Research on image classification algorithm based on horizontal federated learning

Hao Fang[a], Ying Shen[b], Xinbei Li[c], Min Yao[a,*]
[a]School of Electronics and Information Engineering, Wuhan Donghu University, Wuhan, 430212, Hubei, China; [b]School of Continuing Education, Hubei University of Technology, Wuhan, 430068, Hubei, China; [c]School of Management, Wuhan Institute of Technology, Wuhan, 430205, Hubei, China

ABSTRACT

When it comes to image classification at a device terminal a traditional machine learning method tends to pose a risk of privacy disclosure, while the federated learning is able to alleviate such privacy problem to a certain extent by saving device data locally to train the local model. In this paper images are classified based on horizontal federated learning and FedAvg optimization algorithm. Classification model train is carried out based upon CIFAR-10 data set and ResNet-18. When the learning rate is appropriate, the optimization algorithm adopted has faster convergence, fewer communication rounds and better classification effect. The algorithm remains convergent and even shows faster convergence in the case of heterogeneous data.

Keywords: Horizontal federated learning, image classification, FedAvg

1. INTRODUCTION

With the ever-rapid development of high-performance computing, mobile Internet and other technologies, the computing power has been higher and the image collection has become more convenient. The image classification technology has made progress from manually designing to automatically extracting, and from early SVM (Support Vector Machine) and shallow neural networks to the mainstream in-depth learning model[1] currently. In addition, both the image data size and the model complexity have been greatly improved. However, the popularity of image classification application leads to an important question: how to ensure the privacy security in the application of image classification model? There is an urgent need for a joint model that can not only ensure data privacy but also combine data training of all parties, so the federated learning was born. There are many institutions, each of which has its own data that can be collected into an integrated and massive

database. It is this database that can be trained as a big data model. Unfortunately each institution is reluctant or unable to share its own data with others given its privacy and interests. So it is wise for them to make an alliance according to an all-side protocol so that the parameters obtained by each institution can only be exchanged in a confidential way and cannot be used in a dishonest way to encode other institution's data and model. Therefore, local models have been extended and grouped into an integrated one. It is where the federated learning goes.

2. CLASSIFICATION OF FEDERATED LEARNING

The federated learning has been divided by Professor Qiang Yang into three categories: horizontal federation, vertical federation and federal transfer learning[2].

2.1. Horizontal federation

Horizontal federation (also referred to as Horizontal FL in Figure 1) focuses on overlapping data features, that is, the data features of different participants are aligned but their samples are different. Let's take an example of two banks, the banks' data features are overlapping but their users rarely overlap. Horizontal federation is also called sample-division federation. The rows in Figure 1 represent samples' ID and the columns represent data features. So it is data-feature-alignment-based federation.

*minyao@whu.edu.cn

International Conference on Computer Application and Information Security (ICCAIS 2021),
edited by Yingfa Lu, Changbo Cheng, Proc. of SPIE Vol. 12260, 1226018 · © The Authors.
Published under a Creative Commons Attribution CC-BY 3.0 License · doi: 10.1117/12.2637661

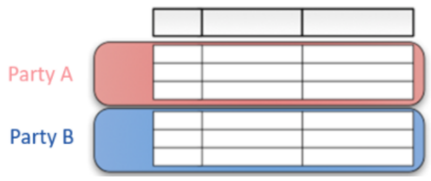


Figure 1. Horizontal federation.

2.2. Vertical Federation

Vertical federation (also referred to as Vertical FL in Figure 2) focuses on overlapping samples, that is, the samples of different participants are aligned but their data features are different. Let's take an example of a bank and an e-commerce platform, their user samples are overlapping but their transactions are different. Under the circumstances, the bank and the e-supermarket can offer a more specific portrait of consumers together, promoting their products to them according to their spending ability. So it is also called feature-division-based federation.

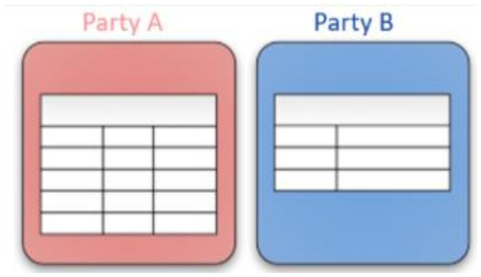


Figure 2. Vertical federation.

2.3. Federal migration learning

Federal Migration Learning focuses on federated problems of heterogeneous data, where the data samples and data features rarely overlap (as shown in Figure 3). Related research has made slow progress and will become a hot topic in the future.

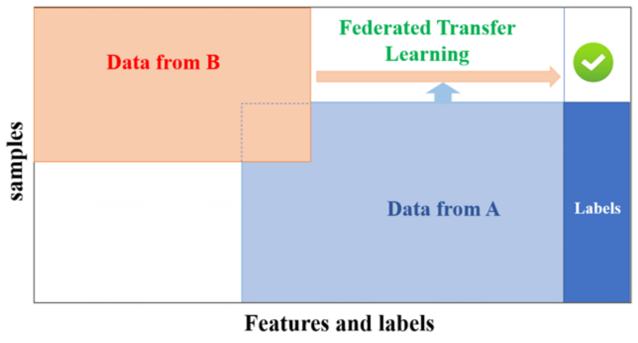


Figure 3. Federated migration.

3. FEDAVG ALGORITHM

It was Google that initiated the concept of federated learning in 2017, and gave a brief introduction of classic FedAvg algorithm, which is the most commonly used federated learning optimization algorithm at present[3-5]. In stark contrast to a conventional optimization algorithm, the FedAvg optimizes the local model by means of local random gradient descent of participants' data, and then makes aggregation at the central server.

It integrates multiple SGD-based in-depth learning models into a global model. Similar to single machine learning, the federation aims to minimize the empirical risk, that is[6]

$$\min_{x\in\mathbb{R}^d}\left[F(x)=\frac{1}{n}\sum_{i=1}^{n}f\left(x;s_i\right)\right] \tag{1}$$

n is the sample capacity, s_i represents the i th sample individual, $f(x,s_i)$ represents the loss function of the model on s_i. Assuming that there are k local models, p_k represents the serial number set of sample individuals of the k th model. Assuming $n_k = \left|p_k\right|$, the objective function is rewritten as

$$F(x)=\sum_{k=1}^{K}\frac{n_k}{n}F_k(x) \tag{2}$$

$$F_k(x)=\frac{1}{n_k}\sum_{i\in \boldsymbol{p}_k}f\left(x;s_i\right)$$

It is worth noting that the data of each terminal device cannot represent the global data, so $\mathrm{E}_{\mathrm{p}_k}[\mathrm{F}_\mathrm{k}(\mathrm{x})]$ is not the same as $F(k)$, i.e. any local model can't be regarded as a global model. A parameter updating of the local model is called an iteration. b represents $batch$, then the iteration formula of k th local model is

$$x_k \leftarrow x_k - \frac{\eta}{|B|}\sum_{i\in b}\nabla f\left(x_k;s_j\right) \tag{3}$$

FedAvg algorithm is intuitive. A training process is divided into several rounds, and $CK(0\le c\le 1)$ local models are selected to learn the data in each round. E represents the number of epoch of the k local model in the first round and B represents the size of batch, E_{n_k}/B represents the times of iterations. At the end of a round, the parameters of all local models participating in learning are weighted and averaged to obtain the global model.

4. ALGORITHM REALIZATION

4.1. Model overview

There are two main roles of FedAvg algorithm in the horizontal federated learning implemented in this paper: client and server. The main function of the server is to aggregate the local models uploaded by the selected clients[7]. The main function of the client is to receive the instructions sent by the server and the global model, and use the local data to train the local model. The structure is shown in Figure 4 as follows:

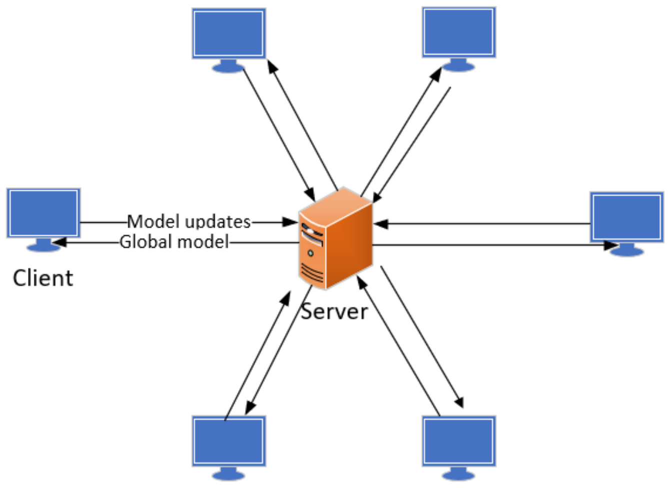


Figure 4. Client and server.

Furthermore, CIFAR-10 image data set is classified by means of horizontal federation, and ResNet-18 model is adopted.

4.2. Server

The server functions to aggregate the local models uploaded by the selected clients in horizontal federation.

When it comes to a full-featured federation framework, the server involves much more complicated functions, for example, network monitoring of every client node, sending reconnecting signals to failure nodes and so on and so forth.

Our experiment is simulated locally, not involving network communication details and troubleshooting, so information about such functional details is not discussed, but only the model aggregation function is mentioned[8-9].

First, Server, a server class is defined, where the main functions in the class are included as following.

4.2.1. Definition of constructor. In the constructor the responsibilities of the server include:

First, to copy the configuration information to the server;

Second, to get the model according to the model information as configurated. Here ResNet-18 model built in the models' module of torchvision is used.

4.2.2. Definition of model aggregation function. As we mentioned earlier, the main function of the server is to aggregate the models. Therefore, after defining the constructor, we need to define the model aggregation function in the class, and update the global model by receiving the model uploaded by the client and using FedAvg algorithm.

$$G^{t+1} = G^t + \lambda \sum_{i=1}^{m} \left(L_i^{t+1} - G_i^t \right) \tag{4}$$

where, G^t represents the global model after the *T*th round of aggregation, L_i^{t+1} represents the model after the local update of the i th client in the $t+1$ th round, and G^t+1 represents the global model after the $t+1$ th round of aggregation.

4.2.3. Definition of model evaluation function. For the current global model, we evaluate the performance of the current global model according to the evaluation data. Usually, the server-side evaluation function is mainly used to analyze the global model after the current aggregation, and to determine whether the current model training needs next iteration or should be terminated in advance, or whether the model is diverging and degenerating. The server can take different measures according to different results.

4.3. Client

The client mainly functions to receive the instructions sent by the server and the global model, and use the local data to train the local model in horizontal federation. Similarly to the server mentioned above, when it comes to a full-featured federation framework, the client involves many complicated functions, for example, the necessity to consider whether the local resources (CPU, memory, etc.) meet the training needs, the current network interruption and training due to external factors[10-12], etc.

Here, we only emphasize the local model training details of the client. First, Client, a client class is defined, where the two main functions in the class are included as following.

(1) Definition of the constructor

In the constructor, the responsibilities of the client include:

First, to copy the configuration information to the client;

Then, to get the model according to the model information as configurated. Usually the model parameters are delivered to the client from the server and the client overwrites the local model with the global model;

Finally, to configure the local training data. In this case, we get the CIFAR-10 data set through the datasets module of torchvision and then divide it according to the client ID. Different clients have different sub-data sets, and there is no intersection between them.

(2) Definition of the local training function of the model

Loss function is selected after the selection of classification model. Here we use Softmax crossover

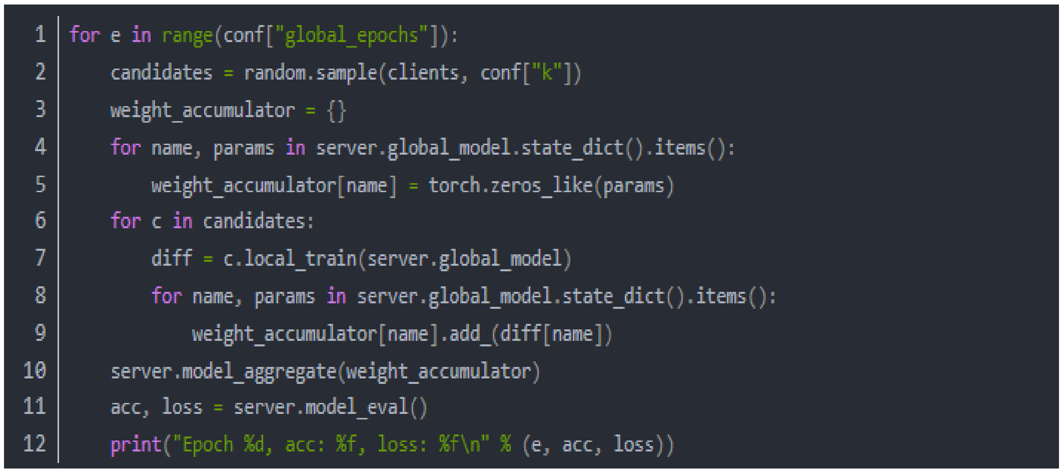
Entropy loss function, which is in the form of:

$$f_j = -\frac{1}{n}\sum_{i=1}^{n} y_{j,i} \log\left(\frac{e^{\hat{y}_{j,i}}}{\sum_{l=1}^{k} e^{\hat{y}_{j,l}}}\right) \tag{5}$$

where, n is the sample size on device $j, y_{j,i}$ is the true label of j th sample of device i th, and $\hat{y}_{j,i}$ is the prediction label of j th sample of device i th, which is actually related to the parameter w of classification model. k is the number of categories, or the dimension of label vector. Next it is necessary to aggregate and average the loss function on each device in federated learning.

4.4. Aggregation

When the configuration file, server class and client class are all defined, we combine this information together. First, read the profile information. Next, we will define a server object and several client objects respectively to simulate the horizontal federation training scene. At every iteration, the server randomly selects some clients to participate in iterative training. The selected clients call the local training interface "local_train" for local training. Finally, the server calls the model aggregation function "model aggregate" to update the global model. The codes are shown in Figure 5 as follows:



```
for e in range(conf["global_epochs"]):
    candidates = random.sample(clients, conf["k"])
    weight_accumulator = {}
    for name, params in server.global_model.state_dict().items():
        weight_accumulator[name] = torch.zeros_like(params)
    for c in candidates:
        diff = c.local_train(server.global_model)
        for name, params in server.global_model.state_dict().items():
            weight_accumulator[name].add_(diff[name])
    server.model_aggregate(weight_accumulator)
    acc, loss = server.model_eval()
    print("Epoch %d, acc: %f, loss: %f\n" % (e, acc, loss))
```

Figure 5. Partial aggregation code.

4.5. Comparison of the effects of federated learning and centralized training of Resnet 18 on CIFAR-10

Training results are shown in Figure 6.

Federal training configuration: There are 10 client devices (no_models=10), 5 of which are randomly selected to participate in training (k=5) in each round, and the number of local training iterations is 3 (local_epochs=3) and the number of global iterations is 20 (global_epochs=20).

Centralized training configuration: We don't need to write centralized training code separately, and only have to modify the federated learning configuration to make it equivalent to centralized training. Specifically, we can set the client device no_models and the number of training devices k selected in each round to 1. This kind of federal training with only one device is equivalent to centralized training. Other parameter configuration information is consistent with federated learning and training.

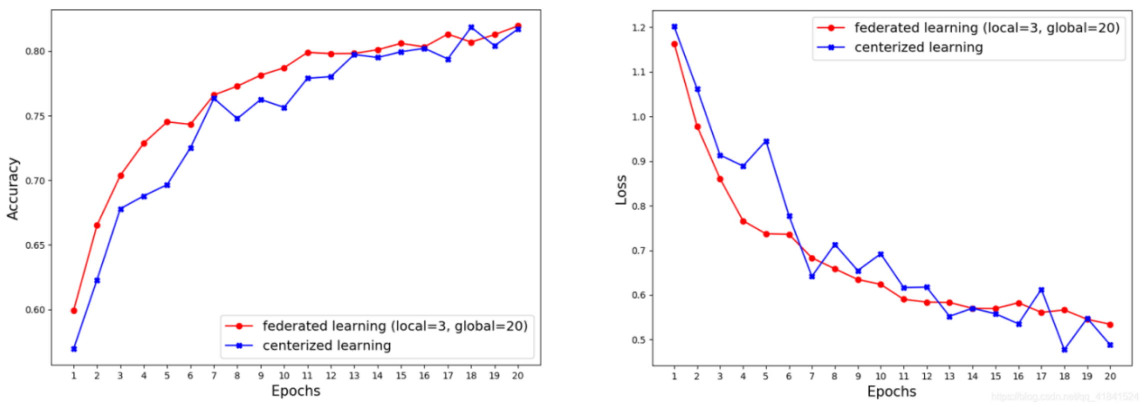


Figure 6. Comparison of effects between federated learning and centralized training.

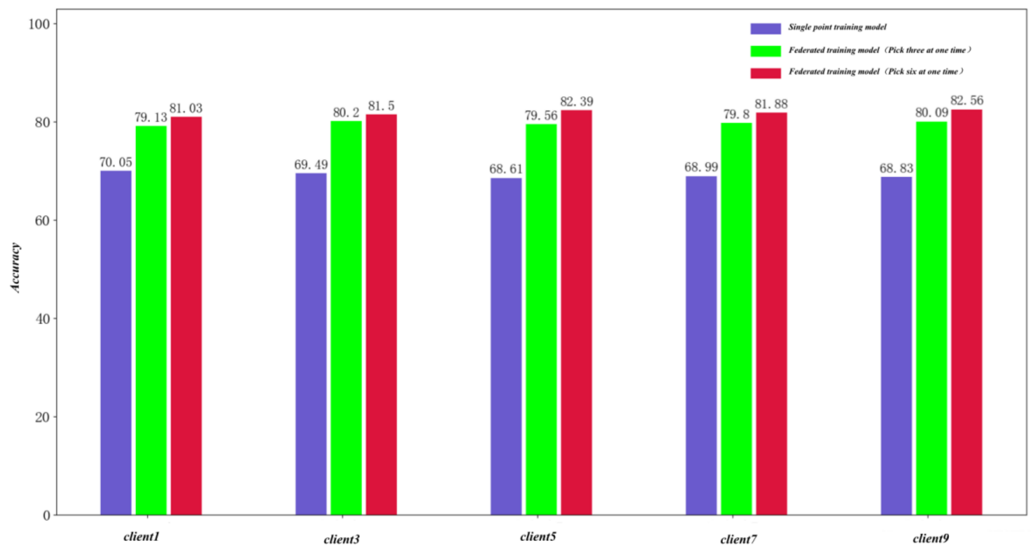


Figure 7. Comparison between single point training and federated training.

Training results are shown in Figure 7.

The single-point training in the figure is the result of model training using local data under a certain client.

We can see that the model effect of single-point training (blue bar) is obviously lower than that of federal training (green bar and red bar), which shows that the global distribution characteristics of data cannot be well learned only through the data of a single client, and the generalization ability of the model is poor.

In addition, for the different number of clients (*k* value) participating in each round of federal training, its performance will be different. The larger the k value, the more clients participating in each round of training, the better its performance will be, but the completion time of each round will be relatively longer.

5. CONCLUSION

Federated learning is a product driven by the trend of privacy protection in the era of big data, while image classification is a very common application in our daily life. They are not separated but symbiotic. Based on the settings in the training process of image classification model of federated learning, this paper analyzes the effect of the federated optimization algorithm. The results show that, when the step size is properly selected, the convergence rate of the federated optimization algorithm is faster, and in the case of heterogeneity, the algorithm can converge stably, and the number of communication rounds also reduces.

However, it should be noted that in this paper we only simulate the communication process between the client and the server locally, but do not carry out the communication between the server and the client in the real network environment, so the communication delay of local simulation is meaningless. We work out this experiment to prove the effectiveness of federated learning and show its performance comparison with centralized learning. It is necessary for us to make further exploration on actual network deployment and privacy protection of communication parameters.

ACKNOWLEDGMENTS

The authors sincerely appreciate the editor and the reviewers for their helpful comments. This work was supported by Following projects: (1) Ministry of education industry university cooperation collaborative education, 202002288065; (2) Research team construction of Wuhan Donghu University; (3) The teaching research project of Wuhan Donghu University.

REFERENCES

[1] Huang, J. R., [Optimization in Image Classification], Nanjing University of Information Science and Technology, chapter 2, 8-13 (2021). (in Chinese)
[2] Yang, Q., "Federated learning: The last on kilometer of artificial intelligence," CAAI Transactions on Intelligent Systems, 15, 183-186 (2020).
[3] Sun, L. L., Li, A., Yu, S. W. and Wang, Y. X., "A summary of the research on the security image classification model for encrypted data," Journal of Cryptography, 4, 525-540 (2020).
[4] Li, S. B., Yang, L. and Li, C. J., "Overview of federated learning: Technology, application and future," Computer Integrated Manufacturing System, 9, 125-130 (2021).
[5] Zhang, Y. L., Chen, Y. X., Tian, H. and Wang, T., "Research progress of application of federated learning in edge computing scenarios," Miniature Microcomputer System, 10, 213-220 (2021).
[6] Cheng, K. W., Fan, T., Jin, Y. L., Liu, Y., Chen. T. J. and Yang, Q., "Secureboost: A lossless federated learning framework," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 5, 225-230 (2019).
[7] Yang, Q., [A Quick Introduction to Federated Learning], Electronic Industry Press, chapter 2, 132-153 (2021). (in Chinese)
[8] Tonellotto, N., Gotta, A. and Nardini, F. M., "Neural network quantization in federated learning at the edge," Information Sciences, 2, 110-120 (2021).
[9] Feki, I., Ammar, S., Kessentini, Y. and Muhammad, K., "Federated learning for COVID-19 screening from Chest X-ray images," Applied Soft Computing Journal, 106, 203-215 (2021).
[10] Tokarev, K. E., Zotov, V. M., Khavronina, V. N. and Rodionova, O. V., "Convolutional neural network of deep learning in computer vision and image classification problems," IOP Conference Series: Earth and Environmental Science, 625-631 (2021).
[11] Wang, G. H., Guo, Z., Wan, X. Z. and Zheng, X., "Study on image classification algorithm based on improved DenseNet," Journal of Physics: Conference Series, 12, 621-632 (2021).
[12] Mohammad, M., Mehdi, S. and Zhang, Y. D., "A noise robust convolutional neural network for image classification," Results in Engineering, 10, 321-322 (2021).