Personalizing Low-Rank Bayesian Neural Networks Via Federated Learning

Boning Zhang[1] **Dongzhu Liu**[1] **Osvaldo Simeone**[2] **Guanchu Wang**[3]
Dimitrios Pezaros[1] **Guangxu Zhu**[4]
[1] University of Glasgow [2] King's College London [3] Rice University [4] Shenzhen Research Institute of Big Data

Abstract

To support real-world decision-making, it is crucial for models to be well-calibrated, i.e., to assign reliable confidence estimates to their predictions. Uncertainty quantification is particularly important in personalized federated learning (PFL), as participating clients typically have small local datasets, making it difficult to unambiguously determine optimal model parameters. Bayesian PFL (BPFL) methods can potentially enhance calibration, but they often come with considerable computational and memory requirements due to the need to track the variances of all the individual model parameters. Furthermore, different clients may exhibit heterogeneous uncertainty levels owing to varying local dataset sizes and distributions. To address these challenges, we propose LR-BPFL, a novel BPFL method that learns a global deterministic model along with personalized low-rank Bayesian corrections. To tailor the local model to each client's inherent uncertainty level, LR-BPFL incorporates an adaptive rank selection mechanism. We evaluate LR-BPFL across a variety of datasets, demonstrating its advantages in terms of calibration, accuracy, as well as computational and memory requirements. The code is available at `https://github.com/Bernie0115/LR-BPFL/`.

1 INTRODUCTION

Federated Learning (FL) has emerged as a powerful distributed learning framework, enabling multiple clients to collaboratively train a global model without sharing their local datasets. FL has demonstrated success across various domains, including healthcare (Xu et al., 2021), finance (C. Li et al., 2018), the Internet of Things (Nguyen et al., 2021), and computer vision (Oh et al., 2021). However, when data distributions vary significantly across clients, the traditional FL framework, which focuses on training a single global model, often struggles to generalize well to individual clients, leading to performance degradation (Zhao et al., 2018; Tan et al., 2022).

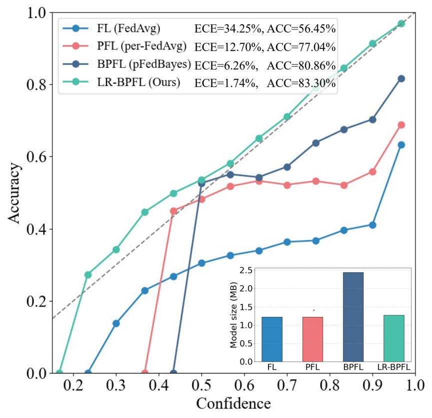

Figure 1: Average reliability diagrams across clients on the CIFAR-10 dataset for FedAvg (McMahan et al., 2017), a standard FL algorithm; Per-FedAvg (Fallah et al., 2020), a benchmark PFL scheme; pFedBayes (Zhang et al., 2022), a state-of-the-art BPFL algorithm; and the proposed LR-BPFL. (See Section 5.2 for details.)

Personalized Federated Learning (PFL) (T Dinh et al., 2020; T. Li et al., 2020) addresses this issue by allowing each client to optimize a personalized model while still leveraging shared information collected from other clients through FL. Despite the benefits of personalization in accounting for data heterogeneity, PFL methods are typically trained on smaller effective datasets, which often exacerbates the uncertainty in the model-parameter space. Moreover, standard PFL methods

Proceedings of the 28th International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258. Copyright 2025 by the author(s).

frequently output models that fail to correctly quantify local uncertainty, producing overconfident decisions (Fallah et al., 2020; Collins et al., 2021). This is illustrated in Fig. 1 for FedAvg (McMahan et al., 2017), a standard FL method, and for Per-FedAvg (Fallah et al., 2020), a benchmark PFL scheme.

Bayesian learning can potentially produce well-calibrated models by representing uncertainty in the model-parameter space. Recent efforts to integrate Bayesian learning into PFL have shown promise in enhancing model performance, offering improved predictive accuracy and calibration in heterogeneous data scenarios (Zhang et al., 2022; Achituve et al., 2021; Jeon et al., 2024). This is illustrated in Fig. 1 for pFedBayes (Zhang et al., 2022), a representative BPFL algorithm. However, these methods often rely on full-dimensional posterior distributions, leading to substantial computational and memory requirements. For example, in Zhang et al. (2022), the distribution over the model parameters is specified by weight-specific means and variances, thus doubling the number of parameters that need to be stored and processed.

Rank-1 Bayesian Neural Networks (BNNs) (Dusenberry et al., 2020) provide a parameter-efficient alternative to full BNNs by modeling the posterior distribution only over a rank-1 correction of deterministic model weights. However, incorporating rank-1 BNNs into FL is non-trivial, as it requires new strategies for model sharing and for personalization in order to address the heterogeneous distributions and sizes of the local datasets.

Thus motivated, in this paper, we propose LR-BPFL, a novel BPFL scheme that learns a global deterministic model along with personalized low-rank Bayesian corrections. To tailor the local model to each client's inherent uncertainty level, LR-BPFL incorporates an adaptive rank selection mechanism. As exemplified in Fig. 1, LR-BPFL is demonstrated via experiments to enhance accuracy and calibration as compared to the state-of-the-art methods, while also significantly reducing the model size, and, with it, computational and memory requirements.

The main contributions are summarized as follows:

• We introduce LR-BPFL, a BPFL protocol that reduces computational and memory requirements by jointly learning a shared deterministic model, while enabling clients to retain personalized Bayesian low-rank corrections.

• To address data heterogeneity, LR-BPFL incorporates a novel adaptive rank selection mechanism, which dynamically adjusts the complexity of the Bayesian corrections based on client-specific uncertainty levels.

• We empirically evaluate LR-BPFL on several real-world datasets, demonstrating significant improvements in calibration and predictive accuracy as compared to state-of-the-art FL, BFL, PFL and BPFL methods, particularly in scenarios with small and heterogeneous client datasets (see Fig. 1).

2 RELATED WORK

Personalized federated learning. PFL methods have been devised based on different principles, including local customization (T. Li et al., 2020; Hanzely & Richtárik, 2020), multi-task learning (Smith et al., 2017; Marfoq et al., 2021; Cai et al., 2023), meta-learning (Fallah et al., 2020; F. Chen et al., 2018; Lee et al., 2024), client clustering (Briggs et al., 2020; Sattler et al., 2020; Ma et al., 2023), and model mixing (Collins et al., 2021; Deng et al., 2020; Xu et al., 2023). While these methodologies enhance predictive accuracy on non-i.i.d. data, they offer suboptimal performance in terms of calibration (Achituve et al., 2021).

Bayesian Neural Network. BNNs combine the flexibility and expressive power of neural networks with the probabilistic properties of Bayesian inference, providing a powerful tool for uncertainty modeling (Neal, 2012). By accounting for uncertainty in the model-parameter space, BNNs enhance calibration performance (Khan & Rue, 2021; Knoblauch et al., 2019; Simeone, 2022). However, the high computational and memory complexity of BNNs has hindered their widespread adoption, particularly for resource-constrained clients. To address this limitation, one line of research advocates the use of sparsity-promoting priors to reduce the computational load by pruning neurons or weights (Molchanov et al., 2017; Bai et al., 2020). Another direction of research focuses on selectively applying Bayesian inference to critical parts of the model, while maintaining a deterministic, frequentist approach for other parts of the model (Daxberger et al., 2021). Some studies aim to reduce computational redundancies by addressing parameter space symmetries (Atzeni et al., 2023; Sen et al., 2024). A further promising strategy involves low-rank approximation. Notably, Dusenberry et al. (2020) proposes a rank-1 parameterization of BNNs.

Bayesian federated learning. While traditional FL methods typically focus on point estimates of model parameters, recent progress has applied Bayesian learning to FL. Bayesian FL (BFL) aims to derive a global posterior distribution that aggregates knowledge from all participating clients within the federated network. Algorithms, like FedPA (Al-Shedivat et al., 2020), QLSD (Vono et al., 2022) and DSVGD (Kassab

& Simeone, 2022), employ Bayesian posterior decomposition techniques to break down the global posterior into the product of local posteriors; while FedBE (H.-Y. Chen & Chao, 2020) constructs a Gaussian or Dirichlet distribution at the server for Bayesian model ensembling. FedNP (Wu et al., 2023) efficiently estimates the inaccessible ground-truth global data distribution using a probabilistic neural network, mitigating performance degradation induced by data heterogeneity.

Bayesian Personalized Federated Learning. To address statistical heterogeneity among clients, pFedGP (Achituve et al., 2021) learns a shared deep kernel function for all clients, while maintaining a personalized Gaussian process classifier at each client. Similarly, pFedBayes (Zhang et al., 2022) enables each client to maintain its own personalized BNN, while using the aggregated global posterior as the prior distribution. Furthermore, MetaVD (Jeon et al., 2024) employs a hypernetwork to predict client-specific posterior distributions of Gaussian noises.

3 PRELIMINARIES

This section reviews the standard FL setup (McMahan et al., 2017), BPFL via variational inference (Zhang et al., 2022; Jeon et al., 2024), as well as the rank-1 BNN method (Dusenberry et al., 2020) as required preliminaries.

3.1 Federated Learning

The typical FL setup involves a server coordinating with K clients to collaboratively train a global model without data sharing. Consider the training of a neural network parameterized by the weight vector $\mathbf{W}$. Each client k has a loss function $F_k(\cdot)$ used for training on its local dataset $\mathcal{D}_k = \{(\mathbf{x}_j, \mathbf{y}_j)\}_{j=1}^{|\mathcal{D}_k|}$. In conventional FL, the objective for all clients is to find a global model that minimizes the weighted sum

$$\min_{\mathbf{W}} \sum_{k=1}^{K} l_k F_k(\mathbf{W}, \mathcal{D}_k), \tag{1}$$

where the weight l_k is typically proportional to the size of the local dataset (i.e., $l_k = |\mathcal{D}_k|/|\mathcal{D}|$, with $\mathcal{D} = \{\mathcal{D}_k\}_{k=1}^K$). To address this problem, in each communication round t, a subset $\mathcal{S}^t$ of clients is selected to conduct local training, starting from the latest global model weights $\mathbf{W}^t$. At the end of the communication round t, the server aggregates the local models from the selected clients to update the global model, e.g., $\mathbf{W}^{t+1} \leftarrow \sum_{k\in\mathcal{S}^t} l_k \mathbf{W}_k^t$, where $\mathbf{W}_k^t$ is the updated parameter vector produced by client k.

3.2 Bayesian Personalized Federated Learning via Variational Inference

Variational inference methods approximate posterior distributions in the model-parameter space for a BNN, by using a family of variational distributions $q(\mathbf{W};\boldsymbol{\phi})$, which depends on a set of variational parameters $\boldsymbol{\phi}$ (see, e.g., (Bishop & Nasrabadi, 2006); (Simeone, 2022)). For each client k, variational inference aims at minimizing the negative evidence lower bound (ELBO), also known as free energy, as in

$$\begin{aligned}\min_{\boldsymbol{\phi}_k} \big\{ -\mathbb{E}_{q(\mathbf{W}_k;\boldsymbol{\phi}_k)} \left[\log p\left(\mathcal{D}_k \mid \mathbf{W}_k\right)\right] \\ + \mathrm{KL}\left(q\left(\mathbf{W}_k;\boldsymbol{\phi}_k\right) \| p\left(\mathbf{W}_k\right)\right) \big\},\end{aligned} \tag{2}$$

where $-\log p\left(\mathcal{D}_k \mid \mathbf{W}_k\right)$ represents the cross-entropy loss accrued by the local model parameter $\mathbf{W}_k$ on the local dataset $\mathcal{D}_k$; $\mathrm{KL}(q\|p)$ is the Kullback-Leibler divergence between distributions p and q; $\boldsymbol{\phi}_k$ is a vector of local variational parameters; and $p(\mathbf{W}_k)$ represents a prior distribution. The free energy (2) is thus a regularized version of the average cross-entropy loss (the first term in (2)), with regularization dictated by the prior distribution $p(\mathbf{W}_k)$ (the second term in (2)).

pFedBayes (Zhang et al., 2022) optimizes jointly the global prior distribution $p(\mathbf{W})$ and local posteriors $q(\mathbf{W}_k;\boldsymbol{\phi}_k)$ for all clients $k = 1, \cdots, K$ by setting $p(\mathbf{W}_k) = p(\mathbf{W})$ in (2). Assuming the prior $p(\mathbf{W})$ and the local posterior $q(\mathbf{W}_k;\boldsymbol{\phi}_k)$ to be Gaussian with diagonal covariance, the number of model parameters to be stored and processed is thus doubled as compared to non-Bayesian PFL methods.

MetaVD (Jeon et al., 2024) models the local posterior distribution $q(\mathbf{W}_k;\boldsymbol{\phi}_k)$ as a Gaussian with a common mean $\mathbf{W}$ and a client-specific diagonal covariance, where the variance terms are produced by a shared hypernetwork that takes a client-specific learnable embedding as input. Like pFedBayes, MetaVD doubles the number of parameters that need to be stored and processed.

3.3 Rank-1 Bayesian Neural Network

As discussed in Section 3.2, a BNN maintains a distribution $q(\mathbf{W};\boldsymbol{\phi})$ over the model parameters at the cost of increased computational and memory requirements. For single-model training, rank-1 BNNs present an efficient alternative in which the weight matrix $\mathbf{G} \in \mathbb{R}^{m\times n}$ for each layer is parameterized as $\mathbf{G} = \mathbf{W} \circ \mathbf{q}\mathbf{r}^\intercal$, where $\mathbf{W}$ represents a deterministic weight matrix, while $\mathbf{q}$ and $\mathbf{r}$ are m- and n-dimensional vectors, respectively, that are treated as random variables within a Bayesian framework. The symbol $\circ$ represents the element-wise product.

Training is based on the minimization of the free energy loss, i.e.,

$$\min_{\mathbf{W},\boldsymbol{\phi},\boldsymbol{\psi}} \Big\{ -\mathbb{E}_{q(\mathbf{q};\boldsymbol{\phi}),q(\mathbf{r};\boldsymbol{\psi})}\left[\log p(\mathcal{D} \mid \mathbf{W} \circ \mathbf{q}\mathbf{r}^{\mathsf{T}})\right] + \mathrm{KL}\big(q(\mathbf{q};\boldsymbol{\phi}) \,\|\, p(\mathbf{q})\big) + \mathrm{KL}\big(q(\mathbf{r};\boldsymbol{\psi}) \,\|\, p(\mathbf{r})\big)\Big\}, \quad (3)$$

where $\mathcal{D}$ is the dataset available at a given client; $q(\mathbf{q};\boldsymbol{\phi})$ and $q(\mathbf{r};\boldsymbol{\psi})$ are the variational distributions parameterized by vectors $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$; and $p(\mathbf{q})$ and $p(\mathbf{r})$ are prior distributions. Rank-1 BNNs support the BatchEnsemble method (Wen et al., 2020), in which the output of multiple models sampled from the variational posteriors can be evaluated simultaneously for a given mini-batch, enabling efficient parallel computations.

4 LOW-RANK BAYESIAN PERSONALIZED FEDERATED LEARNING

This section introduces LR-BPFL, a novel BPFL protocol that optimizes a shared model along with low-rank Bayesian multiplicative corrections for each client, whose rank is adapted to each client's local uncertainty levels. The overall protocol is detailed in Algorithm 1.

4.1 Low-Rank BNNs With Adaptive Rank Selection

LR-BPFL describes the $m \times n$ weight matrix $\mathbf{G}_k$ for each layer of client k's model as

$$\mathbf{G}_k = \mathbf{W} \circ \mathbf{M}_k, \quad (4)$$

where $\mathbf{W}$ represents the shared deterministic model, while $\mathbf{M}_k$ denotes the personalized Bayesian correction. To support personalization of uncertainty quantification, we model the correction $\mathbf{M}_k$, also referred to as the Bayesian mask, as

$$\mathbf{M}_k = \mathbf{Q}_k \boldsymbol{\Lambda}_k \mathbf{R}_k^{\mathsf{T}}, \quad (5)$$

where $\mathbf{Q}_k$ is an $m \times r_{\max}$ matrix, $\mathbf{R}_k$ is a $n \times r_{\max}$ matrix, and $\boldsymbol{\Lambda}_k$ is an $r_{\max} \times r_{\max}$ gating matrix. The parameter $r_{\max}$ represents the maximum rank for the mask $\mathbf{M}_k$, and is set to be much smaller than $\min(m, n)$. To select the optimal rank, the diagonal gating matrix $\boldsymbol{\Lambda}_k$ has binary diagonal elements $\lambda_{k,i} \in \{0, 1\}$ that control whether the corresponding i-th column of the dictionary matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$ are retained (when $\lambda_{k,i} = 1$) or pruned (when $\lambda_{k,i} = 0$). The rank of the personalized Bayesian mask $\mathbf{M}_k$ is then given by

$$r_k = \sum_{i=1}^{r_{\max}} \lambda_{k,i}. \quad (6)$$

Algorithm 1 LR-BPFL

```
Input: Number of global communication rounds $T$, number of clients $K$, fraction ratio $\tau$, learning rate $\eta$, size of the local dataset $N_k$, threshold $\bar{\lambda}$, number of local adaptation steps $T_L$
Output: $\mathbf{W}^\star, \{\boldsymbol{\phi}_k^\star, \boldsymbol{\psi}_k^\star, \boldsymbol{\lambda}_k^\star\}_{k=1}^K$
Initialize: Shared model $\mathbf{W}^0$, $\{\boldsymbol{\phi}_k, \boldsymbol{\psi}_k, \boldsymbol{\lambda}_k\}_{k=1}^K$
for round $t = 0, \dots, T$ do
  $\mathcal{S}^t \leftarrow$ Randomly sample $\tau K$ clients
  for client $k \in \mathcal{S}^t$ in parallel do
    Receive $\mathbf{W}^t$ from the server and reinitialize $\boldsymbol{\phi}_k, \boldsymbol{\psi}_k$
    $(\boldsymbol{\phi}_k,\boldsymbol{\psi}_k,\boldsymbol{\lambda}_k) \xleftarrow{T_L \text{steps}} (\boldsymbol{\phi}_k,\boldsymbol{\psi}_k,\boldsymbol{\lambda}_k) - \eta \nabla F_k(\boldsymbol{\phi}_k, \boldsymbol{\psi}_k, \boldsymbol{\lambda}_k)$
    Set $\mathbf{W}_k^t \leftarrow \mathbf{W}^t - \eta \nabla_{\mathbf{W}^t} F_k(\mathbf{W}^t, \boldsymbol{\phi}_k, \boldsymbol{\psi}_k, \boldsymbol{\lambda}_k)$
    Update $\boldsymbol{\lambda}_k$ using (10)
    Send $\mathbf{W}_k^t$ back to the server
  end for
  Server aggregates local models as $\mathbf{W}^{t+1} = \sum_{k \in \mathcal{S}^t} l_k \mathbf{W}_k^t$
end for
```

4.2 LR-BPFL

As detailed in Algorithm 1, LR-BPFL learns a shared deterministic model $\mathbf{W}$ by following a standard frequentist approach, while the distributions $q(\mathbf{Q}_k;\boldsymbol{\phi}_k)$ and $q(\mathbf{R}_k;\boldsymbol{\psi}_k)$ of the Bayesian mask are optimized via variational learning through their variational parameters $\boldsymbol{\phi}_k$ and $\boldsymbol{\psi}_k$ (see Section 3.3). Accordingly, the learning problem is formulated as

$$\min_{\mathbf{W},\{\boldsymbol{\phi}_k,\boldsymbol{\psi}_k,\boldsymbol{\lambda}_k\}_{k=1}^K} \sum_{k=1}^{K} l_k F_k\big(\mathbf{W}, \boldsymbol{\phi}_k, \boldsymbol{\psi}_k, \boldsymbol{\lambda}_k\big), \quad (7)$$

where $F_k(\cdot)$ denotes the regularized free energy loss in (3), i.e.,

$$F_k\big(\mathbf{W}, \boldsymbol{\phi}_k, \boldsymbol{\psi}_k, \boldsymbol{\lambda}_k\big) = -\mathbb{E}_{q(\mathbf{Q}_k;\boldsymbol{\phi}_k),q(\mathbf{R}_k;\boldsymbol{\psi}_k)}\left[\log p(\mathcal{D}_k \mid \mathbf{W} \circ \mathbf{Q}_k \boldsymbol{\Lambda}_k \mathbf{R}_k^{\mathsf{T}})\right] + \mathrm{KL}\big(q(\mathbf{Q}_k;\boldsymbol{\phi}_k) \,\|\, p(\mathbf{Q}_k)\big) + \mathrm{KL}\big(q(\mathbf{R}_k;\boldsymbol{\psi}_k) \,\|\, p(\mathbf{R}_k)\big), \quad (8)$$

and $\boldsymbol{\lambda}_k$ represents the diagonal vector of the matrix $\boldsymbol{\Lambda}_k$.

To solve this problem, we adopt a coordinate descent strategy, decoupling the updates of the Bayesian mask parameters $\{\boldsymbol{\phi}_k, \boldsymbol{\psi}_k, \boldsymbol{\lambda}_k\}_{k=1}^K$ from the shared model $\mathbf{W}$, along with a continuous relaxation of the binary variables $\{\boldsymbol{\lambda}_k\}_{k=1}^K$. The decoupling of updates allows the global model to capture general patterns across all clients via federated updates, while the Bayesian masks adapt to the local datasets.

Specifically, for a given shared model $\mathbf{W}$, each client k addresses problem (7) over the parameters $(\boldsymbol{\phi}_k, \boldsymbol{\psi}_k, \boldsymbol{\lambda}_k)$ by relaxing the entries of the vector $\boldsymbol{\lambda}_k$ as

$$\lambda_{k,i} = \sigma(\gamma_{k,i}), \quad (9)$$

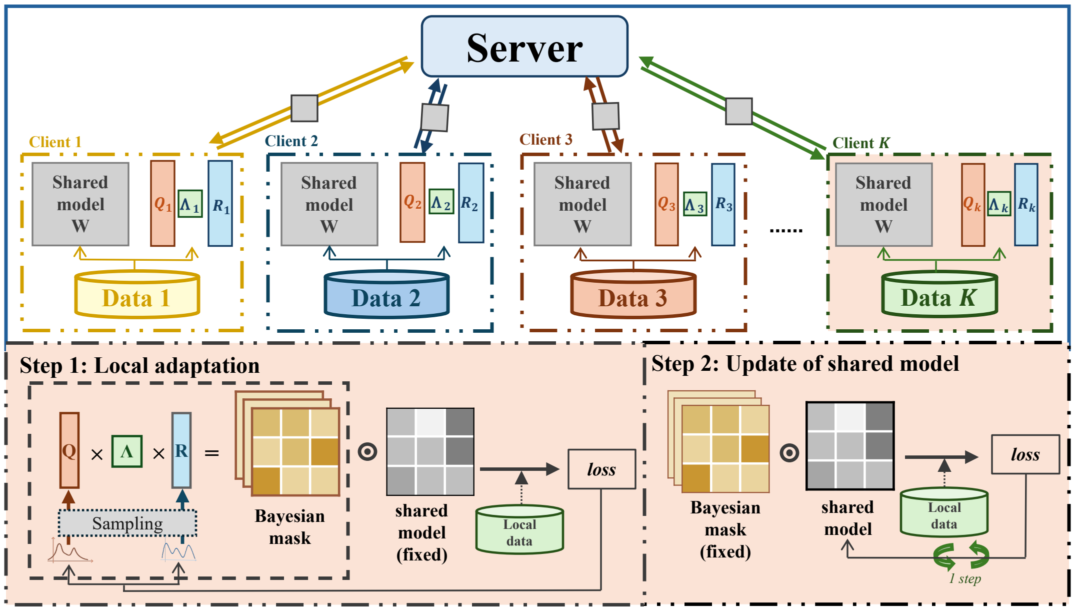

Figure 2: **Top:** In the proposed LR-BPFL scheme, each client uploads its updated shared model to the server and then downloads the deterministic shared model from the server, while retaining and updating the Bayesian low-rank corrections locally. **Bottom:** During local training, after receiving the shared model from the server, clients perform local adaptation of the low-rank corrections before proceeding to update the shared model.

where $\sigma(\gamma) = (1 + \exp(-\gamma))^{-1}$ is the sigmoid function, so that optimization is carried out over the corresponding unconstrained variables $\{\gamma_{k,i}\}_{i=1}^{r_{\max}}$. Furthermore, we model the variational distributions $q(\mathbf{Q}_k; \boldsymbol{\phi}_k), q(\mathbf{R}_k; \boldsymbol{\psi}_k)$ as Gaussian distributions with a trainable mean and diagonal covariance matrix. The priors $p(\mathbf{Q}_k)$ and $p(\mathbf{R}_k)$ also follow Gaussian distributions with fixed means and diagonal covariances. In a manner similar to Hu et al. (2022), we specifically set the means to $1/\sqrt{r_k}$, with r_k being the current rank (6). This way, the average value of the Bayesian mask elements is a priori equal to 1. The variance is set to a small number, such as 0.1.

Optimization over the parameters $(\boldsymbol{\phi}_k, \boldsymbol{\psi}_k)$ is performed by client k using the reparameterization trick, and the KL divergence terms are evaluated in closed form (see, e.g., Simeone (2022)). To approximate the expected likelihood term in (8), we use Monte Carlo averaging, drawing C samples from the variational distributions. To promote the selection of the most effective ranks, an L2-norm regularization term is applied to the vector $\boldsymbol{\lambda}_k$.

After completing the local optimization of the parameters $(\boldsymbol{\phi}_k, \boldsymbol{\psi}_k, \boldsymbol{\lambda}_k)$, each active client updates the shared model $\mathbf{W}$ via stochastic gradient descent on the free energy (8). Subsequently, the rank r_k is updated via a thresholding mechanism, with a fixed threshold $\bar{\lambda}$ as

$$\lambda_{k,i} \leftarrow \lambda_{k,i} \cdot \mathbf{1}_{\{\lambda_{k,i} \geq \bar{\lambda}\}}, \tag{10}$$

where $\mathbf{1}_{\{.\}}$ is the indicator function and $\bar{\lambda}$ is a hyperparameter. We recommend setting the threshold $\bar{\lambda}$ close to 1 (e.g., 0.95).

Finally, the selected clients send their updated local models to the server, which then aggregates these models to obtain the next iterate.

5 EXPERIMENTS

To validate the effectiveness of LR-BPFL, this section reports on the results of extensive experiments across multiple scenarios, including non-i.i.d. data distributions, varying local dataset sizes, and different generalization requirements. Additionally, we performed an ablation study to assess the impact of the adaptive rank selection module on the model performance. We evaluated both accuracy and uncertainty calibration performance on several datasets, including CIFAR-10 and CIFAR-100.

Baselines. We compare LR-BPFL with the standard FL method FedAvg (McMahan et al., 2017), the BFL algorithm FedBE (H.-Y. Chen & Chao, 2020), PFL algorithms such as PerFedAvg (Fallah et al., 2020) and pFedME (T Dinh et al., 2020), as well as the BPFL algorithms MetaVD (Jeon et al., 2024), pFedGP (Achituve et al., 2021) and pFedBayes (Zhang et al., 2022).

Experimental setting. To maintain consistency with previous research, we employ a widely adopted

Table 1: Comparison of computational, communication, memory requirements for different BPFL algorithms.

Method	Training time (s)	Parameters (M)	Communication (MB)
MetaVD (Jeon et al., 2024)	0.026	2.44	9.76
pFedGP (Achituve et al., 2021)	0.023	**1.22**	**4.88**
pFedBayes (Zhang et al., 2022)	0.034	2.44	9.76
LR-BPFL (ours)	**0.018**	1.27	**4.88**

Table 2: Average Uncertainty calibration scores (ECE and MCE) evaluated on the CIFAR-10 and CIFAR-100 datasets. Lower scores indicate better calibration.

	Dataset	CIFAR-10				CIFAR-100	
	Data Heterogeneity	2/10		5/10		5/100	
	Method	ECE	MCE	ECE	MCE	ECE	MCE
FL	FedAvg (McMahan et al., 2017)	0.343	0.495	0.382	0.517	0.623	0.745
BFL	FedBE (H.-Y. Chen & Chao, 2020)	0.285	0.464	0.313	0.471	0.527	0.657
PFL	Per-FedAvg (Fallah et al., 2020)	0.127	0.222	0.316	0.438	0.353	0.491
	pFedMe (T Dinh et al., 2020)	0.224	0.318	0.265	0.445	0.368	0.536
BPFL	MetaVD (Jeon et al., 2024)	0.215	0.337	0.231	0.385	0.343	0.516
	pFedGP (Achituve et al., 2021)	0.169	0.315	0.210	0.333	0.265	0.446
	pFedBayes (Zhang et al., 2022)	0.062	0.287	0.082	0.322	0.148	0.256
	LR-BPFL (Ours)	**0.030**	**0.121**	**0.037**	**0.138**	**0.054**	**0.172**

CNN model across all algorithms and datasets. Specifically, the model comprises three convolutional layers, followed by three fully-connected layers for a total of 1.22 million parameters. For LR-BPFL, we set the maximum rank to $r_{\max} = 8$, the variance of the priors $p(\mathbf{Q}_k)$ and $p(\mathbf{R}_k)$ to 0.1, the threshold for pruning ranks to $\bar{\lambda} = 0.95$, and the ensemble size to $C = 4$ for training and inference. Specifically, the predictive distribution is achieved by averaging the predictive probabilities from $C = 4$ randomly generated Bayesian masks.

We adopt the non-i.i.d. setting from Kotelevskii et al. (2022). For CIFAR-10, we employ two configurations: in the first, each client has 2 out of the 10 labels, denoted as 2/10; and in the second, each client has 5 out of the 10 labels, denoted as 5/10. For CIFAR-100, each client has 5 out of the 100 labels, denoted as 5/100. Data heterogeneity is determined by the number of labels per client, and thus the 2-label setup is more non-i.i.d. than the 5-label setup. In all cases, labels are randomly selected from the label pools. Since small datasets pose a greater challenge to calibration, we train using only 10% of the total training samples. In our experimental setup, we configure the number of communication rounds to $T = 1000$, the number of clients to $K = 50$, the fraction ratio of selected clients for each round to $\tau = 0.2$, and the local update steps to 20. All experiments are conducted on a server equipped with an NVIDIA GeForce 4090 GPU, which includes 24GB of memory. For specific hyperparameter settings, please refer to the Appendix.

5.1 Computational, Memory, and Communication Requirements

In this section, we compare the computational, memory, and communication requirements across the mentioned benchmark BPFL algorithms. Training time is measured as the average time per model update. As shown in Table 1, LR-BPFL achieves the shortest training time per step. Furthermore, in terms of memory requirements, LR-BPFL adds only 50K parameters — a 4% overhead — significantly reducing memory usage as compared to other BFL algorithms. As for the communication overhead, since only the shared model is transmitted, the number of parameters sent remains identical to conventional FL algorithms. In contrast, both MetaVD and pFedBayes double the communication and memory overheads, due to the necessity of transmitting and storing both the means and variances for Bayesian updates.

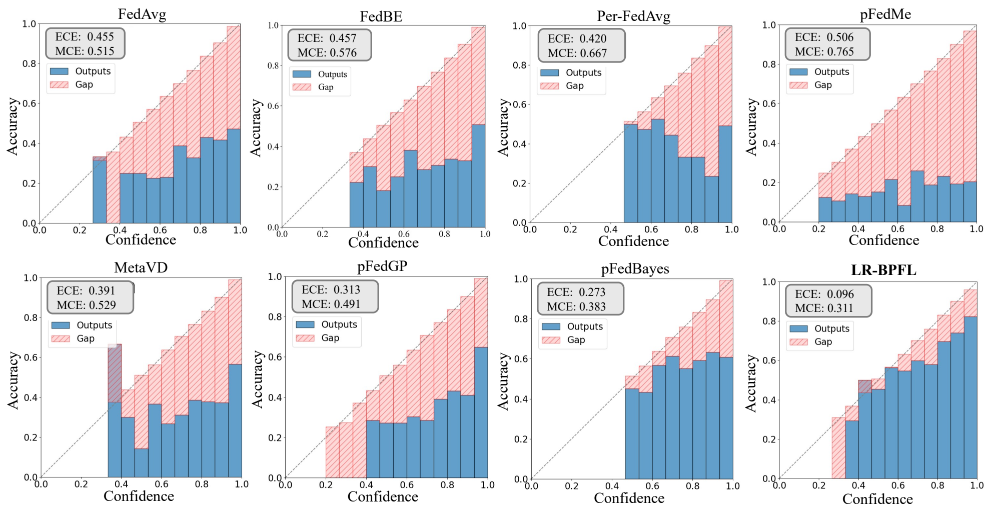

Figure 3: Reliability diagrams of the worst-calibrated client on CIFAR-10 with 50 clients. The diagonal line represents perfect calibration. Each plot also displays the ECE and MCE values.

5.2 Uncertainty Calibration

Calibration is assessed using the expected calibration error (ECE) (Guo et al., 2017), which measures the average deviation between predicted probabilities and actual true class frequencies, and the maximum calibration error (MCE), which captures the largest deviation between predicted and actual classes. Table 2 summarizes the ECE and MCE results for the CIFAR-10 and CIFAR-100 datasets. The results indicate that BPFL algorithms generally achieve lower ECE and MCE values as compared to conventional FL algorithms such as FedAvg, PerFedAvg, and pFedMe. LR-BPFL achieves the best ECE and MCE values among all the benchmark BPFL methods, highlighting its effectiveness in improving model calibration.

To further investigate the impact of heterogeneity, Fig. 3 illustrates the worst calibration performance among all 50 participating clients. Compared to other methods, the reliability diagram for LR-BPFL remains closer to the diagonal line, indicating better calibration. This improvement in the worst client's calibration performance underscores LR-BPFL's capability to achieve personalized calibration.

5.3 Accuracy

To evaluate the accuracy of the proposed method under non-i.i.d. data conditions, we conduct experiments using the same configuration described in the previous subsection, with datasets containing 40% of the training samples. Table 3 presents the average classification accuracy on the CIFAR-10 and CIFAR-100 datasets across various non-i.i.d. settings defined by the ratio between labels present at a client and overall number of labels. As shown in Table 3, PFL methods, such as pFedMe and pFedGP, generally outperform non-personalized FL methods like FedAvg. In all tested settings, LR-BPFL consistently outperforms other benchmark algorithms.

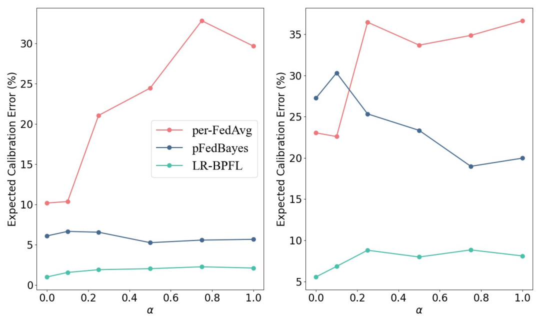

Figure 4: Calibration performance for new clients as a function of the parameter α, which dictates the degree to which the local data distributions of the test clients differ from those of the training clients. As α moves away from 0.1, the distributions of new clients diverge more from those of the training clients.

5.4 Generalization to New Clients

To evaluate the calibration performance in unseen clients, which do not participate in training, we follow the evaluation method outlined in Shamsian et al. (2021). Accordingly, the CIFAR-10 dataset is split into two distinct subsets: the first subset is distributed across 50 clients for model training, while the second is assigned to 10 additional clients for evaluation as new clients. Within both subsets, we set the class probabilities in each client by sampling from a Dirichlet dis-

Table 3: Classification accuracies (in %) with different heterogeneity degrees. Averages over 5 seeds are reported.

Dataset	CIFAR10		CIFAR100
Data heterogeneity	2/10	5/10	5/100
FedAvg (McMahan et al., 2017)	66.37	68.66	33.05
FedBE (H.-Y. Chen & Chao, 2020)	67.05	68.75	34.11
Per-FedAvg (Fallah et al., 2020)	82.38	68.91	60.24
pFedMe (T Dinh et al., 2020)	78.14	70.45	47.15
MetaVD (Jeon et al., 2024)	67.52	68.58	34.23
pFedGP (Achituve et al., 2021)	67.11	68.25	33.29
pFedBayes (Zhang et al., 2022)	82.28	69.43	62.47
LR-BPFL (Ours)	**84.67**	**72.38**	**66.12**

Table 4: Performance comparison of LR-BPFL with and without adaptive rank selection (LR-BPFL w/o ARS) on the CIFAR-10 and CIFAR-100 datasets. The table shows accuracy (in %), average expected calibration error (A-ECE), and worst-client calibration error (W-ECE).

	CIFAR-10 dataset						*CIFAR-100* dataset		
	2/10			5/10			5/100		
Method	ACC	A-ECE	W-ECE	ACC	A-ECE	W-ECE	ACC	A-ECE	W-ECE
LR-BPFL (w/o ARS)	82.62	0.035	0.130	64.88	0.044	0.143	55.64	0.061	0.156
LR-BPFL	**83.30**	**0.030**	**0.096**	**65.68**	**0.038**	**0.111**	**56.49**	**0.054**	**0.124**

tribution with the same α parameter. For the training subset, we set the parameter $\alpha = 0.1$. During testing, we evaluate the performance of the model on new clients by varying $\alpha \in \{0.1, 0.25, 0.5, 0.75, 1.0\}$. As α moves away from 0.1, the distributions of new clients diverge more from those of training clients, making adaptation to new clients more challenging. Fig. 4 illustrates the average ECE and the worst-client ECE as a function of the Dirichlet parameter α, where $\alpha = 0$ corresponds to the results of the clients involved in the training process. LR-BPFL consistently demonstrates superior calibration performance across all α values. Notably, even for the worst-performing clients, LR-BPFL maintains stable and reliable calibration. These findings highlight the robustness of our method in adapting to new clients while preserving calibration quality among all participants.

5.5 On the Role of Adaptive Rank Selection

In this section, we conduct an ablation study to assess the effect of the adaptive rank selection (ARS) module in LR-BPFL. Table 4 presents the classification accuracy, average ECE, and worst-client ECE for LR-BPFL with and without the ARS module. The results clearly demonstrate that incorporating the ARS module significantly improves both accuracy and model calibration. Fig. 5 provides a visual comparison for a specific client, with the left figure representing the results with ARS and the right figure representing the results without it. The comparison highlights the significant contribution of ARS to overall performance.

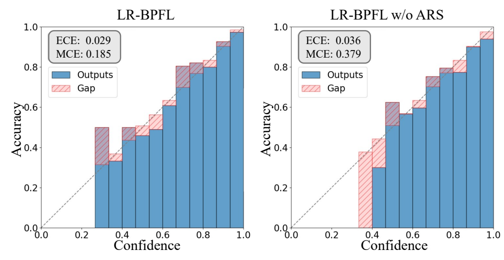

Figure 5: Comparison of reliability diagrams for LR-BPFL with and without the ARS module.

6 CONCLUSION

We have proposed a novel personalized FL protocol that jointly learns a global deterministic model, along with personalized low-rank Bayesian corrections for each client. The local adaptation of Bayesian masks incorporates an adaptive rank selection module, which dynamically tailors the local model by adjusting the rank of the local masks. Our experiments validated that LR-BPFL can effectively address data heterogeneity, local epistemic uncertainty, and local computational and memory constraints, showcasing improved accuracy and calibration, as well as the capability to generalize to new clients as compared to the state of the art. Future work may explore fully decentralized versions of LR-BPFL.

References

Achituve, I., Shamsian, A., Navon, A., Chechik, G., & Fetaya, E. (2021). Personalized federated learning with gaussian processes. *Advances in Neural Information Processing Systems*, *34*, 8392–8406.

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

Atzeni, M., Sachan, M., & Loukas, A. (2023). Infusing lattice symmetry priors in attention mechanisms for sample-efficient abstract geometric reasoning. In *International conference on machine learning* (pp. 1200–1217).

Bai, J., Song, Q., & Cheng, G. (2020). Efficient variational inference for sparse deep learning with theoretical guarantee. *Advances in Neural Information Processing Systems*, *33*, 466–476.

Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern recognition and machine learning* (Vol. 4) (No. 4). Springer.

Briggs, C., Fan, Z., & Andras, P. (2020). Federated learning with hierarchical clustering of local updates to improve training on non-iid data. In *2020 international joint conference on neural networks (ijcnn)* (pp. 1–9).

Cai, R., Chen, X., Liu, S., Srinivasa, J., Lee, M., Kompella, R., & Wang, Z. (2023). Many-task federated learning: A new problem setting and a simple baseline. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 5037–5045).

Chen, F., Luo, M., Dong, Z., Li, Z., & He, X. (2018). Federated meta-learning with fast convergence and efficient communication. *arXiv preprint arXiv:1802.07876*.

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

Collins, L., Hassani, H., Mokhtari, A., & Shakkottai, S. (2021). Exploiting shared representations for personalized federated learning. In *International conference on machine learning* (pp. 2089–2099).

Daxberger, E., Kristiadi, A., Immer, A., Eschenhagen, R., Bauer, M., & Hennig, P. (2021). Laplace redux-effortless bayesian deep learning. *Advances in Neural Information Processing Systems*, *34*, 20089–20103.

Deng, Y., Kamani, M. M., & Mahdavi, M. (2020). Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461*.

Dusenberry, M., Jerfel, G., Wen, Y., Ma, Y., Snoek, J., Heller, K., ... Tran, D. (2020). Efficient and scalable bayesian neural nets with rank-1 factors. In *International conference on machine learning* (pp. 2782–2792).

Fallah, A., Mokhtari, A., & Ozdaglar, A. (2020). Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *Advances in Neural Information Processing Systems*, *33*, 3557–3568.

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. In *International conference on machine learning* (pp. 1321–1330).

Hanzely, F., & Richtárik, P. (2020). Federated learning of a mixture of global and local models. *arXiv preprint arXiv:2002.05516*.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., ... Chen, W. (2022). LoRA: Low-rank adaptation of large language models. In *International conference on learning representations.* Retrieved from `https://openreview.net/forum?id=nZeVKeeFYf9`

Jeon, I., Hong, M., Yun, J., & Kim, G. (2024). Federated learning via meta-variational dropout. *Advances in Neural Information Processing Systems*, *36*.

Kassab, R., & Simeone, O. (2022). Federated generalized bayesian learning via distributed stein variational gradient descent. *IEEE Transactions on Signal Processing*, *70*, 2180–2192.

Khan, M. E., & Rue, H. (2021). The bayesian learning rule. *arXiv preprint arXiv:2107.04562*.

Knoblauch, J., Jewson, J., & Damoulas, T. (2019). Generalized variational inference: Three arguments for deriving new posteriors. *arXiv preprint arXiv:1904.02063*.

Kotelevskii, N., Vono, M., Durmus, A., & Moulines, E. (2022). Fedpop: A bayesian approach for personalised federated learning. *Advances in Neural Information Processing Systems*, *35*, 8687–8701.

Lee, R., Kim, M., Li, D., Qiu, X., Hospedales, T., Huszár, F., & Lane, N. (2024). Fedl2p: Federated learning to personalize. *Advances in Neural Information Processing Systems*, *36*.

Li, C., Farkhoor, H., Liu, R., & Yosinski, J. (2018). Measuring the intrinsic dimension of objective landscapes. *arXiv preprint arXiv:1804.08838*.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., & Smith, V. (2020). Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, *2*, 429–450.

Ma, J., Zhou, T., Long, G., Jiang, J., & Zhang, C. (2023). Structured federated learning through clustered additive modeling. *Advances in Neural Information Processing Systems*, *36*, 43097–43107.

Marfoq, O., Neglia, G., Bellet, A., Kameni, L., & Vidal, R. (2021). Federated multi-task learning under a mixture of distributions. *Advances in Neural Information Processing Systems*, *34*, 15434–15447.

McMahan, B., Moore, E., Ramage, D., Hampson, S., & y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics* (pp. 1273–1282).

Molchanov, D., Ashukha, A., & Vetrov, D. (2017). Variational dropout sparsifies deep neural networks. In *International conference on machine learning* (pp. 2498–2507).

Neal, R. M. (2012). *Bayesian learning for neural networks* (Vol. 118). Springer Science & Business Media.

Nguyen, D. C., Ding, M., Pathirana, P. N., Seneviratne, A., Li, J., & Poor, H. V. (2021). Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, *23*(3), 1622–1658.

Oh, J., Kim, S., & Yun, S.-Y. (2021). Fedbabu: Towards enhanced representation for federated image classification. *arXiv preprint arXiv:2106.06042*.

Sattler, F., Müller, K.-R., & Samek, W. (2020). Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE transactions on neural networks and learning systems*, *32*(8), 3710–3722.

Sen, D., Papamarkou, T., & Dunson, D. (2024). Bayesian neural networks and dimensionality reduction. In *Handbook of bayesian, fiducial, and frequentist inference* (pp. 188–209). Chapman and Hall/CRC.

Shamsian, A., Navon, A., Fetaya, E., & Chechik, G. (2021). Personalized federated learning using hypernetworks. In *International conference on machine learning* (pp. 9489–9502).

Simeone, O. (2022). *Machine learning for engineers*. Cambridge university press.

Smith, V., Chiang, C.-K., Sanjabi, M., & Talwalkar, A. S. (2017). Federated multi-task learning. *Advances in neural information processing systems*, *30*.

Tan, A. Z., Yu, H., Cui, L., & Yang, Q. (2022). Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems*.

T Dinh, C., Tran, N., & Nguyen, J. (2020). Personalized federated learning with moreau envelopes. *Advances in Neural Information Processing Systems*, *33*, 21394–21405.

Vono, M., Plassier, V., Durmus, A., Dieuleveut, A., & Moulines, E. (2022). Qlsd: Quantised langevin stochastic dynamics for bayesian federated learning. In *International conference on artificial intelligence and statistics* (pp. 6459–6500).

Wen, Y., Tran, D., & Ba, J. (2020). Batchensemble: an alternative approach to efficient ensemble and lifelong learning. *arXiv preprint arXiv:2002.06715*.

Wu, X., Huang, H., Ding, Y., Wang, H., Wang, Y., & Xu, Q. (2023). Fednp: Towards non-iid federated learning via federated neural propagation. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 37, pp. 10399–10407).

Xu, J., Glicksberg, B. S., Su, C., Walker, P., Bian, J., & Wang, F. (2021). Federated learning for healthcare informatics. *Journal of healthcare informatics research*, *5*, 1–19.

Xu, J., Tong, X., & Huang, S.-L. (2023). Personalized federated learning with feature alignment and classifier collaboration. *arXiv preprint arXiv:2306.11867*.

Zhang, X., Li, Y., Li, W., Guo, K., & Shao, Y. (2022). Personalized federated learning via variational bayesian inference. In *International conference on machine learning* (pp. 26293–26310).

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

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. See Section 3 and 4.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. See Table 1.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. Not Applicable.
   (b) Complete proofs of all theoretical results. Not Applicable.
   (c) Clear explanations of any assumptions. Not Applicable.

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). Yes.
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). Yes.
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). Yes.
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). Yes.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. Yes.
   (b) The license information of the assets, if applicable. Yes.
   (c) New assets either in the supplemental material or as a URL, if applicable. Yes.
   (d) Information about consent from data providers/curators. Not applicable.
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. No.

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. Not Applicable.
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. Not Applicable.
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. Not Applicable.

A IMPLEMENTATION DETAILS

A.1 Baselines

We present an overview of the baseline models used in our experiments, covering state-of-the-art approaches across four categories: FL, PFL, BFL, and BPFL methods.

The following **FL method** is considered in our experiments:

- FedAvg (McMahan et al., 2017) is a standard FL algorithm that averages gradients weighted by the data size of clients in each FL round.

The following **BFL method** is considered in our experiments:

- FedBE (H.-Y. Chen & Chao, 2020) enhances robust aggregation by sampling high-quality global models and combining them through Bayesian model ensembling, where the global model distribution is derived by considering each client's local model as a potential global model.

The following **PFL methods** are considered in our experiments:

- Per-FedAvg (Fallah et al., 2020) applies model-agnostic meta-learning (MAML) to fine-tune the global model with local gradient updates, enabling personalized adaptation in federated learning.
- pFedMe (T Dinh et al., 2020) addresses personalized FL as a bi-level optimization problem, decoupling the personalized model optimization from global model learning by employing Moreau envelopes as the regularized loss function for each client.

The following **BPFL methods** are considered in our experiments:

- pFedGP (Achituve et al., 2021) employs Gaussian process for personalized federated learning, leveraging shared kernel functions parameterized by a neural network and personalized Gaussian process classifier.
- MetaVD (Jeon et al., 2024) models the local posterior distribution as a Gaussian, where the mean is derived from the deterministic global model, and the client-specific diagonal covariance is produced by a shared hypernetwork.
- pFedBayes (Zhang et al., 2022) enables each client to maintain its own personalized BNN while using the aggregated global posterior as the prior distribution.

A.2 Experimental hyperparameters

For all datasets, we set $T = 1000$ to ensure convergence following standard conventions. The batch size is set to 32, and local steps are set to 20. In order to ensure a fair comparison between the algorithms, the results presented in all of our experiments are obtained using optimal hyperparameters for each model.

For hyperparameter settings, we refer to the configurations in the corresponding papers of each algorithm and fine-tune the parameters around their recommended values. For instance, if an algorithm uses a learning rate $\eta = 0.01$, then we set the learning rate within the range of 0.001 to 0.1 for parameter tuning. For algorithm-specific hyperparameters, we use the values recommended in their papers.

Based on our experimental results, we set the learning rates for FedAvg to 0.1, and for Per-FedAvg and FedBE to 0.01. For pFedMe, the personalized learning rate, global learning rate, and regularization weight are set to 0.01, 0.01, and 15, respectively. The learning rate for pFedGP is set to 0.05. For pFedBayes, the tradeoff parameter is set to $\zeta = 10$ (see Eq. (27) in Zhang et al. (2022)), the learning rates for the personalized and global models are set to $\eta_1 = \eta_2 = 0.001$. For LR-BPFL, we set the learning rate for the global deterministic model to 0.001 and for the Bayesian mask to 0.01. To ensure reproducibility of the experiments, we will release all code on our GitHub repository.

A.3 Dataset

We provide a description of the datasets used in our experiments. The CIFAR-10 and CIFAR-100 datasets are commonly used for 10-class and 100-class image classification, respectively. Each contains 50,000 training images and 10,000 test images, all with a resolution of 32x32 pixels. In our experiments, we combine the training and test images, then distribute them to clients according to different non-i.i.d. settings to verify the impact of data heterogeneity. Finally, each client splits their assigned data into training and testing sets according to the specified percentage.

B ADDITIONAL EXPERIMENTS

B.1 Impact of local dataset sizes

To evaluate the accuracy of the proposed method with varying local dataset sizes, we consider two configurations: the small dataset uses 10% of the data for training, while the medium dataset uses 40% of the data for training. The results are presented in Table 5. In the small dataset scenario, all schemes perform better with the more non-i.i.d. data distribution (i.e., heterogeneity degree 2/10). A possible explanation is that when the dataset size is limited, a more i.i.d. distribution results in fewer samples per label for each client, which can lead to severe overfitting and, consequently, a decrease in accuracy. In all tested settings, our approach consistently outperforms other SOTA algorithms.

Table 5: Classification accuracies with different heterogeneity degrees and varying dataset sizes. The higher score, the better. Averages over 5 seeds are reported.

	Small (Acc. (%))			Medium (Acc. (%))		
Dataset	CIFAR10		CIFAR100	CIFAR10		CIFAR100
Data heterogeneity	2/10	5/10	5/100	2/10	5/10	5/100
FedAvg (McMahan et al., 2017)	56.45	53.80	19.39	66.37	68.66	33.05
FedBE (H.-Y. Chen & Chao, 2020)	57.19	54.42	19.89	67.05	68.75	34.11
PerFedAvg (Fallah et al., 2020)	77.04	54.84	45.49	82.38	68.91	60.24
pFedMe (T Dinh et al., 2020)	61.37	57.53	37.27	78.14	70.45	47.15
MetaVD (Jeon et al., 2024)	57.02	54.58	20.74	67.52	68.58	34.23
pFedGP (Achituve et al., 2021)	55.14	52.27	19.05	67.11	68.25	33.29
pFedBayes (Zhang et al., 2022)	80.86	62.25	52.11	82.28	69.43	62.47
Ours	**83.30**	**64.61**	**54.83**	**84.67**	**72.38**	**66.12**

B.2 Impact of Noisy Dataset

To investigate the impact of noisy data on our algorithm, we use 10% of the CIFAR-10 samples, with each client assigned 5 out of 10 labels, and apply 57 unique image corruption distributions as in Achituve et al. (2021). Each client is assigned a distinct noise model, and within the client, each noisy data sample is randomly drawn based on the assigned model. The results, as illustrated in Fig. 6, demonstrate the impact of noise on both accuracy and calibration. The accuracy is represented along the x-axis, while ECE is plotted on the y-axis, with better performance appearing closer to the lower right corner. The results show that models trained on the noisy dataset suffer significant degradation in accuracy compared to those trained on the original dataset. However, our approach calibrates the model well even under noisy datasets, remaining robust for providing reliable predictions despite showing a lower accuracy.

B.3 On the Role of Coordinate Descent

In this section, we investigate different update strategies by comparing the coordinate descent approach used in LR-BPFL with a joint update strategy, where the global deterministic model and the local Bayesian mask are updated jointly, but only the deterministic model is shared with the server during each communication round.

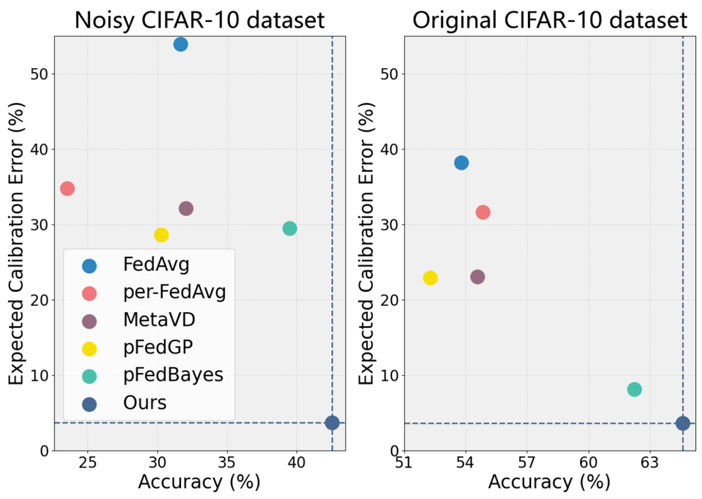

Figure 6: Test accuracy and ECE (%) on noisy and original CIFAR-10 dataset.

This setup is referred to as LR-BPFL (joint). As shown in Table 6, the coordinate descent approach leads to significant improvements in both accuracy and calibration. These results indicate that decoupling the updates of the global model and personalized masks allows the global model to focus on learning general patterns across all clients via federated updates, while the personalized Bayesian masks adapt to local datasets, resulting in enhanced overall performance.

Table 6: Performance comparison of LR-BPFL and LR-BPFL (joint) on the CIFAR-10 and CIFAR-100 datasets. The table shows accuracy (in %), average expected calibration error (A-ECE), and worst-client calibration error (W-ECE).

	CIFAR-10 dataset						*CIFAR-100* dataset		
	2/10			5/10			5/100		
Method	ACC	A-ECE	W-ECE	ACC	A-ECE	W-ECE	ACC	A-ECE	W-ECE
LR-BPFL (joint)	80.79	0.065	0.194	61.76	0.154	0.297	52.93	0.115	0.246
LR-BPFL	**83.30**	**0.030**	**0.096**	**65.68**	**0.038**	**0.111**	**56.49**	**0.054**	**0.124**

B.4 The effect of rank selection

Since each Bayesian mask must retain at least one rank for personalization, we exclude the first rank from the pruning strategy. In our experiments, each binary element of the diagonal vector $\boldsymbol{\lambda}$ in the gating matrix is relaxed to $\lambda = \sigma(\gamma)$, where σ is the sigmoid function. The initial value of γ is set at 3.5, and we set the pruning threshold at 0.95. To encourage the selection of effective ranks, we apply an L2-norm regularization term with a weight of 0.1 to the vector $\boldsymbol{\lambda}$ in the loss function. We also set the learning rate to 0.001 for updating γ. Fig. 7 illustrates an example of the final rank distribution per client. As shown, optimal ranks vary between different layers and clients. This supports our hypothesis that clients require different BNN architectures to adapt to their uncertainty levels.

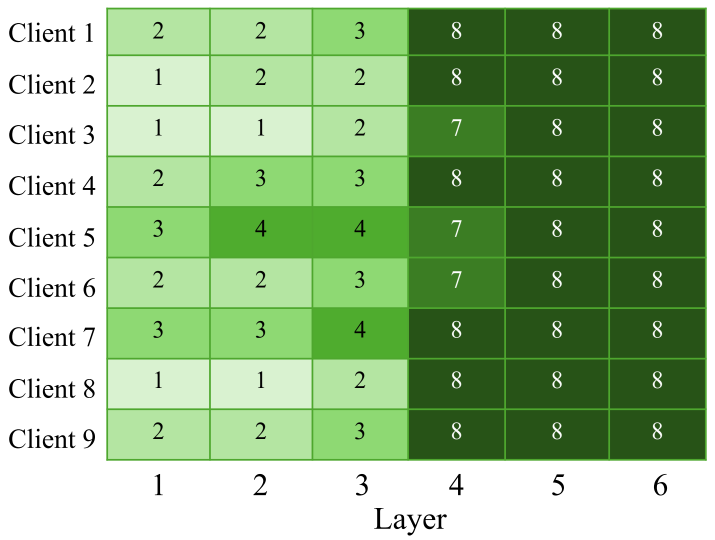

Figure 7: Resulting rank of each Bayesian mask based on the CIFAR-10 dataset using LR-BPFL. The x-axis shows the layer index, and the y-axis represents different clients.