Mapping Gaussian Process Priors to Bayesian Neural Networks

Daniel Flam-Shepherd
University of Toronto
danielfs@utstat.utoronto.ca

James Requeima
University of Cambridge
jrr41@cam.ac.ukl

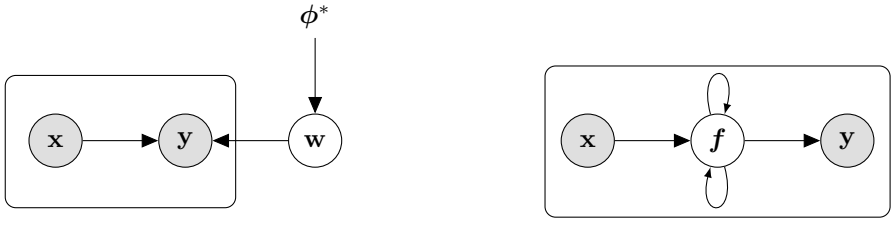
David Duvenaud
University of Toronto
duvenaud@cs.toronto.edu

1 Introduction and Motivation

What defines a reasonable prior to use when forming and training Bayesian models? Recent work, such as [1] seek to improve priors over network parameters, by applying a horseshoe prior over preactivations of a Bayesian neural network (BNN) to effectively turn off weights that do not explain the data. Our goal with this work is to consider priors in function space as well.

It is difficult to incorporate meaningful prior information about functions to be modeled 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 Processes 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 $p_{\text{GP}}(\boldsymbol{f})$ by minimizing their approximate KL divergence via some data distribution of interest $\mathbf{X} \sim p(\mathbf{X})$. We minimize this 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 $p(\mathbf{w}|\boldsymbol{\phi}^*) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}^*_{\boldsymbol{\phi}}, \boldsymbol{\sigma}^*_{\boldsymbol{\phi}})$. Then, variational inference allows us to perform approximate inference in our BNN using this more principled prior. We describe the implementation of both steps and demonstrate its success in the following sections.



(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 Mapping a GP prior to BNN prior

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 $p_{\text{GP}}(\boldsymbol{f}) \equiv \mathcal{GP}(\boldsymbol{f}|\mathbf{0}, \mathbf{K})$ where $\mathbf{K}$ is the Grahm matrix of the kernel or covariance function chosen.

$$\mathcal{K}(\boldsymbol{\phi}) = \mathbb{KL}[p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi}) \mid p_{\text{GP}}(\boldsymbol{f})] \tag{1}$$

$$= \int p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi}) \log \left[\frac{p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})}{p_{\text{GP}}(\boldsymbol{f})} \right] d\boldsymbol{f} \tag{2}$$

$$= -\mathbb{H}[p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})] - \mathbb{E}_{p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})}[\log p_{\text{GP}}(\boldsymbol{f})] \tag{3}$$

We define our first stochastic optimization objective $\mathcal{L}_{\mathbf{X}}(\boldsymbol{\phi})$ by approximating the KL divergence between these infinite dimensional distributions by taking expectations over some data distribution $p(\mathbf{X})$, explicitly : $\mathcal{L}_{\mathbf{X}}(\boldsymbol{\phi}) \equiv \mathbb{E}_{\mathbf{X}\sim p(\mathbf{X})}[\mathcal{K}(\boldsymbol{\phi})]$. This allows us to prioritize where in the input space we want $p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi}) \sim p_{\text{GP}}(\boldsymbol{f})$. We have :

$$\mathcal{L}_{\mathbf{X}}(\boldsymbol{\phi}) = -\mathbb{E}_{\mathbf{X}\sim p(\mathbf{X})}\mathbb{H}[p_{\text{BNN}}(\boldsymbol{f}(\mathbf{X})|\boldsymbol{\phi})] - \mathbb{E}_{\mathbf{X}\sim p(\mathbf{X})}\mathbb{E}_{p_{\text{BNN}}(\boldsymbol{f}(\mathbf{X})|\boldsymbol{\phi})}[\log p_{\text{GP}}(\boldsymbol{f}(\mathbf{X}))] \tag{4}$$

Both terms in equation (4) are intractable and force us to use some approximation. Since we are able to compute $p_{\text{GP}}(\boldsymbol{f}(\mathbf{X}))$ analytically, we can compute a Monte Carlo Estimate of the second term in (4):

$$\mathbb{E}_{\mathbf{X}\sim p(\mathbf{X})}\mathbb{E}_{p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})}[\log p_{\text{GP}}(\boldsymbol{f})] \approx \frac{1}{S}\sum_{s=1}^{S} \mathbb{E}[\log p_{\text{GP}}(\boldsymbol{f}^{(s)}(\mathbf{X}))]$$

where $\boldsymbol{f}^{(s)}(\mathbf{X}) \sim p_{\text{BNN}}(\boldsymbol{f}|\mathbf{X})$. For the entropy term in (4) we propose two estimation methods.

2.1 Estimate of entropy term using moment matching

The first estimate we make involves the approximation that the prior distribution over functions for the BNN is a multivariate Gaussian distribution that takes form $p_{\text{BNN}}(\boldsymbol{f}(\mathbf{X})|\boldsymbol{\phi}) \sim \mathcal{N}(\boldsymbol{f}|\boldsymbol{\mu}_{\boldsymbol{f}(\mathbf{X})},\ \boldsymbol{\Sigma}_{\boldsymbol{f}(\mathbf{X})})$. In this case, the entropy term can be evaluated analytically over $p(\mathbf{X})$ as $\mathbb{H}[p_{\text{BNN}}(\boldsymbol{f}(\mathbf{X})|\boldsymbol{\phi})] = \frac{1}{2}\log|2\pi e\boldsymbol{\Sigma}_{\boldsymbol{f}(\mathbf{X})}|$ where we use the sample mean and covariance for $\boldsymbol{\mu}_{\boldsymbol{f}(\mathbf{X})},\ \boldsymbol{\Sigma}_{\boldsymbol{f}(\mathbf{X})}$ over $p(\mathbf{X})$. This is a reasonable approximation because the overall goal in this work is push $p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi}) \approx p_{\text{GP}}(\boldsymbol{f})$ over $p(\mathbf{X})$ and as $p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi}) \to p_{\text{GP}}(\boldsymbol{f})$ it will get more accurate. With this, the objective is

$$\mathcal{L}_{\mathbf{X}}(\boldsymbol{\phi}) \approx -\frac{1}{2}\log|\boldsymbol{\Sigma}_{\boldsymbol{f}(\mathbf{X})}| - \frac{1}{S}\sum_{s=1}^{S} \mathbb{E}_{\mathbf{X}\sim p(\mathbf{X})}[\log p_{\text{GP}}(\boldsymbol{f}^{(s)}(\mathbf{X}))] \tag{5}$$

We optimize this until convergence $\boldsymbol{\phi}^* = \underset{\boldsymbol{\phi}}{\text{argmin}}\ \mathcal{L}_{\mathbf{X}}(\boldsymbol{\phi})$. The details are in Algorithm 1.

Algorithm 1 Optimization of the prior of the Bayesian neural net

Initialize $\boldsymbol{\phi} = \{\boldsymbol{\mu}_{\boldsymbol{\phi}}, \log \boldsymbol{\sigma}_{\boldsymbol{\phi}}\}$
while $\boldsymbol{\phi}$ not converged **do**
  $\boldsymbol{\epsilon}^{(s)} \sim p(\boldsymbol{\epsilon}) = \mathcal{N}(\mathbf{0}, \mathbb{I})$ ▷ sample prior noise
  $\mathbf{w}^{(s)} \leftarrow t(\boldsymbol{\phi}, \boldsymbol{\epsilon}) = \boldsymbol{\mu}_{\boldsymbol{\phi}} + \boldsymbol{\sigma}_{\boldsymbol{\phi}} \odot \boldsymbol{\epsilon}^{(s)}$ ▷ 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
  $\boldsymbol{f}^{(s)}(\mathbf{X}) \leftarrow f(\mathbf{X}, \mathbf{w}^{(s)})$ ▷ sample S functions $\boldsymbol{f} \sim p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})$
  $\mathbf{g}_{\boldsymbol{\phi}} \leftarrow \nabla_{\boldsymbol{\phi}}\mathcal{L}_{\mathbf{X}}(\boldsymbol{\phi})$ ▷ compute gradients of the objective
  $\boldsymbol{\phi} \leftarrow \text{adam}(\boldsymbol{\phi}, \mathbf{g}_{\boldsymbol{\phi}})$ ▷ update the parameters
Return $\boldsymbol{\phi}^*$

The next section describes a formulation that allows to make an alternate approximation of the entropy term by considering the equivalent problem in observation space.

2.2 Estimate of entropy term by projecting into observation space

Consider the equivalent problem in observation space by adding noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbb{I})$ to our functions $\mathbf{y} = \boldsymbol{f} + \boldsymbol{\epsilon}$ so that we are minimizing the KL divergence of $p(\mathbf{y}|\boldsymbol{\phi})$: the BNN prior distribution over noisy observations to the GP prior distribution over noisy observations $p_{\text{GP}}(\mathbf{y})$.

$$\mathcal{K}(\boldsymbol{\phi}) = \mathbb{KL}[p(\mathbf{y}|\boldsymbol{\phi}) \mid p_{\text{GP}}(\mathbf{y})] = -\mathbb{H}[p(\mathbf{y}|\boldsymbol{\phi})] - \mathbb{E}_{p(\mathbf{y}|\boldsymbol{\phi})}[\log p_{\text{GP}}(\mathbf{y})] \tag{6}$$

Now we use Monte Carlo estimates of both $\mathbb{H}[p(\mathbf{y}|\boldsymbol{\phi})]$ and $\mathbb{E}_{p(\mathbf{y}|\boldsymbol{\phi})}[\log p_{\text{GP}}(\mathbf{y}|\boldsymbol{\theta})]$:

$$\mathcal{K}(\boldsymbol{\phi}) \approx \frac{1}{S}\sum_{s=1}^{S} \log p(\mathbf{y}^{(s)}|\boldsymbol{\phi}) - \frac{1}{S}\sum_{s=1}^{S} \log p_{\text{GP}}(\mathbf{y}^{(s)}) \tag{7}$$

where we are sampling S times : $\mathbf{y}^{(s)} \sim p(\mathbf{y}|\boldsymbol{\phi})$. To estimate $\log p(\mathbf{y}^{(s)}|\boldsymbol{\phi})$ we do approximate inference on an intermediate proposal distribution $r(\mathbf{w}|\boldsymbol{\lambda})$, where $\boldsymbol{\lambda} = \{\boldsymbol{\mu}_{\boldsymbol{\lambda}}, \log \boldsymbol{\sigma}_{\boldsymbol{\lambda}}\}$ are its variational parameters to be optimized. We then estimate $\log p(\mathbf{y}^{(s)}|\boldsymbol{\phi})$ using its ELBO $\mathcal{L}_{\mathcal{D}_s}(\boldsymbol{\lambda})$, optimized on data $\mathcal{D}_s = \{\mathbf{X}, \mathbf{y}^{(s)}\}$. This is similar to Hierarchical variational models by [2].

$$\log p(\mathbf{y}^{(s)}|\boldsymbol{\phi}) \geq \mathcal{L}_{\mathcal{D}_s}(\boldsymbol{\lambda}) \equiv \mathbb{E}_{r(\mathbf{w}|\boldsymbol{\lambda})}\left[\log p(\mathcal{D}_s|\mathbf{w}) + \log p(\mathbf{w}) - \log r(\mathbf{w}|\boldsymbol{\lambda})\right] \tag{8}$$

where $\log p(\mathcal{D}_s|\mathbf{w}) = \sum_i \log \mathcal{N}(\mathbf{y}_i^{(s)}|f(\mathbf{x}_i, \mathbf{w}), \sigma^2\mathbb{I})$ defines a parametric model. Once again we take expectations $p(\mathbf{X})$ to define the objective $\mathcal{L}_{\mathbf{X}}(\boldsymbol{\phi}, \boldsymbol{\lambda}) \equiv \mathbb{E}_{\mathbf{X}\sim p(\mathbf{X})}[\mathcal{K}(\boldsymbol{\phi})]$ by approximating the KL divergence between these infinite dimensional distributions. We obtain :

$$\mathcal{L}_{\mathbf{X}}(\boldsymbol{\phi}, \boldsymbol{\lambda}) = \min_{\boldsymbol{\phi}} \frac{1}{S}\sum_{s=1}^{S} \max_{\boldsymbol{\lambda}} \mathcal{L}_{\mathcal{D}_s}(\boldsymbol{\lambda}) - \frac{1}{S}\sum_{s=1}^{S} \mathbb{E}_{\mathbf{X}\sim p(\mathbf{X})}[\log p_{\text{GP}}(\mathbf{y}^{(s)}(\mathbf{X}))] \tag{9}$$

We optimize (9) till convergence, this is described in the Algorithm 2.

Algorithm 2 Mapping a GP prior to a BNN prior in observation space

Initialize $\boldsymbol{\phi} = \{\boldsymbol{\mu}_{\boldsymbol{\phi}}, \log \boldsymbol{\sigma}_{\boldsymbol{\phi}}\}$
while $\boldsymbol{\phi}$ not converged **do**
  $\boldsymbol{\epsilon}^{(s)} \sim p(\boldsymbol{\epsilon}) = \mathcal{N}(\mathbf{0}, \mathbb{I})$ ▷ sample prior noise
  $\mathbf{w}^{(s)} \leftarrow \boldsymbol{t}(\boldsymbol{\phi}, \boldsymbol{\epsilon}^{(s)}) = \boldsymbol{\mu}_{\boldsymbol{\phi}} + \boldsymbol{\sigma}_{\boldsymbol{\phi}}\boldsymbol{\epsilon}^{(s)}$ ▷ 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 from some data distribution
  $\boldsymbol{f}^{(s)} \leftarrow f(\mathbf{X}, \mathbf{w}^{(s)})$ ▷ sample S functions from the BNN
  $\mathbf{y}^{(s)} \leftarrow \boldsymbol{f}^{(s)}(\mathbf{X}) + \boldsymbol{\epsilon}$ ▷ sample observations
  for `each` $\mathcal{D}_s = \{\mathbf{X}, \mathbf{y}^{(s)}\}$ `hold` $\boldsymbol{\phi}$ `constant and` **do**
    $\boldsymbol{\lambda}_s^* \leftarrow \underset{\boldsymbol{\lambda}}{\text{argmax}} \mathcal{L}_{\mathcal{D}_s}(\boldsymbol{\lambda})$ ▷ do approximate inference
    $\log p(\mathbf{y}^{(s)}) \leftarrow \mathcal{L}_{\mathcal{D}_s}(\boldsymbol{\lambda}_s^*)$ ▷ estimate the marginal log likelihood
  $\mathbf{g}_{\boldsymbol{\phi}} \leftarrow \nabla_{\boldsymbol{\phi}}\mathcal{L}_{\mathbf{X}}(\boldsymbol{\phi}, \boldsymbol{\lambda})$ ▷ compute gradients of the kl
  $\boldsymbol{\phi} \leftarrow \text{adam}(\boldsymbol{\phi}, \mathbf{g}_{\boldsymbol{\phi}})$ ▷ update the parameters
Return $\boldsymbol{\phi}^*$

2.3 Early Stopping

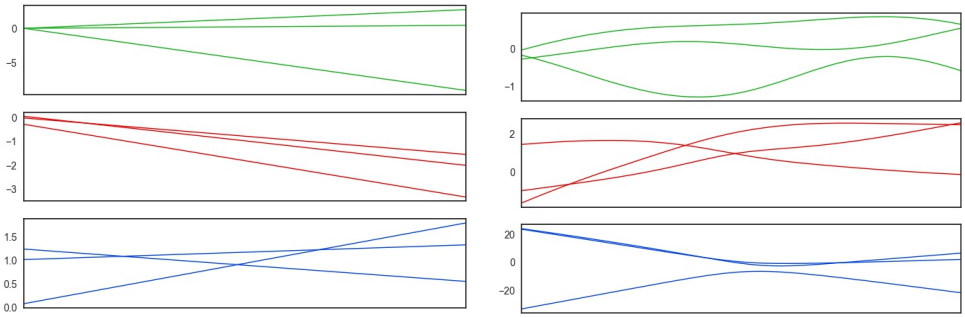
Another rough approximation of (4) is to ignore the entropy term $\mathbb{H}[p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})]$ altogether, as it merely acts as a regularization term, so we avoid collapsing to a single high density value. To address this issue, we found that early stopping works well in practice and yielded our best results. The results presented in the following section use early stopping and no entropy term.

3 Experiments with different kernels and activation functions

We experiment with mapping GP priors (GPPs) with various covariance functions $k(x, x')$ to BNN priors that have various activation functions $a(x)$. We sample on order 10^3 data from a uniform data distribution $\mathbf{X} \sim p(\mathbf{X})$ about a symmetric interval (-10, 10). We use automatic differentiation software : `autograd` [3, 4] to compute gradients of our objective, which is optimized using adam [5]. In all experiments we work directly in function space and found that reasonable results can be found even when ignoring the entropy term $\mathbb{H}[p(\boldsymbol{f})]$ altogether. Experiments are conducted on simple 1 or 2 layer neural networks. The prior proposal distribution on the BNN weights is a diagonal Gaussian in all cases. In the following subsections, plots have color scheme:

$$\boldsymbol{f}(\mathbf{X}) \sim p_{\text{GP}}(\boldsymbol{f}(\boldsymbol{X})), \quad \boldsymbol{f}(\mathbf{X}) \sim p_{\text{BNN}}(\boldsymbol{f}(\boldsymbol{X})|\boldsymbol{\phi}^*), \quad \boldsymbol{f}(\mathbf{X}) \sim p_{\text{BNN}}(\boldsymbol{f}(\boldsymbol{X}))$$

3.1 Linear and Softplus Experiments



$$k(x, x') = xx'$$
$$a(x) = x$$

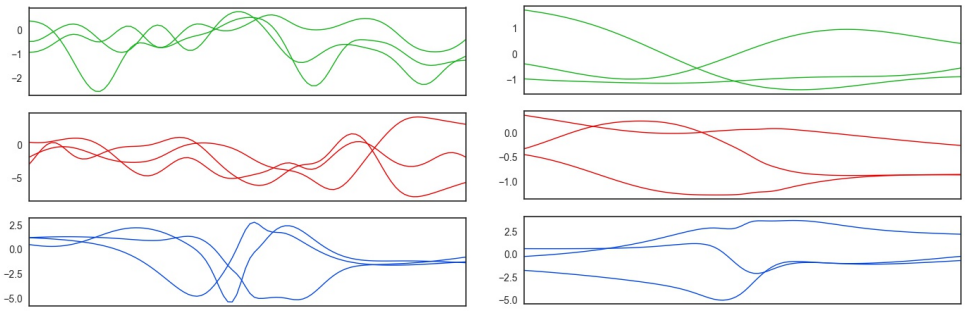
(a)

$$k(x, x') = e^{-(x-x')^2/50}$$
$$a(x) = \log(1 + e^x)$$

(b)

Figure 2: In (a) we map a GPP with a linear kernel to a linear model (Bayesian Linear regression), the optimized prior samples extend from the origin just as the GPP samples do. In (b) we map a GP prior with a RBF kernel to a BNN prior with a softplus activation function, the optimized samples have been mapped near the range of the GPP samples and almost as curvy as the GPP samples.

3.2 Experiments with rbf hyperbolic tan activations



$$k(x, x') = e^{-(x-x')^2/2}$$
$$a(x) = e^{-x^2}$$

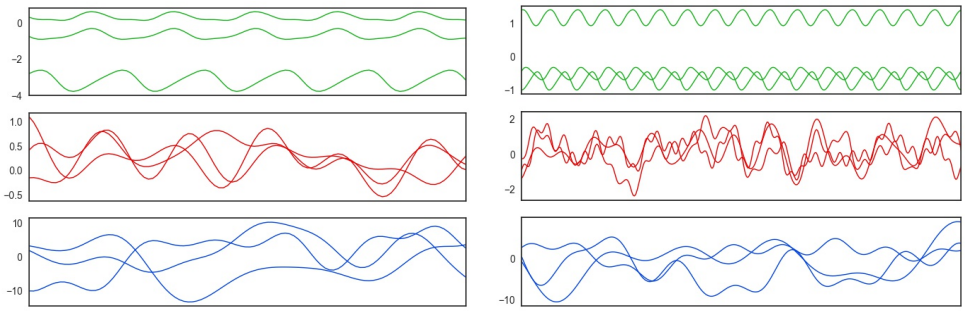
(a)

$$k(x, x') = e^{-(x-x')^2/50}$$
$$a(x) = \tanh x$$

(b)

Figure 3: In (a) and (b) we map GP priors with a RBF kernel to BNNs with a RBF and tanh activation functions It is clear that the optimized samples have taken on the curveness of the GPP samples

3.3 Experiments with periodic kernels and activations



$$k(x, x') = e^{-2\sin^2(\pi|x-x'|)}$$
$$a(x) = \sin x$$

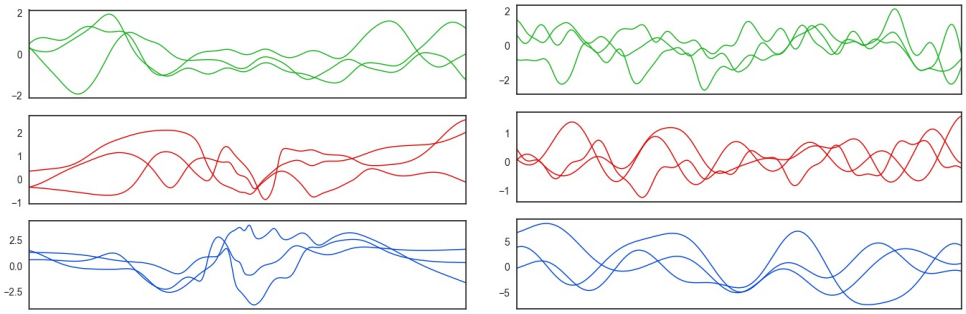
(a)

$$k(x, x') = e^{-2\sin^2(\pi|x-x'|)/25}$$
$$a(x) = \sin x$$

(b)

Figure 4: In (a) and (b) we map GPPs with periodic kernels to BNNs with sin activation functions. The periodic structure of the GPP samples has been mapped to the optimized BNN samples.

3.4 Experiments with other kernels



$$k(x, x') = (1 + (x - x')^2/6)^{-3}$$
$$a(x) = e^{-x^2}$$

(a)

$$k(x, x') = e^{-2\sin^2(\pi|x-x'|)/25} e^{-(x-x')^2/2}$$
$$a(x) = \sin x$$

(b)

Figure 5: In (a) we map a GPP with a rational cubic kernel to a BNN prior, the optimized samples have taken on the "wigglyness" of the prior samples especially away from the origin (the middle). In (b) we map a GPP with a locally periodic kernel to a BNN prior, the optimized samples take on both the "local wigglyness" and the periodicity of the functions from the GPP

4 Inference using the optimized prior

Next we use the optimized parameters $\boldsymbol{\phi}^*$ found by minimizing the kl of the $p_{\text{BNN}}(\boldsymbol{f}|\boldsymbol{\phi})$ to $p_{\text{GP}}(\boldsymbol{f})$ in a prior on the weights $p(\mathbf{w}|\boldsymbol{\phi}^*) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}^*_{\boldsymbol{\phi}}, \boldsymbol{\sigma}^*_{\boldsymbol{\phi}})$. when maximizing the ELBO $\mathcal{L}_{\mathcal{D}}(\boldsymbol{\varphi}) \leq \log p(\mathcal{D})$.

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\varphi}) = \mathbb{H}[q(\mathbf{w}|\boldsymbol{\varphi})] + \mathbb{E}_{q(\mathbf{w}|\boldsymbol{\varphi})}\left[\log p(\mathcal{D}|\mathbf{w}) + \log p(\mathbf{w}|\boldsymbol{\phi}^*)\right] \tag{10}$$

Where $\mathcal{D} = \{\mathbf{y}, \mathbf{X}\}$ is the training data, $\boldsymbol{\varphi} = \{\boldsymbol{\mu}_{\boldsymbol{\varphi}}, \log \boldsymbol{\sigma}_{\varphi}\}$ are the variational parameters of the approximation $q(\mathbf{w}|\boldsymbol{\varphi}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_{\boldsymbol{\varphi}}, \boldsymbol{\sigma}_{\varphi})$ to the true posterior on the weights $p(\mathbf{w}|\mathcal{D})$. Which we can sample from using reparameterization trick [6, 7] $\mathbf{w}^{(\ell)} = \boldsymbol{t}(\boldsymbol{\varphi}, \boldsymbol{\epsilon}) = \boldsymbol{\mu}_{\boldsymbol{\varphi}} + \boldsymbol{\sigma}_{\boldsymbol{\varphi}}\boldsymbol{\epsilon}^{(\ell)}$. Thus we can obtain unbiased stochastic gradients of the ELBO with respect to the variational parameters.

4.1 Experiments on toy data

We test our model (blue) on 2 different toy problems. We work directly in function space and use algorithm one, and we found that no entropy term estimate was necessary for reasonable results. We compare to a BNN (blue) using a standard normal prior distribution on the weights and to the posterior of a Gaussian process (green). The BNNs are trained using Bayes by Backprop [8]. For both Gaussian process priors, the kernel is the RBF covariance function. The observations y are sampled by passing some data sampled uniformly through a function $f(x)$ and adding some Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$. We use $\sigma = 1$. The networks are all 2 layer with rbf activations as well. In the following plots the previous color scheme also applies

$$\boldsymbol{f}(\mathbf{X}) \sim p_{\text{GP}}(\boldsymbol{f}|\mathcal{D}), \quad \boldsymbol{f}(\mathbf{X}) \sim p_{\text{BNN}}(\boldsymbol{f}(\boldsymbol{X})|\varphi^*, \boldsymbol{\phi}^*), \quad \boldsymbol{f}(\mathbf{X}) \sim p_{\text{BNN}}(\boldsymbol{f}(\boldsymbol{X})|\varphi^*)$$

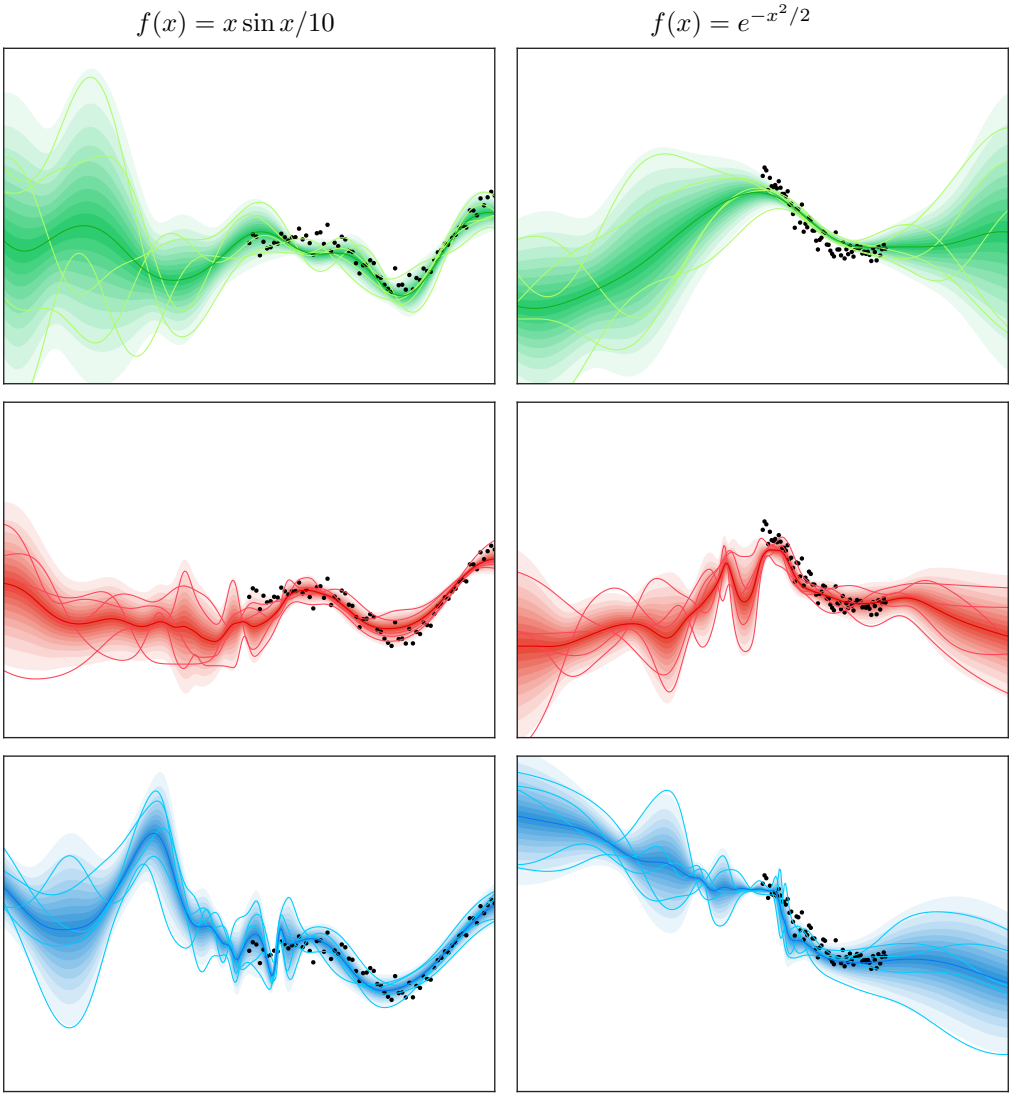


Figure 6: Plots of the 3 posteriors for 2 different functions. Different shades of color correspond to deciles of the predictive density $p(f(x_i))$. The samples from this density are a slightly different shade of green, red, and blue. The darkest is the mean. Notice that the mean and uncertainty bands of the posterior trained with the optimized prior closer resembles the uncertainty of the Gaussian process. This demonstrates that our method of approximating a GP prior via a BNN can successfully transfer characteristics of the GP posterior to the BNN posterior as well.

5 Limitations

In the previous sections we demonstrate the success of our proposed objective in completing the set out task of mapping the properties of a Gaussian process prior to a Bayesian neural network prior. However, there remain significant limitations of our method. We describe them below.

We found that we could not map GP priors to BNNs with certain activation functions, for example we could only map a GP prior with a linear kernel to a BNN with Relu activation functions at any capacity. In this case, the BNN could not, at any capacity, learn the smoothness or periodic properties of any more interesting GP prior.

As well, we could not map the properties of a GP prior with a periodic kernel to a BNN that did not have a periodic activation function. Our experiments focused on mapping properties of GP priors to BNNs with configurations that could reasonably learn those properties, for example periodic kernels to BNN priors with periodic activation functions.

We did not find an accurate estimator of the entropy term in our objective, as a result, training BNN priors to behave like GP priors was difficult and was negatively hampered by lack of a accurate regularization term. Furthermore, rough estimates of the entropy term did not stabilize or improve these difficulties. Also, working in observation space with a more accurate estimation of the entropy term introduced another set of difficulties brought on by the mini max objective. As a consequence, we found that ignoring the entropy term and performing early stopping worked best.

Often, without the entropy term, samples from the optimized BNN prior would take on the properties of the GP prior then gradually flatten and lose the desired properties. On the other hand, with the entropy term the samples would sometimes blow up. There was no stable regime where we could obtain consistent convergence. This would sometimes result in optimized samples which over emphasize the characteristics of the GP prior. An interesting example of this is displayed in Figure 7 where the red optimized samples take on an extremely local periodic behaviour later in training, while the samples of the GP prior being mapped do not have such a extreme periodic behavior.

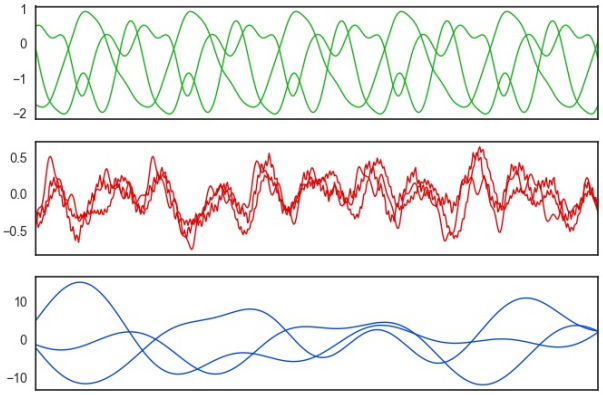


Figure 7: An example of extreme overtraining

6 Conclusions and Future Work

In this work, we formulate a method for mapping a Gaussian process prior to a Bayesian neural network prior. This is done in effort to transfer interesting properties of functions sampled from a GP prior to functions sampled from a Bayesian neural network prior. In this regard, we can implicitly think about priors in BNNs in function space rather than parameter space and overcome the limits that this entails. We demonstrate our method on a variety of kernels and activation functions, and gage its impact on posterior inference in simple toy examples. All together, while there are important described limitations we establish that our method works throughout these demonstrations and examples.

This work attempts to build a small bridge between the flexible models of Bayesian non parametrics and the more scalable models of Bayesian deep learning. In future we seek to continue to build upon this idea of specifying priors in function space. The goal is to building more expressive Bayesian models where the prior better captures necessary prior beliefs and is more useful for posterior inference.

Acknowledgements

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

References

[1] Soumya Ghosh and Finale Doshi-Velez. Model selection in bayesian neural networks via horseshoe priors. In *Neural Information Processing Systems*, 2016.

[2] Rajesh Ranganath, Dustin Tran, and David M. Blei. Hierarchical variational models. *International Conference on Machine Learning*, 2016.

[3] Dougal Maclaurin, David Duvenaud, Matthew Johnson, and Ryan P. Adams. Autograd: Reverse-mode differentiation of native python. 2015.

[4] David Duvenaud and Ryan P. Adams. Black-box stochastic variational inference in five lines of python. *NIPS Workshop on Black-box Learning and Inference*, 2015.

[5] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations*, 2015.

[6] D. P Kingma and M. Welling. Auto-Encoding Variational Bayes. *International Conference on Learning Representations*, 2014.

[7] Danilo J Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *International Conference on Machine Learning*, 2014.

[8] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. *International Conference on Machine Learning*, 2015.