A Federated Learning for Image Classification with Heterogeneous Data

Huy Q. Le, Minh N. H. Nguyen and Choong Seon Hong*
Department of Computer Science and Engineering, Kyung Hee University
Email: {quanghuy69, minhnhn, cshong}@khu.ac.kr

Abstract

Federated Learning has been introduced as a novel approach for large scale decentralized learning systems of heterogeneous data on multiple devices such as mobile phones and IoT devices. It enables many participating clients to train a shared model and keep user data locally. In this paper, we implemented a practical experiments of Federated Learning for image classification tasks on different datasets using TensorFlow Federated (TFF) framework from Google. Thereafter, we implement Convolutional Neural Networks and Federated Averaging algorithm on the unbalanced and non-IID datasets. We also apply decaying learning rate to enhance the robustness of Federated Averaging algorithm and demonstrate the stable convergence in learning performance.

1. Introduction

Nowadays, with the breakthroughs of multiple techniques in Deep Learning, we are living in the massively thriving era of AI. It contributes a prominent part in numerous categories of daily life with many advanced applications such as autonomous vehicles, agriculture and healthcare systems [1, 2]. However, privacy issue in available machine learning techniques has been concerned and motivated many interest in exploring distributed AI paradigms. Thus, Federated Learning (FL) [3] is turning into an important scheme in ML.

The federated training scheme in ML has become a hot research topic among multiple different techniques of deep learning. Federated Learning which has been introduced in 2016 by Google is a distributed machine learning approach which allows multiple users to train a global model without exchanging any private data. For example, mobile devices can download the current model from server and enhances it with their available data and forward the updates to the server using encrypted communication. With this kind of mechanism, federated learning can provide the novel usage of ML tasks with lower latency, less power consumption while guaranteeing the privacy [4].

Federated Learning has been considered in multiple deep learning tasks such as image classification [5] and nature language tasks [6]. In this paper, we implement and evaluate the typical federated learning algorithm (i.e., federated averaging, federated SGD) for image classification task on two datasets. This work is conducted using TensorFlow Federated (TFF) framework [7] from Google. TFF is an extensible, powerful framework for executing federated learning research by simulating federated on realistic proxy datasets. This framework can support the users for experimenting FL algorithms on the custom models as well as contributing the new federated datasets. Researchers can also develop new FL algorithms with this framework and easily adopt real distributed AI systems.

We can summarize our contributions in this paper as follows:

- Our experiments are deployed on TensorFlow Federated which is an open source framework for experimenting with machine learning and other computations on decentralized data.
- The system is trained with CNN network on distributed MNIST[8] and CIFAR-10[9] datasets for evaluations.
- We implement and evaluate the task using Federated Averaging algorithm [1] on unbalanced and non-IID data. Then we incorporate the adaptive learning rate based on decaying scheme to enhance the robustness of the algorithm.

This work was partially supported by the Korea Institute of Energy Technology Evaluation and Planning(KETEP) and the Ministry of Trade, Industry & Energy(MOTIE) of the Republic of Korea (No. 20209810400030) and by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2019-0-01287, Evolvable Deep Learning Model Generation Platform for Edge Computing). *Dr. CS Hong is the corresponding author.

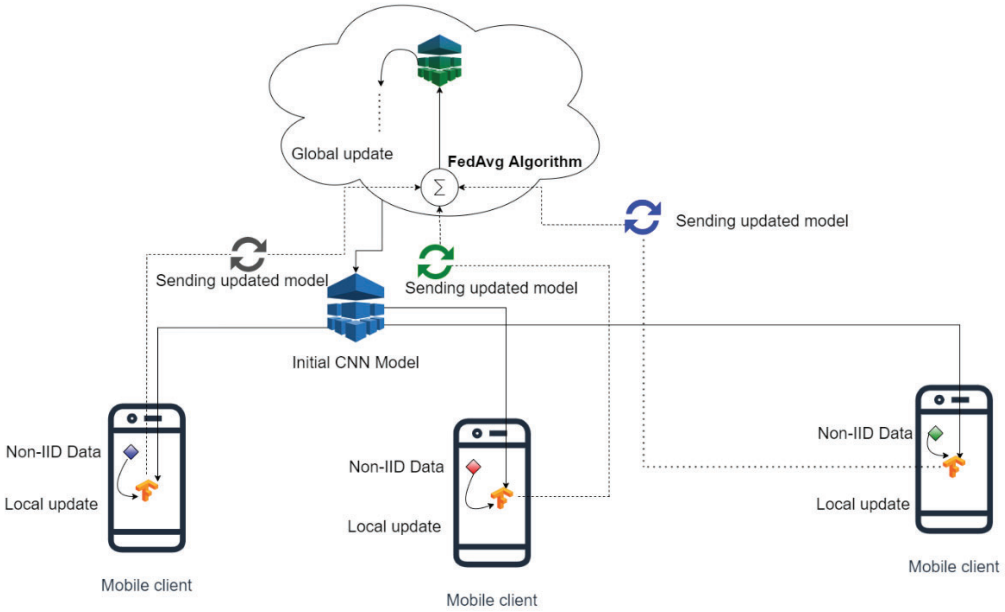

Fig. 1: System model

2. System Model

System mechanism:

The system model is shown in Fig. 1. The implementation is deployed on TensorFlow Federated (TFF) which is specialized framework for Federated Learning research. We consider each mobile device is one client. Due to the federated scenario, each client has different distribution of private dataset. At first, clients receive the initial model for image classification from the server then they will train the model and update it locally. After finishing the local iterative training, each device sends its trained model updates to the server. Sever performs the aggregation based on the average results to the final global model by using federated averaging algorithm. Afterward, server distributes the updated global model to mobile devices and the iterations continue to increase the performance of the learning models.

Client selection:

In a practical training case, only a certain number of devices are available for training at the same time among the massive number of users. Since we are in the simulation scenario, we will randomly choose the set of clients at one time and select batches of local data of those clients for training iteratively.

Training the model on federated data:

Federated averaging algorithm is constructed in TFF framework [7] by invoking the function **tff.learning.build_federated_averaging_process.** With this function, there are two types of optimizers: client optimizer which is used for local updates and server optimizer for global updates. For both optimizers, we use Stochastic Gradient Descent (SGD) algorithm. In this function, TFF has constructed a pair of federated computations which consist of *initialize* and *next* properties and then packaged them into **tff.templates.IterativeProcess.** With the *initialize* computation, it returns the representation of the *state* of federated averaging process on the server which contains the model weight parameters. The second property *next* represents a single round of federated averaging. Each time the *next* method is invoked, the server is broadcast to each client using a broadcast function. For each client, one local training round is conducted via the tf.keras.optimizers.Optimizer method of the client optimizer. Each client then computes the difference between the client model after training and the initial model. These updated models are then accumulated at the server and applied by using the server optimizer.

3. Performance evaluation:

To perform the experimental part of our study, we use the custom federated MNIST and CIFAR-10 dataset.

MNIST: The federated MNIST dataset is already available in TFF library distributed in 3383 users with unbalanced and non-IID data. For MNIST dataset evaluation part, we choose randomly 10 clients and train the model on their local data. The implemented model for this dataset consists of 1 dense layer with 10 units which is represented for 10 classes using softmax at output layer. For federated averaging algorithm, we apply decaying learning rate scheme throughout the learning process. Specifically, in each global round, we decrease the learning rate by 3%. As shown in Fig. 3, the federated averaging algorithm becomes more stable.

Meanwhile, federated SGD algorithm obtains just around 50% of accuracy.

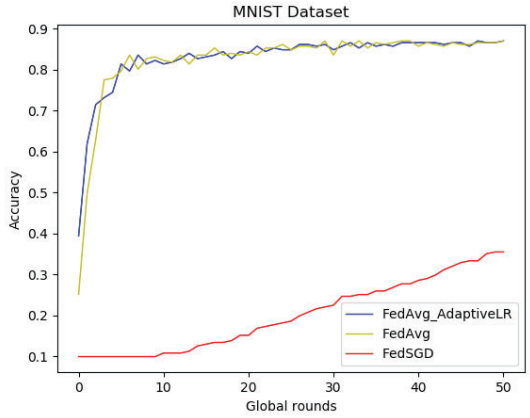

Fig. 2: Test accuracy of MNIST Dataset

CIFAR-10: TFF framework does not support federated CIFAR-10 dataset so that in this paper we customize the original dataset for the federated scenario for testing. We distribute the initial dataset into 100 clients with 500 training images and 100 testing images for each client. For this experiment, we randomly choose 20 clients and their datasets for training. Due to the complexity of the CIFAR-10 task, the implemented CNN model has three convolutional layers and three dense layers. The first two layers contain 6 and 16 channels with 3x3 kernel size for both. The final convolutional layer with same kernel size is followed by dropout [10] with 0.25 probability. The dense layers with 120 and 84 units respectively using ReLu activation [11] are accompanied with 0.2 dropout probability and a softmax output layer. As Fig. 3 shows, the model reaches around 50% accuracy after 50 global rounds. The accuracy is lower than MNIST task due to the complexity of CIFAR-10 dataset. Because of the complexity of the task, we just performed the result on federated averaging algorithm.

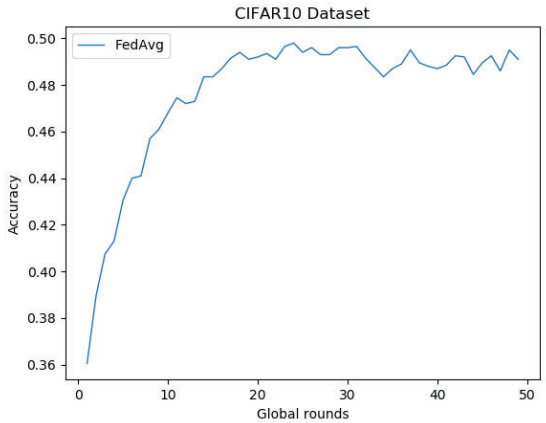

Fig. 3: Test accuracy of CIFAR10 Dataset

4. Conclusion

In this paper, we implemented Federated Learning through image classification task with our customized non-IID and unbalanced dataset with decaying learning rate scheme and achieve the better result comparing to the initial algorithm. Also, we conducted with TFF which is the open source specialized framework for Federated Learning. The training results show that the federated averaging algorithm has difficulty to attain high accuracy on CIFAR-10 dataset as MNIST dataset due to the complexity of this dataset. In future works, we will continue to develop new FL algorithms based on this framework to get the better performance on different tasks.

References

[1] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[2] L. Deng, D. Yu *et al.*, "Deep learning: methods and applications," *Foundations and Trends® in Signal Processing*, vol. 7, no. 3–4, pp.197–387, 2014.

[3] H. B. McMahan, E. Moore, D. Ramage, S. Hampson et al., "Communication-efficient learning of deep networks from decentralized data," *arXiv preprint arXiv:1602.05629, 2016.*.

[4]https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[6] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, B. Kingsbury et al., "Deep neural networks for acoustic modeling in speech recognition*," IEEE Signal processing magazine*, vol. 29, 2012.

[7] https://tensorflow.org/federated

[8] Yann LeCun, Corinna Cortes, and Christopher JC Burges. 2010. MNIST handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist 2 (2010).*

[9] https://cs.toronto.edu/~kriz/cifar.html

[10] L. Wan, M. Zeiler, S. Zhang, Y. Le Cun, and R. Fergus, "Regularization of neural networks using dropconnect," *in International conference on machine learning*, 2013, pp. 1058–1066.

[11] Richard HR Hahnloser, Rahul Sarpeshkar, Misha A Mahowald, Rodney J Douglas, and H Sebastian Seung. 2000. Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature* 405, 6789 (2000), 947.