of clients increases and the inconsistency of index selection increases due to the increase in the amount of data, the expansion of the number of gradients to be uploaded is especially obvious. For this reason, we use the idea of a tree structure (Shaohuai Shi 2019) [16] to control the number of gradients to be updated at the end at K, to prevent the number from expanding.

When the client decrypts and then trains and obtains a local gradient vector, we apply momentum to refine the update instead of just uploading the current gradient. Convergence is accelerated and oscillations are reduced by combining the current gradient with previous gradients. Specifically, momentum introduces a parameter, usually denoted by the Greek letter β, which determines the effect of the previous gradient on the current update. In this way, the update not only relies on the current gradient but also takes into account the historical gradient, making the optimisation process smoother, thus enhancing the robustness of the model in the face of noisy or unstable data and mitigating possible poisoning attacks. In this way, momentum can effectively improve the efficiency and stability of training.

2. RELATED WORK

2.1. Homomorphic Encryption

Phong et al. (2018)[7] proposed a deep learning framework based on additive homomorphic encryption, which is capable of model training while ensuring data privacy. However, the high computational complexity of homomorphic encryption may lead to a significant increase in training time. Cheng, K et al. (2019)[8] proposed a lossless privacy-preserving tree enhancement system called SecureBoost for federated learning. The system performs entity alignment under privacy-preserving protocols and constructs boosted trees across multiple parties with an encryption strategy to support datasets with the same user samples but different feature sets. It protects the information of each private data provider while maintaining the same accuracy as non-privacy-preserving methods. Ma et al. (2021)[9] proposed a multi-key homomorphic encryption protocol, xMK-CKKS, which protects the data privacy of each participant by using different keys while enabling cryptographic computation and model aggregation of data. In addition, the study proposes an efficient key management mechanism that improves the scalability and practicality of the system, making the model training process more efficient while protecting privacy.

2.2. Top-K Gradient Selection

Strom(2015) [10] Significantly compresses subgradients by setting gradient elements whose absolute value exceeds a threshold to 1 to select the upload gradient. Also, 1-bit quantisation of the gradient elements as well as compensation of gradient residuals are used to reduce the bandwidth significantly. Aji et al.(2017) [11] Mapping a certain percentage of minimum updates to zero and then exchanging sparse matrices speeds up the distributed stochastic gradient descent by exchanging sparse updates, where the minimum update threshold uses a single threshold of absolute value. Dryden et al. (2016)[12] proposed adaptive quantisation using a fixed proportion of the gradient updates to be sent for each minimum batch by determining positive and negative thresholds, which improves on the shortcomings of Strom's threshold selection difficulties. Han et al. (2020)[14] proposed a fairness-aware bi-directional top-k application gradient sparsification method, where the sparse gradient vector consists of k elements derived from the original gradients of all clients, uplink (client to server) and

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202411_10(11).0001

downlink (server to client). The method of aggregating the sparsified gradients after each local update step is also used to reduce the communication overhead.

3. THEORETICAL KNOWLEDGE

3.1. Homomorphic Encryption

Homomorphic Encryption (Homomorphic Encryption) refers to the original data after homomorphic encryption, the ciphertext obtained by a specific operation, and then the results of the calculation and then homomorphic decryption of the plaintext obtained by the original plaintext data is equivalent to the original plaintext data directly for the same calculation of the data results obtained.

Encryption Steps:

1. (Local) Generate a pair of public and private keys, public key pub and private key priv, public key for encryption and private key for decryption;

2.(Local) Encrypt m1 and m2 using public key pub to obtain $E_{pub}(m_1)$ and $E_{pub}(m_2)$ respectively;

3. (Third party) Use the Add_{pub} function for $E_{pub}(m_1)$ and $E_{pub}(m_2)$, i.e. $Add_{pub}(E_{pub}(m_1), E_{pub}(m_2))$;

4. (Local) Decrypt $Add_{pub}(E_{pub}(m_1), E_{pub}(m_2))$ using private key priv, i.e. $D_{\text{priv}}(Add_{pub}(E_{pub}(m_1), E_{pub}(m_2)))$;

$D_{\text{priv}}(Add_{pub}(E_{pub}(m_1), E_{pub}(m_2)))$ is then equal to $m_1 + m_2$. The third party implements m1 and m2 in an encrypted state to do the addition operation by step 3 above.

Common homomorphic encryption algorithms.

(1) Paillier's algorithm

Paillier's algorithm, published by French cryptographer Paillier, is a homomorphic encryption algorithm that satisfies addition, based on the hard problem of composite residue classes.

3.2. Gradient Selection Algorithm

The Top-k gradient selection algorithm is an optimisation algorithm designed to select the top-k elements of the gradient to update the parameters at each iteration step, rather than using all gradient elements. This algorithm is typically used to deal with large-scale datasets or parameter spaces to reduce computational and storage overheads and to speed up the optimisation process.

The advantage of the Top-k gradient selection algorithm is that it can significantly reduce computational and storage overheads, especially when dealing with large-scale datasets or parameter spaces. However, choosing an appropriate k-value is crucial; a small k-value may cause the algorithm to converge too slowly, while a large k-value may increase the computational and storage overheads. Therefore, a suitable k-value needs to be chosen according to the specific problem and resource constraints in practical applications.

Initialisation parameters

gradient selection algorithm：

for each iteration.

Forward propagation

Calculate the gradient for all parameters

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202411_10(11).0001

Sort the gradients by absolute magnitude

Select the first k gradients with the largest absolute value

Update only the parameters corresponding to these k gradients

The rest of the parameters remain unchanged or are set to zero

4. AN EFFICIENT FEDERAL LEARNING FRAMEWORK BASED ON HOMOMORPHIC ENCRYPTION

4.1. Description of the Framework

The system model for this experiment consists mainly of a parameter server and multiple clients involved in federated learning. The functions performed by each entity involved in this system model are described in detail below:

Key Generation Center KGC: The Key Generation Centre generates, manages and distributes keys for the system.

Parameter Server PC: The parameter server is mainly responsible for receiving the ciphertext data sent by each participant and performing secure aggregation operations on all the ciphertext data, and then returning the correct results to each participant until a more ideal model is trained.

Client PN: Clients are the demanders of the whole federated learning mechanism, each client owns a portion of the private dataset in its local area, and hopes to obtain a more accurate and ideal model through the federated learning mechanism by co-training with the rest of the trainers. Therefore, each client will first train with its small-scale dataset locally, and then encrypt the model parameters with the help of the encryption scheme in this paper and upload them to the parameter server. Finally, the parameter server decrypts the securely aggregated data and performs subsequent parameter updates until a more accurate model is trained.

Our Efficient Federal Training process goes through four main phases:

1.Key Generation

Key Generation Centre KGC first generates the key pair i.e. public key pk and private key sk and sends the public key to each PC and PN and the private key to each PN for decrypting the data. In this way, the purpose of providing privacy protection is achieved.

2. Local model update

In the client local training phase, in the nth round, each client selectively uploads a new round of training gradient parameters, where two optimisation treatments are applied, the first is to apply the method of momentum(El-Mhamdi, E. M., Guerraoui, R., & Rouault, S. 2020) [16] to update the gradient, which improves the efficiency from the perspective of model training, and the second is to use the improved top-k gradient selection algorithm, to facilitate the encrypted transmission of the information, the object of transmission here is not the gradient information itself, but the corresponding index array of the gradient selected by the top-k, which improves the efficiency from the perspective of communication transmission. Finally, the client encrypts and uploads the index array to the central server using the public key.

3. Security Aggregation

After the central server receives the ciphertext of the index array from each client, it directly performs weighted aggregation of the ciphertext and sends the aggregated ciphertext back to each client, and each client decrypts the ciphertext using the private key to obtain the aggregated index array, instead of uploading all the gradient information, it only uploads the gradient information that corresponds to the index array to the server, and the server performs the aggregation based on the uploaded gradient information, and applies the momentum

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202411_10(11).0001

method to complete a round of global model updating, and finally sends the global model to each client, which updates the local model according to the global model constructed by the server.

4. Candidate index merging (Yu, S. X., & Chen, Z. 2023)[15]

To further reduce the communication overhead, it is necessary to merge the candidate indexes of the index arrays obtained by top-k selection for each client. Firstly, to prevent the server from overflow phenomenon in the process of merging index arrays, each client will quantise the gradient index value in the array into the number of binary bits of the client, and secondly, the batch processing will be used to batch splice the quantised index arrays into large integers. And the server performs homomorphic addition on the uploaded ciphertext, the client finally decrypts the ciphertext and inverse quantisation to determine the gradient that needs to be uploaded.

The four main stages of the efficient federation training process have been described above, and it is divided into the following steps in the concrete implementation: (1) Each client, after a certain number of rounds of model training, selects the appropriate gradient according to the Top-K algorithm, and sends the array of candidate gradient indexes to the parameter server after quantisation, batch processing, and encryption in turn. (2) The parameter server merges all the received encrypted gradient index arrays and sends them to each client. (3) Each client decrypts the ciphertext of the aggregated candidate indexes according to the aggregated candidate index ciphertexts sent by the parameter server, decrypts the ciphertexts with private key and inverse quantisation to get the aggregated candidate index arrays, and then transmits only the gradient information that is needed to be transmitted to the parameter server accordingly. (4) The parameter server uploads the candidate gradients by secure aggregation combined with the momentum method to obtain the global model, encrypt it and send it to each client, which decrypts it and applies the momentum method to update the local model parameters (see Figure 2).

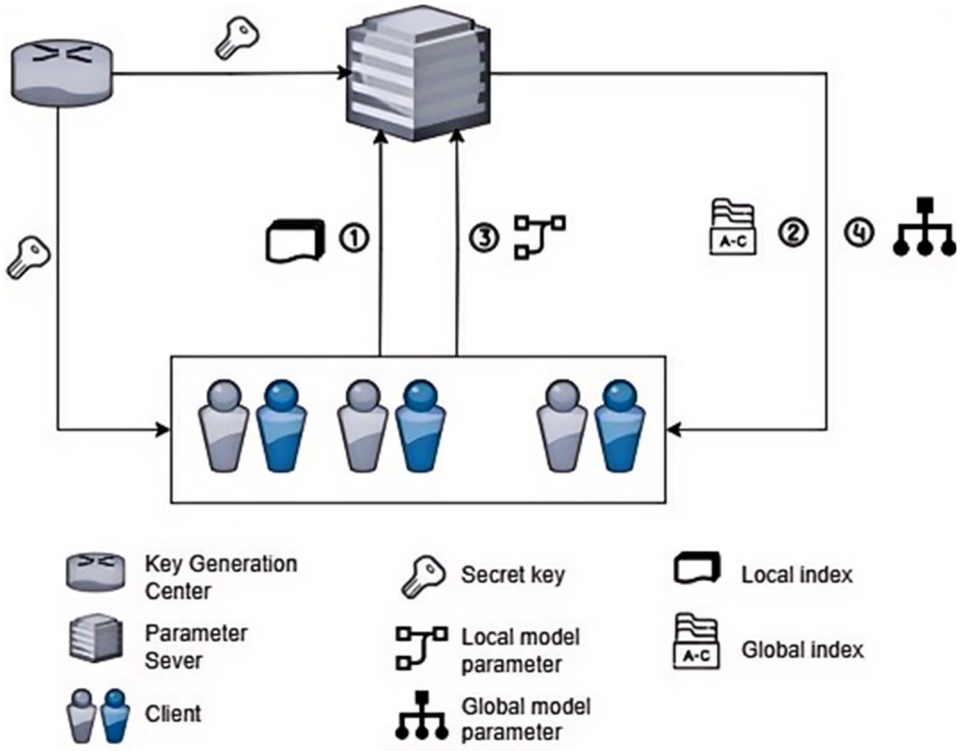

Figure 2. System molded relief map

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202411_10(11).0001

5. EXPERIMENTAL PERFORMANCE EVALUATION

5.1. Experimental Setup

The CIFAR10 dataset is used for the experiments in this paper. In this paper, comparative experiments are conducted to verify the effectiveness of the scheme to reduce the communication overhead based on the optimised top-k gradient selection algorithm. To demonstrate the effectiveness of the proposed optimisation scheme, the gradient selection algorithm is varied in the same experimental setting and the CIFAR10 dataset is used as a training model for the ResNet network, which is a deep neural network that introduces a residual module. The homomorphic encryption algorithm used in this paper is paillier.

5.2. Basic Methods

In this paper, the experimental model we build is based on the ESFL framework proposed by Yu et al. (the simplified ESFL framework, which omits the compression processing of gradient information and retains the quantisation of the gradient index, batch processing). Firstly, the improvement of encryption efficiency by batch processing in the secure candidate indexing algorithm of ESFL is analysed and verified. Secondly, the effect of the improvement of the model convergence speed before and after the application of the Momentum Method is verified through comparative experiments, so that the number of communications can be reduced. Next, the effectiveness of the improved Top-k Gradient Sparsification is verified by comparing it with the originally adopted traditional fixed-proportional Top-k Gradient Sparsification, which proves to reduce the single communication bandwidth. Finally, combining the optimisation of the communication bandwidth, and the number of communications by the above methods, the final optimisation effect on the communication overhead is analysed.

5.3. Data Sets

The CIFAR10 dataset is an image dataset widely used in the field of computer vision and consists of 60,000 colour (RGB) images of 32x32 pixels in 10 categories, with each category containing 6,000 images. The CIFAR-10 dataset is usually provided in the form of a binary file, with each training batch and test batch containing image data and corresponding labels. Each image data is a 32x32x3 array, i.e., a 32x32 pixel RGB image with RGB values ranging from 0 to 255 for each pixel. The labels are given as integers ranging from 0 to 9, corresponding to the 10 categories in the dataset.

5.4. Experimental Performance

By implementing batch processing for candidate indexing, we successfully encrypt multiple quantised index updates into a single ciphertext, which significantly reduces the amount of gradients required for each round of communication, and greatly reduces the number of encryption and decryption times for homomorphic encryption, to reflect this, we react to the reduction in computational consumption by comparing the training time of each of the first 20 rounds of iteration, and the results are shown in Figure 3.

In this experiment, the number of clients is 3, the encryption algorithm is paillier, the local training model is resnet, and the dataset is cifar10. From the figure, it can be seen that the length of a single round of training fluctuates around 31.9s before the modification, whereas the length of a single round of training fluctuates around 29.8s after the modification, and the communication efficiency is improved by about 6.58%, which reflects the effect of the batch optimisation efficiency.

Next, we apply the Momentum Method approach to our model and the results are shown in Figure 4:

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202411_10(11).0001

From the results, it can be seen that the convergence speed of the model is improved after the introduction of the momentum method. The momentum approach accelerates the convergence process by accumulating historical gradients, which helps the model to maintain consistency in the update direction when facing flat regions or local minima, thus reducing the number of communications for model updates. Experiments show that, in the experimental environment where the number of edge nodes is 3, the training model is resnet, and the dataset solution is the small dataset cifar10, the number of communications is reduced by about 20% compared with that in the experimental environment where the number of edge nodes is 3, the training model is resnet, and the dataset solution is the small dataset cifar10. The number of communications is reduced by about 20% until the loss function converges to relative stability.

Then, while reducing the computational overhead and accelerating the convergence, we improve on Top-k Gradient Sparsification, and we reflect the communication bandwidth of the gradient information that finally needs to be transmitted in terms of the number of selected gradients in the array of candidate gradient indexes after aggregation before and after the improvement. We perform comparison experiments with the number of clients 25,50,75,100 respectively:

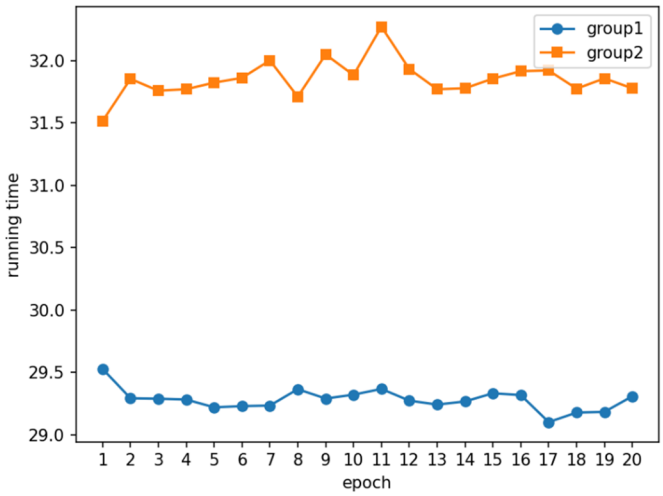

Figure 3. Comparison of Running Times

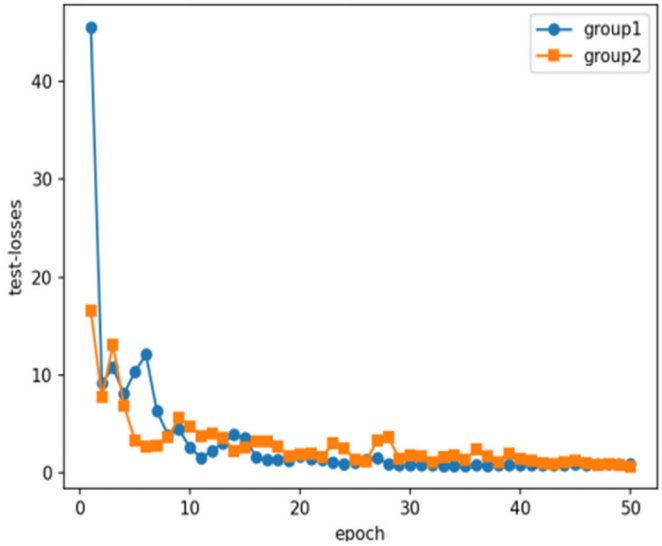

Figure 4. Comparison of Convergence speed

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202411_10(11).0001

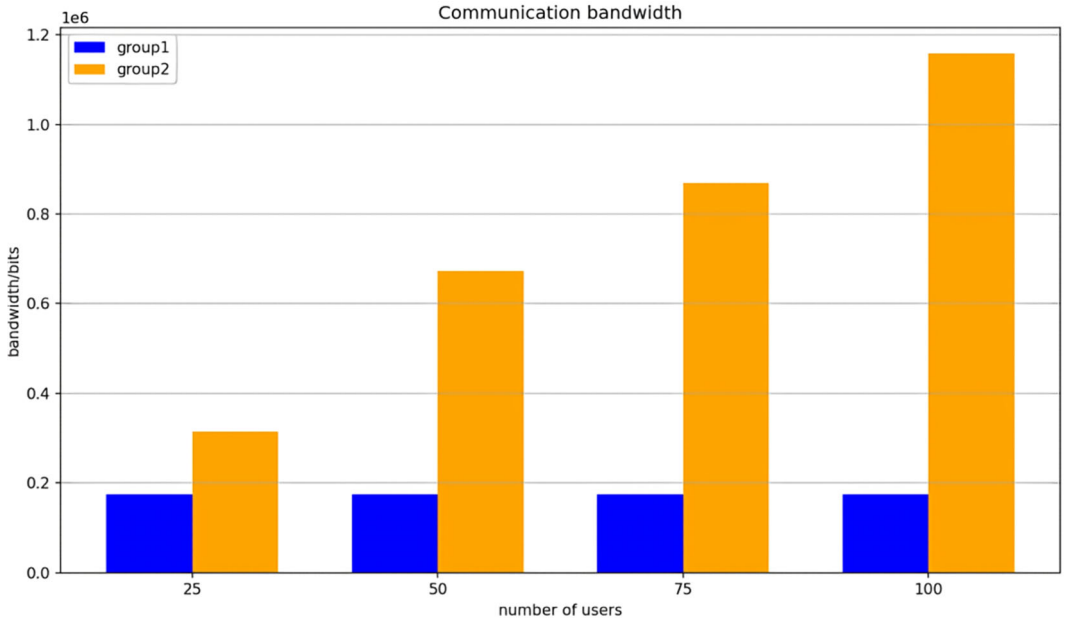

Figure 5. Communication bandwidth

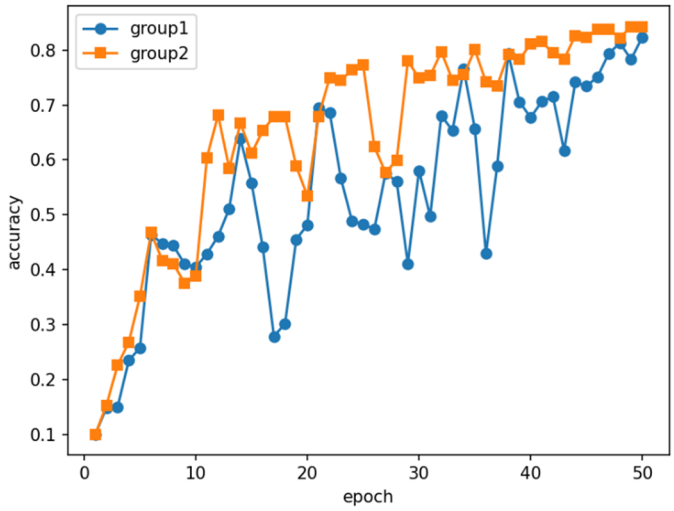

Figure 6. Comparison of Accuracy

From Figure 5 where the horizontal axis is the number of clients and the vertical axis is the number of bits.,it can be seen that the number of gradients to be transmitted after aggregating the index arrays is generally inflated. Especially with the increase in the number of clients, the degree of quantity expansion is not negligible with the increase of the number of clients, so after the change, since the number of gradients selected in the final candidate index is controlled in a fixed proportion, the number of gradients to transmitted is reduced, and with the increase of the number of clients, the effect of this reduction is even stronger, with 25 clients, the degree of compression is about 44.454%, and with 50 clients, the degree of reduction reaches 74.043%.

Finally, we analyse the model training accuracy after making these changes,the comparison of accuracy is shown in Figure 6.

From Figure 6,The changes in model accuracy during the first 50 rounds of iterations and the accuracy values at final convergence show that the model accuracy ultimately fluctuates within the 0.75-0.85 interval, and the loss of accuracy is within acceptable limits.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202411_10(11).0001

6. CONCLUSION

From the above experiments, we have analysed the efficiency enhancement capability of our scheme for homomorphic encryption-based secure federated learning framework in terms of computational consumption, number of communications, and communication bandwidth, and also observed the model accuracy, and the loss of accuracy is within the acceptable range and meets our expectations.

We also analysed the Deficiencies and problems:

1). The above experiments are only conducted in a lightweight experimental setup based on our simplified ESFL framework, which has limitations, and the data obtained can only reflect certain reference values.

2). Computational overhead: While the communication overhead has been reduced, the computational overhead of homomorphic encryption itself is still high, which may affect the application of federated learning on resource-constrained devices.

3). Heterogeneity problem: In practical applications, clients may have different computational power and communication bandwidth, which leads to heterogeneity problems among clients. This heterogeneity may affect the convergence speed and final performance of the model.

4). Momentum parameter tuning: Parameters (e.g., momentum coefficients) in the momentum approach need to be carefully tuned to accommodate different data sets and models. Inappropriate parameter settings may lead to degradation of model performance.

5). Model generalisation capability: While accelerating convergence, there is a need to ensure that the generalisation capability of the model is not compromised. Over-reliance on momentum may lead to model overfitting on the training set.

ACKNOWLEDGEMENTS

This paper is funded by the JiangSu Undergraduate Training Program for Innovation and Entrepreneurship.

Item number: 202310288177Y.

REFERENCES

[1] McMahan, B., Moore, E., Ramage, D., Hampson, S., & Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. AISTATS, pp 1273-1282.

[2] Bonawitz, K., Ivanov, V., Kreuter, B., et al. (2017). Practical Secure Aggregation for Privacy-Preserving Machine Learning. Google Inc., Mountain View, CA, USA.

[3] Gentry, C. (2009). Fully homomorphic encryption using ideal lattices. STOC, pp 169-178.

[4] Fang, H., & Qian, Q. (2021). Privacy Preserving Machine Learning with Homomorphic Encryption and Federated Learning. Future Internet, 13(4), 94. http://doi.org/10.3390/fi13040094.

[5] Zhang, C., Li, S., Xia, J., Wang, W., Yan, F., & Liu, Y. (2020). BatchCrypt: Efficient Homomorphic Encryption for Cross-Silo Federated Learning. USENIX ATC.

[6] Han, P., Wang, S., & Leung, K. K. (2020). Adaptive Gradient Sparsification for Efficient Federated Learning: An Online Learning Approach. ICDCS, pp 300-310.

[7] Phong, L. T., Aono, Y., Hayashi, T., Wang, L., & Moriai, S. (2018). Privacy-Preserving Deep Learning via Additively Homomorphic Encryption. IEEE Transactions on Information Forensics and Security, 13, 1333-1345.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202411_10(11).0001

[8] Cheng, K., Fan, T., Jin, Y., Liu, Y., Chen, T., & Yang, Q. (2019). SecureBoost: A Lossless Federated Learning Framework. IEEE Intelligent Systems, 36, 87-98.

[9] Ma, J., Naas, S., Sigg, S., & Lyu, X. (2021). Privacy-preserving federated learning based on multi-key homomorphic encryption. International Journal of Intelligent Systems, 37, pp 5880-5901.

[10] Strom, N. (2015). Scalable distributed DNN training using commodity GPU cloud computing. arXiv preprint arXiv:1506.03478.

[11] Aji, A. F., & Heafield, K. (2017). Sparse Communication for Distributed Gradient Descent. EMNLP,pp 440-445.

[12] Dryden, N., Moon, T., Jacobs, S. A., et al. (2016). Communication Quantization for Data-Parallel Training of Deep Neural Networks. MLHPC, pp 1-8.

[13] Alistarh, D., Li, J., Tomioka, R., et al. (2016). QSGD: Randomized Quantization for Communication-Optimal Stochastic Gradient Descent. CoRR, abs/1610.02132.

[14] Han, P., Wang, S., & Leung, K. K. (2020). Adaptive Gradient Sparsification for Efficient Federated Learning: An Online Learning Approach. ICDCS, pp 300-310.

[15] Yu, S. X., & Chen, Z. (2023). An Efficient and Secure Federated Learning Aggregation Framework Based on Homomorphic Encryption. Journal of Communications, 44(1), 14-28. http://doi.org/10.11959/j.issn.1000-436x.2023015.

[16] El-Mhamdi, E. M., Guerraoui, R., & Rouault, S. (2020). Distributed Momentum for Byzantine-resilient Learning. arXiv preprint arXiv:2003.00010.

[17] Shaohuai Shi, Qiang Wang, Kaiyong Zhao, Zhenheng Tang, Yuxin Wang, Xiang Huang, Xiaowen Chu. (2019) A Distributed Synchronous SGD Algorithm with Global Top-k Sparsification for Low Bandwidth Networks. In: Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS). IEEE, arXiv:1901.04359v2