Report: Federal Learning with Statistical Heterogeneous

Ziyang Gong *
School of Statistics, Southwest University of Finance and Economics

Abstract

With the increasing number of mobile phones, wearable devices, and autonomous vehicles, the natural idea is to store data locally and push the computation from the centre server to the whole network's edge device, termed "Federal Learning". Although this method has obvious advantages in computational efficiency and data magnitude, it also has many challenges. This report will mainly discuss the challenge of statistical heterogeneity because the data stored on each device is collected by the individual devices/users. Moreover, we summarize some of the articles mentioned by Li et al. (2020a), and then recurrent part of the experimental results.

Keywords: Federal Learning, Statistical Heterogeneous, Multi-task Learning

1 Introduction

The number of mobile phones, wearable devices, and autonomous vehicles is increasing day by day, divided into several distributed networks. Standing in front of the scene of improving devices/users' experience by setting up models, making full use of these distributed networks is a challenging but exciting field. Based on the features of these distributed networks:

- The distributed networks have increasing computational power on each device, which may be a surplus for existing applications.
- Moreover, with the storage and wealthy sensors on each device (such as cameras, microphones, and GPS), it is easier to access a large amount of data; most of them are private.

With the above features of the distributed network, a natural idea is storing data locally and pushing the computation from the centre server to the whole network's edge device. This

*Email: meetziyang@outlook.com

method is called "Federal Learning". The term "Federal Learning" was first proposed by McMahan et al. (2017), that,

> The learning task is solved by a loose federation of participating devices (which we refer to as clients) coordinated by a central server.

In summary, the federal learning problem can be formulated as the following procedures (Kairouz et al., 2021):

1. Client Selection: The server samples from a set of clients meeting eligibility requirements.
2. Broadcast: The selected clients download the current model weights and a training program from server.
3. Client Computation: Each selected device locally computes an update to the model by executing the training program.
4. Aggregation: The server collects an aggregate of the device updates.
5. Model Update: The server locally updates the shared model based on the aggregated update computed from the clients that participated in the current round.

Through the above-mentioned multiple rounds of learning and communication methods, federated learning eliminates the need to aggregate all data on a single device, overcomes privacy and communication challenges in machine learning tasks, and allows machine learning model learning to be distributed on individual devices/users stored data.

The challenge of federal learning, as Li et al. (2020a) saying, are also various, such as communication efficiency, systems heterogeneity, statistical heterogeneity, and privacy.

In this report, we focus on the topic of statistical heterogeneity. In the application scenario of federated learning, the data stored on each device is collected by the individual devices/users, which may be significantly different from other devices/users, different devices/users' non-homologous data have different distribution characteristics. So the local data set of any specific devices/users cannot represent the overall distribution. Therefore, for federated learning, a core challenge is "how to deal with non-independent and identically distributed data from separated devices/users?".

In order to solve this heterogeneity challenges, an effective method is to perform personalized processing at the device, data, and model level to reduce heterogeneity and obtain a high-quality personalized model for each device, that is, personalized federated learning.

As Kulkarni et al. (2020) mentioned, the method to handle statistical heterogeneity can be classified as several techniques, such as averaging, adding user context, transfer learning, multi-task learning, meta-learning, knowledge distillation, base & personalization layers and the mixture of global and local models et al. The paper we discussed in this report which

mentioned by Li et al. (2020a) can be mainly divided into two parts according to the above standard,

1. Averaging:
    - Communication-EfficientLearning of Deep Networks from Decentralized Data (McMahan et al., 2017)
    - Federated Learning with Non-IID Data (Zhao et al., 2018)
2. Multi-task Learning:
    - Federated multi-task learning (Smith et al., 2017)
    - Variational Federated Multi-Task Learning (Corinzia et al., 2021)

then we will introduce the paper by the techniques topic, explore their methods advantage and disadvantages, and make recurrent experimental, etc.

2 Related Works

For convenience, we will introduce the universal symbols for the below federal learning methods in this section, and in order to unify the expression, we refer to the devices/users in the distributed network as clients, and the client for centralized computing is called a server. Suppose the distributed networks has the following property,

- the number of clients in the distributed networks is K,
- the dataset in client k is $\mathcal{D}_k = \{\mathbf{x}_k^i, y_k^i\}_{i=1}^{n_k}$, that there is n_k samples in client k,

and during the broadcast and client computation procedures of the federal learning, which can be also known as hyper-parameters,

- the fraction of used clients at each iteration is C,
- the number of federal learning rounds is T,
- the number of local updates is E,
- local batch size used at each learning iteration B,
- local learning rate η.

2.1 Averaging

FedAvg Method by McMahan et al. (2017) This method is a basic framework for the following few articles we will introduce, and it is also a classic training process in the field of federated learning. And the goal of this method is to minimize the following objective function:

$$\min_{\mathbf{w}} \ell(\mathbf{w}), \quad \text{where} \quad \ell(\mathbf{w}) = \sum_{k=1}^{m} \rho_k \ell_k(\mathbf{w}), \tag{1}$$

where, m is the total number of selected devices, $\rho_k \geq 0$ and $\sum_{k=1}^{m} \rho_k = 1$, and ℓ_k is the local objective function for the k-th device.

This method is relatively simple and is an extension of the SGD method. However, this method's idea is to decompose the global objective function into multiple local objective functions as implied by the objective function mentioned above. Moreover, this idea solves a core challenge in federal learning that can separately train the clients and depart from the need to access raw training data directly. The detaild algorithm is shown in Algorithm 1.

Algorithm 1: FedAvg: Federated Averaging Method by McMahan et al. (2017)

Input: data $\mathcal{D}_k$ from client $k = 1, 2, \ldots, K$, number of federated learning rounds T, fraction of clients used at each iteration for each client C, number of local updates E, local batch size used at each learning iteration B, local learning rate η

Output: estimated model weight w

initialize model weight be $w^{(0)}$;
for *round* $t = 1, 2, \ldots, T$ **do**
  let $S^{(t)}$ be the random selected $(C \times K)$ clients;
  for *client* k *in* $S^{(t)}$ **do**
    $w_k^{(t)} \leftarrow w^{(t-1)}$;
    for *local epoch* $e = 1, 2, \ldots, E$ **do**
      for *batch* b *with batch size* B *in* $\mathcal{D}_k$ **do**
        $w_k^{(t)} \leftarrow w_k^{(t)} - \eta \nabla \ell(w_k^{(t)}; b)$;
      end
    end
  end
  $w^{(t)} \leftarrow \frac{\sum_{k \in S^{(t)}} n_k w_k^{(t)}}{\sum_{k \in S^{(t)}} n_k}$;
end
$w \leftarrow w^{(T)}$;

Another important finding in this paper was that we must average the models in different clients with the same initial parameters. Hence, this method enforces each local model parameters during the reporting procedure during each round to be the same.

The FedAvg method is a version of an implementation of classic federal learning framework, which build a global model for each client in the distributed networks. This method is simple but effective and has also been proved to achieve convergence in IID and Non-IID by Li et al. (2020b).

FedAvg (data-sharing) by Zhao et al. (2018) Zhao et al. (2018) propose a data-sharing strategy to improve FedAvg with non-IID data by creating a small subset of globally shared data between all the edge devices. In addition to the proposed data sharing method, this paper also proves that the classical federal learning algorithm FedAvg will perform poorly for the non-IID data. This paper also defines "Weight Divergence" and indicates that it is the main reason for the performance degradation of FedAvg for the non-IID data.

The proposed strategy shows exciting results, though it sacrifices part of personal privacy and transmission efficiency. As stated in the article, when 5% of all data is shared, nearly 30% can improve the performance of the Cifar10 dataset.

2.2 Multi-task Learning

MOCHA Method by Smith et al. (2017) This method was inspired by multi-task learning, can allow for personalization by learning separate but related models for each device while leveraging a shared representation. And MOCHA method is based on a general multi-task Learning framework that based on the below objective function

$$\min_{\mathbf{W},\mathbf{\Omega}} \left\{ \sum_{k=1}^{K} \ell_t \left(\mathbf{w}_k; \mathcal{D}_k \right) + \mathcal{R}(\mathbf{W}, \mathbf{\Omega}) \right\}, \tag{2}$$

where $\mathbf{W} = (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K)$ is the matrix of the t-th column is the weight vector for the client k, and $\mathbf{\Omega}$ models relationships amongst clients, which either known a priori or estimated while simultaneously learning clients weight, and the penalty term can be expressed as the bi-convex formulation

$$\mathcal{R}(\mathbf{W}, \mathbf{\Omega}) = \lambda_1 \operatorname{tr}\left(\mathbf{W}\mathbf{\Omega}\mathbf{W}^T\right) + \lambda_2 \|\mathbf{W}\|_F^2, \tag{3}$$

with constants $\lambda_1, \lambda_2 > 0$.

Multi-task learning problems differ based on their assumptions on $\mathcal{R}$, which takes $\mathbf{\Omega}$ as input and promotes some suitable structure amongst the tasks. Moreover, MOCHA method used the below penalty term,

$$\mathcal{R}(\mathbf{W}, \mathbf{\Omega}) = \lambda \left(\frac{1}{\sigma^2} \|\mathbf{W}\|^2 + \operatorname{tr}\left(\mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T\right) \right), \mathbf{\Omega} \in \mathcal{Q} = \{\mathbf{Q} \mid \mathbf{Q} \succeq 0, \operatorname{tr}(\mathbf{Q}) = 1\}, \tag{4}$$

which under the assumption that weight matrix has a probabilistic priors, the detailes can be seen in Zhang and Yeung (2010).

With the assumptions established by Smith et al. (2017), MOCHA can be distributed solved with below data-local quadratic subproblems, for client k,

$$\min_{\Delta\alpha_k} \mathcal{G}_k^{\sigma'} \left(\Delta\alpha_k; \mathbf{v}_k, \alpha_k \right) = \sum_{i=1}^{n_k} \ell_k^* \left(-\alpha_k^i - \Delta\alpha_k^i \right) + \langle \mathbf{w}_k(\alpha), \mathbf{X}_k \Delta\alpha_k \rangle + \frac{\sigma'}{2} \left\| \mathbf{X}_k \Delta\alpha_k \right\|_{\mathbf{M}_k}^2 + c(\alpha) \tag{5}$$

where ℓ_k^* and $\mathcal{R}^*$ are the conjugate dual functions of ℓ_k and $\mathcal{R}$, respectively, α_k^i is the dual

variable for the data point $(\mathbf{x}_k^i, y_k^i)$, $c(\alpha) = \frac{1}{m}\mathcal{R}^*(\mathbf{X}\alpha)$, and $\mathbf{M}_k$ is the k-th diagonal block of the symmetric positive define matrix M. Suppose $\mathbf{X} = \text{diag}\,(\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_K)$, then $\mathbf{W}$ can be found via $\mathbf{w}(\alpha) = \nabla\mathcal{R}^*(\mathbf{X}\alpha)$, where $\mathbf{w}_k(\alpha)$ is the k-th block in the vector $\mathbf{w}(\alpha)$.

The detaild algorithm is shown in Algorithm 2.

Algorithm 2: MOCHA: Federated Multi-task Learning Method by Smith et al. (2017)

Input: data $\mathcal{D}_k$ from client $k = 1, 2, \ldots, K$, number of federated learning rounds T

Output: estimated local clients model weights $\mathbf{W} = (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K)$

initialize $\alpha_k = \mathbf{0}, \mathbf{v}_k = \mathbf{0}, k = 1, 2, \ldots, K$;
for *round* $t = 1, 2, \ldots, T$ **do**
  for *client* $k = 1, 2, \ldots, K$ **do**
    $\Delta\alpha_k \leftarrow \arg\min_{\Delta\alpha_k} \mathcal{G}_k^{\sigma'}(\Delta\alpha_k; \mathbf{v}_k, \alpha_k)$;
    $\alpha_k \leftarrow \alpha_k + \Delta\alpha_k$;
    $\Delta\mathbf{v}_k \leftarrow \mathbf{X}_k\Delta\alpha_k$;
    $\mathbf{v}_k \leftarrow \mathbf{v}_k + \Delta\mathbf{v}_k$;
  end
  update $\mathbf{\Omega}$ based on $\mathbf{w}(\alpha)$ for latest α;
end
computes $\mathbf{w} = \mathbf{w}(\alpha)$ based on the lastest α;

This method has provable theoretical convergence guarantees for the considered objectives but is limited in its ability to scale to massive networks and is restricted to convex objectives.

VIRTUAL Method by Corinzia et al. (2021) MOCHA method is an effective paradigm for real-world datasets, though it has been applied only on convex models. In order to apply to the general non-convex situation, VIRTUAL has been proposed.

This method assumes a star-shaped Bayesian network with a server S with model parameters θ, as well as K clients with model parameters $\mathbf{W} = (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K)$. Following a Bayesian approach, this method assume a prior distribution over all network parameters $p(\theta, \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K)$. The posterior distribution over all parameters, given all datasets $\mathcal{D} = (\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_k)$ is

$$p(\theta, \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K \mid \mathcal{D}) \propto \frac{\prod_{k=1}^K p(\theta, \mathbf{w_k} \mid \mathcal{D}_i)}{p(\theta)^{K-1}}. \tag{6}$$

The posterior given above is in general intractable and this method uses an approximation inference scheme, which propose an expectation propagation (EP) method to calculate the posterior. This method define a proxy posterior distribution that factorizes into a server and a

client contribution for every client k as

$$q(\theta, \mathbf{W}) = \left(\prod_{k=1}^{K} s_k(\theta)\right) \left(\prod_{k=1}^{K} c_k\left(\mathbf{w}_k\right)\right), \tag{7}$$

which allows to perform a client update that is independent of other clients, and to perform a server update in aggregated posterior. Then the proxy pdf $s_k^{(t)}(\theta)$ and $c_k^{(t)}\left(\mathbf{w}_k\right)$ at round t and client k are found minimizing the variational free energy function

$$\begin{aligned} \ell_k\left(s_k(\theta), c_k\left(\mathbf{w}_k\right); p(\theta), p(\mathbf{w}_k)\right) = D_{KL}\left(s_k(\theta) \frac{s^{(t-1)}(\theta)}{s_k^{(t-1)}(\theta)} \middle\| p(\theta)^{\frac{1}{K}} \frac{s^{(t-1)}(\theta)}{s_k^{(t-1)}(\theta)}\right) \\ + D_{KL}\left(c_k\left(\mathbf{w}_k\right) \| p\left(\mathbf{w}_k\right)\right) - \mathbb{E}_{s^{(t)}(\theta), c_k(\mathbf{w}_k)} \log p\left(\mathcal{D}_k \mid \theta, \mathbf{w}_i\right) \end{aligned} \tag{8}$$

where $s^{(t)}(\theta) = s_k(\theta) \prod_{j \neq k} s_j^{(t-1)}(\theta)$.

For simplicity, this method uses a Gaussian meanfield approximation of the posterior, hence for server and client parameters, the factorization respectively is

$$s_k(\theta) = \prod_d \mathcal{N}\left(\theta_d \mid \mu_{kd}^s, \sigma_{kd}^s\right), \quad c_k\left(\mathbf{w}_k\right) = \prod_d \mathcal{N}\left(w_{kd} \mid \mu_{kd}^c, \sigma_{kd}^c\right). \tag{9}$$

The detaild algorithm is shown in Algorithm 3.

Although this method can handle non-convex models, it is expensive to generalize to large federated networks.

2.3 Summaries

All in all, we will here make a summary of the papers we discussed above, and the results can be seen in the Table 1.

3 Recurrent Experimental Results

Due to time and computing power constraints, here we only reproduce two papers (McMahan et al., 2017; Smith et al., 2017) that have official codes and are easy to implement, and we modify them based on the following code repositories:

- FedAvg (Jadhav, 2021): `https://github.com/AshwinRJ/Federated-Learning-PyTorch`
- MOCHA (gingsmith, 2021): `https://github.com/gingsmith/fmtl`

And, we will make certain modifications to them to make them fit our next data set and experimental settings.[1]

Dataset Description In order to generate non-IID data, we follow the datasets and experiment settings by Corinzia et al. (2021) to be used in the recurrent experiments, that,

[1]The recurrent experiment code is avaliable at `https://github.com/SignorinoY/overlord`.

Algorithm 3: VIRTUAL: Variational Federated Multi-task Learning Method by Corinzia et al. (2021)

Input: data $\mathcal{D}_k$ from client $k = 1, 2, \ldots, K$, prior distributions $p(\theta), \{p(\mathbf{w}_1), p(\mathbf{w}_2), \ldots, p(\mathbf{w}_K)\}$, number of federated learning rounds T, fraction of clients used at each iteration for each client C, number of local updates E, local batch size used at each learning iteration B, local learning rate η

Output: estimated posterior distributions $s_k(\theta), c_k(\mathbf{w}_k)$

initialize all pdfs $s_k^{(0)}(\theta), c_k^{(0)}(\mathbf{w}_k)$;

$s^{(0)}(\theta) \leftarrow \prod_i s_i^{(0)}(\theta)$;

for *round* $t = 1, 2, \ldots, T$ **do**

  let $S^{(t)}$ be the random selected $(C \times K)$ clients;

  for *client* k *in* $S^{(t)}$ **do**

    receive $s^{(t-1)}$ from server;

    $s_k^{(t)}(\theta), c_k^{(t)}(\mathbf{w}_k)$ joint optimization of ℓ_k with E epochs and B batch size;

    $\Delta_k^{(t)}(\theta) \leftarrow \frac{s_k^{(t)}(\theta)}{s_k^{(t-1)}(\theta)}$;

  end

  $\Delta(\theta) \leftarrow \prod_{k \in S^{(t)}} \Delta_k(\theta)$;

  $s^{(t)}(\theta) \leftarrow s^{(t-1)}(\theta)\Delta(\theta)$;

end

$s_k^{(t)}(\theta) \leftarrow s_k^{(T)}(\theta)$;

$c_k^{(t)}(\mathbf{w}_k) \leftarrow c_k^{(T)}(\mathbf{w}_k)$;

Table 1: Summaries of Federal Learning Methods for Statistical Heterogeneous

Method	Features	Advantage	Disadvantage
FedAvg	Averaging	Simple model structure, Effective calculation	Poor performance on non-IID data
FedAvg (Share)	Averaging	Partially solve the problem of poor performance under non-IID	Sacrifice part of personal privacy and communication efficiency
MOCHA	Multi-task Learning	Good predictability and convergence for non-IID data	Unable to scale to massive networks and to solve non-convex problem
VIRTUAL	Multi-task Learning	Extend MOCHA method to non-convex problem	Unable to scale to massive networks

Human Activity Recognition[2]: Mobile phone accelerometer and gyroscope data collected from 30 individuals, performing one of six activities: {walking, walking-upstairs, walking-downstairs, sitting, standing, lying-down}(Garcia-Gonzalez et al., 2020). We use the provided 561 -length feature vectors of time and frequency domain variables generated for each instance. We model each individual as a separate client.

Experimental Setting The detaild settings for different approaches are as followed,

- FedAvg: We consider a multilayer perceptrons (MLP) with two hidden dense layers with 100 units and ReLU activation functions in the hidden layers, and softmax activation at the output layer.
- MOCHA: In order to select the best regularization parameter,$\lambda \in \{1e-5, 1e-4, 1e-3, 1e-2, 0.1, 1, 10\}$, for MOCHA model using 5-fold cross-validation.

Experiment Results We explored the accuracy and convergence speed of the FedAvg and MOCHA methods on the dataset Huaman Activity Recognition (HAR). We conducted experiments according to the aforementioned experimental settings. The specific results are shown in the Table 2.

Table 2: Prediction Error for FedAvg and MOCHA on HAR dataset

Method	MSE
FedAvg	0.51 (0.62)
MOCHA	0.46 (0.11)

4 Conclusion

Based on the above papers' theoretical analysis and experimental results, combined with our recurring results, it can be seen that personalized federated learning can indeed improve the effectiveness of the classic federated learning method, especially can effectively deal with statistical heterogeneity. At the same time, federated learning has huge application requirements in various practical scenarios. We will continue to pay attention to the technical development and deployment, and application methods of personalized federal learning.

[2]Dataset is avaliable at https://archive.ics.uci.edu/ml/datasets/human+activity+recognition+using+smartphones

References

Corinzia, L., Beuret, A., and Buhmann, J. M. (2021). Variational Federated Multi-Task Learning. *arXiv:1906.06268 [cs, stat]*. arXiv: 1906.06268.

Garcia-Gonzalez, D., Rivero, D., Fernandez-Blanco, E., and Luaces, M. R. (2020). A Public Domain Dataset for Real-Life Human Activity Recognition Using Smartphone Sensors. *Sensors (Basel, Switzerland)*, 20(8).

gingsmith (2021). gingsmith/fmtl. original-date: 2017-10-20T21:10:51Z.

Jadhav, A. R. (2021). AshwinRJ/Federated-Learning-PyTorch. original-date: 2018-11-16T23:51:14Z.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Eichner, H., Rouayheb, S. E., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., Gruteser, M., Harchaoui, Z., He, C., He, L., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Joshi, G., Khodak, M., Konečný, J., Korolova, A., Koushanfar, F., Koyejo, S., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock, R., Özgür, A., Pagh, R., Raykova, M., Qi, H., Ramage, D., Raskar, R., Song, D., Song, W., Stich, S. U., Sun, Z., Suresh, A. T., Tramèr, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, F. X., Yu, H., and Zhao, S. (2021). Advances and Open Problems in Federated Learning. *arXiv:1912.04977 [cs, stat]*. arXiv: 1912.04977.

Kulkarni, V., Kulkarni, M., and Pant, A. (2020). Survey of Personalization Techniques for Federated Learning. *arXiv:2003.08673 [cs, stat]*. arXiv: 2003.08673.

Li, T., Sahu, A. K., Talwalkar, A., and Smith, V. (2020a). Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Processing Magazine*, 37(3):50–60.

Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z. (2020b). On the Convergence of FedAvg on Non-IID Data. *arXiv:1907.02189 [cs, math, stat]*. arXiv: 1907.02189 version: 3.

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and Arcas, B. A. y. (2017). Communication-Efficient Learning of Deep Networks from Decentralized Data. *arXiv:1602.05629 [cs]*. arXiv: 1602.05629.

Smith, V., Chiang, C.-K., Sanjabi, M., and Talwalkar, A. (2017). Federated multi-task learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 4427–4437, Red Hook, NY, USA. Curran Associates Inc.

Zhang, Y. and Yeung, D.-Y. (2010). A convex formulation for learning task relationships in multi-task learning. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, UAI'10, pages 733–742, Arlington, Virginia, USA. AUAI Press.

Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., and Chandra, V. (2018). Federated Learning with Non-IID Data. *arXiv:1806.00582 [cs, stat]*. arXiv: 1806.00582.