

Image Classification Based on Federated Learning and PFLlib

Weiqing Fu^a

College of Computer Science and Technology, Jilin University, Jilin, China

Keywords: Image Classification, Federated Learning, Neural Network.


Abstract: Image classification has always been a research hotspot in the computer vision community, which is also the foundation of many higher-order scene understanding tasks. Based on data-driven ideas, most existing image classification models rely on massive data and centralized large-scale training. However, due to the security and privacy issues of data, practical application scenarios often cannot fully utilize all training data, resulting in significant room for improvement in the accuracy and robustness of the model. Inspired by the rapid development of federated learning, this article introduces the idea of local training and global updates into image classification tasks, exploring the performance boundaries of different representative federated learning algorithms in classification tasks. Specifically, based on the PFLlib platform, this article designs a unified Client and Server end that can integrate common federated learning algorithms. In addition, this article quantitatively compares the impact of neural network structures on the classification performance of different methods. Extensive experiment results have verified the significant improvement of federated learning in classification performance.

1 INTRODUCTION

Image classification aims to predict the category of a given image and has always been a hot research topic in the computer vision community and the foundation of numerous higher-order visual tasks (Xu, 2020; Wang, 2023; Xu, 2021; Sun, 2023). In recent years, with the rapid development of pattern recognition technology, especially deep learning, image classification technology is rapidly developing and showing great application prospects in mobile payments, autonomous driving, environmental monitoring, and other fields. However, traditional image classification models mostly follow a centralized training paradigm, where all training data needs to be centralized locally before model training. This makes traditional image classification face problems such as high privacy leakage risk, high communication overhead, poor data personalization, and high hardware requirements, which restrict the further application of image recognition technology in real life. In this context, exploring distributed methods to achieve image recognition has attracted a lot of research interest.

Due to constraints on data privacy and security such as laws and regulations, policy supervision, trade secrets, and personal privacy, multiple data sources are unable to directly exchange data, resulting in a phenomenon of “data silos” that restricts the further improvement of artificial intelligence model capabilities. Joint learning, as a distributed machine learning method, provides feasible solutions to issues such as privacy leakage and data communication overhead. The technical theoretical foundation of federated learning can be traced back to the association rule mining technology of Distributed Database. In 2006, Yu et al. proposed a distributed support vector machine model with privacy protection on horizontally and vertically segmented data. Federated learning enables multiple clients to train the same model together while protecting the private data of each client.

Inspired by the demand for training data in large-scale image recognition models and the ability of federated learning to solve the problem of data silos, this work constructs a federated learning simulation framework based on federated learning algorithms using various neural network models at the local client, on which experiments were conducted. In the

^a <https://orcid.org/0009-0008-3422-3348>

concrete implementation, the central server is designed as a class, and the servers in various federated learning algorithms jointly inherit a server base class, which has methods such as sending and receiving model parameters and selecting clients. The clients in the various federated learning algorithms jointly inherit a client base class, which has methods for receiving model parameters, local model training, and so on. The client is able to select the neural network used for local model training, including CNN (Convolutional Neural Network), DNN (Deep Neural Network), and so on. This study carries out tests based on the mentioned simulation platform to determine the global average accuracy, training duration per round, local training accuracy, and other indicators to assess the capabilities of image recognition applications under various settings.

Focusing on the above points, this paper is organized as follows. In Section 2, the representative federated learning algorithms are first introduced, including their design ideals and basic steps. Then, Section 3 shows the design of the distributed framework and gives all the implementation details. Section 4 verifies the effectiveness of the proposed distributed framework and Section 5 concludes the work and discusses future development.

2 RELATED WORK

2.1 Federated Learning

The three types of federated learning algorithms that are now in use are federated transfer learning, vertical federated learning, and horizontal federated learning. The development of federated learning has drawn a lot of attention (Durmus, 2021; Liu, 2020; Baldini, 2017). Sample-based federated learning, another name for horizontal federated learning, is mostly used in situations where the participant's dataset has separate sample spaces but the same feature space. The common design ideas for horizontal federated learning are gradient averaging and model averaging. In the gradient averaging method, the participants perform the computation of model gradients locally and send gradient information to the server, which aggregates the received gradient information, often by computing a weighted average. After that, the aggregated gradient information is distributed again by the server to each participant. In addition to sharing gradient information, participants in federated learning can also share model parameters. The model parameters are computed locally, after which the server receives the model parameters from

each participant. The server usually aggregates the model parameters using a weighted average, after which the aggregated model parameters are redistributed by the server to the participants. This approach is called model averaging. Feature-based federated learning (where features are changing) is another name for vertical federated learning, which means that data is divided vertically (by column) and the ID space of the samples remains unchanged. Specifically, for different business purposes, datasets owned by different organizations often have different feature spaces, while these organizations may share a huge user group. Based on this heterogeneous data, a vertical federated learning model has been built. The common vertical federated learning algorithms mainly include secure federated linear regression and secure federated lifting tree SecureBoost (Cheng, 2021). Different from the first two types of methods, federated transfer learning extends traditional transfer learning to privacy-preserving distributed machine learning paradigms, mainly targeting scenarios where two or more datasets do not have similarities in feature space and sample ID space.

2.2 Neural Network

Deep Neural Network (DNN) is a basic neural network with the ability to model complex nonlinear systems. There are three layers in a DNN: input, hidden, and output. A DNN typically includes one or more hidden layers and introduces nonlinear features via an activation function, like a Sigmoid or ReLU function. A DNN model is trained by calculating a loss function, back-propagation, which typically uses cross-entropy loss or mean error. Uses cross-entropy loss or mean square error. A popular type of neural network used for tasks like image classification, target recognition, picture segmentation, etc. is the convolutional neural network (CNN). CNN models can extract features more efficiently and at the same time reduce the number of parameters of the model. Convolutional, pooling, activation functions, and fully connected layers are the typical components of CNNs. LeNet, AlexNet, and other conventional CNN models are examples.

2.3 PFLlib

Personalized Federated Learning Algorithm Library (PFLlib) is an experimental platform that supports running federated learning simulation experiments on a single machine, supports multiple federated learning algorithms, multiple datasets, and multiple neural network models, and provides several

federated learning simulation options, such as differential privacy, client-side deceleration, etc. PFLlib supports scaling federated learning algorithms.

3 METHOD

3.1 Revisiting FedAvg and MOON

Federated averaging (FedAvg) (McMahan, 2017) is a traditional federated learning algorithm that can train models without the local data being shared, thus ensuring data privacy. FedAvg requires each client to perform gradient descent updating locally, and the parameters of all clients are aggregated on the server side, which aggregates the parameters by performing a weighted average of the parameters of each client. FedAvg requires each client to train the model locally and upload the trained model parameters to the server. The server computes the global model by applying a weighted average to the parameters. FedAvg ensures the privacy of each local private dataset by sharing the model parameters among the clients.

A straightforward and efficient federated learning framework with benefits for handling data heterogeneity across several clients is Model-Contrastive Federated Learning (MOON) (Li, 2021). MOON optimizes the training results of local models by taking the similarity of different models into account, i.e., by comparing them at the model level. The MOON algorithm takes advantage of the similarity between model representations to rectify each client's local learning, and employs the contrastive loss between the global and local model representations as a regularization term to constrain the update of the local model. Model comparison loss aims to close the gap between the features produced by the most recent updated model and the global model, as well as widen the gap between the current model and the features produced by the previous iteration of models. This is because the server-side generated global model can yield superior features than the locally updated model.

3.2 Neural Network Structure

The particular neural network structure that is being used will be introduced in this section. The input layer of the DNN model, measuring $1 \times 28 \times 28$, is in charge of taking in input data and distributing the picture data into a vector. Hidden layer fc1 is a fully connected layer that receives data from the input layer and maps it to a hidden layer with `mid_dim` neurons.

In this network, `mid_dim` is set to 100, which means the hidden layer has 100 neurons. The activation function for this layer is ReLU. Data from the hidden layer is mapped to an output layer with `num_classes` neurons by the output layer, another fully connected layer. Here, `num_classes` is set to 10, since the usual handwritten digit classification task has 10 classes (0 to 9). There is no activation function for this layer, as the softmax function is used.

For the CNN model, the dimension of the input features is determined by `in_features`. Since the default input is a single-channel image, features are set to 1. Conv1, the first convolutional layer, is made up of a maximum pooling layer, a ReLU activation function, and a convolution operation. 32 size 1 convolution kernels with a stride of 1 and no padding are used in the convolution operation. The ReLU activation function is used to introduce nonlinearities, and the maximum pooling layer is used to reduce the spatial dimensions of the feature map, thus reducing the computational effort. Conv2, the second convolutional layer, is made up of a maximum pooling layer, a ReLU activation function, and a convolution operation, just like the first one. Here the input channel is 32 and the output channel is 64, and the other parameters are set the same as the first convolutional layer. Fully Connected Layer fc1 is a fully connected layer used to flatten the output of Conv2 into a vector and map it to a 512-dimensional feature space. Nonlinearities are introduced using ReLU. Output Layer is the final fully connected layer that maps the output of fc1 to the final output space, which has the dimension `num_classes` for the number of categories corresponding to the tasks.

3.3 PFLlib Platform Design

The platform integrates the function of generating datasets by executing the corresponding dataset and generating a Python file through the command line to generate the corresponding dataset. The main framework of proposed framework mainly consists of (1) the Command receiving and processing part; (2) the Server emulation part; (3) the Client simulation part and (4) the Model part.

As shown in Figure 1, the command receiving and processing part receives command line commands, generates corresponding model instances, Server objects and Client objects, and sets related parameters. The main logic is to receive the command content through the variable "args", and then instantiate the corresponding model, the corresponding Sever terminal and open the relevant settings according to the command content.

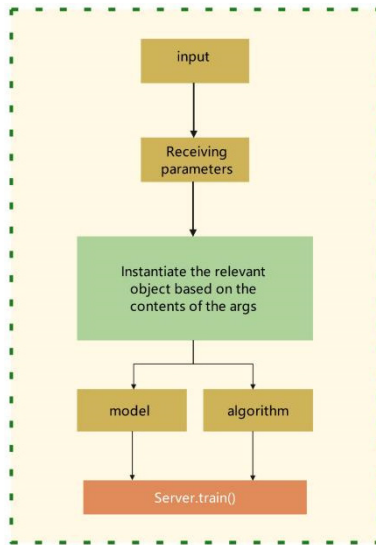


Figure 1: Command reception processing section (Photo/Picture credit: Original).

The workflow of the Server and Client is shown in Figure 2. The Server side of different algorithms inherits from a base class called “Server”, which provides functions such as sending parameters to the client, receiving parameters from the client, selecting the client, and so on. Because of the design differences between the algorithms, the specifics of what is performed on the Server side of each algorithm are different. The purpose of this architectural design is to provide standard interfaces through a unified Server base class to ensure that the Server side of different algorithms can be developed under the same framework. In this way, the system can be more easily adapted to the needs of different algorithms while maintaining consistency and maintainability. The clients of all algorithms inherit from the Client base class, which provides a set of functions required for model training. The main responsibility of the client is to implement the training process for all types of models, while covering the core elements required for algorithm execution. This design allows the system to handle the training tasks of different algorithms in a consistent manner, ensuring code consistency and maintainability through the standard interfaces provided by the base class.

The platform provides a variety of network models such as CNN, DN, etc. Models are inherited from the “nn.Module” class under pytorch framework. Due to the clear architecture of the platform, it has excellent scalability. To introduce new algorithms into the platform, simply follow the steps below:

(1) To achieve the simulation operation on the Server side, write the simulation code on the Server side under the current federal learning algorithm and create the corresponding file.

(2) To achieve the simulation operation of the Client side, write the simulation code of the Client side under the current federated learning algorithm and create the corresponding file.

(3) To ensure that the platform can correctly identify and run the new algorithm, modify the logic in the main file.

Through the above steps, the system can integrate the new algorithm into the platform. This design idea makes extending the platform very intuitive and easy, while ensuring the clarity of the overall structure. The introduction of new algorithms does not affect existing algorithms, and can make use of existing base classes and interfaces, which improves the maintainability and expandability of the code.

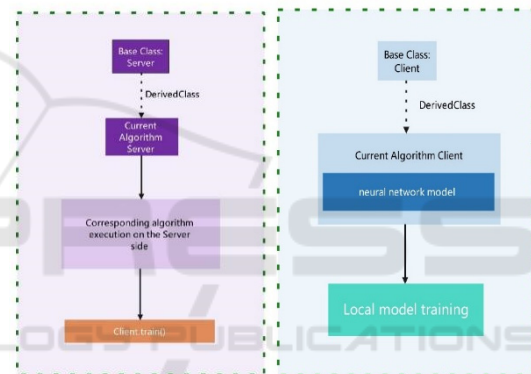


Figure 2: Structure of Server (right) and Client (left) (Photo/Picture credit: Original).

4 EXPERIMENT

4.1 Original Dataset

This experiment selected the Fashion MINIST dataset as the raw training data for the model. Ten categories of T-shirts, jeans, pullovers, dresses, coats, sandals, shirts, sneakers, bags, and short boots are included in the 60000 training set and 10,000 test set photos that make up Fashion MINIST. Each image is a grayscale image with a resolution of 28×28 . Although the images are too small, the quantity is sufficient, and the labels are evenly distributed.

4.2 Performance Analysis

First, tests are carried out to report the loss values of

several federated learning algorithms under various network configurations in order to confirm the convergence of the model. As shown in Table 1, taking the MOON algorithm as an example, when using a CNN network structure, the loss value gradually decreases from 2.31 to 1.00 with increasing training rounds. Similar model convergence trends can be observed in the FedAvg algorithm and DNN network structure. It should be noted that the model can converge with only about 10 epochs, indicating that the distributed model training concept of federated learning can improve the learning efficiency of the model.

Table 1: Comparison of training loss for various models.

Epoch	MOON_cnn	FedAvg_cnn	MOON_dnn	FedAvg_dnn
1	2.3128	2.3128	6.2129	6.2129
2	2.1526	2.1518	2.4193	2.4300
3	1.8138	1.8079	1.8428	1.8334
4	1.5184	1.5178	1.5930	1.5909
5	1.3397	1.3419	1.4467	1.4476
6	1.2336	1.2395	1.3318	1.3350
7	1.1611	1.1661	1.2595	1.2571
8	1.1114	1.1121	1.2034	1.2004
9	1.0700	1.0634	1.1543	1.1564
10	1.0402	1.0410	1.1211	1.1213
11	1.0053	1.0044	1.0954	1.0969

In addition, an additional set of experiments was conducted to compare the recognition accuracy of different algorithms, and Figure 3 shows the results. As the training process increases, the average testing accuracy of MOON_cnn on the Fashion MNIST dataset can be improved from 0.21 to 0.63. In the early stages of training (0-3 epochs), there are some differences in the testing accuracy of different network structures and algorithms, and MOON_cnn shows the best results. As the model is further fully trained, the FedAvg algorithm and MOON algorithm will converge to approximately the same accuracy.

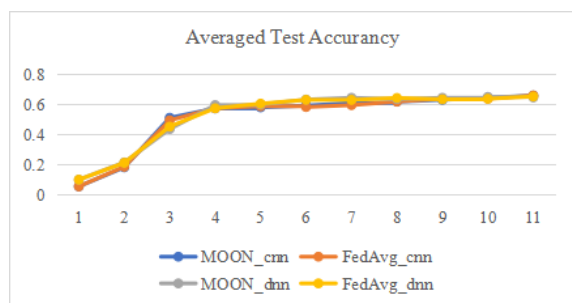


Figure 3: Average test accuracy of different models.

5 CONCLUSIONS

To explore the improvement effect of federated learning on image recognition tasks, this paper designs a unified client and server based on the PFLlib platform, integrating representative federated learning algorithms with different network structures. Numerous trials have verified the effectiveness of the work in this paper. The module partitioning and overall structural design of the PFLlib platform showed significant clarity in the experiment. By storing the client and server implementations of different algorithms in separate folders and adopting the inheritance mechanism of base classes, the platform maintains consistency and maintainability.

REFERENCES

- Baldini, I., Castro, P., Chang, K., Cheng, P., Fink, S., 2017. Serverless computing: Current trends and open problems. *In Research advances in cloud computing*, 1-20.
- Cheng, K., Fan, T., Jin, Y., Liu, Y., Chen, T., Papadopoulos, D., Yang, Q., 2021. Secureboost: A lossless federated learning framework. *IEEE Intelligent Systems*, 2021, 36(6):87-98.
- Durmus, A., E., A., Zhao, Y., Ramon, M., N., Matthew, M., Paul, N., W., Venkatesh, S., 2021. Federated learning based on dynamic regularization. *In arxiv preprint arXiv:2111.04263v2*.
- Li, Q., He, B., Dawn, S., 2021. Model-Contrastive Federated Learning. *In Proceeding of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Liu, R., 2020. FedSel: Federated sgd under local differential privacy with top-k dimension selection. *Database Systems for Advanced Applications: 25th International Conference, DASFAA 2020, Jeju, South Korea, September 24-27, 2020, Proceedings, Part I 25*. Springer International Publishing, 2020.
- McMahan, H., B., Eider, M., Daniel, R., Seth, H., Blaise, A., y., A., 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. *In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics 2017*.
- Sun, H., Lang, W., Xu, C., Liu, N., Zhou, H., 2023. Graph-based discriminative features learning for fine-grained image retrieval. *Signal Processing: Image Communication*, 110:116885.
- Wang, J., Wang, Y., Liu, L., Yin, H., Ye, N., Xu, C., 2023. Weakly Supervised Forest Fire Segmentation in UAV Imagery Based on Foreground-Aware Pooling and Context-Aware Loss. *Remote Sensing*. 15, no.14: 3606.
- Xu, C., Jiang, H., Peter, Y., Khan, Z., A., Chen, Y., 2020. MHW-PD: A robust rice panicles counting algorithm

based on deep learning and multi-scale hybrid window.
Computers and Electronics in Agriculture, 173:105375.
Xu, C., Lang, W., Xin, R., Mao, K., Jiang, H., 2021.
Generative detect for occlusion object based on
occlusion generation and feature completing. *Journal of
Visual Communication and Image Representation*,
78:103189.

