Mapping Gaussian Process Priors to Bayesian Neural Networks

Daniel Flam-Shepherd
University of Toronto

James Requeima
University of Cambridge

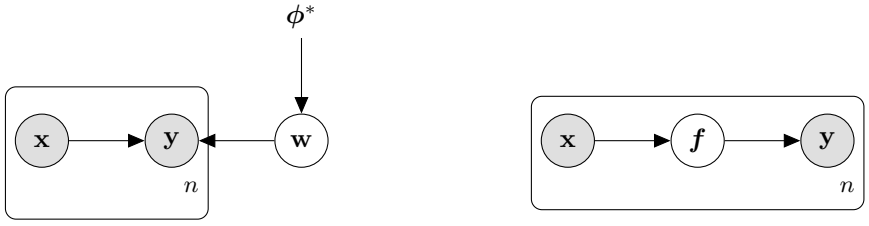
David Duvenaud
University of Toronto

1 Introduction and Motivation

What defines a reasonable prior to use when forming and training Bayesian models and Bayesian neural networks? In a recent work, (Ghosh et al 2016) apply a horseshoe prior over preactivations of a Bayesian neural network to effectively turns off weights that do not help explain the data. That model and many Bayesian models view priors solely in parameter space, in this work we move forward with viewing priors in function space as well.

It is difficult to incorporate meaningful prior information about functions to be modelled by BNNs since priors are generally specified over the network parameters. Often, normal distributions are placed over the weights for convenience and are interpreted as a bias toward less complex functions via smaller weights. Gaussian processes, on the other hand, have a elegant mechanism for incorporating prior beliefs about the underlying function - specifying the mean and covariance functions. However, Gaussian Process have scalability limitations making Bayesian neural networks a more practical model in large data settings. In our work, we present an approach to specify a more principled prior for Bayesian Neural Networks that can leverage the well studied kernel design techniques from Gaussian process regression.

We consider matching the prior of a Bayesian neural network $p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})$ to the prior of a Gaussian process $p_{\text{GP}}(\boldsymbol{f}|\boldsymbol{\theta})$ by minimizing their KL divergence over some data distribution of interest $\mathbf{X} \sim p(\mathbf{X})$. We minimize the KL divergence with respect to the initial variational parameters $\boldsymbol{\phi}$ of the proposal distribution $q(\mathbf{w}|\boldsymbol{\phi})$. These variational parameters $\boldsymbol{\phi}^* = \{\boldsymbol{\mu}^*_{\boldsymbol{\phi}}, \log \boldsymbol{\sigma}^*_{\boldsymbol{\phi}}\}$ yield a prior on the BNN weights $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}^*_{\boldsymbol{\phi}}, \boldsymbol{\sigma}^*_{\boldsymbol{\phi}}) \equiv q(\mathbf{w}|\boldsymbol{\phi}^*)$. Then, variational inference allows us to perform approximate inference in our BNN using this more principled prior. In both stochastic optimization steps, we use the reparameterization trick (Kingma; Rezende 2014) to sample from the weights $\mathbf{w}$ and draw functions from our BNN. We describe the implementation of both steps in the next section.



(a) BNN (b) GP

Figure 1: (a) and (b) display the graphical models of a BNN and GP.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

2 Model Description and Implementation

Algorithm 1 Optimization of the prior of the Bayesian neural net

Initialize $\phi = \{\boldsymbol{\mu}_\phi, \log \boldsymbol{\sigma}_\phi\}$
while ϕ not converged **do**
  $\boldsymbol{\epsilon}^{(s)} \sim p(\boldsymbol{\epsilon}) = \mathcal{N}(\mathbf{0}, \mathbb{I})$ ▷ sample prior noise
  $\mathbf{w}^{(s)} \leftarrow g(\boldsymbol{\phi}, \boldsymbol{\epsilon}) = \boldsymbol{\mu}_\phi + \boldsymbol{\sigma}_\phi \boldsymbol{\epsilon}^{(s)}$ ▷ sample S weights $\mathbf{w} \sim q(\mathbf{w}|\phi)$
  $\mathbf{X} \leftarrow (\mathbf{x}_1, \ldots, \mathbf{x}_n) \sim p(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ ▷ sample data
  $\boldsymbol{f}^{(s)} \leftarrow f_{\text{NN}}(\mathbf{X}, \mathbf{w}^{(s)})$ ▷ sample S functions $\boldsymbol{f} \sim p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})$
  $\nabla_{\boldsymbol{\phi}} \mathcal{L}_{\mathbf{X}}(\boldsymbol{\phi}) \leftarrow -\frac{1}{S} \sum_s \mathbb{E}_{p(\mathbf{X})}[\nabla_{\boldsymbol{\phi}} \log p_{\text{GP}}(\boldsymbol{f}^{(s)}(\mathbf{x})|\boldsymbol{\theta})]$ ▷ compute gradients of the objective
  $\boldsymbol{\phi} \leftarrow \text{adam}(\boldsymbol{\phi}, \nabla_{\boldsymbol{\phi}} \mathcal{L}_{\mathbf{X}}(\boldsymbol{\phi}))$ ▷ update the parameters
Return $\boldsymbol{\phi}^*$

2.1 Step 1. Learn the prior parameters

In this section we describe the procedure used to minimize the KL divergence of the BNN prior distribution over functions $p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})$ and the GP prior distribution over functions $p_{\text{GP}}(\boldsymbol{f}|\boldsymbol{\theta})$ where $\boldsymbol{\theta}$ are hyperparameters of the kernel $\mathbf{K}$.

$$\mathcal{K}_{\mathbf{X}}(\boldsymbol{\phi}) = \mathbb{KL}[p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi}) \mid p_{\text{GP}}(\boldsymbol{f}|\boldsymbol{\theta})] = \int p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi}) \log \left[\frac{p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})}{p_{\text{GP}}(\boldsymbol{f}|\boldsymbol{\theta})} \right] d\boldsymbol{f} \tag{1}$$

$$= -\mathbb{H}[p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})] - \mathbb{E}_{p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})}[\log p_{\text{GP}}(\boldsymbol{f}|\boldsymbol{\theta})] \propto -\frac{1}{S} \sum_{s=1}^{S} \log p_{\text{GP}}(\boldsymbol{f}^{(s)}|\boldsymbol{\theta}) \tag{2}$$

Where we have used a Monte Carlo estimate of $\mathbb{E}_{p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})}[\log p_{\text{GP}}(\boldsymbol{f}|\boldsymbol{\theta})]$, using S samples $\boldsymbol{f}^{(s)} \sim p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})$. We also assume that the entropy term $\mathbb{H}[p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})]$ is not a function of the variational parameters $\boldsymbol{\phi}$. We define our first stochastic optimization objective by approximating the KL divergence between these infinite dimensional distributions by taking expectations over where $p(\mathbf{X})$ allows us to prioritize where in the input space we are want $p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi}) \sim p_{\text{GP}}(\boldsymbol{f}|\boldsymbol{\theta})$

$$\mathcal{L}_{\mathbf{X}}(\boldsymbol{\phi}) \equiv \mathbb{E}_{\mathbf{X} \sim p(\mathbf{X})}[\mathcal{K}_{\mathbf{X}}(\boldsymbol{\phi})] \propto -\frac{1}{S} \sum_{s=1}^{S} \mathbb{E}_{p(\mathbf{X})}[\log p_{\text{GP}}(\boldsymbol{f}^{(s)}(\mathbf{x})|\boldsymbol{\theta})] \tag{3}$$

We optimize (3) until convergence $\boldsymbol{\phi}^* = \underset{\boldsymbol{\phi}}{\text{argmin}}\, \mathcal{L}_{\mathbf{X}}(\boldsymbol{\phi})$. This is described in detail in Algorithm 1.

2.2 Step 2. Optimize the ELBO

Next, we use $\boldsymbol{\phi}^*$ found from step 1 to form a prior on the weights $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}^*_{\boldsymbol{\phi}}, \boldsymbol{\sigma}^*_{\boldsymbol{\phi}}) \equiv q(\mathbf{w}|\boldsymbol{\phi}^*)$. Thereafter we learn the parameters $\boldsymbol{\varphi} = \{\boldsymbol{\mu}_{\boldsymbol{\varphi}}, \log \boldsymbol{\sigma}_\varphi\}$ of the variational approximation $q(\mathbf{w}|\boldsymbol{\varphi}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_{\boldsymbol{\varphi}}, \boldsymbol{\sigma}_\varphi)$ to the true posterior on the weights $p(\mathbf{w}|\mathcal{D})$. To do this we optimize the evidence lower bound (ELBO) $\mathcal{L}(\boldsymbol{\varphi})$ with our optimized prior $p(\mathbf{w}) = q(\mathbf{w}|\boldsymbol{\phi}^*)$

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\varphi}) = \mathbb{E}_{q(\mathbf{w}|\boldsymbol{\varphi})}\left[\log p(\mathcal{D}|\mathbf{w})\right] + \mathbb{KL}[q(\mathbf{w}|\boldsymbol{\varphi})||q(\mathbf{w}|\boldsymbol{\phi}^*)] \tag{4}$$

$$= -\mathbb{H}[q(\mathbf{w}|\boldsymbol{\varphi})] + \mathbb{E}_{q(\mathbf{w}|\boldsymbol{\varphi})}\left[\log p(\mathcal{D}|\mathbf{w})\right] - \mathbb{E}_{q(\mathbf{w}|\boldsymbol{\varphi})}\left[\log q(\mathbf{w}|\boldsymbol{\phi}^*)\right] \tag{5}$$

$$\approx -\log|\boldsymbol{\sigma}_\varphi| + \frac{1}{L} \sum_{\ell=1}^{L} \left[\log p(\mathcal{D}|\mathbf{w}^{(\ell)}) - \log q(\mathbf{w}^{(\ell)}|\boldsymbol{\phi}^*)\right] \quad \text{where } \mathbf{w}^{(\ell)} \sim q(\mathbf{w}|\boldsymbol{\varphi}) \tag{6}$$

We do this by sampling from a deterministic function of the variational parameters $\boldsymbol{\varphi}$ and noise variables $\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon}) = \mathcal{N}(\mathbf{0}, \mathbb{I})$ such that $\mathbf{w}^{(\ell)} = g(\boldsymbol{\varphi}, \boldsymbol{\epsilon}) = \boldsymbol{\mu}_{\boldsymbol{\varphi}} + \boldsymbol{\sigma}_{\boldsymbol{\varphi}} \boldsymbol{\epsilon}^{(\ell)}$. Thus we can obtain unbiased stochastic gradients of the ELBO with respect to the variational parameters. We use adam (Kingma and Ba 2015) to optimize (6) and (3).

3 Experiments and results

We test our model on 3 different toy problems and compare to a Bayesian neural network using a standard normal prior distribution on the weights trained with bayes by backprop (Blundell et al 2015). For the Gaussian process prior, we have $p_{\text{GP}}(\boldsymbol{f}|\boldsymbol{\theta}) \sim \mathcal{N}(\boldsymbol{f}|\mathbf{0}, \mathbf{K})$, where $\mathbf{K}$ is determine by the rbf covariance function. For each row the toy data is generated by sampling from the uniform distribution over some region, then passing it through some function and adding some Gaussian noise. For row $1, 2, 3$: we have $y_1 = e^{-x^2/2} + \mathcal{N}(0, 0.01), y_2 = \cos(x/4 - 1) + \mathcal{N}(0, 0.01), y_3 = 0.1x \sin x + \mathcal{N}(0, 0.01)$.

The resultant plots are displayed below in figure 2.

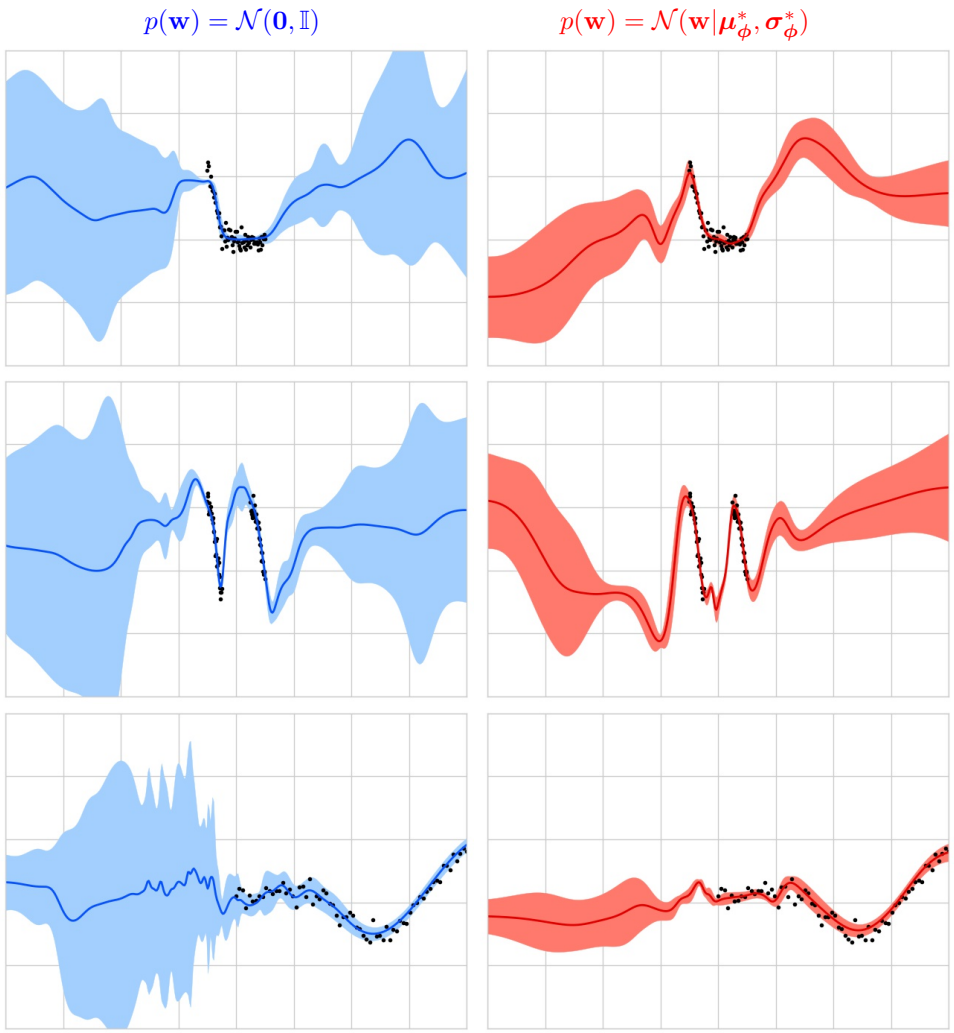


Figure 2: The left column of plots are Bayesian neural networks trained with a standard normal prior distribution on the weights. The right column are Bayesian neural nets trained using the optimized prior found by doing step 1 in section 2.1. The dots are samples from the data distribution. The dark lines are the means of the posteriors and the shaded regions are 95% confidence intervals.

3.1 Results

In these toy cases, using smooth, infinitely differentiable functions to generate the datasets, the rbf kernel assumption proves to be a good one. Notice that the BNNs trained using the GP matched prior fit the data better and have more reasonable uncertainty envelopes.

Acknowledgements

The authors would like to thank Brian Ning and Guodong Zhang for helpful comments.

References

Blundell, Charles, Cornebise, Julien, Kavukcuoglu, Koray, and Wierstra, Daan. Weight uncertainty in neural networks.*arXiv preprint arXiv:1505.05424, 2015.*

Danilo J Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning* (ICML-14), pages 1278–1286, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114, 2013.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization.*arXiv preprint arXiv:1412.6980, 2014*

Dougal Maclaurin, David Duvenaud, Matthew Johnson, and Ryan P. Adams. Autograd: Reversemode differentiation of native Python, 2015.

Soumya Ghosh, Finale Doshi-Velez Model Selection in Bayesian Neural Networks via Horseshoe Priors *arXiv preprint arXiv:1705.10388, 2016*