Share knowledge with the right guy

黎天乐

Abstract

With the fast growing of the Internet of Things(IOT), more and more devices are placed everywhere. The generated data of IOT devices is huge and valuable, but they are also separated. To make use of these data, the revised paper[1] bring up a decentralized method Def-KT and I manage to revise the algorithm. I also bring up a client selection method to improve this algorithm. The experiment show that Def-KT can tackle the heterogeneity problem in IOT environment and the improved method can achieve higher accuracy than Def-KT.

Keywords: Def-KT, decentralized, client selection.

1 Introduction

Nowadays, the Internet of Things(IOT) is growing very fast. There are smart devices everywhere and these devices are generating a great amount of data every minute. In the same time, machine learning has achieved higher performance than any other rule-based algorithm. However, every coin has two sides, the ML is data-driven and its performance strongly rely on the amount of data.

It is a natural thought that we want to build a bridge between the two and leverage the data from IOT devices to train the ML model. However, there are two problems getting in our way. The first one is privacy sensitive. The smart devices capture the information of people and people would not want to leak out their privacy. The second one is that data is distributed in many devices and it would be costly to aggregate them.

In 2016, Google brought up federated learning(FL) to tackle the problem. The FL transfers the model instead of raw data. There are three steps in FL[2]. First, the central server gives out the model. Second, the devices use their private to train the model and upload it. Third, every few epochs the central server would aggregate the model and give out again. Many researches have shown that the FL would greatly reduce the communication cost because the parameters of model are much smaller than the raw data. And it can also guarantee data privacy.

In the primary framework of FL, we need a central server to give out and aggregate models. However, in many IOT environment, it is difficult to find a central server that is both reliable and powerful[3]. The single point of failure and lack of scalability also exist due to the central server.

To overcome these difficulties, we need decentralized federated learning(DFL) which dose not need a central server[4].

The general FL approach also faces two problem because of the imbalance of data. The first one is poor convergence on highly heterogeneous data. Since the data is not independent identically distribution(iid), simply averaging the parameters would lead to client-drift[5] and deteriorate the performance. The second one is the lack of solution personalized. Different IOT devices will capture data with different patterns. One global

FL model could not satisfy the need of every device, which will decrease the willing of participation. Therefore, it would be a way out to perform personalized federated learning in IOT devices[6].

”Decentralized Federated Learning via Mutual Knowledge Transfer”[1] was published on IEEE INTERNET OF THING JOURNAL in 2022. The author is Chengxi Li and Gang Li, who is one of the leaders of federated learning. To overcome the problem of decentralized federated learning(DFL) and client-drift due to data heterogeneity, this paper propose the DFL via a mutual knowledge transfer(Def-KT) algorithm, where local clients gain external knowledge by transferring and fusing models with each other.

The main contribution of this paper is to introduce the mutual knowledge transfer(MKT)[7] into DFL. Using soft predictions to guide the train of the other instead of averaging the parameters of model directly.

The experiments of this paper perform on the MNIST, Fashion-MNIST, CIFAR-10 and CIFAR-100 data sets. Compared with the baseline DFL methods with model averaging, Combo and FullAvg, especially when the training data are not iid, the proposed algorithm gain significant improvement.

As far as I am concerned, there is no open source code of the algorithm of this paper.

The algorithm Def-KT has been revised in this paper and the experimental result shows its advantages compared with centralized methods.

When revising the method, I found that the selection of clients is random and it would sometimes lead to the decrease of accuracy if the received model is backward and lack of training. As a result, I bring up a client selection method to find the approciate client to learn from. The experiment shows that this method brings 4% accuracy improvement compared with Def-KT.

The contrast experiments have been done to test whether the other setting of Def-KT is useful and effective.

2 Method

2.1 Overview

The algorithm has two main stages. The first stage is to select Q clients from K clients(2Q«K) randomly. One thing should be noted, in the real IOT environment, most devices would not willing to join in FL training unless they are plugged-in or charged. The Q clients would train with their own private data in this period. The second stage is to randomly select anther Q clients. Every selected client would received one of previous Q models. And then, the client’s own model and the received model will compute the soft predictions of the local data. Next, the two model both perform training. In this training, the Kullback-Leibler(KL) divergence which quantifies the match of the soft predictions of the two model has been added in the loss function. By doing this, the two model can transfer the knowledge they obtain to the other.

Procedure 1 Def-KT

```
Input: initial parameters $w_0$
Initialization: All clients are initialized with the same model with parameters $w_0$
for $t = 0, \ldots, T$ do
    Randomly select a set $\mathcal{I}_t^A = \left\{k_t^1, k_t^2, \ldots, k_t^Q\right\}$ of clients.
    Randomly select another set $\mathcal{I}_t^B = \left\{k_t^{Q+1}, k_t^{Q+2}, \ldots, k_t^{2Q}\right\}$ of clients. )
    for $j \in \{1, \ldots, Q\}$ in parallel do
        $$\tilde{\boldsymbol{w}}_t^{k_t^j} \leftarrow \mathrm{SGD}_{B_1,M}\left(\boldsymbol{w}_t^{\boldsymbol{k}_t^j}, \mathcal{D}_{k_t^j}\right).$$
        The $k_t^j$-th client stores $\tilde{w}_t^{k_t^j}$ as its local model and transmits $\tilde{w}_t^{k_t^j}$ to the $k_t^{j+Q}$-th client.
    end
    for $j \in \{1, \ldots, Q\}$ in parallel do
        The $k_t^{j+Q}$-th client does:
        $\{\mathcal{B}_l, l = 1, \ldots L\} \leftarrow$ split $\mathcal{D}_{k_i^{j+Q}}$ into minibatches of size $B_2$
    end
    for $e = 1, \ldots, E$ do
        for $l \in \{1, \ldots, L\}$ do
            Compute soft predictions: $\mathcal{P}_{1,l} = \text{model}\left(\mathcal{B}_l, \tilde{w}_t^{k_t}\right)$
            Compute soft predictions: $\mathcal{P}_{2,l} = \text{model}\left(\mathcal{B}_l, \boldsymbol{w}_t^{\mathcal{K}_t^+}\right)$.
            Update $\tilde{w}_t^{k_t^j}$ : $\tilde{w}_t^{k_t^j} \leftarrow \tilde{w}_t^{k_t^j} - \eta_1 \dfrac{\partial \operatorname{Loss}_1\left(\tilde{w}_t^{k_t^j}, \mathcal{B}_l, \mathcal{P}_{2,l}\right)}{\partial \tilde{w}_t^{k_t^j}}$
            Update $w_t^{\boldsymbol{k}_t^{j+Q}}$ : $\boldsymbol{w}_t^{\boldsymbol{k}_t^{j+e}} \leftarrow \boldsymbol{w}_t^{\boldsymbol{k}_t^{j+e}} - \eta_2 \dfrac{\partial \operatorname{Loss}_2\left(\boldsymbol{w}_t^{\boldsymbol{k}_t^{j+e}}, \mathcal{B}_l, \mathcal{P}_{1,l}\right)}{\partial \boldsymbol{w}_t^{\boldsymbol{k}_t^{j+e}}}$
        end
    end
    Store $\,_t^{k_t^j}$ as the local model at the $k_t^{j+Q}$-th client.
end
```

2.2 Feature extraction

2.3 Loss function

To transfer the knowledge between each other, the two models try to imitate the output of the other one by minimizing the loss functions given as:

$$\operatorname{Loss}_1\left(\tilde{\mathbf{w}}_t^{k_t^j}, \mathcal{B}_l, \mathcal{P}_{2,l}\right) = L_C\left(\mathcal{P}_{1,l}, \mathcal{Y}_l\right) + D_{KL}\left(\mathcal{P}_{2,l} \| \mathcal{P}_{1,l} \ \ (1)\ \right)$$

and

$$\operatorname{Loss}_2\left(\mathbf{w}_t^{k_t^{j+Q}}, \mathcal{B}_l, \mathcal{P}_{1,l}\right) = L_C\left(\mathcal{P}_{2,l}, \mathcal{Y}_l\right) + D_{KL}\left(\mathcal{P}_{1,l} \| \mathcal{P}_{2,l}\right) \ (2)$$

respectively, where $\mathcal{Y}_l = \left\{y_z^l\right\}_{z=1}^{B_2}$ denotes the set of true labels of the data samples in $\mathcal{B}_l$ with $y_z^l \in \{1, 2, \ldots, C\}$ being the true label of the z th sample in $\mathcal{B}_l$. In (1) and (2), $\mathcal{P}_{1,l}$ and $\mathcal{P}_{2,l}$are the soft predictions produced as

$$\mathcal{P}_{1,l} = \{\mathbf{p}_{1,l,z}\}_{z=1}^{B_2} = \text{model}\left(\mathcal{B}_l, \tilde{\mathbf{w}}_t^{\tilde{k}_t^j}\right) \quad \forall l$$

$$\mathcal{P}_{2,l} = \{\mathbf{p}_{2,l,z}\}_{z=1}^{B_2} = \text{model}\left(\mathcal{B}_l, \mathbf{w}_t^{k_t^{j+Q}}\right) \quad \forall l$$

where $\mathbf{p}_{1,l,z} = \left[p_{1,l,z}^1, p_{1,l,z}^2, \ldots, p_{1,l,z}^C\right]$ and $\mathbf{p}_{2,l,z} =$

$$L_C\left(\mathcal{P}_{1,l}, \mathcal{Y}_l\right) = -\sum_{z=1}^{B_2} \mathbf{h}^T\left(y_z^l\right) \log \mathbf{p}_{1,l,z}$$

$$L_C\left(\mathcal{P}_{2,l}, \mathcal{Y}_l\right) = -\sum_{z=1}^{B_2} \mathbf{h}^T\left(y_z^l\right) \log \mathbf{p}_{2,l,z}$$

where $\mathbf{h}(y)$ is the $C \times 1$ one-hot vector and all the elements in $\mathbf{h}(y)$ are 0 except that the y th element is 1. The second term in (1) and (2) is the Kullback-Leibler (KL) divergence that quantifies the match of the soft predictions of the two networks, which is given as

$$D_{KL}\left(\mathcal{P}_{2,l} \| \mathcal{P}_{1,l}\right) = \sum_{z=1}^{B_2} \sum_{c=1}^{c} p_{2,l,z}^c \log \frac{p_{2,l,z}^c}{p_{1,l,z}^c}$$

$$D_{KL}\left(\mathcal{P}_{1,l} \| \mathcal{P}_{2,l}\right) = \sum_{z=1}^{B_2} \sum_{c=1}^{c} p_{1,l,z}^c \log \frac{p_{1,l,z}^c}{p_{2,l,z}^c}.$$

After updating $\tilde{\mathbf{w}}_t^{k_t^j}$ and $\mathbf{w}_t^{k_t^{j+Q}}$ by minimizing the loss functions in (2) and (3) through E passes over all the minibatches of data set $\mathcal{D}_{k_t^{j+Q}}$, the k_t^{j+Q} th client stores the resulting model $\tilde{\mathbf{w}}_t^{j_t^j}$ as its local model, which incorporates the knowledge of both models participating in the MKT process. To make it clear, Fig. 2 presents the schematic of a round of the proposed Def-KT algorithm.

3 Implementation details

3.1 Comparing with released source codes

As far as I am concerned, there is no related source code available. The following experiments are done according to the description of this paper.

3.2 Experimental environment setup

Fashion-MNIST, a universal images data set, is used to perform the following experiments. This data set consist of 70k 28*28 images of fashion items from ten classes. There are 60k images in the training set and 10k images in the test set respectively.

To construct the heterogeneous data set based on Fashion-MNIST, I process the data set with the following steps.

First, all feature - label pairs are grouped by the labels. Second, every client was randomly given heterogeneity(h=8) numbers which are from 0 to 9. Third, the clients randomly select the data from the corresponding groups according to the given numbers.

In the experiments, a convolution neural network(CNN) is adopted. The CNN train on the Fashion-

MNIST and heterogeneous Fashion-MNIST[8], and it contains convolution layers, batchnorm layers, relu layers and linear layers, which has 29034 parameters.

The CNN model is tested on the Fashion-MNIST data set in the IID setting and the non-IID setting(h=8), where the number of clients is fixed at $K = 10$. In each round, the number of training passes in the local updating is set as $M = 10$. An SGD optimizer with momentum $= 0.5$, learning rate $= 0.01$, and batchsize $= 200$ is applied for the IID setting and the non-IID setting(h=8). An SGD optimizer with momentum $= 0.5$, learning rate $= 0.01$, and batchsize $= 200$ is applied for the non-IID setting(h=8).

3.3 Main contributions

To perform the DFL experiments, a scheduling matrix is used to simulate the parallel computing. The contrast experiments are done to verify the performance of Def-KT and Def-KT with selection. Two experiments also been conducted to test whether the KL part is useful and whether the model replacement improve the performance.

Procedure 2 Def-KT with selection

```
Input: initial parameters $w_0$
Initialization: All clients are initialized with the same model with parameters $w_0$ for $t = 0, \ldots, T$ do
    Randomly select a set $\mathcal{I}_t^A = \left\{k_t^1, k_t^2, \ldots, k_t^Q\right\}$ of clients.
    for $j \in \{1, \ldots, Q\}$ in parallel do
        $$\tilde{\boldsymbol{w}}_t^{k_t^j} \leftarrow \mathrm{SGD}_{B_1,M}\left(\boldsymbol{w}_t^{k_t^j}, \mathcal{D}_{k_t^j}\right).$$
        The $k_t^j$-th client stores $\tilde{w}_t^{k_t^j}$ as its local model and transmits $\tilde{w}_t^{k_t^j}$ to the $k_t^{j+Q}$-th client.
    end
    Find the data with maximum variance and send out.
    Look for the appropriate clients of the previous clients with highest accuracy on these data.
    Select another set $\mathcal{I}_t^B = \left\{k_t^{Q+1}, k_t^{Q+2}, \ldots, k_t^{2Q}\right\}$ of clients according to the selection.
    for $j \in \{1, \ldots, Q\}$ in parallel do
        The $k_t^{j+Q}$-th client does:
        $\{\mathcal{B}_l, l = 1, \ldots L\} \leftarrow$ split $\mathcal{D}_{k_i^{j+Q}}$ into minibatches of size $B_2$
    end
    for $e = 1, \ldots, E$ do
        for $l \in \{1, \ldots, L\}$ do
            Compute soft predictions: $\mathcal{P}_{1,l} = \text{model}\left(\mathcal{B}_l, \tilde{w}_t^{k_t}\right)$
            Compute soft predictions: $\mathcal{P}_{2,l} = \text{model}\left(\mathcal{B}_l, \boldsymbol{w}_t^{\mathcal{K}_t^+}\right)$.
            Update $\tilde{w}_t^{k_t^j}$ : $\tilde{w}_t^{k_t^j} \leftarrow \tilde{w}_t^{k_t^j} - \eta_1 \frac{\partial \mathrm{Loss}_1\left(\tilde{w}_t^{k_t^j}, \mathcal{B}_l, \mathcal{P}_{2,l}\right)}{\partial \tilde{w}_t^{k_t^j}}$
            Update $w_t^{k_t^{j+Q}}$ : $\boldsymbol{w}_t^{k_t^{j+e}} \leftarrow \boldsymbol{w}_t^{k_t^{j+e}} - \eta_2 \frac{\partial \mathrm{Loss}_2\left(\boldsymbol{w}_t^{k_t^{j+e}}, \mathcal{B}_l, \mathcal{P}_{1,l}\right)}{\partial \boldsymbol{w}_t^{k_t^{j+e}}}$
        end
    end
    Store $_t^{k_t^j}$ as the local model at the $k_t^{j+Q}$-th client.
end
```

4 Results and analysis

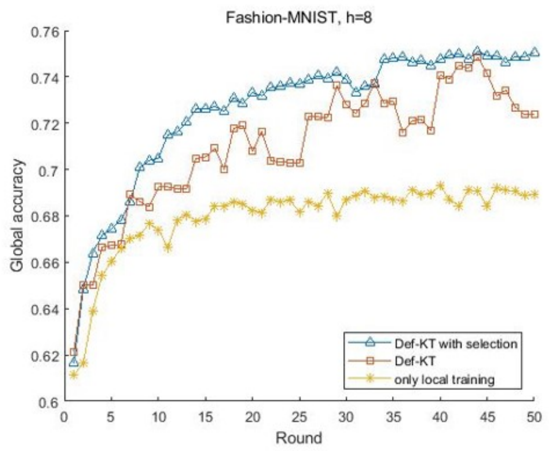

Figure 1: Experimental results

5 Results and analysis

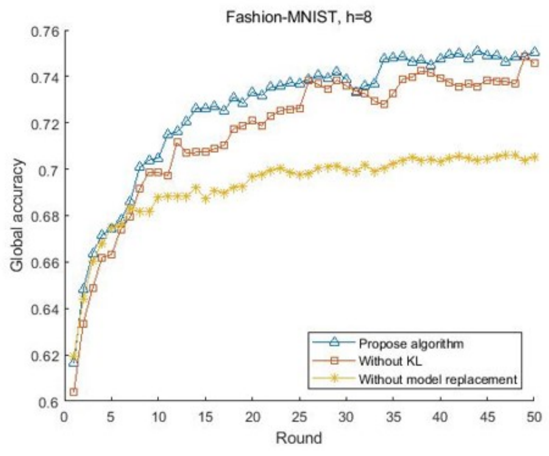

Figure 2: Experimental results

The results show that Def-KT can perform better in the DFL problem compared with the only perform local training one. And with the selection, the performance can get even better. The results also show that the two experimental settings are useful and effective.

6 Conclusion

In this article, I revise the Def-KT of ”Decentralized Federated Learning via Mutual Knowledge Transfer” and bring up a client selection to improve it. I run experiments on the Fashion-MNIST, which demonstrates the superiority of Def-KT algorithm over the baseline and the Def-KT with selection can achieve higher accuracy. The experiments also verify the efficiency of experimental settings.

References

[1] LI C, LI G, VARSHNEY P K. Decentralized Federated Learning via Mutual Knowledge Transfer[J]. IEEE Internet of Things Journal, 2020, 9: 1136-1147.

[2] MCMAHAN H B, MOORE E, RAMAGE D, et al. Communication-Efficient Learning of Deep Networks from Decentralized Data[C]//International Conference on Artificial Intelligence and Statistics. 2016.

[3] KAIROUZ P, MCMAHAN H B, AVENT B, et al. Advances and Open Problems in Federated Learning [J]. Found. Trends Mach. Learn., 2019, 14: 1-210.

[4] LI T, SAHU A K, TALWALKAR A S, et al. Federated Learning: Challenges, Methods, and Future Directions[J]. IEEE Signal Processing Magazine, 2019, 37: 50-60.

[5] KARIMIREDDY S P, KALE S, MOHRI M, et al. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning[C]//International Conference on Machine Learning. 2019.

[6] XIE M, LONG G, SHEN T, et al. Multi-Center Federated Learning[J]. ArXiv, 2020, abs/2108.08647.

[7] ZHANG Y, XIANG T, HOSPEDALES T M, et al. Deep Mutual Learning[J]. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2017: 4320-4328.

[8] XIAO H, RASUL K, VOLLGRAF R. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms[J]. ArXiv, 2017, abs/1708.07747.