The Radial and Directional Posteriors for Bayesian Deep Learning

Changyong Oh[*]
University of Amsterdam
changyong.oh0224@gmail.com

Kamil Adamczewski
Max Planck Institute for Intelligent Systems
kamil.m.adamczewski@gmail.com

Mijung Park
Max Planck Institute for Intelligent Systems
mijung.park@tuebingen.mpg.de

Abstract

We propose a new variational family for Bayesian neural networks. We decompose the variational posterior into two components, where the *radial* component captures the strength of each neuron in terms of its magnitude; while the *directional* component captures the statistical dependencies among the weight parameters. The dependencies learned via the directional density provide better modeling performance compared to the widely-used Gaussian mean-field-type variational family. In addition, the strength of input and output neurons learned via the radial density provides a structured way to compress neural networks. Indeed, experiments show that our variational family improves predictive performance and yields compressed networks simultaneously.

1 Introduction

In the realm of neural networks, Bayesian approaches are relatively new focusing on developing foundations of Bayesian theory for neural networks and tackling fundamental issues such as complexity. Recently, variants of such techniques were proposed in [1, 5, 3, 8], which differ in the choice of prior and posterior pairs that are often chosen for computational tractability. The so-called *mean-field* variational family in these works assumes the posterior distributions to be all factorizing, and hence neglects the possibility of modelling statistical dependencies (i.e., correlations) among weight parameters [4, 1, 7, 12, 11].

Nevertheless, capturing dependencies between the weight parameters and their uncertainties is likely to yield better models in terms of predictability [8, 14]. In this paper, we propose a new variational family with the aim of not only tackling the modelling side of BNNs in terms of capturing correlations among weight parameters but also addressing the issue of sparsification of over-parameterized neural network models. Our contributions are as briefly as follows we propose a new variational family that has two components by decomposing a weight vector into its magnitude (radius) used mostly for sparsity and compression and angle (direction) to capture correlations between the weight parameters; propose an approximation method for numerically stable gradient estimation. Thus we achieve competitive predictive and compression performance.

[*] The work done during the internship at the Max Planck Institute for Intelligent Systems.

Preprint. Under review.

2 Radial and Directional Posteriors

In variational Bayesian neural networks, in an attempt to capture distributional behaviors of the weights, we often assume a tractable parametric family for the prior distribution $p_{\boldsymbol{\theta}}(\mathbf{W})$ and the approximate posterior $q_{\boldsymbol{\phi}}(\mathbf{W})$, where the parameters for each distribution are denoted by $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$, respectively. Given a dataset $\mathcal{D}$, we maximize the variational (evidence) lower bound (ELBO) to the marginal data likelihood

$$\mathcal{L}(\mathcal{D}; \boldsymbol{\phi}, \boldsymbol{\theta}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{W})}[\log p(\mathcal{D}|\mathbf{W})] - D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{W})||p_{\boldsymbol{\theta}}(\mathbf{W})) \tag{1}$$

in order to choose the parameters of prior and posterior distributions.

One of the most commonly used variational family is fully factorized distribution referred to as the *mean-field* variational family [4]

We propose a new variational family, which is an instance of *structured* mean-field approximation where each row (and/or column) of $\mathbf{W}^{(l)}$ is factorized, $q(\mathbf{W}) = \prod_{l=1}^{L} \prod_{r=1}^{n_l} q_r^{(l)}(\mathbf{w}_r^{(l)})$, where $\mathbf{w}_r^{(l)} \in \mathbb{R}^{n_{l-1}}$ is the r-th row of the weight matrix at l-th layer. For the sake of simplicity, we consider row-wise factorization here. But we discuss both row and column-wise factorizations later which are combined with elementwise multiplication to construct a weight matrix.

This formulation assumes independence between variational parameters. However, this variational family ignores any statistical correlations between the weight parameters, which is the issue we address in this paper.

Directional density We take the prior $p_{r,dir}^{(l)}$ and the posterior $q_{r,dir}^{(l)}$ distributions to be the *von Mises-Fisher* (vMF) distribution [10], which is a probability density on a unit (hyper)sphere

$$q_{r,dir}^{(l)}\left(\frac{\mathbf{w}_r^{(l)}}{\|\mathbf{w}_r^{(l)}\|_2}\right) = vMF\left(\frac{\mathbf{w}_r^{(l)}}{\|\mathbf{w}_r^{(l)}\|_2}\middle|\boldsymbol{\mu}_r^{(l)}, \kappa_r^{(l)}\right), \tag{2}$$

where $vMF(\mathbf{x}|\boldsymbol{\mu}, \kappa) = C_d(\kappa) \exp(\kappa \boldsymbol{\mu}^T \mathbf{x})$, where $C_d(\kappa) = \frac{\kappa^{d/2-1}(\kappa)}{(2\pi)^{d/2}\mathcal{I}_{d/2-1}(\kappa)}$. The location parameter $\boldsymbol{\mu}$ is also a d-dimensional unit vector, κ is the concentration parameter, and $I_{d/2-1}$ is the modified Bessel function of the first kind at order $d/2 - 1$. The vMF distribution is intuitively understood as multivariate Gaussian distribution with a diagonal covariance matrix on unit (hyper)sphere.

In our prior and posterior distributions, we assume that the concentration parameter is shared across all the rows in each layer, by assigning a single concentration parameter, while the mean vector parameters are separately assigned for each row. This way we can reduce the number of variational and prior parameters significantly.

Explicitly modelling the directional component using vMF allows us to capture the dependence within the weight parameters of each row. Having the same concentration parameter across all the rows within each layer induces a particular way of dependence in the weight parameters within the same layer. If the mean parameters of each row's weights are close to each other, having the same concentration level, possibly a high concentration level (which we expect if the posterior confidence is high) makes the row-wise directional densities *more similar* to each other, and vice versa.

This particular way of parameterizing the variational parameters allows us to capture the dependence across rows without assigning concentration parameters to each of the rows and layers separately.

Radial density While we could adopt any probability distribution with a non-negative support for the radial density, we focus on distributions that can promote sparsity in the resulting posterior. Specifically, inspired by the group horseshoe prior proposed by [9], we take a product of two Half-Cauchy distributions to be our prior in order to induce sparsity in the *norms* of the weights. First, we write down the norm of each row given a layer as a product of two independent *half-Cauchy* random variables, $\|\mathbf{w}_r^{(l)}\|_2 = s^{(l)} \bar{z}_r^{(l)}$, where $s^{(l)} \sim \mathcal{C}^+(\gamma), \bar{z}_r^{(l)} \sim \mathcal{C}^+(1)$, and the prior is given by

$$p_{r,rad}^{(l)}(\|\mathbf{w}_r^{(l)}\|_2) = \mathcal{C}^+(s^{(l)}|\gamma) \cdot \mathcal{C}^+(\bar{z}_r^{(l)}|1), \tag{3}$$

where the probability density function for a half-Cauchy distributed random variable x is given by $\mathcal{C}^+(x|\gamma) = \frac{2}{\pi\gamma(1+(x/\gamma)^2)}$, with a scale parameter $\gamma > 0$. The smaller the scale parameter gets, the larger the probability mass concentrates around zero. At this point, it might not be immediately clear why we chose to use two Half-Cauchy distributions as a prior rather than one. Our explanation is as follows.

What we ultimately hope to control is the level of sparsity in the weights drawn from the resulting posterior distribution. We allow the posterior to have two different levels of sparsity, namely, *local* (row-wise) sparsity and *global* (layer-wise) sparsity. The reason we write the norm as a product of two terms $\|\mathbf{w}_r^{(l)}\|_2 = s^{(l)}\bar{z}_r^{(l)}$ is that each of these terms affects the local sparsity via $\bar{z}_r^{(l)}$ and global sparsity via $s^{(l)}$ in the posterior distribution, respectively. Even when all radii are small, the largest one among them has significant influence in model performance. Thus, we can use the relative strength of radius densities to prune out.

3 Optimizing evidence lower bound with RDP

Recall that as far as our objective function eq. 1 is concerned, two conditions need to be met for the gradients of this objective function to well behave. The first condition (about MC estimates of the expected log-likelihood term) is whether our posterior is reparameterizable. In fact, we can represent our choice of posterior by a differentiable function $h(\epsilon, \boldsymbol{\phi})$, where the variational parameters $\boldsymbol{\phi}$ are separated from the random source, $\epsilon \sim s(\epsilon)$.

The second condition is whether the KL term is closed-form, which is the case as we choose the prior and posterior pair considering this condition. The KL term $D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{W})\|p_{\boldsymbol{\theta}}(\mathbf{W}))$ is given by

$$\sum_{l}\sum_{r}^{n_l} D_{KL}(vMF(\boldsymbol{\mu}_{q,r}^{(l)}, \kappa_{q,r}^{(l)})\|vMF(\boldsymbol{\mu}_{p,r}^{(l)}, \kappa_p^{(l)})) + D_{KL}(q_{r,rad}^{(l)}(\|\mathbf{w}_r^{(l)}\|_2)\|p_{r,rad}^{(l)}(\|\mathbf{w}_r^{(l)}\|_2)).$$

The closed-form expressions of directional and radial components are given below and in [9], respectively.

Although the KL term between the vMF prior and posterior is elegantly written in closed-form,

$$D_{KL}(vMF(\mu_q, \kappa_q)\|vMF(\mu_p, \kappa_p)) = (\kappa_q - \kappa_p\mu_p^T\mu_q)\frac{I_{d/2}(\kappa_q)}{I_{d/2-1}(\kappa_q)} + \log(C_d(\kappa_q)) - \log(C_d(\kappa_p))$$

the gradient expressions with respect to the variational parameters require computing the ratio $\frac{I_{d/2}(\kappa_q)}{I_{d/2-1}(\kappa_q)}$ [2], which is numerically unstable. This is due to the fact that the modified Bessel function of the first kind (Bessel function) decays rapidly, so the computation of ratios of Bessel functions causes numerical errors when it tries to compute $\frac{0}{0}$. This gets worse with higher dimensions, and occurs even for moderate dimensions such as 50 to 100. Hence, rather than numerically computing the ratio of Bessel functions, we resort to the following Theorem,

Theorem 1.

$$B_2(\nu, z) < \frac{I_\nu(z)}{I_{\nu-1}(z)} < B_0(\nu, z), \quad \textit{when} \quad \nu \geq 1/2 \tag{4}$$

where $B_\alpha(\nu, z) = \frac{z}{\delta_\alpha(\nu,z)+\sqrt{\delta_\alpha(\nu,z)^2+z^2}}$, $\delta_\alpha(\nu, z) = (\nu - 1/2) + \frac{\lambda}{2\sqrt{\lambda^2+z^2}}$, $\lambda = \nu + (\alpha - 1)/2$, and ν denotes the dimension and z denotes the concentration parameter.

We observe that the gap between the upper and lower bounds of the ratio becomes tighter as the dimension grows in preliminary experiment. Even in low dimensions, the gap is less than e^{-10} for various concentration parameter values (z). Using this fact, we simply approximate the ratio by the average over the lower and upper bounds, $\frac{I_\nu(z)}{I_{\nu-1}(z)} \approx \frac{B_2(\nu,z)+B_0(\nu,z)}{2}$.

Empirically we find that this simple approximation allows us to obtain numerically stable gradients on dimensions of several thousands. Furthermore, this approximation saves us from directly computing modified Bessel function. Since the modified Bessel function of the first kind of high order is not supported yet in most deep learning frameworks, using this approximation, variational inference with high dimensional vMF distributions can enjoy GPU acceleration without extra efforts on CUDA implementations of Bessel functions.

Dataset	N	d	**VI**	**PBP**	**Dropout**	**VMG**	**RDP**
Boston	506	13	-2.90±.07	-2.57±.09	-2.46±.25	**-2.46±.09**	-2.60±.03
Concrete	1030	8	-3.33±.02	-3.16±.02	-3.04±.09	-3.01±.03	**-2.61±.02**
Energy	768	8	-2.39±.03	-2.04±.02	-1.99±.09	**-1.06±.03**	-1.18±.03
Kin8nm	8192	8	0.90±.01	0.90±.01	0.95±.03	1.10±.01	**2.17±.00**
Naval	11934	16	3.37±.12	3.73±.01	**3.80±.05**	2.46±.00	2.50±.00
Pow.Plant	9568	4	-2.89±.01	-2.84±.01	-2.80±.15	-2.82±.01	**-0.14±.01**
Protein	45730	9	-2.99±.01	-2.97±.00	-2.89±.01	-2.84±.00	**-1.34±.01**
Wine	1599	11	-0.98±.01	-0.97±.01	-0.93±.06	-0.95±.01	**-0.45±.00**
Yacht	308	6	-3.43±.16	-1.63±.02	-1.55±.12	**-1.30±.02**	-2.36±.04
Year	515345	90	-3.62±NA	-3.60±NA	-3.59±NA	-3.59±NA	**-3.51±NA**

Table 1: Average test log-likelihood on UCI regression tasks. Our method (RDP) achieves the better test likelihoods (6 out of 10 datasets) than other methods.

4 Experiments

Here, we provide empirical evidences supporting RDP's strengths. In the prediction task on UCI datasets, we compare RDP with mean-field based BNNs [4, 1, 5] and a BNN designed to capture dependency [8] in order to show RDP's capability to explain dependency between weights. In the compression task on MNIST dataset with LeNet arcthitecture, in order to check whether effective RDP's structure accommodates compression tasks, we compare RDP with various compression methods in terms of the amount of pruning and FLOPs. In all experiment, we use Adam optimizer [6] with Pytorch default setting. In both tasks, double grouping was used.

Regression using UCI data Only with the change in the first hidden layer, we can see improvement over mean-field based BNNs, such as, Variational Inference (VI) [4], Probabilistic BackPropagation(PBP) [5], Dropout [3]. Compared to another dependency awaring posterior, Variational Matrix Gaussian (VMG) [8], 6 out of 10 dataset, RDP shows better test log-likelihood(LL).

Compression on MNIST Classification The approach achieves competitive results with the state-of-the-art methods. As given in Table 2, the RDP architecture shows better compression for convolutional layers, which makes it score good at FLOPs. The proposed pruning with a third of parameters of FLOPs as a Direct Optimization Objective(100K) (FDOO) is only slightly more computationally heavy. Similarly, RDP comes only second to BC-GHS in terms of parameter number but running with two-thirds of parameters of Bayesian Compression-Group Normal Jeffrey (BC-GNJ).

Method	Architecture	FLOPs	Params	Error
RDP	4-7-110-66	125K	20K	1.0%
5-7-45-20 BC-GNJ	8-13-88-13	307K	22K	1.0%
BC-GHS	5-10-76-16	169K	**15K**	1.0%
FDOO(100K)	2-7-112-478	**119K**	66K	1.1%
FDOO(200K)	3-8-128-499	163K	81K	1.0%
GL	3-12-192-500	236K	134K	1.0%
GD	7-13-208-16	298K	49K	1.1%
SBP	3-18-284-283	295K	164K	**0.9%**

Table 2: The structured pruning of LeNet-5-Caffe with architecture 20-50-800-500. We benchmark our method against BC-GNJ, Bayesian Compression-Group HorseShoe(BC-GHS) [9], FDOO [15], Generalized Dropout(GD) [13], Group Lasso(GL) [16], Structured Bayesian Pruning(SBP) [12].

References

[1] C. e. a. Blundell. Weight uncertainty in neural networks. *arXiv:1505.05424*, 2015.

[2] T. R. e. a. Davidson. Hyperspherical variational auto-encoders. *arXiv preprint arXiv:1804.00891*, 2018.

[3] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation. In *ICML*, pages 1050–1059, 2016.

[4] A. Graves. Practical variational inference for neural networks. In *NIPS*, pages 2348–2356, 2011.

[5] J. M. Hernández-Lobato and R. Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *ICML*, pages 1861–1869, 2015.

[6] D. P. Kingma and J. Ba. A method for stochastic optimization. *arXiv:1412.6980*, 2014.

[7] D. P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. In *NIPS*, pages 2575–2583, 2015.

[8] C. Louizos and M. Welling. Structured and efficient variational deep learning with matrix gaussian posteriors. In *ICML*, pages 1708–1716, 2016.

[9] C. Louizos, K. Ullrich, and M. Welling. Bayesian compression for deep learning. In *NIPS*, pages 3288–3298, 2017.

[10] K. V. Mardia and P. E. Jupp. *Directional statistics*, volume 494. John Wiley & Sons, 2009.

[11] D. e. a. Molchanov. Variational dropout sparsifies deep neural networks. *arXiv:1701.05369*, 2017.

[12] K. e. a. Neklyudov. Structured bayesian pruning via log-normal multiplicative noise. In *NIPS*, pages 6775–6784, 2017.

[13] S. Srinivas and R. V. Babu. Data-free parameter pruning for deep neural networks. *arXiv:1507.06149*, 2015.

[14] S. e. a. Sun. Learning structured weight uncertainty in bayesian neural networks. In *AISTATS*, pages 1283–1292, 2017.

[15] R. Tang, A. Adhikari, and J. Lin. Flops as a direct optimization objective for learning sparse neural networks. *arXiv:1811.03060*, 2018.

[16] W. e. a. Wen. Learning structured sparsity in deep neural networks. In *NIPS*, pages 2074–2082, 2016.