Improving Bayesian Inference in Deep Neural Networks with Variational Structured Dropout

Son Nguyen⋄   Duong Nguyen‡   Khai Nguyen⋄   Nhat Ho†   Khoat Than‡   Hung Bui⋄

VinAI Research, Vietnam⋄; University of Texas, Austin†;
Hanoi University of Science and Technology‡

February 17, 2021

Abstract

Approximate inference in deep Bayesian networks exhibits a dilemma of how to yield high fidelity posterior approximations while maintaining computational efficiency and scalability. We tackle this challenge by introducing a new variational structured approximation inspired by the interpretation of Dropout training as approximate inference in Bayesian probabilistic models. Concretely, we focus on restrictions of the factorized structure of Dropout posterior which is inflexible to capture rich correlations among weight parameters of the true posterior, and we then propose a novel method called *Variational Structured Dropout* (VSD) to overcome this limitation. VSD employs an orthogonal transformation to learn a structured representation on the variational Dropout noise and consequently induces statistical dependencies in the approximate posterior. We further gain an expressive Bayesian modeling for VSD via proposing a hierarchical Dropout procedure that corresponds to the joint inference in a Bayesian network. Moreover, we can scale up VSD to modern deep convolutional networks in a direct way with low computational cost. Finally, we conduct extensive experiments on standard benchmarks to demonstrate the effectiveness of VSD over state-of-the-art methods on both predictive accuracy and uncertainty estimation.

1   Introduction

Bayesian Neural Networks (BNNs) [37, 47] offer a probabilistic interpretation for deep learning models by imposing a prior distribution on the weight parameters and aim to obtain a posterior distribution instead of only point estimates. By marginalizing over this posterior for prediction, BNNs perform a procedure of ensemble learning. These principles facilitate the model to improve generalization, robustness and allow for uncertainty quantification. However, computing exactly the posterior of non-linear Bayesian networks is infeasible and approximate inference has been devised. The core challenge is how to construct an expressive approximation for the true posterior while maintaining computational efficiency and scalability, especially for modern deep learning architectures.

Variational inference is a popular deterministic approximation approach to to deal with this challenge. The first practical methods are proposed in [15, 5, 28], in which, the approximate posterior is assumed to be a fully factorized distribution, also called mean-field variational inference. Generally, the mean-field approximation family encourages some advantages in inference including computational tractability and effective optimization with the stochastic gradient-based methods. However, it will ignore strong statistical dependencies among random weights of the neural networks, which leads to an inability to capture the complicated structure of the true posterior and to estimate true model uncertainty.

To overcome this limitation, many of extensive studies proposed to approximate the true posterior with richer expressiveness. [35] treats the weight matrix as a whole via a matrix variate Gaussian [17] and approximate the posterior based on this parametrization. Several later works have inherited and exploited this distribution to investigate different structured representations for the variational Gaussian posterior, such as Kronecker-factored [64, 54, 53], k-tied distribution [57], non-centered or rank-1 parameterization [14, 8]. Another original idea to represent the true covariance matrix of Gaussian posterior is to employ the low-rank approximation [49, 24, 59]. For robust approximation with multimodality, [36] adopted Hierarchical variational model framework [51] for inferring an implicit marginal distribution in high dimensional Bayesian setting. Despite significant improvements in both predictive accuracy and uncertainty estimation, some of these methods incur a large computational complexity and are difficult to integrate into modern deep convolutional networks.

Motivations. In this paper, we approach the structured posterior approximation in Bayesian neural networks from a different perspective which has been inspired by the Bayesian interpretation of Dropout training [55, 38]. More specifically, the methods proposed in [12, 13, 28] reinterpret Dropout regularization as approximation inference in deep Bayesian models and based on this connection to learn a variational Dropout posterior over the weight parameters. From the literature, we are promoted by the fact that inference approaches based on Bayesian Dropout are often asymptotic or comparable in terms of predictive accuracy to the structured Bayesian methods aforementioned, but with much cheaper computational complexity. Moreover, with impressive empirical results on various tasks along with solid theories on effective regularization [60, 18, 62], generalization bound [39, 44], convergence rate or robust optimization [41, 40], Dropout principle offers several potentials to further improve approximate inference in deep Bayesian nets. However, these Bayesian Dropout methods have also employed a simple structure of the mean-field family, their approximation can suffer from some pathologies such as underestimating the model uncertainty [10], ill-posed or singularity inference [22]. We also clarify that our work has not been motivated primarily for the purpose of improving dropout technique via the Bayesian perspective, but more than that, we aim to exploit the relationship between dropout training and Bayesian statistics to propose a more effective approximate inference framework for deep Bayesian models.

Contributions. With the above insights, we propose a novel structured variational inference framework based on Dropout principle. Our method adopts an orthogonal approximation called Householder transformation to learn a structured representation for multiplicative Gaussian noise in Variational Dropout method [28, 43, 33]. As a consequence of the Bayesian interpretation, we go beyond the mean-field family and obtain a variational Dropout posterior with structured covariance. Furthermore, to make our approximation more expressive, we exploit a hierarchical Dropout procedure, which is equivalent to infer a joint posterior in a hierarchical Bayesian nets. We name the proposed method as *Variational Structured Dropout* (VSD) and summarize its advantages as follows:

1. Our structured approximation is implemented on low dimensional noise with considerable computational efficiency. By low-rank approximation, we can greatly reduce the number of parameters compared to mean-field BNNs.

2. VSD can be employed for deep convolutional networks while maintaining the backpropagation in parallel and optimizing efficiently with stochastic gradient methods. Finally, we carry out extensive experiments with standard datasets and network architectures to demonstrate the effectiveness of the proposed method in predictability and scalability.

Organization. The rest of paper is organized as follows. We provide background for variational

inference in Bayesian neural networks and describe the variational Bayesian Dropout in Section 2. Then we present variational structured Dropout and provide in-depth analyses of its expressiveness and scalability in Section 3. We empirically show that our method improves both predictive accuracy and uncertainty measures over existing methods in Section 4. We conclude the paper in Section 5 while deferring the additional materials in the Supplementary Material.

2 Background

2.1 Variational inference for Bayesian neural networks

Given a dataset $\mathcal{D}$ consisting of input-output pairs $(\mathbf{X}, \mathbf{Y}) = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N$ and let $\mathbf{W} = \{\mathbf{W}^{(l)}\}_{l=1}^L$ denote the weight matrices of L layers. In Bayesian neural networks (BNNs), we impose a prior distribution $p(\mathbf{W})$ whose form is in a tractable parametric family and aim to approximate the intractable posterior distribution $p(\mathbf{W}|\mathcal{D})$. Variational inference (VI) [21, 23] can do this by specifying a variational distribution $q_\phi(\mathbf{W})$ with free parameter and then minimizing the Kullback-Leibler (KL) divergence $\mathbb{D}_{KL}(q_\phi(\mathbf{W})\|p(\mathbf{W}|\mathcal{D}))$. This optimization is equivalent to maximizing the evidence Lower Bound with respect to variational parameters ϕ as follows:

$$\mathcal{L}(\phi) = \mathbb{E}_{q_\phi(\mathbf{W})} \log p(\mathcal{D}|\mathbf{W}) - \mathbb{D}_{KL}(q_\phi(\mathbf{W})\|p(\mathbf{W}))$$

. By combining the reparameterization trick [29] with the Monte Carlo sampling, an unbiased differentiable estimation can be derived for this variational objective. Then, it can be effectively optimized using stochastic gradient methods with the variance reduction technique such as the local reparameterization trick [28].

2.2 Variational Bayesian inference with Dropout regularization

Given a deterministic neural network with the weight parameter Θ of size $K \times L$. Training this model with stochastic regularization technique such as Dropout [20, 55] can be interpreted as approximate inference in Bayesian probabilistic models. This is because injecting a Dropout noise into the inputs of a particular layer is equivalent to multiplying the rows of the subsequent weight matrix by the same random variable, namely with each data point $(\mathbf{x}_n, \mathbf{y}_n)$ and a noise vector $\mathbf{S}$, we have: $\mathbf{y}_n = (\mathbf{x}_n \odot \mathbf{S})\Theta = \mathbf{x}_n \text{diag}(\mathbf{S})\Theta$. This induce a BNN with random weight matrix defined by $\mathbf{W} = \text{diag}(\mathbf{S})\Theta$, where the Dropout noise $\mathbf{S}$ is sampled from Bernoulli(p) or multiplicative Gaussian $\mathcal{N}(\mathbf{1}_K, \text{diag}(\alpha))$. Applying the variational inference to this Bayesian model, with some specific choices for prior and approximate posterior, the variational lower bound can resemble the form of the Dropout objective in the original deterministic network. This principle is referred to as the KL condition [11].

[13] and [28] have based on this principle to proposed Bayesian Dropout inference methods such as MC Dropout (MCD) and Variational Dropout (VD). More specifically, to satisfy the KL condition, MCD performs approximate inference with isotropic Gaussian prior $p(\mathbf{W}) = \mathcal{N}(0, l^{-2}\mathbf{I}_L)$ and the variational posterior is a mixture of delta peaks:

$$q_\phi(\mathbf{W}) = \prod_{k=1}^K (p_k \mathcal{N}(\Theta_k, \sigma^2 \mathbf{I}_L) + (1 - p_k)\mathcal{N}(0, \sigma^2 \mathbf{I}_L)).$$

3

Meanwhile, VD employed log-uniform prior and factorized Gaussian posterior:

$$p(|w_{ij}|) \propto 1/|w_{ij}|, \; q_\phi(w_{ij}) = \mathcal{N}(\Theta_{ij}, \alpha_i \Theta_{ij}^2).$$

Bayesian Dropout inference are practical approximate frameworks especially in high dimensional space. However, the scope of Bayesian modeling in these methods is restricted in terms of flexibility of the prior and approximate posterior. Concretely, the Dropout posteriors $q_\phi(\mathbf{W})$ in the above interpretations have simple structures of mean-field approximation which are not expressive enough to capture complicated correlations in the true posterior, leading to underestimating the model uncertainty. Moreover, the log-uniform prior in VD is an improper-prior, which can result in the singularity of the variational objective and cause the inference of posterior to be ill-posed [22].

3 Variational Structured Dropout

We focus on Bayesian Dropout methods using multiplicative noise, in which the distribution of dropout variable $\mathbf{S}$ has the form of diagonal Gaussian, namely $q_\alpha(\mathbf{S}) = \mathcal{N}(\mathbf{1}_K, D(\alpha))$ with $D(.)$ denotes diagonal matrix. Putting a random noise $\mathbf{S} \sim q_\alpha(\mathbf{S})$ on the weight parameter Θ will induce the Dropout posterior $q_\phi(w_{ij}) = \mathcal{N}(\Theta_{ij}, \alpha_i \Theta_{ij}^2)$ where $\mathbf{W} = \mathbf{S} \odot \Theta$. The parameters $\phi = (\alpha, \Theta)$ are then optimized via maximizing a variational lower bound as follows:

$$\begin{aligned}
\mathbb{E}_{q_\phi(\mathbf{W})} &\log p(\mathcal{D}|\mathbf{W}) - \mathbb{D}_{KL}(q_\phi(\mathbf{W})\|p(\mathbf{W})) \\
&= \mathbb{E}_{q_\alpha(\mathbf{S})} \log p(\mathcal{D}|\mathbf{S}, \Theta) - \mathbb{D}_{KL}(q_\phi(\mathbf{W})\|p(\mathbf{W})).
\end{aligned} \quad (1)$$

Intuitively, a richer representation for the noise distribution can enrich the expressiveness of Dropout posterior via the Bayesian interpretation. The recent advances in variational inference have offered many modern techniques to induce flexible distributions, such as normalizing flow [52, 27, 3], auxiliary random variable and implicit distribution [51, 63], or mixture approximation [65, 1]. However, applying these techniques to Bayesian Dropout frameworks requires dealing with certain challenges including the difficulty in parallelizing backpropagation, high computational complexity, and more importantly, how to ensure the KL condition. We adopt an orthogonal approximation called Householder transformation to address these problems.

3.1 Neural orthogonal approximation

We implement the above idea with an assumption that the Dropout noise is sampled from a Gaussian distribution with full covariance instead of a diagonal structure, namely, $q_\alpha(\mathbf{S}) = \mathcal{N}(\mathbf{1}_K, \Sigma)$ with Σ is a positive definite matrix of size $K \times K$. To make this covariance matrix learnable, we first represent Σ in the form of the spectral decomposition: $\Sigma = \mathbf{P}\Lambda\mathbf{P}^T$, where $\mathbf{P}$ is an orthogonal matrix with its eigenvectors in columns, Λ is a diagonal matrix where diagonal elements are the eigenvalues, and superscript (T) denotes the matrix transposition. By the basis-kernel representation theorem [4, 56], we parameterize the orthogonal matrix $\mathbf{P}$ as a product of Householder matrices in the following form:

$$\mathbf{P} = \mathbf{H}_T \mathbf{H}_{T-1}...\mathbf{H}_1 \quad (2)$$

where $\mathbf{H}_t = \mathbf{I} - 2\mathbf{v}_t\mathbf{v}_t^T/\|\mathbf{v}_t\|_2^2$ with $\mathbf{v}_t$ is the Householder vector of size K, and T is the degree of $\mathbf{P}$. This parameterization relaxes the orthogonal constraint of matrix $\mathbf{P}$, and we can then straightforwardly optimize the covariance matrix Σ via gradient-based methods.

Notably, this transformation can be interpreted in terms of a sequence of invertible mappings. More explicitly, we extract a zero-mean Gaussian noise ξ_0 from the original noise $\mathbf{S}_0 \sim \mathcal{N}(\mathbf{1}_K, D(\alpha))$ in the form of $\mathbf{S}_0 = 1 + \xi_0$, and by successively transforming ξ_0 through a chain of T Householder reflections, we obtain the induced noise and the corresponding density at each step t as follows:

$$\mathbf{S}_t = 1 + \mathbf{H}_t\mathbf{H}_{t-1}...\mathbf{H}_1\xi_0 = 1 + \mathbf{U}\xi_0, \tag{3}$$

$$q_t(\mathbf{S}) = \mathcal{N}(\mathbf{1}_K, \mathbf{U}D(\alpha)\mathbf{U}^T). \tag{4}$$

By injecting the structured noise $\mathbf{S}_t$ into the deterministic weight Θ, we introduce a procedure referred to as *Variational Structured Dropout* (VSD). Let $\mathbf{W}_t = \mathbf{S}_t \odot \Theta$, we obtain a structured representation for the approximate posterior $q_t(.)$ over each column (j) of random weight $\mathbf{W}_t$:

$$q_t(\mathbf{W}_{(j)}) = \mathcal{N}(\Theta_{(j)}, \Theta_{(j)} \odot (\mathbf{U}D(\alpha)\mathbf{U}^T) \odot \Theta_{(j)}). \tag{5}$$

In the next analysis, the subscript j will be ignored for simplicity. With the above derivations, we use variational inference and optimize a variational lower bound given by:

$$\mathbb{E}_{q_t(\mathbf{W})} \log p(\mathcal{D}|\mathbf{W}) - \mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W})). \tag{6}$$

The expectation term of this objective function can be rewritten with the following form:

$$\mathbb{E}_{q_\phi(\mathbf{W})} \log p(\mathcal{D}|\mathbf{W}_t) - \mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W}))$$
$$= \mathbb{E}_{q_\alpha(\mathbf{S})} \log p(\mathcal{D}|\Theta, \mathbf{S}_t) - \mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W})). \tag{7}$$

Most importantly, to benefit VSD from the Bayesian statistics, we need to ensure $\mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W}))$ satisfies the KL condition. We solve this prerequisite by choosing an appropriate prior $p(\mathbf{W})$, such that this KL term does not depend on the Θ, based on the following result.

Proposition 1. *Assume the prior $p(\mathbf{W}|\tau) = \mathcal{N}(0, D(\frac{1}{\tau}))$ and we optimize the precision parameter τ of this distribution via the Empirical Bayes. Then, the KL term in the (7) has the form independent of the weight parameter Θ:*

$$\mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W})) = \frac{1}{2} \sum_{i=1}^K \log \frac{1 + \sum_{j=1}^K \alpha_j U_{ij}^2}{\alpha_i}. \tag{8}$$

The proof of Proposition 1 is in Appendix A. Note that, without Householder transformation, the orthogonal matrix $\mathbf{U}$ will be an identity matrix and the equation equation 8) is simplified as follows: $\mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W})) = 0.5 \sum_{i=1}^K \log(1 + \alpha_i^{-1})$. Therefore, our method generalizes the objective function of Variational Bayesian Dropout (VBD) method proposed in [25, 33].

We further improve the efficiency of Householder transformation in our method by utilizing the idea from [58, 3] where this transformation was deployed to approximate a full covariance Gaussian posterior on latent space of VAEs model [29]. In particular, they made the approximation more flexible by using fully connected layers between the Householder vectors. Conveniently, the low dimension of variational noise of VSD leads to a good adaptation, but with a little trade-off in computational complexity when increasing the number of tranformation steps T. We show the performance of VSD when using this neural approximate technique with $T \in \{1, 2, 3\}$ in Figure 1, where a larger T could potentially improve the results on predictive measures. However, to maintain computational efficiency, we recommend using $T = 1$ or 2 in large-scale experiments.
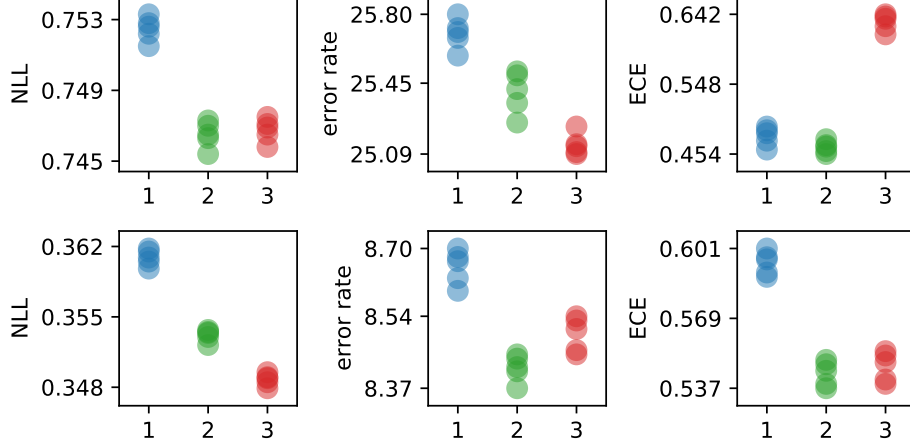
Figure 1: The performance of VSD when using the number of transformations $T \in \{1, 2, 3\}$. Evaluation over 5 runs on CIFAR10 (above) and SVHN (below) with LeNet architecture.

3.2 Joint inference with hierarchical prior

We further improve our proposal by introducing a prior hierarchy in VSD framework and then obtain a joint approximation for the Dropout posterior. This will facilitate to expand the scope of Bayesian modeling in our method in terms of both prior distribution and approximate posterior. We design a two-level hierarchical prior as follows:

$$p(\mathbf{W}, \mathbf{z}) = p(\mathbf{W}|\mathbf{z}, \beta)p(\mathbf{z}), \tag{9}$$

where $p(\mathbf{W}|\mathbf{z}, \beta) = \mathcal{N}(0, D(\frac{\mathbf{z}}{\beta}))$; the hyperprior $p(\mathbf{z})$ is a distribution with positive support such as Gamma or half-Cauchy distribution; the latent $\mathbf{z}$ has the size of the number of rows and is shared across columns of the weight matrix $\mathbf{W}$; the hyper-parameter β is treated as a scaling factor. This prior form has also been investigated in recent works [34, 14, 8]. We implement variational inference with a joint approximate posterior, also referred to as the joint Dropout posterior, which is parameterized as follows:

$$q_\phi(\mathbf{W}, \mathbf{z}) = q_\psi(\mathbf{z})q_\phi(\mathbf{W}|\mathbf{z}), \tag{10}$$

$$q_\phi(\mathbf{W}|\mathbf{z}) = \mathcal{N}(\mathbf{W}|\mathbf{z} \odot \mu, D(\mathbf{z}^2 \odot \sigma^2)) \tag{11}$$

with $\mu = \Theta, \sigma^2 = \alpha \odot \Theta^2$ and $q_\psi(\mathbf{z})$ is chosen depending on the family of prior $p(z)$ so that the reparametrization trick can be utilized. Sampling the random weight $\mathbf{W}$ from the joint variational posterior $q_\phi(\mathbf{W}, \mathbf{z})$ includes two steps:

$$\mathbf{z}^* \sim q_\psi(\mathbf{z}), \quad \mathbf{W}^* \sim q_\phi(\mathbf{W}|\mathbf{z}^*) \tag{12}$$

in which the second one can be reparameterized as follows:

$$\mathbf{W}^* = \mathbf{z}^* \odot (\mu + \sigma \odot \epsilon) = (\mathbf{z}^* \odot \mathbf{S}) \odot \Theta \tag{13}$$

6

with the noise $\mathbf{S} \sim \mathcal{N}(\mathbf{1}_K, D(\alpha))$. Similarly, we apply the Householder transformation to the variational noise $\mathbf{S}$ and obtain a new joint approximate posterior given by:

$$q_t(\mathbf{W}, \mathbf{z}) = q_\psi(\mathbf{z}) q_t(\mathbf{W}|\mathbf{z}) \tag{14}$$

$$q_t(\mathbf{W}|\mathbf{z}) = \mathcal{N}(\mathbf{W}|\mathbf{z} \odot \mu, \mathbf{V}\mathbf{U}D(\alpha)(\mathbf{V}\mathbf{U})^{(T)}) \tag{15}$$

with $\mathbf{V} = D(\mathbf{z} \odot \mu)$. Then, the variational lower bound of this joint inference problem is:

$$\mathcal{L}(\phi, \psi) := \mathbb{E}_{q_\psi(\mathbf{z})q_t(\mathbf{W}|\mathbf{z})} \log p(\mathcal{D}|\mathbf{W}) - \tag{16}$$
$$\mathbb{E}_{q_\psi(\mathbf{z})}(\mathbb{D}_{KL}(q_t(\mathbf{W}|\mathbf{z}, \phi)||p(\mathbf{W}|\mathbf{z}, \beta)) - \mathbb{D}_{KL}(q_\psi(\mathbf{z})||p(\mathbf{z})).$$

Under the particular variational posterior parametrization at equation (10), and a specific value of the scaling factor β, the KL-divergence between the structured conditional posterior $q_t(\mathbf{W}|\mathbf{z}, \phi)$ and the conditional prior $p(\mathbf{W}|\mathbf{z}, \beta)$ is independent of the model parameter Θ.

Proposition 2. *By using Empirical Bayes to specify the scaling factor β, we can make the first KL-term in the objective function $\mathcal{L}(\phi, \psi)$ satisfy the KL condition:*

$$\mathbb{D}_{KL}(q_t(\mathbf{W}|\mathbf{z}, \phi)||p(\mathbf{W}|\mathbf{z}, \beta)) =$$
$$\frac{1}{2} \sum_{i=1}^{K} \left(z_i - \log(z_i) - 1 + \log \frac{1 + \sum_{j=1}^{K} \alpha_j U_{ij}^2}{\alpha_i} \right). \tag{17}$$

The proof of Proposition 2 can be found in Appendix B. The derivation of prior hierarchy in our method is flexible without any simplifying assumptions about hyperprior $p(\mathbf{z})$. We can directly apply it for Variational Bayesian Dropout framework mentioned in the previous section. As the experimental results are shown in Figure 2, the hierarchical prior significantly improves the performance of both VBD and VSD on predictive metrics. Therefore, we aim to introduce Variational Structured Dropout with hierarchical prior as a unified framework of our proposal in this paper.

3.3 The expressiveness of VSD with hierarchical prior

In the following analysis, we derive some insights about the role of hierarchical prior in our framework.

Scale mixture of normals prior and mixture approximate posterior. The well-known property of expanding a model hierarchically is that it induces new dependencies between the data, either through shrinkage or an explicitly correlated prior [9]. The hierarchical representation in our method is a center parameterization, and by integrating out the latent variable $\mathbf{z}$, we obtain a marginal prior distribution as follows:

$$p(\mathbf{W}|\beta, \eta) = \int \mathcal{N}(\mathbf{W}|0, D(\frac{\mathbf{z}}{\beta})) p(\mathbf{z}|\eta) d\mathbf{z}. \tag{18}$$

By approximating this integral with Monte Carlo sampling $\mathbf{z}_i \sim p(\mathbf{z}|\eta)$, we can resemble an informative prior known as Gaussian scale mixtures [5, 7] by the following term: $p(\mathbf{W}|\beta, \eta) \approx \frac{1}{K} \sum_{i=1}^{K} \mathcal{N}(\mathbf{W}|0, D(\frac{\mathbf{z}_i}{\beta}))$. In the scope of this work, instead of investigating extensively the expressiveness of this prior through different parameterizations, we focus on the advantages of the joint inference that can make a mixture approximation in the variational objective function:
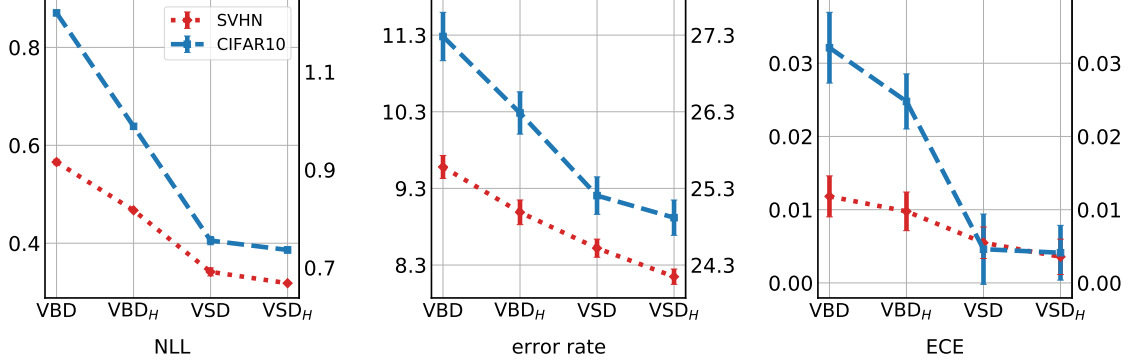
Figure 2: The performance of VBD and VSD when using the prior hierarchy, with the labels are VBD_H and VSD_H respectively.

$\mathbb{E}_{q_\psi(\mathbf{z})q_T(\mathbf{W}|\mathbf{z})} \log p(\mathcal{D}|\mathbf{W}) \approx \frac{1}{K}\sum_{i=1}^{K}\mathbb{E}_{q_T(\mathbf{W}|\mathbf{z}_i)} \log p(\mathcal{D}|\mathbf{W})$. This approximation is equivalent to a mixture of structured covariance Gaussian, which has a reasonable potential to recover the multimodality of the true Bayesian posterior. Moreover, this mixture is also practical in the high dimensional setting of deep Bayesian models, because it does not require the additional computational cost from multiplying the components.

Hierarchical prior imposes a global stochastic noise. At each iteration of the training process, while we draw $\mathbf{W}$ from the conditional posterior $q_t(\mathbf{W}|\mathbf{z})$ separately for each data as the Dropout procedure, the latent $\mathbf{z} \sim q_\psi(\mathbf{z})$ is shared across the entire data batch. This means for each data input $\mathbf{x}_n$, we introduce a joint-structured noise $\mathbf{S}_n^{(h)}$ with the following form:

$$\mathbf{z} \sim q_\psi(\mathbf{z}), \quad \mathbf{S}_n \sim \mathcal{N}(\mathbf{1}_K, \mathbf{U}D(\alpha)\mathbf{U}^{(T)}), \tag{19}$$
$$\mathbf{S}_n^{(h)} = \mathbf{z} \odot \mathbf{S}_n, \quad \mathbf{W} = \mathbf{S}_n^{(h)} \odot \Theta.$$

The above reinterpretation demonstrates that by the Monte Carlo estimation, the joint variational inference with hierarchical prior in our method adapts to the vanilla Dropout procedure. The new representation of Dropout noise allows our method to regularize each unit layer with different levels of stochasticity. The latent $\mathbf{z}$ under this representation can be considered as a global variational noise and by learning its variational distribution $q_\psi(\mathbf{z})$, we can capture correlation characteristics of input samples in each data batch.

3.4 Scalability of Variational Structured Dropout

Approximating a structured posterior directly on the random weights of deep convolution models is challenging. Besides expensive computation, it is difficult to employ the local reparameterization trick (LTR) [28], leading to high variance in training. We apply variational structured Dropout for convolutional layer by learning a structured noise with the size of the number of kernels and imposing it to convolutional weights:

$$\mathbf{S} \sim \mathcal{N}(\mathbf{1}_K, \mathbf{U}D(\alpha)\mathbf{U}^T), \quad \mathbf{W}_{ijk} = \Theta_{ijk} \odot \mathbf{S}_k, \tag{20}$$

with i, j, k are the indexes representing height, width, and filter respectively. This simple solution greatly reduces computational complexity while being able to captures the dependencies among kernels of the convolutional layer. Moreover, to perform backpropagation in parallel across each

Table 1: Computational complexity per layer of different Bayesian inference methods, where K and L are respectively the number of in and out features, $|\mathcal{B}|$ is batchsize.

Method	Time	Memory
MAP	$\mathcal{O}(KL|\mathcal{B}|)$	$\mathcal{O}(L|\mathcal{B}|)$
BBB	$\mathcal{O}(sKL|\mathcal{B}|)$	$\mathcal{O}(sKL + L|\mathcal{B}|)$
BBB-LTR	$\mathcal{O}(2KL|\mathcal{B}|)$	$\mathcal{O}(2L|\mathcal{B}|)$
VD	$\mathcal{O}(KL|\mathcal{B}|)$	$\mathcal{O}(K|\mathcal{B}|)$
VMG	$\mathcal{O}(m^3 + 2KL|\mathcal{B}|)$	$\mathcal{O}(KL|\mathcal{B}|)$
SLANG	$\mathcal{O}(r^2KL + rsKL|\mathcal{B}|)$	$\mathcal{O}(rKL + sKL|\mathcal{B}|)$
ELRG	$\mathcal{O}(r^3 + (r+2)KL|\mathcal{B}|)$	$\mathcal{O}((r+2)L|\mathcal{B}|)$
VSD	$\mathcal{O}(K^2 + KL|\mathcal{B}|)$	$\mathcal{O}(K^2 + K|\mathcal{B}|)$
VSD-low rank	$\mathcal{O}(rK + KL|\mathcal{B}|)$	$\mathcal{O}(K^2 + K|\mathcal{B}|)$

Table 2: Computation time of Bayesian inference methods compared to standard MAP (1x). Training with CIFAR10 dataset on different architectures. Results are averaged over 10 first epochs.

Methods	Time/epochs (s)		
	LeNet5	AlexNet	ResNet18
VD	1.18x	1.15x	1.32x
BBB-LTR	1.53x	1.75x	4.28x
VOGN	2.63x	3.25x	4.44x
VSD $T = 1$	1.25x	1.32x	1.86x
VSD $T = 2$	1.35x	1.49x	2.90x

minibatch, we just duplicate the structured noise and inject it into the current input similar to the Dropout procedure on the fully connected layer.

Computational complexity.

We present the complexity of Bayesian inference methods in terms of computational cost and memory storage in Table 1 (The abbreviations for these methods are shown in Section 4). These Bayesian methods use Monte Carlo sampling combined with the reparameterization trick to approximate the intractable log-likelihood. Our method adopts the advantage of Dropout training by just sampling the low dimension noise instead of whole random weights like BBB, SLANG. This is similar to the effect of the local reparameterization trick and maintains our memory cost at a low level of $\mathcal{O}(K^2 + K|\mathcal{B}|)$. Without the local reparameterization trick, the methods such as vanilla BBB, SLANG also need many Monte Carlo samples for the random weights to reduce the variance in gradient estimator. It leads to loss of parallelism when backpropagating, and incurs extra costs on both time and memory. Moreover, we investigate low-rank structure in VSD by using a low dimensional hidden unit with a ReLU activation in the neural approximation of Householder vectors. We then get a computational time of $\mathcal{O}(rK + KL|\mathcal{B}|)$ without sacrificing much the performance. See Appendix C for more detailed analyses of Table 1.

We give in Table 2 the computation time of VSD and other methods, in which VOGN [50] is one of the latest practical Bayesian method. VSD shows running time even more effective than the mean-field method (BBB-LTR). Although there is a trade-off when using a larger number of T, VSD does not extra computation time much compared to VD.

Table 3: Average test performance for regression task on UCI datasets. Results are reported with RMSE and Std. Errors.

Dataset	BBB	VMG	SLANG	MCD	VD	VBD	VSD
Boston	3.43 ± 0.20	2.70 ± 0.13	3.21 ± 0.19	2.83 ± 0.17	2.98 ± 0.18	2.94 ± 0.18	**2.64 ± 0.17**
Concrete	6.16 ± 0.13	4.89 ± 0.12	5.58 ± 0.19	4.93 ± 0.14	5.16 ± 0.13	5.14 ± 0.12	**4.72 ± 0.11**
Energy	0.97 ± 0.09	0.54 ± 0.02	0.64 ± 0.03	1.08 ± 0.03	0.64 ± 0.02	0.63 ± 0.02	**0.47 ± 0.01**
Kin8nm	0.08 ± 0.00	0.08 ± 0.00	0.08 ± 0.00	0.09 ± 0.00	0.08 ± 0.00	0.08 ± 0.00	**0.08 ± 0.00**
Naval	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	**0.00 ± 0.00**
Power Plant	4.21 ± 0.03	4.04 ± 0.04	4.16 ± 0.04	4.00 ± 0.04	3.99 ± 0.03	3.98 ± 0.04	**3.92 ± 0.04**
Wine	0.64 ± 0.01	0.63 ± 0.01	0.65 ± 0.01	**0.61 ± 0.01**	0.62 ± 0.01	0.62 ± 0.01	0.63 ± 0.01
Yacht	1.13 ± 0.06	0.71 ± 0.05	1.08 ± 0.06	0.72 ± 0.05	1.09 ± 0.09	1.09 ± 0.09	**0.69 ± 0.06**

4 Experiments

In this section, we provide experimental evaluations to show the effectiveness of our proposed methods compared with the existing methods in terms of both predictability and scalability. We focus mainly on variational inference methods of the following two directions: the first one is via approximating the posterior directly on the random weights of Bayesian nets, including Bayes by Backprop (BBB) [5], Variational Matrix Gaussian (VMG) [35], low-rank approximations (SLANG, ELRG) [42, 59]; and the second one is the Bayesian Dropout methods with MC Dropout (MCD) [12, 13], Variational Dropout (VD) [28], Variational Bayesian Dropout (VBD) [25, 33] and our method-Variational Structured Dropout (VSD). In addition, we evaluate the performance of point estimate framework as MAP in most of our experiments. Details about data descriptions, network architectures, hyper-parameter tuning are presented in Appendix H.

4.1 Regression with UCI datasets

We start with implementing a standard experiment for Bayesian regression task on UCI dataset [2] proposed in [19]. We follow the original setup used in [13]. Detailed descriptions of the data and experimental setting can be found in Appendix H.3. We present the performance of methods based on standard metrics including root mean square error (RMSE) in Table 3 and predictive log-likelihood (LL) in Table 9 of Appendix F.1. As shown in the tables, VSD performs better than baselines on most datasets in terms of both criteria (5/8 tasks on RMSE and 7/8 tasks on predictive LL). VSD also achieves better results with a significant margin compared with VD and VBD, specifically on *Boston, Concrete, Energy, Yacht* datasets. This demonstrates the effectiveness of learning a structured representation for multiplicative Gaussian noise instead of using a diagonal distribution as in VD and VBD.

4.2 Image classification

We now compare the predictive performance of the aforementioned methods for classification tasks on three standard image datasets: MNIST [32], CIFAR10 [30], and SVHN [48]. We evaluate the predictive probabilities using the metrics including negative log-likelihood (NLL), error rate, and expected calibration error (ECE) [46, 16], in which, the error rate represents explicitly for the predictive accuracy, while NLL and ECE capture how well a model estimates the predictive uncertainty. Details on experimental settings are available in Appendix H.4.

Table 4: Results for VSD and baselines on vectorized MNIST, CIFAR10 and SVHN. Symbol * refers to the results inherited from the original paper. Results are averaged over 5 random seeds. For all metrics, lower is better. ECE is measured in percentage. On MNIST, MVG achieves err. rates of 1.17% and 1.27% with FC 400x2 and FC 750x3, respectively. SLANG reports 1.72% err. rate with FC 400x2.

| Method | MNIST | | | | | | CIFAR10 | | | SVHN | | |
| | FC 400x2 | | | FC 750x3 | | | CNN 32x64x128 | | | | | |
	NLL	err. rate	ECE	NLL	err. rate	ECE	NLL	err. rate	ECE	NLL	err. rate	ECE
MAP	0.098	1.32	0.234	0.109	1.27	0.224	2.8466	34.04	5.541	0.8549	12.26	1.787
BBB	0.109	1.59	0.235	0.140	1.50	0.235	1.2022	30.11	1.939	0.5449	10.57	0.413
ELRG	0.053*	1.54*	-	-	-	-	0.8711*	29.43*	-	-	-	-
MCD	0.049	1.26	0.129	0.057	1.22	0.150	0.7936	26.91	0.490	0.3649	9.23	0.370
VD	0.051	1.21	0.125	0.061	1.17	0.146	1.1756	27.45	3.118	0.5336	9.47	1.152
VBD	0.054	1.22	0.117	0.063	1.26	0.157	1.2197	27.42	3.215	0.5622	9.54	1.181
VSD	**0.044**	**1.11**	0.118	**0.045**	**1.13**	**0.102**	**0.7497**	**25.28**	**0.461**	**0.3121**	**8.44**	**0.214**
VSD-H	**0.042**	**1.08**	**0.112**	**0.048**	**1.09**	**0.112**	**0.7301**	**24.92**	**0.414**	**0.2994**	**8.39**	**0.110**

Table 5: Image classification using AlexNet architecture. Results are averaged over 5 random seeds. For NLL and ECE metric, a lower number is better, while for ACC-predictive accuracy, a higher number is better.

| AlexNet | CIFAR10 | | | CIFAR100 | | | SVHN | | | STL10 | | |
	NLL	ACC	ECE	NLL	ACC	ECE	NLL	ACC	ECE	NLL	ACC	ECE
MAP	0.895	77.89	0.126	3.808	46.60	0.326	0.293	**91.70**	0.019	3.005	65.30	0.138
BBB	0.994	65.38	0.062	2.659	32.41	0.049	0.476	87.30	0.094	1.707	65.46	0.222
MCD	0.717	75.22	0.023	**2.196**	43.12	0.022	0.361	89.70	0.047	1.059	63.65	0.052
VD	0.702	77.28	0.026	2.582	43.10	0.009	0.327	90.76	0.002	2.130	65.48	0.040
ELRG $K=4$	0.678	76.43	0.013	2.260	42.41	0.038	0.342	89.63	0.019	1.181	56.10	0.020
VSD-Ours	**0.656**	**78.21**	**0.010**	2.241	**46.85**	**0.003**	**0.290**	91.62	**0.001**	**1.019**	**67.98**	**0.015**

The synthesis results of this experiment are in Table 4. By a general observation, VSDs outperform other methods in most settings. More specifically, VSD with hierarchical prior (VSD-H) consistently obtains the best results on all three metrics. While the lowest error rate implies the highest predictive accuracy, the figures on NLL and ECE represent well-calibrated predictions in our model. On the other hand, for the remaining methods, especially MAP and BBB, the error rates are worse by a large margin compared to VSD (respectively about 9% and 5% on CIFAR10, 4% and 2% on SVHN). This indicates that these methods do not generalize well for unseen data. Moreover, on CIFAR10 and SVHN, these two methods and VD, VBD show an inefficiency on the measures NLL and ECE. It shows that the confidence of the predictions in these models is uncalibrated.

For MC Dropout, we observe a pretty good performance with the second-best result in most settings that is similar to those reported of other works [42, 50, 59]. Note that these results of the Bayesian Dropout methods are competitive with structured methods such as VMG, SLANG, ELRG. This has reinforced our motivation about the potential of Dropout methods for improving predictive performance.

4.3 Scaling up Bayesian deep convolutional networks

We conduct additional experiments to integrate VSD into large-scale convolutional networks. We reproduce the experiments proposed in [59] (ELRG), in which, we trained AlexNet and ResNet18 on
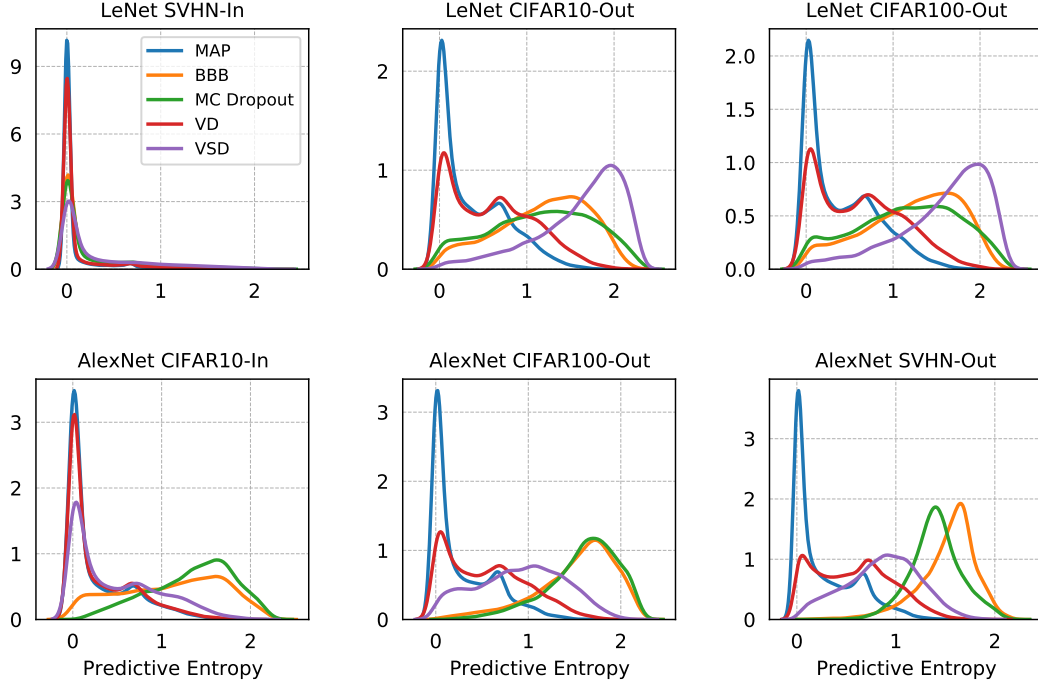
Figure 3: Histograms of predictive entropy for LeNet (top) and AlexNet (bottom) trained on SVHN and CIFAR10 respectively.

4 datasets CIFAR10, SVHN, CIFAR100 [30], and STIL10 [6] to evaluate the predictive performance of our proposal compared to other methods

The final results of AlexNet are presented in Table 5, and those of ResNet18 are left in Table 10 in Appendix F.2. For AlexNet, the performance of VSD is more consistent and better than other methods. Although MAP has the predictive accuracy being competitive with our method, it comes at a trade-off with the worst results on both NLL and ECE in most settings. Modern deeper architectures facilitate deterministic estimates like MAP to better learn discriminative information extracted from training data but also makes its predictions more confident when picking excessively on unique optima. Meanwhile, ELRG with a low-rank structure on the variational posterior has gained desirable properties on uncertainty metrics. Its performance on NLL and ECE are asymptotic to that of VSD, however, our method obtains significant improvements on accuracy metric. For the remaining methods, BBB performs poorly on CIFAR10 and CIFAR100 with bad accuracy, but it still has better performance than MAP in terms of NLL and ECE. Finally, VD and MCD both underperform VSD on almost all metrics.

For ResNet18 architecture, while MAP and BBB still exhibit the same behavior as mentioned above, VSD continues to achieve the convincing results, namely, it has the best performance on CIFAR100 and SVHN over all three metrics. On CIFAR10 and STL10, VSD obtains the best results on ECE metric, and gets competitive statistics on NLL and accuracy. Overall, compared with MCD, VD, and ELRG, our method maintains good performance with better stability.

4.4 Predictive entropy performance

We now evaluate the predictive uncertainty of each model on the out-of-distribution settings that have been implemented in previous works [31, 36, 50, 59]. We expect the predictions of models

to exhibit high confidence (low uncertainty) when the test data has the same distribution to the training data, and in an opposite case, the model should show higher uncertainty. We evaluate the entropy of the predictive distribution $p(y^*|x^*, \mathcal{D})$ and use the density of this entropy to assess the quality of the uncertainty estimates. We conduct three scenarios with different datasets (SVHN, CIFAR10, and CIFAR100) and network architectures (LeNet, AlexNet, and ResNet18).

For LeNet, we train the model on SVHN dataset and then consider out-of-distribution data from CIFAR10 and CIFAR100. The results are shown in the top row of Figure 3. Methods all work well on in-distribution data SVHN with the entropy value being distributed mostly around zero. However, the entropy densities of MAP and VD are concentrated excessively. This indicates that these methods tend to make overconfident predictions on out-of-distribution data. This claim is further supported by the results on CIFAR10 and CIFAR100 datasets. In contrast, MCD, BBB, and VSD are well-calibrated with a moderate level of confidence for in-distribution data. On CIFAR10 and CIFAR100 datasets, VSD gains better results with entropy values being distributed over a larger support, meaning that the predictions of VSD are closer to uniform on unseen classes.

We run a similar experiment, in which, we train AlexNet on CIFAR10 and use SVHN, CIFAR100 as out-of-distribution data. The results are shown in the bottom row of Figure 3. While MAP and VD still exhibit the same overconfident phenomenon as on LeNet, we observe the underconfident predictions of BBB and MC Dropout even on in-distribution data, and of course, leading to a high uncertainty on out-of-distribution data. We hypothesize that this is because the models trained with these methods are underfit with low accuracy on the in-distribution training data. In contrast, VSD estimates reasonably the predictive entropy in both settings. The remaining scenario with ResNet18 trained on CIFAR100 is left in the Figure 6 of Appendix F.2.

5   Conclusions

We proposed a new approximate inference framework for deep Bayesian nets, named Variational Structured Dropout (VSD). The novelty of VSD is that we learn a structured approximate posterior via the Dropout principle. Instead of performing complex and expensive techniques directly on the random weights of deep Bayesian models, we approximate structured representation for low-dimensional noise on deterministic neural nets. VSD is successful in acquiring a flexible inference while maintaining computational efficiency and scalability for deep convolutional models. The extensive experiments have evidenced the advantages of VSD such as well-calibrated prediction, better generalization, good uncertainty estimation. Given a consistent performance of VSD as presented throughout the paper, an extension of that method to other problems, such as Bayesian active learning or reinforcement learning, is of interest.

References

[1] O. Arenz, M. Zhong, and G. Neumann. Trust-region variational inference with gaussian mixture models. *Journal of Machine Learning Research*, 21(163):1–60, 2020. (Cited on page 4.)

[2] A. Asuncion and D. Newman. Uci machine learning repository, 2007. (Cited on page 10.)

[3] R. v. d. Berg, L. Hasenclever, J. M. Tomczak, and M. Welling. Sylvester normalizing flows for variational inference. *arXiv preprint arXiv:1803.05649*, 2018. (Cited on pages 4 and 5.)

[4] C. H. Bischof and X. Sun. On orthogonal block elimination. *Preprint MCS-P450-0794, Mathematics and Computer Science Division, Argonne National Laboratory*, 1994. (Cited on page 4.)

[5] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015. (Cited on pages 1, 7, 10, and 21.)

[6] A. Coates, A. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011. (Cited on page 12.)

[7] T. Cui, A. Havulinna, P. Marttinen, and S. Kaski. Informative gaussian scale mixture priors for bayesian neural networks. *arXiv preprint arXiv:2002.10243*, 2020. (Cited on pages 7 and 20.)

[8] M. W. Dusenberry, G. Jerfel, Y. Wen, Y.-a. Ma, J. Snoek, K. Heller, B. Lakshminarayanan, and D. Tran. Efficient and scalable bayesian neural nets with rank-1 factors. *arXiv preprint arXiv:2005.07186*, 2020. (Cited on pages 2 and 6.)

[9] B. Efron. *Large-scale inference: empirical Bayes methods for estimation, testing, and prediction*, volume 1. Cambridge University Press, 2012. (Cited on page 7.)

[10] A. Y. Foong, D. R. Burt, Y. Li, and R. E. Turner. On the expressiveness of approximate inference in bayesian neural networks. *arXiv preprint arXiv:1909.00719*, 2019. (Cited on page 2.)

[11] Y. Gal. Uncertainty in deep learning. *University of Cambridge*, 1(3), 2016. (Cited on page 3.)

[12] Y. Gal and Z. Ghahramani. Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv preprint arXiv:1506.02158*, 2015. (Cited on pages 2, 10, and 21.)

[13] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016. (Cited on pages 2, 3, 10, 21, and 29.)

[14] S. Ghosh, J. Yao, and F. Doshi-Velez. Structured variational learning of bayesian neural networks with horseshoe priors. *arXiv preprint arXiv:1806.05975*, 2018. (Cited on pages 2 and 6.)

[15] A. Graves. Practical variational inference for neural networks. In *Advances in neural information processing systems*, pages 2348–2356, 2011. (Cited on page 1.)

[16] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017. (Cited on pages 10 and 24.)

[17] A. K. Gupta and D. K. Nagar. *Matrix variate distributions*, volume 104. CRC Press, 2018. (Cited on page 2.)

[18] D. P. Helmbold and P. M. Long. On the inductive bias of dropout. *The Journal of Machine Learning Research*, 16(1):3403–3454, 2015. (Cited on page 2.)

[19] J. M. Hernández-Lobato and R. Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869, 2015. (Cited on pages 10 and 27.)

[20] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012. (Cited on page 3.)

[21] G. E. Hinton and D. 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. (Cited on page 3.)

[22] J. Hron, A. G. d. G. Matthews, and Z. Ghahramani. Variational bayesian dropout: pitfalls and fixes. *arXiv preprint arXiv:1807.01969*, 2018. (Cited on pages 2, 4, 28, and 29.)

[23] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999. (Cited on page 3.)

[24] M. E. Khan, D. Nielsen, V. Tangkaratt, W. Lin, Y. Gal, and A. Srivastava. Fast and scalable bayesian deep learning by weight-perturbation in adam. *arXiv preprint arXiv:1806.04854*, 2018. (Cited on page 2.)

[25] V. Kharitonov, D. Molchanov, and D. Vetrov. Variational dropout via empirical bayes. *arXiv preprint arXiv:1811.00596*, 2018. (Cited on pages 5, 10, and 21.)

[26] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. (Cited on page 29.)

[27] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improving variational inference with inverse autoregressive flow. *arXiv preprint arXiv:1606.04934*, 2016. (Cited on page 4.)

[28] D. P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. In *Advances in neural information processing systems*, pages 2575–2583, 2015. (Cited on pages 1, 2, 3, 8, 10, 21, 28, and 29.)

[29] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. (Cited on pages 3 and 5.)

[30] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009. (Cited on pages 10 and 12.)

[31] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *arXiv preprint arXiv:1612.01474*, 2016. (Cited on page 12.)

[32] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. (Cited on page 10.)

[33] Y. Liu, W. Dong, L. Zhang, D. Gong, and Q. Shi. Variational bayesian dropout with a hierarchical prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7124–7133, 2019. (Cited on pages 2, 5, 10, and 21.)

[34] C. Louizos, K. Ullrich, and M. Welling. Bayesian compression for deep learning. In *Advances in neural information processing systems*, pages 3288–3298, 2017. (Cited on page 6.)

[35] C. Louizos and M. Welling. Structured and efficient variational deep learning with matrix gaussian posteriors. In *International Conference on Machine Learning*, pages 1708–1716, 2016. (Cited on pages 2, 10, and 21.)

[36] C. Louizos and M. Welling. Multiplicative normalizing flows for variational bayesian neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2218–2227. JMLR. org, 2017. (Cited on pages 2, 12, and 23.)

[37] D. J. MacKay. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992. (Cited on page 1.)

[38] S.-i. Maeda. A bayesian encourages dropout. *arXiv preprint arXiv:1412.7003*, 2014. (Cited on page 2.)

[39] D. McAllester. A pac-bayesian tutorial with a dropout bound. *arXiv preprint arXiv:1307.2118*, 2013. (Cited on page 2.)

[40] P. Mianjy and R. Arora. On convergence and generalization of dropout training. *Advances in Neural Information Processing Systems*, 33, 2020. (Cited on page 2.)

[41] P. Mianjy, R. Arora, and R. Vidal. On the implicit bias of dropout. In *International Conference on Machine Learning*, pages 3537–3545, 2018. (Cited on page 2.)

[42] A. Mishkin, F. Kunstner, D. Nielsen, M. Schmidt, and M. E. Khan. Slang: Fast structured covariance approximations for bayesian deep learning with natural gradient. In *Advances in Neural Information Processing Systems*, pages 6245–6255, 2018. (Cited on pages 10, 11, and 21.)

[43] D. Molchanov, A. Ashukha, and D. Vetrov. Variational dropout sparsifies deep neural networks. *arXiv preprint arXiv:1701.05369*, 2017. (Cited on pages 2, 28, and 29.)

[44] W. Mou, Y. Zhou, J. Gao, and L. Wang. Dropout training, data-dependent regularization, and generalization bounds. In *International Conference on Machine Learning*, pages 3645–3653, 2018. (Cited on page 2.)

[45] J. Mukhoti, P. Stenetorp, and Y. Gal. On the importance of strong baselines in bayesian deep learning. *arXiv preprint arXiv:1811.09385*, 2018. (Cited on page 29.)

[46] M. P. Naeini, G. Cooper, and M. Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015. (Cited on page 10.)

[47] R. M. Neal. *BAYESIAN LEARNING FOR NEURAL NETWORKS*. PhD thesis, University of Toronto, 1995. (Cited on page 1.)

[48] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. 2011. (Cited on page 10.)

[49] V. M.-H. Ong, D. J. Nott, and M. S. Smith. Gaussian variational approximation with a factor covariance structure. *Journal of Computational and Graphical Statistics*, 27(3):465–478, 2018. (Cited on page 2.)

[50] K. Osawa, S. Swaroop, A. Jain, R. Eschenhagen, R. E. Turner, R. Yokota, and M. E. Khan. Practical deep learning with bayesian principles. *arXiv preprint arXiv:1906.02506*, 2019. (Cited on pages 9, 11, and 12.)

[51] R. Ranganath, D. Tran, and D. Blei. Hierarchical variational models. In *International Conference on Machine Learning*, pages 324–333, 2016. (Cited on pages 2 and 4.)

[52] D. Rezende and S. Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning*, pages 1530–1538. PMLR, 2015. (Cited on page 4.)

[53] H. Ritter, A. Botev, and D. Barber. Online structured laplace approximations for overcoming catastrophic forgetting. In *Advances in Neural Information Processing Systems*, pages 3738–3748, 2018. (Cited on page 2.)

[54] H. Ritter, A. Botev, and D. 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. (Cited on page 2.)

[55] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014. (Cited on pages 2 and 3.)

[56] X. Sun and C. Bischof. A basis-kernel representation of orthogonal matrices. *SIAM journal on matrix analysis and applications*, 16(4):1184–1196, 1995. (Cited on page 4.)

[57] J. Swiatkowski, K. Roth, B. S. Veeling, L. Tran, J. V. Dillon, S. Mandt, J. Snoek, T. Salimans, R. Jenatton, and S. Nowozin. The k-tied normal distribution: A compact parameterization of gaussian mean field posteriors in bayesian neural networks. *arXiv preprint arXiv:2002.02655*, 2020. (Cited on page 2.)

[58] J. M. Tomczak and M. Welling. Improving variational auto-encoders using convex combination linear inverse autoregressive flow. *arXiv preprint arXiv:1706.02326*, 2017. (Cited on page 5.)

[59] M. Tomczak, S. Swaroop, and R. Turner. Efficient low rank gaussian variational inference for neural networks. *Advances in Neural Information Processing Systems*, 33, 2020. (Cited on pages 2, 10, 11, 12, 21, and 30.)

[60] S. Wager, S. Wang, and P. S. Liang. Dropout training as adaptive regularization. In *Advances in Neural Information Processing Systems*, pages 351–359, 2013. (Cited on page 2.)

[61] S. Wang and C. Manning. Fast dropout training. In *international conference on machine learning*, pages 118–126. PMLR, 2013. (Cited on page 27.)

[62] C. Wei, S. Kakade, and T. Ma. The implicit and explicit regularization effects of dropout. In *International Conference on Machine Learning*, pages 10181–10192. PMLR, 2020. (Cited on page 2.)

[63] M. Yin and M. Zhou. Semi-implicit variational inference. *arXiv preprint arXiv:1805.11183*, 2018. (Cited on page 4.)

[64] G. Zhang, S. Sun, D. Duvenaud, and R. Grosse. Noisy natural gradient as variational inference. In *International Conference on Machine Learning*, pages 5852–5861. PMLR, 2018. (Cited on page 2.)

[65] O. Zobay et al. Variational bayesian inference with gaussian-mixture approximations. *Electronic Journal of Statistics*, 8(1):355–389, 2014. (Cited on page 4.)

Supplement to "Improving Bayesian Inference in Deep Neural Networks with Variational Structured Dropout"

In this supplementary material, we collect proofs and remaining materials that were deferred from the main paper. In Appendix A, we provide proof for Proposition 1. In Appendix B, we present proof for Proposition 2. Details of computational complexity of different Bayesian methods are in Appendix C. The remaining Appendices are for the additional experiments with Variational Structured Dropout (VSD).

A The KL condition in Variational Structured Dropout

The variational lower bound of Variational Structured Dropout (VSD) is given as follows:

$$\mathbb{E}_{q_\phi(\mathbf{W})} \log p(\mathcal{D}|\mathbf{W}_t) - \mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W})) = \mathbb{E}_{q_\alpha(\mathbf{S})} \log p(\mathcal{D}|\Theta, \mathbf{S}_t) - \mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W})). \quad (21)$$

Applying Monte Carlo sampling combined with the reparameterization trick to approximate the expected log-likelihood, we perform a procedure being equivalent to injecting a structured noise $\mathbf{S}_t$ into the model parameters Θ. Therefore, to ensure the KL condition, we will specify the KL term in the form independent of Θ. Then, the above lower bound recovers the Dropout objective function.

We have the Dropout posterior $q_t(\mathbf{W}) = \mathcal{N}(\Theta, \Theta \odot (\mathbf{U}D(\alpha)\mathbf{U}^T) \odot \Theta)$ and the prior $p(\mathbf{W}|\beta) = \mathcal{N}(0, D(\frac{1}{\beta}))$. Let $\boldsymbol{\mu}_1 = \Theta$, $\boldsymbol{\Sigma}_1 = \Theta \odot (\mathbf{U}D(\alpha)\mathbf{U}^T) \odot \Theta$ and $\boldsymbol{\mu}_2 = 0$, $\boldsymbol{\Sigma}_2 = D(\frac{1}{\beta})$. The KL divergence can be calculated as follows:

$$\mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W})) = \frac{1}{2}\left[\log\frac{|\boldsymbol{\Sigma}_2|}{|\boldsymbol{\Sigma}_1|} - K + \text{Trace}(\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1) + (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)^T\boldsymbol{\Sigma}_2^{-1}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)\right]. \quad (22)$$

Since $\mathbf{U}$ is a orthogonal matrix, we have:

$$\log\frac{|\boldsymbol{\Sigma}_2|}{|\boldsymbol{\Sigma}_1|} = -\sum_{i=1}^{K}\log\beta_i - \sum_{i=1}^{K}\log\alpha_i\Theta_i^2, \quad (23)$$

$$\text{Trace}(\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1) = \text{Trace}(D(\beta \odot \Theta^2)\mathbf{U}D(\alpha)\mathbf{U}^T) = \sum_{i=1}^{K}\beta_i\Theta_i^2(\sum_{j=1}^{K}\alpha_j U_{ij}^2), \quad (24)$$

$$(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)^T\boldsymbol{\Sigma}_2^{-1}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1) = \Theta^T D(\beta)\Theta = \sum_{i=1}^{K}\beta_i\Theta_i^2. \quad (25)$$

Given the above equations, the KL term can be rewritten in the following form:

$$\mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W})) = \frac{1}{2}\sum_{i=1}^{K}\left[-\log\beta_i - \log\alpha_i\Theta_i^2 - 1 + \beta_i\Theta_i^2\left(1 + \sum_{j=1}^{K}\alpha_j U_{ij}^2\right)\right]. \quad (26)$$

We can find that the orthogonality of Householder transformations facilitates the tractable calculation for the KL term without complicated analyses. We now choose the prior hyper-parameter β via the Empirical Bayes. This procedure is achieved by optimizing β based upon the data. More specifically, taking the partial derivative of the RHS in equation (26) with respect to β, we obtain

$$\frac{\partial\mathbb{D}_{KL}}{\partial\beta_i} = \frac{1}{2}\left[-\frac{1}{\beta_i} + \Theta_i^2\left(1 + \sum_{j=1}^{K}\alpha_j U_{ij}^2\right)\right]. \quad (27)$$

Letting this derivative to zero, we obtain the optimal value for β in the analytical form: $\beta_i = 1/\left(\Theta_i^2(1 + \sum_{j=1}^{K}\alpha_j U_{ij}^2)\right)$. Substitute this value in the expression of KL term, we get the form independent of the weight parameter Θ as follows:

$$\mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W})) = \frac{1}{2}\sum_{i=1}^{K}\log\frac{1+\sum_{j=1}^{K}\alpha_j U_{ij}^2}{\alpha_i}. \tag{28}$$

As a consequence, we obtain the conclusion of Proposition 1.

B The derivation of the variational objective with hierarchical prior

In Bayesian inference, the prior distribution plays an important role in uncertainty quantification. This distribution also facilitates incorporating external domain knowledge or specific properties (such as feature sparsity, signal-to-noise ratio) into the Bayesian models [7].

However, in Dropout approximate inference, the KL condition restricts the scope of prior distribution family. Our works has overcome this limitation by proposing a unified framework using varational structured Dropout combined with hierarchical prior, in which we guarantees the KL condition without any simplifying assumptions about the prior family. Concretely, with our proposed prior hierachy, we maximize a variational lower bound from joint variational inference as follows:

$$\mathbb{E}_{q_\psi(\mathbf{z})q_t(\mathbf{W}|\mathbf{z})}\log p(\mathcal{D}|\mathbf{W}) - \mathbb{E}_{q_\psi(\mathbf{z})}(\mathbb{D}_{KL}(q_t(\mathbf{W}|\mathbf{z},\phi)||p(\mathbf{W}|\mathbf{z},\beta)) - \mathbb{D}_{KL}(q_\psi(\mathbf{z})||p(\mathbf{z})). \tag{29}$$

For the latent variable $\mathbf{z}$, we choose the prior $p(\mathbf{z}|\eta)$ and the variational distribtuion $q(\mathbf{z})$ as (inverse) Gamma(a,b) and log-Normal(γ,δ) distribution respectively. These distributions have positive support and can be reparametrized. The KL-divergence between them also has a closed-form expression, which is given by:

$$\mathbb{D}_{KL}(q_\psi(\mathbf{z})||p_\eta(\mathbf{z})) = a\log b - \log\Gamma(a) - a\gamma - \beta\exp(-\gamma + 0.5\delta) + 0.5(\log\delta + 1 + \log(2\pi)). \tag{30}$$

For the KL-divergence between the structured conditional posterior $q_t(\mathbf{W}|\mathbf{z},\phi)$ and the conditional prior $p(\mathbf{W}|\mathbf{z},\beta)$, similarly, we need a form that does not depend on Θ. Since $q_t(\mathbf{W}|\mathbf{z}) = \mathcal{N}(\mathbf{W}|\mathbf{z}\odot\Theta, \mathbf{V}\mathbf{U}D(\alpha)(\mathbf{V}\mathbf{U})^{(T)})$, with $\mathbf{V} = D(\mathbf{z}\odot\Theta)$ and $p(\mathbf{W}|\mathbf{z},\beta) = \mathcal{N}(0, D(\frac{\mathbf{z}}{\beta}))$, similar to the analysis in the previous section, we have:

$$\mathbb{D}_{KL}(q_t(\mathbf{W}|\mathbf{z},\phi)||p(\mathbf{W}|\mathbf{z},\beta)) = \frac{1}{2}\sum_{i=1}^{K}\left[-\log z_i - 1 - \log\beta_i - \log\alpha_i\Theta_i^2 + \beta_i z_i\Theta_i^2\left(1 + \sum_{j=1}^{K}\alpha_j U_{ij}^2\right)\right]. \tag{31}$$

Because β is referred to as the scaling factor, we can choose it by: $\beta_i = 1/\left(\Theta_i^2(1+\sum_{j=1}^{K}\alpha_j U_{ij}^2)\right)$. The above KL then can be rewritten in the following form:

$$\mathbb{D}_{KL}(q_t(\mathbf{W}|\mathbf{z},\phi)||p(\mathbf{W}|\mathbf{z},\beta)) = \frac{1}{2}\sum_{i=1}^{K}\left[z_i - \log z_i - 1 - \log\frac{1+\sum_{j=1}^{K}\alpha_j U_{ij}^2}{\alpha_i}\right]. \tag{32}$$

As a consequence, we obtain the conclusion of Proposition 2.

Table 6: Computational complexity per layer of some Bayesian inference methods.

Method	Time	Memory
MAP	$\mathcal{O}(KL\|\mathcal{B}\|)$	$\mathcal{O}(L\|\mathcal{B}\|)$
BBB	$\mathcal{O}(sKL\|\mathcal{B}\|)$	$\mathcal{O}(sKL + L\|\mathcal{B}\|)$
BBB-LTR	$\mathcal{O}(2KL\|\mathcal{B}\|)$	$\mathcal{O}(2L\|\mathcal{B}\|)$
VD	$\mathcal{O}(KL\|\mathcal{B}\|)$	$\mathcal{O}(K\|\mathcal{B}\|)$
VMG	$\mathcal{O}(m^3 + 2KL\|\mathcal{B}\|)$	$\mathcal{O}(KL\|\mathcal{B}\|)$
SLANG	$\mathcal{O}(r^2KL + rsKL\|\mathcal{B}\|)$	$\mathcal{O}(rKL + sKL\|\mathcal{B}\|)$
ELRG	$\mathcal{O}(r^3 + (r+2)KL\|\mathcal{B}\|)$	$\mathcal{O}((r+2)L\|\mathcal{B}\|)$
VSD	$\mathcal{O}(K^2 + KL\|\mathcal{B}\|)$	$\mathcal{O}(K^2 + K\|\mathcal{B}\|)$
VSD-low rank	$\mathcal{O}(rK + KL\|\mathcal{B}\|)$	$\mathcal{O}(K^2 + K\|\mathcal{B}\|)$

C Computational complexity

We describe here in detail the computational costs of the different algorithms, in which the computation is considered when performing a forward pass through per layer of the network. We also discuss the memory usage while constructing the dynamic computation graph. To ease the presentation, we briefly recall the abbreviations of methods from the main text, in particular: Bayes by Backprop (BBB) [5], Variational Matrix Gaussian (VMG) [35], low-rank approximations (SLANG, ELRG) [42, 59]; and the Bayesian Dropout methods including MC Dropout (MCD) [12, 13], Variational Dropout (VD) [28], and Variational Bayesian Dropout (VBD) [25, 33].

Assume the weight matrix of the layer is of size $K \times L$. First, MAP estimation performs a matrix multiplication with time cost $K \times L$ to forward each input $\mathbf{x}_i$ of size K in data batch $\mathcal{B}$. MAP estimation needs to store the output of these calculations which gives a memory cost $L|\mathcal{B}|$.

Next, BBB with naive reparameterization trick, in practice, needs to use $s \geq 2$ sampled weights of dimension $K \times L$ to reduce the variance of gradient estimator. This makes the computation hard to be performed in parallel, thus incurs multiple costs of both time and memory with $\mathcal{O}(sKL|\mathcal{B}|)$ and $\mathcal{O}(sKL + L|\mathcal{B}|)$ respectively. On the other hand, with the local reparameterization trick that translates uncertainty about global random weights into local noise in pre-activation unit, BBB can gain an alternative unbiased estimator with low variance while maintaining low complexity via sampling only a local noise. However, it requires two forward passes to obtain means and variances of the pre-activation.

For VMF, SLANG, and ELRG, the detailed analysis can be found on the original papers, and note that SLANG is a method that fails to employ the local reparameterization trick, thereby leading to very high complexity on both time and memory.

VSD adopts the advantage of Dropout training (VD) via just sampling the low dimension noise instead of whole random weights. An additional benefit is that VSD only requires one forward pass in parallel compared with two steps of the local reparameterization trick. When using a fully connected layer (FC) size of $K \times K$ to parameterize the Householder vector, namely:

$$\mathbf{v}_t = \mathbf{FC}(\mathbf{v}_{t-1}),$$
$$\mathbf{S}_t = \left(\mathbf{I} - 2\frac{\mathbf{v}_t\mathbf{v}_t^T}{\|\mathbf{v}_t\|_2^2}\right)\mathbf{S}_{t-1} = \mathbf{H}_t\mathbf{S}_{t-1},$$

for $t = 1, ..., T$, it will induce a complexity of $\mathcal{O}(K^2)$ to our method in terms of both time and memory cost. However, we also reduce the number of parameters of this FC by adding a low dimensional

Table 7: The performance of VSD when using low-rank approximation, where r is the dimension of hidden unit, $T = 2$ is the number of Householder transformations. For all metrics, lower is better.

$T = 2$	MNIST			CIFAR10			SVHN		
	FC 750x3			CNN 32x64x128					
	NLL	err. rate	ECE	NLL	err. rate	ECE	NLL	err. rate	ECE
$r = 2$	0.049	**1.12**	0.145	0.7547	25.45	**0.418**	**0.3117**	8.54	0.175
$r = 5$	0.046	1.15	0.120	**0.7378**	**25.21**	0.488	0.3138	8.64	**0.160**
$r = 10$	0.049	1.15	0.148	0.7570	25.47	0.502	0.3151	8.55	0.188
full rank	0.045	1.13	0.102	0.7497	25.28	0.461	0.3121	8.44	0.214

hidden layer. This simple solution results in lower computational time of $\mathcal{O}(rK + KL|\mathcal{B}|)$ without sacrificing much the performance (see Table 7 in Appendix D). In general, VSD has shown better computational efficiency than other structured approximation methods, even comparable with the mean-field BBNs.

D   Low-rank approximation for VSD

We investigate a *low-rank* structure in the fully connected layer used to parameterize the Householder vectors in our method. Instead of using one layer with full size $K \times K$, we add a low dimensional hidden layer with ReLU activation. The size of this hidden layer is $r \in \{2, 5, 10\}$. This idea is quite natural because it reduces significantly the number of parameters in our method while ensuring flexible parametrization for the Householder vectors thanks to the nonlinearity in hidden activation.

We show the performance of VSD with low-rank approximation in Table 7, where we repeat the experiment on image classification in Section 4.2 of our main paper. We can see that although the rank r is very small, the decrease in performance is negligible (still outperforms the baselines). This natural idea even improves the results on some settings such as the ECE metric in SVHN dataset.

E   Further discussion of uncertainty measures

We give here several approaches used to measure or assess the quality of the predictive uncertainty of models. We also present some insights into what those metrics mean. For simplicity, we consider the multi-class classification problems on the supervised dataset $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$. For Bayesian methods, we can compute the predictive probabilities for each sample $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}$ that belongs to class c as:

$$\hat{p}_{ic} := \int p(\mathbf{y}_i = c | \mathbf{x}_i, \mathbf{W}) p(\mathbf{W}|\mathcal{D}) d\mathbf{W} \approx \int p(\mathbf{y}_i = c | \mathbf{x}_i, \mathbf{W}) q(\mathbf{W}) d\mathbf{W} \approx \frac{1}{S} \sum_{s=1}^{S} p(\mathbf{y}_i = c | \mathbf{x}_i, \mathbf{W}^{(s)})$$

with $\{\mathbf{W}^{(s)}\}_{s=1}^S$ are variational Monte Carlo samples. The following metrics are all based on this predictive probability.

E.1   Negative log-likelihood

The negative log-likelihood (NLL) is a standard measure of a probabilistic model's quality and also a common uncertainty metric which is defined by: $\frac{1}{N} \sum_{i=1}^N \sum_{c=1}^K -y_{ic} \log \hat{p}_{ic}$. If the predicted
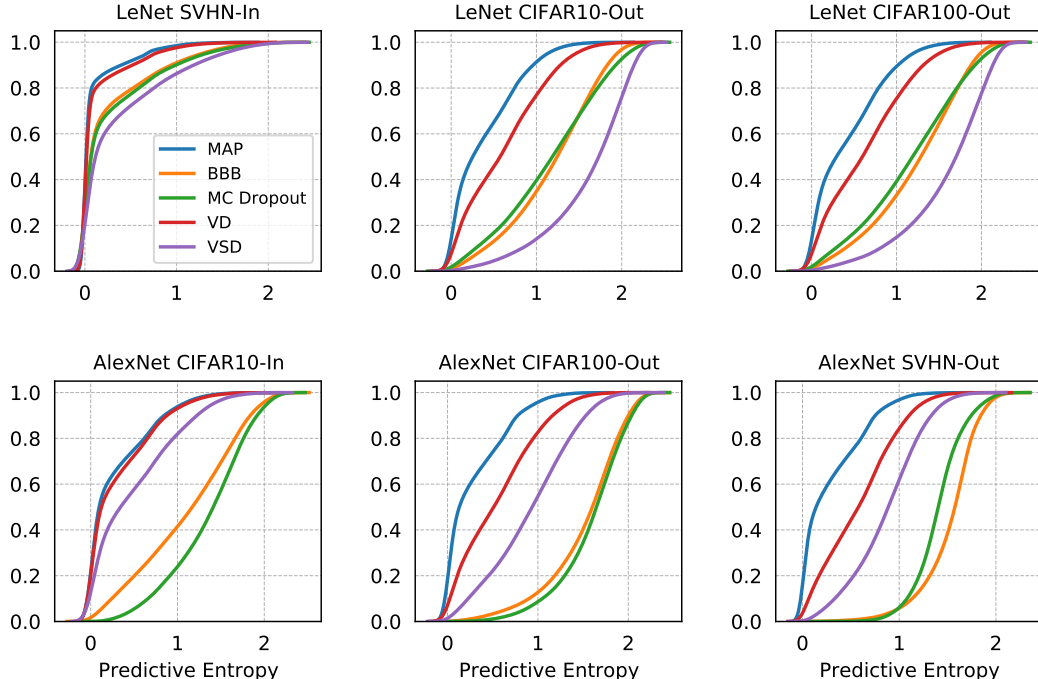
Figure 4: Empirical CDF of the predictive entropy forfor LeNet (top) and AlexNet (bottom) trained on SVHN and CIFAR10 respectively.

probabilities are overconfident, NLL will severely penalize incorrect predictions, causing this quantity to become large even if the test error rate is low. In contrast, an underconfident prediction contributes a substantial amount to the NLL regardless of whether the prediction is correct or not. Therefore, a model that achieves a good test NLL tends to make predictions with sufficiently high confidence on easy samples and hesitant predictions on hard, easy-to-fail samples.

These arguments can be evidenced by the results of experiments on modern convolutional networks (Table 5 and Table 10). Although MAP has the predictive accuracy being competitive with our method, it comes at a trade-off with the worst results on NLL. Thus the predictions of MAP are more likely to be overconfident. Corresponding experiments on out-of-distribution settings (Figure 3 bottom and Figure 6) confirmed this.

E.2   Predictive entropy

Predictive entropy determined on a input sample $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}$ is given by $\frac{1}{K}\sum_{c=1}^{K} -\hat{p}_{ic}\log\hat{p}_{ic}$. Underconfident models give noisy predictive predictions which result in high entropy on even in-distribution data. In contrast, overconfident models with spike predictive distributions tend to produce near zero predictive entropies.

In Figures 3 and Figure 6 we plot the histogram of predictive entropies to quantify uncertainty estimation of the methods on out-of-distribution settings. In some cases, when the histograms are difficult to distinguish from each other, we instead use the empirical CDF which may be more informative [36]. We redraw the Figure 3 in the main text by empirical CDF in Figure 4. The distance between lines gives more visual views on the performance differences between the methods.
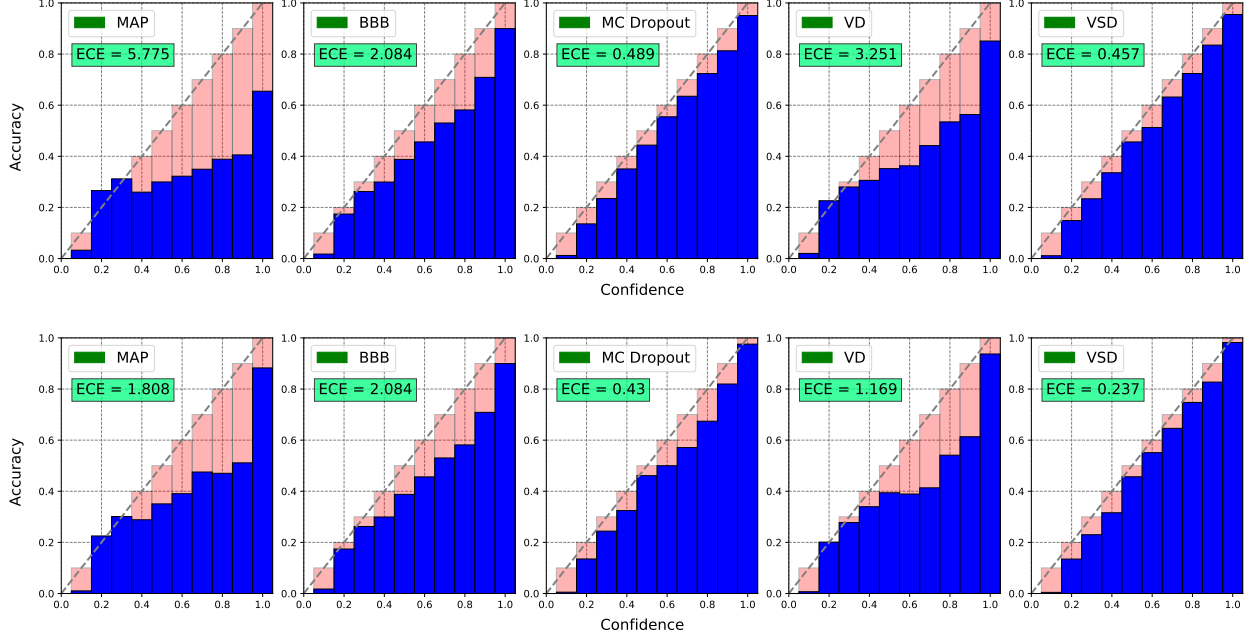
Figure 5: Reliability diagrams of LeNet-5 on CIFAR10 (top) and SVHN (bottom). All results are shown in percentages.

E.3 Expected Calibration Error

Expected Calibration Error (ECE) [16] captures the discrepancy between model's predicted probability estimates and the actual accuracy. This quantity is computed by first binning the predicted probabilities into M distinct bins and calculate the accuracy of each bin. Let B_m be the set of indices of samples whose prediction confidence falls into the interval $I_m = (\frac{m-1}{M}, \frac{m}{M}]$. The accuracy of B_m and the average confidence within bin B_m are defined as follows:

$$\text{acc}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \mathbf{1}[\hat{\mathbf{y}}_i = \mathbf{y}_i],$$

$$\text{conf}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} q(\mathbf{x}_i),$$

where $q(\mathbf{x}_i)$ is the confidence for sample i. In this work, we define the confidence score q as the maximum predictive probability on each data of the classifier. ECE is then computed by taking a weighted average of the bins' accuracy/confidence difference:

$$\text{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} |\text{acc}(B_m) - \text{conf}(B_m)|. \tag{33}$$

Reliability Diagrams [16] in Figure 5 are visual representations of model calibration. These diagrams plot accuracy as a function of confidence. Any deviation from a perfect diagonal represents miscalibration. We can observe that MAP and VD exhibit overconfident prediction (low accuracy on many bins of high confidence). Meanwhile, VSD calibrates well the predictive probabilities resulting in the lowest ECE in both settings.

24

Table 8: Quality of out-of-distribution detection on image classification tasks. (Top) LeNet-5 train on SVHN, evaluate on CIFAR10, CIFAR100. (Middle) AlexNet train on CIFAR10, evaluate on CIFAR100, SVHN. (Bottom) ResNet-18 train on CIFAR, evaluate on CIFAR10, CIFAR100. ↑ indicates larger value is better, and ↓ indicates lower value is better. VSD performs the best on almost all metrics and datasets.

LeNet-5 (SVHN)	CIFAR10					CIFAR100				
	FPR (↓)	Detection err. (↓)	AUROC (↑)	AUPR IN (↑)	AUPR OUT (↑)	FPR	Detection err.	AUROC	AUPR IN	AUPR OUT
MAP	0.78	0.23	0.83	0.93	0.58	0.76	0.22	0.84	0.93	0.60
BBB	0.56	0.17	0.90	0.96	0.73	0.54	0.17	0.90	0.96	0.75
MCD	0.50	0.15	0.92	**0.97**	0.78	0.49	0.15	**0.92**	**0.97**	0.78
VD	0.62	0.17	0.89	0.96	0.71	0.64	0.17	0.89	0.96	0.71
VSD	**0.45**	**0.14**	**0.93**	**0.97**	**0.81**	**0.47**	**0.14**	0.92	**0.97**	**0.79**

AlexNet (CIFAR10)	CIFAR100					SVHN				
	FPR	Detection err.	AUROC	AUPR IN	AUPR OUT	FPR	Detection err.	AUROC	AUPR IN	AUPR OUT
MAP	0.88	0.35	0.70	0.73	0.65	**0.89**	0.33	0.71	0.59	**0.83**
BBB	0.93	0.46	0.55	0.54	0.54	0.99	0.45	0.53	0.33	0.70
MCD	0.91	0.41	0.63	0.63	0.60	0.97	0.39	0.59	0.47	0.74
VD	0.87	0.35	0.69	0.72	0.64	0.89	0.32	0.72	0.60	**0.83**
VSD	**0.85**	**0.33**	**0.72**	**0.76**	**0.68**	0.91	**0.30**	**0.73**	**0.65**	**0.83**

ResNet-18 (CIFAR100)	CIFAR10					SVHN				
	FPR	Detection err.	AUROC	AUPR IN	AUPR OUT	FPR	Detection err.	AUROC	AUPR IN	AUPR OUT
MAP	0.89	0.37	0.67	0.7	0.63	0.91	0.36	0.68	0.56	0.81
BBB	0.93	0.41	0.62	0.66	0.58	0.89	0.37	0.68	0.51	0.82
MCD	0.89	0.37	0.68	0.71	0.63	0.89	0.34	0.71	0.58	0.83
VD	0.90	0.38	0.66	0.70	0.62	0.87	0.34	0.70	0.58	0.83
VSD	**0.87**	**0.37**	**0.69**	**0.72**	**0.65**	**0.83**	**0.31**	**0.76**	**0.65**	**0.86**

E.4 Out-of-distribution detection metrics

We measure some metrics to evaluate the model's ability of distinguishing in- and out-of-distribution images.

An ideal classifier should have a low probability of false alarm, corresponding to a low False Positive Rate (FPR), while maintaining a high sensitivity, corresponding to a high True Positive Rate (TPR). **FPR at 95% TPR** is a suitable metric to measure the reality of this desire.

Receiver Operating Characteristic (ROC) curve plots the TPR against the FPR at all possible decision thresholds. The larger the Area Under the ROC curve (**AUROC**), the better the classifier's ability to separate the negative and positive samples. More intuitively, AUROC indicates the chance that the classifier produces a higher score on a random positive sample than a random negative one.

Besides the ROC curve, which represents the trade-off between the sensitivity and the probability of false alarm, Precision-Recall (PR) curves are often plotted to represent a trade-off between making accurate positive predictions and covering a majority of all positive results. We report the Area Under the Precision-Recall curve (**AUPR**) in two scenarios: (i) in-distribution samples are used as the positive samples (**AUPR-In**), (ii) out-of-distribution samples are used as the positive samples (**AUPR-Out**).

Finally, we report the **detection error**, a measure of the minimum expected probability that the model incorrectly detects whether data samples come from in or out of training data distribution. This quantity is defined as $\min_\delta\{0.5P_{in}(q(\mathbf{x}) \leq \delta) + 0.5P_{out}(q(\mathbf{x}) > \delta)\}$ where δ is the decision threshold and $q(\mathbf{x})$ is the maximum value of softmax probability.

Table 9: Average test performance for regression task on UCI datasets. Results are reported with test LL and Std. Errors.

Dataset	BBB	VMG	SLANG	MCD	VD	VBD	VSD
Boston	-2.66 ± 0.06	-2.46 ± 0.09	-2.58 ± 0.05	-2.40 ± 0.04	-2.39 ± 0.04	-2.38 ± 0.04	**-2.35 ± 0.05**
Concrete	-3.25 ± 0.02	-3.01 ± 0.03	-3.13 ± 0.03	**-2.97 ± 0.02**	-3.07 ± 0.03	-3.06 ± 0.03	**-2.97 ± 0.02**
Energy	-1.45 ± 0.10	-1.06 ± 0.03	-1.12 ± 0.01	-1.72 ± 0.01	-1.30 ± 0.01	-1.29 ± 0.01	**-1.06 ± 0.01**
Kin8nm	1.07 ± 0.00	1.10 ± 0.01	1.06 ± 0.00	0.97 ± 0.00	1.14 ± 0.01	1.15 ± 0.01	**1.17 ± 0.01**
Naval	4.61 ± 0.01	2.46 ± 0.00	4.76 ± 0.00	4.76 ± 0.01	4.81 ± 0.00	4.81 ± 0.00	**4.83 ± 0.01**
Power Plant	-2.86 ± 0.01	-2.82 ± 0.01	-2.84 ± 0.01	**-2.79 ± 0.01**	-2.82 ± 0.01	-2.82 ± 0.01	**-2.79 ± 0.01**
Wine	-0.97 ± 0.01	-0.95 ± 0.01	-0.97 ± 0.01	**-0.92 ± 0.01**	-0.94 ± 0.01	-0.94 ± 0.01	-0.95 ± 0.01
Yacht	-1.56 ± 0.02	-1.30 ± 0.02	-1.88 ± 0.01	-1.38 ± 0.01	-1.42 ± 0.02	-1.41 ± 0.02	**-1.14 ± 0.02**

Table 10: Image classification using ResNet18 architecture. Results are averaged over 5 random seeds. For NLL and ECE metric, a lower number is better, while for ACC-predictive accuracy, a higher number is better.

ResNet18	CIFAR10			CIFAR100			SVHN			STL10		
	NLL	ACC	ECE	NLL	ACC	ECE	NLL	ACC	ECE	NLL	ACC	ECE
MAP	0.523	87.66	0.080	2.111	59.15	0.218	0.200	95.73	0.025	1.220	**77.21**	0.165
BBB	0.697	76.63	0.071	2.239	41.07	0.100	0.218	94.53	0.047	1.290	71.55	0.179
MCD	0.534	87.47	0.084	2.121	59.28	0.227	0.207	95.78	0.026	1.161	76.86	0.165
VD	0.451	**87.68**	0.017	2.888	56.80	0.007	0.164	96.11	0.003	1.084	73.29	0.042
ELRG-K=4	**0.382**	87.24	0.018	1.634	58.14	0.096	0.145	96.03	0.003	0.789	72.01	0.017
VSD	0.464	87.44	**0.014**	**1.504**	**60.15**	**0.006**	**0.140**	**96.41**	**0.001**	**0.769**	74.50	**0.016**

F Additional empirical results

F.1 Predictive likelihood on UCI regression task

We show the performance of methods in terms of predictive log-likelihood on UCI regression task. Although on some datasets such as *Concrete, Energy, Power Plant*, VSD is comparable to MCD and VMG; however, our method overall shows better results consistently above almost all settings. For instance, MCD gets poor results on *Energy, Kin8nm* and *Yacht*, while VMG is worse than VSD in *Boston, Naval* and *Yacht* by large margins. In comparison to VD and VBD, our method improves considerably the performance on *Concrete, Energy* and *Yacht* datasets.

F.2 Performance on ResNet18 architecture

In Figure 6, we show the predictive entropy of methods when training ResNet18 architechture on CIFAR100 and testing out-of-distribution on CIFAR10 and SVHN. While most methods underestimate uncertainty in out-of-distribution data, our method, VSD, calibrates the prediction with moderate confidence on in-distribution data and provides proper uncertainty on out-of-distribution settings. We also observe that BBB fails to estimate the predictive uncertainty even on in-distribution data (same behavior as on AlexNet), and with a very low accuracy, this baseline has exhibited very poor results of the mean-field BNNs compared with the Bayesian Dropout methods in terms of predictive performance.
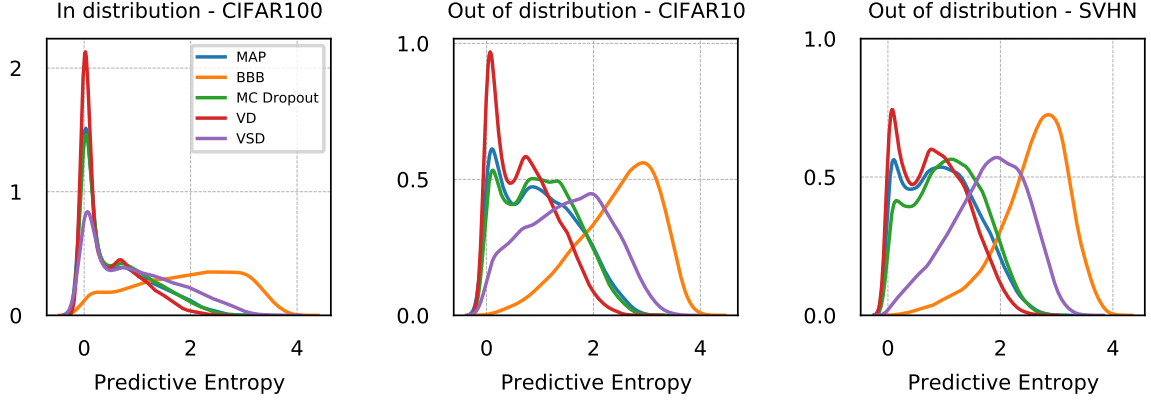
Figure 6: Histograms of predictive entropy for ResNet18 trained on CIFAR100, and then test out-of-distribution on CIFAR10 and SVHN.
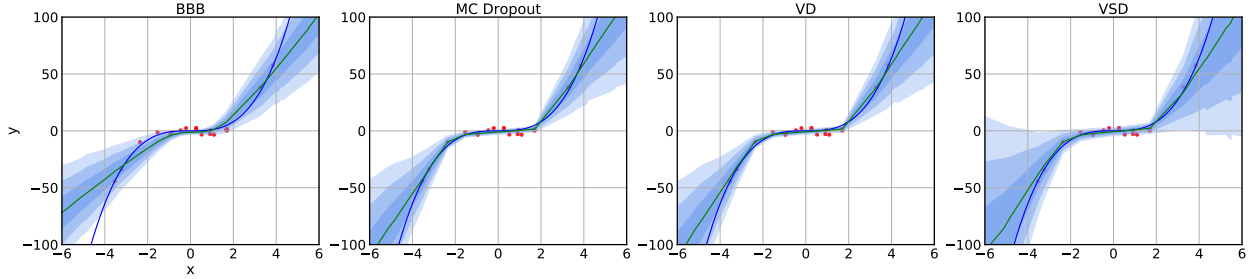

Figure 7: Predictive distributions for the toy dataset. The observations are shown as red dots. The blue line represents the true data generating function and the mean predictions are shown as the green line. Blue areas correspond to ±3 standard deviation around the mean.

G Uncertainty with toy regression

We provide an additional experiment to assess qualitatively the predictive uncertainty of methods using a synthetic regression dataset introduced at [19]. We generated 20 training inputs from $\mathcal{U}[-4, 4]$ and assigned the corresponding target as $y_n = x_n^3 + \epsilon_n$ where $\epsilon_n \sim \mathcal{N}(0, 9)$. We then fitted a neural network with a single hidden layer of 100 units. We also fixed the variance of likelihood regression to the true variance of noise ϵ. We compare the performance of methods including BBB, MCD, VD and VSD. For the Dropout-based methods, we do not apply the dropout noise for the input layer since it is 1-dimensional. At the test time of all methods, we use 1000 MC samples to approximate the predictive distribution. The results are shown in Figure 7.

We would expect that in the area of observed data, the models should obtain the predictive means closer to the ground truth with high confidence, and at the same time, increase the predictive variance when moving away from the data. Thus we can see that our method, VSD, provides a more realistic predictive distribution than the remaining ones.

For MCD, we fixed the dropout rate p at default 0.5, because tuning manually this value does not increase the variance of noise distribution Bernoulli$(1 - p)$, and this will lead to less variance in subsequent pre-activation unit by the Central Limit Theorem [61]. Therefore, with the shallow architecture of one hidden unit in this experiment, any tuning of the droprate p can result into a

27

decrease in the predictive variance of model.

Whereas with VD, the droprate α needs to be restricted in range $(0, 1)$ during the training to avoid the poor local optima or singularity in approximation [28, 43, 22]. As a consequence, this can reduce the ensemble diversity in the predictions of this method that leads to underestimate the uncertainty of predictive distribution.

On the contrary, the parameters in our method can be optimized without any limited assumptions. Moreover, with a structured representation for Gaussian perturbation, our method can capture rich statistical dependencies in the true posterior that facilitates fidelity posterior approximation and then provides proper estimates of the true model uncertainty.

H Details for experimental settings

H.1 Training techniques

Initialization and Learning rate scheduling. It is well known that good initialization and proper learning rate can improve both the speed and quality of convergence in the training process. Concretely, we use `init.xavier_uniform` in PyTorch to initialize the weight parameters of all methods. In each experiment, we use 5 random seeds for different initializations and then report average results.

For positive-valued parameters such as dropout rate α or Gaussian variance σ^2, we optimize on the logarithmic form to avoid numerical issues and bad local optima. We use Adam optimizer with initial learning rate $lr \in \{0.001, 0.002\}$ on all of our experiments, and then we also apply `MultiStepLR` scheduler with multiplicative factor `gamma=0.3` to adjust the learning rate after every 10 epochs.

KL annealing. This technique re-weights the expected log-likelihood and regularized term by a scaling factor γ as follows:

$$\mathbb{E}_{q_\phi \mathbf{W}} \log p(\mathcal{D}|\mathbf{W}) - \gamma \mathbb{D}_{KL}(q_\phi(\mathbf{W})||p(\mathbf{W})). \tag{34}$$

The KL annealing in many contexts is remarkably effective to the problems using variational Bayesian inference. It seems to prevent underfitting, over-parameterization, or to mitigate KL vanishing. We employ this technique for all methods in this paper and then tuning the weighting hyper-parameter γ depending on the data (of course with $\gamma = 1$, we have no modification).

H.2 Hyperparameter tuning for all methods

We present here in detail the hyper-parameter setups of each method used in our experiments. These methods include MAP, Bayes by Backprop (BBB), MC Dropout (MCD), Variational Dropout (VD, VBD), and our method-Variational Structured Dropout (VSD); for the remaining methods such as Variational Matrix Gaussian (VMG), low-rank approximations (SLANG, and ELRG), we inherited the results reported in the original paper with the same experimental settings.

For BBB without the local reparameterization trick, we use two Monte Carlo samples for all of our experiments. This method needs to tune the hyper-parameters in the scale mixture Gaussian prior $p(\mathbf{W}) = \pi\mathcal{N}(0, \sigma_1^2) + (1 - \pi)\mathcal{N}(0, \sigma_2^2)$ with the search as follows: $-\log \sigma_1 \in \{0, 1, 2\}$, $-\log \sigma_2 \in \{6, 7, 8\}$ and mixture ratio $\pi \in \{0.25, 0.5, 0.75\}$.

For MC Dropout in image classification task, we tune the droprate $p \in \{0.2, 0.3, 0.4, 0.5, 0.6\}$ in Bernoulli distribution and the length-scale $l^2 \in \{0.0001, 0.001, 0.01, 0.1, 1, 10, 100\}$ in the isotropic Gaussian prior $p(\mathbf{W}) = \mathcal{N}(0, l^{-2}\mathbf{I}_K)$.

For VD and VSD, we find a good initialization for the Gaussian dropout rate $\alpha = p/(1 - p)$. However, this droprate α in VD methods needs to be restricted in range $(0, 1)$ during the training to avoid the poor local optima or singularity in approximation as suggested in some related papers [28, 43, 22].

We also employ the KL annealing mentioned in the previous section H.1 for BBB, VD and VSD methods. This technique has actually been exploited in the original papers of BBB and VD to improve the predictive performance.

H.3   UCI regression settings

Following the original settings, we used a Bayesian neural network with one hidden layer of 50 units and ReLU activation functions. We also used the 20 splits of the data provided by [13][1] for training and testing. The models are trained to convergence using Adam optimizer [26] with the learning rate $lr = 0.001$, the batchsize $M = 128$ and 2000 epochs for all datasets. To make an ensemble prediction at the test time, we used 10000 Monte Carlo samples for the Bayesian Dropout methods as the suggestion in [13].

For VMG and SLANG, we inherited the results reported in the original paper. For a more fair comparison, we used the results of MC Dropout reported in [45] with the version using 4000 epochs for convergence training and tuning hyper-parameters by Bayesian Optimization.

In this experiment, we used the number of Householder transformations $T = 2$, and need to tune the precision τ of Gaussian likelihood $p(\mathbf{y}|\mathbf{W}, \mathbf{x}, \tau) = \mathcal{N}(\mathbf{y}|f(\mathbf{W}, \mathbf{x}), \tau)$. Similar to MC Dropout and SLANG, we used 40 iterations of Bayesian Optimization (BO) to tune this precision. For each iteration of BO, 5-fold cross-validation is used to evaluate the considered hyperparameter setting. This is repeated for each of the 20 train-test splits for each dataset. The final values of each dataset are reported with the mean and standard error from these 20 runs.

H.4   Image classification settings

With the MNIST dataset, 60,000 training points were split into a training set of 50,000 and a validation set of 10,000. We then vectorized the images and trained using two fully connected Bayesian neural networks with the size of hidden layers of 400x2 and 750x3 respectively.

For the remaining two datasets CIFAR10 and SVHN, we used the same simple convolutional neural network consisting of two convolutional layers with 32 and 64 kernels, followed by a fully connected network with one hidden layer of size 128.

We trained the models with the default Adam optimizer using learning rate 0.001, batchsize 100, and the number of epochs 100. At the test time, we used 10 Monte Carlo samples for all methods. We also used the number of Householder transformations $T \in \{1, 2, 3\}$ for our method on all three datasets. With Bayes by Backprop (BBB), we did not employ the local reparameterization trick, instead we used two MC samples during the training follow the code published by the authors. We utilized the available results of the baselines in the same setting, including NLL and error rate of ELRG on MNIST and CIFAR10 dataset, error rate of VMG and SLANG on MNIST dataset.

---

[1]The splits are publicly available from https://github.com/yaringal/DropoutUncertaintyExps

H.5 Scaling up modern CNNs settings

We follow the experimental setup for Bayesian deep convolutional networks proposed in [59]. We trained all algorithms for 200 epochs using a batch size of 200 and the ADAM optimizer with learning rate 0.001. We normalize datasets using empirical mean and standard deviation and then employ data augmentation for these experiments: random padding followed by flipping left/right (except SVHN). In the testing phase, we used 100 variational Monte Carlo samples for both AlexNet and ResNet18 architecture.

We report average results over 5 random initializations. We also refer to the code for MC Dropout and Bayes by Backprop on AlexNet and ResNet18 that is available at https://github. com/team-approx-bayes/dl-with-bayes.