Flat Posterior For Bayesian Model Averaging

Sungjun Lim[1*] **Jeyoon Yeom**[1] **Sooyon Kim**[2] **Hoyoon Byun**[1] **Jinho Kang**[3]
Yohan Jung[4†] **Jiyoung Jung**[3†] **Kyungwoo Song**[1†]
[1]Yonsei University [2]Ohio State University [3]University of Seoul [4]KAIST

Abstract

Bayesian neural networks (BNNs) estimate the posterior distribution of model parameters and utilize posterior samples for Bayesian Model Averaging (BMA) in prediction. However, despite the crucial role of flatness in the loss landscape in improving the generalization of neural networks, its impact on BMA has been largely overlooked. In this work, we explore how posterior flatness influences BMA generalization and empirically demonstrate that *(1) most approximate Bayesian inference methods fail to yield a flat posterior* and *(2) BMA predictions, without considering posterior flatness, are less effective at improving generalization*. To address this, we propose Flat Posterior-aware Bayesian Model Averaging (FP-BMA), a novel training objective that explicitly encourages flat posteriors in a principled Bayesian manner. We also introduce a Flat Posterior-aware Bayesian Transfer Learning scheme that enhances generalization in downstream tasks. Empirically, we show that FP-BMA successfully captures flat posteriors, improving generalization performance[1].

1 Introduction

Bayesian neural networks (BNNs) provide a theoretically grounded framework for modeling uncertainty in deep learning by approximating the posterior distribution of model parameters (MacKay, 1992; Hinton & Van Camp, 1993; Neal, 2012). The approximated posterior is used for making predictions through Bayesian Model Averaging (BMA) (Wasserman, 2000; Fragoso et al., 2018; Wilson & Izmailov, 2020; Zeng & Van den Broeck, 2024). It allows BNNs to account for uncertainty in predictions, leading to more reliable outcomes compared to the deterministic neural networks (DNNs) (Kapoor et al., 2022; Kristiadi et al., 2022b). The accuracy and robustness of BNN predictions are heavily dependent on the quality of the approximated posterior (Kristiadi et al., 2022a; Wenzel et al., 2020).

The flatness of loss landscape has been strongly associated with better generalization ability, as they represent solutions that are less sensitive to small perturbations in model parameters (Hochreiter & Schmidhuber, 1997; Keskar et al., 2016; Neyshabur et al., 2017). The flatness has been extensively studied in the context of DNNs, but no comprehensive analysis has been conducted on its role in BNNs or its impact on BMA. SA-BNN (Nguyen et al., 2023) incorporated a flat-seeking optimizer into BNNs but merely adapted a DNN-based optimizer without considering the probabilistic nature of BNNs, leading to only limited improvements. On the other hand, E-MCMC (Li & Zhang, 2023) introduced a guidance model to achieve flat posteriors, but this approach is less suited for large-scale models.

In this work, we first demonstrate that BNNs often struggle to capture the flatness. In detail, we compare the flatness of various BNN frameworks against that of DNNs and demonstrate that *(1) most approximate Bayesian inference methods fail to yield a flat posterior* and *(2) BMA predictions, without considering posterior flatness, are less effective at improving generalization*. These findings highlight the need for an optimization strategy that accounts for the probabilistic nature of BNNs to estimate flat posteriors effectively.

[*]Correspondence to: `lsj9862@gmail.com`
[†]Corresponding Authors
[1]Code is available at `https://anonymous.4open.science/r/SA-BMA-A890`

Therefore, we propose Flat Posterior-aware Bayesian Model Averaging (FP-BMA), a novel optimization that explicitly targets the flat posterior. We first compute an adversarial posterior in the vicinity of the current posterior, which maximizes the BNN loss. After that, we update the posterior by employing the gradient of the adversarial posterior with respect to the loss. In addition, we introduce a Flat Posterior-aware Bayesian Transfer Learning scheme integrated with FP-BMA, enabling effective capture of flatness. This approach enhances robustness against model misspecification, when the prior is not well-suited for fine-tuning BNNs on downstream tasks. We show that FP-BMA improves the generalization performance of BNNs, particularly in few-shot classification and distribution shift, by ensuring a flat posterior.

2 Preliminary

2.1 Bayesian Neural Networks

Training Let $w \subseteq \mathbb{R}^p$ be the model parameter of BNN and $\mathcal{D} = \{(x, y)\}$ be the datasets with inputs x and outputs y. In principle, training BNNs aims to estimate the posterior distribution $p(w|\mathcal{D})$ based on Bayes' Rule:

$$p(w|\mathcal{D}) = \frac{p(\mathcal{D}|w)p(w)}{\int_w p(\mathcal{D}|w)p(w)dw}, \tag{1}$$

where $p(\mathcal{D}|w)$ and $p(w)$ denote the likelihood of data $\mathcal{D}$ and the prior distribution over w, respectively.

However, the posterior of BNNs $p(w|\mathcal{D})$ is intractable in general. Hence, many approximate inference methods, including Markov Chain Monte Carlo (MCMC) (Welling & Teh, 2011; Chen et al., 2014) and Variational Inference (VI) (Graves, 2011; Blundell et al., 2015), and other variants (Ritter et al., 2018; Daxberger et al., 2021a; Gal & Ghahramani, 2016; Maddox et al., 2019), have been employed to obtain approximate posterior $q_\theta(w|\mathcal{D})$, with distribution's parameter $\theta \subseteq \mathbb{R}^q$, pursuing $q_\theta(w|\mathcal{D}) \approx p(w|\mathcal{D})$.

Prediction For the approximate posterior $q(w; \theta)$, BNNs make predictions on unobserved data (x^*, y^*):

$$\begin{aligned} p(y^*|x^*, \mathcal{D}) &\approx \int_w p(y^*|f_w(x^*))q_\theta(w|\mathcal{D})dw \\ &\approx \frac{1}{M}\sum_{m=1}^{M} p(y^*|f_{w_m}(x^*)), \;\; w_m \sim q_\theta(w|\mathcal{D}), \end{aligned} \tag{2}$$

where $f_w(\cdot)$ is predictions with parameter w and M denotes the number of sampled model; the first approximation uses $q_\theta(w|\mathcal{D})$ and second approximation in Eq. 2 employs Monte Carlo integration. It is known to be effective for improving generalization ability, as BMA leverages diverse samples $\{w_m\}_{m=1}^M$ from the posterior for prediction (Wilson & Izmailov, 2020).

2.2 Flatness and Optimization

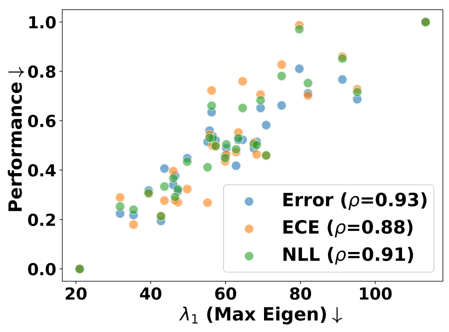
As the flatness of loss surface has been known to be connected to the model generalization (Hochreiter & Schmidhuber, 1994; 1997; Neyshabur et al., 2017), new training methods have been presented to find the flat local optimum.

One direction is to employ the entropy to find a local optimum (Baldassi et al., 2015; 2016). Specifically, Entropy-SGD and Entropy-SGLD (Chaudhari et al., 2019; Dziugaite & Roy, 2018) employ the idea of a nested chain to elaborate the local entropy search. On the other hand, SAM (Foret et al., 2020) finds adversarial parameter in the γ-neighborhood for training, as follows:

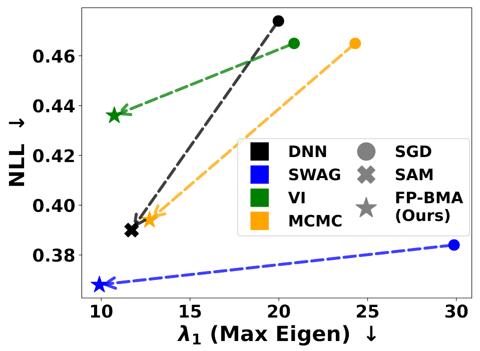
$$\ell^{\gamma}_{\text{SAM}}(w) = \min_w \max_{\|\Delta w\|_p \le \gamma} \ell(f_{w+\Delta w}(x), y),$$

where $\ell(\cdot)$ is the empirical loss function (e.g., cross-entropy for classification tasks) and p is practically set to $p = 2$, yielding $\Delta w = \gamma \nabla_w \ell(w)/\|\nabla_w \ell(w)\|_2$. Furthermore, FSAM (Kim et al., 2022a) improves SAM by replacing the Euclidean ball of SAM with non-Euclidean ball induced by Fisher information:

$$\ell^{\gamma}_{\text{FSAM}}(w) = \min_w \max_{\|F_y(w)\Delta w\|_p \le \gamma^2} \ell(f_{w+\Delta w}(x), y),$$



(a) Correlation between flatness and generalization within sampled models



(b) Flatness and generalization according to the training methods

Figure 1: (a) Flatness, measured via the maximal Hessian eigenvalue (λ_1), is highly correlated with generalization ability (classification error, ECE, and NLL), suggesting that the flatness of models sampled from the posterior is correlated with generalization ability. (b) The existing inferences of BNNs (SWAG, VI, and MCMC) with SGD struggle to capture the flatness compared to DNNs. In contrast, the proposed Bayesian flat posterior-aware optimizer FP-BMA allows BNNs to seek flat minima, improving performance.

where $F_y(w)$ denotes the Fisher information matrix (FIM) and is approximated as $F_y(w) = 1/|B|\nabla_w \log p(y|x,w)^2$ with $|B|$ batch size. SAM and FSAM are both derived under deterministic w, and $F_y(w)$ is defined over the predictive distribution $p(y|f_w(x))$, not in the parameter space.

3 Flatness Does Matter For Bayesian Model Averaging

In this section, we explore the flatness of BNNs' posterior obtained from the widely-used approximate Bayesian inferences and demonstrate that flatness should be considered for BNNs based on both empirical and theoretical grounds.

Experimental Setup We empirically inspect the flatness of BNNs and observe that the generalization ability of BMA prediction improves as weight samples are drawn from a flatter posterior. To this end, we train ResNet18 (He et al., 2016) without Batch Normalization (Ioffe & Szegedy, 2015) on CIFAR10 (Krizhevsky et al., 2009) using following Bayesian inference methods-VI (Blundell et al., 2015), SWAG (Maddox et al., 2019), and MCMC (Welling & Teh, 2011)-to yield the approximate posterior $q_\theta(w|\mathcal{D})$. We then compare the generalization ability, classification error, negative log-likelihood (NLL), and expected calibration error (ECE) with the flatness of the approximate posterior.

Flatness criterion for BNNs To evaluate the flatness of the posterior, we use the average of Hessian's eigenvalues, unlike in DNNs, where flatness is assessed using individual eigenvalues. This difference stems from the fact that the loss of BNNs is formulated as the marginal likelihood over the posterior, incorporating multiple parameter samples $\{w_m\}_{m=1}^M$ drawn from $w_m \sim q_\theta(w|\mathcal{D})$. We use the averaged i-th maximal eigenvalue of Hessian:

$$\lambda_i \approx 1/M \sum_{m=1}^{M} \lambda_i(H_{f_m}), \qquad H_{f_m} = \nabla^2 \ell\big(f_{w_m}(x), y\big), \tag{3}$$

where $\ell\big(f_{w_m}(x), y\big) := -\log p(y|f_{w_m}(x))$ denotes the likelihood using m-th parameter sample $w_m \sim q_\theta(w|\mathcal{D})$. The H_{f_m} denotes the Hessian of the loss $\ell\big(f_{w_m}(x), y\big)$ and $\lambda_i(H_{f_m})$ denotes the i-th maximal eigenvalue of Hessian. Notably, the smaller largest eigenvalues of the Hessian indicate that the model parameters lie in a flatter region of the loss surface. Therefore, the maximal eigenvalue λ_1 or the eigenvalue ratio λ_1/λ_5 is often used to assess the flatness of model parameters (Keskar et al., 2016; Foret et al., 2020; Jastrzebski et al., 2020).

3.1 Need for Flatness in BMA

Takeaway 1: The flatness of models sampled from the posterior is correlated with generalization ability. Figure 1a compares normalized generalization ability—measured by Error, ECE, and NLL—against flatness of BMA models sampled from posterior trained with SWAG. The results reveal a strong positive correlation between flatness and generalization ability, suggesting that *models sampled from the posterior is correlated with generalization ability, same as DNNs.*

Takeaway 2: It is essential to approximate a flat posterior for BMA. We also show that the flatness of BMA is determined by that of individual BMA samples, highlighting the necessity of a flat posterior for effective BMA performance. Specifically, we demonstrate the flatness bound of simple weight averaging and connect it to that of BMA based on Hessian eigenvalues in Theorem 1 (Detailed proof in Appendix A.1).

Theorem 1. *Let twice differentiable loss $\ell(\cdot)$, predictions of model $f_m(\cdot)$ parameterized by w_m, and predictions of BMA $f_{BMA}(\cdot)$. With M model sample $\{w_m\}_{m=1}^{M}$, the maximal eigenvalue of averaged Hessian of loss $\lambda_{max}(H_{f_{BMA}})$ is bounded as follow:*

$$\max\left(\left\{\frac{1}{M}\left(\lambda_{\max}(H_{f_m}) + \sum_{\substack{n=1\\n\neq m}}^{M}\lambda_{\min}(H_{f_n})\right)\right\}_{m=1}^{M}\right) \leq \lambda_{\max}(H_{f_{\text{BMA}}}) \leq \frac{\sum_{m=1}^{M}\lambda_{\max}(H_{f_m})}{M}. \tag{4}$$

Theorem 1 implies that as $\lambda_{\max}(H_{f_m})$ decreases in Eq. 4, where it appears in both the lower and upper bounds, the corresponding $\lambda_{\max}(H_{f_{\text{BMA}}})$ also decreases. This decrease in $\lambda_{\max}(H_{f_{\text{BMA}}})$ represents the reduction in the flatness of the BMA prediction. This suggests that *it is important to ensure each Hessian H_{f_m} of a BMA models sampled from the flat posterior*, which can improve generalization, as demonstrated in the previous section.

3.2 Insufficient Flatness of BMA

Takeaway 3: Most approximate Bayesian inference methods struggle to produce a flat posterior. We investigate whether existing approximate Bayesian inference methods can produce the flat posterior of BNNs. Figure 1b illustrates how NLL and posterior flatness vary depending on whether flatness in the loss surface is taken into account during optimization. We observe that *the approximate posteriors of BNNs do not show better flatness compared to that of DNNs, obtained from the SAM optimizer*. On the other hand, the proposed FP-BMA, which will be described in Section 4.1, allows BNNs to seek flat minima and thus leads to better performance.

4 Bayesian Model Averaging with Flat Posterior

To obtain the flat posterior, we propose a flat posterior-aware optimizer (Section 4.1) and Bayesian transfer learning combined with diverse BNN frameworks (Section 4.2).

4.1 Flat Posterior-aware Optimizer

To deal with the probabilistic nature of BNNs, we suggest a new objective function based on VI:

$$\ell^{\gamma}_{\text{FP-BMA}}(\theta) = \min_{\theta} \max_{d|\theta+\Delta\theta,\theta|\leq\gamma^2} \ell(\theta+\Delta\theta) + \beta \text{D}_{\text{KL}}[q_\theta(w|\mathcal{D})||p(w)] \tag{5}$$

$$\text{s.t.}\ \ d|\theta+\Delta\theta,\theta| = \text{D}_{\text{KL}}\left[q_{\theta+\Delta\theta}(w|\mathcal{D})\ ||\ q_\theta(w|\mathcal{D})\right], \tag{6}$$

where θ and $\Delta\theta$ denote the variational parameters and perturbation on them, respectively. $\ell(\theta+\Delta\theta)$ denotes empirical loss, such as NLL under $q_{\theta+\Delta\theta}(w|\mathcal{D})$, and β is a hyperparameter that controls the influence of the prior. KL divergence allows quantifying perturbations in parameter distributions and, in the typical case of Gaussian distributions, offers the advantage of accounting for total variance and generalized variance beyond just the L_2 norm.

Given new objective $\ell^{\gamma}_{\text{FP-BMA}}(\theta)$ in Eq. 5, the variational parameter θ is updated using the approximate gradient

$$\nabla_\theta \ell^{\gamma}_{\text{FP-BMA}}(\theta) \approx \nabla_\theta \ell(\theta + \Delta\theta_{\text{FP-BMA}}), \tag{7}$$

Table 1: Performances (ACC, ECE, and NLL) of learning from scratch with ResNet18 and modified ViT-B/16†. FP-BMA (VI), FP-BMA (MCMC), and FP-BMA (SWAG) indicate the specific BNN framework combined with FP-BMA. **Bold** highlights the best performance within each BNN framework, while red indicates the overall best performance across all frameworks. FP-BMA achieves superior performance across all BNN frameworks on both CIFAR10 and CIFAR100.

Backbone	ResNet18						ViT-B/16†					
Dataset	CIFAR10			CIFAR100			CIFAR10			CIFAR100		
Method	ACC ↑	ECE ↓	NLL ↓	ACC ↑	ECE ↓	NLL ↓	ACC ↑	ECE ↓	NLL ↓	ACC ↑	ECE ↓	NLL ↓
SGD	83.28$_{\pm 0.49}$	0.058$_{\pm 0.005}$	0.540$_{\pm 0.006}$	50.33$_{\pm 0.62}$	0.123$_{\pm 0.016}$	1.976$_{\pm 0.055}$	81.20$_{\pm 1.31}$	0.050.$_{\pm 0.002}$	0.569$_{\pm 0.027}$	48.66$_{\pm 0.21}$	0.062$_{\pm 0.013}$	1.956$_{\pm 0.021}$
SAM	**87.59**$_{\pm 3.10}$	**0.031**$_{\pm 0.017}$	**0.389**$_{\pm 0.065}$	51.48$_{\pm 0.05}$	0.096$_{\pm 0.026}$	1.873$_{\pm 0.042}$	81.25$_{\pm 0.10}$	**0.020**$_{\pm 0.003}$	**0.550**$_{\pm 0.002}$	54.91$_{\pm 4.20}$	0.053$_{\pm 0.020}$	1.709$_{\pm 0.148}$
FSAM	83.38$_{\pm 0.86}$	0.052$_{\pm 0.003}$	0.540$_{\pm 0.010}$	50.87$_{\pm 1.29}$	0.114$_{\pm 0.008}$	1.963$_{\pm 0.058}$	**81.57**$_{\pm 1.49}$	0.046$_{\pm 0.006}$	0.563$_{\pm 0.036}$	48.75$_{\pm 0.42}$	0.055$_{\pm 0.010}$	1.956$_{\pm 0.003}$
bSAM	84.28$_{\pm 0.32}$	0.051$_{\pm 0.010}$	0.502$_{\pm 0.012}$	**52.55**$_{\pm 0.30}$	**0.087**$_{\pm 0.011}$	**1.802**$_{\pm 0.027}$	80.33$_{\pm 0.88}$	0.037$_{\pm 0.007}$	0.588$_{\pm 0.012}$	**57.75**$_{\pm 0.29}$	**0.040**$_{\pm 0.014}$	**1.573**$_{\pm 0.015}$
VI	82.61$_{\pm 0.51}$	0.067$_{\pm 0.003}$	0.632$_{\pm 0.008}$	51.45$_{\pm 0.32}$	0.037$_{\pm 0.007}$	1.874$_{\pm 0.007}$	75.81$_{\pm 0.88}$	0.027$_{\pm 0.021}$	0.715$_{\pm 0.038}$	48.97$_{\pm 0.20}$	0.037$_{\pm 0.012}$	1.965$_{\pm 0.002}$
FP-BMA (VI)	**85.34**$_{\pm 0.18}$	**0.028**$_{\pm 0.006}$	**0.431**$_{\pm 0.001}$	**54.49**$_{\pm 0.82}$	**0.016**$_{\pm 0.003}$	**1.699**$_{\pm 0.021}$	**76.23**$_{\pm 0.44}$	**0.018**$_{\pm 0.006}$	**0.692**$_{\pm 0.010}$	**51.62**$_{\pm 1.12}$	0.038$_{\pm 0.013}$	**1.884**$_{\pm 0.026}$
MCMC	84.82$_{\pm 0.13}$	0.049$_{\pm 0.001}$	0.523$_{\pm 0.008}$	58.38$_{\pm 0.16}$	0.090$_{\pm 0.002}$	1.742$_{\pm 0.014}$	81.80$_{\pm 0.46}$	0.014$_{\pm 0.003}$	0.542$_{\pm 0.023}$	51.79$_{\pm 0.29}$	0.081$_{\pm 0.001}$	2.068$_{\pm 0.016}$
E-MCMC	85.45$_{\pm 0.27}$	0.037$_{\pm 0.002}$	0.479$_{\pm 0.006}$	60.38$_{\pm 0.21}$	0.074$_{\pm 0.003}$	1.574$_{\pm 0.002}$	81.97$_{\pm 0.49}$	0.034$_{\pm 0.004}$	0.545$_{\pm 0.014}$	50.48$_{\pm 0.13}$	0.068$_{\pm 0.005}$	2.010$_{\pm 0.007}$
FP-BMA (MCMC)	**86.98**$_{\pm 0.19}$	**0.030**$_{\pm 0.004}$	**0.393**$_{\pm 0.001}$	**61.94**$_{\pm 0.37}$	**0.029**$_{\pm 0.003}$	**1.467**$_{\pm 0.006}$	**82.49**$_{\pm 1.95}$	**0.012**$_{\pm 0.003}$	**0.528**$_{\pm 0.067}$	**61.10**$_{\pm 1.44}$	**0.046**$_{\pm 0.005}$	**1.461**$_{\pm 0.067}$
SWAG	88.95$_{\pm 0.09}$	0.044$_{\pm 0.015}$	0.349$_{\pm 0.013}$	59.48$_{\pm 0.19}$	**0.030**$_{\pm 0.002}$	1.594$_{\pm 0.011}$	83.70$_{\pm 0.30}$	0.044$_{\pm 0.011}$	0.493$_{\pm 0.020}$	54.76$_{\pm 2.20}$	0.151$_{\pm 0.025}$	2.008$_{\pm 0.136}$
F-SWAG	89.35$_{\pm 0.19}$	0.028$_{\pm 0.013}$	0.323$_{\pm 0.010}$	60.44$_{\pm 0.20}$	0.074$_{\pm 0.023}$	1.566$_{\pm 0.006}$	83.57$_{\pm 0.41}$	0.046$_{\pm 0.015}$	0.498$_{\pm 0.029}$	56.80$_{\pm 1.44}$	0.061$_{\pm 0.017}$	1.733$_{\pm 0.073}$
FP-BMA (SWAG)	**89.84**$_{\pm 0.30}$	**0.019**$_{\pm 0.002}$	**0.306**$_{\pm 0.006}$	**63.63**$_{\pm 0.60}$	0.052$_{\pm 0.007}$	**1.342**$_{\pm 0.003}$	**84.44**$_{\pm 0.58}$	**0.028**$_{\pm 0.008}$	**0.464**$_{\pm 0.011}$	**57.64**$_{\pm 1.42}$	**0.032**$_{\pm 0.005}$	**1.590**$_{\pm 0.050}$

where the parameter perturbation $\Delta\theta_{\text{FP-BMA}}$ is first computed as:

$$\Delta\theta_{\text{FP-BMA}} = \gamma \frac{F_\theta(\theta)^{-1}\nabla_\theta \ell(\theta)}{\sqrt{\nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1}\nabla_\theta \ell(\theta)}}, \tag{8}$$

using FIM $F_\theta(\theta)=\mathbb{E}_{w,\mathcal{D}}[\nabla_\theta \log q_\theta(w|\mathcal{D})\nabla_\theta \log q_\theta(w|\mathcal{D})^T]$, and then the gradient $\nabla_\theta \ell(\theta)$ is evaluated at $\theta + \Delta\theta_{\text{FP-BMA}}$.

Notably, as the FIM $F_\theta(\theta)$ is defined over variational parameter θ, we obtain the adversarial posterior directly. This allows BNNs to find the flat posterior in a Bayesian principle. We notate our objective as FP-BMA and provide a detailed formula derivation in Appendix A.2.

4.2 Flat Posterior-aware Bayesian Transfer Learning

Additionally, we extend the proposed objective to seek the flat posterior for Bayesian transfer learning. For the given approximate posterior $q_\theta^{\text{pr}}(w|\mathcal{D}^{\text{pr}})$ on source or downstream task $\mathcal{D}^{\text{pr}}$, we set our objective:

$$\ell^\gamma_{\text{FP-BMA}}(\theta) = \min_\theta \max_{d|\theta+\Delta\theta,\theta|\leq\gamma^2} \ell(\theta+\Delta\theta) + \beta \mathrm{D}_{\mathrm{KL}}[q_\theta(w|\mathcal{D}^{\text{ft}})||q_\theta^{\text{pr}}(w|\mathcal{D}^{\text{pr}})] \tag{9}$$

$$\text{s.t.} \;\; d|\theta+\Delta\theta,\theta| = \mathrm{D}_{\mathrm{KL}}\left[q_{\theta+\Delta\theta}(w|\mathcal{D}^{\text{ft}}) \;||\; q_\theta(w|\mathcal{D}^{\text{ft}})\right],$$

where $\mathcal{D}^{\text{ft}}$ is the downstream dataset. Intuitively, this objective replaces the prior distribution of Eq. 6 by the approximate posterior $q_\theta^{\text{pr}}(w|\mathcal{D}^{\text{pr}})$ on source dataset. Notably, *the proposed objective $\ell^\gamma_{\text{FP-BMA}}(\theta)$ can be effective in general transfer learning where the model misspecification (Müller, 2013; Wilson & Izmailov, 2020) exists; the prior is not suitable for the BNNs to be fine-tuned on downstream tasks, and flat parameters have been shown to improve the model's robustness (Kim et al., 2022b; Zhang et al., 2023).* Indeed, we demonstrate that the proposed method achieves superior generalization under distributional shifts in Section 5.3.

For computational efficiency, we adopt a sub-network BNN strategy, focusing training on normalization and last-layer parameters, as explored in prior works (Izmailov et al., 2020; Daxberger et al., 2021b; Sharma et al., 2023). During fine-tuning, we reinitialize the last layer with a Gaussian distribution, $\mathcal{N}(0,\alpha I)$, where α is a hyperparameter to control variance. This approach ensures scalable and stable training by leveraging pre-trained DNNs. The complete FP-BMA procedure is given in Algorithm 1 (Appendix B.4).

5 Experiments

5.1 Learning from scratch

We verify the effectiveness of FP-BMA in improving the performance of BNNs trained from scratch. Specifically, we use Bayesian ResNet18 and a modified ViT-B/16† (Dosovitskiy et al., 2020; Liu et al., 2021; Zhu et al., 2023a) on CIFAR10 and CIFAR100. We adopt the modified ViT-B/16† to address the underfitting issue of ViTs on small datasets. Due to computational constraints in large-scale

Table 2: Downstream task performances (ACC, ECE, and NLL) with ResNet18 and ViT-B/16 pre-trained on IN 1K. **Bold** highlights the best performance within each BNN framework, while red indicates the overall best performance across all frameworks. FP-BMA shows superior performance both on the CIFAR10 and CIFAR100 10-shot, with the sole exception being the ECE on the CIFAR100 10-shot in ResNet18.

Backbone	ResNet18						ViT-B/16					
Dataset	CIFAR10 10-shot			CIFAR100 10-shot			CIFAR10 10-shot			CIFAR100 10-shot		
Method	ACC ↑	ECE ↓	NLL ↓	ACC ↑	ECE ↓	NLL ↓	ACC ↑	ECE ↓	NLL ↓	ACC ↑	ECE ↓	NLL ↓
SGD	$55.52_{\pm0.32}$	$\mathbf{0.062}_{\pm0.006}$	$1.302_{\pm0.020}$	$44.29_{\pm0.83}$	$\mathbf{0.025}_{\pm0.005}$	$2.133_{\pm0.043}$	$84.37_{\pm1.47}$	$0.056_{\pm0.061}$	$0.503_{\pm0.038}$	$68.78_{\pm0.21}$	$0.143_{\pm0.007}$	$1.193_{\pm0.019}$
SAM	$56.54_{\pm2.57}$	$0.129_{\pm0.013}$	$1.354_{\pm0.089}$	$\mathbf{44.51}_{\pm0.07}$	$0.065_{\pm0.007}$	$\mathbf{2.089}_{\pm0.013}$	$84.35_{\pm0.81}$	$\mathbf{0.035}_{\pm0.012}$	$\mathbf{0.486}_{\pm0.023}$	$\mathbf{68.93}_{\pm0.37}$	$0.153_{\pm0.005}$	$1.200_{\pm0.021}$
FSAM	$54.04_{\pm4.11}$	$0.139_{\pm0.010}$	$1.432_{\pm0.068}$	$44.07_{\pm1.21}$	$0.056_{\pm0.005}$	$2.159_{\pm0.064}$	$\mathbf{84.51}_{\pm0.50}$	$0.073_{\pm0.085}$	$0.517_{\pm0.061}$	$68.74_{\pm0.39}$	$\mathbf{0.110}_{\pm0.007}$	$\mathbf{1.166}_{\pm0.024}$
bSAM	$\mathbf{56.56}_{\pm1.18}$	$0.083_{\pm0.006}$	$\mathbf{1.280}_{\pm0.027}$	$43.93_{\pm0.48}$	$0.060_{\pm0.003}$	$2.167_{\pm0.026}$	$82.85_{\pm2.10}$	$0.113_{\pm0.008}$	$0.583_{\pm0.062}$	$68.42_{\pm0.40}$	$0.148_{\pm0.019}$	$1.219_{\pm0.031}$
MOPED	$57.29_{\pm1.20}$	$0.093_{\pm0.006}$	$1.297_{\pm0.045}$	$44.30_{\pm0.42}$	$\mathbf{0.047}_{\pm0.006}$	$2.127_{\pm0.005}$	$84.50_{\pm1.36}$	$\mathbf{0.023}_{\pm0.009}$	$0.474_{\pm0.038}$	$68.80_{\pm0.77}$	$0.111_{\pm0.001}$	$1.165_{\pm0.029}$
FP-BMA (VI)	$\mathbf{64.98}_{\pm1.37}$	$\mathbf{0.016}_{\pm0.007}$	$\mathbf{0.997}_{\pm0.046}$	$\mathbf{49.09}_{\pm1.38}$	$0.071_{\pm0.004}$	$\mathbf{1.893}_{\pm0.036}$	$\mathbf{87.56}_{\pm1.10}$	$0.044_{\pm0.012}$	$\mathbf{0.397}_{\pm0.026}$	$\mathbf{71.37}_{\pm0.36}$	$\mathbf{0.060}_{\pm0.007}$	$\mathbf{1.023}_{\pm0.012}$
MCMC	$56.31_{\pm1.27}$	$0.083_{\pm0.003}$	$1.305_{\pm0.063}$	$44.28_{\pm0.95}$	$0.021_{\pm0.002}$	$2.155_{\pm0.038}$	$83.93_{\pm1.33}$	$0.069_{\pm0.010}$	$0.523_{\pm0.039}$	$66.48_{\pm1.18}$	$0.077_{\pm0.011}$	$1.224_{\pm0.044}$
PTL	$57.26_{\pm1.44}$	$0.116_{\pm0.003}$	$1.345_{\pm0.004}$	$43.00_{\pm1.05}$	$0.120_{\pm0.006}$	$2.383_{\pm0.062}$	$\mathbf{85.76}_{\pm1.37}$	$0.080_{\pm0.014}$	$0.482_{\pm0.027}$	$65.52_{\pm2.45}$	$\mathbf{0.056}_{\pm0.006}$	$1.260_{\pm0.095}$
E-MCMC	$56.69_{\pm2.14}$	$0.142_{\pm0.004}$	$1.266_{\pm0.054}$	$41.57_{\pm0.04}$	$0.046_{\pm0.012}$	$2.370_{\pm0.175}$	$83.91_{\pm1.16}$	$0.333_{\pm0.010}$	$0.877_{\pm0.044}$	$63.40_{\pm0.01}$	$0.280_{\pm0.008}$	$1.655_{\pm0.024}$
FP-BMA (MCMC)	$\mathbf{57.49}_{\pm0.64}$	$\mathbf{0.039}_{\pm0.00.}$	$\mathbf{1.248}_{\pm0.048}$	$\mathbf{45.72}_{\pm0.56}$	$\mathbf{0.016}_{\pm0.003}$	$\mathbf{2.062}_{\pm0.050}$	$84.82_{\pm1.84}$	$\mathbf{0.051}_{\pm0.018}$	$\mathbf{0.449}_{\pm0.048}$	$\mathbf{68.73}_{\pm1.09}$	$0.061_{\pm0.004}$	$\mathbf{1.117}_{\pm0.042}$
SWAG	$56.31_{\pm0.60}$	$0.094_{\pm0.013}$	$1.315_{\pm0.056}$	$44.14_{\pm1.28}$	$\mathbf{0.034}_{\pm0.010}$	$2.161_{\pm0.058}$	$83.51_{\pm2.22}$	$0.022_{\pm0.015}$	$0.510_{\pm0.072}$	$68.72_{\pm0.45}$	$0.065_{\pm0.005}$	$1.136_{\pm0.014}$
F-SWAG	$57.65_{\pm1.20}$	$0.075_{\pm0.003}$	$1.249_{\pm0.038}$	$46.09_{\pm0.44}$	$0.062_{\pm0.006}$	$2.089_{\pm0.002}$	$83.87_{\pm1.28}$	$0.013_{\pm0.005}$	$0.492_{\pm0.040}$	$68.84_{\pm0.77}$	$0.076_{\pm0.012}$	$1.137_{\pm0.020}$
FP-BMA (SWAG)	$\mathbf{61.79}_{\pm4.34}$	$\mathbf{0.026}_{\pm0.004}$	$\mathbf{1.214}_{\pm0.119}$	$\mathbf{47.45}_{\pm0.60}$	$0.055_{\pm0.018}$	$\mathbf{2.044}_{\pm0.022}$	$\mathbf{86.81}_{\pm0.78}$	$\mathbf{0.010}_{\pm0.003}$	$\mathbf{0.399}_{\pm0.034}$	$\mathbf{70.10}_{\pm0.18}$	$\mathbf{0.045}_{\pm0.015}$	$\mathbf{1.063}_{\pm0.023}$

Table 3: Downstream task accuracy with ResNet50 and ViT-B/16 pre-trained on IN 1K. **Bold** and underline denote best and second best performance each. FP-BMA demonstrates superior performance across all 16-shot datasets.

Backbone	ResNet50					ViT-B/16				
Method	EuroSAT	Flowers102	Pets	UCF101	Avg	EuroSAT	Flowers102	Pets	UCF101	Avg
SGD	$86.75_{\pm1.47}$	$93.16_{\pm0.27}$	$89.95_{\pm0.51}$	$66.34_{\pm0.59}$	$84.05_{\pm0.33}$	$81.25_{\pm1.03}$	$91.24_{\pm0.83}$	$88.68_{\pm0.92}$	$68.64_{\pm0.51}$	$82.45_{\pm0.56}$
SAM	$87.85_{\pm0.49}$	$94.80_{\pm0.17}$	$90.23_{\pm0.78}$	$70.40_{\pm0.76}$	$85.82_{\pm0.25}$	$82.53_{\pm0.65}$	$\underline{93.08}_{\pm0.87}$	$\underline{90.66}_{\pm0.74}$	$\underline{70.66}_{\pm1.03}$	$\underline{84.23}_{\pm0.60}$
SWAG	$88.97_{\pm1.56}$	$93.27_{\pm0.15}$	$89.95_{\pm0.46}$	$66.41_{\pm0.30}$	$84.65_{\pm0.37}$	$81.62_{\pm0.66}$	$91.21_{\pm0.91}$	$88.67_{\pm0.42}$	$67.65_{\pm0.45}$	$82.29_{\pm0.31}$
F-SWAG	$90.03_{\pm1.08}$	$\underline{94.84}_{\pm0.26}$	$\underline{90.12}_{\pm0.57}$	$\underline{70.00}_{\pm0.87}$	$\underline{86.25}_{\pm0.19}$	$82.72_{\pm0.49}$	$92.93_{\pm0.93}$	$90.60_{\pm0.55}$	$68.67_{\pm0.39}$	$83.73_{\pm0.35}$
MOPED	$85.21_{\pm3.14}$	$92.15_{\pm0.73}$	$89.25_{\pm0.61}$	$65.85_{\pm0.99}$	$83.11_{\pm0.86}$	$\underline{83.97}_{\pm0.49}$	$91.71_{\pm0.87}$	$89.90_{\pm0.54}$	$69.66_{\pm0.53}$	$83.81_{\pm0.51}$
PTL	$\underline{90.01}_{\pm0.39}$	$92.55_{\pm0.53}$	$89.43_{\pm0.41}$	$65.00_{\pm1.24}$	$84.25_{\pm0.30}$	$83.76_{\pm0.61}$	$88.43_{\pm1.27}$	$88.54_{\pm0.53}$	$60.38_{\pm1.84}$	$80.28_{\pm0.03}$
FP-BMA	$\mathbf{90.16}_{\pm1.04}$	$\mathbf{95.85}_{\pm1.26}$	$\mathbf{90.23}_{\pm0.58}$	$\mathbf{71.57}_{\pm0.27}$	$\mathbf{86.95}_{\pm0.65}$	$\mathbf{84.60}_{\pm0.25}$	$\mathbf{94.15}_{\pm0.80}$	$\mathbf{91.30}_{\pm0.25}$	$\mathbf{72.63}_{\pm1.12}$	$\mathbf{85.67}_{\pm0.14}$

models, we apply variational distributions to the parameters of normalization and last layers. For comparison, we consider SGD, SAM (Foret et al., 2020), and FSAM (Kim et al., 2022a) seeking flat minima in DNNs. For the training of BNNs, we consider SWAG, VI, F-SWAG (Nguyen et al., 2023), bSAM (Möllenhoff & Khan, 2022), and E-MCMC (Li & Zhang, 2023), which utilizes SGLD. For fair comparison, we use the same BNN architecture employed for FP-BMA.

Table 1 shows that FP-BMA consistently improves performances when integrated with VI, MCMC, and SWAG. Also, The FP-BMA leads to superior performances compared to other baselines of SGD, SAM, FSAM, and bSAM. Additional experimental details are provided in Appendix B.1.

5.2 Bayesian Transfer Learning

Finetuning on CIFARs We validate the effectiveness of the FP-BMA on a transfer learning task. We first adopt RN18 and ViT-B/16 pre-trained on ImageNet (IN) 1K (Russakovsky et al., 2015) as a backbone. The pre-trained models are fine-tuned on CIFAR10 and CIFAR100 10-shot, using 10 data instances per class.

For comparison, we consider the following Bayesian transfer learning methods: MOPED (Krishnan et al., 2020) and Pre-Train Your Loss (PTL) (Shwartz-Ziv et al., 2022). We describe additional configurations in Appendix B.2.

Table 4: Downstream task accuracy of CLIP with visual encoder, RN50 and ViT-B/16. **Bold** and underline denote best and second best performance each. SA-BMA shows superior performance in average over five datasets.

Backbone	Method	IN	IN-V2	IN-R	IN-A	IN-S	Avg
RN50	Zero-Shot	$59.83_{\pm0.00}$	$52.89_{\pm0.00}$	$60.73_{\pm0.00}$	$\underline{23.25}_{\pm0.00}$	$35.45_{\pm0.00}$	$46.43_{\pm0.00}$
	SGD	$61.70_{\pm0.01}$	$54.31_{\pm0.01}$	$60.87_{\pm0.01}$	$22.74_{\pm0.01}$	$\underline{35.68}_{\pm0.00}$	$47.06_{\pm0.01}$
	SAM	$61.73_{\pm0.01}$	$\underline{54.35}_{\pm0.01}$	$60.86_{\pm0.01}$	$22.76_{\pm0.01}$	$35.67_{\pm0.00}$	$47.07_{\pm0.01}$
	SWAG	$\underline{61.77}_{\pm0.22}$	$54.10_{\pm0.19}$	$\mathbf{61.25}_{\pm0.21}$	$\mathbf{23.25}_{\pm0.08}$	$35.55_{\pm0.27}$	$\underline{47.18}_{\pm0.19}$
	SA-BMA	$\mathbf{63.33}_{\pm0.92}$	$\mathbf{55.06}_{\pm0.79}$	$\underline{61.14}_{\pm0.37}$	$22.78_{\pm0.68}$	$\mathbf{35.82}_{\pm0.11}$	$\mathbf{47.63}_{\pm0.17}$
ViT-B/16	Zero-Shot	$68.33_{\pm0.00}$	$61.91_{\pm0.00}$	$77.71_{\pm0.00}$	$49.93_{\pm0.00}$	$48.22_{\pm0.00}$	$61.22_{\pm0.00}$
	SGD	$69.97_{\pm0.00}$	$62.97_{\pm0.01}$	$78.05_{\pm0.00}$	$50.31_{\pm0.02}$	$48.76_{\pm0.00}$	$62.01_{\pm0.00}$
	SAM	$70.01_{\pm0.01}$	$63.03_{\pm0.02}$	$78.03_{\pm0.01}$	$50.37_{\pm0.00}$	$48.75_{\pm0.00}$	$62.04_{\pm0.00}$
	SWAG	$\underline{70.11}_{\pm0.02}$	$\underline{63.44}_{\pm0.06}$	$\mathbf{78.33}_{\pm0.03}$	$\mathbf{50.55}_{\pm0.02}$	$\underline{48.95}_{\pm0.01}$	$\underline{62.28}_{\pm0.02}$
	SA-BMA	$\mathbf{72.41}_{\pm0.33}$	$\mathbf{64.85}_{\pm0.11}$	$\underline{78.14}_{\pm0.31}$	$\underline{50.52}_{\pm0.25}$	$\mathbf{49.25}_{\pm0.03}$	$\mathbf{63.03}_{\pm0.04}$

Table 2 shows FP-BMA with diverse BNN frameworks consistently outperforms existing baselines in terms of both accuracy and uncertainty quantification. Unlike scratch learning, FP-BMA (VI) outperforms FP-BMA (SWAG) in few-shot image classification tasks. This can be attributed to the nature of few-shot tasks, where VI, which only learns a diagonal covariance, is less prone to underfitting due to the limited amount of data.

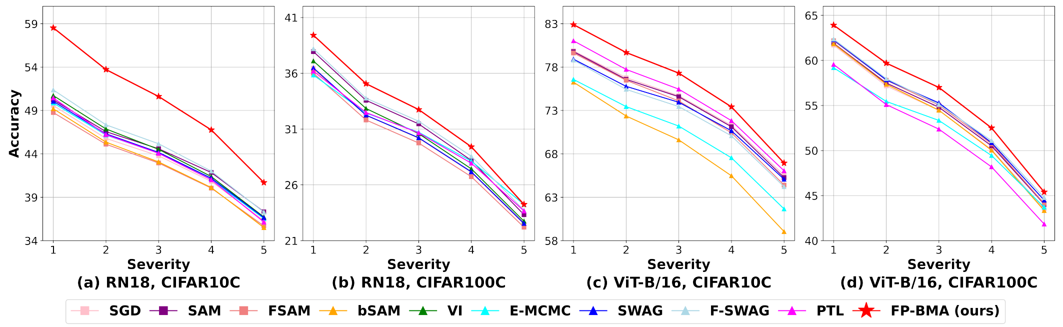


Figure 2: Accuracy under distributional shift. We evaluate the accuracy of RN18 and ViT-B/16 models trained on CIFAR10 and CIFAR100 10-shot across all severity levels of CIFAR10C and CIFAR100C. FP-BMA consistently outperforms all baseline methods across all levels of corruption.

Fine-tuning on fine-grained image classification tasks Furthermore, we confirm the effectiveness of FP-BMA on general fine-grained image classification tasks, including EuroSAT (Helber et al., 2019), Flowers102 (Nilsback & Zisserman, 2008), Pets (Parkhi et al., 2012), and UCF101 (Soomro et al., 2012). All experiments were conducted using a 16-shot setting across all datasets. From this point forward, we perform all experiments using FP-BMA with SWAG only. Table 3 shows that the FP-BMA achieves the best accuracy. Table 11 (Appendix B.6) shows that the FP-BMA achieves the best NLL, as well. This implies that FP-BMA seeking the flat posterior during fine-tuning procedure is effective in improving the performance of Bayesian transfer learning.

Fine-tuning with CLIP We also show the effectiveness of FP-BMA on the pre-trained vision language models. We fine-tune only the last layer of the CLIP visual encoder on the IN 1K 16-shot dataset. Then, we evaluate the trained model on IN and its variants—IN-V2 (Recht et al., 2019), IN-R (Hendrycks et al., 2021a), IN-A (Hendrycks et al., 2021b), and IN-S (Wang et al., 2019)—following the protocols outlined in Radford et al. (2021); Zhu et al. (2023b). Table 4 shows that FP-BMA outperforms baselines on IN set. Also, FP-BMA shows superior or comparable accuracy on out-of-distribution datasets, representing the effectiveness of robustness.

5.3 Robustness on Distribution Shift

We evaluate the trained models on CIFAR10 and CIFAR100 10-shots using the corrupted datasets CIFAR10C and CIFAR100C (Hendrycks & Dietterich, 2019) to demonstrate the robustness of FP-BMA.

Figure 2 presents the accuracy on the corrupted datasets CIFAR10C and CIFAR100C (Hendrycks & Dietterich, 2019), demonstrating that FP-BMA outperforms baselines on corrupted datasets across all corruption levels. FP-BMA consistently outperforms all baselines in NLL, as shown in Figure 4 (Appendix B.7). Detailed results are provided in Appendix B.7.

The results on IN variants in Table 4 and the corrupted datasets in Figure 2 show that FP-BMA enhances the robustness of trained BNNs under distribution shifts, suggesting that the Flat Posterior-aware Bayesian Transfer Learning scheme with FP-BMA effectively improves robustness.

5.4 Flatness Analysis

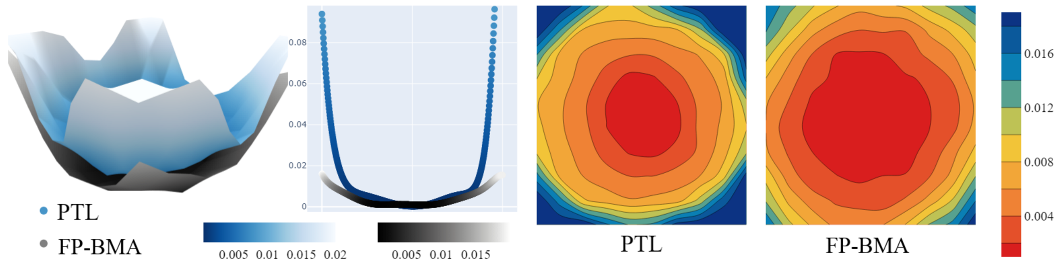


Figure 3: Comparison of the loss surfaces of FP-BMA (grey) and PTL (light blue) models. The comparison of loss surface shows that FP-BMA allows the posterior to be placed on a lower and flatter loss surface compared to that of PTL.

Table 5: Hessian analysis on ResNet18 trained with CIFAR10 10-shot. FP-BMA shows the lowest score on both λ_1 and λ_1/λ_5, proving it leads the model to flatter minima.

	SGD	SAM	FSAM	bSAM	MOPED	PTL	E-MCMC	SWAG	F-SWAG	**FP-BMA**
$\lambda_1 \downarrow$	559.62	381.74	561.15	532.74	686.90	559.16	540.83	602.34	362.33	**275.21**
$\lambda_1/\lambda_5 \downarrow$	2.59	2.23	2.24	2.09	2.41	2.23	1.98	2.13	2.44	**1.69**

We analyze whether FP-BMA encourages the posterior of BNNs to lie in a flatter loss basin. Using ResNet18 trained on CIFAR10 with 10-shot, we compare weight samples from the approximate posterior obtained by FP-BMA and PTL and compare the Hessian's eigenvalue of model.

Figure 3 presents different views of loss surface using sampled weights of FP-BMA and PTL. This result confirms that the posterior of FP-BMA is placed on a flatter loss basin with lower loss.

6 Related Works

6.1 Flatness and BNN

Recent works have suggested flat-seeking optimizers combined with BNN. First, SWAG (Maddox et al., 2019) implicitly approximated posterior toward flatter optima based on SWA (Izmailov et al., 2018). However, SWAG can fail to find flat minima, leading to limited improvement in generalization, as shown in Section 3.2. bSAM (Möllenhoff & Khan, 2022) showed that SAM can be interpreted as a relaxation of the Bayes and quantified uncertainty with SAM. Yet, bSAM only focused on uncertainty quantification by simply modifying Adam-based SAM (Khan et al., 2018), not newly considering the parametric geometry for perturbation. Moreover, scaling the variance with the number of data points hampers the direct implementation of bSAM in few-shot settings. SA-BNN (Nguyen et al., 2023) proposed a sharpness-aware posterior derived directly from the variational objective and proved the effectiveness experimentally and theoretically. However, they simply employ the L2 norm to calculate the perturbation of SAM without considering the difference between the nature of DNN and BNN. Moreover, in contrast to FP-BMA, SA-BNN did not take into account the prior, which is a fundamental component of BNNs, in its pursuit of flatness. On the other hand, E-MCMC (Li & Zhang, 2023) proposed an efficient MCMC algorithm capable of effectively sampling the posterior within a flat basin by removing the nested chain of Entropy-SGD and Entropy-SGLD. However, E-MCMC necessitates a guidance model, which doubles the parameters and heavily hinders its employment over large-scale models. FP-BMA is the first approach to explicitly promote flat posteriors within a rigorous Bayesian framework, providing a principled way to enhance robustness and generalization.

6.2 Bayesian Transfer Learning

There are several works on performing transfer learning on BNN with prior. PTL (Shwartz-Ziv et al., 2022) constructs BNN by learning closed-form posterior approximation of the pre-trained model on the source task and uses it as a prior for the downstream task after scaling. The work requires additional training on the source task, making it restrictive when accessing the source task dataset is impossible. MOPED (Krishnan et al., 2020) employs pre-trained BNN as a prior for VI based on the empirical Bayes method. Using pre-trained DNN, MOPED enhances accessibility to BNN; however, it is only applicable to Mean-field VI. Non-parametric transfer learning (Lee et al., 2024) suggested adopting non-parametric learning to make posterior flexible in terms of distribution shift. The proposed Flat Posterior-aware Bayesian Transfer Learning utilizes a pre-trained model as a prior, improving robustness to model misspecification by promoting a flat posterior.

7 Conclusion

This study shows the limitations of BNNs in capturing the flatness, which is crucial for generalization ability. We also show that BMA can fail to yield optimal results without explicitly considering flatness. To address this issue, we introduce Flat Posterior-aware Bayesian Model Averaging (FP-BMA), which seeks to find a flat posterior by capturing flatness in the parameter space. FP-BMA is the generalized version of existing sharpness-aware optimizers for DNN and aligns with the intrinsic nature of BNN. We further propose a Flat Posterior-aware Bayesian Transfer Learning scheme, which effectively enhances robustness against model misspecification, combined with FP-BMA. Through extensive experiments, we demonstrate that FP-BMA significantly enhances the generalization performance of BNNs in diverse scenarios. Our work highlights the importance of flatness in posterior approximations and provides a practical solution to improve the predictive robustness and accuracy of BNNs.

ACKNOWLEDGEMENTS

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT)(RS-2024-00457216).

REFERENCES

Carlo Baldassi, Alessandro Ingrosso, Carlo Lucibello, Luca Saglietti, and Riccardo Zecchina. Subdominant dense clusters allow for simple learning and high computational performance in neural networks with discrete synapses. *Physical review letters*, 115(12):128101, 2015.

Carlo Baldassi, Christian Borgs, Jennifer T Chayes, Alessandro Ingrosso, Carlo Lucibello, Luca Saglietti, and Riccardo Zecchina. Unreasonable effectiveness of learning neural networks: From accessible states and robust ensembles to basic algorithmic schemes. *Proceedings of the National Academy of Sciences*, 113(48):E7655–E7662, 2016.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pp. 1613–1622. PMLR, 2015.

Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124018, 2019.

Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *International conference on machine learning*, pp. 1683–1691. PMLR, 2014.

Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig. Laplace redux-effortless bayesian deep learning. *Advances in Neural Information Processing Systems*, 34:20089–20103, 2021a.

Erik Daxberger, Eric Nalisnick, James U Allingham, Javier Antorán, and José Miguel Hernández-Lobato. Bayesian deep learning via subnetwork inference. In *International Conference on Machine Learning*, pp. 2510–2521. PMLR, 2021b.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Gintare Karolina Dziugaite and Daniel Roy. Entropy-sgd optimizes the prior of a pac-bayes bound: Generalization properties of entropy-sgd and data-dependent priors. In *International Conference on Machine Learning*, pp. 1377–1386. PMLR, 2018.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

Tiago M Fragoso, Wesley Bertoli, and Francisco Louzada. Bayesian model averaging: A systematic review and conceptual classification. *International Statistical Review*, 86(1):1–28, 2018.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

James E Gentle. Matrix algebra. *Springer texts in statistics, Springer, New York, NY, doi*, 10:978–0, 2007.

Alex Graves. Practical variational inference for neural networks. *Advances in neural information processing systems*, 24, 2011.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 8340–8349, 2021a.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15262–15271, 2021b.

Geoffrey E Hinton and Drew Van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, pp. 5–13, 1993.

Sepp Hochreiter and Jürgen Schmidhuber. Simplifying neural nets by discovering flat minima. *Advances in neural information processing systems*, 7, 1994.

Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural computation*, 9(1):1–42, 1997.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. pmlr, 2015.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

Pavel Izmailov, Wesley J Maddox, Polina Kirichenko, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Subspace inference for bayesian deep learning. In *Uncertainty in Artificial Intelligence*, pp. 1169–1179. PMLR, 2020.

Stanislaw Jastrzebski, Maciej Szymczak, Stanislav Fort, Devansh Arpit, Jacek Tabor, Kyunghyun Cho, and Krzysztof Geras. The break-even point on optimization trajectories of deep neural networks. *arXiv preprint arXiv:2002.09572*, 2020.

Sanyam Kapoor, Wesley J Maddox, Pavel Izmailov, and Andrew G Wilson. On uncertainty, tempering, and data augmentation in bayesian classification. *Advances in Neural Information Processing Systems*, 35:18211–18225, 2022.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

Mohammad Khan, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava. Fast and scalable bayesian deep learning by weight-perturbation in adam. In *International conference on machine learning*, pp. 2611–2620. PMLR, 2018.

Minyoung Kim, Da Li, Shell X Hu, and Timothy Hospedales. Fisher sam: Information geometry and sharpness aware minimisation. In *International Conference on Machine Learning*, pp. 11148–11161. PMLR, 2022a.

Taero Kim, Subeen Park, Sungjun Lim, Yonghan Jung, Krikamol Muandet, and Kyungwoo Song. Sufficient invariant learning for distribution shift. *arXiv preprint arXiv:2210.13533*, 2022b.

Ranganath Krishnan, Mahesh Subedar, and Omesh Tickoo. Specifying weight priors in bayesian deep neural networks with empirical bayes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 4477–4484, 2020. URL https://ojs.aaai.org/index.php/AAAI/article/view/5875.

Agustinus Kristiadi, Runa Eschenhagen, and Philipp Hennig. Posterior refinement improves sample efficiency in bayesian neural networks. *Advances in Neural Information Processing Systems*, 35: 30333–30346, 2022a.

Agustinus Kristiadi, Matthias Hein, and Philipp Hennig. Being a bit frequentist improves bayesian neural networks. In *International Conference on Artificial Intelligence and Statistics*, pp. 529–545. PMLR, 2022b.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Hyungi Lee, Giung Nam, Edwin Fong, and Juho Lee. Enhancing transfer learning with flexible nonparametric posterior sampling. *arXiv preprint arXiv:2403.07282*, 2024.

Bolian Li and Ruqi Zhang. Entropy-mcmc: Sampling from flat basins with ease. In *NeurIPS 2023 Workshop on Symmetry and Geometry in Neural Representations*, 2023.

Yahui Liu, Enver Sangineto, Wei Bi, Nicu Sebe, Bruno Lepri, and Marco Nadai. Efficient training of visual transformers with small datasets. *Advances in Neural Information Processing Systems*, 34: 23818–23830, 2021.

David JC MacKay. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Thomas Möllenhoff and Mohammad Emtiyaz Khan. Sam as an optimal relaxation of bayes. *arXiv preprint arXiv:2210.01620*, 2022.

Ulrich K Müller. Risk of bayesian inference in misspecified models, and the sandwich covariance matrix. *Econometrica*, 81(5):1805–1849, 2013.

Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. *Advances in neural information processing systems*, 30, 2017.

Van-Anh Nguyen, Tung-Long Vuong, Hoang Phan, Thanh-Toan Do, Dinh Phung, and Trung Le. Flat seeking bayesian neural networks. *arXiv preprint arXiv:2302.02713*, 2023.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pp. 722–729. IEEE, 2008.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 3498–3505. IEEE, 2012.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Alexandre Rame, Matthieu Kirchmeyer, Thibaud Rahier, Alain Rakotomamonjy, Patrick Gallinari, and Matthieu Cord. Diverse weight averaging for out-of-distribution generalization. *Advances in Neural Information Processing Systems*, 35:10821–10836, 2022.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International conference on machine learning*, pp. 5389–5400. PMLR, 2019.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *6th International Conference on Learning Representations, ICLR 2018-Conference Track Proceedings*, volume 6. International Conference on Representation Learning, 2018.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

Mrinank Sharma, Sebastian Farquhar, Eric Nalisnick, and Tom Rainforth. Do bayesian neural networks need to be fully stochastic? In *International Conference on Artificial Intelligence and Statistics*, pp. 7694–7722. PMLR, 2023.

Ravid Shwartz-Ziv, Micah Goldblum, Hossein Souri, Sanyam Kapoor, Chen Zhu, Yann LeCun, and Andrew Gordon Wilson. Pre-train your loss: Easy bayesian transfer learning with informative priors. *arXiv preprint arXiv:2205.10279*, 2022.

Avram Sidi. *Vector extrapolation methods with applications*. SIAM, 2017.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. *Advances in Neural Information Processing Systems*, 32, 2019.

Larry Wasserman. Bayesian model selection and model averaging. *Journal of mathematical psychology*, 44(1):92–107, 2000.

Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688, 2011.

Florian Wenzel, Kevin Roth, Bastiaan S Veeling, Jakub Świątkowski, Linh Tran, Stephan Mandt, Jasper Snoek, Tim Salimans, Rodolphe Jenatton, and Sebastian Nowozin. How good is the bayes posterior in deep neural networks really? *arXiv preprint arXiv:2002.02405*, 2020.

Hermann Weyl. Das asymptotische verteilungsgesetz der eigenwerte linearer partieller differentialgleichungen (mit einer anwendung auf die theorie der hohlraumstrahlung). *Mathematische Annalen*, 71(4):441–479, 1912.

Andrew G Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. *Advances in neural information processing systems*, 33:4697–4708, 2020.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pp. 23965–23998. PMLR, 2022.

Peter Wynn. Acceleration techniques for iterated vector and matrix problems. *Mathematics of Computation*, 16(79):301–322, 1962.

Zhe Zeng and Guy Van den Broeck. Collapsed inference for bayesian deep learning. *Advances in Neural Information Processing Systems*, 36, 2024.

Xingxuan Zhang, Renzhe Xu, Han Yu, Yancheng Dong, Pengfei Tian, and Peng Cui. Flatness-aware minimization for domain generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5189–5202, 2023.

Haoran Zhu, Boyuan Chen, and Carter Yang. Understanding why vit trains badly on small datasets: an intuitive perspective. *arXiv preprint arXiv:2302.03751*, 2023a.

Yao Zhu, Yuefeng Chen, Wei Wang, Xiaofeng Mao, Yue Wang, Zhigang Li, Jindong Wang, Xiangyang Ji, et al. Enhancing few-shot clip with semantic-aware fine-tuning. *arXiv preprint arXiv:2311.04464*, 2023b.

A Proof and Derivation

A.1 Proof of Theorem 1

The derivation of Theorem 1 can be straightforward using Wely's inequality (Weyl, 1912). We assume M model $w_m, m = 1, .., M$, whose Hessian matrices H_{f_m} (defined in Eq. 3) are Hermitian. $w_{\text{WA}} = 1/M \sum_{m=1}^{M} w_m$ is simple weight averaging and the Hessian of w_{WA}, $H_{f_{\text{WA}}}$, also be a Hermitian matrix. Weyl's inequality (Theorem 2) is known to bound the eigenvalues of Hermitian matrices.

Theorem 2. *(**Weyl's Inequality**) For Hermitian matrices $C_m \in \mathbb{C}^{p\times p}$, $k, l = 1, ..., p$,*

$$\lambda_{k+l-1}(C_i + C_j) \leq \lambda_k(C_i) + \lambda_l(C_j) \leq \lambda_{k+l-N}(C_i + C_j). \tag{10}$$

Let $k = 1$ and $l = 1$, then Eq. 10 can be written as:

$$\lambda_1(C_i + C_j) \leq \lambda_1(C_i) + \lambda_1(C_j).$$

As we have M Hermitian matrices, it can be expanded as:

$$\lambda_1(\frac{1}{M}\sum_{m=1}^{M} H_{f_m}) \leq \frac{1}{M}\sum_{m=1}^{M} \lambda_1(H_{f_m}). \tag{11}$$

One the other hand, we can let $(k, l) = \{(1, p), (p, 1)\}$ and rewrite the Eq. 10 as:

$$\max\{\lambda_1(C_i) + \lambda_p(C_j), \lambda_p(C_i) + \lambda_1(C_j)\} \leq \lambda_1(C_i + C_j).$$

Again, set M Hermitian matrices we have, it can be expanded as:

$$\max\left(\left\{\frac{1}{M}\left(\lambda_1(H_{f_m}) + \sum_{\substack{n=1\\n\neq m}}^{M} \lambda_p(H_{f_n})\right)\right\}_{m=1}^{M}\right) \leq \lambda_1(\frac{1}{M}\sum_{m=1}^{M} H_{f_m}). \tag{12}$$

By combining Eq. 11 with Eq. 12 and substituting λ_1 to λ_{max} and λ_p to λ_{min}, the flatness of averaged weight parameter is bounded as:

$$\max\left(\left\{\frac{1}{M}\left(\lambda_{\text{max}}(H_{f_m}) + \sum_{\substack{n=1\\n\neq m}}^{M} \lambda_{\text{min}}(H_{f_n})\right)\right\}_{m=1}^{M}\right) \leq \lambda_{\text{max}}(\frac{1}{M}\sum_{m=1}^{M} H_{f_m}) \leq \frac{\sum_{m=1}^{M}\lambda_{\text{max}}(H_{f_m})}{M}. \tag{13}$$

However, Eq 13, as a bound for weight averaging (WA), cannot be directly applied to BMA, which marginalizes diverse predictions. To bridge this gap, we leverage Lemma 1. which characterized the close relation between WA and BMA (Izmailov et al., 2018; Wortsman et al., 2022; Rame et al., 2022).

Lemma 1. *((Rame et al., 2022)) Given predictions of model $f_m(\cdot)$ parameterized by w_m, those of weight averaged model f_{WA} parameterized by $w_{WA} = \frac{1}{M}\sum_{m=1}^{M} w_m$, those of BMA f_{BMA}, and arbitrary twice differentiable loss function $\ell(\cdot)$, let $\Delta = \|f_{BMA}(x) - f_{WA}(x)\|_2$. Then, $\forall(x, y)$*

$$\ell(f_{WA}(x), y) = \ell(f_{BMA}(x), y) + O(\Delta).$$

Lemma 1 shows that the predictions of BMA can be approximated with those of WA linearly. The error term is discarded in the process of obtaining the Hessian:

$$H_{f_{\text{WA}}} = H_{f_{\text{BMA}}} \tag{14}$$

By putting Eq. 14 into Eq. 13, it leads to Theorem 1.

A.2 Derivation of Bayesian Flat-Seeking Optimizer

A.2.1 Setting

Let model parameter $w \subseteq \mathbb{R}^p$ and $w \sim \mathcal{N}(\mu, \Sigma)$. While fully-factorized or mean-field covariance is de facto in Bayesian Deep Learning, it cannot capitalize on strong points of Bayesian approach. Inspired from SWAG, we approximate covariance combining diagonal covariance $\sigma \subseteq \mathbb{R}^p$ and low-rank matrix $L \subseteq \mathbb{R}^{p\times K}$ with low-rank component K. Then, we can simply sample $w = \mu + \frac{1}{\sqrt{2}}(\sigma z_1 + L z_2)$, where $z_1 \sim \mathcal{N}(0, I_p)$ and $z_2 \sim \mathcal{N}(0, I_K)$ where p, K denotes the number of parameter, low-rank component, respectively. We treat flattened μ, σ, and L, and concatenate as $\theta = \text{Concat}(\mu; \sigma; L)$.

A.2.2 Objective function

We compose our objective function with probabilistic weight, using KL Divergence as a metric to compare between two weights.

$$\ell^{\gamma}_{\text{FP-BMA}}(\theta) = \max_{d|\theta+\Delta\theta,\theta|\le\gamma^2} \ell(\theta+\Delta\theta) + \beta \mathrm{D}_{\mathrm{KL}}(p_\theta(w|\mathcal{D})||p(w)) \tag{15}$$

$$\text{s.t. } d|\theta+\Delta\theta,\theta| = \mathrm{D}_{\mathrm{KL}}\left[p_{\theta+\Delta\theta}(w|\mathcal{D})||p_\theta(w|\mathcal{D})\right]. \tag{16}$$

A.2.3 Optimization

From KL Divergence to Fisher Information Matrix We can consider three options of perturbation on mean and covariance parameters of w: 1) Perturbation on mean, 2) perturbation on mean and diagonal variance, 3) Perturbation on mean and whole covariance. All of them can be approximated to Fisher Information Matrix. Here, we show the relation between KLD and FIM considering the probation option 3.

Following FSAM, we deal with parameterized and conditioned as same notation:

$$p_{\theta+\Delta\theta}(w|\mathcal{D}) = p(w|\mathcal{D}, \theta+\Delta\theta).$$

By definition of KL divergence, we rewrite Eq. 16 as:

$$\mathrm{D}_{\mathrm{KL}}[p(w|\mathcal{D},\theta+\Delta\theta)||p(w|\mathcal{D},\theta)] = \int_w p(w|\mathcal{D},\theta+\Delta\theta)\ \log\frac{p(w|\mathcal{D},\theta+\Delta\theta)}{p(w|\mathcal{D},\theta)}dw. \tag{17}$$

In Eq. 17, we apply first-order Taylor Expansion:

$$\begin{aligned} p(w|\mathcal{D},\theta+\Delta\theta) &\approx p(w|\mathcal{D},\theta) + \nabla_\theta p(w|\mathcal{D},\theta)^T\Delta\theta. \\ \log p(w|\mathcal{D},\theta+\Delta\theta) &\approx \log p(w|\mathcal{D},\theta) + \nabla_\theta \log p(w|\mathcal{D},\theta)^T\Delta\theta. \end{aligned} \tag{18}$$

Substitute right terms of Eq. 17 with Eq. 18:

$$\begin{aligned} &\int_w p(w|\mathcal{D},\theta+\Delta\theta)\ \log\frac{p(w|\mathcal{D},\theta+\Delta\theta)}{p(w|\mathcal{D},\theta)}dw \\ &= \int_w \big(p(w|\mathcal{D},\theta) + \Delta\theta^T\nabla_\theta p(w|\mathcal{D},\theta)\big)\nabla_\theta \log p(w|\mathcal{D},\theta)^T\Delta\theta\ dw \\ &= \int_w p(w|\mathcal{D},\theta)\nabla_\theta \log p(w|\mathcal{D},\theta)^T\Delta\theta dw \\ &\quad + \int_w \Delta\theta^T p(w|\mathcal{D},\theta)\nabla_\theta \log p(w|\mathcal{D},\theta)\nabla_\theta \log p(w|\mathcal{D},\theta)^T\Delta\theta\ dw. \end{aligned} \tag{19}$$

First term of Eq. 19 is equal to 0:

$$\begin{aligned}&\int_w p(w|\mathcal{D},\theta)\nabla_\theta \log p(w|\mathcal{D},\theta)\ dw\\ &= \int_w p(w|\mathcal{D},\theta)\frac{\nabla_\theta p(w|\mathcal{D},\theta)}{p(w|\mathcal{D},\theta)}\ dw\\ &= \int_w \nabla_\theta p(w|\mathcal{D},\theta)\ dw\ = \nabla_\theta \int_w p(w|\mathcal{D},\theta) = 0.\end{aligned} \tag{20}$$

Using Eq. 19 and Eq. 20, Eq. 17 can be rewritten as Fisher information matrix by the definition of expectation:

$$\begin{aligned}&D_{KL}[p(w|\mathcal{D},\theta+\Delta\theta)||p(w|\mathcal{D},\theta)]\\ &= \int_w \Delta\theta^T p(w|\mathcal{D},\theta)\nabla_\theta \log p(w|\mathcal{D},\theta)\nabla_\theta \log p(w|\mathcal{D},\theta)^T\Delta\theta\\ &= \Delta\theta^T\mathbb{E}_w[\nabla_\theta \log p(w|\mathcal{D},\theta)\nabla_\theta \log p(w|\mathcal{D},\theta)^T]\Delta\theta\\ &= \Delta\theta^T F_\theta(\theta)\Delta\theta,\end{aligned} \tag{21}$$

where $F_\theta(\theta) = \mathbb{E}_{w,\mathcal{D}}[\nabla_\theta \log p(w|\mathcal{D},\theta)\nabla_\theta \log p(w|\mathcal{D},\theta)^T]$.

It's too expensive to calculate Fisher information matrix $F(\theta)$ in practice. We introduce a pseudo inverse for Fisher information matrix $F_\theta(\theta)^{-1}$ with Samelson inverse of a vector (Gentle, 2007; Sidi, 2017; Wynn, 1962) :

$$F_\theta(\theta)^{-1} = \frac{\nabla_\theta \log p(w|\mathcal{D},\theta)\nabla_\theta \log p(w|\mathcal{D},\theta)^T}{||\nabla_\theta \log p(w|\mathcal{D},\theta)||^4}. \tag{22}$$

Lagrangian Dual Problem From the result of Eq. 21, we can rewrite the Eq. 15:

$$\ell^\gamma_{\text{FP-BMA}}(\theta) = \max_{\Delta\theta^T F_\theta(\theta)\Delta\theta\leq\gamma^2} \ell(\theta+\Delta\theta). \tag{23}$$

We can reach the optimal perturbation of FP-BMA $\Delta\theta^*$ by using Taylor Expansion on $l(\theta+\Delta\theta)$ of Eq. 15:

$$\ell(\theta+\Delta\theta) = \ell(\theta) + \nabla_\theta\ell(\theta)^T\Delta\theta. \tag{24}$$

Using Eq. 24, we can rewrite Eq. 15 as Lagrangian dual problem:

$$L(\Delta\theta,\lambda) = \ell(\theta) + \nabla\ell_\theta(\theta)^T\Delta\theta - \lambda(\Delta\theta^T F_\theta(\theta)\Delta\theta - \gamma^2). \tag{25}$$

Differentiating Eq. 25, we get $\Delta\theta^*$:

$$\begin{aligned}\frac{\alpha L(\Delta\theta,\lambda)}{\alpha\Delta\theta} &= \nabla_\theta\ell(\theta)^T - 2\lambda\Delta\theta^T F_\theta(\theta) = 0\\ \therefore\ \Delta\theta^* &= \frac{1}{2\lambda}F_\theta(\theta)^{-1}\nabla_\theta\ell(\theta).\end{aligned} \tag{26}$$

Putting $\Delta\theta^*$ of Eq. 26 into $\Delta\theta$ of Eq. 25, we can rewrite Eq. 25:

$$\begin{aligned}L(\Delta\theta^*,\lambda) = l(\theta) &+ \frac{1}{2\lambda}\nabla_\theta\ell(\theta)^T F_\theta(\theta)^{-1}\nabla_\theta\ell(\theta)\\ &- \frac{1}{4\lambda}\nabla_\theta\ell(\theta)^T F_\theta(\theta)^{-1}\nabla_\theta\ell(\theta) + \lambda\gamma^2.\end{aligned} \tag{27}$$

By taking derivative of Eq. 27 w.r.t. λ, we can also get λ^*:

$$\begin{aligned}
&\frac{\alpha L(\Delta\theta^*, \lambda)}{\alpha\lambda} = -\frac{1}{2\lambda^2}\nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1}\nabla_\theta \ell(\theta) + \frac{1}{4\lambda^2}\nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1}\nabla_\theta \ell(\theta) + \gamma^2 = 0 \\
&4\lambda^2\gamma^2 = \nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1}\nabla_\theta \ell(\theta) \\
&\therefore\ \lambda^* = \frac{\sqrt{\nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1}\nabla_\theta \ell(\theta)}}{2\gamma}.
\end{aligned} \tag{28}$$

Finally, we get our $\Delta\theta_{\text{FP-BMA}}$ by substituting Eq. 28 into Eq. 26:

$$\Delta\theta_{\text{FP-BMA}} = \gamma\frac{F_\theta(\theta)^{-1}\nabla_\theta \ell(\theta)}{\sqrt{\nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1}\nabla_\theta \ell(\theta)}}. \tag{29}$$

B Experiments

B.1 Learning From Scratch

B.1.1 FP-BMA with diverse BNN frameworks

In Eq. 5, FP-BMA can be applied with various BNN frameworks by using an empirical loss function $\ell(\cdot)$ and adjusting the parameter β. We commonly set $\ell(\cdot)$ as cross-entropy loss in context of image classification task. Note that FP-BMA was applied only to the normalization layers and the last layer, while all other layers were trained using SGD.

FP-BMA (VI) For VI, we follow the loss function of Eq. 5.

FP-BMA (MCMC) We mainly adopt SGLD for MCMC in this work. For SGLD, we incorporated noise into Eq. 5 without KLD term ($\beta = 0$) based on the learning rate and the hyperparameter, temperature. In this approach, during the first step, the adversarial posterior is computed without any noise (Eq. 8). In the second step, both the noise and the adversarial posterior are used together in the learning process.

FP-BMA (SWAG) SWAG updates the first and second moments along the trajectory of SWA and uses these moments to approximate the posterior with a Gaussian distribution. In Eq. 5, β is fixed to 0, and as the trajectory of SWA is optimized through FP-BMA, posterior approximation can be performed accordingly.

B.1.2 Hyperparameters for Experiments

In this section, we provide the details of the experimental setup for Section 5.1. In the other experiments, the range of hyperparameters, excluding the number of epochs, is shared across different backbones and methods. For all experiments, the hyperparameters are selected using grid-search. Configuration of best hyperparameters for each baseline is summarized in Table 6 and Table 7.

Stochastic Gradient Descent with Momentum (SGD) In this study, we adopt Stochastic Gradient Descent with Momentum as an optimizer for DNN. Learning rate schedule is fixed to cosine decay. We run 300 epochs. The hyperparameter tuning range included learning rate in [1e-4, 1e-3, 1e-2].

Sharpness Aware Minimization (SAM) We set SGD with momentum as the base optimizer of SAM. It also ran upon a cosine decay learning rate scheduler. All the range of hyperparameters is shared with SGD with Momenmtum. Additional hyperparameter γ, the ball size of perturbation, is in [1e-2, 5e-2, 0.1].

Fisher SAM (FSAM) We set SGD with momentum as the base optimizer of FSAM. It also ran upon a cosine decay learning rate scheduler. All the range of hyperparameters is shared with SGD with Momenmtum. Additional hyperparameter η, regularize Fisher impact, is in [1e-2, 1e-1, 1].

SAM as an optimal relaxation of Bayes (bSAM) We use a cosine learning rate decay scheme. We run 300 epochs with fixed β_1 and β_2. The hyperparameter tuning rage included: learning rate in [1e-1, 3e-1, 5e-1, 8e-1, 1], weight decay in [1e-4, 5e-4, 1e-3, 1e-2], damping in [1e-1, 1e-2, 1e-3], and γ in [1e-3, 1e-2, 5e-2, 1e-1, 5e-1]. Damping parameter stabilizes the method by adding constant when updating variance estimate.

Variational Inference (VI) We use MOPED to change DNN into BNN, first. We set prior mean and variance as 0 and 1, respectively. Besides, we set the posterior mean as 0 and variance as 1e-3. We adopt Reparameterization as type of VI. The essential hyperparmeter for MOPED is δ, which adjusts how much to incorporate pre-trained weights. The δ was searched in [1e-3, 5e-3, 1e-2]. Moreover, we add a hyperparameter β for MOPED that can balance the loss term in VI. The β is in range [1e-2, 1e-1 ,1]

Table 6: Hyperparameter Configuration for CIFAR10

Backbone	Baseline	learning rate	β_1 (momentum)	β_2	γ	weight decay
RN18	SGD	5e-2	9e-1	$\times$	$\times$	5e-4
	SAM	1e-1	9e-1	$\times$	1e-1	5e-4
	FSAM	5e-2	9e-1	$\times$	1e-2	5e-4
	bSAM	8e-1	9e-1	0.999	1e-1	5e-4
	VI	5e-3	9e-1	$\times$	$\times$	5e-4
	FP-BMA (VI)	5e-2	9e-1	$\times$	1e-1	5e-4
	MCMC	1e-1	$\times$	$\times$	$\times$	5e-4
	E-MCMC	1e-1	$\times$	$\times$	$\times$	5e-4
	FP-BMA (MCMC)	5e-2	9e-1	$\times$	5e-2	5e-4
	SWAG	1e-1	9e-1	$\times$	$\times$	5e-4
	F-SWAG	1e-1	9e-1	$\times$	1e-1	5e-4
	FP-BMA (SWAG)	1e-1	9e-1	$\times$	1e-1	5e-4
ViT-B/16†	SGD	1e-1	9e-1	$\times$	$\times$	5e-4
	SAM	1e-1	9e-1	$\times$	5e-2	5e-4
	FSAM	1e-1	9e-1	$\times$	1e-1	5e-4
	bSAM	5e-1	9e-1	0.999	1e-1	5e-4
	VI	5e-3	9e-1	$\times$	$\times$	5e-4
	FP-BMA (VI)	5e-3	9e-1	$\times$	5e-3	5e-4
	MCMC	2e-2	$\times$	$\times$	$\times$	5e-4
	EMCMC	2e-2	$\times$	$\times$	$\times$	5e-4
	FP-BMA (MCMC)	3e-2	9e-1	$\times$	1e-2	5e-4
	SWAG	5e-2	9e-1	$\times$	$\times$	5e-4
	F-SWAG	5e-2	9e-1	$\times$		5e-4
	FP-BMA (SWAG)	5e-2	9e-1	$\times$	1e-2	5e-4

MCMC We consistently use SGLD (Welling & Teh, 2011) for MCMC in this work. It ran upon a cyclic cosine decay learning rate scheduler. The number of cycles was ranged in [2, 4]. The number of sampled models is in [10, 20, 28]. We search temperature in [1e-5, 5e-4, 1e-4, 5e-3, 1e-3, 1e-2].

Entropy-MCMC (E-MCMC) We use a cosine learning rate decay scheme, annealing the learning rate to zero. We run 300 epochs. We search η in [1e-4, 5e-3, 1e-3, 5e-2, 1e-2, 1e-1] and a system temperature T in [1e-4, 5e-4, 1e-3, 5e-3, 1e-2]. Note that the η handles flatness, and the system temperature adjusts the weight update's step size.

SWAG We use a cosine learning rate decay scheme for SWAG. All the range of hyperparameters is shared with SGD with Momenmtum. Additionally, we search for three additional hyperparameters for SWAG, capturing DNN snapshots and calculating statistics. First, the epoch to start SWA is in [161, 201], and epoch is 300. Second, the frequency of capturing the model snapshot is in [1, 2, 3]. Third, the low rank for covariance is in [2, 3, 5, 7, 10].

F-SWAG F-SWAG shares hyperparameter with SWAG, except γ. We search γ in [1e-2, 5e-2, 1e-1].

Flat Posterior-aware Bayesian Model Averaging (FP-BMA) In case of FP-BMA (VI), we set $\mathcal{N}(0, 1e-3)$ as prior and δ as 1e-3 to make DNN to BNN using MOPED. After getting prior distribution, we search three hyperparameters: learning rate and γ. The hyperparameter tuning range included: learning rate in [1e-3, 5e-3, 1e-2, 5e-2], γ in [1e-2, 5e-2, 1e-1, 5e-1]. We set weight decay as $5e-4$ for all backbones and train the model over 300 epochs with early stopping. We fix β as 1e-8 for all experiments. In case of FP-BMA (MCMC), we search learning rate, temperature for learning rate scheduling, and γ. The hyperparameter ranges are [1e-3, 5e-3, 1e-2, 5e-2] for learning rate, [1e-4, 5e-3, 1e-3, 5e-2, 1e-2, 1e-1] for temperature, and [5e-3, 1e-2, 5e-2, 1e-1, 5e-1] for γ. In case of FP-BMA (SWAG), we follow the hyperparameter for SWAG, except γ in [1e-2, 5e-2, 1e-1].

Table 7: Hyperparameter Configuration for CIFAR100

Backbone	Baseline	learning rate	β_1 (momentum)	β_2	γ	weight decay
RN18	SGD	1e-1	9e-1	$\times$	$\times$	5e-4
	SAM	5e-2	9e-1	$\times$	1e-1	5e-4
	FSAM	1e-1	9e-1	$\times$	1e-2	5e-4
	bSAM	1	9e-1	0.999	1e-1	5e-4
	VI	5e-3	9e-1	$\times$	$\times$	5e-4
	FP-BMA (VI)	8e-3	9e-1	$\times$	2e-1	5e-4
	MCMC	5e-1	$\times$	$\times$	$\times$	5e-4
	E-MCMC	5e-1	$\times$	$\times$	$\times$	5e-4
	FP-BMA (MCMC)	1e-1	9e-1	$\times$	3e-2	5e-4
	SWAG	1e-1	9e-1	$\times$	$\times$	5e-4
	F-SWAG	1e-1	9e-1	$\times$	1e-1	5e-4
	FP-BMA (SWAG)	3e-1	9e-1	$\times$	2e-1	5e-4
ViT-B/16[†]	SGD	1e-1	9e-1	$\times$	$\times$	5e-4
	SAM	1e-1	9e-1	$\times$	1e-1	5e-4
	FSAM	1e-1	9e-1	$\times$	1e-2	5e-4
	bSAM	5e-1	9e-1	0.999	1e-1	5e-4
	VI	3e-2	9e-1	$\times$	$\times$	5e-4
	FP-BMA (VI)	8e-3	9e-1	$\times$	1e-1	5e-4
	MCMC	2e-1	$\times$	$\times$	$\times$	5e-4
	EMCMC	1e-1	$\times$	$\times$	$\times$	5e-4
	FP-BMA (MCMC)	5e-2	9e-1	$\times$	5e-2	5e-4
	SWAG	1e-1	9e-1	$\times$	$\times$	5e-4
	F-SWAG	1e-1	9e-1	$\times$	1e-1	5e-4
	FP-BMA (SWAG)	1e-1	9e-1	$\times$	1e-1	5e-4

B.2 Few-shot Image Classification with Bayesian Transfer Learning

B.2.1 FP-BMA with diverse BNN frameworks

Diverse BNN frameworks can be adopted for Bayesian Transfer Learning. Specifically, there are several options for making pre-trained DNN into BNN. In this work, we mainly adopt MOPED and SWAG for the converting.

In addition, FP-BMA can be applied with various BNN frameworks by using an empirical loss function $\ell(\cdot)$ and adjusting the parameter β in Eq. 9. We commonly set $\ell(\cdot)$ as cross-entropy loss in context of image classification task.

FP-BMA (VI) First, we convert pre-trained DNN into BNN with MOPED. We set the converted BNN as prior, $q_\theta^{\text{pr}}(w|\mathcal{D}^{\text{pr}})$ in Eq. 9, and initial point of model. We only train parameters of normalization and last layer and freeze others. We train them with the loss function of Eq. 9.

FP-BMA (MCMC) For SGLD, it is unnecessary to convert pre-trained DNN into BNN. Instead, we directly set the pre-trained DNN as initialization. We incorporated noise into Eq. 9 without the KLD term ($\beta = 0$) based on the learning rate and the hyperparameter, temperature. During the first step, the adversarial posterior is computed without any noise (Eq. 8). In the second step, both the noise and the adversarial posterior are used together in the learning process.

FP-BMA (SWAG) SWAG is also one of the options to convert pre-trained DNN into BNN. Specifically, we run a few epochs with source or downstream datasets to make BNN from pre-trained DNN. After this step, we set the BNN as the prior, $q_\theta^{\text{pr}}(w|\mathcal{D}^{\text{pr}})$ in Eq. 9. We also let the converted BNN as initialization and train with downstream dataset. We optimize model with the loss function in Eq. 9.

B.3 Hyperparameters for Experiments

In this section, we provide the details of the experimental setup for Section 5.2. In the other experiments, the range of hyperparameters, excluding the number of epochs, is shared across different backbones and methods.

First, we provide remarks for each baseline method, followed by the tables of hyperparameter configuration with respect to downstream datasets and the baselines. For all experiments, the hyperparameters are selected using grid-search. Configuration of best hyperparameters for each baseline is summarized in Table 8 and Table 9. We ran all experiments using GeForce RTX 3090 and NVIDIA RTX A6000 with GPU memory of 24,576MB and 49,140 MB.

Stochastic Gradient Descent with Momentum (SGD) In this study, we adopt Stochastic Gradient Descent with Momentum as an optimizer for DNN. Learning rate schedule is fixed to cosine decay with warmup length of 10. We tested [100, 150] epoch and set 100 epoch as the best option. In overall experiments, we set momentum as 0.9. The hyperparameter tuning range included learning rate in [1e-4, 1e-3, 1e-2], and weight decay in [1e-4, 5e-4, 1e-3, 1e-2].

Sharpness Aware Minimization (SAM) We set SGD with momentum as the base optimizer of SAM. It also ran upon a cosine decay learning rate scheduler. All the range of hyperparameters is shared with SGD with Momenmtum. Additional hyperparameter γ, the ball size of perturbation, is in [1e-2, 5e-2, 1e-1].

Fisher SAM (FSAM) We set SGD with momentum as the base optimizer of FSAM. It also ran upon a cosine decay learning rate scheduler. All the range of hyperparameters is shared with SGD with Momenmtum. Additional hyperparameter η, regularize Fisher impact, is in [1e-2, 1e-1, 1].

SAM as an optimal relaxation of Bayes (bSAM) We use a cosine learning rate decay scheme, annealing the learning rate to zero. We fine-tuned pre-trained models for 150 epochs with fixed β_1 and β_2. The hyperparameter tuning range included: learning rate in [1e-3, 1e-2, 5e-2, 1e-1, 0.25, 0.5, 1], weight decay in [1e-3, 1e-2, 1e-1], damping in [1e-3, 1e-2, 1e-1], noise scaling parameter in

Table 8: Hyperparameter Configuration for CIFAR10

Backbone	Baseline	learning rate	β_1 (momentum)	β_2	γ	weight decay
RN18	SGD	5e-3	9e-1	×	×	1e-3
	SAM	1e-2	9e-1	×	1e-1	1e-4
	FSAM	1e-2	9e-1	×	1e-1	1e-4
	bSAM	1e-1	9e-1	0.999	5e-2	1e-1
	MOPED	1e-2	9e-1	×	×	1e-4
	SA-BMA (VI)	1e-2	9e-1	×	7e-1	1e-3
	MCMC	5e-2	9e-1	×	×	5e-4
	PTL	1e-1	×	×	×	1e-3
	E-MCMC	5e-2	×	×	×	1e-3
	SA-BMA (MCMC)	5e-3	9e-1	×	8e-3	5e-4
	SWAG	5e-3	9e-1	×	×	1e-5
	F-SWAG	5e-3	9e-1	×	5e-2	5e-4
	SA-BMA (SWAG)	5e-2	9e-1	×	1e-1	5e-4
ViT-B/16	SGD	1e-3	9e-1	×	×	1e-4
	SAM	1e-3	9e-1	×	1e-2	1e-3
	FSAM	5e-3	9e-1	×	1e-2	1e-3
	bSAM	1e-1	9e-1	0.999	1e-2	1e-1
	MOPED	1e-3	9e-1	×	×	1e-4
	SA-BMA (VI)	1e-2	9e-1	×	1e-1	5e-4
	MCMC	3e-2	9e-1	×	×	5e-4
	PTL	6e-2	×	×	×	1e-3
	EMCMC	5e-3	×	×	×	1e-2
	SA-BMA (MCMC)	5e-3	9e-1	×	8e-3	5e-4
	SWAG	1e-3	9e-1	×	×	1e-3
	F-SWAG	1e-3	9e-1	×	1e-2	1e-3
	SA-BMA (SWAG)	5e-3	9e-1	×	5e-1	5e-4

[1e-4, 1e-3, 1e-2, 1e-1], and γ in [1e-3, 1e-2, 5e-2, 1e-1]. Damping parameter stabilizes the method by adding constant when updating variance estimate. Since SAM as Bayes optimizer depends on the number of samples to scale the prior, we introduced additional noise scaling parameters to mitigate the gap between the experimental settings, where SAM as Bayes assumed training from scratch and our method assumed few-shot fine-tuning on the pre-trained model. We multiplied noise scaling parameter to the variance of the Gaussian noise to give strong prior, assuming pre-trained model.

Model Priors with Empirical Bayes using DNN (MOPED) MOPED was a baseline to compare for Bayesian Transfer Learning. It employs pre-trained DNN and transforms it into Mean-Field Variational Inference (MFVI). We set prior mean and variance as 0 and 1, respectively. Besides, we set the posterior mean as 0 and variance as 1e-3. We adopt Reparameterization as type of VI. The essential hyperparameter for MOPED is δ, which adjusts how much to incorporate pre-trained weights. The δ was searched in [5e-2, 1e-1, 2e-1]. Moreover, we add a hyperparameter β for MOPED that can balance the loss term in VI. The β is in range [1e-2, 1e-1, 1].

MCMC We consistently use SGLD (Welling & Teh, 2011) for MCMC in this work. It ran upon a cyclic cosine decay learning rate scheduler. The number of cycles was ranged in [2, 4]. The number of sampled models is in [10, 20, 28]. We search temperature in [1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1].

Pre-train Your Loss (PTL) The backbones both ResNet18 and Vit-B/16 were refined through fine-tuning with a classification head for the target task, leveraging a prior distribution learned from SWAG on the ImageNet 1k dataset using SGD. First, the hyperparameter tuning range of the pre-training epoch is [2, 3, 5, 15, 30] to generate the prior distribution on the source task, ImageNet 1k. The learning rate was 0.1. We approximated the covariance low rank as 5. Second, in the downstream task, the fine-tuning optimizer is SGLD with a cosine learning rate schedule, sampling 30 in 5 cycles. The hyperparameter tuning range included: learning rate in [1e-4, 1e-3, 1e-2, 5e-2, 6e-2, 1e-1, 5e-1],

Table 9: Hyperparameter Configuration for CIFAR100

Backbone	Baseline	learning rate	β_1 (momentum)	β_2	γ	weight decay
RN18	SGD	1e-2	9e-1	$\times$	$\times$	5e-3
	SAM	1e-2	9e-1	$\times$	5e-2	1e-2
	FSAM	1e-2	9e-1	$\times$	1e-1	1e-4
	bSAM	1	9e-1	0.999	1e-2	1e-2
	MOPED	1e-2	9e-1	$\times$	$\times$	1e-3
	SA-BMA (VI)	5e-2	9e-1	$\times$	1e-2	5e-4
	MCMC	3e-2	9e-1	$\times$	$\times$	5e-4
	PTL	5e-1	$\times$	$\times$	$\times$	1e-3
	E-MCMC	5e-2	$\times$	$\times$	$\times$	1e-3
	SA-BMA (MCMC)	1e-2	9e-1	$\times$	1e-1	5e-4
	SWAG	1e-2	9e-1	$\times$	$\times$	1e-4
	F-SWAG	1e-2	9e-1	$\times$	5e-2	1e-2
	SA-BMA (SWAG)	5e-2	9e-1	$\times$	5e-1	5e-4
ViT-B/16	SGD	1e-3	9e-1	$\times$	$\times$	1e-2
	SAM	1e-3	9e-1	$\times$	1e-2	1e-2
	FSAM	5e-3	9e-1	$\times$	1e-2	1e-4
	bSAM	2.5e-1	9e-1	0.999	1e-2	1e-3
	MOPED	1e-3	9e-1	$\times$	$\times$	1e-3
	SA-BMA (VI)	1e-2	9e-1	$\times$	5e-2	5e-4
	MCMC	5e-2	9e-1	$\times$	$\times$	5e-4
	PTL	1e-1	$\times$	$\times$	$\times$	1e-3
	E-MCMC	5e-2	$\times$	$\times$	$\times$	1e-3
	SA-BMA (MCMC)	8e-3	9e-1	$\times$	8e-3	5e-4
	SWAG	1e-3	9e-1	$\times$	$\times$	1e-2
	F-SWAG	1e-3	9e-1	$\times$	1e-2	1e-2
	SA-BMA (SWAG)	1e-2	9e-1	$\times$	5e-1	5e-4

weight decay in [1e-4, 1e-3 ,1e-2 ,1e-1], and prior scale in [1e+4, 1e+5, 1e+6]. Prior scaling in the downstream task is to reflect the mismatch between the pre-training and downstream tasks and to add coverage to parameter settings that might be consistent with the downstream. Training was conducted over 150 epochs; tuning range of fine-tuning epoch is [100, 150, 200, 300, 1000].

Entropy-MCMC (E-MCMC) We use a cosine learning rate decay scheme, annealing the learning rate to zero. We set the range of the hyperparameter sweep to the surroundings of the best hyperparameter in E-MCMC for ResNet18: learning rate in [5e-3, 5e-2, 5e-1], weight decay in [1e-4, 1e-3, 1e-2], η in [1e-6, 5e-6, 1e-5, 5e-5, 1e-4, 4e-4, 5e-3, 8e-3, 1e-2] and a system temperature T in [1e-5, 1e-4, 1e-3]. In this study, we performed an extensive exploration of the hyperparameter space of ViT-B/16, as it has a mechanism different from the CNN family and may not be found near the best hyperparameter range of ResNet18: learning rate in [1e-3, 5e-3, 1e-2, 5e-2, 5e-1], weight decay in [1e-5, 1e-4, 5e-4, 1e-3, 1e-2, 5e-2], η in [5e-7, 1e-6, 5e-6, 5e-5, 1e-4, 4e-4, 5e-4, 1e-3, 8e-3, 1e-2, 1e-1] and a system temperature T in [1e-6, 5e-6, 1e-5, 5e-5, 1e-4, 1e-3, 1e-2, 1e-1]. We fine-tuned pre-trained models for 150 epochs. Note that the η handles flatness, and the system temperature adjusts the weight update's step size.

SWAG We use a cosine learning rate decay scheme for SWAG. All the range of hyperparameters is shared with SGD with Momenmtum. Additionally, we search three additional hyperparameters for SWAG, capturing DNN snapshots and calculating statistics. First, the epoch to start SWA is in [51, 76, 101] and epoch is in [100, 150]. Second, the frequency to capture the model snapshot is in [1, 2, 3]. Third, the low rank for covariance is in [2, 3, 5, 7, 10].

F-SWAG F-SWAG shares hyperparameter with SWAG, except γ. We search γ in [1e-2, 5e-2, 1e-1].

Flat Posterior-awre Bayesian Model Averaging (FP-BMA) In case of FP-BMA (SWAG), we train SWAG on source task IN 1K to make prior distribution and follow the pre-training protocol of PTL. In case of employing MOPED to make prior distribution, we do not go through any training step. In case of FP-BMA (VI), we just set δ as 0.05 for MOPED and make DNN into BNN. In case of FP-BMA (MCMC), we just set pre-trained weight as initialization and run experiments. After getting prior distribution, we search three hyperparameters: learning rate, γ, and α. The hyperparamter tuning range included: learning rate in [1e-3, 5e-3, 1e-2, 5e-2], γ in [5e-3, 8e-3, 1e-2, 5e-2, 1e-1, 5e-1, 7e-1], and α in [1e-6, 1e-5, 1e-4, 1e-3]. We set weight decay as $5e-4$ for all backbones and train the model over 150 epochs with early stopping. We fix β as 1e-8 for all experiments.

B.4 Algorithm of FP-BMA

Training algorithm of FP-BMA with Bayesian transfer learning can be depicted as Algorithm 1. In the first step, load a model pre-trained on the source task. Note that the pre-trained models do not have to be BNN. Namely, it is capable of using DNN, which can be easier to find than pre-trained BNN. Second, change the loaded DNN into BNN on the source or downstream task. Every BNN framework, containing VI, SWAG, LA, etc., can be adopted to make DNN into BNN. This study mainly employs PTL (Shwartz-Ziv et al., 2022) and MOPED (Krishnan et al., 2020) for this step. We can skip this second step if you load a pre-trained BNN model before. Third, train the subnetwork of the converted BNN model with the proposed flat-seeking seeking optimizer. It allows model to converge into flat minina efficiently.

Algorithm 1 FP-BMA with Bayesian Transfer Learning

```
Require: Variational parameter $\theta$, Neighborhood size $\gamma$, Epochs $E$, and Learning rate $\eta_{\text{FP-BMA}}$
  1) Load pre-trained DNN
  2) Make pre-trained DNN model into BNN $q_\theta^{\text{pr}}(w|\mathcal{D}^{\text{pr}})$ and set as prior
  for $t = 1, 2, ..., E$ do
      3-1) $w \sim q_\theta(w|\mathcal{D}^{\text{ft}})$                ▷ Sample weight from posterior
      3-4) Forward and calculate the loss $l(\theta)$ with the sampled $w$
      3-5) Backward pass and compute $\nabla_\theta \log p_\theta(w|\mathcal{D})$
      3-6) Compute $F_\theta^{-1}(\theta) = \frac{\nabla_\theta \log p_\theta(w|\mathcal{D}) \nabla_\theta \log p_\theta(w|\mathcal{D})^T}{\|\nabla_\theta \log p_\theta(w|\mathcal{D})\|^4}$
      3-7) Compute the perturbation $\Delta\theta_{\text{FP-BMA}} = \gamma \frac{F_\theta(\theta)^{-1} \nabla_\theta l(\theta)}{\sqrt{\nabla_\theta l(\theta)^T F_\theta(\theta)^{-1} \nabla_\theta l(\theta)}}$
      3-8) Compute gradient approximation for the FP-BMA $\nabla_\theta l_{\text{FP-BMA}}(\theta) = \frac{\partial l(\theta)}{\partial \theta}|_{\theta + \Delta\theta_{\text{FP-BMA}}}$
      3-9) Update $\theta \to \theta - \eta \nabla_\theta l_{\text{FP-BMA}}(\theta)$
  end for
```

B.5 Efficiency of FP-BMA with Bayesian Transfer Learning

BNN often struggles with high computation and memory complexity, which makes optimizing large-scale BNN hard. However, FP-BMA only optimizes the last (classifier) and normalization layer, which only requires vector-sized learnable parameters. Table 10 provides the scalability of FP-BMA and baselines in the fine-tuning stage given pre-trained model. FP-BMA only requires fewer learnable parameters since $p_1 \ll p$ and low rank K are even fewer than DNN, where p_1 denotes the number of parameters in normalization and last layers. It only needs 1% of learnable parameters compared to other methods in case of RN18 and ViT-B/16. FP-BMA efficiently adapts the model in a few-shot setting.

Table 10: Efficiency of FP-BMA with Bayesian Transfer Learning.

Method	Optim	Num. of Tr Param.
DNN	SGD	p
	SAM	p
	FSAM	p
SWAG	SGD	p
F-SWAG	SAM	p
VI	bSAM	p
MOPED	SGD	$2p$
E-MCMC	SGLD	$2p$
PTL	SGLD	p
FP-BMA	FP-BMA	$(K+2)p_1$

B.6 Fine-Grained Image Classification

In addition to classification accuracy, FP-BMA shows superior performance compared to the baseline in NLL metric, indicating that FP-BMA effectively quantifies uncertainty.

Table 11: Downstream task NLL with RN50 and ViT-B/16 pre-trained on IN 1K. FP-BMA (SWAG) denotes using SWAG to convert pre-trained model into BNN. **Bold** and underline denote best and second best performance each. FP-BMA demonstrates superior performance across all 16-shot datasets, including EuroSAT , Oxford Flowers, Oxford Pets, and UCF101.

Backbone	RN50					ViT-B/16				
Method	EuroSAT	Oxford Flowers	Oxford Pets	UCF101	Avg	EuroSAT	Oxford Flowers	Oxford Pets	UCF101	Avg
SGD	$0.416_{\pm 0.043}$	$0.265_{\pm 0.010}$	$0.367_{\pm 0.008}$	$1.331_{\pm 0.024}$	$0.595_{\pm 0.010}$	$0.573_{\pm 0.044}$	$0.361_{\pm 0.027}$	$0.385_{\pm 0.044}$	$1.246_{\pm 0.044}$	$0.641_{\pm 0.020}$
SAM	$0.376_{\pm 0.003}$	$\underline{0.190}_{\pm 0.001}$	$\underline{0.344}_{\pm 0.014}$	$\underline{1.157}_{\pm 0.035}$	$0.517_{\pm 0.005}$	$0.522_{\pm 0.023}$	$\underline{0.276}_{\pm 0.029}$	$\underline{0.287}_{\pm 0.022}$	$\underline{1.140}_{\pm 0.034}$	$\underline{0.556}_{\pm 0.020}$
SWAG	$0.343_{\pm 0.046}$	$0.264_{\pm 0.011}$	$0.367_{\pm 0.007}$	$1.347_{\pm 0.022}$	$0.580_{\pm 0.009}$	$0.547_{\pm 0.021}$	$0.361_{\pm 0.027}$	$0.366_{\pm 0.010}$	$1.286_{\pm 0.045}$	$0.640_{\pm 0.006}$
F-SWAG	$\underline{0.301}_{\pm 0.039}$	$0.190_{\pm 0.002}$	$0.351_{\pm 0.010}$	$1.186_{\pm 0.034}$	$\underline{0.507}_{\pm 0.008}$	$0.514_{\pm 0.018}$	$0.276_{\pm 0.033}$	$0.297_{\pm 0.030}$	$1.234_{\pm 0.031}$	$0.580_{\pm 0.017}$
MOPED	$0.481_{\pm 0.100}$	$0.347_{\pm 0.019}$	$0.388_{\pm 0.007}$	$1.367_{\pm 0.029}$	$0.646_{\pm 0.028}$	$\underline{0.484}_{\pm 0.018}$	$0.354_{\pm 0.025}$	$0.309_{\pm 0.015}$	$1.180_{\pm 0.028}$	$0.582_{\pm 0.017}$
PTL	$0.319_{\pm 0.006}$	$0.307_{\pm 0.010}$	$0.360_{\pm 0.015}$	$1.391_{\pm 0.036}$	$0.594_{\pm 0.010}$	$0.493_{\pm 0.012}$	$0.616_{\pm 0.066}$	$0.381_{\pm 0.008}$	$1.670_{\pm 0.050}$	$0.790_{\pm 0.013}$
FP-BMA	$\mathbf{0.297}_{\pm 0.038}$	$\mathbf{0.147}_{\pm 0.037}$	$\mathbf{0.339}_{\pm 0.023}$	$\mathbf{1.113}_{\pm 0.009}$	$\mathbf{0.474}_{\pm 0.023}$	$\mathbf{0.455}_{\pm 0.006}$	$\mathbf{0.219}_{\pm 0.037}$	$\mathbf{0.272}_{\pm 0.006}$	$\mathbf{1.071}_{\pm 0.036}$	$\mathbf{0.504}_{\pm 0.012}$

B.7 Performance under Distribution shift

We adopt the corrupted dataset CIFAR10/100C to test the robustness over distribution shift. The corrupted dataset transform the CIFAR10/100-test dataset, which has been modified to shift the distribution of the test data further away from the training data. It contains 19 kinds of corrupt options, such as varying brightness or contrast to adding Gaussian noise. The severity level indicates the strength of the transformation and is typically expressed as a number from 1 to 5, where the higher the number, the stronger the transformation. In Figure 4, our method ensures relatively robust performance in the data distribution shift, even as the severity increases.

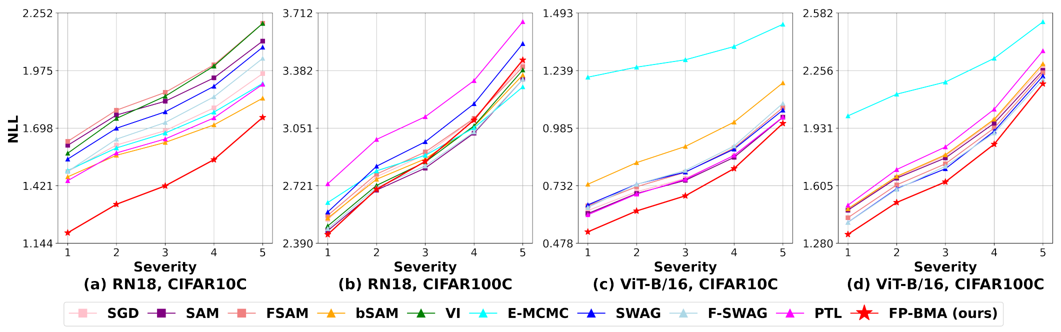


Figure 4: NLL performance of ResNet 18 and ViT-B/16 on corrupted CIFAR10 and CIFAR100, respectively (Hendrycks & Dietterich, 2019).

We also provide the detailed results of three repeated experiments with corrupted sets.

(a) RN18 CIFAR10C

Method	Severity									
	1		2		3		4		5	
	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓
SGD	49.57$_{\pm0.97}$	1.49$_{\pm0.02}$	45.78$_{\pm1.43}$	1.62$_{\pm0.04}$	43.78$_{\pm1.44}$	1.69$_{\pm0.04}$	40.83$_{\pm1.59}$	1.80$_{\pm0.06}$	36.30$_{\pm1.79}$	1.96$_{\pm0.08}$
SAM	50.23$_{\pm2.11}$	1.62$_{\pm0.07}$	46.56$_{\pm2.00}$	1.76$_{\pm0.03}$	44.59$_{\pm2.26}$	1.83$_{\pm0.03}$	41.85$_{\pm2.42}$	1.94$_{\pm0.04}$	37.33$_{\pm2.52}$	2.12$_{\pm0.07}$
FSAM	48.76$_{\pm4.00}$	1.63$_{\pm0.03}$	45.11$_{\pm3.91}$	1.78$_{\pm0.01}$	42.94$_{\pm3.88}$	1.87$_{\pm0.03}$	40.06$_{\pm3.85}$	2.00$_{\pm0.08}$	35.70$_{\pm3.50}$	2.20$_{\pm0.12}$
SWAG	50.05$_{\pm0.76}$	1.55$_{\pm0.09}$	46.31$_{\pm1.16}$	1.70$_{\pm0.11}$	44.17$_{\pm1.07}$	1.78$_{\pm0.11}$	41.20$_{\pm1.13}$	1.90$_{\pm0.13}$	36.64$_{\pm1.26}$	2.09$_{\pm0.15}$
F-SWAG	51.37$_{\pm1.08}$	1.49$_{\pm0.05}$	47.35$_{\pm0.71}$	1.64$_{\pm0.04}$	45.16$_{\pm0.66}$	1.72$_{\pm0.06}$	42.01$_{\pm0.57}$	1.85$_{\pm0.06}$	37.27$_{\pm0.64}$	2.03$_{\pm0.07}$
bSAM	49.20$_{\pm2.40}$	1.46$_{\pm0.05}$	45.35$_{\pm1.93}$	1.57$_{\pm0.04}$	43.07$_{\pm2.10}$	1.63$_{\pm0.04}$	40.12$_{\pm1.74}$	1.71$_{\pm0.03}$	35.50$_{\pm1.36}$	1.84$_{\pm0.02}$
VI	50.72$_{\pm0.80}$	1.58$_{\pm0.11}$	46.87$_{\pm0.32}$	1.74$_{\pm0.11}$	44.52$_{\pm0.39}$	1.85$_{\pm0.12}$	41.38$_{\pm0.29}$	2.00$_{\pm0.12}$	36.73$_{\pm0.17}$	2.20$_{\pm0.10}$
E-MCMC	49.86$_{\pm1.54}$	1.49$_{\pm0.03}$	46.17$_{\pm1.55}$	1.60$_{\pm0.04}$	44.07$_{\pm1.72}$	1.67$_{\pm0.07}$	41.05$_{\pm1.65}$	1.77$_{\pm0.10}$	36.53$_{\pm1.74}$	1.91$_{\pm0.13}$
PTL	50.44$_{\pm1.65}$	1.45$_{\pm0.06}$	46.22$_{\pm1.96}$	1.58$_{\pm0.09}$	44.06$_{\pm1.67}$	1.65$_{\pm0.09}$	41.02$_{\pm1.66}$	1.75$_{\pm0.11}$	36.14$_{\pm1.51}$	1.91$_{\pm0.13}$
FP-BMA	**58.53**$_{\pm0.75}$	**1.19**$_{\pm0.02}$	**53.72**$_{\pm0.70}$	**1.33**$_{\pm0.00}$	**50.61**$_{\pm0.84}$	**1.42**$_{\pm0.01}$	**46.76**$_{\pm1.15}$	**1.55**$_{\pm0.03}$	**40.70**$_{\pm1.34}$	**1.75**$_{\pm0.05}$

(b) RN18 CIFAR100C

Method	Severity									
	1		2		3		4		5	
	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓
SGD	36.01$_{\pm0.86}$	2.55$_{\pm0.06}$	31.81$_{\pm0.73}$	2.79$_{\pm0.06}$	29.75$_{\pm0.57}$	2.91$_{\pm0.04}$	26.73$_{\pm0.25}$	3.11$_{\pm0.02}$	22.20$_{\pm0.08}$	3.40$_{\pm0.00}$
SAM	37.94$_{\pm0.52}$	2.46$_{\pm0.02}$	33.57$_{\pm0.50}$	**2.69**$_{\pm0.03}$	31.46$_{\pm0.67}$	**2.82**$_{\pm0.03}$	28.19$_{\pm0.75}$	3.02$_{\pm0.05}$	23.32$_{\pm0.69}$	3.33$_{\pm0.06}$
FSAM	36.46$_{\pm0.44}$	2.53$_{\pm0.05}$	32.24$_{\pm0.36}$	2.77$_{\pm0.04}$	30.19$_{\pm0.42}$	2.90$_{\pm0.03}$	27.12$_{\pm0.37}$	3.10$_{\pm0.02}$	22.48$_{\pm0.39}$	3.42$_{\pm0.01}$
bSAM	36.20$_{\pm0.59}$	2.73$_{\pm0.03}$	32.48$_{\pm0.34}$	2.99$_{\pm0.03}$	30.66$_{\pm0.33}$	3.12$_{\pm0.02}$	27.94$_{\pm0.14}$	3.32$_{\pm0.05}$	23.66$_{\pm0.29}$	3.66$_{\pm0.06}$
SWAG	35.84$_{\pm5.17}$	2.62$_{\pm0.30}$	32.43$_{\pm4.55}$	2.81$_{\pm0.27}$	30.71$_{\pm4.21}$	2.89$_{\pm0.25}$	28.13$_{\pm3.81}$	3.05$_{\pm0.22}$	24.24$_{\pm2.99}$	**3.29**$_{\pm0.17}$
F-SWAG	37.10$_{\pm0.60}$	2.49$_{\pm0.03}$	32.84$_{\pm0.62}$	2.72$_{\pm0.03}$	30.59$_{\pm0.72}$	2.86$_{\pm0.04}$	27.43$_{\pm0.91}$	3.06$_{\pm0.06}$	22.74$_{\pm0.93}$	3.38$_{\pm0.08}$
VI	38.20$_{\pm0.57}$	2.47$_{\pm0.02}$	33.77$_{\pm0.59}$	2.71$_{\pm0.03}$	31.70$_{\pm0.75}$	2.83$_{\pm0.03}$	28.56$_{\pm0.77}$	**3.03**$_{\pm0.04}$	23.72$_{\pm0.78}$	3.33$_{\pm0.05}$
E-MCMC	36.49$_{\pm0.89}$	2.57$_{\pm0.06}$	32.25$_{\pm0.76}$	2.83$_{\pm0.06}$	30.22$_{\pm0.63}$	2.97$_{\pm0.05}$	27.17$_{\pm0.38}$	3.19$_{\pm0.03}$	22.54$_{\pm0.27}$	3.54$_{\pm0.01}$
PTL	36.43$_{\pm0.35}$	2.53$_{\pm0.03}$	32.24$_{\pm0.40}$	2.76$_{\pm0.03}$	30.20$_{\pm0.42}$	2.87$_{\pm0.03}$	27.17$_{\pm0.55}$	3.06$_{\pm0.04}$	22.56$_{\pm0.54}$	3.36$_{\pm0.05}$
FP-BMA	**39.41**$_{\pm0.72}$	**2.44**$_{\pm0.04}$	**35.07**$_{\pm0.64}$	2.70$_{\pm0.05}$	**32.75**$_{\pm0.71}$	2.86$_{\pm0.05}$	**29.41**$_{\pm0.67}$	3.10$_{\pm0.05}$	**24.25**$_{\pm0.70}$	3.44$_{\pm0.05}$

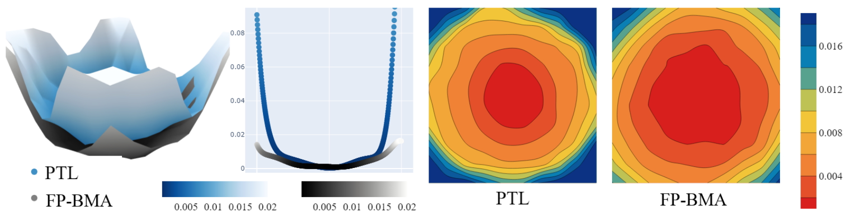
(c) VIT-B/16 CIFAR10C

Method	Severity									
	1		2		3		4		5	
	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓
SGD	79.62$_{\pm0.56}$	0.64$_{\pm0.06}$	76.47$_{\pm0.67}$	0.73$_{\pm0.06}$	74.10$_{\pm0.83}$	0.79$_{\pm0.05}$	70.42$_{\pm1.23}$	0.90$_{\pm0.05}$	64.41$_{\pm1.85}$	1.08$_{\pm0.05}$
SAM	79.78$_{\pm0.49}$	0.61$_{\pm0.01}$	76.59$_{\pm0.64}$	0.70$_{\pm0.02}$	74.58$_{\pm0.94}$	0.75$_{\pm0.02}$	71.12$_{\pm1.06}$	0.86$_{\pm0.03}$	65.26$_{\pm1.46}$	1.03$_{\pm0.04}$
FSAM	79.87$_{\pm0.83}$	0.62$_{\pm0.02}$	76.78$_{\pm0.78}$	0.70$_{\pm0.02}$	74.70$_{\pm0.60}$	0.76$_{\pm0.01}$	71.29$_{\pm0.49}$	0.86$_{\pm0.01}$	65.53$_{\pm0.56}$	1.03$_{\pm0.03}$
bSAM	78.80$_{\pm1.18}$	0.64$_{\pm0.04}$	75.43$_{\pm1.14}$	0.74$_{\pm0.04}$	73.45$_{\pm1.43}$	0.80$_{\pm0.04}$	70.07$_{\pm1.50}$	0.91$_{\pm0.05}$	64.21$_{\pm1.57}$	1.09$_{\pm0.05}$
SWAG	76.58$_{\pm1.69}$	1.21$_{\pm0.04}$	73.45$_{\pm1.98}$	1.25$_{\pm0.04}$	71.20$_{\pm2.18}$	1.29$_{\pm0.04}$	67.54$_{\pm2.46}$	1.35$_{\pm0.04}$	61.65$_{\pm2.82}$	1.44$_{\pm0.04}$
F-SWAG	81.03$_{\pm2.20}$	0.60$_{\pm0.05}$	77.73$_{\pm2.63}$	0.69$_{\pm0.06}$	75.45$_{\pm2.96}$	0.76$_{\pm0.07}$	71.82$_{\pm3.31}$	0.87$_{\pm0.08}$	66.05$_{\pm3.59}$	1.03$_{\pm0.10}$
E-MCMC	78.91$_{\pm2.31}$	0.65$_{\pm0.08}$	75.78$_{\pm2.36}$	0.74$_{\pm0.08}$	73.94$_{\pm2.56}$	0.79$_{\pm0.09}$	70.66$_{\pm2.63}$	0.89$_{\pm0.10}$	65.07$_{\pm2.77}$	1.06$_{\pm0.11}$
PTL	76.26$_{\pm2.46}$	0.74$_{\pm0.06}$	72.36$_{\pm2.41}$	0.83$_{\pm0.06}$	69.61$_{\pm2.46}$	0.90$_{\pm0.07}$	65.47$_{\pm2.52}$	1.01$_{\pm0.07}$	59.04$_{\pm2.26}$	1.18$_{\pm0.06}$
FP-BMA	**82.89**$_{\pm1.09}$	**0.53**$_{\pm0.04}$	**79.68**$_{\pm1.26}$	**0.62**$_{\pm0.04}$	**77.30**$_{\pm1.43}$	**0.69**$_{\pm0.05}$	**73.41**$_{\pm1.62}$	**0.81**$_{\pm0.06}$	**66.94**$_{\pm1.79}$	**1.01**$_{\pm0.07}$

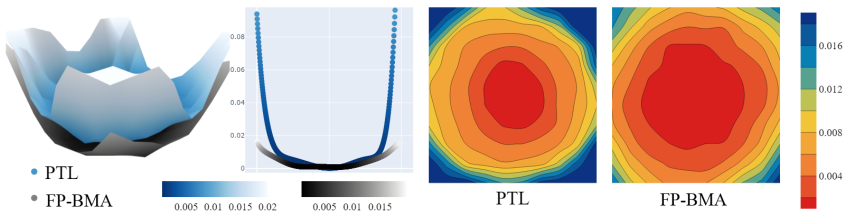
(d) VIT-B/16 CIFAR100C

Method	Severity									
	1		2		3		4		5	
	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓
SGD	62.19$_{\pm0.52}$	1.42$_{\pm0.02}$	57.81$_{\pm0.37}$	1.61$_{\pm0.02}$	55.04$_{\pm0.14}$	1.73$_{\pm0.02}$	50.73$_{\pm0.24}$	1.93$_{\pm0.01}$	44.12$_{\pm0.39}$	2.24$_{\pm0.01}$
SAM	61.90$_{\pm0.53}$	1.47$_{\pm0.02}$	57.49$_{\pm0.43}$	1.65$_{\pm0.02}$	54.80$_{\pm0.29}$	1.76$_{\pm0.01}$	50.52$_{\pm0.25}$	1.96$_{\pm0.01}$	44.04$_{\pm0.24}$	2.26$_{\pm0.01}$
FSAM	61.70$_{\pm0.52}$	1.47$_{\pm0.02}$	57.16$_{\pm0.44}$	1.65$_{\pm0.02}$	54.46$_{\pm0.37}$	1.77$_{\pm0.02}$	50.11$_{\pm0.39}$	1.97$_{\pm0.01}$	43.53$_{\pm0.42}$	2.28$_{\pm0.01}$
bSAM	62.36$_{\pm0.73}$	1.40$_{\pm0.03}$	57.97$_{\pm0.70}$	1.58$_{\pm0.03}$	55.32$_{\pm0.61}$	1.70$_{\pm0.03}$	51.09$_{\pm0.49}$	1.90$_{\pm0.03}$	44.77$_{\pm0.42}$	2.21$_{\pm0.03}$
SWAG	59.19$_{\pm0.90}$	2.00$_{\pm0.03}$	55.45$_{\pm0.88}$	2.12$_{\pm0.03}$	53.34$_{\pm0.94}$	2.19$_{\pm0.03}$	49.44$_{\pm0.81}$	2.33$_{\pm0.03}$	43.71$_{\pm0.93}$	2.53$_{\pm0.03}$
F-SWAG	59.55$_{\pm2.94}$	1.49$_{\pm0.11}$	55.10$_{\pm2.82}$	1.70$_{\pm0.10}$	52.37$_{\pm2.80}$	1.82$_{\pm0.10}$	48.18$_{\pm2.63}$	2.04$_{\pm0.09}$	41.84$_{\pm2.43}$	2.37$_{\pm0.09}$
E-MCMC	62.28$_{\pm0.47}$	1.40$_{\pm0.02}$	57.84$_{\pm0.46}$	1.59$_{\pm0.02}$	55.14$_{\pm0.29}$	1.71$_{\pm0.02}$	50.87$_{\pm0.21}$	1.91$_{\pm0.02}$	44.49$_{\pm0.13}$	2.22$_{\pm0.02}$
PTL	61.84$_{\pm0.33}$	1.47$_{\pm0.02}$	57.36$_{\pm0.22}$	1.66$_{\pm0.02}$	54.47$_{\pm0.08}$	1.78$_{\pm0.01}$	50.03$_{\pm0.23}$	1.98$_{\pm0.01}$	43.34$_{\pm0.36}$	2.29$_{\pm0.01}$
FP-BMA	**63.91**$_{\pm0.02}$	**1.33**$_{\pm0.00}$	**59.70**$_{\pm0.00}$	**1.51**$_{\pm0.00}$	**57.00**$_{\pm0.01}$	**1.63**$_{\pm0.00}$	**52.51**$_{\pm0.03}$	**1.84**$_{\pm0.00}$	**45.39**$_{\pm0.04}$	**2.18**$_{\pm0.00}$

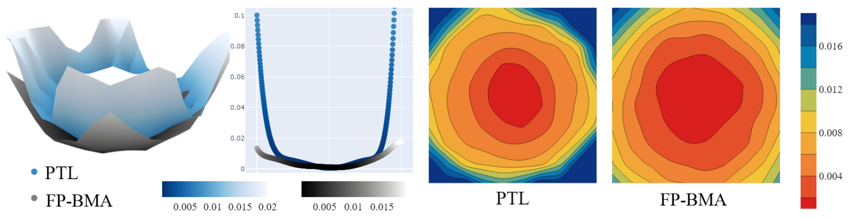
B.8 Loss Surface Of Sampled Model



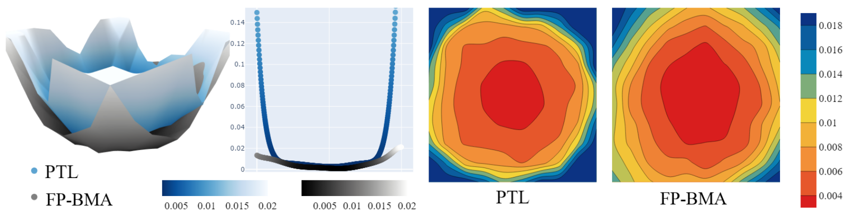
(a) seed 1



(b) seed 2



(c) seed 3



(d) seed 4

Figure 6: Four instances of sampled weights, including (b) as presented in Figure 3. Across all plots, it is consistently observed that FP-BMA converges to a flatter loss surface compared to PTL.

As shown in Figure 3, we sampled four model parameters from the posterior, which were trained on CIFAR10 with RN18. It shows the consistent and robust trend of flatness of FP-BMA in the loss surface. In Figure 6, commencing with the leftmost panel, a 3D surface plot illustrates the loss surface, revealing the FP-BMA model's comparatively flatter topology against the PTL model. This initial plot intuitively demonstrates that the FP-BMA model exhibits a flatter loss surface compared to the PTL model. Following this, the second visualization compresses the information along a diagonal plane into a 1D scatter plot. This transformation reveals areas obscured in the 3D view, highlighting that FP-BMA maintains a considerably flatter and lower-loss landscape. The third and fourth images showcase the loss surface through 2D contour plots, from which one can easily discern that the area representing the lowest loss is significantly more expansive for FP-BMA than for PTL.