AIMS Mathematics, 8(7): 15058–15080.
DOI:10.3934/math.2023769
Received: 24 February 2023
Revised: 01 April 2023
Accepted: 13 April 2023
Published: 23 April 2023

http://www.aimspress.com/journal/Math

Research article

A novel Bayesian federated learning framework to address multi-dimensional heterogeneity problem

Jianye Yang, Tongjiang Yan* and Pengcheng Ren

College of Science, China University of Petroleum, Qingdao 266580, China

* **Correspondence:** Email: yantoji@163.com.

Abstract: Federated learning (FL) has attracted a lot of interests as a promising machine learning approach to protect user privacy and data security. It requires the clients to send model parameters to the server rather than private datasets, thus protecting privacy to a significant extent. However, there are several types of heterogeneities (data, model, objective and systems) in FL scenario, posing distinct challenges to the canonical FL algorithm (FedAvg). In this work, we propose a novel FL framework that integrates knowledge distillation and Bayesian inference to address this multi-dimensional heterogeneity problem. On the client side, we approximate the local likelihood function using a scaled multi-dimensional Gaussian probability density function (PDF). Moreover, each client is allowed to design customized model according to the requirement through knowledge distillation. On the server side, a multi-Gaussian product mechanism is employed to construct and maximize the global likelihood function, greatly enhancing the accuracy of the aggregated model in the case of data heterogeneity. Finally, we show in extensive empirical experiments on various datasets and settings that global model and local model can achieve better performance and require fewer communication rounds to converge compared with other FL techniques.

Keywords: federated learning; heterogeneity problem; knowledge distillation; Bayesian inference; Gaussian probability density function
Mathematics Subject Classification: 68T09

1. Introduction

Federated learning (FL) is an emerging artificial intelligence technology, which was first proposed by Google in 2016 to tackle the issue of updating local models for Android phone users [1]. It is an important machine learning paradigm where a federation of clients builds a centralized global model collaboratively without transferring their local data. To be more specific, there is a center server coordinating the training process and each client only communicates the model parameters to the center

server while keeping local data private and reducing the risk of data leakage.

Nevertheless, FL can still meet a variety of challenges [2]. Among them, the heterogeneity problem is the most prominent [3, 4]. Our work focuses on four heterogeneities:

(1) Data heterogeneity. The data among clients is frequently distributed in a heterogenous nature in the network, making the independently and identically distributed (IID) assumption invalid. Simply averaging the uploaded parameters from clients may lead to significant accuracy reduction due to large aggregation errors and severe local forgetting [5].

(2) Model heterogeneity. In the original federated framework, all participants agree on the particular architecture of a shared global model [6]. This may be acceptable for a large number of low-capacity devices like mobile phones, but it is not appropriate for business or industry-oriented settings. For example participants may come from different organizations, who have different hardware and storage capabilities [7, 8] and desire to design their own unique models. A more important fact is that the architecture of model involves significant privacy and intellectual property issue, so the participants reluctant to share their personalized models.

(3) Objective heterogeneity. The objectives of FL are distinct, where global model and local model serve different purposes. On the one hand, server intends to train a single generalized global model to fit joint global data distribution for all clients and new participants. On the other hand, each client intends to train a personalized model to fit its own local data distribution for itself. Furthermore, although the clients have similar features (e.g., visual features), they may have different tasks (e.g., 10/100 classification) [9]. It's evident that FedAvg can't meet these requirements as it can only produce a shared global model of a specific architecture.

(4) Systems heterogeneity. Bandwidth and computational powers vary widely among participants leading to different update rates, which has been improved by asynchronous [10] and semi-synchronous [11] schemes. However, these methods converge slowly, especially when the data samples among clients are Non-IID. Though the single round can take shorter time, it requires more rounds to reach a satisfactory accuracy.

Recent works have studied the heterogeneities inherent in real-world FL settings. Zhang et al. [12] introduced a personalized federated contrastive learning model to handle the heterogeneity of features and samples. Concretely, they constructed a double optimization problem concluding maximizing the agreement by comparing the shared parts of local models with the global model and minimizing the agreement by comparing the personalized parts of local models with the global model. Xiao et al. [13] proposed a new federated learning algorithm. Accuracy Based Averaging (ABAVG), which promotes the server-side aggregation process in Non-IID situations by setting weights based on local accuracies. To deal with model heterogeneity, Li et al. [3] introduced FedMD based on knowledge distillation and used the probability distribution outputs of public samples as the interaction information. Hu et al. [14] proposed MHAT which exploited knowledge distillation to extract the update information of heterogenous local models and trained an auxiliary model on the server to realize information aggregation. Aiming at tackling data and systems heterogeneities, Li et al. [15] introduced FedProx which added a proximal term: $\frac{\mu}{2}\left\|\theta-\theta^{t}\right\|_{2}^{2}$ to the local objective function and allowed each participating client to perform a variable amount of work. The term restricts the training parameter to be close to the initial global parameters to some extent, where the scale parameter μ controls the degree of restriction. In addition, Li et al. [16] proposed a similar reparameterization idea inspired by contrastive learning. More recently, Mendieta et al. [17] did not focus on reparameterization tricks but

utilized regularization techniques instead. There are other works that introduced Bayesian inference into federated optimization aiming to improve the aggregation algorithm under the condition of heterogenous data. Maruan et al. [18] analyzed FL problem through inferring a global posterior distribution by having each client infer the posterior of their local data, noting that the round to start local posterior sampling and the shrinkage coefficient ρ should be carefully tuned. Liu et al. [5] proposed a novel FL framework that used online Laplace approximation to approximate posteriors on both the client and server side. They utilized the diagonal form of Fisher information matrix to approximate the local covariance matrix, which effectively reduced the computational complexity. However, the methods above only consider addressing partial types of heterogeneities.

Shen et al. [9] proposed a novel paradigm for FL, named Federated Mutual Learning (FML), in response to DOM (data, objective, and model). FML trained a personalized local model on the client side and a generalized global model on the server side, which solved the objective heterogeneity well. Hence, each client is entitled to determine the size and type of the model, no longer dependent on the server or the consensus of clients. Moreover, the Non-IIDness of data leads to improvement of personalized model. However, the accuracy of the global model decreases to some extent by directly averaging the clients' parameters when the data is distributed in a heterogenous manner. Based on this, we design and implement a new FL framework, named Heterogenous Bayesian Federated Learning (HBFL), aiming to cope with all these four heterogeneities. By formulating FL as a Bayesian inference problem, we find a better solution for the global model, contributing to improvement on the learning process of the client personalized model. What's more, knowledge distillation on the client side can find a more steady (robust) extremum and allow each client to design customized local model. Our work is the first to integrate both knowledge distillation and Bayesian inference into FL, addressing this multi-dimensional heterogeneity problem. The contributions of this paper are summarized as follows:

- We propose a novel FL scheme from the probabilistic perspective in response to these four heterogeneities, which broadens the way of analysing FL, not just optimization techniques.
- We express the objective function of the classification problem in the form of likelihood function and temper the likelihood part of the posterior which significantly improves the performance of the global model and local model. Moreover, we analyze why the tempered likelihood function can be approximated by a scaled multi-dimensional Gaussian PDF.
- Our proposed method is efficient in computation and communication. We show that the communicational and computational complexity of our framework are the same as FedAvg through theoretical analysis.
- Finally, we conduct various experiments to verify the effectiveness of our scheme, including investigating its performance on different hyperparameters and different datasets. We also compare with FML and other methods and present detailed analysis for our experiments.

The reminder of this paper is organized as follows. Section 2 describes some necessary concepts for our framework. Section 3 describes how our approach is conceived and defines a new global objective function, while Section 4 introduces the proposed framework in details. Finally, we report and analyze the experimental results in Section 5 and draw our conclusions in Section 6.

2. Preliminaries

2.1. Federated learning

FL enables participating clients to jointly train a centralized global model by only communicating the model parameters to the server, which is an ingenious technique to protect privacy. Eventually each client can obtain a global model that performs better than the one trained on its own data. The Federated Averaging (FedAvg) [6] algorithm provides a general perspective of the original FL process.

In the original FL scenario, there are N clients collaborating on training a model whose architecture is dependent on the center server or the consensus of clients. At the beginning of the training, the server initializes the model parameters θ and distributes them to all participating clients. Client k ($k = 1, 2, ..., N$) owns a private dataset D_k containing n_k samples $X_k := \{x_i\}_{i=1}^{n_k}$ whose corresponding label set is $Y_k := \{y_i\}_{i=1}^{n_k}$. Then each client uses local data to update the parameters by gradient descent optimization algorithm for several epochs:

$$\begin{aligned} F_k(\theta_k) &= \frac{1}{n_k} \sum_{i=1}^{n_k} f_i(\theta_k), \\ \theta_k &\leftarrow \theta_k - \alpha \nabla F_k(\theta_k), \end{aligned} \tag{2.1}$$

where $f_i(\theta_k)$ is the sample loss function, $F_k(\theta_k)$ is the local objective function on the dataset D_k with respect to θ_k, α is the learning rate, and $\nabla F_k(\theta_k)$ is the gradient of $F_k(\theta_k)$. After a period of local updates, the parameters are communicated to the server for aggregating:

$$\theta_{global} = \sum_{k=1}^{N} \frac{n_k}{n} \theta_k, \tag{2.2}$$

where θ_{global} is global model, and $n := \sum_{k=1}^{N} n_k$. The process above circulates until the global model converges or the pre-configured communication rounds expire. However, the trained global model is identical and known to each other among the clients. It's urgent to come up with a new idea to allow each client to design personalized model in the practical FL setting. Knowledge distillation emerges as the time requires.

2.2. Knowledge distillation

Knowledge distillation was first proposed in [19], aiming to transfer the knowledge from a complex model or multiple models (as teacher model) to another lightweight model (as student model) without significant loss of performance. For a data sample, its hard label is the ground truth label and the soft label is denoted as q whose i-th component q_i is

$$\frac{\exp\left(\frac{z_i}{T}\right)}{\sum_j \exp\left(\frac{z_j}{T}\right)}, \tag{2.3}$$

where z_i represents the value corresponding to the i-th category output of the last layer by the teacher model or the student model, and T is the distillation temperature which is the hyperparameter greater

than 0. It is obvious that the soft label is smoother predictive probability distribution than the hard label.

Unlike general data training, the student model not only fits the hard label, but also learns the knowledge conveyed by the soft label from the teacher model. In this way, it can learn which category the sample belongs to and which categories are more similar to the belonging category. In addition to fitting the hard label, the another goal is to make the student soft label $q_{student}$ close to the teacher soft label $q_{teacher}$, that is, to minimize the Kullback-Leibler (KL) divergence between $q_{teacher}$ and $q_{student}$ (KL divergence describes the distance between probability distributions). Hence, our objective function only needs to add one term to the original one:

$$L = \alpha L_{CE} + (1-\alpha) D_{KL}\left(q_{teacher} \| q_{student}\right), \tag{2.4}$$

where α controls the proportion of knowledge derived from the data and the teacher model. L_{CE} is denoted as the cross entropy of the client's datasets and $L_{CE}(p,q) = -\sum_{i=1}^{m} p_i \log(q_i)$, where m is dimension of vector p and vector q. Besides, p is the hard label (one-hot encoding) of the sample and q is the soft label obtained by the local model. D_{KL} is denoted as the KL divergence of the client's datasets and $D_{KL}(p\|q) = \sum_{i=1}^{m} p_i \log(\frac{p_i}{q_i})$, where p is the distribution that is approximated by the q. Besides, p is the soft label obtained by the teacher model and q is the soft label obtained by the student model. In fact, the cross entropy of the client's datasets can be deemed as a special case of the KL divergence, for $\sum_{i=1}^{m} p_i \log p_i$ is a constant. A KL divergence term is added to the original cross entropy term, which is like adding a regularization term to avoid overfitting. The goal of knowledge distillation is to make the student model learn the generalization ability of the teacher's model. In theory, the result will be better than that simply by fitting the training data.

As a one-way transfer, knowledge distillation works only if there is a well-trained teacher model. Existing related papers improve the knowledge distillation in the FL scenario. A new strategy is presented in [20], named deep mutual learning. There is an ensemble of students learning collaboratively and teaching each other throughout the training process. If the deep mutual learning is introduced into FL when each participating client is performing the local update, the global model and personalized model will learn from each other. A novel federated learning paradigm is presented in [9], named Federated Mutual Learning (FML). There are two models that need to be trained on the client side, including the meme model and the local model, where the meme model inherits from the previous global model and the local model remains on the client continuously. The objective functions of two models at the stage of local update are given by

$$\begin{aligned} L_{local} &= \alpha L_{CE}^{local} + (1-\alpha) D_{KL}\left(q_{meme} \| q_{local}\right), \\ L_{meme} &= \beta L_{CE}^{meme} + (1-\beta) D_{KL}\left(q_{local} \| q_{meme}\right). \end{aligned} \tag{2.5}$$

As we can see, the direction of knowledge transfer between meme model and local model is two-way. The most important thing is that they can have different architectures, so each client is allowed to design the customized model. However, similarly to FedAvg, directly averaging the local parameters can create problems like aggregation error and local forgetting [5] when the data among clients is heterogenous. So it's urgent to propose a new aggregation method to address these issues.

3. A Bayesian inference perspective on federated learning

A nature question arises that how to aggregate parameters on the server side to ensure the accuracy of the global model in the case of data heterogeneity. To answer the question, this section first introduces a new idea to analyze FL from the perspective of Bayesian inference and derives a new objective function applicable to our framework. Finally, we propose a novel aggregation method.

3.1. Rethinking objective function

From the perspective of optimization, the goal of FL is typically formulated as follows:

$$\begin{aligned} \min_{\theta} F(\theta) &:= \sum_{k=1}^{N} \frac{n_k}{n} F_k(\theta), \\ F_k(\theta) &:= \frac{1}{n_k} \sum_{i=1}^{n_k} f_i(\theta), \end{aligned} \tag{3.1}$$

where $\theta \in \mathbb{R}^d$. First, we consider the linear regression problem, where the sample objective function $f_i(\theta) = \frac{1}{2} \left\| \theta^T x_i - y_i \right\|_2^2$. Hence, the global objective function $F(\theta) = \frac{1}{2n} \sum_{i=1}^{n} \left\| \theta^T x_i - y_i \right\|_2^2$. In this way, FL is formulated as an optimization problem. Now we treat FL from the view of statistics. Supposing $y_i = \theta^T x_i + \epsilon_i$, where ϵ_i subjects to the Gaussian distribution $N(0, \sigma)$. We can easily prove that the global objective is to minimize $-\frac{1}{n} \ln p(Y|X, \theta)$, where $Y := \{y_i\}_{i=1}^n$, and $X := \{x_i\}_{i=1}^n$. Since

$$\begin{aligned} p(D|\theta) &= p(X|\theta) p(Y|X, \theta) \\ &= p(X) p(Y|X, \theta), \end{aligned} \tag{3.2}$$

where $D := \{(x_i, y_i)\}_{i=1}^n = (X, Y)$, the $p(X)$ is not related to θ and $p(D|\theta)$ is a likelihood function for the local data D parameterized by θ. So the global objective can be rewritten as $\min_{\theta} -\frac{1}{n} \ln p(D|\theta)$. We can further express it as $\min_{\theta} -\frac{1}{n} p(D|\theta)$, where the likelihood part can be denoted as a scaled multi-dimensional Gaussian PDF with respect to θ. We prove the conclusion in Supplementary A.

Similarly to the linear regression problem, the classification problem can also express the global objective as $\min_{\theta} -\ln p(D|\theta)$. Hopefully, the likelihood function $p(D|\theta)$ can be also approximated by a scaled multi-dimensional Gaussian PDF. In fact, this is predictive. According to Bayes rule,

$$p(D|\theta) = \frac{p(\theta|D) \int p(D|\theta) p(\theta) \, d\theta}{p(\theta)}, \tag{3.3}$$

the posterior probability is commonly assumed to follow multi-dimensional Gaussian distribution in variational Bayes for Bayesian neural networks [21, 22]. Additionally, in [23], the authors put forward to a systematic way to improve the Bayes posterior in many models by tempering its likelihood part. That is

$$p(\theta|D) = \frac{p(D|\theta)^{\frac{1}{T}} p(\theta)}{\int p(D|\theta)^{\frac{1}{T}} p(\theta) \, d\theta}, \tag{3.4}$$

where T is the temperature which is much less than 1. Hence,

$$p(D|\theta)^{\frac{1}{T}} = \frac{p(\theta|D)\int p(D|\theta)^{\frac{1}{T}} p(\theta)\, d\theta}{p(\theta)}. \tag{3.5}$$

Then, we rewrite the global objective as

$$\max_{\theta} p(D|\theta)^{\frac{1}{T}} = \max_{\theta} \frac{p(\theta|D)}{p(\theta)} \int p(D|\theta)^{\frac{1}{T}} p(\theta)\, d\theta. \tag{3.6}$$

Both the posterior and the prior probability can be supposed to subject to multi-dimensional Gaussian distribution whose covariance matrix is diagonal matrix:

$$\begin{aligned} p(\theta|D) = p_1(\theta) \approx N(\theta|\mu_1, \Sigma_1) = \exp\left[\zeta_1 + \eta_1^T\theta - \frac{1}{2}\theta^T\Lambda_1\theta\right], \\ p(\theta) = p_2(\theta) \approx N(\theta|\mu_2, \Sigma_2) = \exp\left[\zeta_2 + \eta_2^T\theta - \frac{1}{2}\theta^T\Lambda_2\theta\right], \end{aligned} \tag{3.7}$$

where $\Lambda_i = \Sigma_i^{-1}$, $\eta_i = \Lambda_i\mu_i$ and $\zeta_i = -\frac{1}{2}(d\log 2\pi - \log|\Lambda_i| + \eta_i^T\Lambda_i^{-1}\eta_i)$, $i = 1, 2$. So,

$$\frac{p_1(\theta)}{p_2(\theta)} \approx \rho \exp\left[\zeta + \eta^T\theta - \frac{1}{2}\theta^T\Lambda\theta\right], \tag{3.8}$$

where $\Lambda = \Lambda_1 - \Lambda_2$, $\eta = \eta_1 - \eta_2$, $\zeta = -\frac{1}{2}\left(d\log 2\pi - \log|\Lambda| + \eta^T\Lambda^{-1}\eta\right)$ and $\rho = \exp\left[\zeta_1 - \zeta_2 - \zeta\right]$, noting that ρ is not related to θ. Due to the fact that the variance of each parameter following the posterior probability is generally less than that following the prior probability, the diagonal elements of Λ_1 are all greater than those of Λ_2. Hence, Λ is a diagonal matrix whose diagonal elements are all positive, which is same to Σ. In conclusion, the posterior probability divided by the prior probability can obtain a scaled Gaussian PDF. Moreover, $\int p(D|\theta)^{\frac{1}{T}} \times p(\theta)\, d\theta$ is not related to θ. Therefore, it is reasonable to approximate the global objective $p(D|\theta)^{\frac{1}{T}}$ by a scaled multi-dimensional Gaussian PDF:

$$p(D|\theta)^{\frac{1}{T}} \approx \gamma N(\theta|\mu, \Sigma), \tag{3.9}$$

where γ is a constant. Finally, after the above theoretical analysis, we can approximate the global objective as $\max_{\theta} N(\theta|\mu, \Sigma)$. Hence, the extremum of global objective can be approached by the mode of the Gaussian distribution (i.e., μ).

3.2. Bayesian aggregation method

The global objective function $p(D|\theta)^{\frac{1}{T}}$ can be obviously decomposed into a product of N clients' objective functions:

$$p\left(D = \bigcup_{k=1}^{N} D_k|\theta\right)^{\frac{1}{T}} = \prod_{k=1}^{N} p(D_k|\theta)^{\frac{1}{T}}. \tag{3.10}$$

In the same way, each $p(D_k|\theta)^{\frac{1}{T}}$ can be also approximated by a scaled multi-dimensional Gaussian PDF. So,

$$p(D|\theta)^{\frac{1}{T}} \approx \prod_{k=1}^{N} \gamma_k N(\theta|\mu_k, \Sigma_k), \tag{3.11}$$

where γ_k is a constant.

It is easy to prove that a product of N Gaussian PDFs leads to the scaled Gaussian PDF, which is consistent with (3.9). At the same time, the covariance matrix is $\left(\sum_{k=1}^{N} \Sigma_k^{-1}\right)^{-1}$. It is worth noting that the variance of each parameter following the global Gaussian distribution decreases with the number of clients increasing. We expect it to be of the same order of magnitude as the local one. Naturally, we assume the approximation of the global objective is rewritten as

$$\max_{\theta} \prod_{k=1}^{N} p(D_k|\theta)^{\frac{q_k}{T}}, \tag{3.12}$$

where q_k is set as the proportion of the amount of data owned by the client k. This is identical to the view proposed in [5] that a product of local posteriors is used to approximate global posterior. Consequently, the global Σ is $\left(\sum_{k=1}^{N} q_k \Sigma_k^{-1}\right)^{-1}$ and the global μ is $\left(\sum_{k=1}^{N} q_k \Sigma_k^{-1}\right)^{-1} \left(\sum_{k=1}^{N} q_k \Sigma_k^{-1} \mu_k\right)$.

So the extremum of global objective function can be easily obtained by the local means and local covariances uploaded by the clients. Importantly, this also means that FedAvg can be viewed as our approach that each local covariance matrix is the same and, as a result, obtains the global extremum $\sum_{k=1}^{N} q_k \mu_k$, where μ_k is both the mean vector of approximated multi-dimensional Gaussian distribution and the extremum of the local objective function.

We start from the global objective function and then introduce Bayes rule to analyze it. It's a novel approach to use a scaled multi-dimensional Gaussian distribution to fit the tempered likelihood function. We will prove the feasibility of our aggregation method in theory below and in Section 5 prove it in practice with experiments.

3.3. Feasibility in theory

We take the one-dimensional linear regression problem in a two clients' federation as an example. It has been proved that the local objective function of each client can be expressed as a scaled one-dimensional Gaussian PDF in this case. Hence, the global objective function can be rewritten as a product of two Gaussian PDFs, which is still a scaled Gaussian PDF. For simplicity, we exploit normalized Gaussian distributions to represent the local and global objective functions respectively in Figure 1. From the perspective of aggregation on the server, it is reasonable to take a weighted element-wise average on the parameters of all local models if the data among clients follows the same distribution. In the classical FL framework, the local objective function

$$E_{(x,y)\sim\mathcal{P}_k(x,y)}\left[f((x,y);\theta)\right] = \int_{(x,y)\sim\mathcal{P}_k(x,y)} f((x,y);\theta)\mathcal{P}_k(x,y)d(x,y)$$

is approximated by $\frac{1}{n_k}\sum_{i=1}^{n_k} f_i((x_i,y_i);\theta)$. With the same distribution, each client has the same objective function which is identical to the global objective. Hence, they have the same extremum. However, the weighted average of the solution of each local objective obviously deviates from the solution of the global objective when the data among clients is Non-IID, due to the distinct local objective functions and local extremums. As shown in Figure 1, the modes of two Gaussian distributions are different

in the case of heterogeneous data. In FedAvg, each client aims to find the mode of its own Gaussian distribution and Figure 1 shows that there is a high probability that the weighted average of two local modes will deviate from the global mode. Unlike the previous method, our Bayesian aggregation treats a product of the Gaussian distribution as the global objective function. The extremum of the local objective function does not require to be the same, instead it is necessary to calculate the mean vector and the covariance matrix approximated by the respective likelihood function to get the global mode, which can be viewed as the extremum of the global objective function. Therefore, we don't have to worry about data being heterogenous. Additionally, from the perspective of training on the client, our approach also has the significant advantage over FedAvg. In details, after receiving the global aggregated model, each client takes it as the initialization for further local training. In FedAvg, a larger number of local epochs may lead each device towards the extremum of its local objective as opposed to the global objective in the case of heterogenous data, which potentially hurts convergence or even causes the method to diverge [5]. In our approach, the deviation from the global extremum on the client side is no longer a defect but an advantage that the client can calculate the personal mean and the personal covariance more accurately.

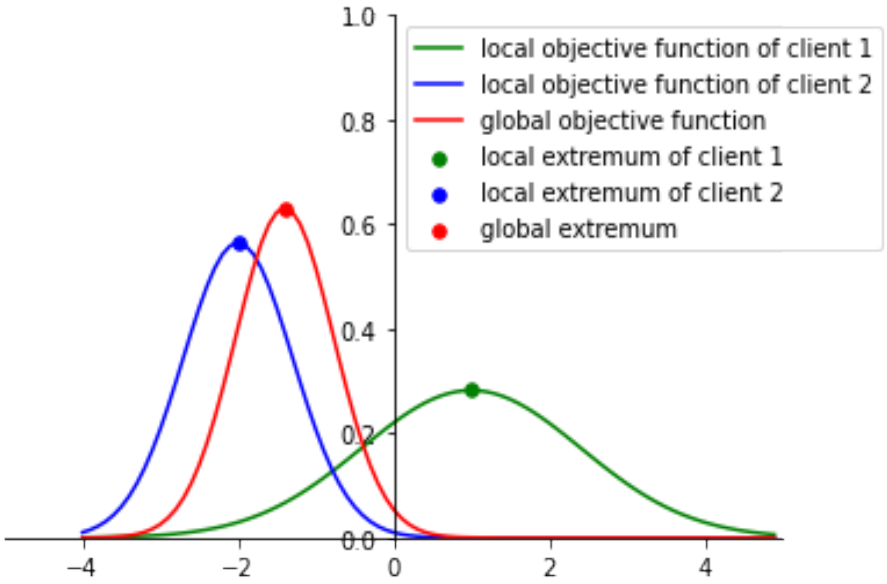

Figure 1. The objective function curve.

4. Methods

4.1. The framework for heterogenous federated learning

Inspired by Bayesian Neural Network, we adopt KL divergence to measure difference between two probability distributions on the client side. Due to the fact that each $p(D_k|\theta)^{\frac{1}{T}}$ can be approximated by a scaled multi-dimensional Gaussian PDF, our goal is to minimize the KL divergence between two normalized distributions, which is defined as

$$\min_{\mu_k,\Sigma_k} D_{KL}^{meme} := D_{KL}\left(N(\theta|\mu_k,\Sigma_k)||\frac{1}{\gamma_k}p(D_k|\theta)^{\frac{1}{T}}\right), \tag{4.1}$$

where $p(D_k|\theta) = p(X_k)p(Y_k|X_k,\theta)$ as in (3.2), γ_k is the normalization factor and $p(X_k)$, γ_k are independent of θ.

We adopt gradient descent optimization algorithm to minimize the goal on the client side. In each round, after several epochs of training, the client gets final iteration results μ_k^t and σ_k^t which are the training parameters of the meme model. Additionally, the personalized local model θ_{local}^t is updated based on knowledge distillation. In order to allow each client to design its own customized model, we introduce knowledge distillation into our framework. Specifically, each local model is trained with two losses: a conventional cross entropy loss and a KL divergence loss that aligns each student's class posterior with the class probabilities of the teacher. On account of the fact that the local μ_k^t can be considered as the extremum of the local objective, the meme model, as a teacher model, provides q_{meme} from the local μ_k^t to the client's local model which plays the role of the student model. Through the meme model, the local model obtains the knowledge from the aggregated global model. The loss function of local model

$$L_{local} = \alpha L_{CE}^{local} + (1-\alpha) D_{KL}\left(q_{meme} \| q_{local}\right). \tag{4.2}$$

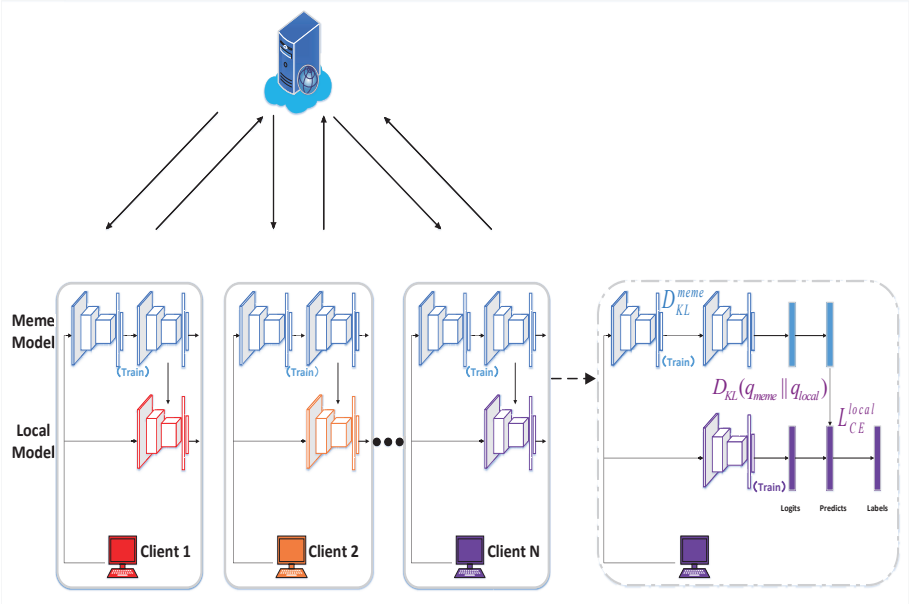

Figure 2. The whole process of our framework for N clients.

Previous aggregation algorithms mainly relied on weighted average of the information uploaded by the clients to obtain global information. We consider this choice is not optimal, especially when the data among client are generated in a non-identically distributed fashion. Instead, we propose to calculate the mean of a product of multi-dimensional Gaussian distribution to aggregate information. Our scheme aims to make the aggregation result more accurate and the model convergence faster while allowing clients to independently design their model architectures. Consequently, we design two models on the client side. The meme model includes μ_k^t and σ_k^t, where μ_k^t inherits from the previous global model, whose architecture is determined by the server. While, the local model θ_{local}^t can be designed flexibly by its client. The process of Bayesian federated learning framework is shown in Figure 2, where each client uses the local datasets to train the meme model and local model respectively. Concretely, the meme model is trained by minimizing D_{KL}^{meme}. While, the local

model exploits the soft labels the meme model provides for knowledge distillation, so it is trained by minimizing weighting of $D_{KL}(q_{meme}||q_{local})$ and L_{CE}^{local}. It is worth noting that each participant is able to design the customized local model, so all the models among participants can be different. At the same time, the client doesn't need to share the local model, which is a good approach to protect its privacy.

The overview of our proposed algorithm is shown in Algorithm 1. The entire interaction process includes four steps. First, the server initializes the global μ^t and distributes it to all participating clients. Second, each client forks the global μ^t as the μ_k^t of its meme model which is a Bayesian neural network and treats the global μ^t as the initial value of the local μ_k^t and initializes the local Σ_k^t. Then, the client updates the parameters of meme model by gradient descent optimization algorithm for several epochs to minimize the KL divergence. Meanwhile, the client also trains the parameter θ_{local}^t of the personalized local model through knowledge distillation. Third, each participant uploads the interactive information to the server. Finally, the global μ^{t+1} can be calculated by the aggregation of each client's information on the server side. We return to the first step to repeat the whole process until the global model converges or pre-configured communication rounds expire.

Algorithm 1 Heterogenous Bayesian Federated Learning (HBFL)

θ: the model parameters
μ: the mean vector of multi-dimensional Gaussian distribution
Σ: the covariance matrix of multi-dimensional Gaussian distribution
q_k: the proportion of the amount of data owned by the client k

Server executes:

Initialize the global model $\theta_{global}^t\,(\mu_{global}^t)$
for each round $t = 1, 2...$ **do**
  for each client k from 1 to N in parallel **do**
    $\mu_k^{t+1}, \Sigma_k^{t+1} = \text{ClientUpdate}(\mu_k^t, \Sigma_k^t)$
  $\theta_{global}^{t+1} = \mu_{global}^{t+1} = \left(\sum_{k=1}^{N} q_k\left(\Sigma_k^{t+1}\right)^{-1}\right)^{-1}\left(\sum_{k=1}^{N} q_k\left(\Sigma_k^{t+1}\right)^{-1}\mu_k^{t+1}\right)$

ClientUpdate:

Initialize the local model θ_{local}^t and Σ_k^t
Fork the $\theta_{global}^t\,(\mu_{global}^t)$ as its μ_k^t
for each each local epoch e from 1 to E **do**
  B = split D_k into mini−batches
  for b in B **do**
    Update μ_k^t, Σ_k^t by minimizing D_{KL}^{meme} (4.1)
    Update the predictions q_{meme} by (2.3) for the current mini-batch
    Update θ_{local}^t by minimizing L_{local} (4.2)
Return μ_k^{t+1}, Σ_k^{t+1} to the server

From the perspective of the server, the mean of a product of multi-dimensional Gaussian distribution can be calculated by aggregating information uploaded by the participants, where the global μ can be also regarded as the global objective extremum. Hence, the global μ is a generalized model that fits the joint distribution $\mathcal{P}_{joint}(x, y)$ over all data among the participants. From the perspective of the client, in

each communication round the personalized local model keeps training over the client's private data and distilling knowledge from the meme model. On the one hand, the local model fits the personalized distribution $\mathcal{P}_k(x, y)$ over private data. On the other hand, it also learns how to generalize to new samples. Additionally, we allow for partial work to be performed across client to deal with systems heterogeneity in Section 5.3.3. In conclusion, our framework can handle all these four heterogeneities discussed in Section 1.

4.2. Algorithm complexity and privacy security analysis

4.2.1. Algorithm complexity

In our algorithm, the client needs to upload μ_k^t and Σ_k^t in each communication round. Since Σ_k^t is a diagonal matrix, the communication complexity is $O(d)$, which is the same communicational complexity as FedAvg.

Although our means requires more computation steps, it also has the same computational complexity as FedAvg. On the client side, the back-propagation algorithm is used to train the neural network in standard FedAvg, where we can easily conclude that the computational cost is $O(d)$. During local training, our algorithm needs to train a meme model and a local model, where the former complexity is $O(2 \times d)$ and the latter complexity is $O(d)$. Therefore, the overall computational complexity of our method on the client side is $O(3 \times d)$. On the server side, FedAvg takes a weighted element-wise average on N neural networks to aggregate local models, which has a complexity of $O(N \times d)$. While, in our algorithm, the server first needs to take the inverse of each Σ_k^t. Due to the assumption that Σ_k^t is a diagonal matrix, the above operation requires a complexity of $O(N \times d)$. Then, it calculates each product of the inverse of Σ_k^t and the corresponding μ_k^t, which has a complexity of $O(N \times d)$. A weighted element-wise average on the products requires $O(N \times d)$ complexity. In the same way, a weighted element-wise average on the inverses of Σ_k^t requires $O(N \times d)$ complexity. In the end, the multiplication between the inverse of weighted matrix and the weighted vector requires a complexity of $O(2 \times d)$, for the inverse of weighted matrix is also diagonal. These operations require a complexity of $O(4N \times d + 5 \times d)$ in all which is equivalent to $O(d)$ (i.e., same as FedAvg), for N is much smaller than d.

In summary, the communicational complexity and the computational complexity of our framework are the same as FedAvg.

4.2.2. Privacy security

Our proposed framework, like many existing FL methods, requires to communicate the models between the server and each client, resulting in the risk of privacy disclosure. There are a number of existing protection mechanisms that can be added to our framework to protect clients from malicious attack. These methods include secure multi-party computation [24], homomorphic encryption [25] and differential privacy [26]. Among them, differential privacy is the most widely used in FL, which is a standard approach to prevent clients' private information from inference attacks. We leave further explorations of this aspect in the future work.

5. Experiments

In this section we evaluate the performance of our algorithm over three commonly used image classification datasets in various settings implemented by PyTorch.

5.1. Experimental setup

We test our framework in a simulated cross-silo federated learning environment. The clients in the cross-silo setting are a small number of organizations (e.g., financial and medical) with reliable communications and abundant computing resources in datacenters [27, 28]. Compared to being in the cross-device FL setting, the clients are more expected to design their own customized local models and more capable of training a Bayesian neural network and a conventional neural network. Therefore, we focus on cross-silo FL in this paper. There are 10 clients ($N = 10$) under the orchestration of a central server. Moreover, the total communication rounds $T = 100$ and the number of local epochs $E = 5$.

5.1.1. Datasets and models

There are three datasets used in our experiments that are widely employed in FL. MNIST [29] is a 10-classification handwriting digit dataset, with the number of 0 to 9. It consists of 28×28 gray images which are split into 60000 training and 10000 test images. CIFAR10/100 [30] are 10/100 classification datasets respectively, which are 32×32 3-channel color images. CIFAR10 consists of 50000 training and 10000 test pictures in 10 classes. While, CIFAR100 has the same number of pictures as CIFAR10, but it can be divided into 100 classes. The models we apply are similar to [9]. We apply four types of models in our experiments: multi-layer perceptron (MLP) [6], LeNet5 [31], a convolutional neural networks (CNN1) with two 5×5 convolution layers (the first with 6 channels, the second with 16 channels, each followed with 2×2 max pooling and ReLu activation) and two FC layers, and a convolutional neural networks (CNN2) with three 3×3 convolution layers (the first with 32 channels followed with 2×2 max pooling and ReLu activation, the second with 64 channels followed with 2×2 max pooling and Relu activation, the third with 64 channels followed with Relu activation) and two FC layers. The optimizer of the conventional neural network we choose is the SGD algorithm, with learning rate = 0.01, momentum = 0.9, weight_decay = 5×10^{-4} and batchsize = 128. While the optimizer of the Bayesian neural network we choose is the ADAM algorithm, with learning rate = 0.001, weight_decay = 5×10^{-4} and batchsize = 128. We don't decay the learning rate through all rounds. Additionally, we conduct experiments with extra hyperparameter: the low temperature T. As in [32], we use T equal to 10^{-4} for our experiments. In our proposed framework, the global model and the local model are conventional neural networks, while the meme model that inherits the parameters of previous global model as initial μ_k^t is Bayesian neural network. Moreover, the trained μ_k^t of the meme model and the trained θ_{local}^t of the local model are used to infer test sets and we record the best accuracy between them, which is same to FML, except that the meme model in FML is a conventional neural network.

5.1.2. Heterogenous distribution of data

Since we assume that 10 clients are in the network, we divide the dataset into 10 parts for 10 clients in IID and Non-IID settings. Additionally, we use Dirichlet distribution as in [33, 34] to create disjoint

Non-IID client training and test data. The value of α controls the degree of Non-IIDness: the smaller α is, the more likely the clients are only possession of examples from one class and we use $\alpha = 100$ to simulate IID setting. It is worth noting that we use the same sampling of Dirichlet distribution for the client's training and test dataset. We conduct experiments with $\alpha = 1, 0.1, 0.05$ to analyze our algorithm performance in the case of heterogenous data.

5.1.3. Baselines

We compare our algorithm with other ones, including ABAVG [13], FedAvg [6], FedProx [15], and FML [9]. Among them, ABAVG, FedAvg and FedProx require that each client's local model is identical. While, FML allows each client to design the customized model. Moreover, FedProx adds a proximal term to mitigate the oscillation of the global model accuracy curve but fails to reach a high accuracy compared to FedAvg. The following experiments are conducted on the basis that all clients are sampled in each round.

5.2. Evaluation on the common federated learning settings

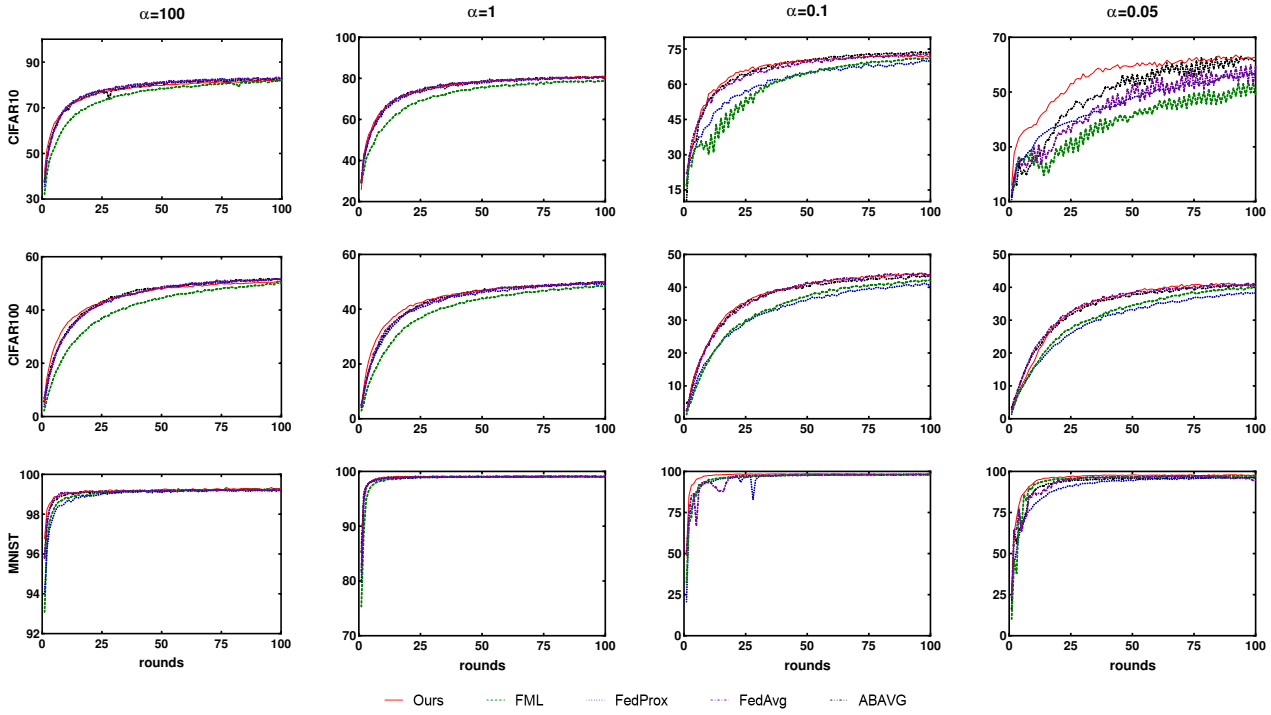

Figure 3. Global accuracy curves with different levels of data heterogeneity on four datasets.

We first assume that each client's model has the same architecture, which is the most general case that all clients just intent to train a shared global model. We apply FedAvg, FedProx and FML as three baselines and compare the performance with our algorithms in both IID and Non-IID manners. We put up four types of models (MLP, LeNet5, CNN1, CNN2) over three datasets (MNIST, CIFAR10, CIFAR100) in four data settings (α = 100, 1, 0.1, 0.05). Since all clients in this case aim to acquire a shared global model, we only measure the accuracy of global model. We show the global accuracy

curves in Figure 3 with different FL frameworks, including our proposed method, ABAVG, FedAvg, FedProx and FML.

We can see from the Figure 3 that with α increasing (i.e., the heterogeneity degree of data enlarging), the accuracy of global model becomes decreasing. However, our algorithm still performs well. Specifically, in most of settings, the global accuracy of our method is higher in the initial phase and converges faster compared with baselines. What's more, the improvement of our algorithm over FML is visually apparently. In the Non-IID setting, the method we proposed gains a more stable curve compared with ABAVG, FedAvg and a more accurate curve compared with FedProx. Given all of that, our algorithm possesses the benefits of other three baselines. It is worth noting that our method performs better on CIFAR10 and MNIST compared with CIFAR100. The reason is that it is more effective to fit the likelihood function with Gaussian PDF on less classified data (small-scale data).

5.3. Federated learning on heterogenous environments (model, objective, systems heterogeneity)

In the following experiments, we set the Non-IID degree α to be 100 and 0.05 respectively to represent IID and Non-IID settings. The data heterogeneity lies in each experiment, for we set IID and Non-IID environments. The model, objective and systems heterogeneities in FL are common in a realistic scenario. The client wishes to design customized model according to the requirement, which leads to different architectures of models among all clients. Besides, the server tends to train a generalized model that fits all data distribution, while the client tends to train a personalized model that only fits its own data distribution. At the same time, if the client desires to obtain a generalized model, it can also receive from the server. Moreover, there also exists significant variability in terms of the systems characteristics on each device.

5.3.1. Model heterogeneity

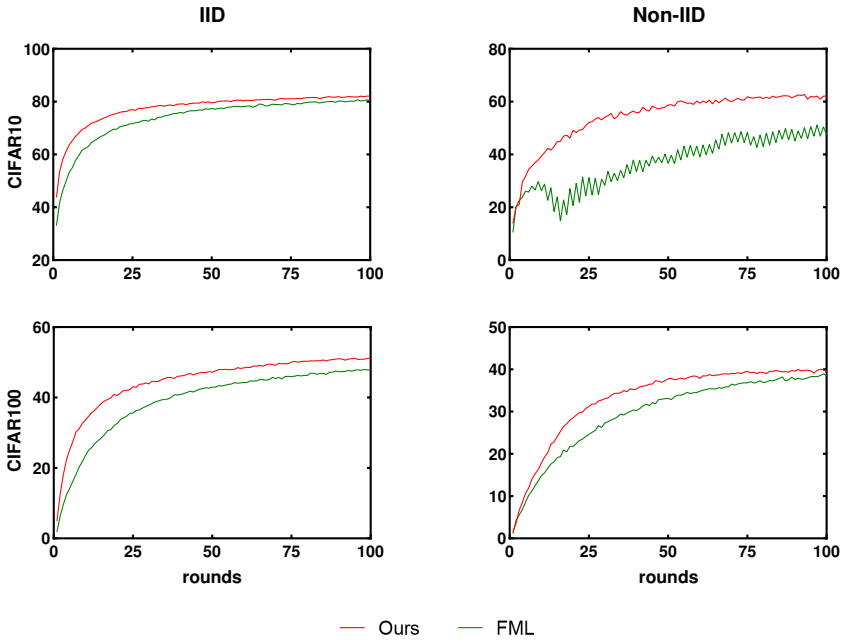

Figure 4. Global accuracy curves in IID and Non-IID settings on CIFAR10/100.

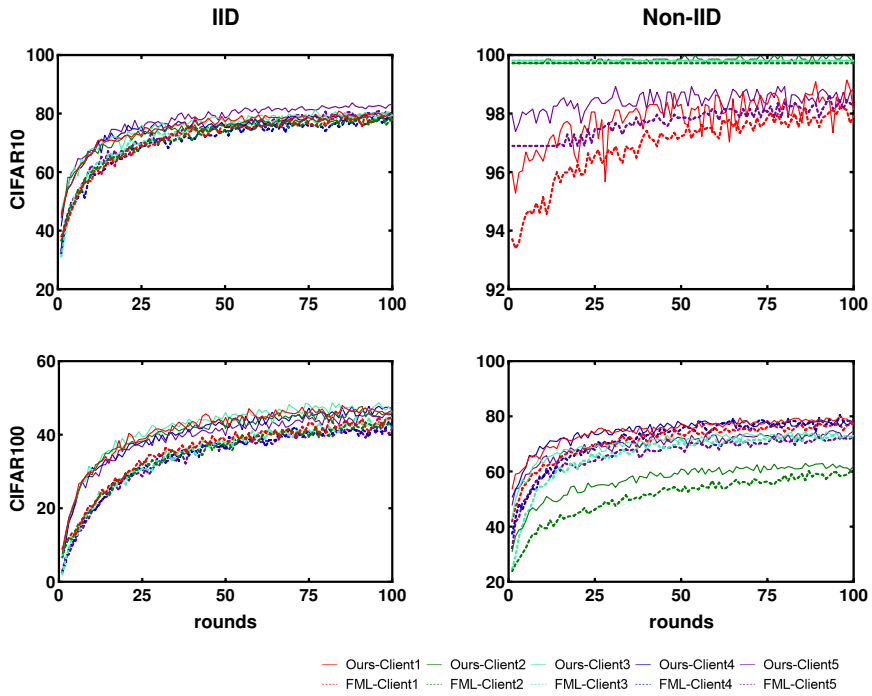

Figure 5. No.1 to No.5 client accuracy curves in IID and Non-IID settings on CIFAR10/100.

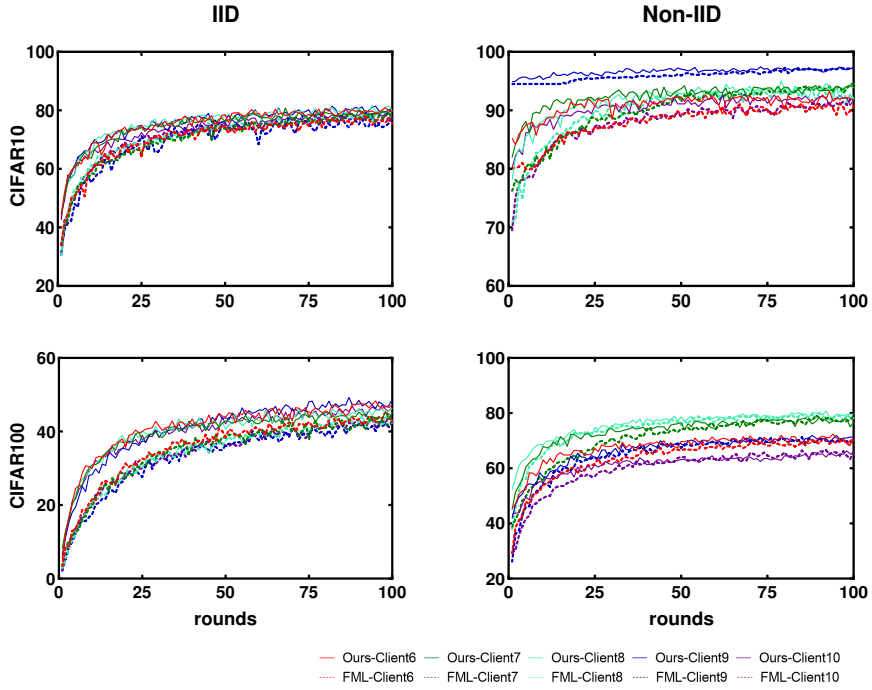

Figure 6. No.6 to No.10 client accuracy curves in IID and Non-IID settings on CIFAR10/100.

Table 1. Top-1 test accuracy of global model in IID and Non-IID settings on CIFAR10/100.

Datasets	Settings	FML	Ours
CIFAR10	IID	80.71	82.10
	Non-IID	51.26	62.69
CIFAR100	IID	47.96	51.19
	Non-IID	38.95	40.01

Table 2. Top-1 test accuracy of No.1 to No.5 client models in IID and Non-IID settings on CIFAR10/100.

Datasets	Settings	Client1		Client2		Client3		Client4		Client5	
		FML	Ours	FML	Ours	FML	Ours	FML	Ours	FML	Ours
CIFAR10	IID	80.16	80.88	78.91	79.30	80.40	81.21	78.98	80.30	80.83	83.67
	Non-IID	98.19	99.16	99.72	100	99.81	99.81	100	100	98.64	98.93
CIFAR100	IID	45.03	47.96	43.33	47.74	45.51	48.74	43.61	47.77	43.34	45.73
	Non-IID	77.47	80.10	60.20	62.97	74.13	74.51	80.41	79.38	72.98	74.93

Table 3. Top-1 test accuracy of No.6 to No.10 client models in IID and Non-IID settings on CIFAR10/100.

Datasets	Settings	Client6		Client7		Client8		Client9		Client10	
		FML	Ours	FML	Ours	FML	Ours	FML	Ours	FML	Ours
CIFAR10	IID	77.78	80.90	79.39	80.81	79.52	81.82	78.74	81.44	78.55	79.36
	Non-IID	91.19	93.27	94.57	94.28	94.14	95.05	97.25	97.45	91.89	92.75
CIFAR100	IID	45.62	48.31	43.61	45.39	44.42	46.33	42.46	49.25	44.53	47.24
	Non-IID	70.87	72.45	79.27	78.27	80.13	80.75	71.43	71.88	66.55	66.31

Aiming at handling the heterogeneity problem, knowledge distillation is introduced to our framework. We set CNN2 as the global model and 2×MLP, 2×LeNet5, 3×CNN1 and 3×CNN2 as the local models. Our experiments are conducted on CIFAR10 and CIFAR100 with four neural architectures in the IID and Non-IID settings. We test the performance of the product of FL (the personalized local model and generalized global model) on validate set, presented in Figures 4–6 and exhibit the best Top-1 test accuracy in Tables 1–3.

We can see that compared with FML, the global and local curves of our method perform better in term of converge speed, stability and accuracy when encounting the model heterogeneity issue and the best Top-1 test accuracy of our method is higher in most cases.

5.3.2. Objective heterogeneity

Then, as in [9], we study the multi-task FL, where clients have different tasks to handle. In our experiment, there are two clients running CIFAR10 and CIFAR100 tasks with LeNet5 and CNN1 respectively which have the same convolution layers. Besides, we use the joint convolution layers as global model. When clients receive the global model, the FC layers are spliced on it as required, dealing with 10/100 classification task. The local curves and the best results are shown in Figure 7 and

Table 4.

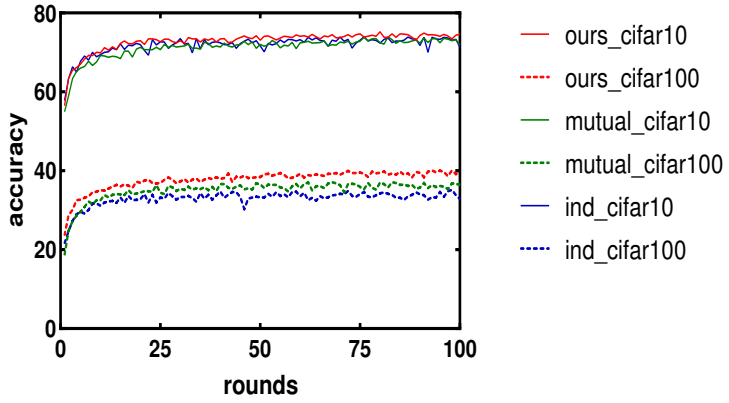

Figure 7. Client accuracy curves by independent training, FML and our method respectively.

Table 4. Top-1 test accuracy of client models by independent training, FML and our method respectively.

Method	Client1(CIFAR10)	Client2(CIFAR100)
Independent	73.99	35.16
FML	73.94	37.12
Ours	75.20	40.13

We can observe that our algorithm benefits all clients who have different tasks as opposed to running the task independently and clearly outperforms FML.

5.3.3. Systems heterogeneity

Finally, we simulate FL settings with systems heterogeneity. As [15] described, we allow for partial solutions to be sent to deal with systems heterogeneity. In particular, we fix the number of local epochs E to be 5, and force some clients to perform fewer updates than 5 epochs given their current systems constraints. Furthermore, we assign x epochs (chosen uniformly at random between [1, 5]) to 50%, and 90% of the selected devices, respectively for varying heterogenous settings in each round. To make valid comparisons, all methods have the same experimental setup in each round. Figures 8 and 9 show the results.

We can observe that our method consistently converge faster than other baselines for various systems heterogeneity. Particularly, our method outperforms FML in all settings. However, there exits a certain gap compared with FedProx when the data is IID. While, our method is superior to FedProx with the Non-IID degree increasing.

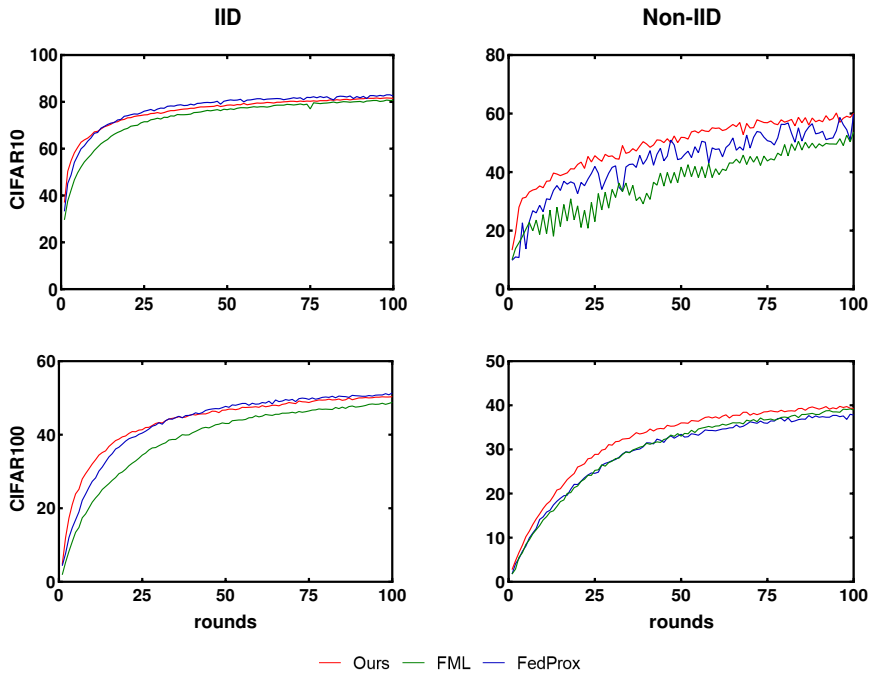

Figure 8. Global accuracy curves in IID and Non-IID settings on CIFAR10/100, where there are 50% stragglers.

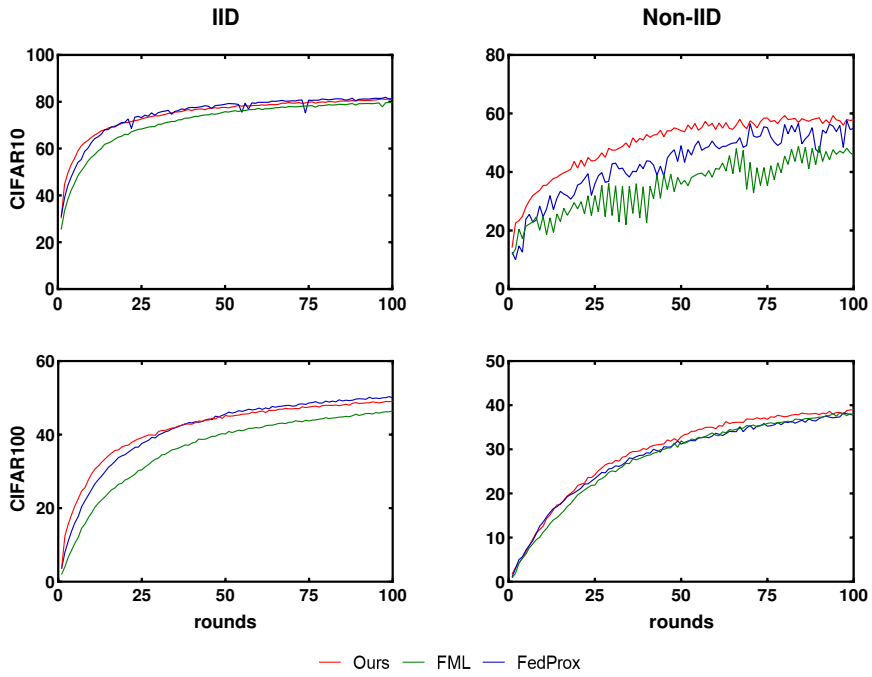

Figure 9. Global accuracy curves in IID and Non-IID settings on CIFAR10/100, where there are 90% stragglers.

6. Conclusions

Motivated by the Bayesian neural network and probability theories, we propose a novel Bayesian aggregation method. In our work, a new algorithm is designed that includes a new training pattern on the client side and a new aggregation strategy on the server side. Since the objective function of the classification problem can be denoted as the form of likelihood, we assume that it can be approximated by a scaled multi-dimensional Gaussian PDF. Therefore, the aggregation process is deemed as a product of scaled Gaussian PDFs. Due to the fact that a product of Gaussian distributions is still a Gaussian distribution, it is easy to evaluate the mode of global distribution based on the parameters communicated from clients. Additionally, in order to get better fitting result, we temper the likelihood part of the posterior which significantly improves the performance of the global model and local model. Moreover, knowledge distillation is introduced to our framework, aiming at dealing the heterogeneity problem. The experiments show that our method can tackle data, model, objective and systems heterogeneities inherent in FL properly. In addition, the convergence is accelerated compared with commonly used baselines. Particularly, our method achieves state-of-the-art results when data is generated in a Non-IID fashion which is common in practical scenario. Finally, we note that the local likelihood can be approximated by other form, considering that the prior probability or the posterior probability doesn't follow Gaussian distribution, which we leave as future work.

Acknowledgments

This work was supported by the Fundamental Research Funds for the Central Universities (22CX03015A, 20CX05012A), the Major Scientific and Technological Projects of CNPC under Grant (ZD2019-183-008) and the National Natural Science Foundation of China (61902429).

Conflict of interest

The authors declare no conflicts of interest.

References

1. H. B. Mcmahan, E. Moore, D. Ramage, B. A. y Arcas, Federated learning of deep networks using model averaging, arXiv: 1602.05629.
2. T. Li, A. Sahu, A. Talwalkar, V. Smith, Federated learning: challenges, methods, and future directions, *IEEE Signal Proc. Mag.*, **37** (2020), 50–60. https://doi.org/10.1109/MSP.2020.2975749
3. D. Li, J. Wang, FedMD: heterogenous federated learning via model distillation, arXiv: 1910.03581.
4. T. Nishio, R. Yonetani, Client selection for federated learning with heterogeneous resources in mobile edge, *2019 IEEE International Conference on Communications (ICC)*, Shanghai, China, 2019, 1–7. https://doi.org/10.1109/ICC.2019.8761315
5. L. Liu, F. Zheng, H. Chen, G. J. Qi, H. Huang, L. Shao, A Bayesian federated learning framework with online Laplace approximation, arXiv: 2102.01936.

6. B. Mcmahan, E. Moore, D. Ramage, S. Hampson, B. A. y Arcas, Communication-efficient learning of deep networks from decentralized data, In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, New York: PMLR, 2017, 1273–1282.
7. B. Wu, X. Dai, P. Zhang, Y. Wang, F. Sun, Y. Wu, FBNet: hardware-aware efficient convnet design via differentiable neural architecture search, *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 2019, 10726–10734. https://doi.org/10.1109/CVPR.2019.01099
8. C. He, M. Annavaram, S. Avestimehr, Fednas: federated deep learning via neural architecture search, arXiv: 2004.08546.
9. T. Shen, J. Zhang, X. Jia, F. Zhang, G. Huang, P. Zhou, et al., Federated mutual learning, arXiv: 2006.16765.
10. C. Xie, S. Koyejo, I. Gupta, Asynchronous federated optimization, arXiv: 1903.03934.
11. W. Wu, L. He, W. Lin, R. Mao, C. Maple, S. Jarvis, SAFA: a semi-asynchronous protocol for fast federated learning with low overhead, *IEEE T. Comput.*, **70** (2021), 655–668. https://doi.org/10.1109/TC.2020.2994391
12. Y. Zhang, Y. Xu, S. Wei, Y. Wang, Y. Li, X. Shang, Doubly contrastive representation learning for federated image recognition, *Pattern Recogn.*, **139** (2023), 109507. https://doi.org/10.1016/j.patcog.2023.109507
13. J. Xiao, C. Du, Z. Duan, W. Guo, A novel server-side aggregation strategy for federated learning in Non-IID situations, *2021 20th International Symposium on Parallel and Distributed Computing (ISPDC)*, Cluj-Napoca, Romania, 2021, 17–24.
14. L. Hu, H. Yan, L. Li, Z. Pan, X. Liu, Z. Zhang, MHAT: an efficient model-heterogenous aggregation training scheme for federated learning, *Inform. Sciences*, **560** (2021), 493–503. https://doi.org/10.1016/j.ins.2021.01.046
15. T. Li, A. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, V. Smith, Federated optimization in heterogeneous networks, *Proceedings of Machine Learning and Systems*, **2** (2020), 429–450.
16. Q. Li, B. He, D. Song, Model-contrastive federated learning, *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nashville, TN, USA, 2021, 10708–10717. https://doi.org/10.1109/CVPR46437.2021.01057
17. M. Mendieta, T. Yang, P. Wang, M. Lee, Z. Ding, C. Chen, Local learning matters: rethinking data heterogeneity in federated learning, *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, New Orleans, LA, USA, 2022, 8397–8406. https://doi.org/10.1109/cvpr52688.2022.00821
18. M. Al-Shedivat, J. Gillenwater, E. Xing, A. Rostamizadeh, Federated learning via posterior averaging: a new perspective and practical algorithms, arXiv: 2010.05273.
19. H. Chang, V. Shejwalkar, R. Shokri, A. Houmansadr, Cronus: robust and heterogeneous collaborative learning with black-box knowledge transfer, arXiv: 1912.11279.
20. Y. Zhang, T. Xiang, T. Hospedales, H. Lu, Deep mutual learning, *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018, 4320–4328. https://doi.org/10.1109/CVPR.2018.00454

21. C. Blundell, J. Cornebise, K. Kavukcuoglu, D. Wierstra, Weight uncertainty in neural network, *The 32nd International Conference on Machine Learning (ICML)*, Lille, France, 2015, 1613–1622.
22. K. Shridhar, F. Laumann, M. Liwicki, A comprehensive guide to bayesian convolutional neural network with variational inference, arXiv: 1901.02731.
23. A. Wilson, P. Izmailov, Bayesian deep learning and a probabilistic perspective of generalization, *The 34th Conference on Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada, 2020, 4697–4708. https://doi.org/10.5555/3495724.3496118
24. O. Goldreich, S. Micali, A. Wigderson, How to play any mental game, or a completeness theorem for protocols with honest majority, In: *Providing sound foundations for cryptography: on the work of shafi goldwasser and silvio micali*, New York: Association for Computing Machinery, 2019, 307–328. https://doi.org/10.1145/3335741.3335755
25. L. T. Phong, Y. Aono, T. Hayashi, L. Wang, S. Moriai, Privacy-preserving deep learning via additively homomorphic encryption, *IEEE T. Inf. Foren. Sec.*, **13** (2018), 1333–1345. https://doi.org/10.1109/TIFS.2017.2787987
26. R. Geyer, T. Klein, M. Nabi, Differentially private federated learning: a client level perspective, arXiv: 1712.07557.
27. P. Kairouz, H. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, et al., Advances and open problems in federated learning, *Found. Trends Mach. Le.*, **14** (2021), 1–210. https://doi.org/10.1561/2200000083
28. Q. Yang, Y. Liu, T. Chen, Y. Tong, Federated machine learning: concept and applications, *ACM T. Intel. Syst. Tec.*, **10** (2019), 12. https://doi.org/10.1145/3298981
29. Y Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, *P. IEEE*, **86** (1998), 2278–2324. https://doi.org/10.1109/5.726791
30. A. Krizhevsky, G. Hinton, Learning multiple layers of features from tiny images, *Technical Report TR-2009*, University of Toronto, Toronto, 2009.
31. Y. Lecun, B. Boser, J. S. Denker, R. E. Howard, W. Habbard, L. D. Jackel, et al., Handwritten digit recognition with a back-propagation network, In: *Advances in Neural Information Processing systems 2*, San Francisco: Morgan Kaufmann Publishers Inc., 1989, 396–404. https://doi.org/10.5555/109230.109279
32. A. Ashukha, A. Lyzhov, D. Molchanov, D. Vetrov, Pitfalls of in-domain uncertainty estimation and ensembling in deep learning, arXiv: 2002.06470.
33. M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, N. Hoang, Y. Khazaeni, Bayesian nonparametric federated learning of neural networks, *The 36th International Conference on Machine Learning*, Long Beach, California, USA, 2019, 7252–7261.
34. T. H. Hsu, H. Qi, M. Brown, Measuring the effects of non-identical data distribution for federated visual classification, arXiv: 1909.06335.

A. Scaled multi-dimensional Gaussian PDF for the linear regression problem

In this section, we try to prove that the likelihood function for the linear regression problem can be denoted as a scaled multi-dimensional Gaussian PDF. Due to the fact that we have proved $p(D|\theta)$ =

$p(X)p(Y|X,\theta)$ in (3.2), we only need to express the term $p(Y|X,\theta)$ in the form of Gaussian PDF with respect to θ.

For the linear regression problem, according to the assumption that random error follows the Gaussian distribution:

$$\begin{aligned} p(Y|X,\theta) &= \prod_{i=1}^{n} p(y_i|x_i,\theta) \\ &= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y_i-\theta^T x_i)^2}{2\sigma^2}\right) \\ &= \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^n exp\left(-\frac{\|\mathcal{X}\theta-\mathcal{Y}\|_2^2}{2\sigma^2}\right) \\ &= \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^n exp\left(-\frac{\theta^T\mathcal{X}^T\mathcal{X}\theta-2\mathcal{Y}^T\mathcal{X}\theta+\mathcal{Y}^T\mathcal{Y}}{2\sigma^2}\right), \end{aligned} \tag{A.1}$$

where $\mathcal{X}\in\mathbb{R}^{n\times d}$ is the design matrix, $\mathcal{Y}\in\mathbb{R}^n$ is the response matrix. As in (3.7), the Gaussian PDF can be written in canonical notation as

$$N(\theta|\mu,\Sigma) = \exp\left[\zeta+\eta^T\theta-\frac{1}{2}\theta^T\Lambda\theta\right], \tag{A.2}$$

where $\Lambda=\Sigma^{-1}$, $\eta=\Lambda\mu$ and $\zeta=-\frac{1}{2}\left(d\log 2\pi-\log|\Lambda|+\eta^T\Lambda^{-1}\eta\right)$. So,

$$\begin{aligned} \exp\left(\eta^T\theta\right) &= \exp\left(\frac{\mathcal{Y}^T\mathcal{X}\theta}{\sigma^2}\right), \\ \exp\left(-\frac{1}{2}\theta^T\Lambda\theta\right) &= \exp\left(-\frac{\theta^T\mathcal{X}^T\mathcal{X}\theta}{2\sigma^2}\right). \end{aligned} \tag{A.3}$$

We can conclude from the above:

$$\eta = \frac{\mathcal{X}^T\mathcal{Y}}{\sigma^2}, \tag{A.4}$$

$$\Lambda = \frac{\mathcal{X}^T\mathcal{X}}{\sigma^2}. \tag{A.5}$$

As in (A.1), it is easy to see that there is only a constant term left in $p(Y|X,\theta)$, which is independent of θ. The left term obviously can be denoted as a multiple of $\exp\zeta$. In conclusion, $p(Y|X,\theta)$ can be denoted as a scaled multi-dimensional Gaussian PDF, which is same to $p(D|\theta)$.

Hence, our claim is true.