3rd Symposium on Advances in Approximate Bayesian Inference, 2020 1–11

Marginal Likelihood Gradient for Bayesian Neural Networks

Marcin B. Tomczak MBT27@CAM.AC.UK and **Richard E. Turner** RET26@CAM.AC.UK
University of Cambridge, Cambridge, CB2 1PZ, UK

Abstract

Bayesian learning of neural networks is attractive as it can protecting against over-fitting and provide automatic methods for inferring important hyperparameters by maximizing the marginal probability of the data. However, existing approaches in this vein, such as those based on variational inference, do not perform well. In this paper, we take a different approach and directly derive a practical estimator of the gradient of the marginal log-likelihood for BNNs by combining local reparametrization of the network w.r.t. the prior distribution with the self-normalized importance sampling estimator. We show promising preliminary results on a toy example and on vectorized MNIST classification where the new method results in significantly improved performance of variational inference compared to existing approaches to tune hyperparameters.

1. Introduction

We consider a supervised learning task given a data set consisting of input-target pairs $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$. We denote the observation model for data as $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\beta})$ defining the likelihood $p(\mathcal{D}|\boldsymbol{\theta}, \boldsymbol{\beta}) = \prod_{i=1}^N p(\boldsymbol{y}_i|\boldsymbol{x}_i, \boldsymbol{\theta}, \boldsymbol{\beta})$ and prior $p(\boldsymbol{\theta}|\boldsymbol{\alpha})$, where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ denote hyperparameters. Marginal likelihood of data is defnied as $p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta}) = p(\{\boldsymbol{y}_i\}_{i=1}^N|\{\boldsymbol{x}_i\}_{i=1}^N, \boldsymbol{\alpha}, \boldsymbol{\beta})$. Given a new data point $(\boldsymbol{x}^*, \boldsymbol{y}^*)$, a fully Bayesian approach to modeling makes a prediction by marginalizing out posterior over parameters $\boldsymbol{\theta}$ and hyperparameters $\boldsymbol{\alpha}, \boldsymbol{\beta}$ (MacKay, 1992a):

$$p(\boldsymbol{y}^*|\boldsymbol{x}^*, \mathcal{D}) = \int\int\int p(\boldsymbol{y}^*|\boldsymbol{x}^*, \boldsymbol{\theta}, \boldsymbol{\beta}) p(\boldsymbol{\theta}|\mathcal{D}, \boldsymbol{\alpha}, \boldsymbol{\beta}) p(\boldsymbol{\alpha}, \boldsymbol{\beta}|\mathcal{D}) d\boldsymbol{\theta} d\boldsymbol{\alpha} d\boldsymbol{\beta}. \tag{1}$$

The integrals in Equation (1) inevitably carry high computational burden for non-trivial models, requiring us to recourse to approximate inference. We subsequently discuss approximations to posterior distributions $p(\boldsymbol{\theta}|\mathcal{D}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ and $p(\boldsymbol{\alpha}, \boldsymbol{\beta}|\mathcal{D})$.

Variational inference (VI). Variational inference (Barber and Bishop, 1998; Hinton and van Camp, 1993) reformulates finding an approximation to a posterior $p(\boldsymbol{\theta}|\mathcal{D}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ into an optimization problem finding a a simpler distribution $q(\boldsymbol{\theta}|\boldsymbol{\lambda})$ approximating $p(\boldsymbol{\theta}|\mathcal{D})$. This is done by minimizing the divergence $D_{KL}\big(q(\boldsymbol{\theta}|\boldsymbol{\lambda})||p(\boldsymbol{\theta}|\mathcal{D}, \boldsymbol{\alpha}, \boldsymbol{\beta})\big)$, which is equivalent to maximizing the Evidence Lower BOund (ELBO), a lower bound to $\log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})$, defined as:

$$\mathcal{L}(\boldsymbol{\lambda}; \boldsymbol{\alpha}, \boldsymbol{\beta}) = \sum_{i=1}^N \mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\lambda})} \log p(\boldsymbol{y}_i|\boldsymbol{x}_i, \boldsymbol{\theta}, \boldsymbol{\beta}) - D_{KL}\big(q(\boldsymbol{\theta}|\boldsymbol{\lambda})||p(\boldsymbol{\theta}|\boldsymbol{\alpha})\big). \tag{2}$$

Evidence framework. The evidence framework (MacKay, 1992a,b) provides a systematic framework for adaptation of hyperparameters $\boldsymbol{\alpha}, \boldsymbol{\beta}$. Equation (1) indicates that $p(\boldsymbol{\alpha}, \boldsymbol{\beta}|\mathcal{D})$, (and the marginal likelihood $p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})$ if the prior over $\boldsymbol{\alpha}, \boldsymbol{\beta}$ is uniform) embodies the influence of predictions arising from hyperparameters $\boldsymbol{\alpha}, \boldsymbol{\beta}$ on marginalized test predictions

© M.B. Tomczak & R.E. Turner.

$p(\boldsymbol{y}^*|\boldsymbol{x}^*, \mathcal{D})$. If the integral in Equation (1) has no closed form, it can be approximated as $p(\boldsymbol{y}^*|\boldsymbol{x}^*, \mathcal{D}) \approx \int p(\boldsymbol{y}^*|\boldsymbol{x}^*, \boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{D}, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) d\boldsymbol{\theta}$ where $\boldsymbol{\alpha}^*, \boldsymbol{\beta}^* = \text{argmax}_{\boldsymbol{\alpha},\boldsymbol{\beta}} p(\boldsymbol{\alpha}, \boldsymbol{\beta}|\mathcal{D})$. Relying only on the mode $\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*$ can lead to poor approximation of $p(\boldsymbol{y}^*|\boldsymbol{x}^*, \mathcal{D})$, depending on the concentration of density $p(\boldsymbol{\alpha}, \boldsymbol{\beta}|\mathcal{D})$. Estimating marginal log-likelihood $\log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})$ for non trivial models is a well-known problem (Barber and Bishop, 1998). Poorly chosen prior can result in ELBO not correlating with the predictive performance of the learned posterior (Trippe and Turner, 2018; Turner and Sahani, 2011).

Bayesian Neural Networks. We consider a neural network $\mathcal{F}_{\boldsymbol{W}}$ represented as composition of L fully-connected layers interleaved with activation functions $\mathcal{F}_{\boldsymbol{W}} = \mathcal{F}_{\boldsymbol{W}_L} \circ \phi \circ \mathcal{F}_{\boldsymbol{W}_{L-1}} \ldots \phi \circ \mathcal{F}_{\boldsymbol{W}_1}$ where ϕ denotes an activation function. We denote a forward pass with input $\boldsymbol{x}_l$ through l-th layer as $\mathcal{F}_{\boldsymbol{W}_l}(\boldsymbol{x}_l) = \boldsymbol{W}_l \boldsymbol{x}_l$ where biases are handled by expanding the dimension of $\boldsymbol{x}_l$. We restrict the attention to Gaussian mean-field priors $p(\boldsymbol{W}_l|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})$ and mean-field approximate posteriors $q(\boldsymbol{W}_l|\boldsymbol{\mu}_l, \boldsymbol{\sigma}^2_l)$, see Foong et al. (2019) for discussion of limitations, independent over layers and use the notation $\boldsymbol{W}_l \sim \mathcal{N}(\boldsymbol{\mu}_l, \boldsymbol{\sigma}^2_l)$ meaning $w_{l,ij} \sim \mathcal{N}(\mu_{l,ij}, \sigma^2_{l,ij})$. We adopt the notation $\boldsymbol{W} = (\boldsymbol{W}_1, \boldsymbol{W}_2, \ldots, \boldsymbol{W}_l)$ for weights, $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_l)$, $\boldsymbol{\sigma} = (\boldsymbol{\sigma}^2_1, \boldsymbol{\sigma}^2_2, \ldots, \boldsymbol{\sigma}^2_L)$ for variational parameters and $\boldsymbol{\mu}_{\boldsymbol{\alpha}} = (\boldsymbol{\mu}_{\boldsymbol{\alpha}_1}, \boldsymbol{\mu}_{\boldsymbol{\alpha}_2}, \ldots, \boldsymbol{\mu}_{\boldsymbol{\alpha}_L})$, $\boldsymbol{\sigma}^2_{\boldsymbol{\alpha}} = (\boldsymbol{\sigma}^2_{\boldsymbol{\alpha}_1}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}_1}, \ldots, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}_L})$ for prior hyperarameters. Hyperparameter $\boldsymbol{\beta}$ belongs to an observation model, for instance, the scale of Gaussian noise $\mathcal{N}(\boldsymbol{y}|\boldsymbol{0}, \text{diag}[\boldsymbol{\beta}^2])$.

Local Reparametrization Trick (LRT). The forward pass $\mathcal{F}_{\boldsymbol{W}_l}(\boldsymbol{x}_l)$ through a fully-connected layer with mean-field Gaussian distribution over weights $\boldsymbol{W}_l \sim \mathcal{N}(\boldsymbol{\mu}_l, \boldsymbol{\sigma}^2_l)$, can be reparametrized as (Kingma et al., 2015; Tomczak et al., 2020):

$$\mathcal{F}_{\boldsymbol{W}_l}(\boldsymbol{x}_l) = \mathcal{F}_{\boldsymbol{\mu}_l}(\boldsymbol{x}_l) + \sqrt{\mathcal{F}_{\boldsymbol{\sigma}^2_l}(\boldsymbol{x}_l)} \odot \boldsymbol{\epsilon}_l, \quad \boldsymbol{\epsilon}_l \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}), \tag{3}$$

where $\sqrt{\ }$ is applied element-wise and the Equation (3) denotes the equality between two random variables (LHS and RHS follow the same distribution). We denote the reparametrization noise as $\boldsymbol{\epsilon} = (\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2, \ldots, \boldsymbol{\epsilon}_L)$ and reparametrization of the network $f_{\boldsymbol{W}}$ resulting from applying Equation (3) to every layer l by $g_{\boldsymbol{\mu},\boldsymbol{\sigma}^2,\boldsymbol{\epsilon}}$. Then we have that $\mathbb{E}_{q(\boldsymbol{W}|\boldsymbol{\mu},\boldsymbol{\sigma}^2)} h(f_{\boldsymbol{W}}(\boldsymbol{x})) = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I})} h(g_{\boldsymbol{\mu},\boldsymbol{\sigma}^2,\boldsymbol{\epsilon}}(\boldsymbol{x}))$ for a continuous function h.

Approaches to adjust priors for BNNs. Hyperparameters $\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*$ can be also learned by optimizing $\mathcal{L}(\boldsymbol{\lambda}; \boldsymbol{\alpha}, \boldsymbol{\beta})$ as proxy for marginal log-likelihood since $\log p(\mathcal{D}; \boldsymbol{\alpha}, \boldsymbol{\beta}) \geq \mathcal{L}(\boldsymbol{\lambda}; \boldsymbol{\alpha}, \boldsymbol{\beta})$ (MacKay, 1999; Wu et al., 2019). The local optima of $\log p(\mathcal{D}; \boldsymbol{\alpha}, \boldsymbol{\beta})$ and $\mathcal{L}(\boldsymbol{\lambda}; \boldsymbol{\alpha}, \boldsymbol{\beta})$ can significantly vary. This is manifested when a flexible prior distribution $p(\boldsymbol{\theta}|\boldsymbol{\alpha})$ learned by optimizing $\mathcal{L}(\boldsymbol{\lambda}; \boldsymbol{\alpha}, \boldsymbol{\beta})$ collapses to the approximate posterior $q(\boldsymbol{\theta}|\boldsymbol{\lambda})$, resulting in no regularization (Blundell et al., 2015). Laplace approximation assumes $\log p(\mathcal{D}, \boldsymbol{\theta}|\boldsymbol{\alpha}, \boldsymbol{\beta})$ is a quadratic w.r.t. $\boldsymbol{\theta}$ enabling approximating $\log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})$ in closed form (MacKay, 1992c), which can be used to compare different hyperparameters $\boldsymbol{\alpha}, \boldsymbol{\beta}$. Hyperparameters $\boldsymbol{\alpha}, \boldsymbol{\beta}$ can be also chosen using cross-validation (Farquhar et al., 2020; Tomczak et al., 2018), but this requires increased computational effort, limiting the practicability of this approach.

2. Methods

First, we discuss the problems with the naive application of self-normalized importance sampling (SNIS) to estimate the gradient of marginal log-likelihood of data $\nabla_{\boldsymbol{\alpha}} \log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})$ for BNNs. We then demonstrate how taking advantage of local reparametrization of the

network allows us to mitigate these problems and derive a practical estimator of the gradient of $\nabla_{\boldsymbol{\alpha}} \log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})$ given a BNN with mean-field Gaussian prior. Recall that the gradient $\nabla_{\boldsymbol{\alpha}} \log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})$ can be derived as (Bornschein and Bengio, 2014; Tucker et al., 2019):

$$\nabla_{\boldsymbol{\alpha}} \log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta}) = \mathbb{E}_{p(\boldsymbol{W}|\mathcal{D}, \boldsymbol{\alpha}, \boldsymbol{\beta})} \nabla_{\boldsymbol{\alpha}} \log p(\boldsymbol{W}|\boldsymbol{\alpha}). \tag{4}$$

See Appendix A for similar expression for $\nabla_{\boldsymbol{\beta}} \log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})$ and associated derivations. A naive attempt to apply SNIS to Equation (4) results in estimator:

$$\hat{I}\big(\nabla_{\boldsymbol{\alpha}} \log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})\big) = \sum_{s=1}^{S} \frac{w(\boldsymbol{W}^{(s)})}{\sum_{j=1}^{S} w(\boldsymbol{W}^{(j)})} \nabla_{\boldsymbol{\alpha}} \log p(\boldsymbol{W}^{(s)}|\boldsymbol{\alpha}, \boldsymbol{\beta}), \tag{5}$$

where $w(\boldsymbol{W}^{(s)}) = p(\mathcal{D}|\boldsymbol{W}^{(s)}, \boldsymbol{\beta}) p(\boldsymbol{W}^{(s)}|\boldsymbol{\alpha}) / q(\boldsymbol{W}^{(s)})$ and weights are proposed from $\boldsymbol{W}^{(s)} \sim q(\boldsymbol{W})$. There are several challenges associated with applying the estimator given by Equation (5) making it impractical to use with BNNs. First, sampling multiple weights $\boldsymbol{W}^{(s)}$ is excessively costly even for networks of moderate size as this corresponds to sampling separate copies of the network $f_{\boldsymbol{W}}$. Second, RHS of Equation (5) has high variance being a score gradient estimator (Williams, 1992). Third, a separate objective needs to be optimized to derive proposal $q(\boldsymbol{W}|\boldsymbol{\alpha})$, e.g. by minimizing a divergence between $q(\boldsymbol{W})$ and $p(\boldsymbol{W}|\mathcal{D}, \boldsymbol{\alpha}, \boldsymbol{\beta})$. Lastly, calculating weights $w(\boldsymbol{W}^{(s)})$ requires iterating over whole data set $\mathcal{D}$.

We now demonstrate that reparametrizing the network using LRT leads to the estimator free of the two first described issues. More precisely, rather than operating in the space of random weights $\boldsymbol{W}_l$, we can reparametrize the network using LRT and operate in the space of preactivation reparametrization noise $\boldsymbol{\epsilon}_l \sim N(\boldsymbol{0}, \boldsymbol{I})$. This reduces the cost of sampling, as noise $\boldsymbol{\epsilon}_l$ has significantly lower dimensionality than weight matrices $\boldsymbol{W}_l$. In addition, the derived estimator is an improvement over one given by Equation (5) in a similar way as reparametrizing (Kingma and Welling, 2014; Rezende et al., 2014; Kingma et al., 2015) improves upon the score function estimator applied to the ELBO (Paisley et al., 2012; Ranganath et al., 2014; Titsias and Lázaro-Gredilla, 2014), since path derivative usually exhibits lower variance than score function gradient (Roeder et al., 2017). We subsequently discuss how to derive better proposal distributions, reduce the cost of computing IS weights, and derive an approximation using one optimization objective in Appendix A. We now state Lemma deriving the reparametrized gradient of marginal log-likelihood of data $\log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})$.

Lemma 1 (Marginal log-likelihood gradient for BNN with MF Gaussian prior) *Given a neural network $f_{\boldsymbol{W}}$ with layer-wise mean-field Gaussian prior $\mathcal{N}(\boldsymbol{W}_l|\boldsymbol{\mu}_{\boldsymbol{\alpha}_l}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}_l})$ where $\boldsymbol{\alpha}, \boldsymbol{\beta}$ are hyperparameters, the marginal log-likelihood gradient can be computed as:*

$$\nabla_{\boldsymbol{\alpha}} \log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta}) = \mathbb{E}_{p(\boldsymbol{\epsilon}|\mathcal{D}, \boldsymbol{\alpha}, \boldsymbol{\beta})} \sum_{i=1}^{N} \nabla_{\boldsymbol{\alpha}} \log p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}}(\boldsymbol{x}), \boldsymbol{\beta}), \tag{6}$$

where $p(\boldsymbol{\epsilon}|\mathcal{D}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ is a posterior distribution over layer's preactivations resulting from applying reparametrization in Equation (3) to layers of a network $f_{\boldsymbol{W}}$. Substituting $\nabla_{\boldsymbol{\alpha}}$ with $\nabla_{\boldsymbol{\beta}}$ in Equation (6) yields the gradient for hyperparameters of observation model $\nabla_{\boldsymbol{\beta}} \log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})$.

Note that Equation (6) still requires sampling from posterior distribution $p(\boldsymbol{\epsilon}|\mathcal{D}, \boldsymbol{\alpha}, \boldsymbol{\beta})$, where $\boldsymbol{\epsilon} = (\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2, \ldots, \boldsymbol{\epsilon}_l)$ denotes the local reparametrization noise for consecutive layers $\mathcal{F}_{\boldsymbol{W}_l}$. While analyzing the posterior distribution over reparametrization noise might

be uncommon, it is well-defined: for fixed $\boldsymbol{\alpha}, \boldsymbol{\beta}$ and for every $\boldsymbol{\epsilon}$ we define the likelihood of data $\prod_{i=1}^{N} p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}}(\boldsymbol{x}), \boldsymbol{\beta})$ and hence we can calculate posterior distribution over reparametrization noise $p(\boldsymbol{\epsilon}|\mathcal{D}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ given prior $p(\boldsymbol{\epsilon})$. The gradient in the RHS of Equation (6) resembles gradients derived for generative models (Bornschein and Bengio, 2014; Burda et al., 2015; Tucker et al., 2019). However, it is important to note not all derivations for generative models transfer to BNNs if local reparametrization is desired, as LRT requires the expression under the expectation to be only a function of layer's outputs $\{\mathcal{F}(\boldsymbol{x}_l)\}_l$. For instance, the expectation $\mathbb{E}_{q(\boldsymbol{W})} p(\boldsymbol{y}|\boldsymbol{x}, f_{\boldsymbol{W}}) p(\boldsymbol{W})/q(\boldsymbol{W})$ cannot be locally reparametrized due to the dependency on $p(\boldsymbol{W})/q(\boldsymbol{W})$. Applying SNIS to Equation (6) by proposing from distribution $q(\boldsymbol{\epsilon})$ gives us the following estimator of $\nabla_{\boldsymbol{\alpha}} \log p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})$:

$$\hat{I}(\nabla_{\boldsymbol{\alpha}} \log p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})) = \sum_{i=1}^{N} \sum_{s=1}^{S} \frac{w(\boldsymbol{\epsilon}^{(i,s)})}{\sum_{j=1}^{S} w(\boldsymbol{\epsilon}^{(i,j)})} \nabla_{\boldsymbol{\alpha}} \log p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}^{(i,s)}}(\boldsymbol{x}_i)), \ \boldsymbol{\epsilon}^{(i,s)} \sim q(\boldsymbol{\epsilon}), \tag{7}$$

where $w(\boldsymbol{\epsilon}^{(i,s)}) = \prod_{i=1}^{N} p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}^{(i,s)}}(\boldsymbol{x}), \boldsymbol{\beta}) p(\boldsymbol{\epsilon}^{(i,s)})/q(\boldsymbol{\epsilon}^{(i,s)})$ are normalized IS weights. We will employ the estimator of $\nabla_{\boldsymbol{\alpha}} \log p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})$ in Equation (7) to optimize marginal log-likelihood of data $\log p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})$ w.r.t. prior means $\boldsymbol{\mu}_{\boldsymbol{\alpha}}$ and variances $\boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}$.

Better proposal distributions. We now focus on deriving better proposal distributions $q(\boldsymbol{\epsilon})$ for estimator in Equation (7). This requires finding a distribution over noise $\boldsymbol{\epsilon}$ closer to posterior distribution $p(\boldsymbol{\epsilon}|\mathcal{D}, \boldsymbol{\alpha}, \boldsymbol{\beta})$. We define means and variances for Gaussian distributions over layer's preactivations as $\boldsymbol{\mu}_{\boldsymbol{\epsilon}} = (\boldsymbol{\mu}_{1,\boldsymbol{\epsilon}}, \boldsymbol{\mu}_{2,\boldsymbol{\epsilon}}, \ldots, \boldsymbol{\mu}_{L,\boldsymbol{\epsilon}})$ and $\boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}} = (\boldsymbol{\sigma}^2_{1,\boldsymbol{\epsilon}}, \boldsymbol{\sigma}^2_{2,\boldsymbol{\epsilon}}, \ldots, \boldsymbol{\sigma}^2_{L,\boldsymbol{\epsilon}})$. We assume factors $q(\boldsymbol{\epsilon}_l|\boldsymbol{\mu}_{l,\boldsymbol{\epsilon}}, \boldsymbol{\sigma}^2_{l,\boldsymbol{\epsilon}})$ to be mean-field Gaussian distributions and construct $q(\boldsymbol{\epsilon}|\boldsymbol{\mu}_{\boldsymbol{\epsilon}}, \boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}})$ by assuming independence between layers $q(\boldsymbol{\epsilon}|\boldsymbol{\mu}_{\boldsymbol{\epsilon}}, \boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}}) = \prod_{l}^{L} q(\boldsymbol{\epsilon}_l|\boldsymbol{\mu}_{\boldsymbol{\epsilon}_l}, \boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}_l})$. We learn the approximate posterior distribution $q(\boldsymbol{\epsilon}|\boldsymbol{\mu}_{\boldsymbol{\epsilon}}, \boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}})$ by minimizing $D_{KL}\big(q(\boldsymbol{\epsilon}|\boldsymbol{\mu}_{\boldsymbol{\epsilon}}, \boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}})||p(\boldsymbol{\epsilon}|\mathcal{D}, \boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})\big)$ leading to ELBO-like expression:

$$\mathcal{L}(\boldsymbol{\mu}_{\boldsymbol{\epsilon}}, \boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}}) = \sum_{i=1}^{N} \mathbb{E}_{q(\boldsymbol{\epsilon}|\boldsymbol{\mu}_{\boldsymbol{\epsilon}}, \boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}})} \log p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}}(\boldsymbol{x}_i)) - \sum_{l=1}^{L} D_{KL}\big(q(\boldsymbol{\epsilon}_l|\boldsymbol{\mu}_{\boldsymbol{\epsilon}_l}, \boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}_l})||\mathcal{N}(\boldsymbol{\epsilon}_l|\boldsymbol{0}, \boldsymbol{I})\big). \tag{8}$$

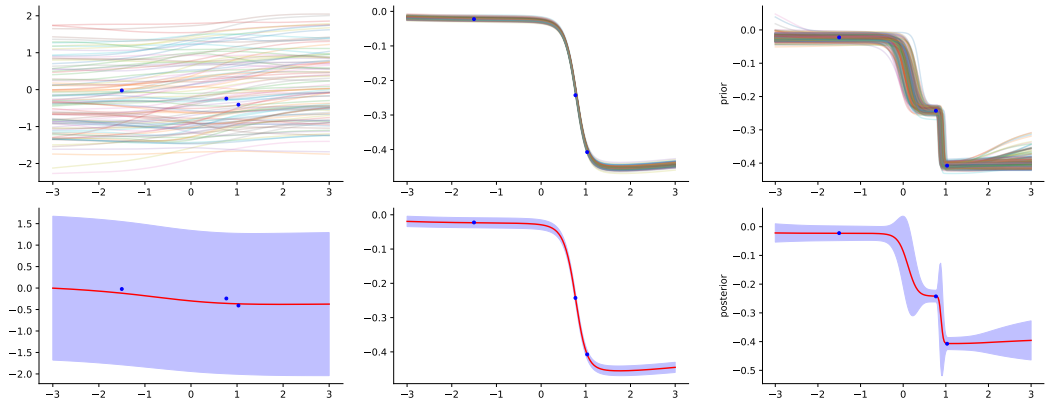
The optimal posterior distribution $q^*(\boldsymbol{\epsilon}|\boldsymbol{\mu}_{\boldsymbol{\epsilon}}, \boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}})$ derived by optimizing the $\mathcal{L}(\boldsymbol{\mu}_{\boldsymbol{\epsilon}}, \boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}})$ depends on the prior hyperparameters $\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}$, so in practice we will optimize $\mathcal{L}(\boldsymbol{\mu}_{\boldsymbol{\epsilon}}, \boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}})$ by performing one (or possibly a few) updates while we simultaneously learning $\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}$. This setup of learning jointly learning model parameters $\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}$ and auxiliary parameters $\boldsymbol{\mu}_{\boldsymbol{\epsilon}}, \boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}}$ resembles algorithms developed for generative models (Kingma and Welling, 2014; Bornschein and Bengio, 2014; Burda et al., 2015; Rezende et al., 2014).

Computing weights $w(\boldsymbol{\epsilon})$. The cost of calculating weights $w(\boldsymbol{\epsilon})$ grows linearly with the size of data set $\mathcal{D}$. To ease the computational cost of iterating over whole data set $\mathcal{D}$ to calculate $w(\boldsymbol{\epsilon})$, we propose to employ the "average" likelihood calculated based on subsample set of data points, similarly as Li and Turner (2016). While the theoretical properties of this approximation are not well-studied, it appears to be a sensible heuristic.

Preventing over-fitting. Adjusting hyperparameters by optimizing $\log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})$ (type II MLE) can be prone to over-fitting. The standard approach to mitigate over-fitting hyperparameters is to introduce hyperpriors $p(\boldsymbol{\alpha})$ and $p(\boldsymbol{\beta})$. In addition, for BNNs, the variances of units $\boldsymbol{\sigma}^2_l$ can be shared across the layer l, so that $\boldsymbol{\sigma}^2_l = \sigma^2_l \boldsymbol{1}$. The gradients of log hyperprior $\nabla_{\boldsymbol{\alpha}} \log p(\boldsymbol{\alpha})$ and $\nabla_{\boldsymbol{\beta}} \log p(\boldsymbol{\beta})$ are then added to the Equation (7).

3. Experimental demonstration

We will now empirically investigate the differences between learning hyperparameters $\boldsymbol{\alpha}, \boldsymbol{\beta}$ via optimizing $\log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})$ and $\mathcal{L}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2; \boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})$. We demonstrate that relying on fixed hyperparameters and optimizing ELBO w.r.t. hyperparameters $\boldsymbol{\alpha}, \boldsymbol{\beta}$ leads to underconfident and overconfident predictions, respectively. Employing the derived estimator of $\nabla_{\boldsymbol{\alpha}} \log p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})$ allows us to mitigate this behavior and leads to $\boldsymbol{\alpha}$ yielding better generalization. In our experiments we assume layer-wise Gaussian hyperprior over shared (within every layer) prior log variances $p(\log \sigma^2_{\alpha_l}) = \mathcal{N}(\log \sigma^2_{\alpha_l} | \log[50/(N_{in}+1)], \gamma_l^2)$, where N_{in} is the size of the layer's input and we set $\gamma_l^2 = 500$.



(a) Fixed hyperparameters (b) optimizing ELBO (c) Log marginal gradient

Figure 1: Comparison of different approaches to infer prior means, variances and observation noise for regression with one hidden layer BNN. Prior distribution over function (top) and posterior learned with VI (bottom).

One dimensional regression. We first consider one dimensional regression where we attempt to learn function $f(x) = (x+1)\sin(2x+2) + 0.3\epsilon$, $\epsilon \sim \mathcal{N}(0,1)$ after observing 3 data points. We learn a variational posterior for two hidden layer network with 128 hidden units, tanh activations and assume Gaussian observation model $\log p(y|f_{\boldsymbol{W}}(x), \beta) \propto -(y - f_{\boldsymbol{W}}(x))^2/2\beta^2 - \log \beta$. We use ADAM optimizer (Kingma and Ba, 2014) to optimize $\mathcal{L}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2; \boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})$. We consider three approaches: (a) use fixed value of observation noise $\beta = 1$, zero prior unit means $\boldsymbol{\mu}$ and fixed variances $\sigma^2_\alpha = 1/N_{in}$ (Neal, 1996), (b) optimize $\mathcal{L}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2; \boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \beta)$ w.r.t. both variational parameters $\boldsymbol{\mu}, \boldsymbol{\sigma}^2$, and prior hyperparameters $\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \beta$, and (c) learn $\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \beta$ using evidence gradient given by Equation (7) and then learn variational posterior $q(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ by optimizing $\mathcal{L}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2; \boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \beta)$. We report prior and posterior predictive distribution in Figure 1. First, in Figure 1(a) we observe that MF-VI with predefined variances of units can severely underfit data if prior variances of the units are not tuned. This happens when the optimization of $\mathcal{L}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2; \boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \beta)$ is driven by reducing complexity penalty $D_{KL}\big(q(\boldsymbol{\theta}|\boldsymbol{\lambda})||p(\boldsymbol{\theta}|\boldsymbol{\alpha})\big)$ and variational posterior explains data $\mathcal{D}$ by noise. Next, in Figure 1(b) we see that learning prior hyperparameters by optimizing $\mathcal{L}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2; \boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \beta)$ leads to an improved performance, but results in overconfident predictive distribution. This happens as optimizing prior $p(\boldsymbol{W}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})$ using Equation (2) can

set it very close to posterior $q(\boldsymbol{W}|\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$, yielding almost no regularization. Lastly, we see from Figure 1(c) that using the derived marginal log-likelihood gradient results in a prior explaining the data $\mathcal{D}$ with subsequent application of VI yielding reasonable performance.

Vectorized MNIST classification. Next we investigate if adjusting hyperparameters leads to improved performance of Gaussian MF-VI on a vectorized MNIST classification task, a standard benchmark for Bayesian deep learning (Blundell et al., 2015). We learn a two hidden layer network with 400 hidden units and ReLU activations (Nair and Hinton, 2010) and report learning curves as optimization proceeds in Figure 2. We use the same hyperpriors for all approaches and compare using predefined hyperparameters with learning them via optimizing ELBO and using the derived marginal log-likelihood gradient.

Overall, we observe similar patterns as in 1D regression experiment. Using predefined set of hyperparameters yields underfitting with test NLL 0.095 and test error rate 2.49%, as optimizing ELBO constrains $q(\boldsymbol{W}|\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ to prior distribution incapable of modeling data. Tuning prior by optimizing ELBO leads to over-fitting (test NLL 0.177 and test error rate 1.57%), as prior distribution $p(\boldsymbol{W}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})$ collapse to posterior $q(\boldsymbol{W}|\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ leading to insufficient regularization. Using marginal log-likelihood gradient given by Equation (7) optimizes prior $p(\boldsymbol{W}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})$ directly to fit the data and hence introduces sufficient regularization when optimizing ELBO, leading to the best test log likelihood 0.047 and test error rate 1.37%.

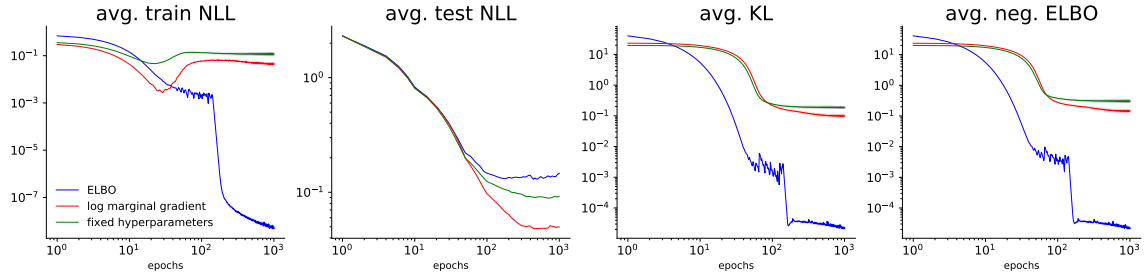


Figure 2: Different approaches to infer prior means and shared variances across layers for MF-VI on vectorized MNIST classification. Using marginal log-likelihood gradient given by Equation (7) (red curve) leads to the best predictive performance.

4. Conclusions

We have derived a pragmatic algorithm adjusting hyperparameters of BNNs with mean-field Gaussian priors by optimizing marginal log-likelihood of data. To this end, we applied local reparametrization of the network combined with self-normalized importance sampling. The optimized priors yielded predictive distributions explaining the data and subsequent application of VI resulted in posterior distributions with good predictive performance. We hope our work will catalyze research on benefits of reparametrization and algorithms inferring hyperparameters for BNNs.

References

D. Barber and Christopher Bishop. Ensemble learning in bayesian neural networks. In *Generalization in Neural Networks and Machine Learning*, pages 215–237. Springer Verlag, January 1998. URL https://www.microsoft.com/en-us/research/publication/ensemble-learning-in-bayesian-neural-networks/.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 1613–1622. JMLR.org, 2015.

Jörg Bornschein and Yoshua Bengio. Reweighted wake-sleep. 2014.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. 2015.

Sebastian Farquhar, Michael A. Osborne, and Yarin Gal. Radial bayesian neural networks: Beyond discrete support in large-scale bayesian deep learning. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 1352–1362. PMLR, 26–28 Aug 2020.

Andrew YK Foong, David R Burt, Yingzhen Li, and Richard E Turner. On the expressiveness of approximate inference in bayesian neural networks. *arXiv e-prints*, pages arXiv–1909, 2019.

Geoffrey E. Hinton and Drew van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, COLT '93, page 5–13, New York, NY, USA, 1993. Association for Computing Machinery. ISBN 0897916115. doi: 10.1145/168304.168306. URL https://doi.org/10.1145/168304.168306.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6114.

Diederik P. Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, page 2575–2583, Cambridge, MA, USA, 2015. MIT Press.

Yingzhen Li and Richard E Turner. Rényi divergence variational inference. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1073–1081. Curran Associates, Inc., 2016. URL http://papers.nips.cc/paper/6208-renyi-divergence-variational-inference.pdf.

David J. C. MacKay. Bayesian interpolation. *Neural Comput.*, 4(3):415–447, May 1992a. ISSN 0899-7667. doi: 10.1162/neco.1992.4.3.415. URL https://doi.org/10.1162/neco.1992.4.3.415.

David J. C. MacKay. The evidence framework applied to classification networks. 4(5): 720–736, September 1992b. ISSN 0899-7667. doi: 10.1162/neco.1992.4.5.720. URL https://doi.org/10.1162/neco.1992.4.5.720.

David J. C. MacKay. A practical bayesian framework for backpropagation networks. *Neural Comput.*, 4(3):448–472, May 1992c. ISSN 0899-7667. doi: 10.1162/neco.1992.4.3.448. URL https://doi.org/10.1162/neco.1992.4.3.448.

David J. C. MacKay. Comparison of approximate methods for handling hyperparameters. *Neural Comput.*, 11(5):1035–1068, July 1999. ISSN 0899-7667. doi: 10.1162/089976699300016331. URL https://doi.org/10.1162/089976699300016331.

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. ICML'10, page 807–814, Madison, WI, USA, 2010. Omnipress. ISBN 9781605589077.

Radford M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag, Berlin, Heidelberg, 1996. ISBN 0387947248.

John Paisley, David M. Blei, and Michael I. Jordan. Variational bayesian inference with stochastic search. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, page 1363–1370, Madison, WI, USA, 2012. Omnipress. ISBN 9781450312851.

Rajesh Ranganath, Sean Gerrish, and David Blei. Black Box Variational Inference. In Samuel Kaski and Jukka Corander, editors, *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, volume 33 of *Proceedings of Machine Learning Research*, pages 814–822, Reykjavik, Iceland, 22–25 Apr 2014. PMLR. URL http://proceedings.mlr.press/v33/ranganath14.html.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1278–1286, Bejing, China, 22–24 Jun 2014. PMLR. URL http://proceedings.mlr.press/v32/rezende14.html.

Geoffrey Roeder, Yuhuai Wu, and David K Duvenaud. Sticking the landing: Simple, lower-variance gradient estimators for variational inference. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6925–6934. Curran Associates, Inc., 2017.

Michalis Titsias and Miguel Lázaro-Gredilla. Doubly stochastic variational bayes for non-conjugate inference. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine*

Learning Research, pages 1971–1979, Bejing, China, 22–24 Jun 2014. PMLR. URL http://proceedings.mlr.press/v32/titsias14.html.

Marcin B. Tomczak, Siddharth Swaroop, and Richard E. Turner. Neural network ensembles and variational inference revisited. *In 1st Symposium on Advances in Approximate Bayesian Inference*, pages 1–11, 2018.

Marcin B. Tomczak, Siddharth Swaroop, and Richard E. Turner. Efficient low rank gaussian variational inference for neural networks. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://papers.nips.cc/paper/2020/file/310cc7ca5a76a446f85c1a0d641ba96d-Paper.pdf.

Brian Trippe and Richard Turner. Overpruning in variational bayesian neural networks. *arXiv preprint arXiv:1801.06230*, 2018.

George Tucker, Dieterich Lawson, Shixiang Gu, and Chris J. Maddison. Doubly reparameterized gradient estimators for monte carlo objectives. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HkG3e205K7.

R. E. Turner and M. Sahani. Two problems with variational expectation maximisation for time-series models. In D. Barber, T. Cemgil, and S. Chiappa, editors, *Bayesian Time series models*, chapter 5, pages 109–130. Cambridge University Press, 2011.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3–4):229–256, May 1992. ISSN 0885-6125. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Anqi Wu, Sebastian Nowozin, Edward Meeds, Richard E. Turner, Jose Miguel Hernandez-Lobato, and Alexander L. Gaunt. Deterministic variational inference for robust bayesian neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1l08oAct7.

Appendix A.

We provide derivations of $\nabla_{\boldsymbol{\alpha}} \log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})$ and $\nabla_{\boldsymbol{\beta}} \log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})$ in Equation (6).

$$\nabla_{\boldsymbol{\alpha}} \log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})} \nabla_{\boldsymbol{\alpha}} p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta}) \tag{9}$$

$$= \frac{1}{p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})} \nabla_{\boldsymbol{\alpha}} \int p(\mathcal{D}|\boldsymbol{W}, \boldsymbol{\beta}) p(\boldsymbol{W}|\boldsymbol{\alpha}) d\boldsymbol{W} \tag{10}$$

$$= \frac{1}{\log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})} \int \nabla_{\boldsymbol{\alpha}} p(\boldsymbol{W}|\boldsymbol{\alpha}) p(\mathcal{D}|\boldsymbol{W}, \boldsymbol{\beta}) d\boldsymbol{W} \tag{11}$$

$$= \frac{1}{p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})} \int p(\boldsymbol{W}|\boldsymbol{\alpha}) \nabla_{\boldsymbol{\alpha}} \log p(\boldsymbol{W}|\boldsymbol{\alpha}) p(\mathcal{D}|\boldsymbol{W}, \boldsymbol{\beta}) d\boldsymbol{W} \tag{12}$$

$$= \int \frac{p(\mathcal{D}|\boldsymbol{W}, \boldsymbol{\beta}) p(\boldsymbol{W}|\boldsymbol{\alpha})}{p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})} \nabla_{\boldsymbol{\alpha}} \log p(\boldsymbol{W}|\boldsymbol{\alpha}) d\boldsymbol{W} \tag{13}$$

$$= \mathbb{E}_{p(\boldsymbol{W}|\mathcal{D}, \boldsymbol{\alpha}, \boldsymbol{\beta})} \nabla_{\boldsymbol{\alpha}} \log p(\boldsymbol{W}|\boldsymbol{\alpha}). \tag{14}$$

$$\nabla_{\boldsymbol{\beta}} \log p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})} \nabla_{\boldsymbol{\beta}} p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta}) \tag{15}$$

$$= \frac{1}{p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})} \nabla_{\boldsymbol{\beta}} \int p(\mathcal{D}|\boldsymbol{W}, \boldsymbol{\beta}) p(\boldsymbol{W}|\boldsymbol{\alpha}) d\boldsymbol{W} \tag{16}$$

$$= \frac{1}{p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})} \int p(\boldsymbol{W}|\boldsymbol{\alpha}) \nabla_{\boldsymbol{\beta}} p(\mathcal{D}|\boldsymbol{W}, \boldsymbol{\beta}) d\boldsymbol{W} \tag{17}$$

$$= \frac{1}{p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})} \int p(\boldsymbol{W}|\boldsymbol{\alpha}) p(\mathcal{D}|\boldsymbol{W}, \boldsymbol{\beta}) \nabla_{\boldsymbol{\beta}} \log p(\mathcal{D}|\boldsymbol{W}, \boldsymbol{\beta}) d\boldsymbol{W} \tag{18}$$

$$= \int \frac{p(\mathcal{D}|\boldsymbol{W}, \boldsymbol{\beta}) p(\boldsymbol{W}|\boldsymbol{\alpha})}{p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta})} \nabla_{\boldsymbol{\beta}} \log p(\mathcal{D}|\boldsymbol{W}, \boldsymbol{\beta}) \tag{19}$$

$$= \mathbb{E}_{p(\boldsymbol{W}|\mathcal{D}, \boldsymbol{\alpha}, \boldsymbol{\beta})} \nabla_{\boldsymbol{\beta}} \log p(\mathcal{D}|\boldsymbol{W}, \boldsymbol{\beta}). \tag{20}$$

We now present the proof of Lemma 1. The derivation uses the reparametrization of the network $f_{\boldsymbol{W}}$ into $g_{\boldsymbol{\mu}, \boldsymbol{\sigma}^2, \boldsymbol{\epsilon}}$ together with applying log trick $\nabla f(x) = f(x) \nabla \log f(x)$.

Proof Derivation of Lemma 1.

$$\nabla_{\boldsymbol{\alpha}} \log p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}) = \tag{21}$$

$$= \frac{1}{p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})} \nabla_{\boldsymbol{\alpha}} p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}) \tag{22}$$

$$= \frac{1}{p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})} \nabla_{\boldsymbol{\alpha}} \int p(\mathcal{D}|\boldsymbol{W}, \boldsymbol{\beta}) p(\boldsymbol{W}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}) d\boldsymbol{W} \tag{23}$$

$$= \frac{1}{p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})} \nabla_{\boldsymbol{\alpha}} \int \prod_{i=1}^{N} p(\boldsymbol{y}_i|f_{\boldsymbol{W}}(\boldsymbol{x}_i), \boldsymbol{\beta}) p(\boldsymbol{W}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}) d\boldsymbol{W} \tag{24}$$

$$= \frac{1}{p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})} \nabla_{\boldsymbol{\alpha}} \int \prod_{i=1}^{N} p(\boldsymbol{y}_i|g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}}(\boldsymbol{x}_i), \boldsymbol{\beta}) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon} \tag{25}$$

$$= \frac{1}{p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})} \int p(\boldsymbol{\epsilon}) \nabla_{\boldsymbol{\alpha}} \prod_{i=1}^{N} p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}}(\boldsymbol{x}_i), \boldsymbol{\beta}) d\boldsymbol{\epsilon} \tag{26}$$

$$= \frac{1}{p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})} \int p(\boldsymbol{\epsilon}) \prod_{i=1}^{N} p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}}(\boldsymbol{x}_i), \boldsymbol{\beta}) \nabla_{\boldsymbol{\alpha}} \log \prod_{i=1}^{N} p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}}(\boldsymbol{x}_i), \boldsymbol{\beta}) d\boldsymbol{\epsilon} \tag{27}$$

$$= \frac{1}{p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})} \int p(\boldsymbol{\epsilon}) \prod_{i=1}^{N} p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}}(\boldsymbol{x}_i), \boldsymbol{\beta}) \nabla_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \log p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}}(\boldsymbol{x}_i), \boldsymbol{\beta}) d\boldsymbol{\epsilon} \tag{28}$$

$$= \int \frac{p(\boldsymbol{\epsilon}) \prod_{i=1}^{N} p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}}(\boldsymbol{x}_i))}{p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})} \sum_{i=1}^{N} \nabla_{\boldsymbol{\alpha}} \log p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}}(\boldsymbol{x}_i), \boldsymbol{\beta}) d\boldsymbol{\epsilon} \tag{29}$$

$$= \int \frac{p(\boldsymbol{\epsilon}) p(\mathcal{D}|\boldsymbol{\epsilon}, \boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})}{p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})} \sum_{i=1}^{N} \nabla_{\boldsymbol{\alpha}} \log p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}}(\boldsymbol{x}_i), \boldsymbol{\beta}) d\boldsymbol{\epsilon} \tag{30}$$

$$= \mathbb{E}_{p(\boldsymbol{\epsilon}|\mathcal{D}, \boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})} \sum_{i=1}^{N} \nabla_{\boldsymbol{\alpha}} \log p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}}(\boldsymbol{x}_i), \boldsymbol{\beta}) \tag{31}$$

Following the same derivations for $\nabla_{\boldsymbol{\beta}}$ yields

$$\nabla_{\boldsymbol{\beta}} \log p(\mathcal{D}|\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}) = \mathbb{E}_{p(\boldsymbol{\epsilon}|\mathcal{D}, \boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}})} \sum_{i=1}^{N} \nabla_{\boldsymbol{\beta}} \log p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\epsilon}}(\boldsymbol{x}_i), \boldsymbol{\beta}), \tag{32}$$

■

Approximate scheme using one objective. Approximating weights with uniform distribution $w(\boldsymbol{\epsilon}^{(i,s)}) = \frac{1}{S}$ can be used to formulate the objective that can be used to learn both $\boldsymbol{\mu}_{\boldsymbol{\epsilon}}, \boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}}$ and $\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}$:

$$\tilde{\mathcal{L}}_1(\boldsymbol{\mu}_{\boldsymbol{\epsilon}}, \boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}}, \boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}) \approx$$
$$\sum_{i=1}^{N} \mathbb{E}_{\boldsymbol{\epsilon}'} \log p(\boldsymbol{y}_i | g_{\boldsymbol{\mu}_{\boldsymbol{\alpha}}, \boldsymbol{\sigma}^2_{\boldsymbol{\alpha}}, \boldsymbol{\mu}_{\boldsymbol{\epsilon}} + \boldsymbol{\sigma}_{\boldsymbol{\epsilon}} \boldsymbol{\epsilon}'}(\boldsymbol{x}_i)) - \sum_{l=1}^{L} D_{KL}\big(q(\boldsymbol{\epsilon}_l | \boldsymbol{\mu}_{\boldsymbol{\epsilon}_l}, \boldsymbol{\sigma}^2_{\boldsymbol{\epsilon}_l}) || \mathcal{N}(\boldsymbol{\epsilon}_l | \mathbf{0}, \boldsymbol{I})\big) + \log p(\boldsymbol{\alpha}, \boldsymbol{\beta}). \tag{33}$$