FedNP: Towards Non-IID Federated Learning via Federated Neural Propagation

Xueyang Wu[1*], **Hengguan Huang**[2*†], **Youlong Ding**[3], **Hao Wang**[4], **Ye Wang**[2], **Qian Xu**[1]
[1]Hong Kong University of Science and Technology, Hong Kong SAR, China
[2]National University of Singapore, Singapore
[3]Shenzhen University, Shenzhen, China
[4]Rutgers University, Piscataway, NJ, USA
xwuba@connect.ust.hk, huang.hengguan@u.nus.edu, dingyoulon@gmail.com, hw488@cs.rutgers.edu, wangye@comp.nus.edu.sg, qianxu@ust.hk

Abstract

Traditional federated learning (FL) algorithms, such as FedAvg, fail to handle non-i.i.d data because they learn a global model by simply averaging biased local models that are trained on non-i.i.d local data, therefore failing to model the global data distribution. In this paper, we present a novel Bayesian FL algorithm that successfully handles such a non-i.i.d FL setting by enhancing the local training task with an auxiliary task that explicitly estimates the global data distribution. One key challenge in estimating the global data distribution is that the data are partitioned in FL, and therefore the ground-truth global data distribution is inaccessible. To address this challenge, we propose an expectation-propagation-inspired probabilistic neural network, dubbed federated neural propagation (FedNP), which efficiently estimates the global data distribution given non-i.i.d data partitions. Our algorithm is sampling-free and end-to-end differentiable, can be applied with any conventional FL frameworks and learns richer global data representation. Experiments on both image classification tasks with synthetic non-i.i.d image data partitions and real-world non-i.i.d speech recognition tasks demonstrate that our framework effectively alleviates the performance deterioration caused by non-i.i.d data.

1 Introduction

Federated learning (FL) is an increasingly more important machine learning paradigm where many clients jointly train a powerful global model with cross-silo training data. The major target of FL is to utilize the massive data created and collected by different clients while obeying privacy protection regulations such as the European Union's General Data Protection Regulation (GDPR) (Voss 2016). The most representative FL algorithm is FederatedAveraging (`FedAvg`) (McMahan et al. 2017a), which successfully trains a powerful global model while keeping training data on each local client (i.e., each mobile phone). Thanks to its feasibility and effectiveness, `FedAvg` has been successfully adopted in many applications, such as speech recognition (Tan et al. 2021) and language modeling (McMahan et al. 2017b). Recently, more and more FL researchers have turned their attention to a more practical setting where the data distributions are non-i.i.d (Zhu et al. 2021); in practice, a federation usually consists of different clients from diverse sources whose data distributions are intrinsically distinct. In such non-i.i.d settings, typical FL algorithms often fail to achieve reasonable performance.

Taking a simple polynomial curve fitting task as an example, as shown in Figure 1(a), our goal is to train a model that can fit the data points sampled from a polynomial function. To simulate an extremely non-i.i.d federated data partition, the data points are split into three groups with disjoint ranges of x-coordinates, denoted as $\mathbf{X} = \{\mathbf{X}_k\}_{k=1}^{3}$. Three federated clients, each training a local model ($\boldsymbol{\theta} = \{\boldsymbol{\theta}_k\}_{k=1}^{3}$), then collaboratively learn a global model via FL algorithms, e.g., `FedAvg`. The result shows that while the local models successfully fit their local data in the local training steps, the final prediction is nearly random for test data points. The reason is that the averaging of defective local models adopted by FedAvg cannot provide a global model with sufficient capacity to describe the entire data distribution (see Figure 1(b)). With such insight, we then approach non-i.i.d FL from a new perspective by explicitly modeling a latent global data distribution to enforce local models to have a global view as an auxiliary task.

Existing works in Bayesian federated learning (BFL) (Al-Shedivat et al. 2020; Chen and Chao 2020) focus on directly inferring the global model distribution by aggregating over local model distributions, consequently failing to capture a latent global data distribution. A BFL framework augmented with latent global data distribution should offer several advantages, including (1) a lower dimensional representation of input data uncertainty, caused by limited access to global data ; and (2) a greater expressive power to capture the complex dependency underlying the input data across clients than conventional BFL frameworks.

Despite the enormous successes of Bayesian learning and deep learning, it is still challenging to infer such a global data distribution. The reason is that in non-i.i.d federated

*These authors contributed equally.
†Correspondence to: Hengguan Huang <huang.hengguan@u.nus.edu>

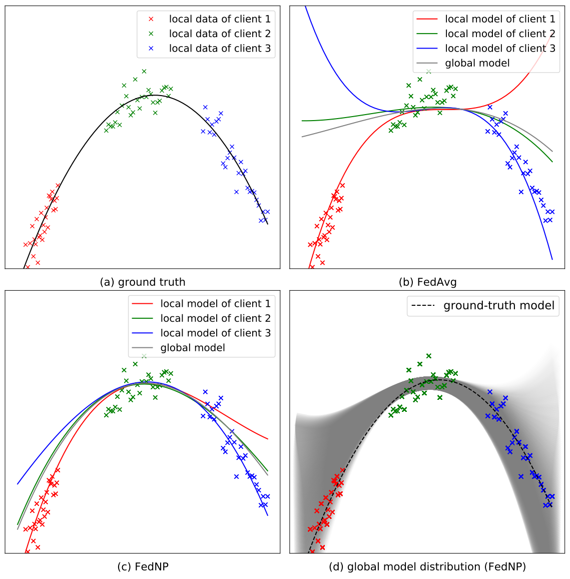


Figure 1: Toy example on polynomial curve fitting task. Data points are denoted by '$\times$' and models are denoted by '—'. (a) The ground-truth curve where the observed points are sampled (with Gaussian noise). (b) and (c) The local models of three clients and the global model trained by FedAvg and FedNP, respectively. (d) global model distribution estimated by FedNP (within three standard deviations of the mean).

scenarios, each client only has access to their local data, and therefore the ground-truth global data distribution is inaccessible due to their distinct local data distributions. For instance, Laplace approximation (LA) and Monte-Carlo (MC) approximation, adopted by conventional BFL frameworks (Al-Shedivat et al. 2020; Chen and Chao 2020; Liu et al. 2021), are widely adopted techniques to approximate an intractable distribution, but it relies on the observation of global data and therefore cannot apply to our problem settings directly. Another Bayesian inference (BI) method, expectation propagation (EP) (Minka 2013), does not require access to global data and has the potential to approximate the global data distribution. However, applying EP to facilitate our auxiliary task is highly non-trivial:

- EP impractically requires to calculate the intractable likelihood term $\mathcal{L}(\boldsymbol{\theta}_k, \mathbf{X}_k|\mathbf{z})$ due to the unavailability of $\boldsymbol{\theta}_k$, the model parameters for the current mini-batch.
- Since only model parameters for the previous mini-batch are available, we empirically tested an intuitive approach – using such parameters to evaluate the likelihood – and found that the model would not reliably converge.

To overcome these challenges, we reformulate EP by factorizing the latent global data distribution $p(\mathbf{z}|\mathbf{X})$ into several approximate posterior factors $q(\mathbf{z}|\mathbf{X}_k)$, without the need to calculate the likelihood term. We further propose an EP-like probabilistic neural network, dubbed federated neural propagation (FedNP), which follows the general update rule of EP and efficiently estimates the global data distribution. More specifically, FedNP enhances the local training task with an auxiliary task that explicitly estimates a latent global data distribution, stabilizing and correcting the process of local training (See Figure 1(c)), whereby a probabilistic neural network is adopted to map such distribution to a global model distribution, consequently regularizing the local model by avoiding it sinking into local data distribution. Figure 1(d) shows that FedNP successfully estimates a correct global model parameter distribution and avoids catastrophic failure of local models on unseen data, resulting in a more accurate global model.

Furthermore, unlike existing algorithms for deep-learning-based EP (Jylänki, Nummenmaa, and Vehtari 2014; Soudry, Hubara, and Meir 2014; Heess, Tarlow, and Winn 2013) that require numerical approximation or sampling, we develop a closed-form approximation algorithm for inferring the latent global data distribution, thereby improving the efficiency of modeling federated data.

The major contributions of our work are three-fold:

- We present a new perspective on handling non-i.i.d federated data, which explicitly considers the global data distribution when performing the local training steps.
- We reformulate EP to remove the dependence on the intractable likelihood term and derive a closed-form approximation for the latent global data distribution using deep neural networks, allowing our algorithm to be end-to-end differentiable.
- Our experiments on real-world non-i.i.d image and speech datasets demonstrate that FedNP effectively alleviates the performance deterioration caused by non-i.i.d data compared to state-of-the-art baselines.

2 Related Work

Federated learning (FL), as a new collaborative learning paradigm, has been gaining more attention in recent years (McMahan et al. 2017a; Yang et al. 2019), and the challenges of FL from non-i.i.d data have been noticed, especially for supervised learning tasks (Li et al. 2021, 2020; Zhao et al. 2018; Xie, Koyejo, and Gupta 2019; Yu et al. 2020). A fundamental idea of these works is to regularize local models during the local training step in FL. Most of them either directly take the averaged model of previous federated turn as the ground truth to regularize the local model training, e.g. `FedProx` (Li et al. 2020) and `MOON` (Li, He, and Song 2021), or use a dynamic regularizer, e.g. `FedDyn` (Acar et al. 2021), `FedDC` (Gao et al. 2022), or estimate a *correction* term for local models based on previous aggregated models, e.g. `SCAFFOLD` (Karimireddy et al. 2020). In contrast, we introduce into local model training an auxiliary task that explicitly estimates the global data distribution from partitioned data, thereby encouraging the local models to be more expressive and preventing them from sinking into local distributions.

From the perspective of Bayesian inference (BI), Bayesian federated learning (BFL) has been studied before, but mainly under the context of model aggregation. For instance, most existing BFL methods (Al-Shedivat et al. 2020; Liu et al. 2021; Chen and Chao 2020) focus on inferring global model distribution by aggregating over local model distributions, while our method aims to regularize local model distributions by inferring a latent global data distribution from input data across clients. Furthermore, similar to `FedAvg` (McMahan et al. 2017a), FedPA (Al-Shedivat et al. 2020) targets a general federated learning setting, without considering a more challenging extremely non-i.i.d federated settings, such as ours. To handle non-i.i.d data, FedBE (Chen and Chao 2020) propose a Bayesian ensemble method for server-side model aggregation. However, it requires collecting an unlabeled data set across clients, failing to handle our non-i.i.d FL setting. Other key aspects that distinguish our FedNP from the above methods are as follows. First, FedNP reformulates expectation propagation (EP) with neural networks for handling non-i.i.d FL settings. Inherits from EP, FedNP does not require access to global data during training. In contrast, the Laplace approximation (LA) and Monte-Carlo (MC) approximation, adopted by the above methods, relies on the observation of global data. Second, our FedNP algorithm is sampling-free and end-to-end differentiable, leading to a more efficient and scalable framework. (Please refer to Appendix C for related work on EP with neural networks.)

3 Background

Federated Learning (FL). Following (Li et al. 2019), we formulate FL with the non-i.i.d data distributed on separate clients We denote the number of clients in the FL system as K and the private annotated dataset in the party $k \in \{1...K\}$ as $(\mathbf{X}_k, \mathbf{Y}_k)$, where $\mathbf{X}_k = \{\mathbf{x}_{k,i}\}_{i=1}^{n_k}$ contains n_k observations and $\mathbf{Y}_k = \{\mathbf{y}_{k,i}\}_{i=1}^{n_k}$ includes the corresponding training labels.

The FL target is to find a global model $\boldsymbol{\theta}$ with private data distributed on different clients by the following iterative process (i.e., `FedAvg`):

1. Each client k optimizes their local model $\boldsymbol{\theta}_k$ on its private data set $(\mathbf{X}_k, \mathbf{Y}_k)$ with the objective $J(\cdot)$ according to Eq. 1 below, and send $\boldsymbol{\theta}_k$ to the server.
2. The server computes the global model $\boldsymbol{\theta}_{avg}$ with weighted average of client models $\{\boldsymbol{\theta}_k\}_{k=1}^{K}$ with $\boldsymbol{\theta}_{avg} \triangleq \sum_{k=1}^{K} v_k \boldsymbol{\theta}_k$, where v_k is the weight of the corresponding party k such that $v_k \geq 0$ and $\sum_{k=1}^{K} v_k = 1$.
3. The server sends $\boldsymbol{\theta}$ to all clients and repeat step 1-3 until it reaches the stop criterion.

$$\min_{\boldsymbol{\theta}_k} J(\boldsymbol{\theta}_k; \mathbf{X}_k, \mathbf{Y}_k) \triangleq \frac{1}{n_k} \sum_{i=1}^{n_k} \ell\left(\mathbf{x}_{k,i}, \mathbf{y}_{k,i}; \boldsymbol{\theta}_k\right), \tag{1}$$

where $\ell(\cdot;\cdot)$ is the localized loss function, e.g. cross-entropy loss for classification tasks.

Expectation Propagation (EP). Before introducing our FedNP, we begin by reviewing the traditional EP algorithm (Minka 2013). EP is a deterministic approximation algorithm, often used for Bayesian inference of posterior distributions of model parameters, which is believed to be able to provide significantly more accurate approximations than VI (Jordan et al. 1999) and LA (MacKay 1992) methods.

Consider a regression task that predicts some attributes of interest $\boldsymbol{\theta} = \{\boldsymbol{\theta}_k\}_{k=1}^{K}$ given observations $\mathbf{X} = \{\mathbf{X}_k\}_{k=1}^{K}$, where K is number of data partitions. Assume both $\{\boldsymbol{\theta}_k\}_{k=1}^{K}$ and $\{\mathbf{X}_k\}_{k=1}^{K}$ are conditionally independent given the latent variable $\mathbf{z}$. As the posterior distribution of interest $p(\mathbf{z}|\mathbf{X})$ is computationally intractable, EP attempts to approximate it with a tractable approximating distribution $q(\mathbf{z})$, which can be further factorized into multiple approximate factors:

$$q(\mathbf{z}) \propto p_0(\mathbf{z}) \prod_{i=1}^{N} q_i(\mathbf{z}), \tag{2}$$

where $p_0(\mathbf{z})$ is the prior distribution; the approximate factor $q_i(\mathbf{z})$ is iteratively refined so that they capture the contribution of each data partition $\mathbf{X}_k$ to the posterior $p(\mathbf{z}|\mathbf{X})$:

$$p(\mathbf{z}|\mathbf{X}) \propto p_0(\mathbf{z}) \prod_{k=1}^{K} \mathcal{L}(\boldsymbol{\theta}_k, \mathbf{X}_k|\mathbf{z}), \tag{3}$$

where $\mathcal{L}(\boldsymbol{\theta}_k, \mathbf{X}_k|\mathbf{z})$ denotes likelihood. If mini-batch optimization is adopted, only mini-batches are used to evaluate such likelihood instead of the entire data partition iteratively. Specifically, EP iterates over the following steps:

1. Construct the cavity distribution by removing one of the approximate factors, i.e., the k-th factor. It can be written as: $q_{-k}(\mathbf{z}) \propto p_0(\mathbf{z}) \prod_{j \neq k} q_j(\mathbf{z})$
2. Integrate the likelihood $\mathcal{L}(\boldsymbol{\theta}_k, \mathbf{X}_k|\mathbf{z})$ to the cavity to produce the hybrid distribution: $h_k(\mathbf{z}) \propto q_{-k}(\mathbf{z})\mathcal{L}(\theta_k, \mathbf{X}_k|\mathbf{z})$.
3. Update the parameters of the k-th approximated factor $q_k(\mathbf{z})$ through minimizing the KL divergence between the hybrid distribution $h_k(\mathbf{z})$ and the approximated distribution $q(\mathbf{z})$, namely, KL $[h_k(\mathbf{z}) \| q_{-k}(\mathbf{z}) q_k(\mathbf{z})]$.
4. Update the approximated distribution $q(\mathbf{z})$ by including the updated approximated factor $q_k(\mathbf{z})$: $q(\mathbf{z}) \propto q_{-k}(\mathbf{z}) q_k(\mathbf{z})$.

When $q_k(\mathbf{z})$ is assumed to follow an exponential family distribution (e.g., a Gaussian), minimization of the KL divergence in Step 3 can be reduced to moment matching (ichi Amari and Nagaoka 2000). However, when applying EP to deep neural networks, this moment matching step is computationally intractable, requiring numerical approximation or sampling, and therefore has to compromise between accuracy and efficiency. It's also worth noting that $\boldsymbol{\theta}_k$ in our problem settings is the local model parameters for the current mini-batch. Due to its unavailability when evaluating the likelihood term during training, EP cannot be directly applied to our problem settings. Addressing these problems are primary focuses of our model.

4 Methodology

As shown in the toy experiment, local models tend to sink into local data distributions, resulting in a model drift in the local training step of FL with non-i.i.d data. In the following sections, we firstly formalize such problems and then present *a sampling-free and fully differentiable deep probabilistic neural network* for achieving *end-to-end goals* of inferring latent global data distribution from non-i.i.d partitioned data, thereby regularizing local models and preventing them from sinking into local data distributions.

4.1 Problem Formulation

We consider *non-i.i.d* FL settings with classification as major task and aim to enhance the local training steps of FL

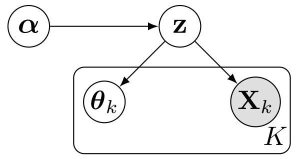


Figure 2: Graphical Model of FedNP. α is the uniform prior of the latent variable $\mathbf{z}$.

with an auxiliary task that explicitly models a latent global data distribution to constrain local training. Specifically, suppose we are given K annotated data partitions $(\mathbf{X}, \mathbf{Y}) = \{(\mathbf{X}_k, \mathbf{Y}_k)\}_{k=1}^K$ privately maintained by K clients, where $\mathbf{X}_k = \{\mathbf{x} | \mathbf{x} \sim p_k(\mathbf{x})\}$ containing data items that are drawn from non-identical distributions, i.e., $p_k(\mathbf{x}) \neq p_{k'}(\mathbf{x})$ for all $k \neq k'$, and $\mathbf{Y}_k$ includes the corresponding training labels. Suppose each client optimizes their local model $\boldsymbol{\theta}_k$ on their private data $(\mathbf{X}_k, \mathbf{Y}_k)$. Assume both $\boldsymbol{\theta}_k$ and $\mathbf{X}_k$ are conditionally independent given the latent variable $\mathbf{z}$ which represents the global data distribution. The probabilistic model is shown using the standard directed graphical notation in Figure 2. Our model belongs to the broad category of *Bayesian deep learning* (Wang and Yeung 2016, 2020), with a graphical model component as additional inductive bias for the probabilistic neural network. The auxiliary task aims to infer the posterior distribution $p(\mathbf{z}|\mathbf{X})$, whereby the conditional global model distributions $p(\boldsymbol{\theta}_k|\mathbf{z})$ are produced, regularizing each local model $\boldsymbol{\theta}_k$ to have a global view.

4.2 Federated Neural Propagation

Our proposed Federated Neural Propagation (FedNP) is a special type of message passing algorithm for inferring a latent global data distribution using localized inference over non-i.i.d data partitions and preventing FL local models from sinking into local data distributions. FedNP follows a general updating rule of another message passing algorithm, EP, but adopts a different factorization of the target distribution to remove the dependence on the intractable likelihood. The advantage of FedNP is that it can approximate the global data distribution using fully differentiable neural networks that encode the respective closed-form approximation, rather than inefficient sampling and numerical quadrature adopted by existing neural network-based EP methods.

Specifically, assuming a uniform distribution for the prior $p_0(\mathbf{z})$, the posterior of the latent global data distribution $p(\mathbf{z}|\mathbf{X})$ can be factorized as the product of the posteriors conditioned on local data:

$$\begin{aligned} p(\mathbf{z}|\mathbf{X}) &\propto p_0(\mathbf{z}) \prod_{k=1}^{K} p(\mathbf{X}_k|\mathbf{z}) \\ &\propto \prod_{k=1}^{K} p_0(\mathbf{z}) p(\mathbf{X}_k|\mathbf{z}) \propto \prod_{k=1}^{K} p(\mathbf{z}|\mathbf{X}_k). \end{aligned} \tag{4}$$

Eq. (4) is derived by simply applying Bayes' theorem with the assumption of a uniform prior for $p_0(\boldsymbol{\theta})$. We then follow EP to adopt K approximate factors to approximate $p(\mathbf{z}|\mathbf{X})$. Specifically, we use $q_k(\mathbf{z})$ as the Gaussian distributed approximate factor for $p(\mathbf{z}|\mathbf{X}_k)$ and have

$$q(\mathbf{z}) \propto \prod_{k=1}^{K} q_k(\mathbf{z}), \tag{5}$$

where $q(\mathbf{z})$ approximates global data distribution $p(\mathbf{z}|\mathbf{X})$; $q(\mathbf{z}) = \mathcal{N}(\mathbf{z}_m, \mathbf{z}_s)$, where $\mathbf{z}_m$ and $\mathbf{z}_s$ are the mean and variance, respectively; $q_k(\mathbf{z}) = \mathcal{N}(\mathbf{z}_{m,k}, \mathbf{z}_{s,k})$, where $\mathbf{z}_{m,k}$ and $\mathbf{z}_{s,k}$ are the mean and variance, respectively. Unlike Eq. (2), we omit the prior $p_0(\mathbf{z})$ in the expression as we assume it is a uniform distribution. In the following subsections, we will show how to calculate the mean and variance of $q(\mathbf{z})$ analytically using fully differentiable neural networks.

Infer the Approximate Global Data Distribution $q(\mathbf{z})$. We follow the general updating steps of EP and propose a closed-form approximation of $p(\mathbf{z}|\mathbf{X})$ as $q(\mathbf{z})$. We firstly initialize mean and variance of each Gaussian factor, $q_k(\mathbf{z}) = \mathcal{N}(\mathbf{z}_{m,k}, \mathbf{z}_{s,k})$. We follow Step 1 of EP in Section 3 to construct the cavity and the hybrid distributions $q_{-k}(\mathbf{z})$, $h_k(\mathbf{z})$:

$$q_{-k}(\mathbf{z}) \propto \prod_{j \neq k} q_j(\mathbf{z}), \tag{6}$$

where the cavity distribution $q_{-k}(\mathbf{z}) = \mathcal{N}(\mathbf{z}_{m,-k}, \mathbf{z}_{s,-k})$, whose mean $\mathbf{z}_{m,-k}$ and variance $\mathbf{z}_{s,-k}$ are calculated by taking product of Gaussain factors.

We then follow Step 2 of EP in Section 3 to construct the hybrid distribution $h_k(\mathbf{z})$:

$$h_k(\mathbf{z}) \propto p(\mathbf{z}|\mathbf{X}_k) q_{-k}(\mathbf{z}), \tag{7}$$

Inspired by nature-parameter-network (NPN) (Wang, Shi, and Yeung 2016), to parameterize the posterior $p(\mathbf{z}|\mathbf{X}_k)$, we adopt deep neural networks to probabilistically propagate information from X_k to the latent variable $\mathbf{z}$:

$$p(\mathbf{z}|\mathbf{X}_k) = \phi(\mathbf{z}, \mathbf{X}_k), \tag{8}$$

where ϕ is a neural network (see implementation details in Appendix of corresponding experiment settings).

When following Step 3 and 4 of EP in Section 3 to update $q(\mathbf{z})$, we find that moment matching is not analytical, due to deep neural networks involved in $\phi(\mathbf{z}, \mathbf{X}_k)$. To overcome this challenge, we propose the following theorem which provides a closed-form solution for moment matching, and thus obtaining closed-form approximation, $q(\mathbf{z})$, for the posterior of the global data distribution, $p(\mathbf{z}|\mathbf{X})$.

Theorem 1. *Suppose we are given a data partition $\mathbf{X}_k$ located at the k-th party during FL. Assume data partitions $\{\mathbf{X}_k\}_{k=1}^K$ are conditionally independent given a latent variable $\mathbf{z}$. Let $f : \mathcal{R}^{|\mathbf{X}_k|} \to \mathcal{R}^1$ be a neural network taking as input $\mathbf{X}_k$. Let $C = f(\mathbf{X}_k)\mathbf{z}$ and $C \sim \mathcal{N}(C_{m,-k}, C_{s,-k})$. Let $q_{-k}(\mathbf{z})$ and $h_k(\mathbf{z})$ be the cavity distribution (defined in Eq. (6)) and the hybrid distribution (defined in Eq. (7)), respectively. We further define $q(\mathbf{z}) = \mathcal{N}(\mathbf{z}_m, \mathbf{z}_s)$ as in Eq. (5). There exists a function $\phi : \mathcal{R}^{|\mathbf{z}|} \times \mathcal{R}^{|\mathbf{X}_k|} \to \mathcal{R}^1$, such that the update rules of $\mathbf{z}_m$ and $\mathbf{z}_s$ can be written in closed form as:*

$$\mathbf{z}_m = S_1, \tag{9}$$

$$\mathbf{z}_s = S_2 - S_1^2, \tag{10}$$

where

$$S_1 = \left[(C_{m,-k} + C_{s,-k}) E_C(\sigma(C)) - C_{s,-k} E_C(\sigma^2(C))\right] / S_0 f(\mathbf{X}_k), \tag{11}$$

$$S_2 = \left[(C_{m,-k} + 2C_{s,-k}) E_C(\sigma^2(C)) - 2C_{s,-k} E_C(\sigma^3(C))\right] / S_0 f^2(\mathbf{X}_k), \tag{12}$$

$$S_0 = E_C(\sigma(C)). \tag{13}$$

We leave the proof in Appendix A.2.

Given a data partition $\mathbf{X}_k$, Theorem 1 provides the closed-form updates for mean and variance of $q(\mathbf{z}) = \mathcal{N}(\mathbf{z}_m, \mathbf{z}_s)$. Note that Theorem 1 requires an auxiliary distribution of the latent variable $C = f(\mathbf{X}_k)\mathbf{z}$. If the cavity distribution is Gaussian, i.e., $q_{-k}(\mathbf{z}) = \mathcal{N}(\mathbf{z}_{m,-k}, \mathbf{z}_{s,-k})$, C also follows a Gaussian distribution. Therefore denoting this distribution of C as $q_{c,-k}(C)$, we have that:

$$q_{c,-k}(C) = \mathcal{N}(C_{m,-k}, C_{s,-k}) \propto \prod_{j \neq k} q_j(C), \qquad (14)$$

where $\mathcal{N}(C_{m,-k}, C_{s,-k})$ is a Gaussian distribution with the mean and variance:

$$\left[\begin{array}{c} C_{m,-k}, C_{s,-k} \end{array}\right]^{\top} = \left[\begin{array}{c} f(\mathbf{X}_k)\,\mathbf{z}_{m,-k}, f^2(\mathbf{X}_k)\,\mathbf{z}_{s,-k} \end{array}\right]^{\top}. \qquad (15)$$

Theorem 1 further requires the first three moments of $q_{c,-k}(C)$. We therefore adopt Theorem 2 below to calculate these first three moments, i.e., $E_C(\sigma(C))$, $E_C(\sigma^2(C))$, and $E_C(\sigma^3(C))$, thereby providing an analytical solution for calculating the first two moments of the hybrid distribution $h_k(\mathbf{z})$, S_1 and S_2. Theorem 1 then follows Step 4 (Section 3) to update $q(\mathbf{z}) = \mathcal{N}(\mathbf{z}_m, \mathbf{z}_s)$, whose parameters $\mathbf{z}_m$ and $\mathbf{z}_s$ can be updated in closed form as well.

Theorem 2. *Suppose $C \sim \mathcal{N}(C_{m,-k}, C_{s,-k})$. Let $d \geqslant 1$ be a positive integer. There exist two real constants a and b, such that the first d moments can be expressed in closed form:*

$$E_C(\sigma^d(C)) \approx \sigma\left(\frac{a(C_{m,-k}+b)}{\sqrt{1+\zeta^2 a^2 C_{s,-k}}}\right). \qquad (16)$$

The proof for Theorem 2 is in Appendix A.3.

Generally, *Theorem 1 & 2 allow our FedNP algorithm to be end-to-end differentiable.* We present such an algorithm with a mild extension to the commonly used FL framework (`FedAvg`) in Algorithm 1 (see Appendix B).

Infer the Conditional Global Model Distribution $p(\boldsymbol{\theta}_k|\mathbf{z})$. In this section, we aim to infer the conditional global model distribution $p(\boldsymbol{\theta}_k|\mathbf{z})$ based on the approximate global data distribution $q(\mathbf{z})$. Since natural parameter network (NPN) (Wang, Shi, and Yeung 2016) allows inferring a target distribution based on an input distribution through layers of efficient sampling-free probabilistic transformation, we adopt such a model to infer $p(\boldsymbol{\theta}_k|\mathbf{z})$ given $q(\mathbf{z})$, i.e.:

$$p(\boldsymbol{\theta}_k|\mathbf{z}) = \mathrm{NPN}(q(\mathbf{z})), \qquad (17)$$

where $\mathrm{NPN}(\cdot)$ represents the natural parameter network, whose implementation is detailed in Appendix.

4.3 Learning

Our design of the loss function aims at regularizing local models to have a global view using the conditional global model distribution $p(\boldsymbol{\theta}_k|\mathbf{z})$. Since we only have $\boldsymbol{\theta}_k$ for the previous mini-batch, we take the conditional global model distribution $p(\boldsymbol{\theta}_k|\mathbf{z})$ to match an auxiliary Gaussian distribution $p(\hat{\boldsymbol{\theta}}_k) = \mathcal{N}(\boldsymbol{\theta}_k, \epsilon)$, where ϵ is a vector with all entries set as small constants. Following (NPN) (Wang, Shi, and Yeung 2016), the loss can be written as:

$$\ell = KL\left[p(\boldsymbol{\theta}_k|\mathbf{z}) \| p(\hat{\boldsymbol{\theta}}_k)\right] \qquad (18)$$

To jointly train FedNP and local models of FL, our final loss for local training can be written as:

$$J(\boldsymbol{\theta}_k; \mathbf{X}_k, \mathbf{Y}_k) + \lambda\ell, \qquad (19)$$

where $J(\cdot)$ is the loss for local training (Eq. 1) and λ is the hyperparameter balancing two losses.

Approximation Error and Computational Efficiency. In Lemma 2, we use the probit function $\Phi(\zeta a(C+b))$ to approximate $\sigma^d(C)$. Similar approximation is also adopted by (Wang, Shi, and Yeung 2016) and (Wang and Manning 2013). Both the numerical and theoretical approximation analysis has been well studied in Section 3.2 (Wang, Shi, and Yeung 2016). Our closed-form update steps are efficient in computation. Suppose that there is a system with K data partitions/clients (note that K is usually known beforehand) and the likelihood term is tractable, for EP with neural networks in (Jylänki, Nummenmaa, and Vehtari 2014) which requires numerical quadratures, its computational complexity is $O(MK)$, where M is the number of quadratures points. For EP with MCMC, its computational complexity is $O(NK)$, where N is the number of MCMC samples. To approach a good approximation, both N and M should be sufficiently large (Barthelmé and Chopin 2014). In contrast, with our closed-form update, FedNP's computational complexity is reduced to $O(K)$.

We show the computational efficiency as well as the approximation accuracy in Appendix D.3.

5 Experiments

In this section, we evaluate our FedNP on non-i.i.d. cross-silo datasets on numerical regression, image classification, and speech recognition tasks to demonstrate its effectiveness. We describe the experimental details, including the environments, implementations, etc., in Appendix D. The datasets used in our experiments are public, and codes can be found in the supplementary file.

5.1 Toy Experiments

Dataset. We evaluate our FedNP on a toy cross-silo non-i.i.d dataset. Given x, we will use the following quadratic polynomial function to generate the labels.

$$y = -x^2 + 2x + 5 + \epsilon \qquad (20)$$

where $\epsilon \sim \mathcal{N}(0, 5)$. We assume three clients and sample data points from three disjoint segments as the local training data for each client (see Figure 1(a)).

Results. Figure 1(b) visualizes the trained local models and global model of FedAvg. The local training leads the model to learn a biased curve due to the skewed local data distribution. Although all the three local models fit their own data points perfectly, they fail to generalize well to the unseen global data. This is because the global model obtained by averaging fails to capture the global data distribution. In contrast, our FedNP is more robust (see Figure 1(c)), and the estimated global model distribution is more accurate (Figure 1(d)), which corrects the local training and avoids performance deterioration. The experiment setup, quantitative results along with qualitative comparison with FedPA are shown in Appendix D.2.

5.2 Image Classification with Non-IID Image Datasets

Task. We evaluate the performance of FedNP on image classification, which is the most fundamental task in image datasets under the supervised learning setting. We focus on a setting where the local data distributions across parties are non-i.i.d, which increases the difficulty of training.

Datasets. We conduct experiments on CIFAR-100 (Krizhevsky, Nair, and Hinton 1995) and Tiny-Imagenet (Le and Yang 2015). Similar to previous study (Wang et al. 2020), we use Dirichlet distribution to generate the non-i.i.d data partition among clients. With the above partitioning strategy, each client may have relatively few data samples in some classes. We leave details in Appendix D.3.

Baselines. We compare FedNP with four approaches including (1) `FedAvg` (McMahan et al. 2017a), (2) `FedProx` (Li et al. 2020), (3) `MOON` (Li, He, and Song 2021), (4) `SCAFFOLD` (Karimireddy et al. 2020), (5) `FedDyn` (Acar et al. 2021), (6) `FedDC` (Gao et al. 2022), (7) `FedPA` (Al-Shedivat et al. 2020), and (8) `FedLA` (Liu et al. 2021) as discussed in related work. For more details, please refer to Appendix D.3. To further validate the robustness and scalability of FedNP, we vary the number of clients from 10, 50, and 100. To keep the main paper concise, we leave the full performance table in Appendix D.3 and show the performance on the 10-client default setting in Table 1.

Results. Table 1 shows the top-1 test accuracy of all approaches with the 10-client default setting. Comparing different FL approaches, we can observe that FedNP is consistently the best approach. It outperforms FedAvg by around 2% accuracy on average. FedProx's accuracy is worse than FedAvg. This is because directly minimizing the distance between the global model and the local model will negatively affect convergence (note that the initial local model in the local training phase is exactly the server-side model). MOON also minimizes the distance between the global model and local models. Its difference from FedProx is that MOON defines the distance in the feature space instead of in the parameter space. FedDyn's and FedDC's dynamic regularizers do not offer significant performance gain for sophisticate models. Therefore, their performances are also close to FedAvg. The theoretical guarantee of SCAFFOLD relies on the strong smoothness assumption, which does not necessarily hold true in deep learning. Therefore, its performance suffers a lot on these image classification tasks, which has also been verified by Li et al. (2021). Compared with the state-of-the-art BFL frameworks, our FedNP outperforms FedLA (Liu et al. 2021) and FedPA (Al-Shedivat et al. 2020), which employ Laplace approximation/MC-approximation for model aggregation, by around 5% and 2% in terms of accuracy, respectively. This demonstrates the importance of inferring a latent global data distribution with a sampling-free and end-to-end differentiable BFL framework. Generally, our experimental results on CIFAR 100 and TinyImageNet show superior performance and demonstrate the effectiveness of the proposed FedNP.

To facilitate the qualitative evaluation of our FedNP, we randomly sample data from the first 10 classes of CIFAR-100 and utilize t-SNE (Van der Maaten and Hinton 2008) to visualize the corresponding ResNet18 backbone (served as the shared encoder for classifier and $q(\mathbf{z})$) output produced by FedNP in Figure 3. Figure 3 shows two advantages of our FedNP compared to FedAvg. First, FedNP is capable of learning more discriminative features among classes with more inner-class compactness and larger inter-class margins. Second, FedNP is able to preserve the structure of global data distribution even in the clients (e.g., client 1 in Figure 3(b)) whose class distributions are extremely imbalanced; in contrast, the data distribution learned by FedAvg is relatively messy. Besides, we visualize the predicted model distribution by FedNP in Figure 5 to verify the efficacy of estimating the global model distribution, as shown in Appendix D.3. Moreover, to evaluate the computational efficiency and accuracy of the approximation for the proposed closed-form update, we compare FedNP with the closed-form update and its sampling-based variant. We leave the details and results in Appendix D.3.

Table 1: The top-1 accuracy of FedNP and the other baselines of the 10-client setting on test datasets. We run three trials and report the mean and standard deviation.

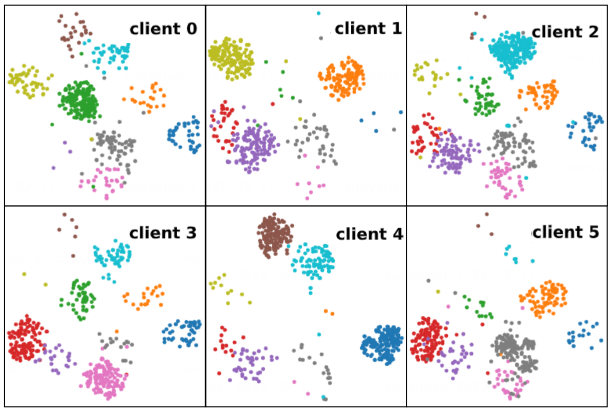
Methods	CIFAR100 ↑	TinyImageNet ↑
FedAvg (McMahan et al. 2017a)	$62.93\% \pm 0.3\%$	$51.89\% \pm 0.5\%$
FedProx (Li et al. 2020)	$62.11\% \pm 0.2\%$	$50.63\% \pm 0.2\%$
MOON (Li, He, and Song 2021)	$63.07\% \pm 0.3\%$	$51.44\% \pm 0.3\%$
SCAFFOLD (Karimireddy et al. 2020)	$53.50\% \pm 0.3\%$	$46.28\% \pm 0.2\%$
FedDyn (Acar et al. 2021)	$62.04\% \pm 0.3\%$	$47.34\% \pm 0.2\%$
FedDC (Gao et al. 2022)	$63.15\% \pm 0.3\%$	$47.34\% \pm 0.2\%$
FedPA (Al-Shedivat et al. 2020)	$63.44\% \pm 0.3\%$	$49.57\% \pm 0.2\%$
FedLA (Liu et al. 2021)	$60.98\% \pm 0.4\%$	$50.23\% \pm 0.2\%$
FedNP (ours)	**65.03%** $\pm 0.2\%$	**53.18%** $\pm 0.3\%$

Moreover, to evaluate the computational efficiency and accuracy of the approximation for the proposed closed-form update, we compare FedNP with the closed-form update and its sampling-based variant. We leave the details and results in Appendix D.3.

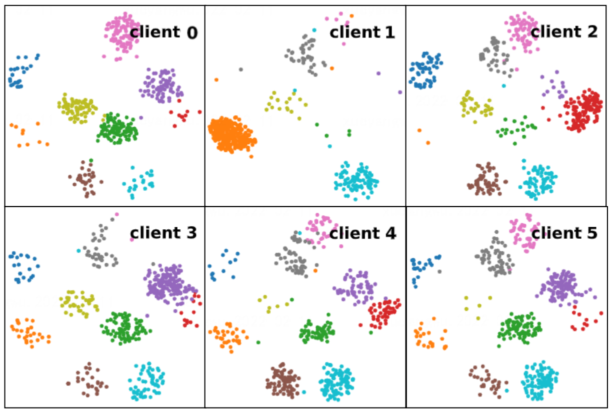
5.3 Speech Recognition with Large-scale Non-IID Conversational Speech Dataset

Task. We evaluate our method on a speech task to demonstrate the efficacy of our proposed FedNP on natural real-world non-i.i.d speech datasets. We applied FedNP to the learning of deep neural network acoustic models. The goal of this task is to transcribe a piece of speech to text.

Datasets. We evaluate our proposed method on a challenging real conversational speech dataset CHiME-5 (Barker et al. 2018), whose data are collected from daily life with diverse environments and various speakers. CHiME-5 is a large-scale corpus of real-world multi-speaker conversa-



(a) Features of local images extracted by the FedAvg model.



(b) Features of local images extracted by the FedNP model.

Figure 3: T-SNE visualizations on features of local images extracted by the models trained via FedAvg and FedNP on CIFAR-100. For better presentation, only the first 10 classes (with the class id of 0, 1, ..., 9) and the first 6 clients are presented.

tional speech in home environments. The training set is originally collected by 16 conversation sessions, each consisting of different speakers at different places and talking about different topics; these sessions compose a natural non-i.i.d. data partition. Hence, the non-i.i.d. federated speech setting has 16 clients, one for each session during training, and applies the original testing set, containing 4 conversation sessions. The detailed composition of 16 sessions is reported in Appendix D.4.

Baselines. The baseline speech recognition systems are deep neural network acoustic models on clients trained using typical FL algorithms. Specifically, the deep neural networks are trained with an SRU-HMM-based acoustic model scheme (Huang et al. 2020, 2021), where SRU is a popular and efficient recurrent neural network for acoustic modeling.

Table 2: The WERs of different methods on CHiME-5 under non-i.i.d FL settings. The WERs are reported with the mean and standard deviation of three trials.

Methods	WER ↓
FedAvg (McMahan et al. 2017a)	68.86±0.11
FedProx (Li et al. 2020)	66.72±0.28
FedPA (Al-Shedivat et al. 2020)	69.92±0.22
FedNP (ours)	**64.30±0.21**

In the implementation, SRU is used as a backbone of the acoustic model, which contains 12 stacked layers with 1280 hidden nodes. We compare FedNP with three baselines: (1) `FedAvg` (McMahan et al. 2017a), (2) `FedProx` (Li et al. 2020), and (3) `FedPA` (Al-Shedivat et al. 2020). `SCAFFOLD` and `FedLA` are unstable in our preliminary experiments due to the large models and complex tasks; therefore, we do not include them as baselines. All baselines and our FedNP adopt identical federated configurations. More implementation details can be found in Appendix D.4.

Results. We train the models using three different random seeds and report the mean and standard deviation (STD) of the test word error rates (WERs) in Table 2. Our FedNP achieves around 4.56 % absolute WER reduction compared to the FedAvg baseline, a large margin in ASR, suggesting that our FedNP is capable of improving the performance under realistic non-i.i.d speech datasets. Notably, FedPA performs slightly worse than FedAvg due to its instability. Even compared to the state-of-the-art non-i.i.d FL algorithm FedProx, our FedNP achieves around 2.42 % absolute WER reduction, which demonstrates that compared to naively typical methods that naively push local models closer to the global model averaged in the last turn, FedNP can estimate a more accurate global model distribution, leading to better performance for non-i.i.d data.

6 Conclusion

We present a novel sampling-free and end-to-end differentiable Bayesian federated learning framework, dubbed FedNP, for non-i.i.d cross-silo data by enhancing the local model with an auxiliary task that explicitly estimates the global data distribution. We successfully alleviate the critical challenge in estimating the global data distribution on partitioned non-i.i.d data with an expectation-propagation-inspired probabilistic neural network. More specifically, we tackle defects of existing algorithms for deep-learning-based EP inference and derive a closed-form solution for estimating the global data distribution, leading to a more efficient solution for global data modeling. Experiments on toy non-i.i.d data and real-world extremely non-i.i.d image and speech data partitions demonstrate that our framework effectively alleviates the performance deterioration caused by non-i.i.d. data compared to other representative baselines.

So far, FedNP has not been evaluated on the unsupervised learning setting, nor on datasets with other modalities such as text. Another current limitation of FedNP is that though FedNP helps facilitate the joint modeling across data centers as an FL algorithm, such cooperation might also bring the risk of personal privacy leakage. Therefore, it is also necessary to explore more privacy-protection methods that can further improve the data privacy and security protection of FL algorithms, including FedNP.

Acknowledgement

The authors thank the reviewers/SPC/AC for the constructive comments to improve the paper. HH and YW are partially supported by a grant A-0008153-00-00 from Ministry of Education of Singapore. HW is partially supported by NSF Grant IIS-2127918 and an Amazon Faculty Research Award. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors.

References

Acar, D. A. E.; Zhao, Y.; Navarro, R. M.; Mattina, M.; Whatmough, P. N.; and Saligrama, V. 2021. Federated learning based on dynamic regularization. *arXiv preprint arXiv:2111.04263*.

Airey, S.; and Gales, M. 2003. Product of Gaussians for speech recognition.

Al-Shedivat, M.; Gillenwater, J.; Xing, E.; and Rostamizadeh, A. 2020. Federated learning via posterior averaging: A new perspective and practical algorithms. *arXiv preprint arXiv:2010.05273*.

Barker, J.; Watanabe, S.; Vincent, E.; and Trmal, J. 2018. The fifth'CHiME'speech separation and recognition challenge: dataset, task and baselines. *arXiv preprint arXiv:1803.10609*.

Barthelmé, S.; and Chopin, N. 2014. Expectation propagation for likelihood-free inference. *Journal of the American Statistical Association*, 109(505): 315–333.

Biem, A.; Katagiri, S.; McDermott, E.; and Juang, B.-H. 2001. An application of discriminative feature extraction to filter-bank-based speech recognition. *IEEE Transactions on Speech and Audio Processing*, 9(2): 96–110.

Bui, T. D.; Nguyen, C. V.; Swaroop, S.; and Turner, R. E. 2018. Partitioned variational inference: A unified framework encompassing federated and continual learning. *arXiv preprint arXiv:1811.11206*.

Chen, H.-Y.; and Chao, W.-L. 2020. Fedbe: Making bayesian model ensemble applicable to federated learning. *arXiv preprint arXiv:2009.01974*.

Gao, L.; Fu, H.; Li, L.; Chen, Y.; Xu, M.; and Xu, C.-Z. 2022. FedDC: Federated Learning with Non-IID Data via Local Drift Decoupling and Correction. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Gómez, P.; Toftevaag, H. H.; and Meoni, G. 2021. torchquad: Numerical Integration in Arbitrary Dimensions with PyTorch. *Journal of Open Source Software*, 6(64): 3439.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Heess, N.; Tarlow, D.; and Winn, J. 2013. Learning to pass expectation propagation messages. *Advances in Neural Information Processing Systems*, 26: 3219–3227.

Huang, H.; Liu, H.; Wang, H.; Xiao, C.; and Wang, Y. 2021. STRODE: Stochastic Boundary Ordinary Differential Equation. In *International Conference on Machine Learning*, 4435–4445. PMLR.

Huang, H.; Xue, F.; Wang, H.; and Wang, Y. 2020. Deep graph random process for relational-thinking-based speech recognition. In *International Conference on Machine Learning*, 4531–4541. PMLR.

ichi Amari, S.; and Nagaoka, H. 2000. Methods of Information Geometry, volume 191 of Translations of Mathematical Monographs.

Ingersoll, J. E. 1987. *Theory of financial decision making*, volume 3. Rowman & Littlefield.

Jordan, M. I.; Ghahramani, Z.; Jaakkola, T. S.; and Saul, L. K. 1999. An introduction to variational methods for graphical models. *Machine learning*, 37(2): 183–233.

Jylänki, P.; Nummenmaa, A.; and Vehtari, A. 2014. Expectation Propagation for Neural Networks with Sparsity-Promoting Priors. *Journal of Machine Learning Research*, 15(54): 1849–1901.

Karimireddy, S. P.; Kale, S.; Mohri, M.; Reddi, S.; Stich, S.; and Suresh, A. T. 2020. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, 5132–5143. PMLR.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krizhevsky, A.; Nair, V.; and Hinton, G. 1995. CIFAR-100 (Canadian Institute for Advanced Research). *Canadian Institute for Advanced Research*.

Le, Y.; and Yang, X. 2015. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7): 3.

Li, Q.; Diao, Y.; Chen, Q.; and He, B. 2021. Federated learning on non-iid data silos: An experimental study. *arXiv preprint arXiv:2102.02079*.

Li, Q.; He, B.; and Song, D. 2021. Model-Contrastive Federated Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10713–10722.

Li, T.; Sahu, A. K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; and Smith, V. 2020. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2: 429–450.

Li, X.; Huang, K.; Yang, W.; Wang, S.; and Zhang, Z. 2019. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*.

Liu, L.; Zheng, F.; Chen, H.; Qi, G.-J.; Huang, H.; and Shao, L. 2021. A Bayesian Federated Learning Framework with Online Laplace Approximation. *arXiv preprint arXiv:2102.01936*.

Liu, W.; Wen, Y.; Yu, Z.; and Yang, M. 2016. Large-margin softmax loss for convolutional neural networks. In *ICML*, volume 2, 7.

MacKay, D. J. 1992. A practical Bayesian framework for backpropagation networks. *Neural computation*, 4(3): 448–472.

McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017a. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, 1273–1282. PMLR.

McMahan, H. B.; Ramage, D.; Talwar, K.; and Zhang, L. 2017b. Learning differentially private recurrent language models. *arXiv preprint arXiv:1710.06963*.

Minka, T. P. 2013. Expectation propagation for approximate Bayesian inference. *arXiv preprint arXiv:1301.2294*.

Povey, D.; Ghoshal, A.; Boulianne, G.; Burget, L.; Glembek, O.; Goel, N.; Hannemann, M.; Motlicek, P.; Qian, Y.; Schwarz, P.; et al. 2011. The Kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, CONF. IEEE Signal Processing Society.

Soudry, D.; Hubara, I.; and Meir, R. 2014. Expectation backpropagation: Parameter-free training of multilayer neural networks with continuous or discrete weights. In *NIPS*, volume 1, 2.

Tan, C.; Jiang, D.; Peng, J.; Wu, X.; Xu, Q.; and Yang, Q. 2021. A de novo divide-and-merge paradigm for acoustic model optimization in automatic speech recognition. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 3709–3715.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11).

Voss, W. G. 2016. European union data privacy law reform: General data protection regulation, privacy shield, and the right to delisting. *The Business Lawyer*, 72(1): 221–234.

Wang, H.; Shi, X.; and Yeung, D.-Y. 2016. Natural-parameter networks: A class of probabilistic neural networks. *arXiv preprint arXiv:1611.00448*.

Wang, H.; and Yeung, D.-Y. 2016. Towards Bayesian deep learning: A framework and some existing methods. *TDKE*, 28(12): 3395–3408.

Wang, H.; and Yeung, D.-Y. 2020. A Survey on Bayesian Deep Learning. *CSUR*, 53(5): 1–37.

Wang, H.; Yurochkin, M.; Sun, Y.; Papailiopoulos, D.; and Khazaeni, Y. 2020. Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440*.

Wang, S.; and Manning, C. 2013. Fast dropout training. In *international conference on machine learning*, 118–126. PMLR.

Xie, C.; Koyejo, S.; and Gupta, I. 2019. Asynchronous federated optimization. *arXiv preprint arXiv:1903.03934*.

Yang, Q.; Liu, Y.; Chen, T.; and Tong, Y. 2019. Federated Machine Learning: Concept and Applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2): 12.

Yu, F.; Rawat, A. S.; Menon, A.; and Kumar, S. 2020. Federated learning with only positive labels. In *International Conference on Machine Learning*, 10946–10956. PMLR.

Zhao, J.; Liu, X.; He, S.; and Sun, S. 2020. Probabilistic inference of Bayesian neural networks with generalized expectation propagation. *Neurocomputing*, 412: 392–398.

Zhao, Y.; Li, M.; Lai, L.; Suda, N.; Civin, D.; and Chandra, V. 2018. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*.

Zhu, H.; Xu, J.; Liu, S.; and Jin, Y. 2021. Federated Learning on Non-IID Data: A Survey. 1–29.

A Proofs

A.1 Supplementary Lemma for Proofs

We formally state the Stein's Lemma here, following the version in (Ingersoll 1987).

Lemma 1. *Let $C \sim \mathcal{N}(C_{m,-k}, C_{s,-k})$, and let g be a differentiable function satisfying $E[g'(X)] < \infty$. Then*

$$E[g(C)(C - C_{m,-k})] = C_{s,-k} E[g'(C)] \tag{21}$$

A.2 Proof of Theorem 1

Theorem 1. *Suppose we are given a data partition $\mathbf{X}_k$ located at the k-th party during FL. Assume data partitions $\{\mathbf{X}_k\}_{k=1}^K$ are conditionally independent given a latent variable $\mathbf{z}$. Let $f : \mathcal{R}^{|\mathbf{X}_k|} \to \mathcal{R}^1$ be a neural network taking as input $\mathbf{X}_k$. Let $C = f(\mathbf{X}_k)\mathbf{z}$ and $C \sim \mathcal{N}(C_{m,-k}, C_{s,-k})$. Let $q_{-k}(\mathbf{z})$ and $h_k(\mathbf{z})$ be the cavity distribution (defined in Eq. (6)) and the hybrid distribution (defined in Eq. (7)), respectively. We further define $q(\mathbf{z}) = \mathcal{N}(\mathbf{z}_m, \mathbf{z}_s)$ as in Eq. (5). There exists a function $\phi : \mathcal{R}^{|\mathbf{z}|} \times \mathcal{R}^{|\mathbf{X}_k|} \to \mathcal{R}^1$, such that the update rules of $\mathbf{z}_m$ and $\mathbf{z}_s$ can be written in closed form as:*

$$\mathbf{z}_m = S_1, \tag{22}$$

$$\mathbf{z}_s = S_2 - S_1^2, \tag{23}$$

where

$$S_1 = {\left[(C_{m,-k}+C_{s,-k})E_C(\sigma(C)) - C_{s,-k}E_C(\sigma^2(C))\right]}/{S_0 f(\mathbf{X}_k)}, \tag{24}$$

$$S_2 = {\left[(C_{m,-k}+2C_{s,-k})E_C(\sigma^2(C)) - 2C_{s,-k}E_C(\sigma^3(C))\right]}/{S_0 f^2(\mathbf{X}_k)}, \tag{25}$$

$$S_0 = E_C(\sigma(C)). \tag{26}$$

Proof. With $C \sim \mathcal{N}(C_{m,-k}, C_{s,-k})$ and Theorem 2, we have that the first three moments of $\sigma(C)$ can be expressed in closed form, e.g.:

$$E_C(\sigma(C)) \approx \sigma\left(\frac{(C_{m,-k})}{\sqrt{1+\zeta^2 C_{s,-k}}}\right); \tag{27}$$

$$E(\sigma^2(C)) \approx \sigma\left(\frac{a\,(C_{m,-k}+b)}{\sqrt{1+\zeta^2 a^2 C_{s,-k}}}\right), \tag{28}$$

where $\zeta^2 = \frac{\pi}{8}$, $a = 4 - 2\sqrt{2}$, and $b = \ln(\sqrt{2}+1)$;

$$E(\sigma_C^3(C)) \approx \sigma\left(\frac{a\,(C_{m,-k}+b)}{\sqrt{1+\zeta^2 a^2 C_{s,-k}}}\right), \tag{29}$$

where $\zeta^2 = \frac{\pi}{8}$, $a = 6\left(1 - \frac{1}{\sqrt[3]{2}}\right)$, and $b = \ln(\sqrt[3]{2} - 1)$.

Let $q(\mathbf{z}|\mathbf{X}_k) = \phi(\mathbf{z}, \mathbf{X}_k) = \sigma(f(\mathbf{X}_k)\mathbf{z})$, where σ denotes Sigmoid activation function. Let the normalizer $S_0 = \int h_k(\mathbf{z})\mathrm{d}\mathbf{z}$. With Eq. (27), we have:

$$\begin{aligned} S_0 &= \int \sigma\left(f(\mathbf{X}_k)\mathbf{z}\right) q_{-k}(\mathbf{z})\mathrm{d}\mathbf{z} \\ &= E_C(\sigma(C)) \\ &\approx \sigma\left(\frac{(C_{m,-k})}{\sqrt{1+\zeta^2 C_{s,-k}}}\right) \end{aligned} \tag{30}$$

Let the first moment of $\mathbf{z}$, $S_1 = \int \mathbf{z} h_k(\mathbf{z})\mathrm{d}\mathbf{z} / \int h_k(\mathbf{z})\mathrm{d}\mathbf{z}$. We have:

$$S_1 = \frac{\int f(\mathbf{X}_k)\mathbf{z}\sigma\left(f(\mathbf{X}_k)\mathbf{z}\right) q_{-k}(\mathbf{z})\mathrm{d}\mathbf{z}}{f(\mathbf{X}_k)S_0} = \frac{E_C\left(C\cdot\sigma(C)\right)}{f(\mathbf{X}_k)S_0} \tag{31}$$

We then take Lemma 1 to compute $E_C(C\cdot\sigma(C))$, we have

$$E_C\left(\sigma(C)\cdot(C - C_{m,-k})\right) = C_{s,-k}E_C\left(\sigma'(C)\right) \tag{32}$$

Then we have:

$$\begin{aligned} E_C(\sigma(C)\cdot C) =& \left(C_{s,-k} + C_{m,-k}\right)E_C(\sigma(C)) \\ & - C_{s,-k}E_C(\sigma^2(C)), \end{aligned} \tag{33}$$

With Eq. (31) and Eq. (33), we therefore have:

$$S_1 = \frac{\left[\left(C_{m,-k}+C_{s,-k}\right)E(\sigma(C)) - C_{s,-k}E(\sigma^2(C))\right]}{S_0 f(\mathbf{X}_k)}. \tag{34}$$

Combining Eq. (27), Eq. (28), Eq. (30) and Eq. (34) , we have the first moment S_1 being expressed in closed form.

Let the second moment of $\mathbf{z}$, $S_2 = \int \mathbf{z}^2 h_k(\mathbf{z})\mathrm{d}\mathbf{z} / \int h_k(\mathbf{z})\mathrm{d}\mathbf{z}$. We have:

$$S_2 = \frac{\int f^2(\mathbf{X}_k)\mathbf{z}^2\sigma\left(f(\mathbf{X}_k)\mathbf{z}\right) q_{-k}(\mathbf{z})\mathrm{d}\mathbf{z}}{f^2(\mathbf{X}_k)S_0} = \frac{E_C\left(C^2\cdot\sigma(C)\right)}{f^2(\mathbf{X}_k)S_0} \tag{35}$$

With Lemma 1, we have

$$\begin{aligned} & E_C\left(\sigma(C)\cdot C\cdot(C - C_{m,-k})\right) = \\ & C_{s,-k}E_C\left(\sigma(C) + \left(\sigma(C) - \sigma^2(C)\right)\cdot C\right) \end{aligned} \tag{36}$$

Then we have:

$$\begin{aligned} E_C\left(C^2\cdot\sigma(C)\right) =& C_{s,-k}E\left(\sigma(C)\right) \\ & + \left(C_{s,-k} + C_{m,-k}\right)E_C(C\cdot\sigma(C)) \\ & - C_{s,-k}E_C(\sigma^2(C)\cdot C) \end{aligned} \tag{37}$$

We further adopt Lemma 1 to derive $E_C(\sigma^2(C)\cdot C)$, then we have:

$$E_C\left(\sigma^2(C)(C - C_{m,-k})\right) = C_{s,-k}E_C\left(2\sigma^2(C) - 2\sigma^3(C)\right) \tag{38}$$

Then

$$\begin{aligned} E_C\left(\sigma^2(C)\cdot C\right) =& \left(C_{m,-k} + 2C_{s,-k}\right)E_C(\sigma^2(C)) \\ & - 2C_{s,-k}E_C(\sigma^3(C)) \end{aligned} \tag{39}$$

With Eq. (35), Eq. (37) and Eq. (39), we have:

$$S_2 = \frac{\left[(C_{m,-k} + 2C_{s,-k})\, E(\sigma^2(C)) - 2C_{s,-k}E(\sigma^3(C))\right]}{S_0 f^2(\mathbf{X}_k)}. \tag{40}$$

Combining Eq. (28)-(30) and Eq. (40), we have the second moment S_2 being expressed in closed form as well.

With first two moments of $\mathbf{z}$, S_1 and S_2, being expressed in closed form, we have the approximated posterior of global data distribution $q(\mathbf{z}) = \mathcal{N}(\mathbf{z}_m, \mathbf{z}_s)$, whose parameters $\mathbf{z}_m$ and $\mathbf{z}_s$ can be updated in close-form:

$$\mathbf{z}_m = S_1, \tag{41}$$

$$\mathbf{z}_s = S_2 - S_1^2, \tag{42}$$

□

A.3 Proof of Theorem 2

Theorem 2. *Suppose $C \sim \mathcal{N}(C_{m,-k}, C_{s,-k})$. Let $d \geqslant 1$ be a positive integer. There exist two real constants a and b, such that the first d moments can be expressed in closed form:*

$$E_C(\sigma^d(C)) \approx \sigma\left(\frac{a\,(C_{m,-k} + b)}{\sqrt{1 + \zeta^2 a^2 C_{s,-k}}}\right), \tag{43}$$

Proof. Let a, b be two real constants. Taking the probit funtion $\Phi(\zeta a(C + b))$ to approximate $\sigma^d(C)$ by matching their value and derivative at median of the probit function, we have:

$$\sigma^k(C) \approx \Phi(\zeta a(C + b))$$

where

$$\begin{aligned} a &= 2d(1 - 2^{-1/d}) \\ b &= \log(2^{1/d} - 1) \end{aligned} \tag{44}$$

Let $d = 1$, we have:

$$\sigma(C) \approx \Phi(\zeta C) \tag{45}$$

With Theorem 3 (Wang, Shi, and Yeung 2016) and Eq. (45), we have:

$$\begin{aligned} E_C(\sigma^d(C)) &= \int \sigma^d(C)\mathcal{N}(C_{m,-k}, C_{s,-k})\, \mathrm{d}C \\ &\approx \int \Phi(\zeta a(x + b)\mathcal{N}(C_{m,-k}, C_{s,-k})\, \mathrm{d}C \\ &= \Phi\left(\frac{\zeta a\,(C_{m,-k} + b)}{\sqrt{1 + \zeta^2 a^2 C_{s,-k}}}\right) \\ &\approx \sigma\left(\frac{a\,(C_{m,-k} + b)}{\sqrt{1 + \zeta^2 a^2 C_{s,-k}}}\right) \end{aligned} \tag{46}$$

Combining Eq. (44) and Eq. (46) concludes the proof. □

B The FedNP Algorithm

The FedNP algorithm (see Algorithm 1) runs on a federation consisting of a central server and K clients, where the k-th client has n_k training examples. Each client conducts T local updating iterations. Between every L updates, all clients send updated parameters to the server; the server aggregates the received model parameters with weighted averaging (McMahan et al. 2017a) and computes the updated cavities parameters with Product-of-Gaussian (Airey and Gales 2003).

Practically, to further restrict the complexity of the approximate global model distribution, we extend the objective function (19) with an additional loss, i.e., the Kullback–Leibler divergence between the approximate global data distribution and a Gaussian prior p($\mathbf{z}$):

$$\tilde{\ell} = KL\left[p(\boldsymbol{\theta}_k|\mathbf{z}) \| p(\hat{\boldsymbol{\theta}}_k)\right] + KL\left[q(\mathbf{z}) \| p(\mathbf{z})\right] \tag{47}$$

C Related Work: EP with Neural Networks

To efficiently model the global data distribution from partitioned data, we follow the spirit of Expectation Propagation (EP). EP is one of the most popular Bayesian inference methods (Minka 2013), which approximates the posterior with approximate factors that are iteratively updated via moment matching. However, the moment matching can be intractable if the likelihood term has a complex form, which results in the intractability of the moments of the hybrid distribution. An intuitive solution is to approximate the likelihood term by numerical quadrature (Jylänki, Nummenmaa, and Vehtari 2014; Soudry, Hubara, and Meir 2014), but it fails to scale to large datasets or complex neural networks. Along a different line of research, (Heess, Tarlow, and Winn 2013) achieves this by incorporating neural networks to map EP message inputs to EP message outputs.Zhao et al. (2020) alleviate this problem by considering the EP's KL divergence between the target distribution and a mixture of exponential family approximate factors as the objective function. The closest work to ours is (Bui et al. 2018), which adopts variational inference (VI) to simulate EP on federated settings with synthetic data, but their EP update requires performing gradient descent with a deliberately designed KL loss. Furthermore, their method still relies on the likelihood term, which is not applicable to our federated setting. In contrast, we reformulate EP to remove the dependence on the intractable likelihood term and propose a closed-form solution for moment matching without requiring numerical approximation and more efficient than estimating through gradient descent.

To confirm the significance of the contribution in terms of EP with neural networks, we give a comparison of related works with our FedNP in Table 3.

D Experiment Details

D.1 Experimental Environments

We run all experiments on a machine with 314 GB RAM, an Intel Xeon PHI 7290 CPU Processor, and two Tesla

Table 3: The comparison of supported features of representative methods

Representative method	Sampling-free EP update	Closed-form EP update approximation	Numerical approximation/quadrature free
Jylänki, Nummenmaa, and Vehtari (2014)	✓	✓	✗
Bui et al. (2018)	✗	✗	✗
Heess, Tarlow, and Winn (2013)	✗	✗	✓
Zhao et al. (2020)	✗	✗	✗
Ours	✓	✓	✓

V100 GPUs. The operating system is CentOS 7. For detailed versions of software environments, please refer to the `README.md` files in corresponding code projects. Note that the datasets used in our experiments are public, and codes can be found in https://anonymous.4open.science/r/FedNP-Neurips/ or the supplementary material.

D.2 Details of Toy Experiment

Experimental Setup We use the RMSprop optimizer with a learning rate of 0.01 for all approaches. The batch size is set to 10. The number of local epochs is set to 1. The total number of communication rounds is 10.

Quantitative Results Table 4 shows the Mean Squared Error (MSE) of FedNP and baselines under the above setting. The MSE is evaluated on global training data. The local training of the baselines leads the model to fit their data points perfectly, failing to generalize well to the unseen global data, thus having a higher value of MSE. In contrast, our FedNP is more robust since we correct the local training and avoids performance deterioration.

Table 4: MSE of FedNP and the baselines. We run three trials and report the means and standard deviations accordingly.

Method	MSE $\downarrow$
FedAvg (McMahan et al. 2017a)	116.48 ± 2.54
FedProx (Li et al. 2020)	117.11 ± 2.96
SCAFFOLD (Karimireddy et al. 2020)	100.48 ± 3.24
FedPA (Al-Shedivat et al. 2020)	97.85 ± 3.44
FedNP (ours)	**72.42** $\pm$ 2.39

Case Study of FedPA and FedNP Figure 4 shows the comparison of FedPA and FedNP. The performance of FedPA is slightly better than FedAvg, but it is also biased towards local data and failed to capture the global information, as opposed to our FedNP.

D.3 FedNP for Image Classification on Extremely Non-IID Image Datasets

Dataset Preparation. CIFAR-100 (Krizhevsky, Nair, and Hinton 1995) (60,000 images with 100 classes) and Tiny-Imagenet (Le and Yang 2015) (100,000 images with 200 classes) datasets are with the MIT-License, so they are publicly available.

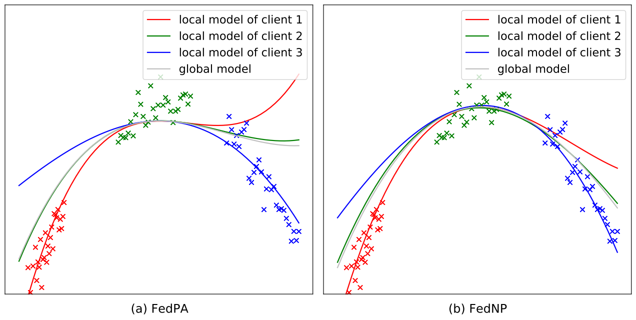


Figure 4: Toy example on polynomial curve fitting task. Data points are denoted by '$\times$' and models are denoted by '—'. (a) The local models of three clients and the global model trained by FedAvg. (b) The local models of three clients and the global model trained by FedNP.

Similar to (Li et al. 2021), we sample $p_k \sim Dir_N(\beta)$ and allocate a $p_{k,j}$ proportion of the images of class k to client j, where $Dir(\beta)$ is the Dirichlet distribution with a concentration parameter β (0.5 by default). With the above partitioning strategy, the data distribution of in each partition is extremely non-i.i.d, i.e. each client may have relatively few data samples in some classes.

Model Architectures. We use ResNet18 (He et al. 2016) as the model architecture for all approaches. For FedNP, we implement the auxiliary neural network f mentioned in Section 4.4 by stacking a linear layer after the ResNet18 backbone, which takes the input X_k, and the output size is 10. We then apply mean pooling to the output vector to achieve $f(\mathbf{X}_k)$ with dimension 1. The non-linear function ϕ (Eq. 8) is implemented as $\phi = \sigma(f(\mathbf{X}_k)\mathbf{z})$, where σ is the sigmoid function.

The dimension of $\mathbf{z}$ is set to 10. We then use an NPN with one hidden layer to obtain $p(z|\mathbf{X}_k)$. For simplicity, we only apply our FedNP to the parameter of the classifier instead of the whole model, so the dimension of $z_{m,k}$ and $z_{s,k}$ is equal to that of the classifier (i.e., the last MLP layer of ResNet18). For MOON (Li, He, and Song 2021), we use the input of the last MLP layer as the feature of the image.

Training Protocol. The number of clients and the dimension of $\mathbf{z}$ are set to 10. We use the SGD optimizer with a learning rate of 0.01 for all approaches. The batch size is set to 128. The number of local epochs is set to 10. The total number of communication rounds is 100. We tune λ from $\{0.001, 0.01, 0.1, 1\}$ and report the best result. The best λ

Algorithm 1: The FedNP algorithm that is conducted on the server and K clients. Notably, the operations below on vectors or tensors are element-wise.

```
1: Input: $K$ clients, and $n_k$ examples at the $k$-th client; $L$ be the local training iterations; $T$ be the total number of training iterations.
2: Server initializes model parameters $\boldsymbol{\theta}^{(0)}$, cavities $\left\{\left(\mathbf{z}_{m,-k}^{(0)}, \mathbf{z}_{s,-k}^{(0)}\right)\right\}_{k=1\dots K}$ (initialized as standard normal distribution) and broadcasts to all clients.
3: for $t = 0, ..., T-1$ do
4:   for Client $k = 1, ..., K$ (parallelly) do
5:     Update the parameters of approximated global data distribution $(\mathbf{z}_m^{(t)}, \mathbf{z}_s^{(t)})$ according to Eq. (9) and Eq. (10).
6:     Optimize the local model parameters $\boldsymbol{\theta}_k^{(t)}$ with the extended objective 19.
7:     if $(t > 0$ and $(t-1) \mod L == 0)$ or $(t == T-1)$ then
8:       Compute the approximated posterior factor $q_k(\mathbf{z})$ with the approximation:
           $$\mathbf{z}_{s,k}^{(t)} = \left(\mathbf{z}_s^{(t)^{-1}} - \mathbf{z}_{s,-k}^{(t)^{-1}}\right)^{-1},$$
           $$\mathbf{z}_{m,k}^{(t)} = \mathbf{z}_{s,k}^{(t)} \left[\frac{\mathbf{z}_m^{(t)}}{\mathbf{z}_s^{(t)}} - \frac{\mathbf{z}_{m,-k}^{(t)}}{\mathbf{z}_{s,-k}^{(t))}}\right].$$
9:       Upload $\left(\mathbf{z}_{m,k}^{(t)}, \mathbf{z}_{s,k}^{(t)}\right)$ and $\boldsymbol{\theta}_k^{(t)}$ to server.
10:      Server updates the global model parameters by weighted averaging:
           $$\boldsymbol{\theta}_{avg}^{(t+1)} = \sum_{k=1}^{K} \frac{1}{n_k} \boldsymbol{\theta}_k^{(t)}.$$
11:      Server updates the approximated cavities for each client $k$ with the mixture as Product-of-Gaussian (Airey and Gales 2003)
           $$\mathbf{z}_{s,-k}^{(t+1)} = \left(\sum_{j\neq k}^{K} \left(\mathbf{z}_{s,j}^{(t)}\right)^{-1}\right)^{-1},$$
           $$\mathbf{z}_{m,-k}^{(t+1)} = \mathbf{z}_{s,-k}^{(t+1)} \sum_{j\neq k}^{K} \frac{\mathbf{z}_{m,j}^{(t)}}{\mathbf{z}_{s,j}^{(t)}},$$
12:      The server broadcasts $\boldsymbol{\theta}_{avg}^{(t+1)}$ and $\left(\mathbf{z}_{s,-k}^{(t+1)}, \mathbf{z}_{m,-k}^{(t+1)}\right)$ to the $k$-th client.
13:    end if
14:  end for
15: end for
```

Table 5: The top-1 accuracy of FedNP and the other baselines on CIFAR-100 dataset with varying numbers of clients. We run three trials and report the mean and standard deviation.

Methods	50 clients	100 clients
FedAvg (McMahan et al. 2017a)	$61.32\% \pm 0.3\%$	$50.78\% \pm 0.3\%$
FedProx (Li et al. 2020)	$60.65\% \pm 0.2\%$	$48.90\% \pm 0.3\%$
MOON (Li, He, and Song 2021)	$62.13\% \pm 0.3\%$	$52.88\% \pm 0.3\%$
SCAFFOLD (Karimireddy et al. 2020)	$50.91\% \pm 0.3\%$	$46.28\% \pm 0.2\%$
FedDyn (Acar et al. 2021)	$61.32\% \pm 0.3\%$	$50.30\% \pm 0.2\%$
FedDC (Gao et al. 2022)	$60.93\% \pm 0.3\%$	$50.45\% \pm 0.2\%$
FedPA (Al-Shedivat et al. 2020)	$61.77\% \pm 0.3\%$	$51.51\% \pm 0.2\%$
FedLA (Liu et al. 2021)	$57.39\% \pm 0.3\%$	$47.34\% \pm 0.2\%$
FedNP (ours)	**63.74%** $\pm 0.3\%$	**53.37%** $\pm 0.3\%$

of FedNP is 0.01. Note that FedProx and MOON also have a hyper-parameter λ to control the weight of its proximal term. We tune λ from $\{0.001, 0.01, 0.1, 1, 10\}$, the best λ is 0.01 for FedProx and 1 for MOON.

Results of 50-, and 100-Client Settings on CIFAR-100. We conduct experiments on the CIFAR-100 dataset with varying numbers of clients. As the number of clients increases to 100, the performances of all methods drop significantly due to the sparsity of local training data. Our method has consistent advantages over baselines. More specifically, when the number of clients is 100, the local data is highly label-skew, i.e., some of the classes are not seen in the local client. Therefore, the local models severely collapse, resulting in unstable aggregation. In this case, the methods with explicit local model regularization, MOON and FedNP perform better, and the performance of the methods with dynamic regularizers is still close to FedAvg. Moreover, FedNP outperforms MOON as its local models are regularized by the estimated global model distribution, which might be more accurate than MOON's regularizer, i.e., a contrastive loss with the previous global model. Furthermore, to show the computational efficiency of our proposed closed-form update. On a single NVIDIA GeForce RTX 2080 GPU node, we compare FedNP with other baselines in terms of efficiency using CIFAR100 and ResNet18. The time cost for each client per communication round is 15s for FedAvg and FedProx, 19s for FedNP, and 25s for MOON, demonstrating the effectiveness of our proposed FedNP.

Analysis on Predicted Model Distributions by FedNP. We visualize the predicted model distribution by FedNP in Figure 5. For the convenience of visualization, we focus on parameters of the classifier of the model, denoted by a matrix $\mathbf{C} \in \mathcal{R}^{d\times c}$, where c equals the number of classes and d is the dimension of the features. More specifically, the matrix

$\mathbf{C}$ comprises c vectors with the dimension of d, and the j^{th} vector can be regarded as the centroid of the j^{th} class in the literature of metric learning (Liu et al. 2016). Figure 5 visualizes the predicted distribution of parameters (shown within three standard deviations of the mean), where the contour lines represent probability densities of predicted model distributions and the deeper color indicates the higher probability. As Figure 5 shows, the predicted model distribution coincides with the ground-truth data distribution (the solid points), which also verifies the effectiveness of our FedNP.

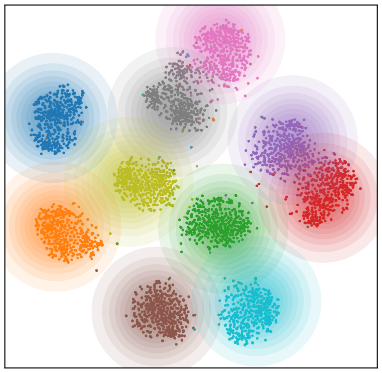


Figure 5: T-SNE visualizations of predicted distribution estimated by FedNP on CIFAR-100.

Experiments on Computational Efficiency and Approximation Accuracy of the Proposed Closed-form Update of FedNP. To evaluate the computational efficiency and approximation accuracy of the proposed closed-form update of FedNP, one may ask whether we can compare FedNP with existing EP algorithms. However, most of the EP-based algorithms are not applicable to our problem setting due to the need for $\boldsymbol{\theta}_k$, the local model parameter at the current mini-batch during training, and thus cannot directly compare with our FedNP. Therefore, we construct another variant of FedNP, dubbed Quad-FedNP, which adopt numerical-quadrature-based update (Jylänki, Nummenmaa, and Vehtari 2014) for calculating moments of the hybrid distribution in FedNP. Quad-FedNP requires one-dimensional numerical quadratures implemented using TorchQuad (Gómez, Toftevaag, and Meoni 2021), which provides a very efficient numerical integration approximation toolkit optimized for graphics processing units (GPUs). Notably, numerical-quadrature-based updates are usually adopted by numerical-quadrature-based EP (Jylänki, Nummenmaa, and Vehtari 2014), which is similar to MCMC-based EP (Barthelmé and Chopin 2014) in terms of computation of moments of the hybrid distribution.

We report the classification accuracy and the average running time per update of hybrid distribution for Quad-FedNP and our FedNP in Table 6. We can see that our FedNP runs much faster than Quad-FedNP with varied numbers of quadrature points. For Quad-FedNP, the image classification performance is improved by increasing the number of quadrature points. It's worth noting that the running time of Quad-FedNP does not grow linearly with quadrature points due to TorchQuad's GPU-based optimization. Generally, our FedNP runs 2.4 times faster than Quad-FedNP while achieving close evaluation scores (Acc.). The result demonstrates the efficacy and efficiency of our proposed closed-form update formulations.

Table 6: The accuracy and average running time per update on CIFAR-100 of Quad-FedNP with different numbers of quadrature points and our proposed FedNP.

Method	Acc.	Running Time
Quad-FedNP-100	62.92%	639ms
Quad-FedNP-1000	64.51%	650ms
Quad-FedNP-10000	65.07%	721ms
FedNP (ours)	65.03%	306ms

D.4 FedNP for Speech Recognition on Large-scale Realistic Non-IID Conversation Corpus

Detailed Configuration of CHiME-5. CHiME-5 is a large-scale corpus of real-world multi-speaker conversational speech in home environments. We hold a ChiME-5 non-commercial usage license [1], which is issued by LPC. The training dataset, development dataset, and test dataset include about 40 hours, 4 hours, and 5 hours of conversational speech. Table 7 shows the splits of training, development, and evaluation sets. In each session, there are four speakers with around 130-180 minutes of conversation records. In the non-i.i.d setting, we treat each session as the cross-silo data in each client, creating a natural non-i.i.d dataset. For example, the speakers in clients 1, 2, 7, and 8 are all male, and sessions are recorded in different environments.

Table 7: The data have been split into training, development, and evaluation set as follows.

Dataset	Sessions	Speakers	Hours	Utterances
Train	16	32	40:33'	79,980
Dev	2	8	4:27'	7,440
Eval	2	8	5:12'	11,028

Training Details. We adopt the same configuration with (Huang et al. 2020) to train all GMM-HMM. More specifically, we follow the GMM-HMMs ASR structure to train CHiME-5 using the adapted Kaldi s5b recipe (Povey et al. 2011) with single-channel audio data from GMM-HMM training.

The speech data is preprocessed as 40-dimensional Mel-filter bank coefficients (Biem et al. 2001), which are calculated every 10ms. Notably, only the audio data recorded by binaural microphones are employed to train and evaluate this experiment. Inputs of all models consist of the current frame and its 4 future contextual frames. The input sequence is chunked into a fixed length of 20. We performed

[1]https://chimechallenge.github.io/chime6/download.html

speaker-level mean and variance normalization on the inputs. The HMM states aligned by GMM-HMM are used to train the subsequent neural network modules. The SRU model is trained to fit the mapping from acoustic features to HMM states. For FedNP, we feed SRU hidden state outputs into a two-layer neural network, whose output size is the same as the dimension of $\mathbf{z}$, to utilize the feature extraction capability offered by the SRU fully. We then apply the mean pooling to the output vector to achieve $f(\mathbf{X}_k)$ with dimension 1. The non-linear function ϕ (Eq. 8) is implemented as $\phi = \sigma(f(\mathbf{X})\mathbf{z})$, where σ is the sigmoid function.

The evaluation is performed with a tri-gram language model trained from the transcription of CHiME-5. The models are optimized with the categorical cross-entropy loss using BPTT with a dropout rate of 0.1 between the recurrent layers. The batch size is set to 128. The optimizer is Adam (Kingma and Ba 2014) with a learning rate of 3×10^{-4}.

For FL, there are 12 turns of aggregation with 1 local training epoch between each turn. We tune λ from $\{0.0001, 0.001, 0.01, 0.1\}$ and report the best result. The best λ of FedNP is 0.001. For FedNP, the dimension of $\mathbf{z}$ is set as 16. For FedProx, which has a hyper-parameter μ to control the weight of its proximal term, we also tune it from $\{0.001, 0.01, 0.1, 1\}$, and the best λ is 0.01 for FedProx.