FedPrune: Personalized and Communication-Efficient Federated Learning on Non-IID Data

Yang Liu[1], Yi Zhao[1(✉)], Guangmeng Zhou[1], and Ke Xu[1,2,3]

[1] Department of Computer Science and Technology, Tsinghua University, Beijing, China
{liuyang19,zgm19}@mails.tsinghua.edu.cn, {zhao_yi,xuke}@tsinghua.edu.cn
[2] Beijing National Research Center for Information Science and Technology (BNRist), Beijing, China
[3] Peng Cheng Laboratory (PCL), Shenzhen, China

Abstract. Federated learning (FL) has been widely deployed in edge computing scenarios. However, FL-related technologies are still facing severe challenges while evolving rapidly. Among them, statistical heterogeneity (i.e., non-IID) seriously hinders the wide deployment of FL. In our work, we propose a new framework for communication-efficient and personalized federated learning, namely *FedPrune*. More specifically, under the newly proposed FL framework, each client trains a converged model locally to obtain critical parameters and substructure that guide the pruning of the network participating FL. FedPrune is able to achieve high accuracy while greatly reducing communication overhead. Moreover, each client learns a personalized model in FedPrune. Experimental results has demonstrated that FedPrune achieves the best accuracy in image recognition task with varying degrees of reduced communication costs compared to the three baseline methods.

Keywords: Federated learning · Private preserving · Network pruning

1 Introduction

Federated learning (FL) [7,11] is a new machine learning paradigm that is already widely used in personal devices and financial enterprises. FL has been widely accepted as an artificial intelligence (AI) application that protects the data privacy of users [16]. While FL promises better privacy and efficiency, there are still two major challenges [8] in FL. The first one is the significant communication overhead, which has hampered the development of federated learning. In fact, the federated network is likely to be composed of many clients and the communication in the network is very frequent and more time consuming compared to local computing. The second challenge is the statistical heterogeneity, meaning that the distribution of data across clients is non-IID (non-identically independently distributed). The data at the client side may be very different

T. Mantoro et al. (Eds.): ICONIP 2021, CCIS 1516, pp. 430–437, 2021.
https://doi.org/10.1007/978-3-030-92307-5_50

in size and type. In this work, we focus on both challenges and thus propose a framework that can jointly take on them.

In our work, we propose a new framework for communication-efficient and personalized federated learning, namely *FedPrune*. Specifically, each client trains a converged model locally to obtain critical parameters and substructure that guide the pruning of the network participating federated learning. Clients with large differences in data distribution do not interfere with each other, while clients with similar data distribution can enhance each other. Finally, a personalized model will be learned at each client. We show that FedPrune is able to achieve high accuracy while greatly reducing communication overhead. Moreover, it only requires negligible computational and storage costs. We conduct experiments on the MNIST, CIFAR-10 and CIFAR-100 datasets and compared FedPrune with FedAvg [11], FedProx [9] and LG-FedAvg [10]. The experimental results show that FedPrune is significantly better than the compared methods in terms of accuracy and communication cost on non-IID data.

2 Related Work

AI has been integrated into many fields, such as mobile social networks [15] and smart cities [17]. To further protect user privacy, academia and industry propose to use federated learning [8,11] to achieve intelligence. However, Zhao et al. [18] shows that non-IID data distribution may significantly reduce the prediction accuracy of FL. FedProx [9] solves this problem by adding regularization terms to the local optimization so that the local model does not change too much compared to the global model. Model personalization is a worthwhile approach to tackle statistical heterogeneity. Jiang et al. [6] introduce the MAML [3] algorithm in the field of meta learning into federated learning to realize the personalization of models on each client. Vahidian et al. [13] obtain personalized models by structured pruning and unstructured pruning, but introduce hyperparameters that are very dependent on the network structure, making it difficult to tune and deploy in practice.

How to reduce the communication overhead of federated learning is another problem that puzzles researchers. Previous work [1,5,7,12] reduces the size of the model transferred between the client and the server through data compression techniques such as sketching, sparsification and quantization. Wang et al. [14] dynamically tunes the frequency of updating the model according to the available communication resources.

3 Design of FedPrune

We denote N clients by $\mathcal{C} = \{C_1, ..., C_N\}$. We denote w_g as the weights of the global model, and w_k $(k = 1, ..., N)$ as the local model weights on each client C_k. We let $\{w_{ij,k}\}$ denote the weights of the connections between pairs of neurons $n_{i,k}$ and $n_{j,k}$ in the model w_k. We denote $\Omega_{ij,k}$ as the importance value for each parameter $\{w_{ij,k}\}$. We use the superscript t, e.g., w_i^t, to indicate the weights

learned in round t. Each client C_k learns a local mask $m_k \in \{0,1\}^{w_k}$, which indicates whether the weights are pruned or not. In a local mask, a value of 0 means that the corresponding weight is pruned, and 1 means vice versa.

3.1 Estimating Parameter Importance

According to *the lottery ticket hypothesis* [4], there always exists the optimal sub-network, also called the winning ticket, that can achieve similar performance as the original network. That is, if the sub-networks of each client participate in federated learning, they can not only achieve the performance of the original model, but also avoid the interference of the model parameters of other clients.

In this work, we adapt the MAS [2] algorithm, which measures the importance of parameters, to the federated learning scenario. Each client trains a model locally using local data before participating in federated learning. The model is considered to have learned an approximation F to the true function $\bar{F}$ when it reaches a local optimum. We characterize the importance of a parameter in the network in terms of the sensitivity of the function F to that parameter. When the input is x_d, the output of the function is $F(x_d; w)$. Applying a small perturbation $\delta = \{\delta_{ij}\}$ to the parameters $w = \{w_{ij}\}$, the output of the function can be approximated by:

$$F(x_d; w + \delta) - F(x_d; w) \approx \sum_{i,j} g_{ij}(x_d)\delta_{ij} \tag{1}$$

where $g_{ij}(x_d) = \frac{\partial(F(x_d;w))}{\partial w_{ij}}$ is the gradient of the function F with respect to the parameter w_{ij} at the data point x_d. δ_{ij} is the small perturbation applied to the parameter w_{ij}. Assuming that δ_{ij} is a constant, we can use the magnitude of the gradient g_{ij} to characterize the importance of the parameter. We accumulate the gradients obtained from all the input data and sum up to obtain the importance weight Ω_{ij} for parameter w_{ij}:

$$\Omega_{ij} = \frac{1}{N_{dp}} \sum_{d=1}^{N_{dp}} \|g_{ij}(x_d)\| \tag{2}$$

where N_{dp} is the number of input data points.

3.2 Training Process of FedPrune

The details of FedPrune are described in Algorithm 1. Typically, the training process of FedPrune is as follows:

Prior to training for federated learning, the server initializes the global model w_g^0 and sends that model to each client. Once the global model is received, each client trains the local model $w_k^{'}$ as a way to obtain the masks needed to prune the models involved in federated learning. Specifically, we can obtain the importance of the parameters by the approach introduced in Sect. 3.1. Given a target pruning

Algorithm 1: Training of FedPrune. K is the random sampling rate, $\mathcal{B}$ is the set of local mini-batches, η is the learning rate, and $l(\cdot)$ is the loss function.

Server executes: `// Run on the server`
initialize the global model w_g^0;
ClientGetMask(w_g^0); `// executed in parallel`
for *each round* $t = 1$ **to** T **do**
  $k \leftarrow \max(N \times K, 1)$;
  $\mathcal{S}_t \leftarrow \{C_1, ..., C_k\}$;
  for *each client* $k \in S_t$ ***in parallel*** **do**
    $w_k^{t+1} \leftarrow$ClientUpdate(w_g^t);
  end
  $w_g^{t+1} \leftarrow$ aggregate subnetworks of clients, w_k^{t+1}, and average the intersection of them;
end

ClientGetMask(w_g^0): `// Run on client` k
Train the local model $w_k^{'}$ for E_l epochs based on w_g^0;
Compute $\{\Omega_{ij,k}\}$ by Eq.(2);
m_k, the mask for w_k, is obtained based on $\{\Omega_{ij,k}\}$ and target pruning rate p;

ClientUpdate(w_g^t): `// Run on client` k
$w_k^t \leftarrow w_g^t \odot m_k$;
$\mathcal{B} \leftarrow$split local training data into batches;
for *each local epoch from 1* **to** E **do**
  for *batch* $b \in \mathcal{B}$ **do**
    $w_k^{t+1} \leftarrow w_k^t - \eta \nabla_{w_k^t} l(w_k^t; b) \odot m_k$;
  end
end
return w_k^{t+1} to server;

ratio p, a binary mask of the same size as the model is derived. The process of client training a local model to obtain a mask is asynchronous to the whole process of federated learning, and clients who have already obtained a mask can start federated training earlier.

Given the round t, the server samples a random set of clients $\mathcal{S}$ and distributes a global model to each of them. Note that C_k trains $w_g^t \odot m_k$, the global model w_g^t pruned by the mask m_k, as the initial model for this round, instead of training the global model directly. Then C_k performs training for E epochs with the local data, and then uploads the updated w_k^{t+1} to the server.

At the end of the round, the server performs aggregation on all received local models (i.e., w_k^{t+1}). Different from FedAvg, we only take the average on the intersection of unpruned parameters for each client, just like the **By-unit** approach described in Zhou et al. [19]. This aggregation method allows networks with different structures that imply large differences in data distribution not

to interfere with each other. Meanwhile, this approach enables networks with similar structures to further enhance each other.

4 Experiments

4.1 Experimental Setup

We conduct an empirical study of FedPrune and compare it with classical FL algorithms i.e. FedAvg [11], FedProx [9] and LG-FedAvg [10]. Our experimental studies are conducted over three datasets, MNIST, CIFAR-10 and CIFAR-100.

To evaluate each method in terms of statistical heterogeneity, we divide the data in the same way as in McMahan et al. [11]. The architecture we used for MNIST is a CNN with two 5×5 convolution layers, a fully connected layer with 50 units, and a final softmax output layer. We add ReLU activation functions to all layers except the last one. For CIFAR-10 and CIFAR-100 datasets we use LeNet-5 architecture. In all experiments, we have 100 clients, each with local batch size 10 and local epoch 5. In addition, we use an SGD optimizer with learning rate and momentum of 0.01 and 0.5, respectively. For FedPrune, we set the number of epochs for the local model $E_l = 50$. For FedProx, we show the experimental results at the coefficient of the regularization term $\mu = 0.01$.

We compare FedPrune with three methods, i.e., FedAvg, FedProx and LG-FedAvg. FedAvg is a classical federated learning method. FedProx improves on FedAvg by adding a regularisation term called *proximal term*. In LG-FedAvg, each client learns a compact local representations and all clients learn a global model together. We use the classification accuracy of the test data on each client to evaluate the performance of personalization and report the average accuracy of all clients. We use the number of parameters of the model to measure the communication overhead.

4.2 Results and Analysis

We compare the results of our proposed algorithms against several baselines, as shown in Table 1.

Accuracy: We show the accuracy of the model after pruning 30%, 50% and 70% of the parameters in Table 1. As can be seen from the table, the variation of accuracy with the pruning rate is not drastic and the accuracy always maintains at a high level. Even with 70% of the parameters pruned, the accuracy of the FedPrune algorithm is still much higher than that of other methods. This result illustrates that more parameters of the model do not mean better performance.

Overhead: As seen in Table 1, FedPrune achieves communication efficiency with a small loss of accuracy. In the experiments on FedPrune, the client needs to train a local model for 50 epochs as a way to get the critical parameters and substructure, which seems to impose some computational overhead on the client. However, the computational overhead of this part is only 10% or less compared

Table 1. Comparing the classification accuracy and communication overhead of FedPrune against several baselines.

Dataset	Method	Acc	% Pruned param	Communication cost
MNIST	FedAvg	98.75%	0	1.75 GB
	FedProx	98.75%	0	1.75 GB
	LG-FedAvg	98.20%	0	1.71 GB
	FedPrune	**99.39%**	30%	**1.23 GB**
	FedPrune	**99.49%**	50%	**0.88 GB**
	FedPrune	**99.39%**	70%	**0.53 GB**
CIFAR-10	FedAvg	49.21%	0	4.96 GB
	FedProx	50.21%	0	4.96 GB
	LG-FedAvg	76.28%	0	4.54 GB
	FedPrune	**80.68%**	30%	**3.47 GB**
	FedPrune	**81.02%**	50%	**2.48 GB**
	FedPrune	**79.63%**	70%	**1.49 GB**
CIFAR-100	FedAvg	14.91%	0	5.57 GB
	FedProx	13.13%	0	5.57 GB
	LG-FedAvg	47.60%	0	5.17 GB
	FedPrune	**84.91%**	30%	**3.90 GB**
	FedPrune	**83.95%**	50%	**2.79 GB**
	FedPrune	**81.98%**	70%	**1.67 GB**

to the whole federated learning process. For the vast majority of edge devices, it is acceptable. Theoretically, the storage overhead of FedPrune is small. We need only 1 bit to encode the mask per parameter. For example, in our experiments, the network size of LeNet-5 for CIFAR-100 is 0.28 MB. The overhead of adding a mask to this network is about 8.7 KB. A parameter is typically represented by 4 bytes, and adding a mask results in an additional storage overhead of 1/32 of the initial model size, which is ideal for edge computing devices with small storage space. Note that it is not necessary for the local model and the model participating in federated learning to exist simultaneously.

Sensitivity Evaluation: We will study the variation of accuracy with target pruning ratio p. Figure 1 plots the average test accuracy over all clients versus various pruning percentages. At the beginning, the accuracy of the model keep improving as the number of parameters being pruned increases. As we expect, in federated learning, too many parameters are not beneficial for model training, but lead to mutual interference among clients. As the number of parameters being pruned continues to increase, the accuracy of the model begins to slowly decrease. This is because the critical parameters are also pruned and the optimal substructure is corrupted. Surprisingly, however, even at very high pruning ratio, the accuracy does not drop dramatically and remains even higher than

baselines. From the figure we can see that for CIFAR-10, the accuracy of the classification can still reach 73.94% and 64.75% when the pruning ratio is 95% and 99%, respectively. For CIFAR-100, the accuracy reaches 76.01% and 68.67% at the same pruning ratio, respectively. This result sufficiently illustrates that our method does find the critical parameters and optimal substructure, which guarantee a good performance even when the model is extremely compressed.

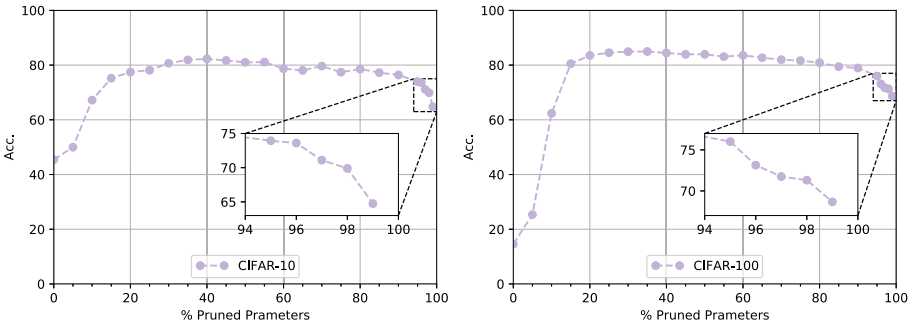

Fig. 1. Average test accuracy of FedPrune over all clients for the CIFAR-10 (left) and CIFAR-100 (right) datasets.

5 Conclusion

In this work, we propose a federated learning framework, FedPrune, that maintains a high level of accuracy while greatly reducing communication overhead. Moreover, the framework is easy to implement and has limited computational and storage overhead, making it suitable for deployment in mobile and edge computing devices. In addition, FedPrune has only one more hyperparameter than FedAvg, target pruning ratio p, making it easy to tune and deploy to production environments.

References

1. Alistarh, D., Grubic, D., Li, J., Tomioka, R., Vojnovic, M.: QSGD: Communication-efficient SGD via gradient quantization and encoding, pp. 1709–1720 (2017)
2. Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., Tuytelaars, T.: Memory aware synapses: learning what (not) to forget. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018, Part III. LNCS, vol. 11207, pp. 144–161. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01219-9_9
3. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 70, pp. 1126–1135. PMLR, 06–11 August 2017. http://proceedings.mlr.press/v70/finn17a.html

4. Frankle, J., Carbin, M.: The lottery ticket hypothesis: finding sparse, trainable neural networks. In: International Conference on Learning Representations (2019). https://openreview.net/forum?id=rJl-b3RcF7
5. Ivkin, N., Rothchild, D., Ullah, E., Braverman, V., Stoica, I., Arora, R.: Communication-efficient distributed SGD with sketching. In: Proceedings of NeurIPS, pp. 1–23 (2019)
6. Jiang, Y., Konečný, J., Rush, K., Kannan, S.: Improving Federated Learning Personalization via Model Agnostic Meta Learning. CoRR abs/1909.12488 (2019). http://arxiv.org/abs/1909.12488
7. Konečnỳ, J., McMahan, H.B., Yu, F.X., Richtárik, P., Suresh, A.T., Bacon, D.: Federated learning: strategies for improving communication efficiency. In: Proceedings of NeurIPS Workshop (2016)
8. Li, T., Sahu, A.K., Talwalkar, A., Smith, V.: Federated learning: challenges, methods, and future directions. IEEE Signal Process. Mag. **37**(3), 50–60 (2020)
9. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. In: Proceedings of MLSys (2018)
10. Liang, P.P., Liu, T., Ziyin, L., Salakhutdinov, R., Morency, L.P.: Think locally, act globally: Federated learning with local and global representations. arXiv preprint arXiv:2001.01523 (2020)
11. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Proceedings of AISTATS, pp. 1273–1282 (2017)
12. Reisizadeh, A., Mokhtari, A., Hassani, H., Jadbabaie, A., Pedarsani, R.: FedPAQ: a communication-efficient federated learning method with periodic averaging and quantization. In: Proceedings of AISTATS, pp. 2021–2031 (2020)
13. Vahidian, S., Morafah, M., Lin, B.: Personalized Federated Learning by Structured and Unstructured Pruning under Data Heterogeneity. CoRR abs/2105.00562 (2021), https://arxiv.org/abs/2105.00562
14. Wang, S., et al.: Adaptive federated learning in resource constrained edge computing systems. IEEE J. Sel. Areas Commun. (JSAC) **37**(6), 1205–1221 (2019)
15. Zhao, Y., et al.: TDFI: two-stage deep learning framework for friendship inference via multi-source information. In: IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, pp. 1981–1989 (2019). https://doi.org/10.1109/INFOCOM.2019.8737458
16. Zhao, Y., Xu, K., Wang, H., Li, B., Jia, R.: Stability-based analysis and defense against backdoor attacks on edge computing services. IEEE Netw. **35**(1), 163–169 (2021). https://doi.org/10.1109/MNET.011.2000265
17. Zhao, Y., Xu, K., Wang, H., Li, B., Qiao, M., Shi, H.: MEC-enabled hierarchical emotion recognition and perturbation-aware defense in smart cities. IEEE IoT J. 1 (2021). https://doi.org/10.1109/JIOT.2021.3079304
18. Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., Chandra, V.: Federated learning with non-iid data. arXiv preprint arXiv:1806.00582 (2018)
19. Zhou, G., Xu, K., Li, Q., Liu, Y., Zhao, Y.: AdaptCL: Efficient Collaborative Learning with Dynamic and Adaptive Pruning. CoRR abs/2106.14126 (2021), https://arxiv.org/abs/2106.14126