BayesAdapter: Being Bayesian, Inexpensively and Reliably, via Bayesian Fine-tuning

[1]**Zhijie Deng,** [2]**Hao Zhang,** [1]**Xiao Yang,** [1]**Yinpeng Dong,** [1]**Jun Zhu***
[1]Dept. of Comp. Sci. & Tech., BNRist Center, Institute for AI, Tsinghua-Bosch Joint ML Center,
THBI Lab, Tsinghua University, Beijing, 100084 China [2]Carnegie Mellon University
dzj17@mails.tsinghua.edu.cn, sjtu.haozhang@gmail.com,
{yangxiao19,dyp17}@mails.tsinghua.edu.cn, dcszj@tsinghua.edu.cn

Abstract

Despite their theoretical appealingness, Bayesian neural networks (BNNs) are left behind in real-world adoption, mainly due to persistent concerns on their scalability, accessibility, and reliability. In this work, we develop the *BayesAdapter* framework to relieve these concerns. In particular, we propose to adapt pre-trained deterministic NNs to be variational BNNs via cost-effective *Bayesian fine-tuning*. Technically, we develop a modularized implementation for the learning of variational BNNs under two representative variational distributions. We refurbish the generally applicable *exemplar reparameterization* trick through exemplar parallelization to efficiently reduce the gradient variance in stochastic variational inference. Based on the developed lightweight paradigm for learning variational BNNs, we conduct a detailed investigation on *do variational BNNs know what they do not know*. We uncover an unreliability issue of variational BNNs' uncertainty estimates, and provide a corresponding prescription. Through extensive experiments on diverse benchmarks, we show that *BayesAdapter* can consistently induce posteriors with higher quality than competitive baselines, especially in large-scale settings, yet significantly reducing training overheads.

1 Introduction

Much effort has been devoted to developing expressive Bayesian neural networks (BNNs) to make accurate and reliable decisions [36, 40, 14, 2]. The principled uncertainty quantification inside BNNs is critical for realistic decision-making, finding applications in scenarios ranging from model-based reinforcement learning [8], active learning [19] to healthcare [31] and autonomous driving [23]. BNNs are also known to be capable of resisting over-fitting and over-confidence.

Nonetheless, BNNs are falling far behind in terms of adoption in real-world applications compared with deterministic NNs [17, 52, 7], due to various issues. For example, typical approximate inference methods for BNNs are often difficult to simultaneously obtain effectiveness and scalability [62, 37]. Implementing a BNN algorithm requires substantially more expertise than implementing a deterministic NN program. Moreover, as revealed, BNNs trained from scratch without the "cold posterior" trick are often systematically worse than their point-estimate counterparts in terms of predictive performance [56]; and some easy-to-use BNNs (e.g., Monte Carlo dropout) tend to suffer from mode collapse in function space, thus usually give uncertainty estimates of poor fidelity [11].

To mitigate these issues, we develop a pre-training & fine-tuning workflow for learning variational BNNs conditioned on the inherent connections between variational BNNs [2] and regular deep neural networks (DNNs). The resultant *BayesAdapter* framework learns a variational BNN by performing several rounds of *Bayesian fine-tuning*, starting from a pre-trained deterministic NN. BayesAdapter is effective and lightweight, and conjoins the complementary benefits from deterministic training and Bayesian reasoning, e.g., good performance, resistance to over-fitting, reliable uncertainty estimates, etc. (find evidence in Figure 1).

*Corresponding author.

Preprint. Under review.

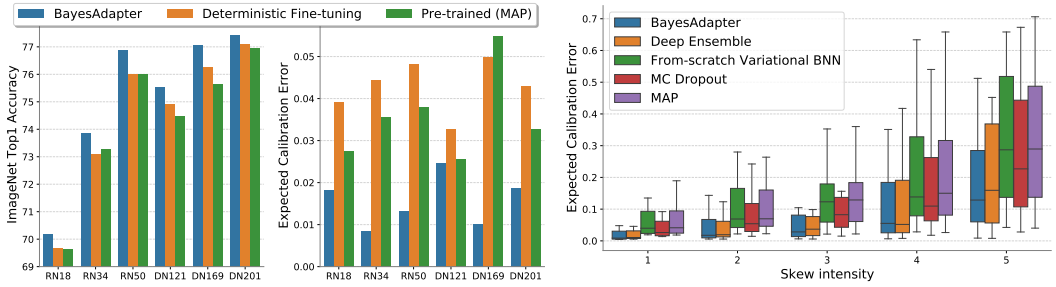


Figure 1: (left): BayesAdapter boosts the accuracy of ImageNet classifiers significantly without compromising model calibration (estimated by expected calibration error (ECE) [15]), while deterministic fine-tuning only marginally improves the accuracy of pre-trained models, yet aggravates over-confidence issue. RN refers to ResNet [17] and DN refers to DenseNet [21]. (right): BayesAdapter learns a CIFAR-10 classifier which approaches or outperforms competing baselines in terms of ECE for CIFAR-10 corruptions [18]. Each box summarizes the ECE across 19 types of skew. We use the *PSE* variational (detailed in Sec 2.2) in these experiments and perform *Bayesian fine-tuning* for only **4** and **12** epochs on ImageNet and CIFAR-10 respectively.

To make *BayesAdapter* comfortably accessible, we provide an easy-to-use implementation for the stochastic variational inference (SVI) under two representative variational distributions: *mean-field Gaussian* and *parameter-sharing ensemble*. Variance reduction for the gradients in SVI plays a critical role in making approximate inference successful, while the pioneering works like local reparameterization [26] or Flipout [55] typically impose restrictive assumptions on the variational form, e.g., a Gaussian or a distribution whose samples can be re-parameterized with symmetric perturbations. To tackle this, we refurbish the widely-criticized *exemplar reparameterization* [26] by accelerating the exemplar-wise computations through parallelization, giving rise to an efficient and general-purpose gradient variance reduction strategy. Based on these designs, we perform an investigation on *do variational BNNs know what they do not know*, and find that typical variational BNNs can seldom yield calibrated uncertainty estimates on realistic, malicious out-of-distribution (OOD) data. We then provide a corresponding prescription, where Bayesian inference is augmented with biased yet meaningful regularization, to ameliorate this pathology.

We conduct extensive experiments to validate the advantages of *BayesAdapter* over competing baselines, in aspects covering learning efficiency, predictive performance, and quality of uncertainty estimates. Desirably, we scale up *BayesAdapter* to big data (e.g., ImageNet [6]), deep architectures (e.g., ResNets [17]), and practical scenarios (e.g., face recognition [7]), and observe promising results.

2 BayesAdapter

In this section, we first motivate *BayesAdapter* by drawing a connection between variational BNNs and DNNs under *maximum a posteriori* (MAP) estimation. We then describe the rationale of *Bayesian fine-tuning*, as well as two implementations of it. Figure 2 illustrates the overall workflow of *BayesAdapter*.

2.1 From DNNs to BNNs

Let $\mathcal{D} = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{n}$ be a given training set, where $\boldsymbol{x}^{(i)} \in \mathbb{R}^d$ and $y^{(i)} \in \mathcal{Y}$ denote the input data and label, respectively. A DNN model can be fit via MAP as follows:

$$\max_{\boldsymbol{w}} \frac{1}{n} \sum_i [\log p(y^{(i)}|\boldsymbol{x}^{(i)}; \boldsymbol{w})] + \frac{1}{n} \log p(\boldsymbol{w}). \tag{1}$$

We use $\boldsymbol{w} \in \mathbb{R}^p$ to denote the high-dimensional model parameters, and $p(y|\boldsymbol{x}; \boldsymbol{w})$

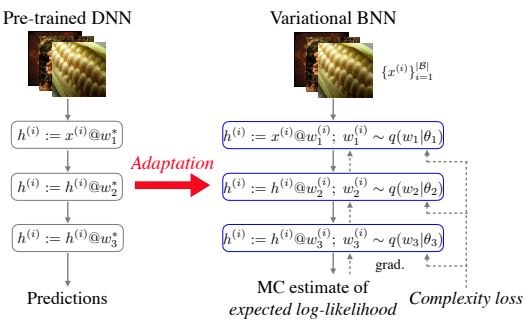


Figure 2: The workflow of *BayesAdapter*: we adapt the pre-trained DNNs to be variational BNNs and then perform several rounds of *Bayesian fine-tuning*. We provide a modularized implementation for *Bayesian fine-tuning* under mean-field Gaussian and parameter-sharing ensemble variational distributions, permitting the users to learn a variational BNN as if one were training a regular DNN under weight decay regularizer.

as the predictive distribution associated with the model. The prior $p(\boldsymbol{w})$, when taking the form of an isotropic Gaussian $\mathcal{N}(\boldsymbol{w}; \mathbf{0}, \sigma_0^2\mathbf{I})$, reduces to the weight decay regularizer with coefficient $\lambda = 1/(\sigma_0^2 n)$ in optimization. Yet, deterministic training may easily cause over-fitting and over-confidence, rendering the learned models of poor fidelity (see Figure 1). Naturally, Bayesian neural networks (BNNs) come into the picture to address these limitations.

Typically, a BNN imposes a prior $p(\boldsymbol{w})$ on model parameters, which is put together with the likelihood $p(\mathcal{D}|\boldsymbol{w})$ to infer the posterior $p(\boldsymbol{w}|\mathcal{D})$. Among the wide spectrum of BNN algorithms [36, 40, 14, 2, 33, 13], variational BNNs are particularly promising due to their analogy to ordinary backprop. Formally, variational BNNs use a $\boldsymbol{\theta}$-parameterized variational distribution $q(\boldsymbol{w}|\boldsymbol{\theta})$ to approximate the true posterior $p(\boldsymbol{w}|\mathcal{D})$, by maximizing the evidence lower bound (ELBO) (scaled by $1/n$):

$$\max_{\boldsymbol{\theta}} \underbrace{\mathbb{E}_{q(\boldsymbol{w}|\boldsymbol{\theta})}\big[\frac{1}{n}\sum_i \log p(y^{(i)}|\boldsymbol{x}^{(i)}; \boldsymbol{w})\big]}_{\mathcal{L}_{ell}} \underbrace{- \frac{1}{n} D_{\mathrm{KL}}\left(q(\boldsymbol{w}|\boldsymbol{\theta}) \| p(\boldsymbol{w})\right)}_{\mathcal{L}_c}, \tag{2}$$

where $\mathcal{L}_{ell}$ is the *expected log-likelihood* and $\mathcal{L}_c$ is the *complexity loss*. By casting posterior inference into optimization, Eq. (2) makes the training of BNNs resemble that of DNNs. After training, the variational posterior is leveraged for prediction by marginalizing over all likely modes:

$$p(y|\boldsymbol{x}, \mathcal{D}) \approx \mathbb{E}_{q(\boldsymbol{w}|\boldsymbol{\theta})} p(y|\boldsymbol{x}; \boldsymbol{w}) \approx \frac{1}{S}\sum_{s=1}^{S} p(y|\boldsymbol{x}; \boldsymbol{w}^{(s)}), \tag{3}$$

where $\boldsymbol{w}^{(s)} \sim q(\boldsymbol{w}|\boldsymbol{\theta}), s = 1, ..., S$, with S denoting the number of Monte Carlo (MC) samples. Eq. (3) is also known as *posterior predictive*, *Bayes ensemble*, or *Bayes model average*.

We can simultaneously quantify the *epistemic* uncertainty with these MC samples. A principled uncertainty metric is the mutual information between the model parameter and the prediction [49], estimated by (H denotes the Shannon entropy):

$$\mathcal{I}(\boldsymbol{w}, y|\boldsymbol{x}, \mathcal{D}) \approx H\left(\tfrac{1}{S}\textstyle\sum_{s=1}^{S} p(y|\boldsymbol{x}; \boldsymbol{w}^{(s)})\right) - \tfrac{1}{S}\textstyle\sum_{s=1}^{S} H\left(p(y|\boldsymbol{x}; \boldsymbol{w}^{(s)})\right). \tag{4}$$

However, most of the existing variational BNNs exhibit limitations in scalability and performance [42, 56] compared with their deterministic counterparts, primarily due to the higher difficulty of learning high-dimensional distributions from scratch than point estimates.

Given that MAP converges to a mode of the Bayesian posterior, it might be plausible to *adapt pre-trained deterministic DNNs to be Bayesian economically*. Following this hypothesis, we repurpose the converged parameters $\boldsymbol{w}^*$ of MAP – take $\boldsymbol{w}^*$ as the initialization of the parameters of the approximate posterior. Laplace approximation [36] is a classic method in this spirit, which assumes a Gaussian posterior, and adapts $\boldsymbol{w}^*$ and the local curvature at $\boldsymbol{w}^*$ as the posterior mean and variance, respectively. Yet, Laplace approximation is inflexible and usually computationally prohibitive. Alternatively, we develop the more practical *Bayesian fine-tuning* scheme, whose core notion is to fine-tune the imperfect variational posterior by maximizing ELBO.

2.2 Bayesian Fine-tuning

In *Bayesian fine-tuning*, the configuration of the variational distribution $q(\boldsymbol{w}|\boldsymbol{\theta})$ plays a decisive role. Although a wealth of variationals have emerged for adoption [34, 32, 55], on one hand, more complicated ones [35, 48] are routinely accompanied by less scalable and less amenable learning procedures; on the other hand, the aforementioned hypothesis inspiring *Bayesian fine-tuning* entails an explicit alignment between the DNN parameters $\boldsymbol{w}^*$ and the variational parameters $\boldsymbol{\theta}$. In this sense, we primarily concern the typical, efficient, yet less effective *mean-field Gaussian* distribution as well as a more powerful one which resembles Deep Ensemble [30].

Mean-field Gaussian (*MFG*) variational. Without losing generality, we write the *MFG* variational as $q(\boldsymbol{w}|\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{\mu}, \mathbf{diag}(\exp(2\boldsymbol{\psi})))$, with $\boldsymbol{\mu}, \boldsymbol{\psi} \in \mathbb{R}^p$ denoting the mean and the logarithm of standard deviation respectively. In this sense, we can naturally initialize $\boldsymbol{\mu}$ with $\boldsymbol{w}^*$ at the beginning of fine-tuning to ease the approximate posterior inference and to enable the investigation of more qualified posterior modes. As in MAP, we also assume an isotropic Gaussian prior $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \mathbf{0}, \sigma_0^2\mathbf{I})$. Then the gradients of the *complexity loss* can be derived analytically:

$$\nabla_{\boldsymbol{\mu}}\mathcal{L}_c = -\lambda\boldsymbol{\mu}, \quad \nabla_{\boldsymbol{\psi}}\mathcal{L}_c = -\lambda\exp(2\boldsymbol{\psi}) + \frac{1}{n}, \quad \lambda = \frac{1}{\sigma_0^2 n}. \tag{5}$$

Intuitively, the above gradients for the variational parameters correspond to a variant of the vanilla weight decay in DNNs. Having identified this, we can implement a weight decay-like module to implicitly be responsible for the *complexity loss*, leaving only the *expected log-likelihood* $\mathcal{L}_{ell}$ required to be explicitly handled. We will elaborate the details of solving $\max \mathcal{L}_{ell}$ after presenting a more expressive variational configuration.

Parameter-sharing ensemble (*PSE*) variational. Despite simplicity, the *MFG* variational can be limited in expressiveness for capturing the multi-modal parameter posterior of over-parameterized neural networks. Empowered by the observation that Deep Ensemble [30] is a compelling Bayesian marginalization mechanism in deep learning [58], we intend to develop a low-cost ensemble-like variational for more practical Bayesian deep learning.

Specifically, we first define the variational as a uniform mixture of C Gaussians: $q(\boldsymbol{w}|\boldsymbol{\theta}) = \frac{1}{C}\sum_c \mathcal{N}(\boldsymbol{w}; \boldsymbol{w}^{(c)}, \boldsymbol{\Sigma}^{(c)})$, where $\boldsymbol{\Sigma}^{(c)} \in \mathbb{R}^{p\times p}$ is positive-definite and its elements are independent of the dimension p.[2] In this sense, the *complexity loss* boils down to the KL divergence between a mixture of Gaussians and a Gaussian, which, yet, cannot be calculated analytically in general. Nevertheless, under the mild assumption that $\boldsymbol{w}^{(c)} \in \mathbb{R}^p$ is normally distributed and p is large enough, the KL divergence can be approximated by a weighted sum of the KL divergences between the Gaussian components and the Gaussian prior (refer to [12] for detailed discussion and proof). Namely:

$$-\frac{1}{n} D_{\mathrm{KL}}\left(\frac{1}{C}\sum_c \mathcal{N}(\boldsymbol{w}; \boldsymbol{w}^{(c)}, \boldsymbol{\Sigma}^{(c)}) \| \mathcal{N}(\boldsymbol{w}; \mathbf{0}, \sigma_0^2\mathbf{I})\right) \approx -\frac{1}{2\sigma_0^2 nC}\sum_{c=1}^{C}\left(\|\boldsymbol{w}^{(c)}\|_2^2 + \mathrm{trace}(\boldsymbol{\Sigma}^{(c)}) - \sigma_0^2 \log|\boldsymbol{\Sigma}^{(c)}|\right) + \text{constant}. \tag{6}$$

Based on the observation that Bayesian model average benefits significantly more from the exploration of new modes than navigation around a local mode [58], we assume $\boldsymbol{\Sigma}^{(c)}, c = 1, ..., C$, to be a constant diagonal matrix $\sigma^2\mathbf{I}$ with σ^2 approaching $\mathbf{0}$. Namely, we purely chase multi-mode exploration and leave the joint optimization of $\boldsymbol{\Sigma}^{(c)}$ and $\boldsymbol{w}^{(c)}$ for future investigation. Then, $q(\boldsymbol{w}|\boldsymbol{\theta})$ almost amounts to a mixture of deltas (i.e., an ensemble) and with high probability we can approximate the realisation of $\boldsymbol{w}$ by a uniform sample from $\{\boldsymbol{w}^{(1)}, ..., \boldsymbol{w}^{(C)}\}$. Meanwhile, the *complexity loss* approximately becomes $-\frac{\lambda}{2C}\sum_{c=1}^{C}\|\boldsymbol{w}^{(c)}\|_2^2 + \text{constant}$, and we can easily implement a weight decay-like module to be responsible for its gradient. We comment here that it may be more plausible to alternatively leverage the rigorous quasi-KL divergence [20] for estimating the divergence between a mixture of deltas and the Gaussian prior, left as a future work.

Simulating an ensemble is far from our ultimate goal due to the required high cost. To make the variational economical, we explore a valuable insight from recent works [54, 57] that the parameters of different ensemble components can be partially shared without undermining effectiveness.

Specifically, abusing $\boldsymbol{w}$ to notate the parameter matrix of size $m_{\text{in}} \times m_{\text{out}}$ in a neural network layer, we generate C components via: $\boldsymbol{w}^{(c)} = \boldsymbol{l}^{(c)}\boldsymbol{r}^{(c)} \circ \bar{\boldsymbol{w}}, c = 1, ..., C$, where $\bar{\boldsymbol{w}} \in \mathbb{R}^{m_{\text{in}}\times m_{\text{out}}}$ are the shared parameters and $\boldsymbol{l}^{(c)} \in \mathbb{R}^{m_{\text{in}}\times r}$ and $\boldsymbol{r}^{(c)} \in \mathbb{R}^{r\times m_{\text{out}}}$ correspond to r-rank decomposition of some component-specialized perturbations. $\circ$ is the element-wise multiplication. The shared parameters $\bar{\boldsymbol{w}}$ can be initialized as $\boldsymbol{w}^*$ to ease and speedup the *Bayesian fine-tuning*. When the rank r is suitably small, the above design can significantly reduce the model size, and save the training effort. Of note that the previous works [54, 57] confine r to be 1 to permit the adoption of a specific gradient variance reduction trick. Conversely, we loosen this constraint by using a more generally applicable variance reduction tactic, detailed below.

Estimation of the expected log-likelihood $\mathcal{L}_{ell}$. Given the high non-linearity of deep NNs and the large volume of data in modern settings, we follow the stochastic variational inference (SVI) pipeline for estimating $\mathcal{L}_{ell}$. Formally, given a mini-batch of data $\mathcal{B} = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{|\mathcal{B}|}$, we solve

$$\max_{\boldsymbol{\theta}} \mathcal{L}'_{ell} = \frac{1}{|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}\log p(y^{(i)}|\boldsymbol{x}^{(i)}; \boldsymbol{w}), \tag{7}$$

where $\boldsymbol{w}$ is drawn from the *MFG* or *PSE* variational via reparameterization [25]. The gradients w.r.t. the variational parameters can be derived automatically with autodiff libraries, thus the training resembles that of regular DNNs.

[2] We define the variational as a mixture of Gaussians instead of a mixture of deltas to ensure the variational is *absolutely continuous* w.r.t. the prior. This avoids the *singularity* of the approximate posterior distribution [20].

```python
def ER_conv(input, theta, stride, padding, groups):
    """input: [b, i, h, w]; theta: variational parameters"""
    b = input.shape[0]
    # sample a batch of conv filters w: [b, o, i, k, k]
    w = mc_sample(theta, num_mc_samples=b)
    # reshape w to have shape [b*o, i, k, k]
    w = w.flatten(start_dim=0, end_dim=1)
    # reshape input to have shape [1, b*i, h, w]
    input = input.flatten(start_dim=0, end_dim=1).unsqueeze(0)
    # perform b convs in parallel
    output = conv2d(input, w, stride, padding, groups*b)
    # reshape the result to standard format
    return output.view(b, -1, output.shape[2], output.shape[3])
```

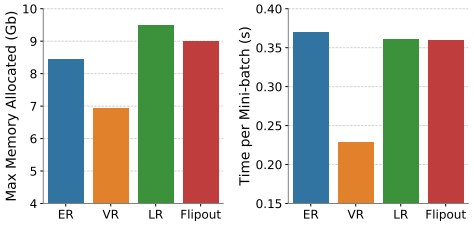


Figure 3: (left): Implementation of *exemplar reparametrization* for 2D convolution in PyTorch [43]. (right): Memory and time cost comparison among *exemplar reparametrization* (ER), vanilla reparametrization (VR) [25], local reparametrization (LR) [26], and Flipout [55] with mean-field Gaussian variational used (estimated on ImageNet with ResNet-50 architecture and batch size as 32).

However, gradients derived by $\mathcal{L}'_{ell}$ might exhibit high variance, caused by sharing the sampled parameters $\boldsymbol{w}$ across data in the mini-batch [26]. Popular techniques for addressing this issue typically assume a restrictive form of variational distribution [26, 55], struggling to handle structured distributions like the proposed *PSE* with > 1 rank. Fortunately, there is a generally applicable strategy for reducing gradient variance in stochastic variational inference named *exemplar reparametrization* (ER), which samples dedicated parameters for every exemplar in the minibatch for estimating $\mathcal{L}_{ell}$:

$$\mathcal{L}^*_{ell} = \frac{1}{|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|} \log p(y^{(i)}|\boldsymbol{x}^{(i)};\boldsymbol{w}^{(i)}),\ \boldsymbol{w}^{(i)} \overset{\text{i.i.d.}}{\sim} q(\boldsymbol{w}|\boldsymbol{\theta}),\ i = 1, ..., |\mathcal{B}|. \tag{8}$$

We can see that the FLOPS of ER are identical to that of vanilla reparameterization, but ER was criticized for that the involved exemplar-wise computations could not be efficiently done within the popular computation libraries in 2015 [26]. With the rapid development of high-performance device-propriety kernel backends (e.g. cuDNN [4]) in recent years, we wonder *is the criticism still hold*? To this end, we first refurbish ER to fit nowadays ML frameworks. Our key insight here is to perform multiple exemplar-wise computations in parallel with a single kernel launch, e.g., organize exemplar-wise matrix multiplications as a `batch matrix multiplication`; organize exemplar-wise convolutions as a `group convolution` (see Figure 3 (left)). We then conduct an empirical study on the computation cost of ER and relevant methods using *MFG* variational. Figure 3 (right) shows the results.

Surprisingly, the time efficiency of ER is comparable with that of local reparameterization [26] and Flipout [55], while the memory cost of ER is even lower. We deduce this is because local reparameterization and Flipout both need to calculate and store one extra mini-batch of feature maps, which are rather large in ImageNet models. Note that the added memory cost of ER upon vanillar reparameterization comes from the storage of a mini-batch of temporary parameters.

A plug-and-play library. We wrap the details of the aforementioned modularized stochastic variational inference and ER strategy for *MFG* and *PSE* in a plug-and-play Python library (attached in supplementary materials) to free the users from the difficulties of implementing *BayesAdapter*.

3 Do Variational BNNs Know What They Do Not Know?

Based on the efficient *BayesAdapter* paradigm, we can perform faster learning of variational BNNs, and hence deeper investigations on their properties.

We first evaluate the predictive uncertainty of *BayesAdapter* as well as popular baselines under regular domain shift. Following [16], we train models on CIFAR-10 [29] training set, and evaluate them on both CIFAR-10 and SVHN [41] test sets. For every confidence threshold $0 \leq \tau < 1$, we calculate the average error rate for points with $\geq \tau$ confidence (all predictions on SVHN data are regarded as incorrect). The results are depicted in Figure 4. It is clear that *BayesAdapter* with *PSE* variational has made more conservative predictions on out-of-distribution (OOD) SVHN data than all baselines. We perform only **12**

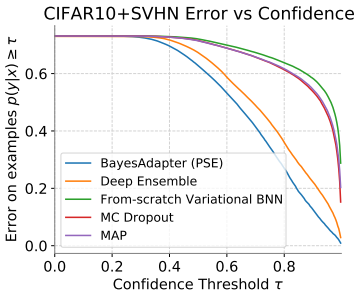
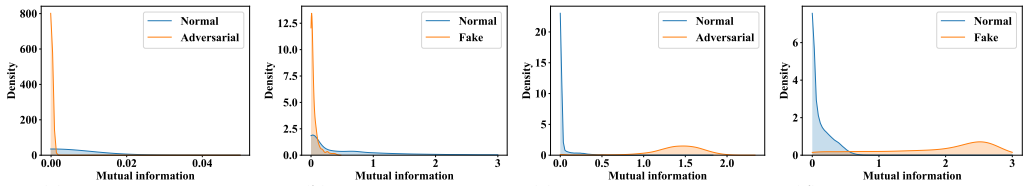


Figure 4: Error vs. confidence plots for models trained on CIFAR-10 and tested on both CIFAR-10 and SVHN.



(a) *BayesAdapter*, C10 (b) *BayesAdapter*, IN (c) *BayesAdapter w/ reg*, C10 (d) *BayesAdapter w/ reg*, IN

Figure 5: (a)-(b): The histograms for the mutual information uncertainty of normal data and OOD data given by *BayesAdapter (MFG)*. (c)-(d): The histograms for the mutual information uncertainty of normal data and OOD data given by *BayesAdapter w/ reg (MFG)*. (C10 refers to CIFAR-10; IN refers to ImageNet.)

epochs of *Bayesian fine-tuning* upon MAP, while the model calibration is significantly promoted. *BayesAdapter* with PSE variational even outperforms the expensive Deep Ensemble, implying that the parameter-sharing mechanism may impose further regularization on model parameters. The comparison also confirms that from-scratch variational BNNs have difficulties to find good posteriors.

We then move to more realistic and malicious OOD data frequently encountered in real world. We train models on CIFAR-10 and ImageNet [6], and test them on the adversarial examples from PGD attack [38] and the fake samples from BigGAN [3], respectively. We only plot the histograms for the *epistemic* uncertainty (estimated by Eq. (4)) from *BayesAdapter* in Figure 5a and 5b, while the other BNN methods behave similarly (see Figure 6 in Sec 4 for evidence). It is evident that the two kinds of OOD data are undesirably assigned systematically lower uncertainty than normal data.

Such an unreliability issue of the uncertainty estimates pose a road-block for the practical application of variational BNNs. To address this pathology, we propose to calibrate the *epistemic* uncertainty of variational BNNs on top of ELBO maximization during *Bayesian fine-tuning*. Basically, we regularize the BNNs to give high *epistemic* uncertainty for a cheap collection of OOD data, so that they acquire the ability of yielding high uncertainty for unseen OOD samples with similar fingerprints. Namely, assuming access to some OOD data $\{\boldsymbol{x}_{\text{o}}^{(i)}\}_{i=1}^{n_{\text{o}}}$, we optimize the following margin loss:

$$\max_{\theta} \mathcal{L}_{reg} = \frac{1}{|\mathcal{B}_{\text{o}}|} \sum_{\boldsymbol{x}_{\text{o}}^{(i)} \in \mathcal{B}_{\text{o}}} \min \Big(\mathcal{I}(\boldsymbol{w}, y | \boldsymbol{x}_{\text{o}}^{(i)}, \mathcal{D}), \gamma \Big), \tag{9}$$

where $\mathcal{B}_{\text{o}}$ refers to a mini-batch of OOD data and γ is a pre-defined threshold. For efficiency, we adopt $S = 2$ MC samples for estimating the uncertainty $\mathcal{I}$ in Eq. (9) during training. The overall objective for *BayesAdapter w/ reg* is then $\mathcal{L}_{ell}^{*} + \alpha \mathcal{L}_{reg}$ with α representing a trade-off coefficient.

We comment that though the regularization may make the learned posterior deviate from the principled posterior associated with the imposed prior, it, yet, is practically effective for improving the quality of learned posteriors, partially because the imposed prior beliefs on model parameters may not be meaningful, especially for deep networks [50]. For verification, we deploy *BayesAdapter w/ reg* to the aforementioned two settings, and observe calibrated *epistemic* uncertainty on OOD data in Figure 5c and 5d. Note that we take uniformly contaminated samples (as a proxy of the expensive adversarial examples crafted by PGD) and 1000 BigGAN samples as the training OOD data for CIFAR-10 and ImageNet respectively, indicating the *sample efficiency* of the uncertainty regularization.

4 Experiments

In this section, we apply *BayesAdapter* to a diverse set of benchmarks for empirical verification.

Settings. In general, we pre-train DNNs following standard protocols or fetch the pre-trained checkpoints available online, and then perform *Bayesian fine-tuning*. We randomly initialize the newly added variational parameters (e.g., $\boldsymbol{\psi}$, $\boldsymbol{l}^{(c)}, \boldsymbol{r}^{(c)}$). Unless otherwise stated, we set rank $r = 1$ and component number $C = 20$ for *PSE*. We use $S = 20$ MC samples for predicting and quantifying *epistemic* uncertainty for both *MFG* and *PSE*. For *BayesAdapter w/ reg*, we set $\alpha = 3$ without tuning and set uncertainty threshold $\gamma = 0.75$ according to an observation that the normal examples usually present < 0.75 mutual information uncertainty across various scenarios. We conduct experiments on 8 RTX 2080Ti GPUs. Full details are deferred to Appendix B.

Baselines. We consider extensive baselines including: (1) *MAP*, which is the fine-tuning start point, (2) *Laplace Approx.*: which preforms Laplace approximation with diagonal Fisher information matrix, (3) *MC Dropout*, which is a dropout variant of *MAP*, (4) *VBNN*, which refers to from-scratch trained variational BNNs under ELBO maximization. In particular, the variational BNN methods

Table 1: Comparison on predictive performance and negative log-likelihood (NLL). We use underline to emphasize the results obtained given significantly more training effort. For *BayesAdapter*, we repeat every experiment for 3 times and report the error bar.

Method	CIFAR-10		ImageNet	
	TOP1 (%) ↑	NLL ↓	TOP1 (%) ↑	NLL ↓
MAP	96.92	0.1312	76.13	0.9618
Laplace Approx.	96.41	0.1204	75.89	0.9739
MC Dropout	96.95	0.1151	74.88	0.9884
SWAG	96.32	0.1122	-	-
Deep Ensemble	**97.40**	**0.0869**	-	-
VBNN (MFG)	96.95	0.0994	75.97	0.9435
VBNN (PSE)	96.88	0.1328	75.12	0.9865
BayesAdapter (MFG)	97.10±0.03	0.1007±0.0014	76.45±0.05	0.9303±0.0005
BayesAdapter (PSE)	**97.13**±0.03	**0.0936**±0.0010	**76.80**±0.03	**0.9159**±0.0010

Table 2: Accuracy ↑ comparison on open-set face recognition with MobileNetV2 architecture.

Method	LFW	CPLFW	CALFW	CFP-FF	CFP-FP
MAP	98.2%	84.0%	87.6%	**97.8**%	92.7%
MC Dropout	98.2%	83.6%	87.3%	**97.8**%	92.8%
BayesAdapter (MFG)	**98.4**%	83.9%	85.8%	97.6%	92.9%
BayesAdapter (PSE)	**98.4**%	**84.7%**	**87.8**%	**97.8**%	**93.1**%

like *BayesAdapter* and *VBNN* are evaluated on both the *MFG* and *PSE* variationals. We also include *Deep Ensemble* [30], and *SWAG* [37] into the comparison on CIFAR-10.[3]

4.1 CIFAR-10 Classification

We first conduct experiments on CIFAR-10 classification with wide-ResNet-28-10 architecture [61]. We perform *Bayesian fine-tuning* for 12 epochs with the weight decay coefficient λ set as 2e-4 following common practice. Table 1 (left) outlines the comparison on prediction performance.

It is worth noting that *BayesAdapter* notably outperforms *MAP*, *Laplace Approx.*, *MC Dropout*, and *SWAG* in aspect of predictive performance. The accuracy upper bound is *Deep Ensemble*, which trains 5 isolated *MAP*s and assembles their predictions to explicitly investigate diverse function modes, but it is orders of magnitude more expensive than *BayesAdapter*. *VBNN* is clearly defeated by *BayesAdapter*, confirming our claim that performing *Bayesian fine-tuning* from the converged deterministic checkpoints is beneficial to explore more qualified posteriors.

Converged ELBO. We compare the converged (training) ELBO of *VBNN* and *BayesAdapter*: the former gives $\mathcal{L}_{ell} = -0.032$ and $\mathcal{L}_c = -2384.3$ while the latter gives $\mathcal{L}_{ell} = -0.019$ and $\mathcal{L}_c = -2806.8$. The comparison implies *Bayesian fine-tuning* can make the approximate posterior *converge*, while the convergence is *distinct* from that of from-scratch variational inference.

CIFAR-10 vs CIFAR-10 corruptions/SVHN. We then compare the predictive uncertainty of *BayesAdapter* as well as the baselines on CIFAR-10 corruptions [18] and SVHN [41]. Figure 1 (right) and 4 present the results, which substantiate the efficacy of *BayesAdapter* for resisting over-fitting.

Speedup. Based on our implementation, SVI with *PSE* takes around 2 minutes for one epoch. Thus, *VBNN* trained from scratch consumes 400 minutes for 200-epoch training, while *BayesAdapter* needs only 24 minutes for 12-epoch fine-tuning, saving 376 minutes ($\mathbf{94}\%$) training time than *VBNN*.

4.2 ImageNet Classification

We then scale up *BayesAdapter* to the large ImageNet dataset with ResNet-50 [17] architecture. We launch fine-tuning for merely **4 epochs** with the weight decay coefficient λ set as 1e-4.

Table 1 (right) reports the empirical comparison. As expected, most of results are consistent with those on CIFAR-10. On this large-scale scenario, it is more clear that the from-scratch learning baseline *VBNN* would suffer from local optima. The striking improvement of *BayesAdapter* upon *MAP* validates the benefits of Bayesian treatment. Zooming in, we also note that *BayesAdapter (PSE)* reveals remarkably higher accuracy than the *BayesAdapter (MFG)*, testifying the enhanced expressiveness of *PSE* over *MFG*.

[3] Currently, we have not scaled *Deep Ensemble* and *SWAG* up to ImageNet due to resource constraints.

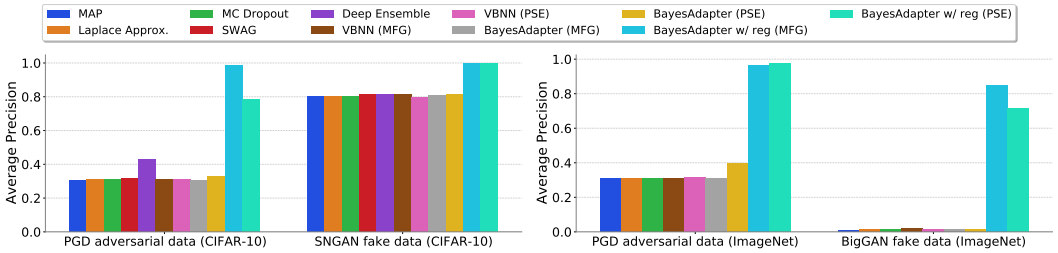


Figure 6: Comparison on the average precision for detecting adversarial/fake examples on CIFAR-10/ImageNet.

Table 4: Ablation study on the rank r of *PSE*. (ImageNet)

Method	TOP1 (%)	# of Param. (M)
MAP	76.13	25.56
BayesAdapter (PSE, r=1)	76.80	27.21
BayesAdapter (PSE, r=8)	76.78	38.76
BayesAdapter (PSE, r=16)	76.80	51.95

4.3 Face Recognition

To demonstrate the universality of *BayesAdapter*, we further apply it to the challenging face recognition task based on MobileNetV2 [45]. We train models on the CASIA dataset [60], and perform comprehensive evaluation on face verification datasets including LFW [22], CPLFW [63], CALFW [64], and CFP [46].We launch fine-tuning for 4 epochs with $\lambda = 5e - 4$. We compare our method to *MAP* and *MC Dropout*, two popular baselines in face recognition field. We depict the comparison on recognition accuracy in Table 2.

It is noteworthy that Bayesian principle can induce better predictive performance for face recognition models. *BayesAdapter (PSE)* has outperformed the fine-tuning start point *MAP* and the popular baseline *MC Dropout* in most verification datasets, despite being fine-tuned for only several rounds.

4.4 Detection of Adversarial and Fake Examples

We then concern the quality of the uncertainty estimates yielded by the trained models for realistic OOD data, including adversarial examples crafted by 20-step PGD following standard protocols and fake samples from performant GANs [39, 3] or DeepFake (see Appendix F for some examples). Concretely, we estimate the *epistemic* uncertainty of the trained models on normal data as well as OOD data, based on which we distinguish the latter from the former. We compute the average precision (AP) of such a binary classification (detection) and plot the comparison in Figure 6.

Briefly speaking, *BayesAdapter w/ reg* significantly surpasses all other methods. In particular, *Deep Ensemble* also yields unreliable predictive uncertainty for the difficult OOD data. These results confirm the unreliability issue of existing BNNs' predictive uncertainty and prove the efficacy of the developed uncertainty regularization. See Table 1 in Appendix D for the classification performance of *BayesAdapter w/ reg*.

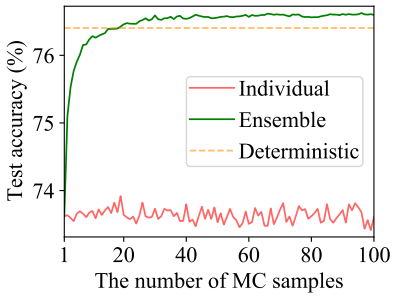
4.5 More Empirical Analyses

Model calibration on in-distribution data. We estimate model calibration, measured by *expected calibration error* (ECE) [15], of the studies methods on in-distribution data, and report the results in Table 3. Notably, the ECE of *BayesAdapter (PSE)* is on par with *Deep Ensemble*, significantly better than the other baselines.

Table 3: Comparison on model calibration (ECE ↓).

Method	CIFAR-10	ImageNet
MAP	0.0198	0.0373
SWAG	0.0088	-
Deep Ensemble	**0.0057**	-
VBNN (MFG)	0.0074	0.0183
VBNN (PSE)	0.0188	0.0202
BayesAdapter (MFG)	0.0091	0.0289
BayesAdapter (PSE)	**0.0058**	**0.0129**

The impact of the rank r for *PSE*. As stated, we set $r = 1$ for all the above studies for maximal parameter saving. Yet, does the small rank r confine the expressiveness of *PSE*? We perform an ablation study to pursue the answer. As shown in Table 4, the capacity of *PSE* can already be sufficiently unleashed when the rank is 1, indicating the merit of *PSE* for efficient learning.



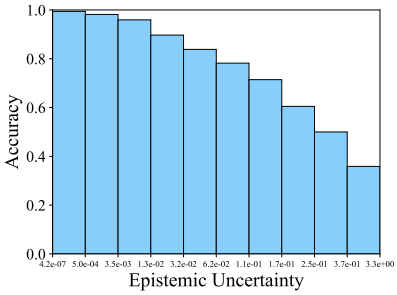


Figure 7: (left): Test accuracy varies w.r.t. the number of MC samples for *Bayes ensemble*. (right): Comparison on the accuracy for instance buckets of equal size but with rising uncertainty. (*BayesAdapter (MFG)*, ImageNet)

The effectiveness of *exemplar reparameterization*. We build a toy model with only a convolutional layer and fix the model input and the target output. We employ the *MFG* variational on the convolutional parameters and computing the variance of stochastic gradients across 500 runs. We average the gradient variance of $\boldsymbol{\mu}$ and $\boldsymbol{\psi}$ over all their coordinates, and observe that vanilla reparameterization typically introduces more than $100\times$ variance than *exemplar reparameterization*.

The impact of ensemble number. We draw the change of test accuracy w.r.t. the number of MC samples S for *Bayes ensemble* in Figure 7 (left). The model is trained by *BayesAdapter (MFG)* on ImageNet. The points on the red line represent the individual accuracies of the 100 parameter samples. The yellow dashed line refers to the deterministic inference with only the Gaussian mean. The green line displays the effects of *Bayes ensemble* – the predictive performance increases from $< 74\%$ to $> 76\%$ quickly before seeing 20 parameter samples, and gradually saturates after that.

Uncertainty-based rejective decision. In practice, we expect our models to be accurate on the data that they are certain about. In this spirit, we gather the *epistemic* uncertainty estimates for ImageNet validation data given by *BayesAdapter (MFG)*, based on which we divide the data into 10 buckets of equal size but with increasing uncertainty. We depict the average accuracy of each bucket in Figure 7 (right). As expected, our BNN is more accurate for instances with smaller uncertainty.

5 Related Work

Fruitful works have emerged in the BNN community in the last decade [14, 53, 2, 25, 1, 33, 23, 59]. However, most of the existing works cannot achieve the goal of practicability. For example, some works trade learning efficiency for flexible variational posteriors, leading to restrictive scalability [34, 35, 47, 50]. [24, 62, 42] build Adam-like optimizers to do variational inference, but their parallel training throughput and compatibility with data augmentation are inferior to SGD. Approximate Bayesian methods, e.g., Monte Carlo (MC) dropout [13] and deep ensemble [30], usually maintain impressive predictive performance, but suffer from degenerated uncertainty estimates [11] or high cost.

Laplace approximation [36, 44] is a known approach to transform a DNN to a BNN, but it is inflexible due to its postprocessing nature and some strong assumptions made for practical concerns. Alternatively, *BayesAdapter* works in the style of fine-tuning, which is more natural and economical for deep networks. Bayesian modeling the last layer of a DNN is proposed recently [28], and its combination with *BayesAdapter* deserves an investigation. *BayesAdapter* connects to MOPED [27] in that their variational configurations are both based on *MAP*. Yet, beyond MOPED, we make valuable technical contributions including *PSE* variational and the refinement of *exemplar reparameterization*. We have also done a thorough investigation on how pre-training benefits variational inference. Anyway, we provide an empirical comparison between *BayesAdapter* and MOPED in Appendix C.

6 Conclusions

In this work, we propose the scalable *BayesAdapter* framework to learn variational BNNs. Our core idea is to perform *Bayesian fine-tuning* instead of expensive from-scratch Bayesian learning. We develop plug-and-play implementations for the stochastic variational inference under two representative variational distributions, and refine *exemplar reparameterization* to efficiently reduce gradient variance. We also propose a biased yet meaningful uncertainty regularization to calibrate the *epistemic* uncertainty of variational BNNs. We evaluate *BayesAdapter* in diverse real-world scenarios and report promising results.

References

[1] Anoop Korattikara Balan, Vivek Rathod, Kevin P Murphy, and Max Welling. Bayesian dark knowledge. In *Advances in Neural Information Processing Systems*, pages 3438–3446, 2015.

[2] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pages 1613–1622, 2015.

[3] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[4] Sharan Chetlur, Cliff Woolley, Philippe Vandermersch, Jonathan Cohen, John Tran, Bryan Catanzaro, and Evan Shelhamer. cudnn: Efficient primitives for deep learning. *arXiv preprint arXiv:1410.0759*, 2014.

[5] Deepfakes, 2018. `https://github.com/deepfakes/faceswap`. Accessed: 2018-10-29.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[7] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019.

[8] Stefan Depeweg, José Miguel Hernández-Lobato, Finale Doshi-Velez, and Steffen Udluft. Learning and policy search in stochastic dynamical systems with Bayesian neural networks. *arXiv preprint arXiv:1605.07127*, 2016.

[9] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[10] Faceswap, 2018. `https://github.com/MarekKowalski/FaceSwap`. Accessed: 2018-10-29.

[11] Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019.

[12] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: appendix. *arXiv preprint arXiv:1506.02157*, 420, 2015.

[13] 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, 2016.

[14] Alex Graves. Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems*, pages 2348–2356, 2011.

[15] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. *arXiv preprint arXiv:1706.04599*, 2017.

[16] Bobby He, Balaji Lakshminarayanan, and Yee Whye Teh. Bayesian deep ensembles via the neural tangent kernel. *arXiv preprint arXiv:2007.05864*, 2020.

[17] 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.

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

[19] José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869, 2015.

[20] Jiri Hron, Alex Matthews, and Zoubin Ghahramani. Variational bayesian dropout: pitfalls and fixes. In *International Conference on Machine Learning*, pages 2019–2028. PMLR, 2018.

[21] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708, 2017.

[22] Gary B Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. In *Technical report*, 2007.

[23] Alex Kendall and Yarin Gal. What uncertainties do we need in Bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems*, pages 5574–5584, 2017.

[24] Mohammad Emtiyaz 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 2616–2625, 2018.

[25] Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[26] Durk P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems*, pages 2575–2583, 2015.

[27] 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.

[28] Agustinus Kristiadi, Matthias Hein, and Philipp Hennig. Being bayesian, even just a bit, fixes overconfidence in relu networks. *arXiv preprint arXiv:2002.10118*, 2020.

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

[30] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pages 6402–6413, 2017.

[31] Christian Leibig, Vaneeda Allken, Murat Seçkin Ayhan, Philipp Berens, and Siegfried Wahl. Leveraging uncertainty information from deep neural networks for disease detection. *Scientific Reports*, 7(1):1–14, 2017.

[32] Yingzhen Li and Richard E Turner. Gradient estimators for implicit models. *arXiv preprint arXiv:1705.07107*, 2017.

[33] Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose Bayesian inference algorithm. In *Advances in Neural Information Processing Systems*, pages 2378–2386, 2016.

[34] Christos Louizos and Max Welling. Structured and efficient variational deep learning with matrix gaussian posteriors. In *International Conference on Machine Learning*, pages 1708–1716, 2016.

[35] Christos Louizos and Max Welling. Multiplicative normalizing flows for variational Bayesian neural networks. In *International Conference on Machine Learning*, pages 2218–2227, 2017.

[36] David JC MacKay. A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992.

[37] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems*, pages 13153–13164, 2019.

[38] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[39] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[40] Radford M Neal. *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, 1995.

[41] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[42] Kazuki Osawa, Siddharth Swaroop, Anirudh Jain, Runa Eschenhagen, Richard E Turner, Rio Yokota, and Mohammad Emtiyaz Khan. Practical deep learning with Bayesian principles. *arXiv preprint arXiv:1906.02506*, 2019.

[43] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.

[44] 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.

[45] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[46] Soumyadip Sengupta, Jun-Cheng Chen, Carlos Castillo, Vishal M Patel, Rama Chellappa, and David W Jacobs. Frontal to profile face verification in the wild. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9. IEEE, 2016.

[47] Jiaxin Shi, Shengyang Sun, and Jun Zhu. Kernel implicit variational inference. In *International Conference on Learning Representations*, 2018.

[48] Jiaxin Shi, Shengyang Sun, and Jun Zhu. A spectral approach to gradient estimation for implicit distributions. *arXiv preprint arXiv:1806.02925*, 2018.

[49] Lewis Smith and Yarin Gal. Understanding measures of uncertainty for adversarial example detection. *arXiv preprint arXiv:1803.08533*, 2018.

[50] Shengyang Sun, Guodong Zhang, Jiaxin Shi, and Roger Grosse. Functional variational Bayesian neural networks. In *International Conference on Learning Representations*, 2019.

[51] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Niebner. Face2face: Real-time face capture and reenactment of rgb videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2387–2395, 2016.

[52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[53] 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.

[54] Yeming Wen, Dustin Tran, and Jimmy Ba. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. *arXiv preprint arXiv:2002.06715*, 2020.

[55] Yeming Wen, Paul Vicol, Jimmy Ba, Dustin Tran, and Roger Grosse. Flipout: Efficient pseudo-independent weight perturbations on mini-batches. *arXiv preprint arXiv:1803.04386*, 2018.

[56] Florian Wenzel, Kevin Roth, Bastiaan S Veeling, Jakub Swiatkowski, 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.

[57] Florian Wenzel, Jasper Snoek, Dustin Tran, and Rodolphe Jenatton. Hyperparameter ensembles for robustness and uncertainty quantification. *arXiv preprint arXiv:2006.13570*, 2020.

[58] Andrew Gordon Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. *arXiv preprint arXiv:2002.08791*, 2020.

[59] Anqi Wu, Sebastian Nowozin, Edward Meeds, Richard E Turner, Jose Miguel Hernandez-Lobato, and Alexander L Gaunt. Deterministic variational inference for robust bayesian neural networks. *arXiv preprint arXiv:1810.03958*, 2018.

[60] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014.

[61] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[62] Guodong Zhang, Shengyang Sun, David Duvenaud, and Roger Grosse. Noisy natural gradient as variational inference. In *International Conference on Machine Learning*, pages 5847–5856, 2018.

[63] Tianyue Zheng and Weihong Deng. Cross-pose lfw: A database for studying cross-pose face recognition in unconstrained environments. *Beijing University of Posts and Telecommunications, Tech. Rep*, 5, 2018.

[64] Tianyue Zheng, Weihong Deng, and Jiani Hu. Cross-age lfw: A database for studying cross-age face recognition in unconstrained environments. *arXiv preprint arXiv:1708.08197*, 2017.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See L215.
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Sec 4.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Table 1.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Sec 4.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Exemplar Fully-connected Layer

As introduced in Sec 2.2, the regular convolution can be elegantly converted into an exemplar version by resorting to group convolution. The other popular operators are relatively easy to handle. Take the fully-connected (FC) layer as an example: assuming a feature $x \in \mathbb{R}^{b \times i}$, we draw b i.i.d. FC weights and concatenate them as $w \in \mathbb{R}^{b \times i \times o}$, then invoke $\texttt{batch\_matmul}(x, w)$ to get the output.

B More Experimental Details

The only important hyper-parameter is the weight decay coefficient λ. Other hyper-parameters for defining PGD or specifying learning rates, etc., all follow standard practice in the DL community. The number of fake data training (1000) and the number of MC samples for evaluation (S) are flexible and not tuned.

Table 5: Predictive performance of *BayesAdapter w/ reg*.

Method	CIFAR-10		ImageNet	
	TOP1 (%) ↑	NLL ↓	TOP1 (%) ↑	NLL ↓
BayesAdapter (MFG)	97.10±0.03	0.1007±0.0014	76.45±0.05	0.9303±0.0005
BayesAdapter (PSE)	**97.13**±0.03	**0.0936**±0.0010	**76.80**±0.03	**0.9159**±0.0010
BayesAdapter w/ reg (MFG)	96.82±0.07	0.1004±0.0026	76.26±0.06	0.9428±0.0020
BayesAdapter w/ reg (PSE)	96.86±0.06	0.1173±0.0030	76.48±0.06	0.9752±0.0030

For λ, we keep it consistent between pre-training and fine-tuning, without elaborated tuning, for example, $\lambda = 2e-4$ for the wide-ResNet-28-10 architecture on CIFAR-10, $\lambda = 1e-4$ for ResNet-50 architecture on ImageNet, and $\lambda = 5e-4$ for MobileNet-V2 architecture on CASIA. These values correspond to isotropic Gaussian priors with σ_0^2 as 0.1, 0.0078, and 0.0041 on CIFAR-10, ImageNet, and CASIA, respectively. It is notable that for a "small" dataset like CIFAR-10, a flatter prior is preferred. While on larger datasets with stronger data evidence, we need a sharper prior for regularization.

For the pre-training, we follow standard protocols available online. On CIFAR-10, we perform CutOut [9] transformation upon popular resize/crop/flip transformation for data augmentation. On ImageNet, we leverage the ResNet-50 checkpoint on PyTorch Hub as the converged deterministic model. On face tasks, we train MobileNetV2 following popular hyper-parameter settings, and the pre-training takes 90 epochs. We use the same weight decay coefficients in both the pre-training and the fine-tuning.

For models on face recognition, we utilize the features before the last FC layer of the MobileNetV2 architecture to conduct feature distance-based face classification in the validation phase, due to the open-set nature of the validation data. The *Bayes ensemble* is similarly achieved by assembling features from multiple runs as the final feature for estimating predictive performance. But we still adopt the output from the last FC layer for uncertainty estimation (i.e., calculating Eq. (4)).

For the success of *BayesAdapter w/ reg*, the uncertainty threshold γ plays a vital role. We use $\gamma = 0.75$ for training across all the scenarios as introduced in Sec 4. But it is not used for OOD detection in the testing phase. For estimating the results of OOD detection, we use the non-parametric metric average precision (see the experiment setup in Sec 4), which is the Area Under the Precision-Recall Curve and is more suitable than the ROC-AUC metric when there is class imbalance. When uniformly perturbing samples (as said, they are a cheap proxy of "real" adversarial examples) to construct training OOD set, the budget is identical to the evaluation budget. We adopt PGD for generating adversarial samples in the validation phase. Concretely, we attack the *posterior predictive* objective, i.e., Eq. (3), with $S = 20$ MC samples. On CIFAR-10, we set perturbation budget as 0.031 and perform PGD for 20 steps with step size at 0.003. On ImageNet and face recognition, we set perturbation budget as $16/255$ and perform PGD for 20 steps with step size at $1/255$.

Regarding the fake data, we craft 1000 fake samples for training and 10000 ones for evaluation with SNGAN [39] on CIFAR-10; we craft 1000 fake samples for training and 1000 ones for evaluation with BigGAN [3] on ImageNet; we randomly sample 1000 fake samples for training and 10000 ones for evaluation from DeepFakes [5], FaceSwap [10] and Face2Face [51] on face recognition. We perform intensive data augmentation for fake training data with a random strategy including Gaussian blur, JPEG compression, *etc.*

As for the *MC dropout*, we add dropout-0.3 (0.3 denotes the dropout rate) before the second convolution in the residual blocks in wide-ResNet-28-10, dropout-0.2 after the second and the third convolutions in the bottleneck blocks in ResNet-50, and dropout-0.2 before the last fully connected (FC) layer in MobileNetV2.

For reproducing *Deep Ensemble*, we train 5 *MAP*s separately, and assemble them for prediction and uncertainty quantification. For reproducing *SWAG*, we take use of its official implementation, and leverage 20 MC samples for decision making.

C Comparison Between *BayesAdapter* and MOPED

We emphasize that MOPED solves the *prior specification* problem for BNNs while *BayesAdapter* constitutes a *practical* framework to *bring variational BNNs to the masses*. Empirically, we evaluate

Table 6: Comparison on the quality of uncertainty estimates for *adversarial* samples in terms of AP ↑ on face recognition.

Method	LFW	CPLFW	CALFW	CFP-FF	CFP-FP
MAP	0.191	0.192	0.191	0.211	0.205
MC dropout	0.965	0.946	0.959	0.965	0.949
BayesAdapter (MFG)	0.232	0.212	0.236	0.242	0.219
BayesAdapter (PSE)	0.939	0.746	0.936	0.923	0.667
BayesAdapter w/ reg (MFG)	**0.998**	**0.981**	**0.999**	**0.999**	**0.983**
BayesAdapter w/ reg (PSE)	**1.000**	**1.000**	**0.999**	**1.000**	**1.000**

Table 7: Comparison on the quality of uncertainty estimates for *DeepFake* samples in terms of AP ↑ on face recognition.

Method	LFW	CPLFW	CALFW	CFP-FF	CFP-FP
MAP	0.389	0.456	0.375	0.394	0.454
MC dropout	0.846	0.664	0.862	0.874	0.685
BayesAdapter (MFG)	0.761	0.520	0.788	0.738	0.441
BayesAdapter (PSE)	0.885	0.577	0.899	0.864	0.504
BayesAdapter w/ reg (MFG)	**0.998**	**0.987**	**0.999**	**0.999**	**0.986**
BayesAdapter w/ reg (PSE)	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**

MOPED on CIFAR-10 with MFG variational, and get 0.0143 training loss ($\mathcal{L}_{ell}$), 96.92% top1 accuracy, 0.1001 test NLL, and 0.0100 ECE. Compared to *BayesAdapter*'s results (0.0191, 97.10%, 0.1007, and 0.0091), we find MOPED exhibits more seriously over-fitting, implying that taking MAP as prior poses under-regularization.

D The Predictive Performance of *BayesAdapter w/ reg*

We report the predictive performance of *BayesAdapter w/ reg* in Table 5. As shown, the regularization $\mathcal{L}_{reg}$ slightly undermines the performance. We think this is reasonable since the uncertainty regularization enforces the model to trade partial capacity for the fidelity of uncertainty estimates. Nevertheless, on ImageNet, *BayesAdapter w/ reg* is still better than its fine-tuning start point *MAP* and the from-scratch baseline *VBNN*.

E Detection of Adversarial and Fake Examples on Face Recognition

We provide the results for the detection of adversarial and fake examples on face recognition in Table 6 and 7. It is an immediate observation that *BayeAdapter w/ reg* outperforms the baselines significantly, and can detect almost all the OOD instances across the validation datasets. By contrast, *BayeAdapter* and *MAP* are similarly unsatisfactory. Surprisingly, *MC dropout* exhibits some capacity to detect adversarial instances and DeepFake ones in the face tasks.

F Visualization of Realistic OOD Data

We provide some random samples of the realistic OOD data used for evaluation in Figure 8.

G Visualization of the Learned Posterior

We plot the parameter posterior of the first convolutional kernel in ResNet-50 architecture learned by *BayesAdapter (MFG)* on ImageNet in Figure 9. The learned posterior variance seems to be disordered, unlike the mean. We leave more explanations as future work.

Figure 8: Some random samples of the realistic OOD data used for evaluation. The first row refers to the fake samples from BigGAN on ImageNet. The second row refers to the adversarial examples generated by PGD on ImageNet. The third row refers to the fake samples from DeepFake.

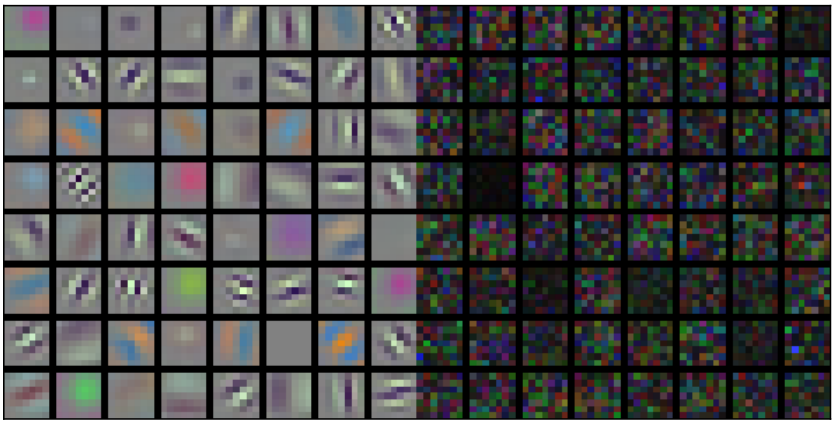


Figure 9: Left: the mean of the *MFG* posterior. Right: the variance of the *MFG* posterior. These correspond to a convolutional kernel with 64 output channels and 3 input channels, where every output channel is plotted as a separate image.