Weight Uncertainty in Neural Networks

Alexander J. Chan, Andrew Lee, Wen Wu

UNIVERSITY OF CAMBRIDGE

Background

Point estimates for neural networks are not enough:

- No way to quantify uncertainty in predictions - results in overconfident predictions.
- Not robust - can be effectively fooled by adversarial examples.

Exact Bayesian inference completely intractable over weights:

- Functional form doesn't allow for analytic integration.
- Huge number of weights make numerical methods intractable too.

Solution:

- Propose a fast, backpropagation-style, algorithm for learning an approximate posterior distribution over the weights.

Bayes by Backprop

- Variational Bayesian paradigm replaces integration problem with optimisation task - leverage gradient methods and auto-diff.
- Make use of Monte Carlo approximations for training and predictions.

Approximate $P(w|D)$ by minimizing KL divergence:

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, KL[q(w|\theta)||P(w|D)]$$

Equivalently maximise the Evidence Lower BOund (ELBO):

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

Monte Carlo approximation:

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

$$w^{(i)} \sim q(w|\theta)$$

Gaussian variational posterior $q(w^{(i)}|\theta)$:

Reparameterisation trick:

$$w = \mu + \log(1+\exp(\rho)) \circ \epsilon$$

$$\epsilon \sim \mathcal{N}(0, I)$$

$$\theta = (\mu, \rho)$$

Scale mixture prior $P(w)$:

$$P(w) = \prod_j \pi\mathcal{N}(w_j|0,\sigma_1^2) + (1-\pi)\mathcal{N}(w_j|0,\sigma_2^2), \sigma_1 > \sigma_2, \sigma_2 \ll 1$$

- Only double the number of parameters yet trains an infinite ensemble

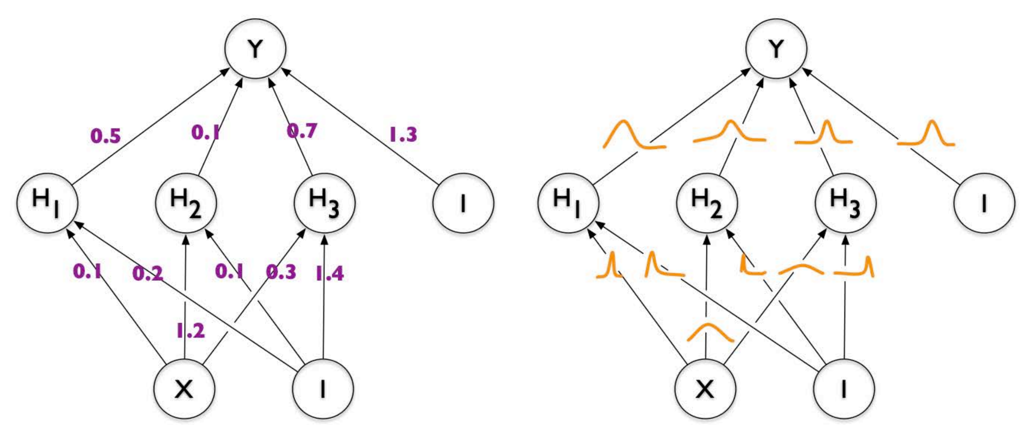

Figure 1. Left: classical BP, fixed value on weights. Right: BBB, distribution over weights. Image taken from [1].

1-D Regression: Visualising Uncertainty

- Simple regression task using Bayes by Backprop (BBB). We compare to predictions from a regular NN and MC Dropout NN, as well as a Gaussian process.
- Uncertainty estimates are quite conservative.
- Training done on 100 randomly sampled points from function with Gaussian noise:

$$y = x + 0.3\ \sin(2\pi(x+\epsilon)) + 0.3\sin(4\pi(x+\epsilon)) + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, 0.02)$$

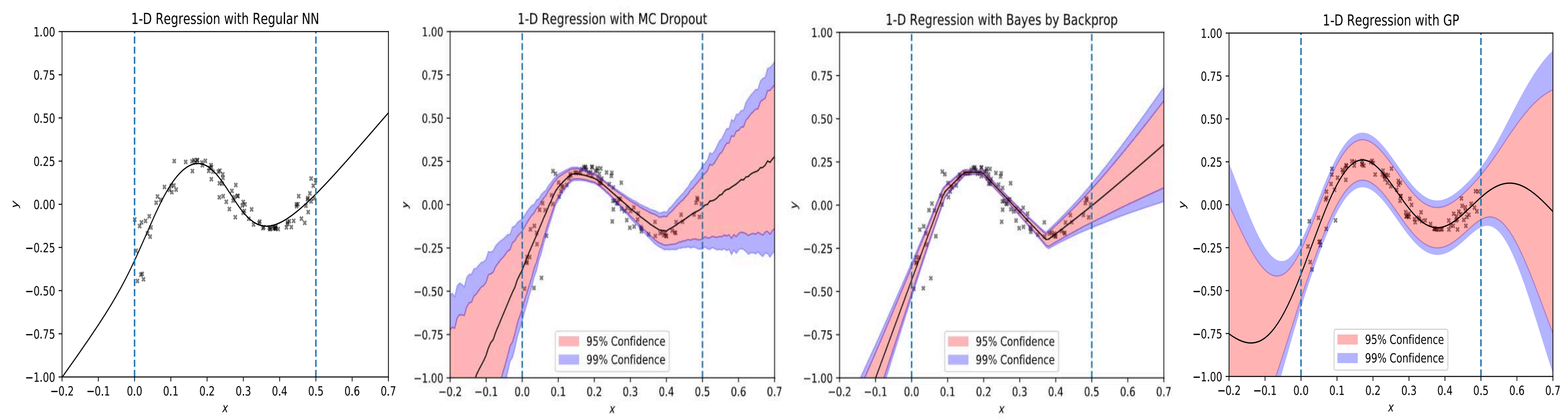

Figure 2. Regression of noisy data with credible intervals. Black crosses are training samples. Black lines are mean predictions. Pink/purple region is shows confidence. Left-to-right: Standard MLP, MC Dropout, BBB, and an RBF kernel GP. Implementations of BBB and MC Dropout built on code provided in [2].

Bayesian Optimisation

- Taking advantage of the uncertainty information in Bayesian neural networks we can perform Bayesian optimisation.
- We maximise a very simple negative quadratic function while sequentially selecting acquisition points.
- We use Thompson sampling to pick a single function and choose the next point of be the value that maximises that function.
- After only six observations we have a pretty good model of the function.

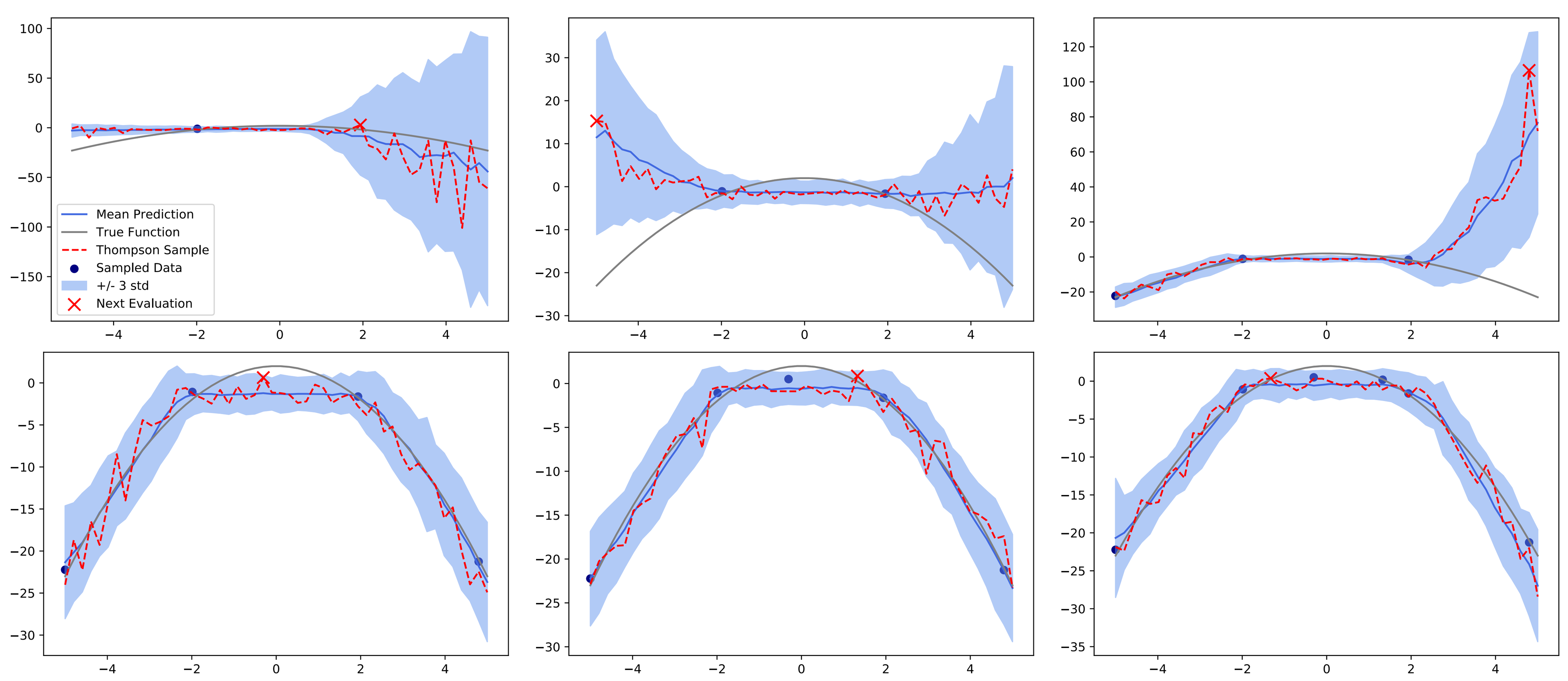

Figure 3. Results of BBB applied to Bayesian optimisation.

References

[1] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, p.1613–1622, JMLR.org, 2015.

[2] https://github.com/JavierAntoran/Bayesian-Neural-Networks

Classification on MNIST

Model	Error Rate (%) 400 Units	Error Rate (%) 1200 Units
Vanilla SGD	1.84	1.92
MC Dropout	1.99	1.85
Bayes-by-Backprop	2.01	2.35

Table 1. MNIST classification result of SGD, dropout, BBB applied to a feedforward NN with two 400/1200 unit layers.

What does the distribution over weights look like?

- BBB produces the weights with the highest variance.
- We calculate the signal-to-noise ratio (SNR) for all of the weights and see how pruning those weights with the lowest ratio affects performance - BBB is much less affected than other methods.

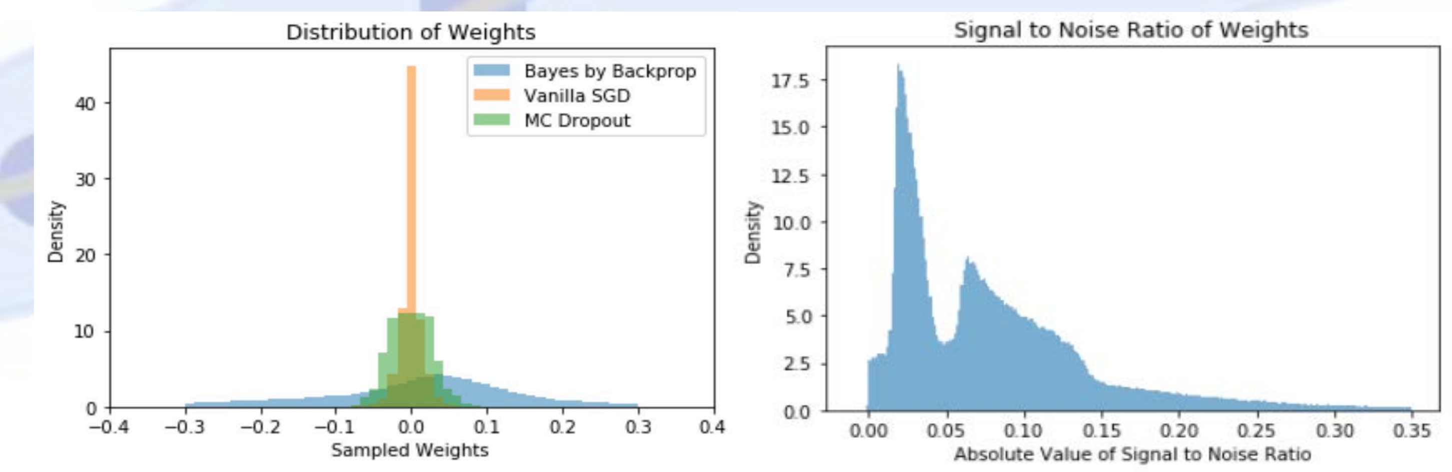

Figure 4. Left: Comparison between weight distribution of BBB, SGD, dropout. Right: Signal to noise ratio of all weights.

Weights Removed (%)	No. of Active Weights	Test Error Rate (%)
0	478410	2.59
50	239205	2.52
75	119603	2.62
95	23921	2.75
98	9569	3.14

Table 2. Classification error on MNIST after weight pruning.

- Pruning weights with low SNR results in minimal accuracy impact on BBB but leads to catastrophic failure in other methods including dropout.
- Can consider this Bayesian model selection with unnecessary parameters removed.

Conclusions

- Bayesian treatment allows for appropriate uncertainty estimation.
- Can be seen as an easy way to train an infinite ensemble of networks with only double the number of parameters.
- The induced predictive uncertainty allows for principled exploration in in RL and Bayesian optimisation.
- Future directions include developing more flexible approximate posteriors and extending to different neural net architectures.