Weight Uncertainty in Neural Networks

UNIVERSITY OF CAMBRIDGE

Tennison Liu, Chelsea Murray, Eli Persky

Introduction

Traditional neural networks use **point estimates** of the weights	Bayesian Neural Networks (BNNs) use **posterior distribution** over weights
Poor calibration	Uncertainty estimates
No uncertainty estimates	Regularisation
Poor generalisation	Exploration in RL

Exact Bayesian inference over a neural network is intractable.

- Inference: use **variational approximation** to the posterior.
- Prediction: **ensemble of networks** by repeatedly sampling weights.

Approximate Inference

Bayes-by-Backprop (BBB) [1] objective function:

$$\mathcal{F}(\mathcal{D}, \theta) = \mathrm{KL}\left[q(\mathbf{w}|\theta)||P(\mathbf{w})\right] - \mathbb{E}_{q(\mathbf{w}|\theta)}\left[\log P(\mathcal{D}|\mathbf{w})\right]$$

$$\approx \sum_{i=1}^{n} \log q(\mathbf{w}^{(i)}|\theta) - \log P(\mathbf{w}^{(i)}) - \log P(\mathcal{D}|\mathbf{w}^{(i)})$$

We explore single Gaussian and Mixture of Gaussian priors.
Variational posterior $q(\mathbf{w}|\theta)$ is Gaussian, sampled using reparameterisation:

$$\mathbf{w} = \mu + \sigma \circ \epsilon \quad \text{where} \quad \epsilon \sim \mathcal{N}(0, I)$$

Or sample activations b conditioned on inputs a and weights $\mathbf{w}$ using **local reparameterisation trick (LR Trick)** [2]:

$$b_{m,j} = \gamma_{m,j} + \sqrt{\delta_{m,j}}\zeta_{m,j} \qquad \text{where} \quad \zeta \sim \mathcal{N}(0,1)$$

$$\text{with } \gamma_{m,j} = \sum_i a_{m,i}\mu_{i,j}, \qquad \delta_{m,j} = \sum_i a_{m,i}^2 \sigma_{i,j}^2$$

Computationally efficient, decreases variance of gradient estimates leading to faster convergence.

Monte Carlo (MC) Dropout [3] Bayesian interpretation of dropout i.e. draw samples at test time by repeatedly masking random weights.

Functional Variational Inference (FVI) [4] optimisation against distributions over functions with a Gaussian Process prior:

$$\mathcal{F}(D, \theta) = \mathrm{KL}[q(\mathbf{f}|\theta)||P(\mathbf{f})] - \mathbb{E}_{q(\mathbf{f}|\theta)}[\log P(\mathcal{D}|\mathbf{f})]$$

$$\text{with } \; P(\mathbf{f}) \sim \mathcal{GP}(0, K_L + K_{RBF})$$

$$q(\mathbf{f}|\theta) \quad \text{NN with Gaussian weights}$$

Classification

	SGD	SGD Dropout	MC Dropout	BBB Gaussian	BBB Mixture	LR Trick
Error (%)	1.90	1.26	1.58	**1.22**	**1.16**	1.35

Table 1. Results for MNIST classification. Models trained for 300 epochs using 10 samples in training (for BNNs).

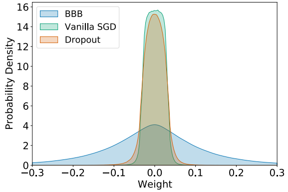

Figure 1. Histograms of trained weights for SGD, SGD dropout and sampled weights from BBB.

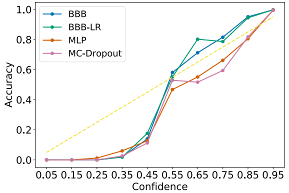

Figure 2. Calibration curves. Bayesian models **avoid overconfidence.**

Removed (%)	0	50	75	95	98	100
# Weights	2.4M	1.2M	600k	120k	48k	0
Error (%)	1.29	**1.28**	1.33	1.58	1.66	89.71

Table 2. Classification accuracy in BNN after pruning weights with the lowest Signal-to-Noise ratio.

- BNNs achieve **superior performance** and **improved calibration** over regularisation methods such as dropout and MC dropout.
- The Bayesian approach provides a **principled method for pruning** the network. Weights with a low Signal-to-Noise ratio in the posterior distribution can be masked out with minimal effect on performance.

Reinforcement Learning

- UCI Mushroom Bandit: agent selects action (eat vs. not eat) with highest reward.
- Using Thompson Sampling, BBB naturally balances **exploration vs. exploitation.**

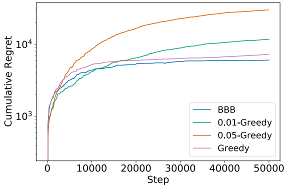

Figure 3. Cumulative regret. BBB agent achieves **flat regret** early.

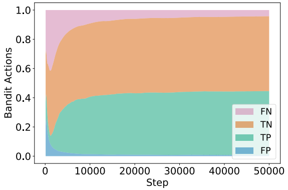

Figure 4. Cumulative decisions. BBB converges to **optimal decisions.**

Regression

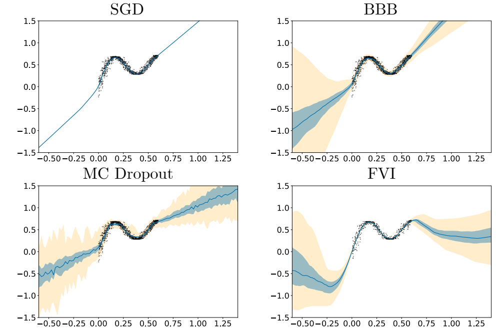

Figure 5. Regression on synthetic data. Blue line is the median prediction; blue and orange regions show interquartile range and range.

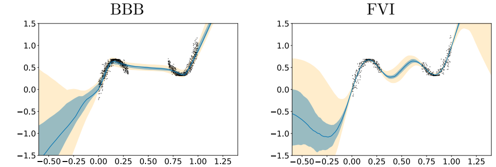

Figure 6. Regression with data clusters. FVI **handles uncertainty** between clusters.

Conclusions

- BNNs match and exceed the **performance** of other methods while providing **sensible uncertainty estimates** and **better calibration**.
- Training a BNN can be viewed as training an infinite **ensemble** on neural networks while only doubling the number of parameters.
- LR and FVI provide improvements in certain situations.

References

[1] Blundell, Charles, et al. "Weight uncertainty in neural network." *International Conference on Machine Learning*. PMLR, 2015.

[2] Kingma, Diederik P., Tim Salimans, and Max Welling. "Variational dropout and the local reparameterization trick." *arXiv preprint arXiv:1506.02557* (2015).

[3] Gal, Yarin, and Zoubin Ghahramani. "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning." *International Conference on Machine Learning*. PMLR, 2016.

[4] Sun, Shengyang, et al. "Functional variational Bayesian neural networks." arXiv preprint arXiv:1903.05779 (2019).