Approaches on improving privacy and communication efficiency of FedFTG

Dongjun Geng[1,3] and Dingxiang Wang[2]

[1]School of Electronic Information Engineering, Beijing Jiaotong University, Weihai, 264200, China
[2]School of Information Technology, Shenyang Institute of Engineering, Shenyang, 110000, China

[3]20721045@bjtu.edu.cn

Abstract. The FedFTG plug-in can effectively solve the problem of knowledge forgetting caused by the server-side direct aggregation model in Federated Learning. But FedFTG runs the risk of compromising customer privacy, as well as additional transmission costs. Therefore, this paper introduces methods to enhance the privacy and communication efficiency of FedFTG, they are: Mixing Neural Network Layers method which can avoid various kinds of inference attack, Practical Secure Aggregation strategy which uses cryptography to encrypt transmitted data; The Federated Dropout model which focuses on reducing the downward communication pressure, and the Deep Gradient Compression method that can substantially compress the gradient. Experimental results show that, MixNN can ensure the privacy protection without affecting the accuracy of the model; Practical Secure Aggregation saves the communication cost when dealing with large data vector size while protecting the privacy; Federated Dropout reduces communication consumption by up to 28×; DGC can compress the gradient by 600× while maintaining the same accuracy. Therefore, if these methods are used in FedFTG, its privacy and communication efficiency will be greatly improved, which will make distributed training more secure and convenient for users, and also make it easier to realize joint learning training on mobile devices.

Keywords: FedFTG, privacy, communication efficiency, federated learning.

1. Introduction

The FedFTG algorithm is a recently proposed federated learning algorithm that is designed to tackle non-i.i.d. data distribution in federated learning [1]. It combines the dynamic federated averaging algorithm and the Federated Newton Method to achieve high accuracy and fast convergence. While FedFTG has shown great potential in various applications, there is still room for improvement.

To address this, researchers have proposed several improvements and enhancements to the FedFTG algorithm. One such improvement is the use of a customized optimization solver that adapts to the heterogeneity of the participating clients [2]. Another improvement is the incorporation of local quasi-Newton methods to enhance the efficiency and accuracy of the algorithm. Additionally, the use of a variance reduction technique has been shown to improve the convergence speed and stability of the FedFTG algorithm [3].

The benefits of these improvements to the FedFTG algorithm are significant. The customized optimization solver can improve the performance of FedFTG in heterogeneous environments and improve the convergence rate. The use of local quasi-Newton methods can accelerate the convergence speed of the algorithm and enhance its accuracy. The variance reduction technique can reduce the variance of the gradients, which can stabilize the convergence and improve the overall performance of the algorithm.

Overall, the improvements and enhancements to the FedFTG algorithm offer a promising direction for future research in federated learning. These advancements can lead to faster convergence rates, increased accuracy, and improved stability, making the FedFTG algorithm more suitable for a wider range of applications in real-world scenarios.

2. The theoretical basis of the FedFTG

The federated learning security aggregation algorithm is a computational privacy protection technology that realizes the aggregation of model parameters in federated learning. It leverages cryptography to protect the privacy of parties while ensuring the accuracy and integrity of model parameters. The federated learning security aggregation algorithm has wide application value in the current artificial intelligence research field, and has a great role in promoting the protection of personal privacy, big data sharing and cloud computing.

2.1. Principle and general introduction of FedFTG

The data of horizontal federated learning is horizontally segmented, the data schema of the participants are consistent, with the same characteristics, which is the most mature direction of the current development of the most federated learning, the basic framework of horizontal federated learning is similar to the current common distributed machine learning framework, the main feature is the use of blockchain technology and homomorphic encryption technology in federated learning to achieve data privacy protection and interactive security.

Here are the main ideas and processes [4].1. Data privacy protection: Each participant encrypts local data through homomorphic encryption technology to achieve data privacy protection; 2. Blockchain storage: use blockchain technology to distribute the encrypted model parameters and gradient information to ensure that the data is not maliciously tampered with; 3. Secure aggregation: Homomorphic encryption technology is used to encrypt and aggregate the encrypted data of participants to protect the privacy of model parameters; 4. Verification and training: Other nodes in the blockchain network can verify the submitted data and implement the model parameter update and training process through the smart contract mechanism.

Federated learning security aggregation algorithms have been widely used in fields such as protecting personal privacy, big data sharing, and cloud computing. The effectiveness depends largely on factors such as the number of participants, the quality of network communication, and the complexity of the model. Nevertheless, studies have shown that federated learning secure aggregation algorithms can achieve high model accuracy and privacy protection in multiple domains. It contains machine learning, distributed, privacy protection is a three-in-one cross-technology of machine learning, distribution, and privacy protection. Unlike existing distributed machine learning, federated learning is mainly subject to the strict constraints of the distribution of raw data in different locations, and there can be no risk of leaking raw data, and privacy protection technology is the key to preventing leakage [5]. When the data provider puts the data into the safe for transmission, other federation members cannot open the box to see the real data, but they can complete the operations required for training on the data in the safe without unlocking it; When the training is completed, the data provider can take out the calculation results by opening the box. This ideal function requires us to implement it using special techniques, such as differential privacy, homomorphic encryption, and so on.

2.2. Problems in FedFTG

2.2.1. Privacy issues. If we just package and encrypt local model parameters and send them to the cloud for aggregation, this will expose the data and make it possible for attackers to obtain sensitive information. Therefore, the federated learning security aggregation algorithm adopts a combination of encryption and decryption, and cooperates with the encryption and decryption operations of model parameters to ensure data security. The federated learning security aggregation algorithm mainly adopts homomorphic encryption technology and multi-party secure computing methods [6]. At the same time, in the process of model aggregation, the algorithm will also add special noise to prevent attackers from constantly guessing to obtain the real model parameters. These technical measures greatly improve the reliability and security of the algorithm. In conclusion, federated learning secure aggregation algorithm is a powerful privacy protection technology, which can protect the privacy of participants while ensuring the accuracy and completeness of model parameters.

2.2.2. Communication efficiency problem. Because I think the traditional FedFTG algorithm is a synchronous update algorithm, that is, all clients need to upload the model/gradient information synchronously to update the server model. Therefore, every time the model aggregation is performed, the server side needs to accept most or all of the client's model data, which greatly increases the data communication pressure on the server side. Asynchronous federated learning means that the client uploads and updates the server-side model in turn. An asynchronous federated learning based on homomorphic encryption is proposed, in which the client first uses a third-party server to set the same homomorphic encryption public and private key, and uploads the public key to the federated training server [7]. Before each client training, obtain the latest encryption model from the server, and use the private key decryption to obtain the plaintext model to update the client model. Client trains locally to get gradients.

3. Improved approach for the communication efficiency problem of FedFTG

This article summarizes two approaches to the FedFTG privacy problem, namely Mixing Neural Network Layers and Practical Secure Aggregation, and presents experimental analyses of each of these two approaches.

3.1. Mixing Neural Network Layers

3.1.1. Working principle

Mixing Neural Network Layers is a new privacy-preserving service used in federated learning which is designed for protecting the privacy by defending other aggregation servers' inference attacks.

Only a proxy for corresponding traffic is requested when add MixNN into the system. Different from the traditional federated learning process, the parameter updates will be delivered to MixNN proxy before being sent to the aggregation server [8].

Then the public key of the enclave is used to secure the updates received. This step can make sure that no one else can read and process the original updates except for the MixNN proxy. Decryption is executed after updates are loaded in the enclave. And the parameter updates will be stored in different lists classified by different layers. Then the proxy picks one update in each corresponding list to write the message which includes the parameter updates. Finally, the message is delivered to the aggregation server [9].

3.1.2. Optimized result

3.1.2.1. No decrease on utility. MixNN is tested to be compared with noisy gradient widely which is another method to protect privacy [10,11]. The specific aspects for comparison are utility and privacy. The figure 1 shows how accuracy changes when the learning rounds grows. The model accuracy of traditional federated learning and MixNN both grow with the increasing of learning rounds. In addition, the average accuracy of noisy gradient decrease to 90% of the original value.

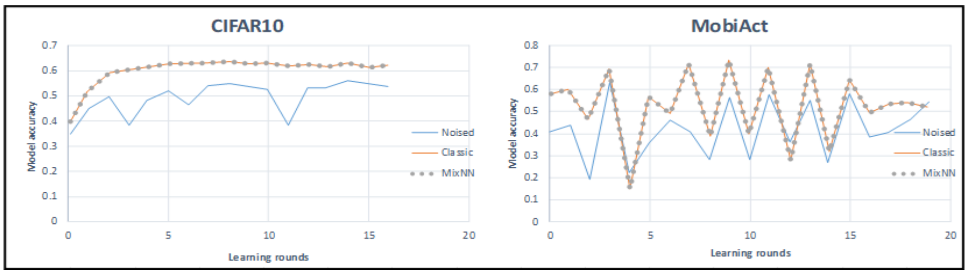


Figure 1. MixNN does not decrease the same utility while noisy gradient decreases it and more learning rounds are needed.

The figure 2 shows how cumulative distribution perform on the population of the participants in the 6th learning round. For all kinds of datasets, MixNN perform well on the accuracy of noisy gradient. The accuracy of MixNN (0.68) is 0.12 higher than the average value which is 0.56.

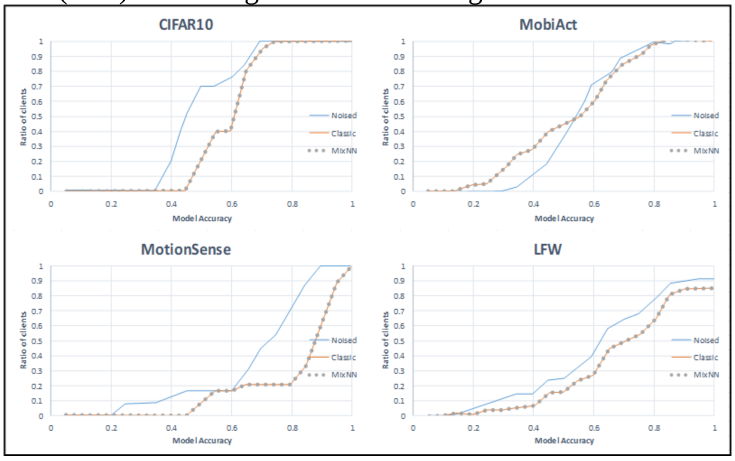


Figure 2. Noisy gradient has decline on all participants' the utility.

3.1.2.2. Prevent information leakage. The figure 3 shows how the accuracy of MixNN, Classical Federated Learning and Noisy gradient on datasets change when the learning rounds grow.

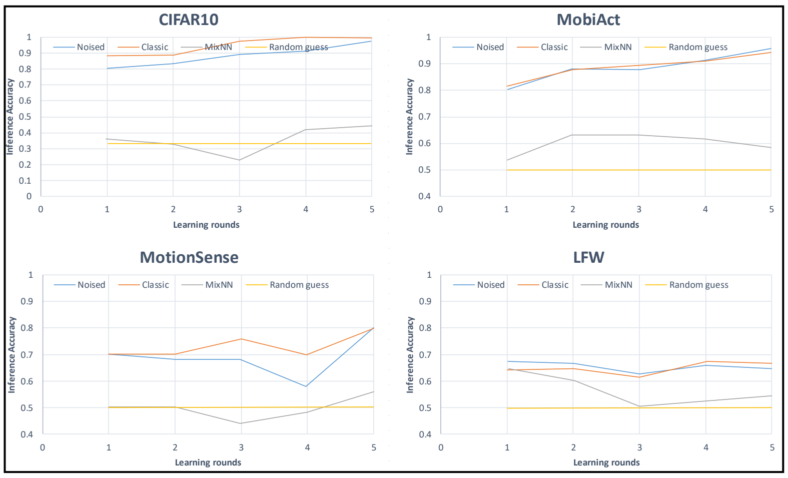


Figure 3. MixNN protect attribute security better than noisy gradient.

After 5 learning rounds, the accuracy is about 100% (CIFAR10), 80% (MotionSense), 94%(MobiAct), and 66%(LFW). And the success of preventing privacy leakage is shown. The inference accuracy is close to the random guess. MotionSense, MobiAct and LFW is about 0.5 while CIFAR10 has a smaller value which is 0.33. The accuracy increases when the learning rounds grow and finally stable. For all datasets, MixNN leak less information than other approaches.

3.2. Practical secure aggregation

3.2.1. Working principle. Practical Secure Aggregation is a protocol which gather the local models of the users instead of their real values. It can protect the privacy of the user's information. The message which contains the condition of global model is sent to the users by server. And then users will update it with their local datasets and deliver a masked model back to the server. All these models will be gathered and the sum of true models will be learned by the server. Finally, the server use these to update the global model.

Practical Secure Aggregation consists of some algorithms which are Secret Sharing, Key Agreement, Authenticated Encryption, Pseudorandom Generator and Signature Scheme. The protocol has four rounds and it operates between the communication of a server and n users. A vector will be input by the users. The server can communicate with the users through authenticated channels. Users can get out of the protocol when they stop delivering messages. Finally, the server will produce a correct output until the last round.

Drawback: There are very high requirements on the format of the user's input. If the input does not conform to a specific format or does not fall within the specified range it will result in an incorrect output [12].

3.2.2. Optimized result. The following figure show that the running time of each client increases when the clients and data vector entries increase. The ratio of dropout has a very insignificant effect on the results. Increasing the size of the data vector influence more prominently on the communication expansion factor than the impact of the change of client number. The communication expansion factor decreases when the data vector's size increases. The results figure 4 show that the cost of communication can be amortized well at the condition of the increasing size of the data vector.

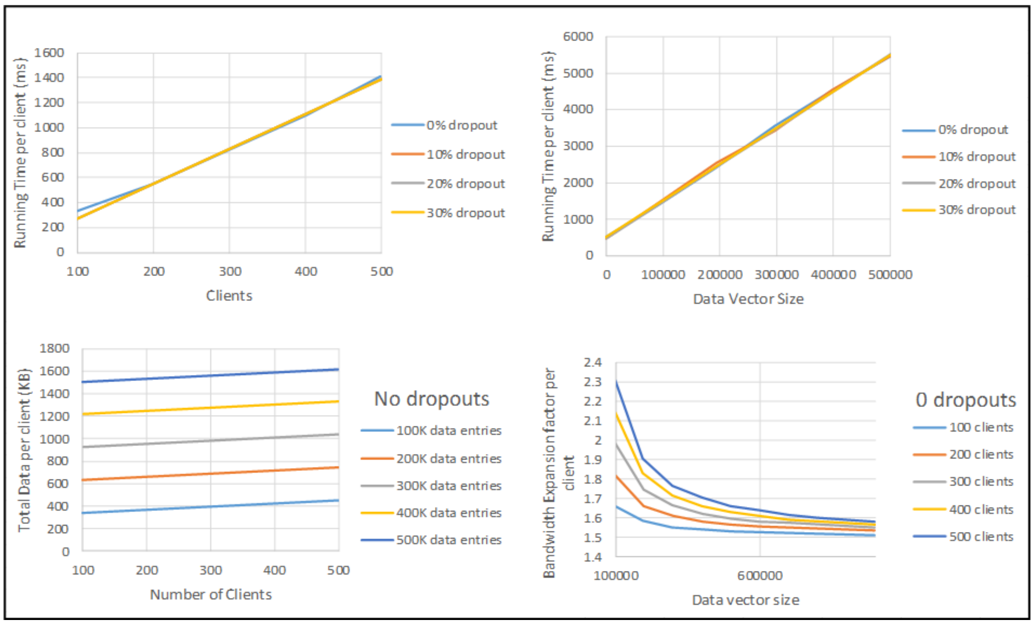


Figure 4. Client running time and data transfer costs, ignore communication latency, the points are the averages over 10 end-to-end iterations.

4. Improved approach for the communication efficiency problem of FedFTG

In addition to privacy, improvements to the FedFTG plug-in can also be seen in communication efficiency. Here we focus on two algorithms, the Federated Dropout and Deep Gradient Compression, both of which reduce communication consumption significantly while maintaining transmission accuracy.

4.1. Federated dropout

4.1.1. Working principle. Federated Dropout is an approach that concentrates mainly on reducing the cost of Federal Learning communications on downlink [13].

In the process of updating the local model using the Dropout method, what is transmitted in the channel is not the global model itself, but a smaller subset of the global model built on the Federated Dropout technique. If the lossy compression method is applied, the size of the communication model can be further reduced. The sub-model of the compressed global model is moved to the client side so that the local model updates made by the client are also based on the sub-model, which greatly reduces the communication consumption. The training process combined with these methods in Federated Learning is shown in figure 5.

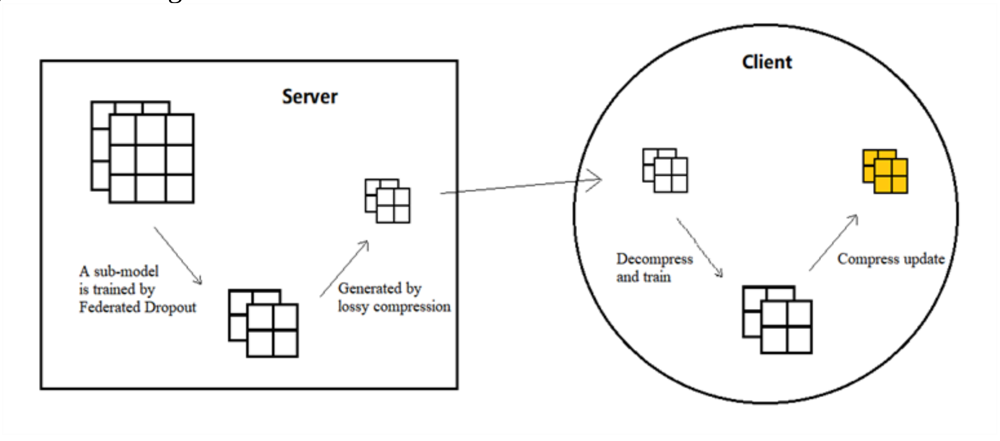

Server
A sub-model is trained by Federated Dropout
Generated by lossy compression
Client
Decompress and train
Compress update

Figure 5. Combined federated dropout and lossy compression in FL training.

The core of the traditional dropout algorithm is to randomly select a part of the neurons in the hidden layer to participate in the work [14]. All the networks share weights, so each training model is different. In order to effectively apply the traditional dropout algorithm to Federated Learning and reduce communication costs, the Federate Dropout removes a certain amount of activation at the full-connection layer to ensure that almost all the sub-models have the same structure.

It follows from this that in the Federate Dropout, the sub-model consists mostly of meaningful data. This means less and denser sub-model data is transmitted to the client, which can effectively reduce communication losses. At the same time, the local model updates that the client needs to transmit to the central server are reduced in size with Federated Dropout algorithmic compression. Further more, local models can now be trained with fewer gradient updates. Thus, using Federated Dropout further reduces the cost of local computing in federated learning.

4.1.2. Optimized result. Experimenting with the lossy compression strategy and the Federated Dropout integrated with the client-to-server compression approach [15], we can see that using them together can better reduce the communication consumption at the time of the transfer models. The best setting for the federated dropout rate (the ratio of neurons or the filters that are stored in each layer of the models) is 0.75, see figure 6, in which case, it saves 14× server-to-client communication and 28× client-to-server communication without losing model accuracy, while reducing local computing by 1.7×.

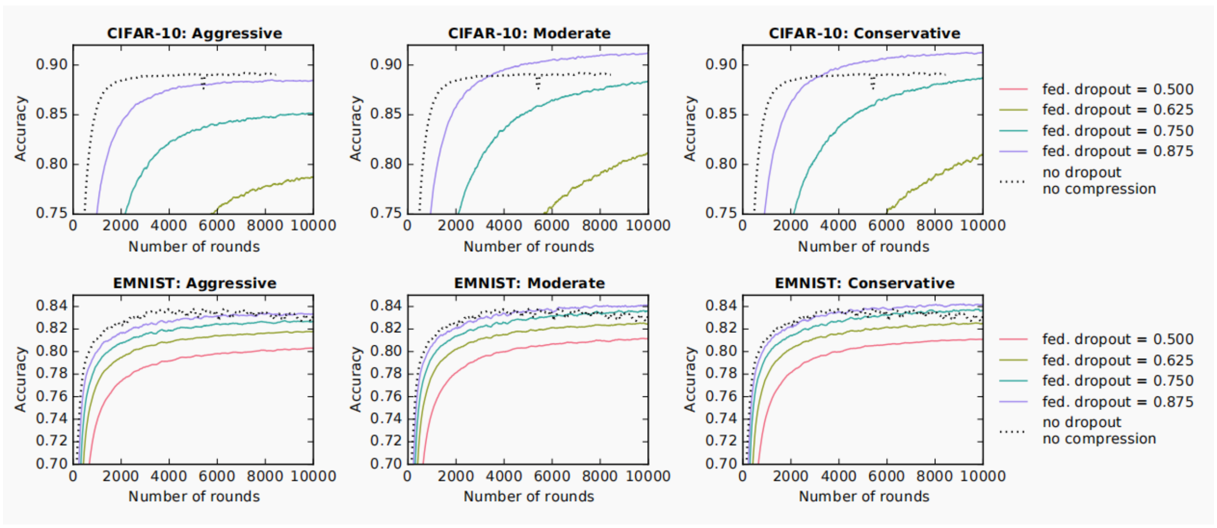


Figure 6. Experimental results on CIFAR-10 and EMNIST using the federated dropout and lossy compression strategy.

4.2. *Deep gradient compression*

4.2.1. Working principle. Deep Gradient Compression (DGC) is a method of reducing communication bandwidth consumption by compression the gradient.

Inspired by the Gradient Sparsification, DGC also only sends important gradient updates that are larger than the threshold. The remaining small gradients are accumulated locally until they become large enough to be transmitted.

The traditional Momentum SGD algorithm carries out a new update on top of the previous update direction, while fine-tuning the final update direction using the gradient of the current batch. If Momentum SGD is updated directly using the Gradient Sparsification method, the model convergence will slow down due to the lack of momentum coefficient before each loss function. Therefore, DGC adjusted the direct gradient dilution updating formula of Momentum SGD and introduced momentum correction to make the weight update normally.

To prevent Gradient explosion, the Local Gradient Clipping method is used here. Since DGC accumulates gradients independently at each node through iteration, it performs local Gradient clipping on the basis of Gradient clipping, before adding the current gradient Gt to the previous accumulation. If all N nodes are distributed identically, then the threshold scales by N − 1 / 2 (the proportion of the current node to the global threshold).

Because of the DGC delays updates for smaller gradients in the communication, by the time the smaller gradient updates are finally transmitted, they are out of date. Therefore, DGC inspired by momentum factor masking, uses the mask for accumulated gradients vk,t and momentum factor uk,t to mitigate the gradient momentum delay [16]. In addition, DGC adopts the warm-up training method to first apply a soft learning rate in the early stages of training, and then increase the gradient sparsity from a relatively small value exponent to the final value. These are conducive to improve convergence speed and model performance, preventing the influence of older data.

4.2.2. Optimized result. Experiments show that based on guaranteed accuracy or even improved accuracy, the compression ratio of DGC to the gradient is nearly 600 × and 277 × that of the baseline level, respectively for model AlexNet and model ResNet-50. Table 1 shows these experimental data in detail.

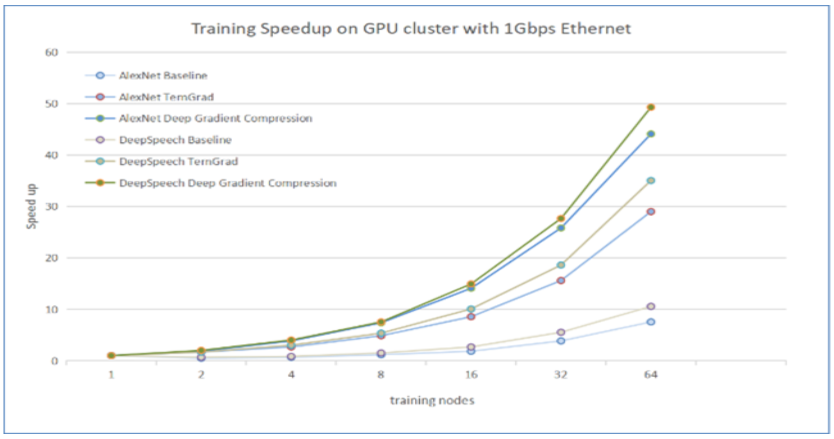
Table 1. Summary of experimental results(continue).

Model / Task	Training Method	Accuracy / Perplexity / Word Error Rate (WER)	Gradient Size	Compression Ratio
AlexNet	Baseline	58.17%	232.56 MB	1 ×
	Deep Gradient Compression	58.20% (+0.03%)	0.39 MB	597 ×
ResNet - 50	Baseline	75.96	97.49 MB	1 ×
	Deep Gradient Compression	76.15 (+0.19%)	0.35 MB	277×
Language Modeling on PTB	Baseline	72.30	194.68MB	1 ×
	Deep Gradient Compression	72.24 (-0.06)	0.42MB	462 ×
Speech Recognition on LibriSpeech	Baseline	9.45%	488.08MB	1 ×
	Deep Gradient Compression	9.06% (-0.39%)	0.74MB	608 ×

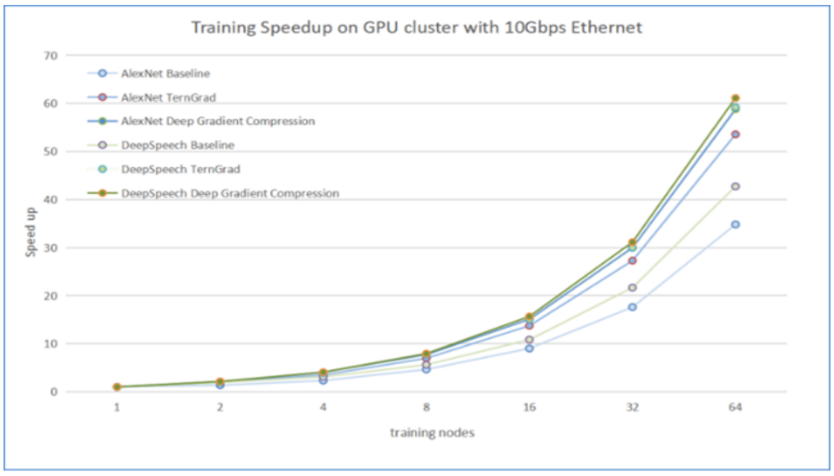
For language modelling and speech recognition, the DGC showed good recognition and matched the baseline accuracy very well. It also reduced the size of the gradient to 462 × that of the baseline, while reducing perplexity slightly. At the same time, the communication bandwidth of the model using DGC can be effectively compressed under the condition of noise, for 608 ×.

For the acceleration of training using DGC strategy on communication network, figure 7 shows that the training performance of the model without using DGC in 10Gbps Ethernet was at least 20% worse than that of the model with DGC strategy in 1Gbps Ethernet.

DGC greatly reduces the bandwidth required for communication. At the same time, it can effectively improve the possibility of distributed training for users in the environment of low communication bandwidth. For instance, when training with 1Gbps Ethernet, the DGC strategy can achieve about 50× speedup, which is about 5.5× faster than traditional training methods.



(a)



(b)

Figure 7. In real communication networks, DGC dramatically reduces communication consumption: (a)1Gbps (b)10 Gbps.

5. Conclusion

In this paper, the approaches to the problems of privacy and communication efficiency encountered in FedFTG applications are mentioned. First, two methods to improve the privacy of FedFTG are mentioned, which are Mixing Neural Network Layers (MixNN) and Practical Secure Aggregation. MixNN is a technology used for federated learning that defends against inference attacks from aggregation servers. Compared with other methods, MixNN can ensure that the utility is not reduced while preventing information leakage. Practical Secure Aggregation is a privacy protocol that protects user information. It collects data from users' local models rather than their actual values. Compared with other methods, it can significantly reduce the cost of communication. The experimental data in this article also proves the application effectiveness of the above two methods.

In order to improve the communication efficiency of FedFTG, this article proposes two methods: Federated Dropout and Deep Gradient Compression. Federated Dropout compresses the global model and transfers its smaller subset to the client, while updating the client based on the submodel to reduce communication consumption. This method can effectively reduce the local computing cost. Deep

Gradient Compression reduces communication bandwidth consumption by compressing gradients. It uses Gradient Sparsificio to only send important gradient updates that are greater than the threshold, and accumulates small gradients locally until they are large enough to be transmitted. This article proves through experimental data that these two methods indeed effectively improve communication efficiency.

References

[1] Z. Lin, S. Li, D. Liang, T. Dacheng, and D. Ling-Yu, "Fine-tuning Global Model via Data-Free Knowledge Distillation for Non-IID Federated Learning," Mar. 2022.
[2] Z. Zhen, Z. Wu, L. Feng, W. Li, F. Qi, and S. Guo, "A Secure and Effective Energy-Aware Fixed-Point Quantization Scheme for Asynchronous Federated Learning," Computers, Materials & Continua, vol. 75, no. 2, pp. 2939–2955, 2023.
[3] S. Ji, J. Zhang, Y. Zhang, Z. Han, and C. Ma, "LAFED: A lightweight authentication mechanism for blockchain-enabled federated learning system," Future Generation Computer Systems, vol. 145, no. 1, pp. 56–67, Aug. 2023.
[4] L. Wang, X. Zhao, Z. Lu, L. Wang, and S. Zhang, "Enhancing Privacy Preservation and Trustworthiness for Decentralized Federated Learning," Information Sciences, vol. 12, no. 21, Feb. 2023.
[5] H. Gao, N. He, and T. Gao, "SVeriFL: Successive Verifiable Federated Learning with Privacy-Preserving," Information Sciences, vol. 3, no. 17, Dec. 2022.
[6] J. Liu et al., "From distributed machine learning to federated learning: a survey," Knowledge and Information Systems, vol. 64, no. 4, pp. 885–917, Mar. 2022.
[7] V. Rey, P. M. Sánchez Sánchez, A. Huertas Celdrán, and G. Bovet, "Federated learning for malware detection in IoT devices," Computer Networks, vol. 204, no. 5, p. 108693, Feb. 2022.
[8] C. Liu, H. Chen, Y. Wu, and R. Jin, "MixNN: A Design for Protecting Deep Learning Models," Sensors, vol. 22, no. 21, p. 8254, Oct. 2022.
[9] T. Lebrun, A. Boutet, J. Aalmoes, and A. Baud, "MixNN," Proceedings of the 23rd ACM/IFIP International Middleware Conference, Nov. 2022.
[10] M. Naseri, J. Hayes, and E. D. Cristofaro, "Toward robustness and privacy in federated learning: Experimenting with local and central differential privacy. ," 2021.
[11] L. Zhu, Z. Liu, and S. Han, "Deep Leakage from Gradients," 2019.
[12] K. Bonawitz et al., "Practical Secure Aggregation for Privacy-Preserving Machine Learning," Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, Oct. 2017.
[13] S. Caldas, J. Koneˇcn`y, H. B. McMahan, and A. Talwalkar, "Expanding the Reach of Federated Learning by Reducing Client Resource Requirements,", 2018.
[14] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," J. Mach. Learn. Res., pp. 1929–1958, 2014.
[15] J. Koneˇcný, H. B. McMahan, F. X. Yu, P. Richtárik, and D. Bacon, "Federated learning: Strategies for improving communication efficiency,", 2016.
[16] I. Mitliagkas, C. Zhang, S. Hadjis and C. Ré, "Asynchrony begets momentum, with an application to deep learning," 2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton), Monticello, IL, USA, 2016, pp. 997-1004.