Flat Posterior Does Matter 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]RIKEN AIP

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[a].

[a]Code is available at https://github.com/MLAI-Yonsei/FP-BMA

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 and 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 and Izmailov, 2020, Zeng and 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 and 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 and 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.

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

[*]Corresponding Authors
[†]Work conducted while affiliated with KAIST
[‡]Correspondence to: kyungwoo.song@gmail.com

respect to the loss. We show that the proposed FP-BMA is an extended version of previous flatness-aware optimizers, Sharpness-aware Minimization (SAM) [Foret et al., 2020], Fisher SAM (FSAM) [Kim et al., 2022a], and Natural Gradient (NG) [Amari, 1998] with specific conditions. 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.

Our major contributions are summarized as follows:

- We demonstrate that BNNs often struggle to capture the flatness, and BMA can be ineffective without flatness.
- We propose FP-BMA, a flat posterior-seeking optimizer that generalizes loss geometric optimizers such as SAM, FSAM, and NG.
- We introduce Flat Posterior-aware Bayesian Transfer Learning, which leverages a pre-trained model as a prior and effectively enhances robustness against model misspecification through 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 and Teh, 2011, Chen et al., 2014] and Variational Inference (VI) [Graves, 2011, Blundell et al., 2015], and other variants [Ritter et al., 2018, Gal and 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 $q_\theta(w|\mathcal{D})$, BNNs make predictions on unobserved data (x^*, y^*) via *Bayesian Model Averaging (BMA)*, which integrates predictions over the posterior distribution of the model parameters:

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

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. This approach is known to improve generalization by averaging over a diverse set of models sampled from the approximate posterior, which is the core idea of BMA [Wilson and Izmailov, 2020].

2.2 FLATNESS AND OPTIMIZATION

As the flatness of loss surface has been known to be connected to the generalizability [Hochreiter and Schmidhuber, 1994, 1997, Neyshabur et al., 2017], new training methods have been presented to find the flat local optimum. Sharpness-Aware Minimization (SAM) [Foret et al., 2020] is a widely adopted technique that seeks flat minima by making the model robust to small perturbations in parameters. SAM performs adversarial training by minimizing the worst-case loss in an L_2 neighborhood of the weights:

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

where $\ell(\cdot)$ is the empirical loss function (e.g., cross-entropy for classification tasks) and p is practically set to $p = 2$, yielding $\Delta w = \gamma \nabla_w \ell(w)/\|\nabla_w \ell(w)\|_2$.

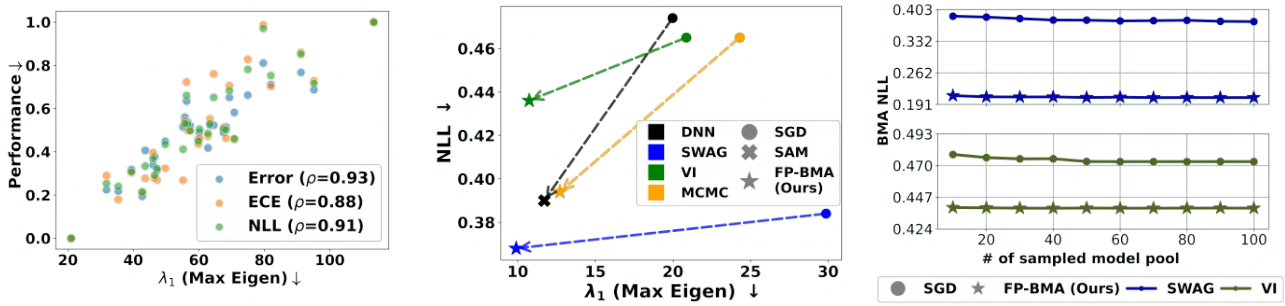
However, SAM's isotropic L_2 ball may not accurately reflect the an isotropic geometry of the loss landscape. To address this, Fisher SAM (FSAM) [Kim et al., 2022a] improves SAM by replacing the Euclidean ball with a non-Euclidean one defined by the Fisher information matrix (FIM):

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

where $F_y(w)$ denotes the 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.



(a) Correlation between flatness and generalization within sampled models

(b) Flatness and generalization according to the training methods

(c) NLL with the number of BMA models

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. (c) Flat posteriors are necessary, as increasing the number of BMA samples does not lead to better performance if the posterior is not flat. The proposed FP-BMA enhances posterior quality by capturing flatness and requires fewer samples for improved BMA.

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 and Szegedy, 2015] on CIFAR10 [Krizhevsky et al., 2009] using following Bayesian inference methods-VI, SWAG, and MCMC-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:

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

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.* We confirm that this property holds across different learning rate schedulers and the CIFAR-100 dataset, as shown in Figure 6 (Appendix A.2.1).

Takeaway 2: It is essential to approximate a flat posterior for BMA. We also establish a generalization error bound for BMA that explicitly involves the flatness of the posterior. First, we 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.

Lemma 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:*

$$\max\left(\left\{\frac{1}{M}\left(\lambda_{\max}(H_{f_m}) + \sum_{\substack{n=1\\ n\neq m}}^{M} \lambda_{\min}(H_{f_n})\right)\right\}_{m=1}^{M}\right) \quad (4)$$

$$\leq \lambda_{\max}(H_{f_{BMA}}) \leq \frac{\sum_{m=1}^{M} \lambda_{\max}(H_{f_m})}{M}. \quad (5)$$

Lemma 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 that that the BMA prediction operates within flatter minima. Given Lemma 1, the following theorem shows that the generalization error of the BMA predictor is directly controlled by the flatness of the posterior, as measured by the maximal Hessian eigenvalue [Luo et al., 2024].

Theorem 1 (Informal)**.** *Let f_{BMA} be the BMA predictor obtained by averaging over posterior samples. Then, with high probability,*

$$\ell_{\mathcal{D}}(f_{BMA}) \leq \ell_{\mathcal{S}}(f_{BMA}) + \frac{p\sigma^2}{2}\lambda_{\max}(H_{f_{BMA}}) + O\left(\sigma^3 p^3\right)$$

where $\ell_{\mathcal{D}}$ and $\ell_{\mathcal{S}}$ denote the population and empirical loss, respectively, p is the number of model parameters, n is the sample size, σ is a smoothing parameter, and $H_{f_{BMA}}$ is the Hessian of the loss evaluated at f_{BMA}.

This result formally supports our main message: *BMA with a flat posterior—that is, a posterior whose samples f_m exhibit smaller $\lambda_{\max}(H_{f_m})$—leads to a tighter generalization error bound.* Thus, posterior flatness is not only empirically correlated with generalization, but also a theoretically well-justified objective for BMA. A detailed proof of Theorem 1 is provided in Appendix B.1.

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. We also confirm consistent results on various learning rate schedulers and generalization performance metrics, as described in Appendix A.3.

Takeaway 4: Increasing the number of BMA samples does not lead to better performance without a flat posterior. Figure 1c compares the NLL of BMA predictions for two posteriors—one considering flatness and the other not. The results show that *simply increasing the number of weight samples in BMA does not outperform BMA with a flat posterior*, highlighting the importance of posterior flatness for better generalization. On the other hand, the proposed FP-BMA, which will be described in Section 4.1, enhances posterior quality by capturing flatness and requires fewer samples for improved BMA. Additional results in Appendix A.4 confirm this trend across different learning rate schedulers and metrics.

4 BAYESIAN MODEL AVERAGING WITH FLAT POSTERIOR

Our theoretical analysis and empirical findings suggest that a flat posterior in BNNs is crucial for generalization but is not achieved by existing approximate Bayesian inference methods. To address this, we propose an optimizer that encourages a flat posterior (Section 4.1) introduce 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:

$$\begin{aligned}\ell^{\gamma}_{\text{FP-BMA}}(\theta) = \min_{\theta} \max_{d|\theta+\Delta\theta,\theta|\leq\gamma^2} \ell(\theta+\Delta\theta) \\ + \beta \mathrm{D}_{\mathrm{KL}}[q_\theta(w|\mathcal{D})||p(w)]\end{aligned} \tag{6}$$

$$\text{s.t. } d|\theta+\Delta\theta,\theta| = \mathrm{D}_{\mathrm{KL}}\left[q_{\theta+\Delta\theta}(w|\mathcal{D}) \,||\, q_\theta(w|\mathcal{D})\right], \tag{7}$$

where θ and $\Delta\theta$ denote the variational parameters and their perturbation, respectively. $\ell(\theta+\Delta\theta)$ denotes empirical loss under perturbated posterior $q_{\theta+\Delta\theta}(w|\mathcal{D})$, and β is a hyper-parameter that controls the influence of the prior.

Given new objective $\ell^{\gamma}_{\text{FP-BMA}}(\theta)$ in Eq. 6, the variational parameter θ is updated using the approximate gradient

$$\nabla_\theta \ell^{\gamma}_{\text{FP-BMA}}(\theta) \approx \nabla_\theta \ell(\theta + \Delta\theta_{\text{FP-BMA}}), \tag{8}$$

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

$$\Delta\theta_{\text{FP-BMA}} = \gamma \frac{F_\theta(\theta)^{-1}\nabla_\theta \ell(\theta)}{\sqrt{\nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1} \nabla_\theta \ell(\theta)}}, \tag{9}$$

using FIM $F_\theta(\theta) = \mathbb{E}_{w,\mathcal{D}}[\nabla_\theta \log q_\theta(w|\mathcal{D}) \nabla_\theta \log q_\theta(w|\mathcal{D})^T]$. After that, the gradient $\nabla_\theta \ell(\theta)$ is evaluated at $\theta + \Delta\theta_{\text{FP-BMA}}$. We notate our objective as FP-BMA and provide a detailed formula derivation in Appendix B.2.

Our proposed FP-BMA optimizer offers several key advantages, which are detailed in the following paragraphs. The main advantages of FP-BMA are summarized as follows:

- **Implicit Flatness Control**
- **KL-based Bayesian Perturbation Ball**
- **Generalized Version of Geometric Optimizers**

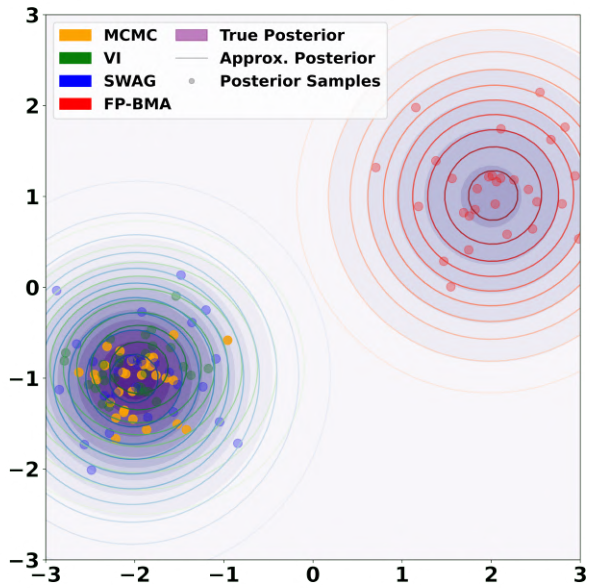


Figure 2: Posterior approximation with synthetic example. When both flat and sharp modes coexist, we compared how optimizers approximate the posterior. Unlike other methods, the proposed FP-BMA converged to the flat mode.

Implicit Flatness Control Eq. 6 implicitly controls sharpness by penalizing solutions that are sensitive to parameter perturbations. This can be formalized via a second-order Taylor expansion of the inner maximization:

$$\max_{d|\theta+\Delta\theta,\theta|\leq\gamma^2} \ell(\theta+\Delta\theta) \approx \max_{d|\theta+\Delta\theta,\theta|\leq 1} \left(\ell(\theta) + \gamma^2 \Delta\theta^\top \nabla_\theta \ell(\theta) + \frac{\gamma^4 \lambda_1(H_{f_\theta})}{2}\right) \tag{10}$$

where $\lambda_1(H_{f_\theta})$ is the maximal eigenvalue of the Hessian. Thus, minimizing Eq. 6 inherently seeks solutions with lower sharpness, ensuring the variational posterior is concentrated in flatter regions of the loss landscape.

KL-based Bayesian Perturbation Ball Unlike deterministic optimizers such as SAM and FSAM, our method constrains the perturbation in distributional space via the KL-divergence, as shown in Eq. 7. This approach leverages local curvature information without expensive inner-loop optimization, making the method scalable and practical for large models. In addition, for Gaussian variational posteriors, the KL-based constraint naturally captures both mean and variance changes—providing a richer and more Bayesian-consistent notion of flatness.

Generalized version of geometric optimizers FP-BMA is a generalized version of SAM and FSAM and approximates NG under deterministic parameters, as shown in Theorem 2. Proof of Theorem 2 is provided in Appendix B.3.

Theorem 2. *(Informal) Suppose the model parameter w is deterministic and the loss function $\ell(\cdot)$ is twice continuously differentiable. Let $\gamma' = \gamma/\sqrt{\nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1} \nabla_\theta \ell(\theta)}$, then*

i) FP-BMA degenerates to SAM if FIM is an identity matrix.

ii) FP-BMA degenerates to FSAM by using the diagonal terms of FIM.

iii) FP-BMA approximates the update rule of NG with learning rate $\eta_{FP\text{-}BMA} = \frac{\eta_{NG}}{(1+\gamma')} F_\theta(\theta)^{-1}$, where η_{NG} denotes the learning rate of NG.

This unifying perspective implies that FP-BMA can seamlessly adapt to both deterministic and Bayesian scenarios, providing a principled way to leverage geometric properties of the loss landscape in probabilistic models. As a result, FP-BMA inherits the empirical benefits of sharpness-aware and natural gradient optimizers—such as improved generalization and robustness—while extending their applicability to Bayesian neural networks in a theoretically grounded manner.

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:

$$\begin{aligned} \ell^{\gamma}_{\text{FP-BMA}}(\theta) = \min_{\theta} \max_{d|\theta+\Delta\theta,\theta|\leq\gamma^2} \ell(\theta+\Delta\theta) \\ + \beta \mathrm{D}_{\mathrm{KL}}[q_\theta(w|\mathcal{D}^{\text{ft}})||q_\theta^{\text{pr}}(w|\mathcal{D}^{\text{pr}})] \end{aligned} \tag{11}$$

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

where $\mathcal{D}^{\text{ft}}$ is the downstream dataset. Intuitively, this objective replaces the prior distribution of Eq. 7 by the approximate posterior $q_\theta^{\text{pr}}(w|\mathcal{D}^{\text{pr}})$ on source dataset. Notably, *the proposed objective $\ell^{\gamma}_{FP\text{-}BMA}(\theta)$ can be effective in general transfer learning where the model misspecification [Müller, 2013, Wilson and 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].*

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., 2021, 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 C.4).

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**$_{\pm 0.050}$

5 EXPERIMENTS

5.1 SYNTHETIC EXAMPLE

We demonstrate whether the proposed FP-BMA can estimate a flat posterior when a sharp and flat minima coexists. To this end, we consider a synthetic dataset generated from true posterior having flat mode and sharp mode, as depicted in Figure 2. This controlled setting allows us to directly observe the optimizer's preference for flat versus sharp regions in the loss landscape, isolating the effect of posterior flatness from other confounding factors. We then estimate the posterior using the proposed loss FP-BMA using SWAG. For comparison, we consider the following baseline methods: SGD, MCMC, SWAG, and VI, to estimate posterior. Figure 2 shows that MCMC, SWAG, and VI yield the posterior at sharp mode. In contrast, the proposed FP-BMA captures the flat posterior, demonstrating its effectiveness in identifying solutions with better generalization potential. We provide additional results in Appendix C.1.

5.2 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 models, we apply variational distributions to the parameters of normalization and last layers. We then train these variational parameters using approximate Bayesian inference (VI, MCMC, and SWAG) with the gradient $\nabla_\theta \ell^{\gamma}_{\text{FP-BMA}}(\theta)$ in Eq. 8, while updating the remaining parameters using the gradient $\nabla_\theta \ell(\theta)$. This setup allows us to assess the benefits of FP-BMA in both convolutional and transformer-based architectures under realistic training constraints.

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 and Khan, 2022], and E-MCMC [Li and Zhang, 2023], which utilizes SGLD. For fair comparison, we use the same BNN architecture employed for FP-BMA. All baseline methods are carefully tuned with respect to their key hyperparameters to ensure a fair and meaningful comparison of generalization performance, and the detailed hyperparameter configurations for each baseline are provided in Appendix C.2.2.

Table 1 showcases the generalization performance, including accuracy (ACC), ECE, and NLL. The 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 C.2.

5.3 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 C.3.

Table 2 shows FP-BMA with diverse BNN frameworks consistently outperforms existing baselines in terms of both accuracy and uncertainty quantification. Unlike scratch learn-

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$_{\pm 0.32}$	**0.062**$_{\pm 0.006}$	1.302$_{\pm 0.020}$	44.29$_{\pm 0.83}$	**0.025**$_{\pm 0.005}$	2.133$_{\pm 0.043}$	84.37$_{\pm 1.47}$	0.056$_{\pm 0.061}$	0.503$_{\pm 0.038}$	68.78$_{\pm 0.21}$	0.143$_{\pm 0.007}$	1.193$_{\pm 0.019}$
SAM	56.54$_{\pm 2.57}$	0.129$_{\pm 0.013}$	1.354$_{\pm 0.089}$	**44.51**$_{\pm 0.07}$	0.065$_{\pm 0.007}$	**2.089**$_{\pm 0.013}$	84.35$_{\pm 0.81}$	**0.035**$_{\pm 0.012}$	**0.486**$_{\pm 0.023}$	**68.93**$_{\pm 0.37}$	0.153$_{\pm 0.005}$	1.200$_{\pm 0.021}$
FSAM	54.04$_{\pm 4.11}$	0.139$_{\pm 0.010}$	1.432$_{\pm 0.068}$	44.07$_{\pm 1.21}$	0.056$_{\pm 0.005}$	2.159$_{\pm 0.064}$	**84.51**$_{\pm 0.50}$	0.073$_{\pm 0.085}$	0.517$_{\pm 0.061}$	68.74$_{\pm 0.39}$	**0.110**$_{\pm 0.007}$	**1.166**$_{\pm 0.024}$
bSAM	**56.56**$_{\pm 1.18}$	0.083$_{\pm 0.006}$	**1.280**$_{\pm 0.027}$	43.93$_{\pm 0.48}$	0.060$_{\pm 0.003}$	2.167$_{\pm 0.026}$	82.85$_{\pm 2.10}$	0.113$_{\pm 0.008}$	0.583$_{\pm 0.062}$	68.42$_{\pm 0.40}$	0.148$_{\pm 0.019}$	1.219$_{\pm 0.031}$
MOPED	57.29$_{\pm 1.20}$	0.093$_{\pm 0.006}$	1.297$_{\pm 0.045}$	44.30$_{\pm 0.42}$	**0.047**$_{\pm 0.006}$	2.127$_{\pm 0.005}$	84.50$_{\pm 1.36}$	**0.023**$_{\pm 0.009}$	0.474$_{\pm 0.038}$	68.80$_{\pm 0.77}$	0.111$_{\pm 0.001}$	1.165$_{\pm 0.029}$
FP-BMA (VI)	**64.98**$_{\pm 1.37}$	**0.016**$_{\pm 0.007}$	**0.997**$_{\pm 0.046}$	**49.09**$_{\pm 1.38}$	0.071$_{\pm 0.004}$	**1.893**$_{\pm 0.036}$	**87.56**$_{\pm 1.10}$	0.044$_{\pm 0.012}$	**0.397**$_{\pm 0.026}$	**71.37**$_{\pm 0.36}$	**0.060**$_{\pm 0.007}$	**1.023**$_{\pm 0.012}$
MCMC	56.31$_{\pm 1.27}$	0.083$_{\pm 0.003}$	1.305$_{\pm 0.063}$	44.28$_{\pm 0.95}$	0.021$_{\pm 0.002}$	2.155$_{\pm 0.038}$	83.93$_{\pm 1.33}$	0.069$_{\pm 0.010}$	0.523$_{\pm 0.039}$	66.48$_{\pm 1.18}$	0.077$_{\pm 0.011}$	1.224$_{\pm 0.044}$
PTL	57.26$_{\pm 1.44}$	0.116$_{\pm 0.003}$	1.345$_{\pm 0.004}$	43.00$_{\pm 1.05}$	0.120$_{\pm 0.006}$	2.383$_{\pm 0.062}$	**85.76**$_{\pm 1.37}$	0.080$_{\pm 0.014}$	0.482$_{\pm 0.027}$	65.52$_{\pm 2.45}$	**0.056**$_{\pm 0.006}$	1.260$_{\pm 0.095}$
E-MCMC	56.69$_{\pm 2.14}$	0.142$_{\pm 0.004}$	1.266$_{\pm 0.054}$	41.57$_{\pm 0.04}$	0.046$_{\pm 0.012}$	2.370$_{\pm 0.175}$	83.91$_{\pm 1.16}$	0.333$_{\pm 0.010}$	0.877$_{\pm 0.044}$	63.40$_{\pm 0.01}$	0.280$_{\pm 0.008}$	1.655$_{\pm 0.024}$
FP-BMA (MCMC)	**57.49**$_{\pm 0.64}$	**0.039**$_{\pm 0.00.}$	**1.248**$_{\pm 0.048}$	**45.72**$_{\pm 0.56}$	**0.016**$_{\pm 0.003}$	**2.062**$_{\pm 0.050}$	84.82$_{\pm 1.84}$	**0.051**$_{\pm 0.018}$	**0.449**$_{\pm 0.048}$	**68.73**$_{\pm 1.09}$	0.061$_{\pm 0.004}$	**1.117**$_{\pm 0.042}$
SWAG	56.31$_{\pm 0.60}$	0.094$_{\pm 0.013}$	1.315$_{\pm 0.056}$	44.14$_{\pm 1.28}$	**0.034**$_{\pm 0.010}$	2.161$_{\pm 0.058}$	83.51$_{\pm 2.22}$	0.022$_{\pm 0.015}$	0.510$_{\pm 0.072}$	68.72$_{\pm 0.45}$	0.065$_{\pm 0.005}$	1.136$_{\pm 0.014}$
F-SWAG	57.65$_{\pm 1.20}$	0.075$_{\pm 0.003}$	1.249$_{\pm 0.038}$	46.09$_{\pm 0.44}$	0.062$_{\pm 0.006}$	2.089$_{\pm 0.002}$	83.87$_{\pm 1.28}$	0.013$_{\pm 0.005}$	0.492$_{\pm 0.040}$	68.84$_{\pm 0.77}$	0.076$_{\pm 0.012}$	1.137$_{\pm 0.020}$
FP-BMA (SWAG)	**61.79**$_{\pm 4.34}$	**0.026**$_{\pm 0.004}$	**1.214**$_{\pm 0.119}$	**47.45**$_{\pm 0.60}$	0.055$_{\pm 0.018}$	**2.044**$_{\pm 0.022}$	**86.81**$_{\pm 0.78}$	**0.010**$_{\pm 0.003}$	**0.399**$_{\pm 0.034}$	**70.10**$_{\pm 0.18}$	**0.045**$_{\pm 0.015}$	**1.063**$_{\pm 0.023}$

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$_{\pm 1.47}$	93.16$_{\pm 0.27}$	89.95$_{\pm 0.51}$	66.34$_{\pm 0.59}$	84.05$_{\pm 0.33}$	81.25$_{\pm 1.03}$	91.24$_{\pm 0.83}$	88.68$_{\pm 0.92}$	68.64$_{\pm 0.51}$	82.45$_{\pm 0.56}$
SAM	87.85$_{\pm 0.49}$	94.80$_{\pm 0.17}$	90.23$_{\pm 0.78}$	70.40$_{\pm 0.76}$	85.82$_{\pm 0.25}$	82.53$_{\pm 0.65}$	93.08$_{\pm 0.87}$	90.66$_{\pm 0.74}$	70.66$_{\pm 1.03}$	84.23$_{\pm 0.60}$
SWAG	88.97$_{\pm 1.56}$	93.27$_{\pm 0.15}$	89.95$_{\pm 0.46}$	66.41$_{\pm 0.30}$	84.65$_{\pm 0.37}$	81.62$_{\pm 0.66}$	91.21$_{\pm 0.91}$	88.67$_{\pm 0.42}$	67.65$_{\pm 0.45}$	82.29$_{\pm 0.31}$
F-SWAG	90.03$_{\pm 1.08}$	94.84$_{\pm 0.26}$	90.12$_{\pm 0.57}$	70.00$_{\pm 0.87}$	86.25$_{\pm 0.19}$	82.72$_{\pm 0.49}$	92.93$_{\pm 0.93}$	90.60$_{\pm 0.55}$	68.67$_{\pm 0.39}$	83.73$_{\pm 0.35}$
MOPED	85.21$_{\pm 3.14}$	92.15$_{\pm 0.73}$	89.25$_{\pm 0.61}$	65.85$_{\pm 0.99}$	83.11$_{\pm 0.86}$	83.97$_{\pm 0.49}$	91.71$_{\pm 0.87}$	89.90$_{\pm 0.54}$	69.66$_{\pm 0.53}$	83.81$_{\pm 0.51}$
PTL	90.01$_{\pm 0.39}$	92.55$_{\pm 0.53}$	89.43$_{\pm 0.41}$	65.00$_{\pm 1.24}$	84.25$_{\pm 0.30}$	83.76$_{\pm 0.61}$	88.43$_{\pm 1.27}$	88.54$_{\pm 0.53}$	60.38$_{\pm 1.84}$	80.28$_{\pm 0.03}$
FP-BMA	**90.16**$_{\pm 1.04}$	**95.85**$_{\pm 1.26}$	**90.23**$_{\pm 0.58}$	**71.57**$_{\pm 0.27}$	**86.95**$_{\pm 0.65}$	**84.60**$_{\pm 0.25}$	**94.15**$_{\pm 0.80}$	**91.30**$_{\pm 0.25}$	**72.63**$_{\pm 1.12}$	**85.67**$_{\pm 0.14}$

ing, 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.

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 and 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 C.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.

5.4 ROBUSTNESS ON DISTRIBUTION SHIFT

We evaluate the trained models on CIFAR10 and CIFAR100 10-shots using the corrupted datasets CIFAR10C and CIFAR100C [Hendrycks and Dietterich, 2019] to demonstrate the robustness of FP-BMA. These benchmarks simulate a wide variety of real-world corruptions, including noise, blur, weather, and digital effects, thereby providing a comprehensive testbed for evaluating model reliability under distribution shift.

Figure 3 presents the accuracy on the corrupted datasets CIFAR10C and CIFAR100C [Hendrycks and 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 12. Detailed results are provided in Appendix C.7.

Table 4: Downstream task accuracy of CLIP with visual encoder, ResNet50 and ViT-B/16. **Bold** and underline denote best and second best performance each. FP-BMA shows superior performance in average over five datasets.

Backbone	ResNet50						ViT-B/16					
Method	IN	IN-V2	IN-R	IN-A	IN-S	Avg	IN	IN-V2	IN-R	IN-A	IN-S	Avg
Zero-Shot	59.83$_{\pm 0.00}$	52.89$_{\pm 0.00}$	60.73$_{\pm 0.00}$	23.25$_{\pm 0.00}$	35.45$_{\pm 0.00}$	46.43$_{\pm 0.00}$	68.33$_{\pm 0.00}$	61.91$_{\pm 0.00}$	77.71$_{\pm 0.00}$	49.93$_{\pm 0.00}$	48.22$_{\pm 0.00}$	61.22$_{\pm 0.00}$
SGD	61.70$_{\pm 0.01}$	54.31$_{\pm 0.01}$	60.87$_{\pm 0.01}$	22.74$_{\pm 0.01}$	35.68$_{\pm 0.00}$	47.06$_{\pm 0.01}$	69.97$_{\pm 0.00}$	62.97$_{\pm 0.01}$	78.05$_{\pm 0.00}$	50.31$_{\pm 0.02}$	48.76$_{\pm 0.00}$	62.01$_{\pm 0.00}$
SAM	61.73$_{\pm 0.01}$	54.35$_{\pm 0.01}$	60.86$_{\pm 0.01}$	22.76$_{\pm 0.01}$	35.67$_{\pm 0.00}$	47.07$_{\pm 0.01}$	70.01$_{\pm 0.01}$	63.03$_{\pm 0.02}$	78.03$_{\pm 0.01}$	50.37$_{\pm 0.00}$	48.75$_{\pm 0.00}$	62.04$_{\pm 0.00}$
SWAG	61.77$_{\pm 0.22}$	54.10$_{\pm 0.19}$	**61.25**$_{\pm 0.21}$	**23.25**$_{\pm 0.08}$	35.55$_{\pm 0.27}$	47.18$_{\pm 0.19}$	70.11$_{\pm 0.02}$	63.44$_{\pm 0.06}$	**78.33**$_{\pm 0.03}$	**50.55**$_{\pm 0.02}$	48.95$_{\pm 0.01}$	62.28$_{\pm 0.02}$
FP-BMA	**63.33**$_{\pm 0.92}$	**55.06**$_{\pm 0.79}$	61.14$_{\pm 0.37}$	22.78$_{\pm 0.68}$	**35.82**$_{\pm 0.11}$	**47.63**$_{\pm 0.17}$	**72.41**$_{\pm 0.33}$	**64.85**$_{\pm 0.11}$	78.14$_{\pm 0.31}$	50.52$_{\pm 0.25}$	**49.25**$_{\pm 0.03}$	**63.03**$_{\pm 0.04}$

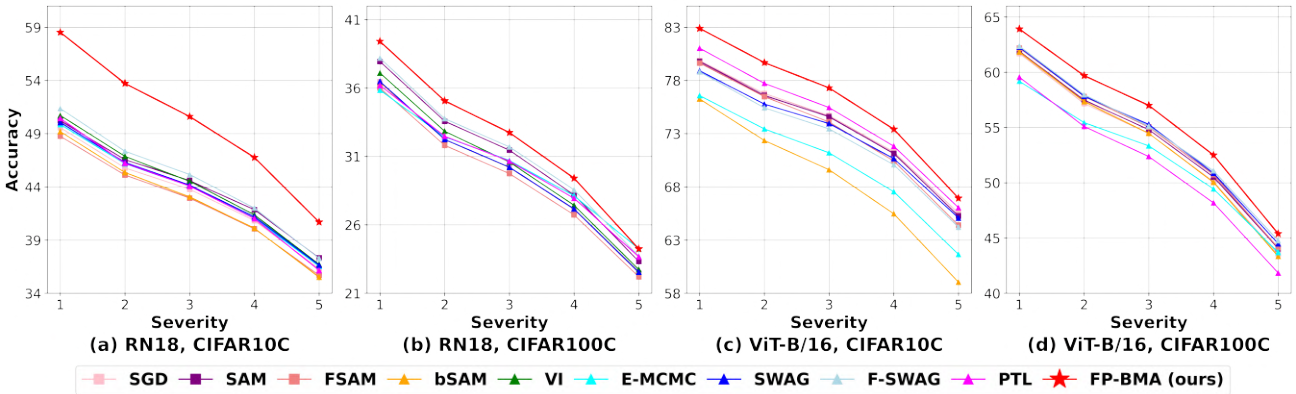


Figure 3: Accuracy under distributional shift. We evaluate the accuracy of RN18 and ViT-B/16 models trained on CIFAR10 and CIFAR100 10-shot across all severity levels of CIFAR10C and CIFAR100C. FP-BMA consistently outperforms all baseline methods across all levels of corruption.

The results on IN variants in Table 4 and the corrupted datasets in Figure 3 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.

5.5 FLATNESS ANALYSIS

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 4 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. Additional results and the protocol to visualize the loss basin are provided in Appendix C.9.

Table 5 compares the Hessian's eigenvalue of model λ_i (Eq. 3) where λ_1 and λ_5 represent the largest eigenvalue and the fifth largest eigenvalue, respectively. This result indicates that FP-BMA achieves the lowest values compared to all baselines, implying that the posterior of BNNs is formed on the flattest local surface. This further supports our empirical observations that FP-BMA enhances generalization by encouraging a flatter posterior distribution.

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 and 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 and Zhang, 2023] proposed an efficient MCMC algorithm capable of effectively sampling the posterior within a flat basin

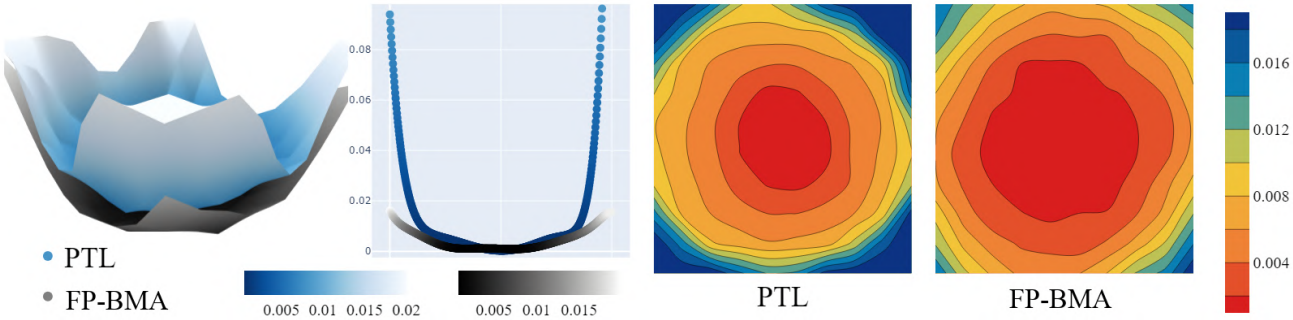


Figure 4: Comparison of the loss surfaces of FP-BMA (grey) and PTL (light blue) models. The comparison of loss surface shows that FP-BMA allows the posterior to be placed on a lower and flatter loss surface compared to that of PTL.

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**

by removing the nested chain of Entropy-SGD [Dziugaite and Roy, 2018] and Entropy-SGLD [Chaudhari et al., 2019]. 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

Applying Bayesian methods to transfer learning is a natural and theoretically well-founded approach, as the Bayesian framework systematically incorporates prior knowledge and quantifies uncertainty when adapting models to new tasks. Theoretical foundations for this perspective can be found in the literature on probabilistic machine learning and hierarchical Bayesian models [Bishop and Nasrabadi, 2006, Murphy, 2012, Gelman et al., 1995], as well as early works on Bayesian transfer and domain adaptation [Lawrence and Platt, 2004, Raina et al., 2006]. Building on these principles, a variety of Bayesian transfer learning methods have been developed, including approaches leveraging pre-trained models as priors, empirical Bayes techniques, and flexible posterior approximations [Krishnan et al., 2020, Shwartz-Ziv et al., 2022, Lee et al., 2024]. 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 demonstrates the limitations of BNNs in capturing flatness—a property crucial for generalization—and reveals that BMA may fail to yield optimal results without considering flatness. To address this, we introduce FP-BMA, which seeks a flat posterior by effectively capturing flatness in the parameter space. FP-BMA generalizes existing sharpness-aware optimizers and aligns with the intrinsic nature of BNNs. We further propose a Flat Posterior-aware Bayesian Transfer Learning scheme, which enhances resilience against model misspecification. Our extensive experiments demonstrate that FP-BMA significantly improves the generalization ability of BNNs, underscoring the importance of flatness in posterior approximations. However, there are several limitations to our study. Specifically, our theoretical insights rely on strong assumptions, and the empirical evaluation does not cover the full spectrum of MCMC algorithms. Future work could extend FP-BMA to a wider variety of Bayesian inference methods and investigate its effectiveness on more complex datasets. Additionally, exploring automated ways to quantify and enforce flatness during model training could further enhance the robustness and applicability of the proposed approach.

ACKNOWLEDGEMENT

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

References

Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pages 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*, pages 1683–1691. PMLR, 2014.

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*, pages 2510–2521. PMLR, 2021.

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*, pages 1377–1386. PMLR, 2018.

Runa Eschenhagen, Erik Daxberger, Philipp Hennig, and Agustinus Kristiadi. Mixtures of laplace approximations for improved post-hoc uncertainty in deep learning. *arXiv preprint arXiv:2111.03577*, 2021.

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*, pages 1050–1059. PMLR, 2016.

Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. *Bayesian data analysis*. Chapman and Hall/CRC, 1995.

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.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

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*, pages 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*, pages 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*, pages 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*, pages 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*, pages 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*, pages 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*, pages 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*, pages 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, pages 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*, pages 529–545. PMLR, 2022b.

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

Neil D Lawrence and John C Platt. Learning to learn with the informative vector machine. In *Proceedings of the twenty-first international conference on Machine learning*, page 65, 2004.

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.

Haocheng Luo, Tuan Truong, Tung Pham, Mehrtash Harandi, Dinh Phung, and Trung Le. Explicit eigenvalue regularization improves sharpness-aware minimization. *Advances in Neural Information Processing Systems*, 37: 4424–4453, 2024.

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.

Martin Marek, Brooks Paige, and Pavel Izmailov. Can a confident prior replace a cold posterior? *arXiv preprint arXiv:2403.01272*, 2024.

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.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

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*, pages 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*, pages 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*, pages 8748–8763. PMLR, 2021.

Rajat Raina, Andrew Y Ng, and Daphne Koller. Constructing informative priors using transfer learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 713–720, 2006.

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*, pages 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*, pages 7694–7722. PMLR, 2023.

Yuesong Shen, Nico Daheim, Bai Cong, Peter Nickl, Gian Maria Marconi, Clement Bazan, Rio Yokota, Iryna Gurevych, Daniel Cremers, Mohammad Emtiyaz Khan, et al. Variational learning is effective for large deep networks. *arXiv preprint arXiv:2402.17641*, 2024.

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)*, pages 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.

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*, pages 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*, pages 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.

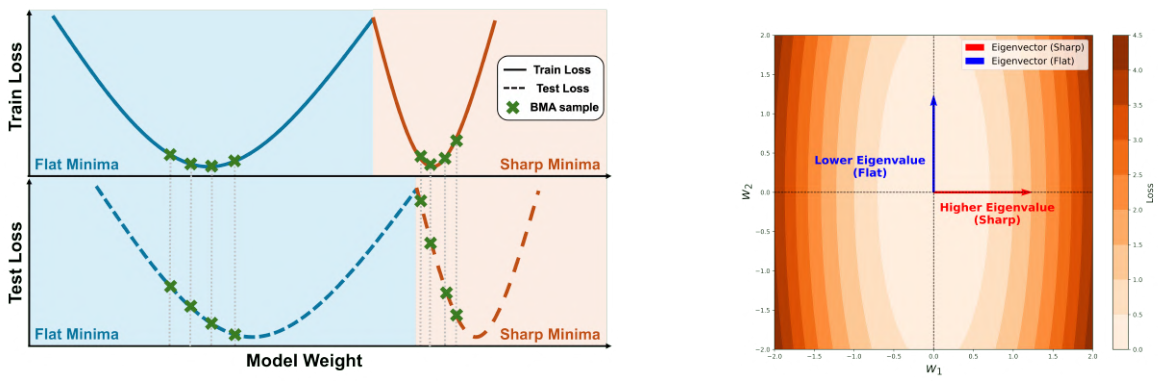
Flat Posterior Does Matter For Bayesian Model Averaging (Appendix)

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]RIKEN AIP

A FLATNESS DOES MATTER FOR BAYESIAN MODEL AVERAGING

A.1 DETAILS ABOUT HESSIAN EIGENVALUE OF LOSS WITH BMA



(a) Flatness of BMA

(b) Hessian Eigenvalue of loss

Figure 5: Description of flatness of BMA and Hessian Eigenvalue of loss. (a) depicts how flatness is measured in BNNs. We measure the flatness of individual sampled model weights and subsequently ensemble the flatness of them. (b) represents how the Hessian eigenvalue of loss corresponds to flatness. It reveals that direction of steep curvature (sharp minima) exhibits with larger eigenvalues, while that of gentle curvature (flat minima) exhibits smaller eigenvalues. Based on this understanding, we measure flatness using the maximal eigenvalue of the Hessian at the minima.

To measure the flatness of BNNs and compare them with DNNs, we introduce a new metric specifically designed for this study. Unlike DNNs, where model parameters are typically treated as point estimate, BNNs represent model parameters as random variables, necessitating an appropriate approach for measuring flatness. As shown in Figure 5b, the maximal eigenvalue of the Hessian of the loss function is commonly used to evaluate flatness quantitatively in DNNs [Keskar et al., 2016, Foret et al., 2020, Jastrzebski et al., 2020]. To assess flatness in BNNs, we followed BMA protocol. BMA samples model weights from the approximated posterior, calculates the outputs of the sampled individual models, and ensemble the outputs, as shown in Figure 5a. Thus, similar to how BMA operates, we measured the flatness of individual model weights and subsequently ensemble these measurements to derive a comprehensive metric.

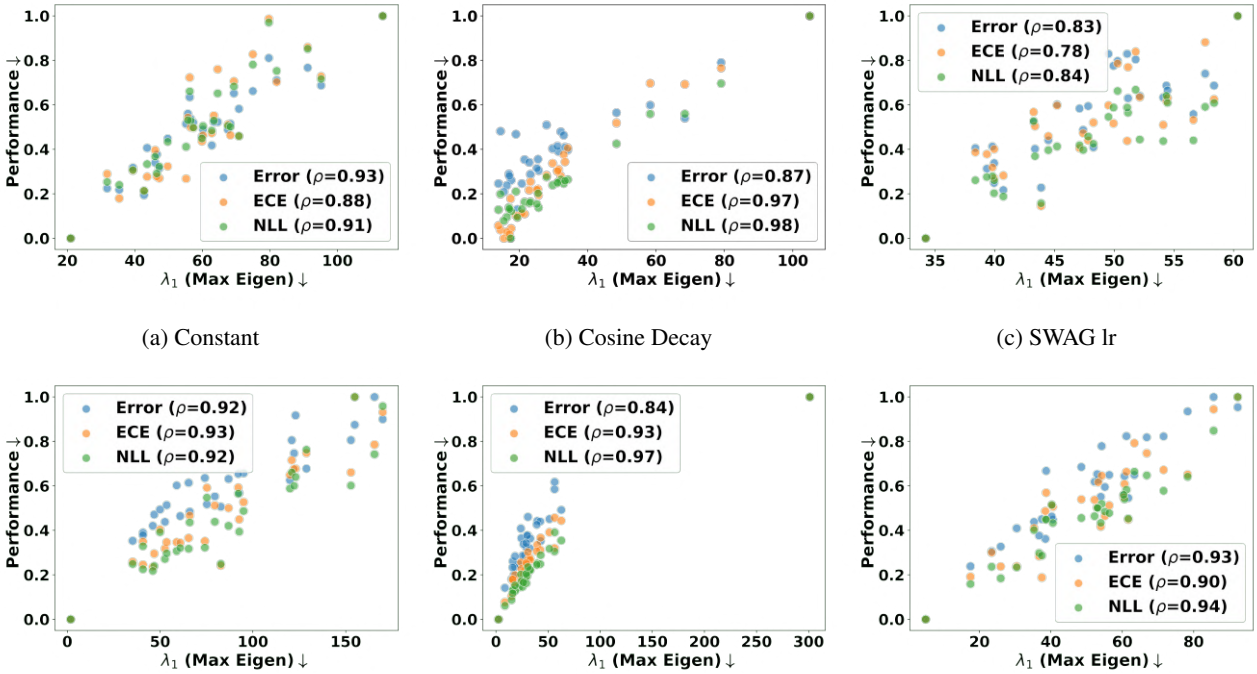
[*]Corresponding Authors
[†]Work conducted while affiliated with KAIST
[‡]Correspondence to: kyungwoo.song@gmail.com

A.2 NEED FOR FLATNESS IN BMA

Experimental Details To measure the flatness of BNNs, M of Eq. 3 is set to 30 for experiments in Section 3.1. We primarily use RN18 as the backbone. Our evaluation includes Error (100−Accuracy), Expected Calibration Error (ECE) [Guo et al., 2017], and Negative Log-Likelihood (NLL) to assess generalization on CIFAR10 and CIFAR100. To minimize confounding effects on flatness measurements, we do not adjust BN and data augmentation. For BNN frameworks, we consider VI, MCMC, and SWAG. We also consider three different learning rate scheduler: Constant, Cosine Decay (Cos Deacy), and SWAG learning rate (SWAG lr).

A.2.1 Correlation Between Flatness And Generalization

We check the correlation between flatness and generalization performance of sampled models throughout all considered learning rate schedulers. We present the scatter plot of the model, sampled from ResNet18 trained on CIFAR10 and CIFAR100 in the first and second rows of Figure 6. Each column of Figure 6 denotes Constant scheduler, Cosine Decay scheduler, and SWAG lr scheduler, respectively. All the models are trained with SWAG and SGD momentum, and we set maximal eigenvalue λ_1 as a flatness measure. Correlation with flatness and each generalization performance metric is suggested in the legend, as well. Regardless of the scheduler and dataset, all generalization performances, error, ECE, and NLL strongly correlate with flatness.



(a) Constant (b) Cosine Decay (c) SWAG lr

(d) Constant (e) Cosine Decay (f) SWAG lr

Figure 6: Correlation between maximal eigenvalue and performances of 30 sampled models from SWAG throughout all considered schedulers. It shows classification error, ECE, and NLL are distinctly correlated with flatness. We conjecture that the flatness is crucial for the generalization performance of BNN

A.3 INSUFFICIENT FLATNESS OF BMA

Figure 7 and Figure 8 show results consistent with Figure 1b across various learning rate schedulers and metrics. Specifically, (1) BNNs struggle to ensure flatness compared to DNNs when using SGD, and (2) the proposed FP-BMA enables BNN frameworks to achieve flat minima, thereby enhancing performance.

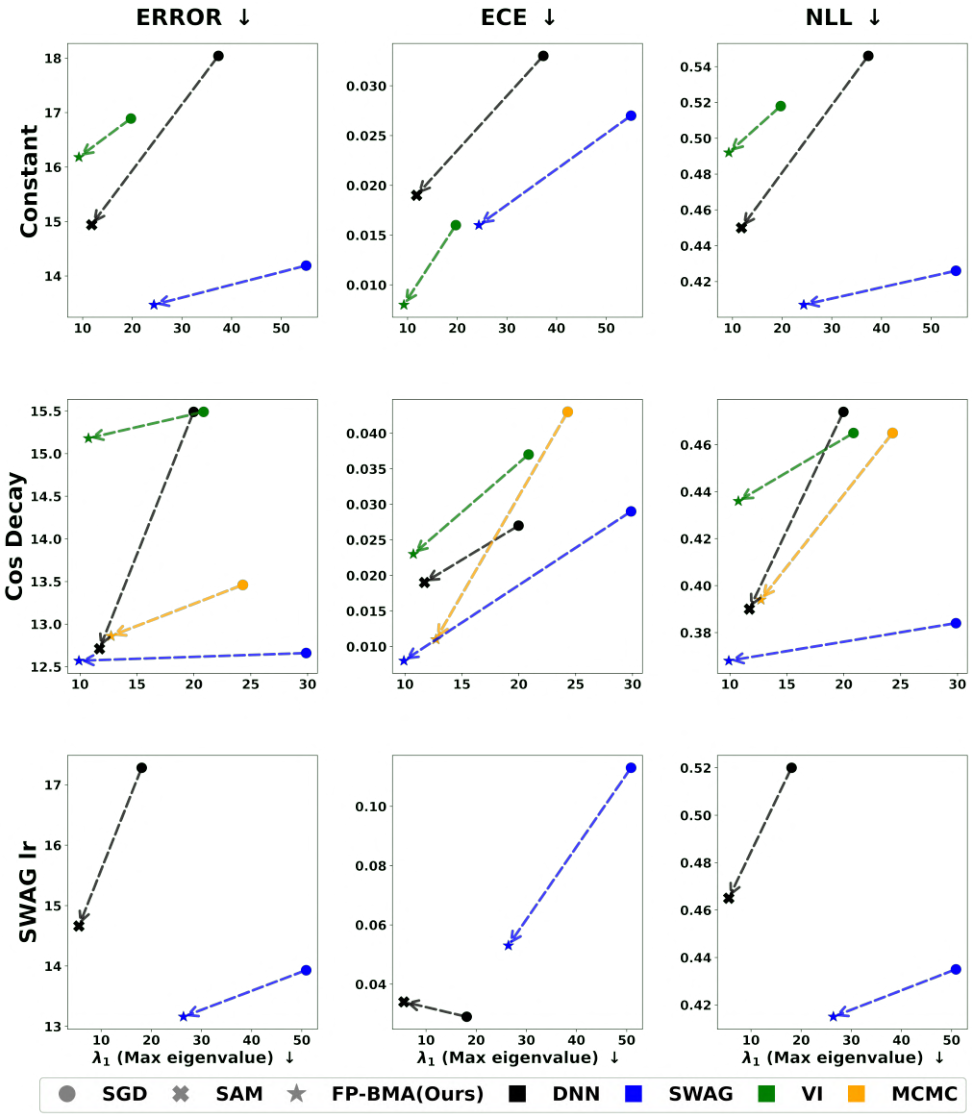


Figure 7: Comparison of Error, NLL, and ECE with various schedulers on CIFAR10 in relation to the maximum eigenvalue λ_1.

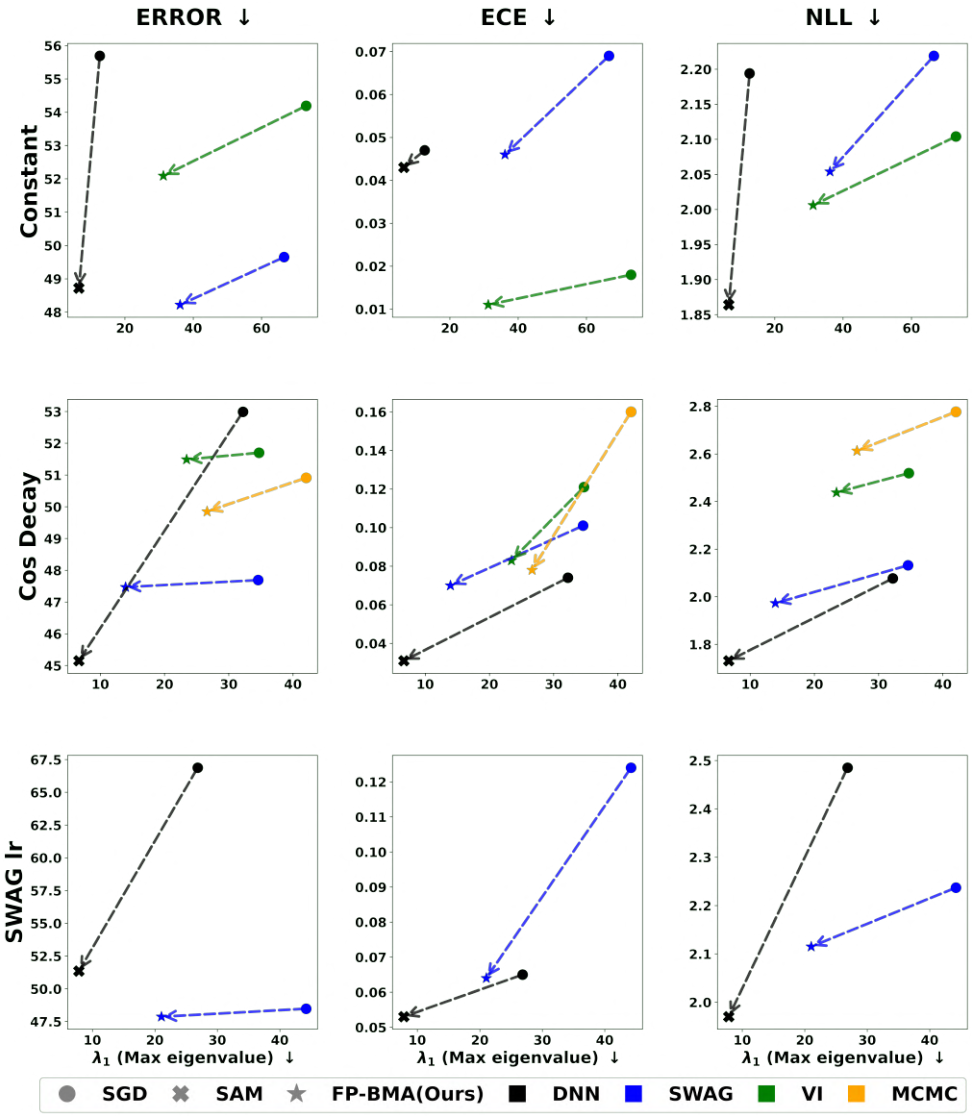
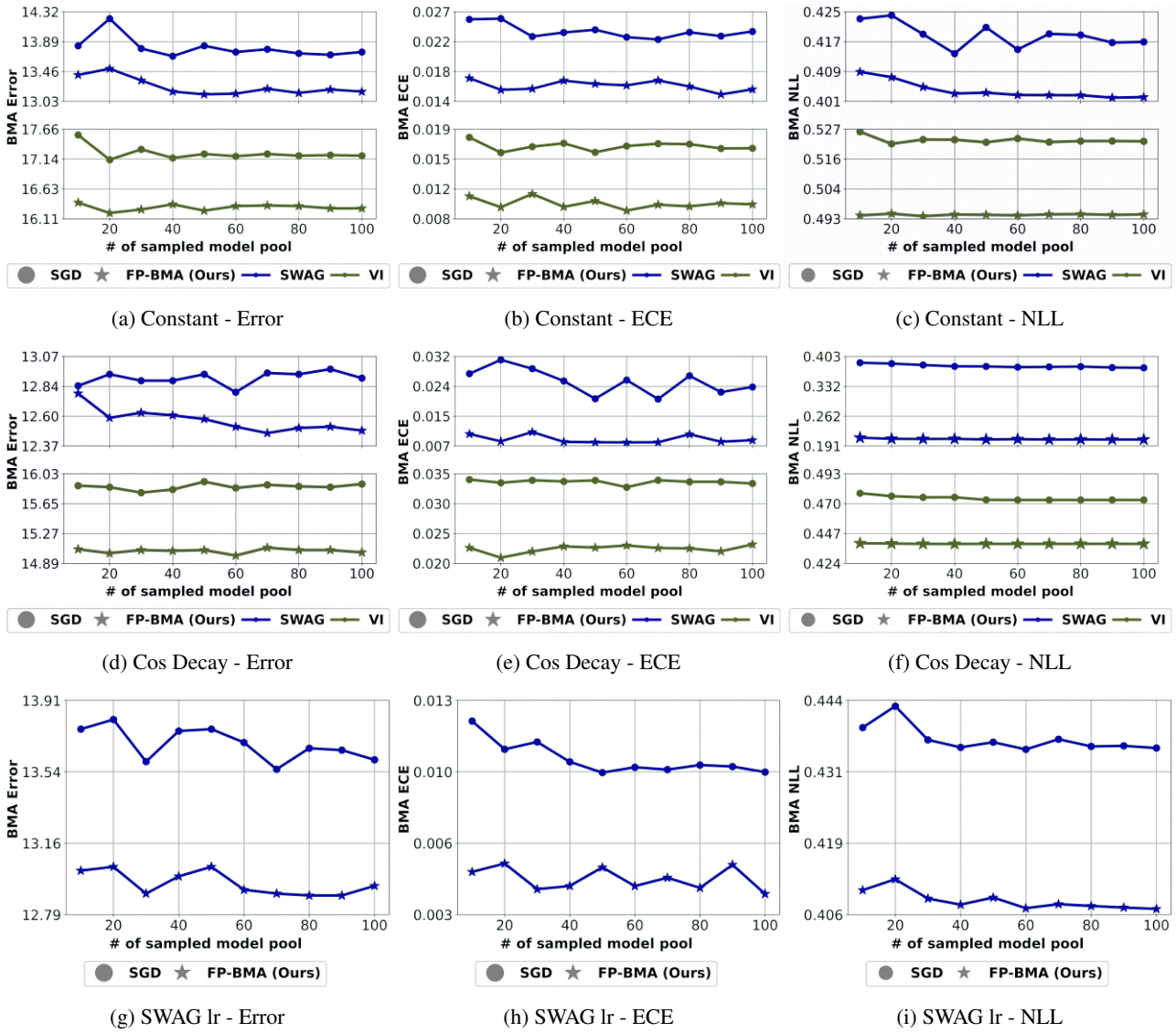


Figure 8: Comparison of Error, NLL, and ECE with various schedulers on CIFAR100 in relation to the maximum eigenvalue λ_1.

A.4 PERFORMANCE CHANGES BASED ON THE NUMBER OF MODELS IN BMA

We also inspect the influence of flatness on BMA performance throughout all considered schedulers. We train ResNet18 on CIFAR10 and CIFAR100, again. Figure 9 and Figure 10 show the results in CIFAR10 and CIFAR100, respectively. Each row means Constant, Cosine Decay, and SWAG lr scheduler, and each column denotes the classification error, ECE, and NLL.

Two main findings were observed consistent with Figure 1c: (1) BNNs trained using the proposed FP-BMA showed superior performance compared to those trained with SGD, suggesting that flatness influences posterior quality and contributes to enhanced BMA performance. (2) FP-BMA training allowed predictive distributions to converge with fewer BMA samples, meaning effective approximation can be achieved with a smaller number of samples.

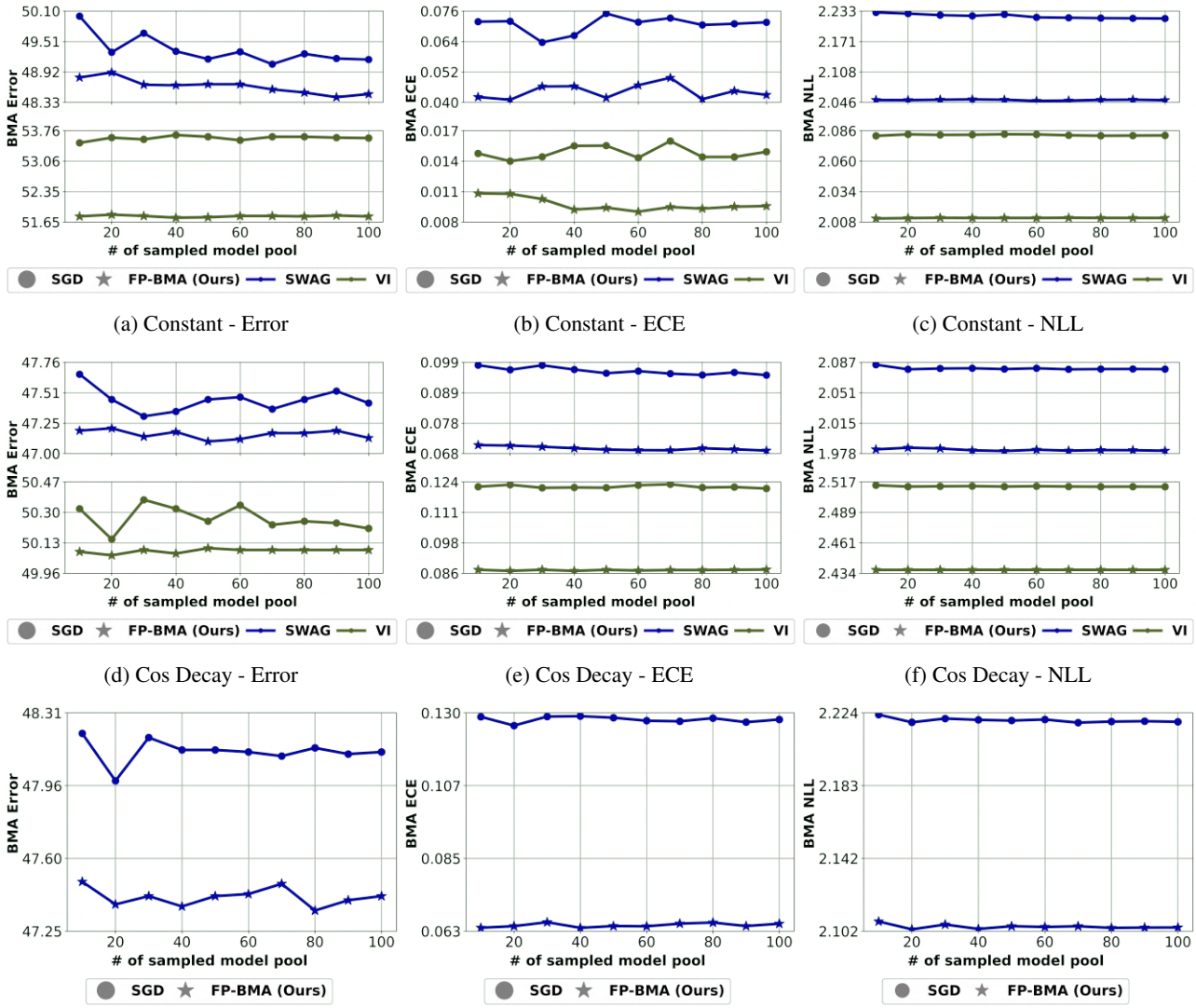


(a) Constant - Error (b) Constant - ECE (c) Constant - NLL

(d) Cos Decay - Error (e) Cos Decay - ECE (f) Cos Decay - NLL

(g) SWAG lr - Error (h) SWAG lr - ECE (i) SWAG lr - NLL

Figure 9: Performance variation based on sampling considering flatness among BMA on CIFAR10. Each row means the Constant, Cos Decay, and SWAG lr scheduler. Each column denotes classification error, ECE, and NLL. It reveals that the flatness should be taken into account for efficient BMA.



(a) Constant - Error (b) Constant - ECE (c) Constant - NLL

(d) Cos Decay - Error (e) Cos Decay - ECE (f) Cos Decay - NLL

(g) SWAG lr - Error (h) SWAG lr - ECE (i) SWAG lr - NLL

Figure 10: Performance variation based on sampling considering flatness among BMA on CIFAR100. Each row means the Constant, Cos Decay, and SWAG lr scheduler. Each column denotes classification error, ECE, and NLL. It reveals that the flatness should be taken into account for efficient BMA.

B PROOF AND DERIVATION

B.1 PROOF OF THEOREM 1

Flatness Bound of BMA We first derive flatness bound of BMA. 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 is known to bound the eigenvalues of Hermitian matrices.

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

$$\lambda_{k+l-1}(C_i + C_j) \leq \lambda_k(C_i) + \lambda_l(C_j) \leq \lambda_{k+l-N}(C_i + C_j). \tag{12}$$

Let $k = 1$ and $l = 1$, then Eq. 12 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:

$$\lambda_1(\frac{1}{M}\sum_{m=1}^{M} H_{f_m}) \leq \frac{1}{M}\sum_{m=1}^{M} \lambda_1(H_{f_m}). \tag{13}$$

One the other hand, we can let $(k, l) = \{(1, p), (p, 1)\}$ and rewrite the Eq. 12 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:

$$\max\left(\left\{\frac{1}{M}\left(\lambda_1(H_{f_m}) + \sum_{\substack{n=1\\ n\neq m}}^{M} \lambda_p(H_{f_n})\right)\right\}_{m=1}^{M}\right) \leq \lambda_1(\frac{1}{M}\sum_{m=1}^{M} H_{f_m}). \tag{14}$$

By combining Eq. 13 with Eq. 14 and substituting λ_1 to λ_{max} and λ_p to λ_{min}, the flatness of averaged weight parameter is bounded as:

$$\max\left(\left\{\frac{1}{M}\left(\lambda_{\text{max}}(H_{f_m}) + \sum_{\substack{n=1\\ n\neq m}}^{M} \lambda_{\text{min}}(H_{f_n})\right)\right\}_{m=1}^{M}\right) \leq \lambda_{\text{max}}(\frac{1}{M}\sum_{m=1}^{M} H_{f_m}) \leq \frac{\sum_{m=1}^{M} \lambda_{\text{max}}(H_{f_m})}{M}. \tag{15}$$

However, Eq 15, as a bound for weight averaging (WA), cannot be directly applied to BMA, which marginalizes diverse predictions. To bridge this gap, we leverage Proposition 2. which characterized the close relation between WA and BMA [Izmailov et al., 2018, Wortsman et al., 2022, Rame et al., 2022].

Proposition 2. *(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 2 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:

$$H_{f_{\text{WA}}} \approx H_{f_{\text{BMA}}} \tag{16}$$

By putting Eq. 16 into Eq. 15, it leads to Lemma 1.

Lemma 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:*

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

Generalization Error Bound for BMA To further elucidate the theoretical connection between posterior flatness and generalization, we present the following proposition which is adapted from the generalization analysis of Eigen-SAM [Luo et al., 2024]:

Proposition 3 (Adapted from Eigen-SAM)**.** *Let $\ell : \mathbb{R}^p \to \mathbb{R}$ be a loss function upper bounded by L, and assume its third-order derivatives are uniformly bounded by a constant C. Suppose the following inequality holds for any parameter θ:*

$$\ell_{\mathcal{D}}(\theta) \leq \mathbb{E}_{\epsilon}[\ell_{\mathcal{D}}(\theta+\epsilon)] \qquad \text{where } \epsilon \sim \mathcal{N}(0, \sigma^2 I_p).$$

Then, for any $\delta \in (0,1)$ and $\sigma > 0$, with probability at least $1-\delta$ over the training set $\mathcal{S} \sim \mathcal{D}^n$, we have:

$$\ell_{\mathcal{D}}(\theta) \leq \ell_{\mathcal{S}}(\theta) + \frac{p\sigma^2}{2}\lambda_{\max}(\nabla^2\ell_{\mathcal{S}}(\theta)) + \frac{Cp^3\sigma^3}{6} + \frac{L}{2\sqrt{n}}\sqrt{p\log\left(1+\frac{\|\theta\|^2}{p\sigma^2}\right) + O(1) + 2\log\frac{1}{\delta} + 4\log(n+p)}$$

Building upon Lemma 2 and Proposition 3, we prove a new generalization bound that formally connects posterior flatness to the generalization performance of BMA:

Theorem 1 (Generalization Bound for BMA)**.** *Let w_m be samples from a variational posterior $q(w)$, and let f_{BMA} denote the BMA predictor obtained by averaging predictions over these samples. Assume the conditions in Proposition 2 and Proposition 3 hold for each sampled model f_m with parameter w_m. Then, for any $\delta \in (0,1)$ and $\sigma > 0$, with probability at least $1-\delta$ over the training set $\mathcal{S} \sim \mathcal{D}^n$, the generalization error of the BMA predictor is upper bounded as:*

$$\ell_{\mathcal{D}}(f_{BMA}) \leq \ell_{\mathcal{S}}(f_{BMA}) + \frac{p\sigma^2}{2}\lambda_{\max}(H_{f_{BMA}}) + \frac{Cp^3\sigma^3}{6} + \frac{L}{2\sqrt{n}}\sqrt{p\log\left(1+\frac{\|f_{BMA}\|^2}{p\sigma^2}\right) + O(1) + 2\log\frac{1}{\delta} + 4\log(n+p)}$$

where $H_{f_{BMA}}$ denotes the Hessian of the loss evaluated at f_{BMA}.

B.2 DERIVATION OF BAYESIAN FLAT-SEEKING OPTIMIZER

B.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 $w = \mu + \frac{1}{\sqrt{2}}(\sigma z_1 + Lz_2)$, 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 $\theta = \text{Concat}(\mu; \sigma; L)$.

B.2.2 Objective function

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

$$\ell^{\gamma}_{\text{FP-BMA}}(\theta) = \max_{d|\theta+\Delta\theta,\theta|\leq\gamma^2} \ell(\theta+\Delta\theta) + \beta \mathrm{D}_{\mathrm{KL}}(p_\theta(w|\mathcal{D})||p(w)) \tag{17}$$

$$\text{s.t. } d|\theta+\Delta\theta,\theta| = \mathrm{D}_{\mathrm{KL}}\left[p_{\theta+\Delta\theta}(w|\mathcal{D})||p_\theta(w|\mathcal{D})\right]. \tag{18}$$

B.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. 18 as:

$$\mathrm{D_{KL}}[p(w|\mathcal{D}, \theta + \Delta\theta)||p(w|\mathcal{D}, \theta)] = \int_w p(w|\mathcal{D}, \theta + \Delta\theta) \ \log \frac{p(w|\mathcal{D}, \theta + \Delta\theta)}{p(w|\mathcal{D}, \theta)} dw. \tag{19}$$

In Eq. 19, we apply first-order Taylor Expansion:

$$\begin{aligned} p(w|\mathcal{D}, \theta + \Delta\theta) &\approx p(w|\mathcal{D}, \theta) + \nabla_\theta p(w|\mathcal{D}, \theta)^T \Delta\theta. \\ \log p(w|\mathcal{D}, \theta + \Delta\theta) &\approx \log p(w|\mathcal{D}, \theta) + \nabla_\theta \log p(w|\mathcal{D}, \theta)^T \Delta\theta. \end{aligned} \tag{20}$$

Substitute right terms of Eq. 19 with Eq. 20:

$$\begin{aligned} &\int_w p(w|\mathcal{D}, \theta + \Delta\theta) \ \log \frac{p(w|\mathcal{D}, \theta + \Delta\theta)}{p(w|\mathcal{D}, \theta)} dw \\ &= \int_w \big(p(w|\mathcal{D}, \theta) + \Delta\theta^T \nabla_\theta p(w|\mathcal{D}, \theta)\big) \nabla_\theta \log p(w|\mathcal{D}, \theta)^T \Delta\theta \ dw \\ &= \int_w p(w|\mathcal{D}, \theta) \nabla_\theta \log p(w|\mathcal{D}, \theta)^T \Delta\theta dw \\ &\quad + \int_w \Delta\theta^T p(w|\mathcal{D}, \theta) \nabla_\theta \log p(w|\mathcal{D}, \theta) \nabla_\theta \log p(w|\mathcal{D}, \theta)^T \Delta\theta \ dw. \end{aligned} \tag{21}$$

First term of Eq. 21 is equal to 0:

$$\begin{aligned} &\int_w p(w|\mathcal{D}, \theta) \nabla_\theta \log p(w|\mathcal{D}, \theta) \ dw \\ &= \int_w p(w|\mathcal{D}, \theta) \frac{\nabla_\theta p(w|\mathcal{D}, \theta)}{p(w|\mathcal{D}, \theta)} \ dw \\ &= \int_w \nabla_\theta p(w|\mathcal{D}, \theta) \ dw \ = \nabla_\theta \int_w p(w|\mathcal{D}, \theta) = 0. \end{aligned} \tag{22}$$

Using Eq. 21 and Eq. 22, Eq. 19 can be rewritten as Fisher information matrix by the definition of expectation:

$$\begin{aligned} &D_{KL}[p(w|\mathcal{D}, \theta + \Delta\theta)||p(w|\mathcal{D}, \theta)] \\ &= \int_w \Delta\theta^T p(w|\mathcal{D}, \theta) \nabla_\theta \log p(w|\mathcal{D}, \theta) \nabla_\theta \log p(w|\mathcal{D}, \theta)^T \Delta\theta \\ &= \Delta\theta^T \mathbb{E}_w[\nabla_\theta \log p(w|\mathcal{D}, \theta) \nabla_\theta \log p(w|\mathcal{D}, \theta)^T] \Delta\theta \\ &= \Delta\theta^T F_\theta(\theta) \Delta\theta, \end{aligned} \tag{23}$$

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] :

$$F_\theta(\theta)^{-1} = \frac{\nabla_\theta \log p(w|\mathcal{D}, \theta) \nabla_\theta \log p(w|\mathcal{D}, \theta)^T}{\|\nabla_\theta \log p(w|\mathcal{D}, \theta)\|^4}. \tag{24}$$

Lagrangian Dual Problem From the result of Eq. 23, we can rewrite the Eq. 17:

$$\ell^{\gamma}_{\text{FP-BMA}}(\theta) = \max_{\Delta\theta^T F_\theta(\theta)\Delta\theta \leq \gamma^2} \ell(\theta + \Delta\theta). \tag{25}$$

We can reach the optimal perturbation of FP-BMA $\Delta\theta^*$ by using Taylor Expansion on $\ell(\theta + \Delta\theta)$ of Eq. 17:

$$\ell(\theta + \Delta\theta) = \ell(\theta) + \nabla_\theta \ell(\theta)^T \Delta\theta. \tag{26}$$

Using Eq. 26, we can rewrite Eq. 17 as Lagrangian dual problem:

$$L(\Delta\theta, \lambda) = \ell(\theta) + \nabla \ell_\theta(\theta)^T \Delta\theta - \lambda(\Delta\theta^T F_\theta(\theta)\Delta\theta - \gamma^2). \tag{27}$$

Differentiating Eq. 27, we get $\Delta\theta^*$:

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

Putting $\Delta\theta^*$ of Eq. 28 into $\Delta\theta$ of Eq. 27, we can rewrite Eq. 27:

$$\begin{aligned} L(\Delta\theta^*, \lambda) = \ell(\theta) &+ \frac{1}{2\lambda} \nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1} \nabla_\theta \ell(\theta) \\ &- \frac{1}{4\lambda} \nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1} \nabla_\theta \ell(\theta) + \lambda\gamma^2. \end{aligned} \tag{29}$$

By taking derivative of Eq. 29 w.r.t. λ, we can also get λ^*:

$$\begin{aligned} &\frac{\alpha L(\Delta\theta^*, \lambda)}{\alpha \lambda} = -\frac{1}{2\lambda^2} \nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1} \nabla_\theta \ell(\theta) + \frac{1}{4\lambda^2} \nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1} \nabla_\theta \ell(\theta) + \gamma^2 = 0 \\ &4\lambda^2\gamma^2 = \nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1} \nabla_\theta \ell(\theta) \\ &\therefore\ \lambda^* = \frac{\sqrt{\nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1} \nabla_\theta \ell(\theta)}}{2\gamma}. \end{aligned} \tag{30}$$

Finally, we get our $\Delta\theta_{\text{FP-BMA}}$ by substituting Eq. 30 into Eq. 28:

$$\Delta\theta_{\text{FP-BMA}} = \gamma \frac{F_\theta(\theta)^{-1} \nabla_\theta \ell(\theta)}{\sqrt{\nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1} \nabla_\theta \ell(\theta)}}. \tag{31}$$

B.3 PROOF OF THEOREM 2

B.3.1 FP-BMA to FSAM

Theorem 2 shows that FP-BMA is degenerated to FSAM under DNN and diagonal FIM setting. Deterministic parameters draw out the constant prior $p(w|x) = c$ and mean-only variational parameters $w = \theta$.

First, we can rewrite the log posterior $\log p_\theta(w|x,y)$ with Bayes rule:

$$\log p_\theta(w|x,y) = \log p_\theta(y|x,w) + \log p_\theta(w|x) - Z, \tag{32}$$

where Z is constant independent of w. Is is noted that the log posterior is divided into the log predictive distribution and log prior. Also, note that the prior is conditioned on the data to align with a generalized notation. The prior can depend on the input; however, this dependence is often ignored in practice [Marek et al., 2024].

By taking derivative with respect to θ on Eq. 32, the constant Z goes to 0:

$$\nabla_\theta \log p_\theta(w|x,y) = \nabla_\theta p_\theta(y|x,w) + \nabla_\theta \log p_\theta(w|x).$$

We have constant prior $p(w|x) = c$ in deterministic setting and it makes the gradient of log posterior and log predictive distribution:

$$\nabla_\theta \log p_\theta(w|x,y) = \nabla_\theta p_\theta(y|x,w). \tag{33}$$

Underlying Eq. 33, it is possible to substitute the gradient of log posterior into the gradient of log predictive distribution and FIM over posterior goes to FIM over predictive distribution:

$$\begin{aligned} F_\theta(\theta) =& \mathbb{E}_{w,\mathcal{D}}[\nabla_\theta \log p_\theta(w|x,y) \nabla_\theta \log p_\theta(w|x,y)^T] \\ =& \mathbb{E}_{w,\mathcal{D}}[\nabla_\theta \log p_\theta(y|x,w) \nabla_\theta \log p_\theta(y|x,w)^T]. \end{aligned} \tag{34}$$

By taking diagonal computation over Eq. 34, it goes to $F_y(\theta)$. After that, using the fact that mean-only variational parameters, FP-BMA degnerates to FSAM with $F_y(\theta)$ finally.

$$\Delta\theta_{\text{FP-BMA}} = \gamma \frac{F_y(\theta)^{-1}\nabla_\theta \ell(\theta)}{\sqrt{F_y(\theta)^{-1}\nabla_\theta \ell(\theta) F_y(\theta)^{-1}}}. \tag{35}$$

B.3.2 FP-BMA to SAM

It is simple to show that FP-BMA is extended version of SAM by defining FIM over output distribution $F_y(w)$ as identity matrix I in Eq. 35, FP-BMA goes to SAM.

$$\Delta\theta_{\text{FP-BMA}} = \gamma \frac{\nabla_w \ell(w)}{\|\nabla_w \ell(w)\|_2}. \tag{36}$$

B.3.3 FP-BMA to NG

Theorem 2 also states the NG can be approximated with FP-BMA under specific conditions. The update rule of natural gradient and FP-BMA can be written as Eq. 37 and Eq. 38, respectively.

$$\theta \leftarrow \theta + \eta_{\text{NG}} F_y(\theta)^{-1}\nabla_\theta \ell(\theta). \tag{37}$$

$$\theta \leftarrow \theta + \eta_{\text{FP-BMA}} \nabla_\theta \ell(\theta + \Delta\theta). \tag{38}$$

where η_{NG} and $\eta_{\text{FP-BMA}}$ denote the learning rate of NG and FP-BMA. Note that we assume the log likelihood as loss fuction.

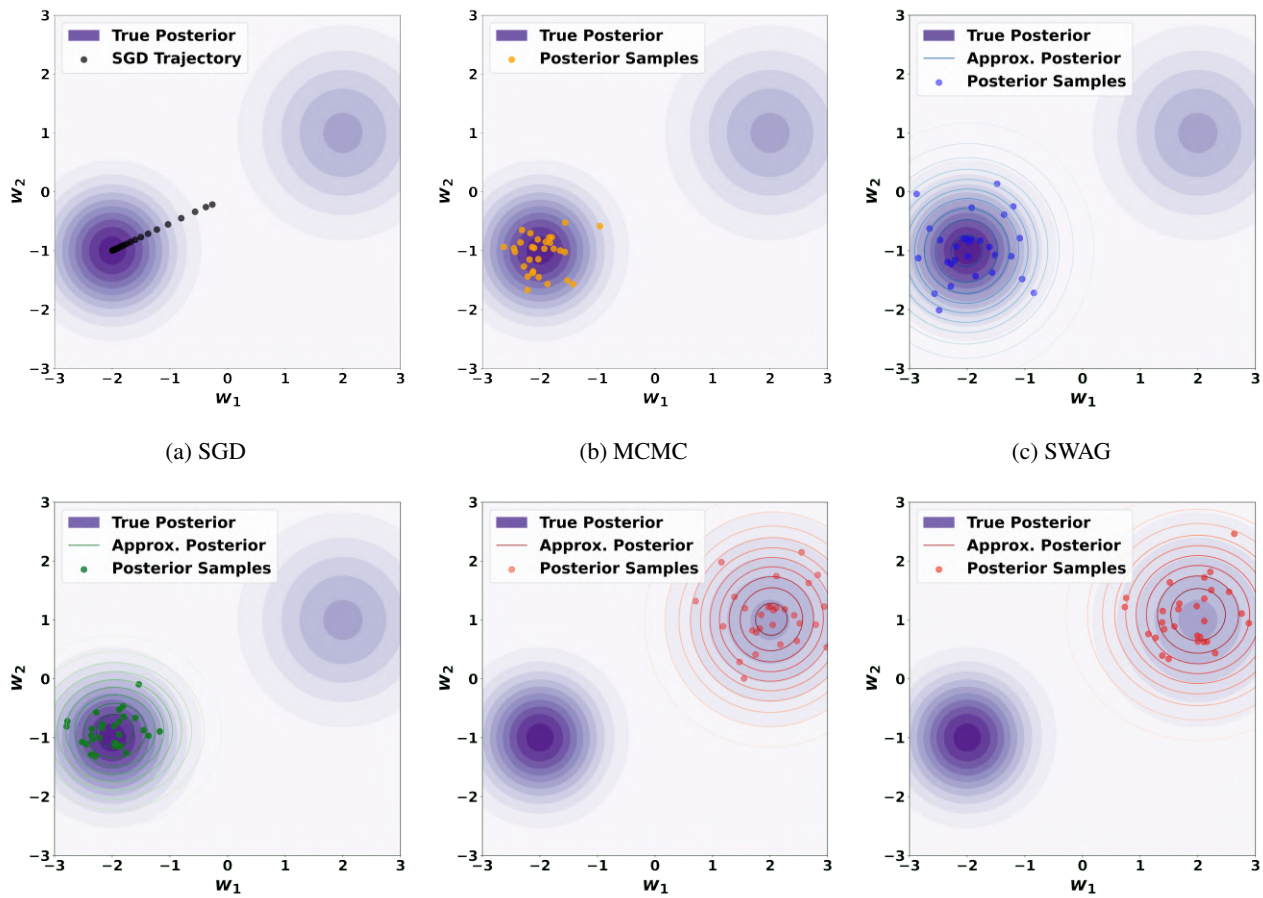
The $\nabla_\theta \ell(\theta + \Delta\theta)$ in Eq. 38 can be approximated with Taylor Expansion, the connection between Hessian and FIM, and Eq. 34 in DNN setup:

$$\begin{aligned} \nabla_\theta \ell(\theta + \Delta\theta) &\approx \nabla_\theta \ell(\theta) + \nabla_\theta^2 \Delta\theta \\ &= \nabla_\theta \ell(\theta) + \nabla_\theta^2 \ell(\theta) \cdot \gamma \frac{F_\theta(\theta)^{-1}\nabla_\theta \ell(\theta)}{\sqrt{\nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1}\nabla_\theta \ell(\theta)}} \\ &= \nabla_\theta \ell(\theta) + \gamma' \nabla_\theta^2 \ell(\theta) F_\theta(\theta)^{-1}\nabla_\theta \ell(\theta) \left(\because \text{Let } \gamma' = \frac{\gamma}{\sqrt{\nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1}\nabla_\theta \ell(\theta)}} \right) \\ &= [I + \gamma' \nabla_\theta^2 \ell(\theta) F_\theta(\theta)^{-1}]\nabla_\theta \ell(\theta) \\ &\approx (1+\gamma')\nabla_\theta \ell(\theta) \; (\because \nabla_\theta^2 \ell(\theta) \approx F_y(\theta), F_\theta(\theta) = F_y(\theta)). \end{aligned} \tag{39}$$

By using the denoted learning rate $\eta_{\text{FP-BMA}} = \frac{\eta_{\text{NG}}}{I+\gamma'} F_\theta(\theta)^{-1}$, Eq. 34, and Eq. 39, update rule of FP-BMA approximates to NG.

C EXPERIMENTS

C.1 SYNTHETIC EXAMPLE



(a) SGD (b) MCMC (c) SWAG

(d) VI (e) FP-BMA (SWAG) (f) FP-BMA (VI)

Figure 11: Posterior approximation with synthetic example. When both flat and sharp modes coexist, we compared how optimizers approximate the posterior. Unlike other methods, the proposed FP-BMA converged to the flat mode, demonstrating its effectiveness in finding more stable solutions.

Following Li and Zhang [2023], we construct a loss surface following the distribution $\frac{1}{2}(\mathcal{N}([-2,-1]^T, 0.5I)) + \frac{1}{2}(\mathcal{N}([2,1]^T, I))$ and set the initial point at $(-0.4, -0.4)$. Unlike other SGD-based methods, FP-BMA efficiently identifies flat modes regardless of the underlying BNN frameworks.

C.2 LEARNING FROM SCRATCH

C.2.1 FP-BMA with diverse BNN frameworks

In Eq. 6, 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. 6.

FP-BMA (MCMC) We mainly adopt SGLD for MCMC in this work. For SGLD, we incorporated noise into Eq. 6 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. 9). 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. 6, β is fixed to 0, and as the trajectory of SWA is optimized through FP-BMA, posterior approximation can be performed accordingly.

C.2.2 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. 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]

MCMC We consistently use SGLD [Welling and 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].

Table 6: Hyperparameter Configuration for CIFAR10

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

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].

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

C.3 BAYESIAN TRANSFER LEARNING

C.3.1 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. 11. 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. 11, 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. 11.

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. 11 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. 9). 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. 11. We also let the converted BNN as initialization and train with downstream dataset. We optimize model with the loss function in Eq. 11.

C.3.2 Hyperparameters for Experiments

In this section, we provide the details of the experimental setup for Section 5.3. 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 [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.

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
	FP-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
	FP-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
	FP-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
	FP-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
	FP-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
	FP-BMA (SWAG)	5e-3	9e-1	×	5e-1	5e-4

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 and 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

Table 9: Hyperparameter Configuration for CIFAR100

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

hyperparameter tuning range included: learning rate in [1e-4, 1e-3, 1e-2, 5e-2, 6e-2, 1e-1, 5e-1], 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].

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.

C.4 ALGORITHM OF FP-BMA

Algorithm 1 FP-BMA with Bayesian Transfer Learning

Require: Variational parameter θ, Neighborhood size γ, 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
3-2) Forward and calculate the loss $\ell(\theta)$ with the sampled w
3-3) Backward pass and compute $\nabla_\theta \log q_\theta(w|\mathcal{D})$
3-4) Compute $F_\theta^{-1}(\theta) = \frac{\nabla_\theta \log q_\theta(w|\mathcal{D}) \nabla_\theta \log q_\theta(w|\mathcal{D})^T}{\|\nabla_\theta \log q_\theta(w|\mathcal{D})\|^4}$
3-5) Compute the perturbation $\Delta\theta_{\text{FP-BMA}} = \gamma \frac{F_\theta(\theta)^{-1}\nabla_\theta \ell(\theta)}{\sqrt{\nabla_\theta \ell(\theta)^T F_\theta(\theta)^{-1} \nabla_\theta \ell(\theta)}}$
3-6) Compute gradient approximation for the FP-BMA $\nabla_\theta \ell^\gamma_{\text{FP-BMA}}(\theta) = \frac{\partial \ell(\theta)}{\partial \theta}|_{\theta + \Delta\theta_{\text{FP-BMA}}}$
3-7) Update $\theta \rightarrow \theta - \eta \nabla_\theta \ell_{\text{FP-BMA}}(\theta)$
end for

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 can be adopted to make DNN into BNN. 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.

C.5 EFFICIENCY OF FP-BMA

The following Table 10 summarizes the per-epoch wall-clock time, theoretical time complexity, and memory usage across methods under a unified experimental setting. Evaluation conducted on ResNet-18 with CIFAR-10 10-shot classification. AMP (automatic mixed precision) was enabled for fair efficiency comparison.

Notation:

- p: total number of model parameters
- p_1: number of trainable parameters used in FP-BMA subnetwork ($p_1 \ll p$)
- M: number of MCMC samples

Method	Time Comp.	Wall Clock	Mem. Comp.
SGD	$O(p)$	2.78s	$O(p)$
SAM	$O(2p)$	4.58s	$O(p)$
FSAM	$O(2p)$	4.65s	$O(2p)$
bSAM	$O(2p)$	4.62s	$O(3p)$
MF VI	$O(2p)$	4.09s	$O(2p)$
FF VI	$O(p^2)$	–	$O(p^2)$
MCMC	$O(p)$	2.95s	$O(Mp)$
E-MCMC	$O(2p)$	5.13s	$O(Mp)$
SWAG	$O(p)$	7.89s	$O(Kp)$
F-SWAG	$O(2p)$	11.48s	$O(Kp)$
FP-BMA	$O(2p)$	6.21s	$O(Kp_1)$

Table 10: Time and memory complexity for all methods.

- K: rank for low-rank approximations (e.g., in SWAG or FP-BMA)

To ensure practical efficiency, FP-BMA is implemented with a subnetwork strategy and inverse vector product approximation (as shown in Algorithm 1). These design choices allow us to limit both runtime and memory overhead, which we found to be comparable to standard baselines.

C.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	$\mathbf{0.297}_{\pm 0.038}$	$\mathbf{0.147}_{\pm 0.037}$	$\mathbf{0.339}_{\pm 0.023}$	$\mathbf{1.113}_{\pm 0.009}$	$\mathbf{0.474}_{\pm 0.023}$	$\mathbf{0.455}_{\pm 0.006}$	$\mathbf{0.219}_{\pm 0.037}$	$\mathbf{0.272}_{\pm 0.006}$	$\mathbf{1.071}_{\pm 0.036}$	$\mathbf{0.504}_{\pm 0.012}$

C.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 12, our method ensures relatively robust performance in the data distribution shift, even as the severity increases.

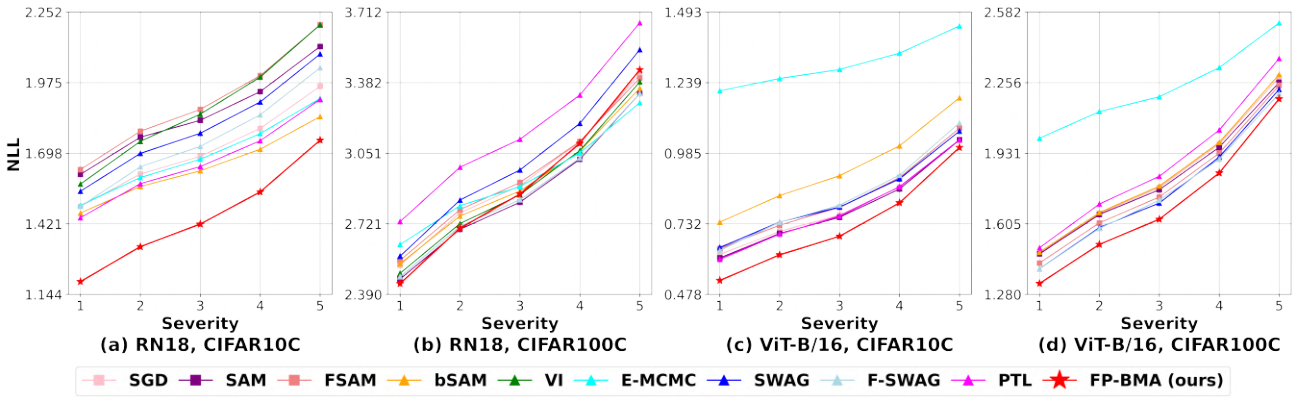


Figure 12: NLL performance of ResNet 18 and ViT-B/16 on corrupted CIFAR10 and CIFAR100, respectively [Hendrycks and Dietterich, 2019].

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	$49.57_{\pm 0.97}$	$1.49_{\pm 0.02}$	$45.78_{\pm 1.43}$	$1.62_{\pm 0.04}$	$43.78_{\pm 1.44}$	$1.69_{\pm 0.04}$	$40.83_{\pm 1.59}$	$1.80_{\pm 0.06}$	$36.30_{\pm 1.79}$	$1.96_{\pm 0.08}$
SAM	$50.23_{\pm 2.11}$	$1.62_{\pm 0.07}$	$46.56_{\pm 2.00}$	$1.76_{\pm 0.03}$	$44.59_{\pm 2.26}$	$1.83_{\pm 0.03}$	$41.85_{\pm 2.42}$	$1.94_{\pm 0.04}$	$37.33_{\pm 2.52}$	$2.12_{\pm 0.07}$
FSAM	$48.76_{\pm 4.00}$	$1.63_{\pm 0.03}$	$45.11_{\pm 3.91}$	$1.78_{\pm 0.01}$	$42.94_{\pm 3.88}$	$1.87_{\pm 0.03}$	$40.06_{\pm 3.85}$	$2.00_{\pm 0.08}$	$35.70_{\pm 3.50}$	$2.20_{\pm 0.12}$
SWAG	$50.05_{\pm 0.76}$	$1.55_{\pm 0.09}$	$46.31_{\pm 1.16}$	$1.70_{\pm 0.11}$	$44.17_{\pm 1.07}$	$1.78_{\pm 0.11}$	$41.20_{\pm 1.13}$	$1.90_{\pm 0.13}$	$36.64_{\pm 1.26}$	$2.09_{\pm 0.15}$
F-SWAG	$51.37_{\pm 1.08}$	$1.49_{\pm 0.05}$	$47.35_{\pm 0.71}$	$1.64_{\pm 0.04}$	$45.16_{\pm 0.66}$	$1.72_{\pm 0.06}$	$42.01_{\pm 0.57}$	$1.85_{\pm 0.06}$	$37.27_{\pm 0.64}$	$2.03_{\pm 0.07}$
bSAM	$49.20_{\pm 2.40}$	$1.46_{\pm 0.05}$	$45.35_{\pm 1.93}$	$1.57_{\pm 0.04}$	$43.07_{\pm 2.10}$	$1.63_{\pm 0.04}$	$40.12_{\pm 1.74}$	$1.71_{\pm 0.03}$	$35.50_{\pm 1.36}$	$1.84_{\pm 0.02}$
VI	$50.72_{\pm 0.80}$	$1.58_{\pm 0.11}$	$46.87_{\pm 0.32}$	$1.74_{\pm 0.11}$	$44.52_{\pm 0.39}$	$1.85_{\pm 0.12}$	$41.38_{\pm 0.29}$	$2.00_{\pm 0.12}$	$36.73_{\pm 0.17}$	$2.20_{\pm 0.10}$
E-MCMC	$49.86_{\pm 1.54}$	$1.49_{\pm 0.03}$	$46.17_{\pm 1.55}$	$1.60_{\pm 0.04}$	$44.07_{\pm 1.72}$	$1.67_{\pm 0.07}$	$41.05_{\pm 1.65}$	$1.77_{\pm 0.10}$	$36.53_{\pm 1.74}$	$1.91_{\pm 0.13}$
PTL	$50.44_{\pm 1.65}$	$1.45_{\pm 0.06}$	$46.22_{\pm 1.96}$	$1.58_{\pm 0.09}$	$44.06_{\pm 1.67}$	$1.65_{\pm 0.09}$	$41.02_{\pm 1.66}$	$1.75_{\pm 0.11}$	$36.14_{\pm 1.51}$	$1.91_{\pm 0.13}$
FP-BMA	$\mathbf{58.53}_{\pm 0.75}$	$\mathbf{1.19}_{\pm 0.02}$	$\mathbf{53.72}_{\pm 0.70}$	$\mathbf{1.33}_{\pm 0.00}$	$\mathbf{50.61}_{\pm 0.84}$	$\mathbf{1.42}_{\pm 0.01}$	$\mathbf{46.76}_{\pm 1.15}$	$\mathbf{1.55}_{\pm 0.03}$	$\mathbf{40.70}_{\pm 1.34}$	$\mathbf{1.75}_{\pm 0.05}$

(b) RN18 CIFAR100C

Method	Severity									
	1		2		3		4		5	
	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓
SGD	$36.01_{\pm 0.86}$	$2.55_{\pm 0.06}$	$31.81_{\pm 0.73}$	$2.79_{\pm 0.06}$	$29.75_{\pm 0.57}$	$2.91_{\pm 0.04}$	$26.73_{\pm 0.25}$	$3.11_{\pm 0.02}$	$22.20_{\pm 0.08}$	$3.40_{\pm 0.00}$
SAM	$37.94_{\pm 0.52}$	$2.46_{\pm 0.02}$	$33.57_{\pm 0.50}$	$\mathbf{2.69}_{\pm 0.03}$	$31.46_{\pm 0.67}$	$\mathbf{2.82}_{\pm 0.03}$	$28.19_{\pm 0.75}$	$3.02_{\pm 0.05}$	$23.32_{\pm 0.69}$	$3.33_{\pm 0.06}$
FSAM	$36.46_{\pm 0.44}$	$2.53_{\pm 0.05}$	$32.24_{\pm 0.36}$	$2.77_{\pm 0.04}$	$30.19_{\pm 0.42}$	$2.90_{\pm 0.03}$	$27.12_{\pm 0.37}$	$3.10_{\pm 0.02}$	$22.48_{\pm 0.39}$	$3.42_{\pm 0.01}$
bSAM	$36.20_{\pm 0.59}$	$2.73_{\pm 0.03}$	$32.48_{\pm 0.34}$	$2.99_{\pm 0.03}$	$30.66_{\pm 0.33}$	$3.12_{\pm 0.02}$	$27.94_{\pm 0.14}$	$3.32_{\pm 0.05}$	$23.66_{\pm 0.29}$	$3.66_{\pm 0.06}$
SWAG	$35.84_{\pm 5.17}$	$2.62_{\pm 0.30}$	$32.43_{\pm 4.55}$	$2.81_{\pm 0.27}$	$30.71_{\pm 4.21}$	$2.89_{\pm 0.25}$	$28.13_{\pm 3.81}$	$3.05_{\pm 0.22}$	$24.24_{\pm 2.99}$	$\mathbf{3.29}_{\pm 0.17}$
F-SWAG	$37.10_{\pm 0.60}$	$2.49_{\pm 0.03}$	$32.84_{\pm 0.62}$	$2.72_{\pm 0.03}$	$30.59_{\pm 0.72}$	$2.86_{\pm 0.04}$	$27.43_{\pm 0.91}$	$3.06_{\pm 0.06}$	$22.74_{\pm 0.93}$	$3.38_{\pm 0.08}$
VI	$38.20_{\pm 0.57}$	$2.47_{\pm 0.02}$	$33.77_{\pm 0.59}$	$2.71_{\pm 0.03}$	$31.70_{\pm 0.75}$	$2.83_{\pm 0.03}$	$28.56_{\pm 0.77}$	$\mathbf{3.03}_{\pm 0.04}$	$23.72_{\pm 0.78}$	$3.33_{\pm 0.05}$
E-MCMC	$36.49_{\pm 0.89}$	$2.57_{\pm 0.06}$	$32.25_{\pm 0.76}$	$2.83_{\pm 0.06}$	$30.22_{\pm 0.63}$	$2.97_{\pm 0.05}$	$27.17_{\pm 0.38}$	$3.19_{\pm 0.03}$	$22.54_{\pm 0.27}$	$3.54_{\pm 0.01}$
PTL	$36.43_{\pm 0.35}$	$2.53_{\pm 0.03}$	$32.24_{\pm 0.40}$	$2.76_{\pm 0.03}$	$30.20_{\pm 0.42}$	$2.87_{\pm 0.03}$	$27.17_{\pm 0.55}$	$3.06_{\pm 0.04}$	$22.56_{\pm 0.54}$	$3.36_{\pm 0.05}$
FP-BMA	$\mathbf{39.41}_{\pm 0.72}$	$\mathbf{2.44}_{\pm 0.04}$	$\mathbf{35.07}_{\pm 0.64}$	$2.70_{\pm 0.05}$	$\mathbf{32.75}_{\pm 0.71}$	$2.86_{\pm 0.05}$	$\mathbf{29.41}_{\pm 0.67}$	$3.10_{\pm 0.05}$	$\mathbf{24.25}_{\pm 0.70}$	$3.44_{\pm 0.05}$

(c) VIT-B/16 CIFAR10C

Method	Severity									
	1		2		3		4		5	
	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓
SGD	$79.62_{\pm 0.56}$	$0.64_{\pm 0.06}$	$76.47_{\pm 0.67}$	$0.73_{\pm 0.06}$	$74.10_{\pm 0.83}$	$0.79_{\pm 0.05}$	$70.42_{\pm 1.23}$	$0.90_{\pm 0.05}$	$64.41_{\pm 1.85}$	$1.08_{\pm 0.05}$
SAM	$79.78_{\pm 0.49}$	$0.61_{\pm 0.01}$	$76.59_{\pm 0.64}$	$0.70_{\pm 0.02}$	$74.58_{\pm 0.94}$	$0.75_{\pm 0.02}$	$71.12_{\pm 1.06}$	$0.86_{\pm 0.03}$	$65.26_{\pm 1.46}$	$1.03_{\pm 0.04}$
FSAM	$79.87_{\pm 0.83}$	$0.62_{\pm 0.02}$	$76.78_{\pm 0.78}$	$0.70_{\pm 0.02}$	$74.70_{\pm 0.60}$	$0.76_{\pm 0.01}$	$71.29_{\pm 0.49}$	$0.86_{\pm 0.01}$	$65.53_{\pm 0.56}$	$1.03_{\pm 0.03}$
bSAM	$78.80_{\pm 1.18}$	$0.64_{\pm 0.04}$	$75.43_{\pm 1.14}$	$0.74_{\pm 0.04}$	$73.45_{\pm 1.43}$	$0.80_{\pm 0.04}$	$70.07_{\pm 1.50}$	$0.91_{\pm 0.05}$	$64.21_{\pm 1.57}$	$1.09_{\pm 0.05}$
SWAG	$76.58_{\pm 1.69}$	$1.21_{\pm 0.04}$	$73.45_{\pm 1.98}$	$1.25_{\pm 0.04}$	$71.20_{\pm 2.18}$	$1.29_{\pm 0.04}$	$67.54_{\pm 2.46}$	$1.35_{\pm 0.04}$	$61.65_{\pm 2.82}$	$1.44_{\pm 0.04}$
F-SWAG	$81.03_{\pm 2.20}$	$0.60_{\pm 0.05}$	$77.73_{\pm 2.63}$	$0.69_{\pm 0.06}$	$75.45_{\pm 2.96}$	$0.76_{\pm 0.07}$	$71.82_{\pm 3.31}$	$0.87_{\pm 0.08}$	$66.05_{\pm 3.59}$	$1.03_{\pm 0.10}$
E-MCMC	$78.91_{\pm 2.31}$	$0.65_{\pm 0.08}$	$75.78_{\pm 2.36}$	$0.74_{\pm 0.08}$	$73.94_{\pm 2.56}$	$0.79_{\pm 0.09}$	$70.66_{\pm 2.63}$	$0.89_{\pm 0.10}$	$65.07_{\pm 2.77}$	$1.06_{\pm 0.11}$
PTL	$76.26_{\pm 2.46}$	$0.74_{\pm 0.06}$	$72.36_{\pm 2.41}$	$0.83_{\pm 0.06}$	$69.61_{\pm 2.46}$	$0.90_{\pm 0.07}$	$65.47_{\pm 2.52}$	$1.01_{\pm 0.07}$	$59.04_{\pm 2.26}$	$1.18_{\pm 0.06}$
FP-BMA	$\mathbf{82.89}_{\pm 1.09}$	$\mathbf{0.53}_{\pm 0.04}$	$\mathbf{79.68}_{\pm 1.26}$	$\mathbf{0.62}_{\pm 0.04}$	$\mathbf{77.30}_{\pm 1.43}$	$\mathbf{0.69}_{\pm 0.05}$	$\mathbf{73.41}_{\pm 1.62}$	$\mathbf{0.81}_{\pm 0.06}$	$\mathbf{66.94}_{\pm 1.79}$	$\mathbf{1.01}_{\pm 0.07}$

(d) VIT-B/16 CIFAR100C

Method	Severity									
	1		2		3		4		5	
	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓	ACC ↑	NLL ↓
SGD	$62.19_{\pm 0.52}$	$1.42_{\pm 0.02}$	$57.81_{\pm 0.37}$	$1.61_{\pm 0.02}$	$55.04_{\pm 0.14}$	$1.73_{\pm 0.02}$	$50.73_{\pm 0.24}$	$1.93_{\pm 0.01}$	$44.12_{\pm 0.39}$	$2.24_{\pm 0.01}$
SAM	$61.90_{\pm 0.53}$	$1.47_{\pm 0.02}$	$57.49_{\pm 0.43}$	$1.65_{\pm 0.02}$	$54.80_{\pm 0.29}$	$1.76_{\pm 0.01}$	$50.52_{\pm 0.25}$	$1.96_{\pm 0.01}$	$44.04_{\pm 0.24}$	$2.26_{\pm 0.01}$
FSAM	$61.70_{\pm 0.52}$	$1.47_{\pm 0.02}$	$57.16_{\pm 0.44}$	$1.65_{\pm 0.02}$	$54.46_{\pm 0.37}$	$1.77_{\pm 0.02}$	$50.11_{\pm 0.39}$	$1.97_{\pm 0.01}$	$43.53_{\pm 0.42}$	$2.28_{\pm 0.01}$
bSAM	$62.36_{\pm 0.73}$	$1.40_{\pm 0.03}$	$57.97_{\pm 0.70}$	$1.58_{\pm 0.03}$	$55.32_{\pm 0.61}$	$1.70_{\pm 0.03}$	$51.09_{\pm 0.49}$	$1.90_{\pm 0.03}$	$44.77_{\pm 0.42}$	$2.21_{\pm 0.03}$
SWAG	$59.19_{\pm 0.90}$	$2.00_{\pm 0.03}$	$55.45_{\pm 0.88}$	$2.12_{\pm 0.03}$	$53.34_{\pm 0.94}$	$2.19_{\pm 0.03}$	$49.44_{\pm 0.81}$	$2.33_{\pm 0.03}$	$43.71_{\pm 0.93}$	$2.53_{\pm 0.03}$
F-SWAG	$59.55_{\pm 2.94}$	$1.49_{\pm 0.11}$	$55.10_{\pm 2.82}$	$1.70_{\pm 0.10}$	$52.37_{\pm 2.80}$	$1.82_{\pm 0.10}$	$48.18_{\pm 2.63}$	$2.04_{\pm 0.09}$	$41.84_{\pm 2.43}$	$2.37_{\pm 0.09}$
E-MCMC	$62.28_{\pm 0.47}$	$1.40_{\pm 0.02}$	$57.84_{\pm 0.46}$	$1.59_{\pm 0.02}$	$55.14_{\pm 0.29}$	$1.71_{\pm 0.02}$	$50.87_{\pm 0.21}$	$1.91_{\pm 0.02}$	$44.49_{\pm 0.13}$	$2.22_{\pm 0.02}$
PTL	$61.84_{\pm 0.33}$	$1.47_{\pm 0.02}$	$57.36_{\pm 0.22}$	$1.66_{\pm 0.02}$	$54.47_{\pm 0.08}$	$1.78_{\pm 0.01}$	$50.03_{\pm 0.23}$	$1.98_{\pm 0.01}$	$43.34_{\pm 0.36}$	$2.29_{\pm 0.01}$
FP-BMA	$\mathbf{63.91}_{\pm 0.02}$	$\mathbf{1.33}_{\pm 0.00}$	$\mathbf{59.70}_{\pm 0.00}$	$\mathbf{1.51}_{\pm 0.00}$	$\mathbf{57.00}_{\pm 0.01}$	$\mathbf{1.63}_{\pm 0.00}$	$\mathbf{52.51}_{\pm 0.03}$	$\mathbf{1.84}_{\pm 0.00}$	$\mathbf{45.39}_{\pm 0.04}$	$\mathbf{2.18}_{\pm 0.00}$

C.8 COMPARISON WITH DIVERSE BASELINES AND INFERENCE METHODS

To further validate the broad applicability and effectiveness of FP-BMA, we compare it with a variety of inference algorithms and baselines, including MCMC-based, multi-modal, and advanced VI-based methods. All results are reported for CIFAR-10 (10-shot) with ResNet-18.

Method	Acc (%) ↑	ECE ↓	NLL ↓
SGHMC	$55.41_{\pm 0.88}$	$0.112_{\pm 0.009}$	$1.371_{\pm 0.025}$
SGHMC + FP-BMA (Ours)	$\mathbf{56.41}_{\pm 1.75}$	$\mathbf{0.055}_{\pm 0.008}$	$\mathbf{1.276}_{\pm 0.021}$
MoLA	65.77	0.045	1.058
MoLA + FP-BMA (Ours)	**66.77**	0.063	**0.998**
IVON	$56.23_{\pm 1.01}$	$0.023_{\pm 0.004}$	$1.262_{\pm 0.037}$
FP-BMA (VI)	$\mathbf{64.98}_{\pm 1.37}$	$\mathbf{0.016}_{\pm 0.007}$	$\mathbf{0.997}_{\pm 0.046}$

Table 12: Comparison of FP-BMA with various inference baselines. All results are based on CIFAR-10 (10-shot) and ResNet-18.

The table above demonstrates that **FP-BMA consistently improves predictive performance and calibration across a range of inference backbones and posterior structures:**

- When applied on top of **SGHMC** [Chen et al., 2014] (a standard MCMC method), FP-BMA yields clear improvements in accuracy, ECE, and NLL. This shows that our approach is compatible with and beneficial to MCMC-based inference, extending its utility beyond VI-based methods.
- In a multi-modal posterior setting (**MoLA** [Eschenhagen et al., 2021]), FP-BMA remains effective, improving accuracy and NLL. However, the gains are less pronounced than in unimodal cases, suggesting that further extension of FP-BMA for multi-modal posteriors could be fruitful.
- Compared to **IVON** [Shen et al., 2024] (which leverages efficient second-order optimization but does not explicitly encourage flatness), FP-BMA achieves significantly better results on all metrics. This highlights the effectiveness of explicitly promoting posterior flatness in Bayesian model averaging.

Overall, these results support the broad applicability and complementary nature of FP-BMA, demonstrating its value as a general-purpose enhancement for Bayesian inference, regardless of the underlying approximation strategy.

C.9 LOSS SURFACE OF SAMPLED MODEL

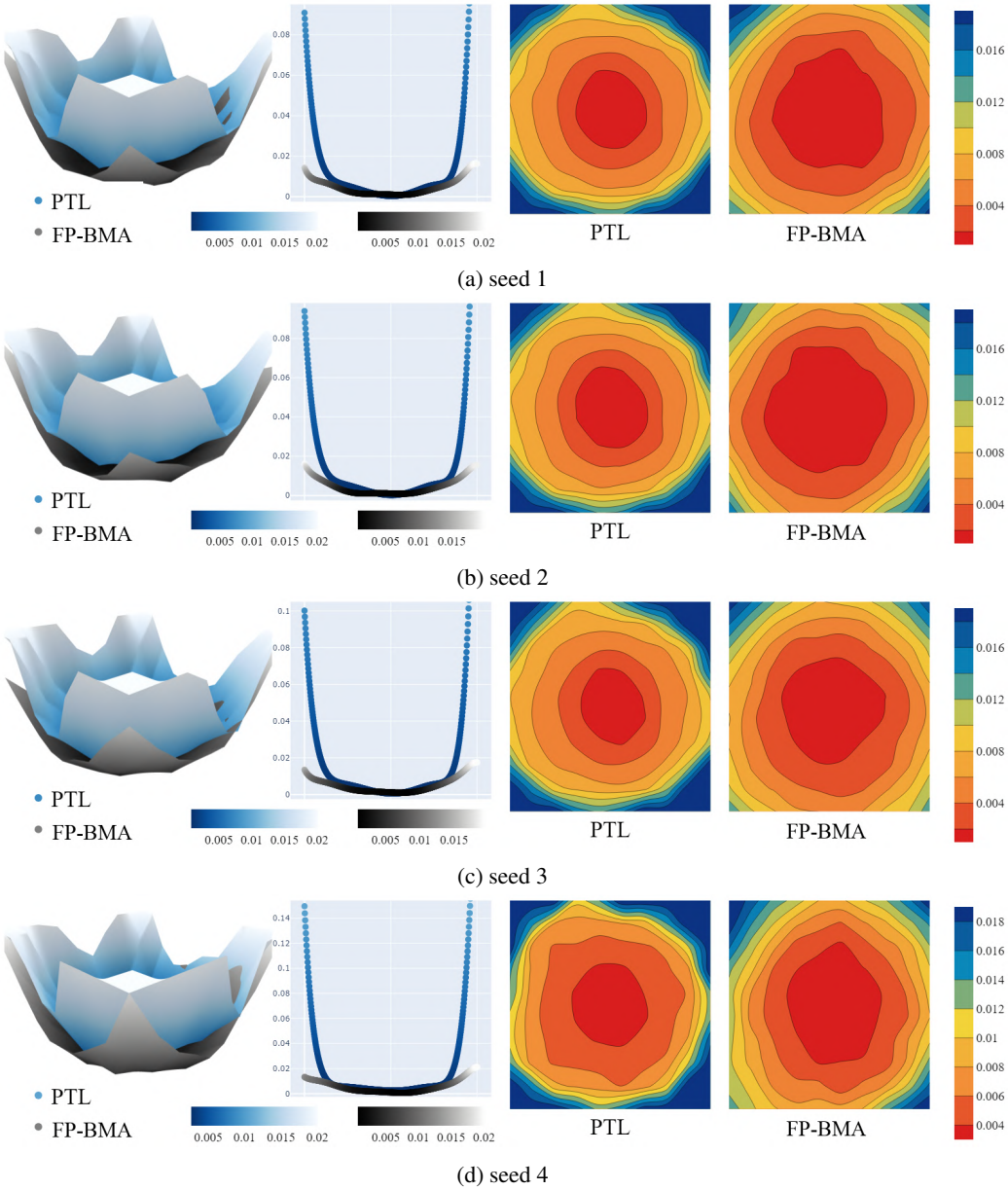


Figure 14: Four instances of sampled weights, including (b) as presented in Figure 4. Across all plots, it is consistently observed that FP-BMA converges to a flatter loss surface compared to PTL.

As shown in Figure 4, 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 14, 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.