Uncertainty Quantification Bayesian Neural Networks vs Gaussian Processes

What is Uncertainty?

- Uncertainty - captures our inability to precisely predict outcomes
- Variability - distribution derived from many instances of the data

- Uncertainty - I predict it will rain tomorrow with 80% certainty
- Variability - it rains 30% of days in Autumn

Why is it important to capture uncertainty?

- Uncertainty can tell us when a model is likely to make mistakes
- Useful for self-driving cars as if the model is uncertain about which action to take safety measures can be engaged
- Useful for when we want to approximate a physical process in order to quantify our expected error in any predictions

What was the aim of this project?

- Understanding whether Bayesian Neural Networks (BNNs) are able to capture variability in the data using uncertainty in a similar way to Gaussian processes
- Wanted to understand in what ways it might be appropriate and safe to use a Bayesian neural network over a Gaussian process
- Neural networks may be more efficient in some cases but we don’t know if their uncertainty can be trusted

Methodology

- Used the function $f(x) = x \times sin(x+5) + \epsilon \qquad \epsilon \sim \mathcal{N}(0,1)$

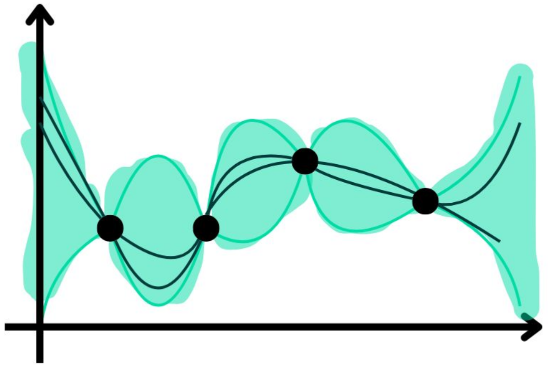
  With 10000 data points generated for the training set and another 10000 for the test set

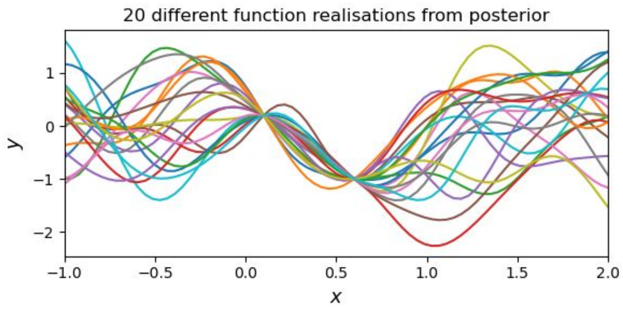
- Ran Bayesian neural networks in two forms (variational inference and MCMC) including versions with and without dropout and compared it to running a Gaussian process on the same function to see if they would produce similar results
- Used Python and Pytorch for my calculations
- Used the libraries torchbnn, GPyTorch and Uber's Pyro

What is a Gaussian Process?

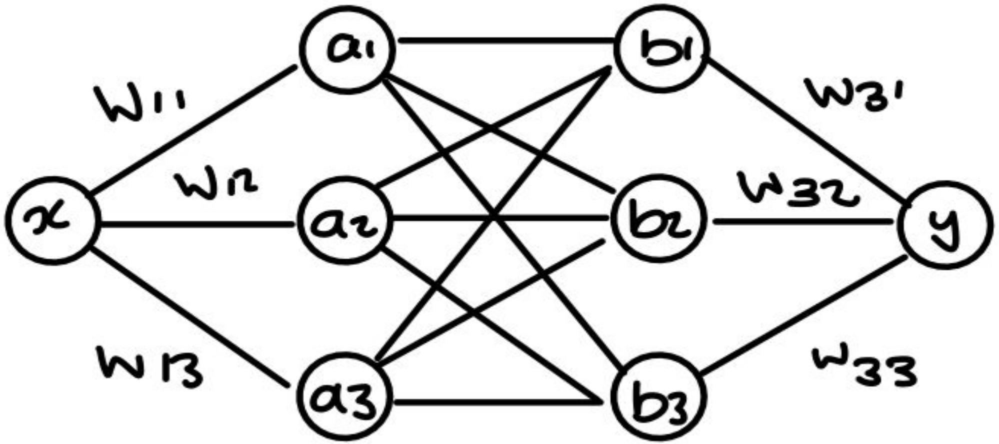
- Gaussian process as being an ensemble of infinite possible functions (usually continuous and smooth depending on the kernel) which are conditional on the data.

- It is defined by a mean function which gives a mean prediction and a covariance function known as a kernel which describes the similarity between any two particular observations.





What is a Neural Network?



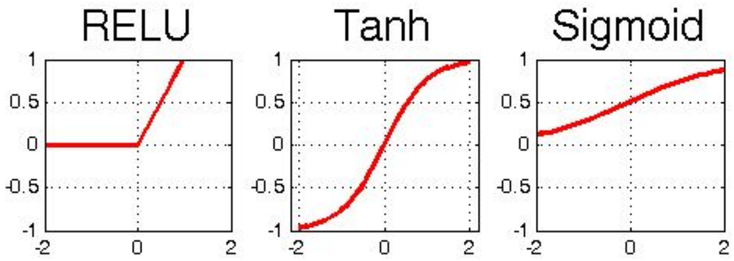
label connecting a_i and b_j is w_{2ij}

$$a_i = \sigma(w_{1i}x + \beta_{1i})$$

$$b_i = \sigma(\sum_{n=j}^{3} w_{2ji}a_j + \beta_{2i})$$

$$y = \sigma(\sum_{n=i}^{3} w_{3i}b_i + \beta_3)$$

Activation functions



$$a_i = \sigma(w_{1i}x + \beta_{1i})$$

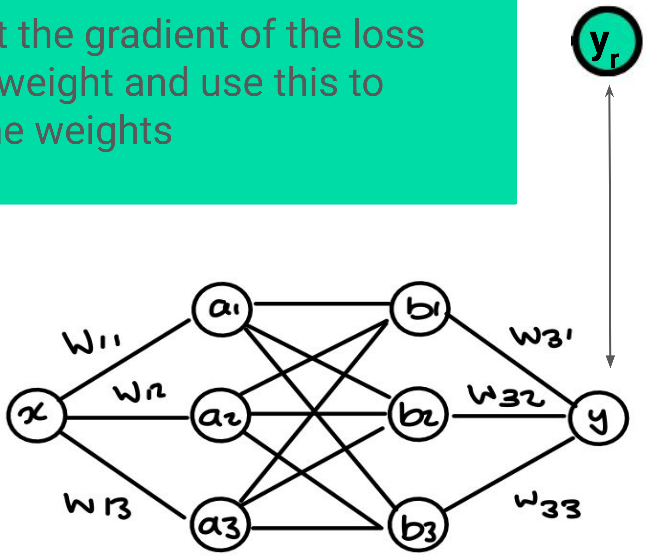
$$b_i = \sigma(\sum_{n=j}^{3} w_{2ji}a_j + \beta_{2i})$$

$$y = \sigma(\sum_{n=i}^{3} w_{3i}b_i + \beta_3)$$

Purpose: To make the outcome non-linear

Backpropagation

Work out the gradient of the loss by each weight and use this to adjust the weights



real value

$$MSE : L = (y_r - y)^2$$

We have our estimated value of y as seen before

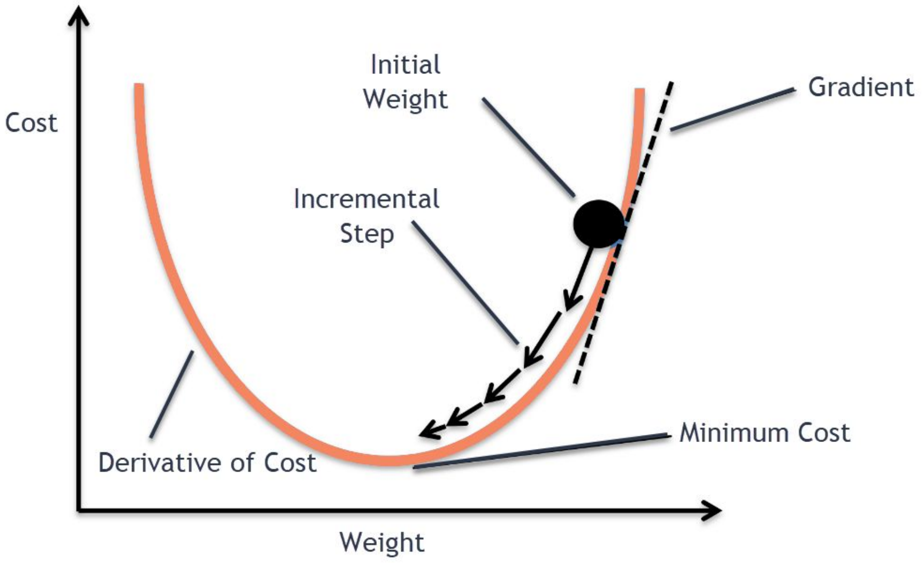
$$y = \sigma(z) \qquad z = \sum_{n=i}^{3} w_{3i} b_i + \beta_3$$

We calculate the gradient of the loss (how the loss changes) for each specific weight

$$\frac{\delta L}{\delta w_{31}} = b_1 \times \sigma(z)' \times 2(\sigma(z) - y_r)$$

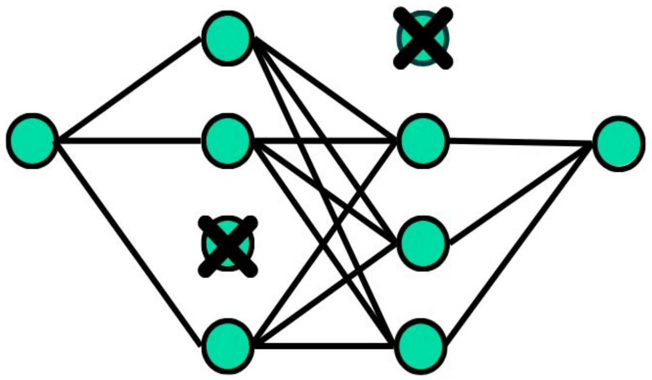
Backpropagation 2

- Once we have the gradients for loss given each of the weights, we use a form of gradient descent for adjusting our weights

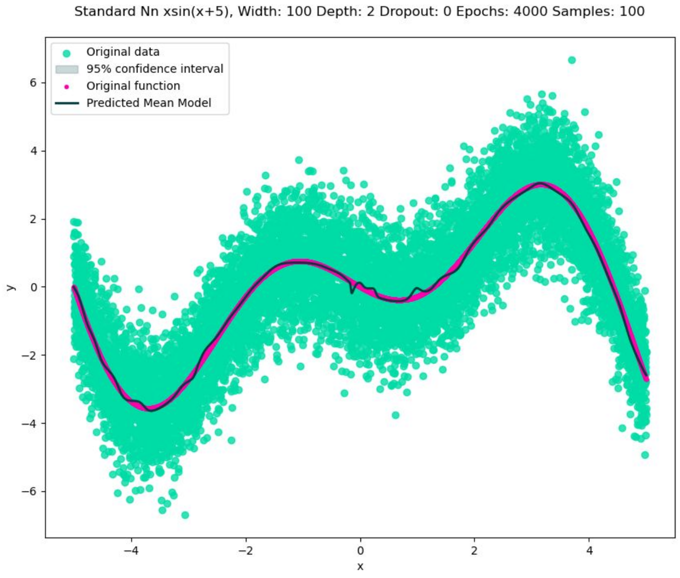


What is dropout?

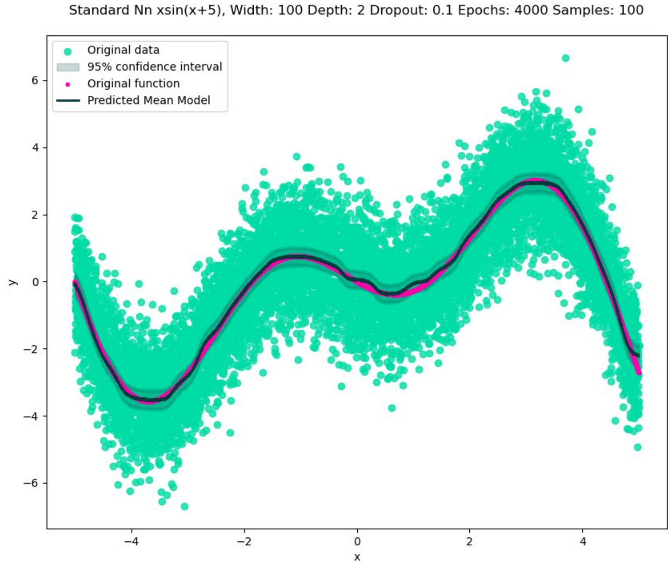
- Dropout is when during training of the neural network a certain percentage of nodes at each layer are removed
- Works as a form of regularization
- Also produces uncertainty if used when getting results from a regular neural network



Dropout comparison for regular neural network



Without dropout



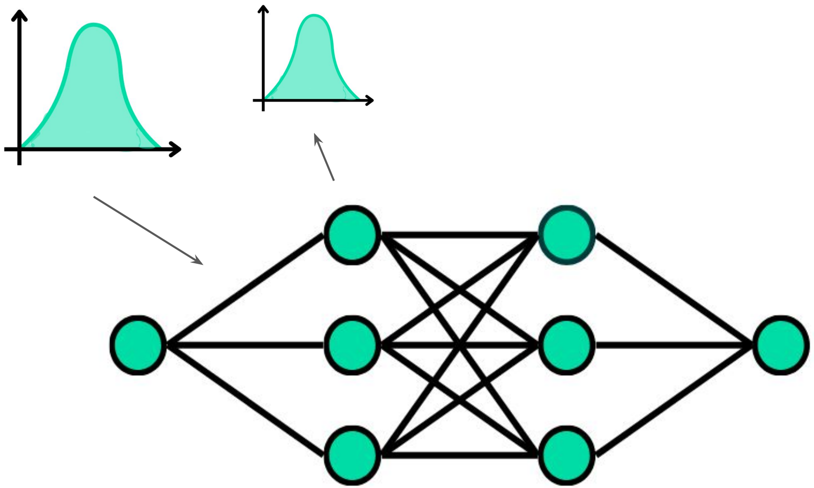
With dropout

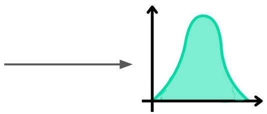
What is a Bayesian Neural Network?

We use (usually normal distribution) priors for all the weights which means all nodes other than the input node have a distribution

We use Bayes' Theorem to train the network

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$



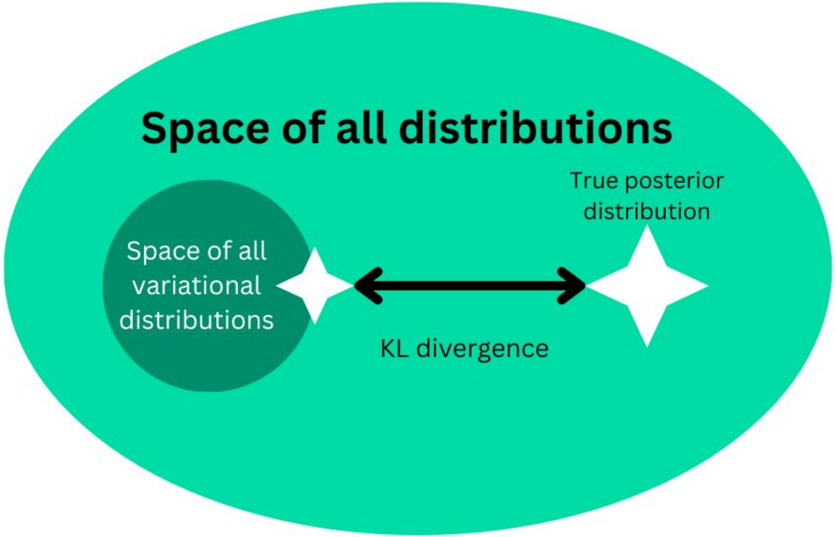


We can do this in a method similar to backpropagation with some key changes (Bayes by backprop)

What is Variational Inference

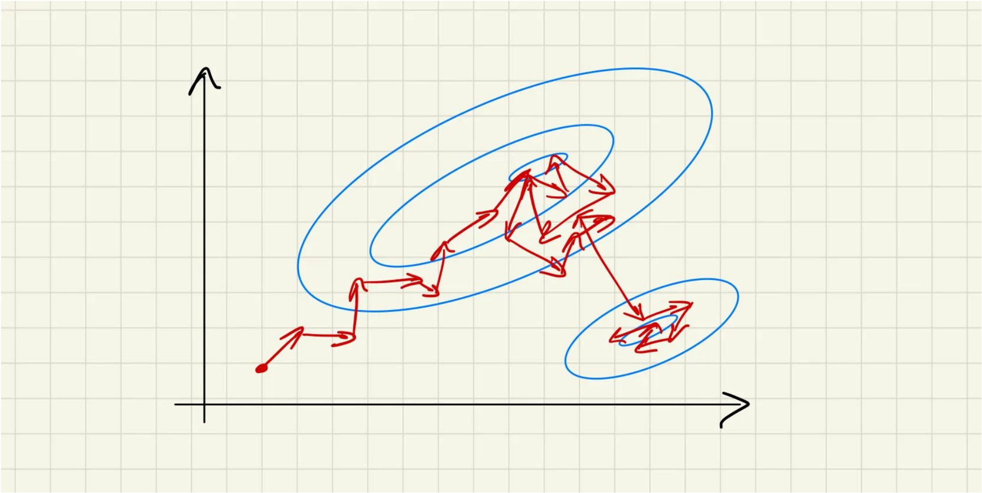
- Instead of sampling from the posterior directly, instead a distribution that is easy to sample from is created

- Then the distance between the real posterior and our approximated distribution is added as a form of loss for optimisation

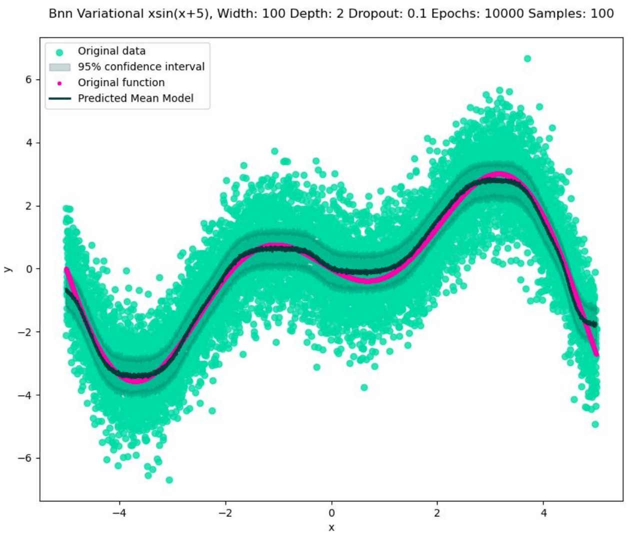


What is MCMC (Markov Chain Monte Carlo)

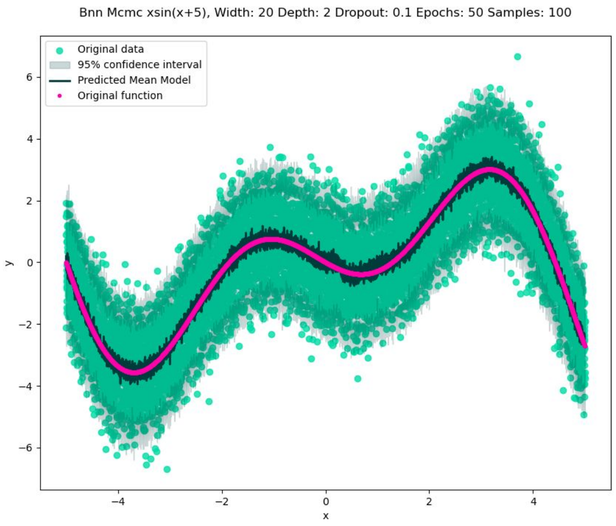
- Method for sampling the posterior distribution
- Involves exploring the space in a non-deterministic way
- Samples areas of higher probability more



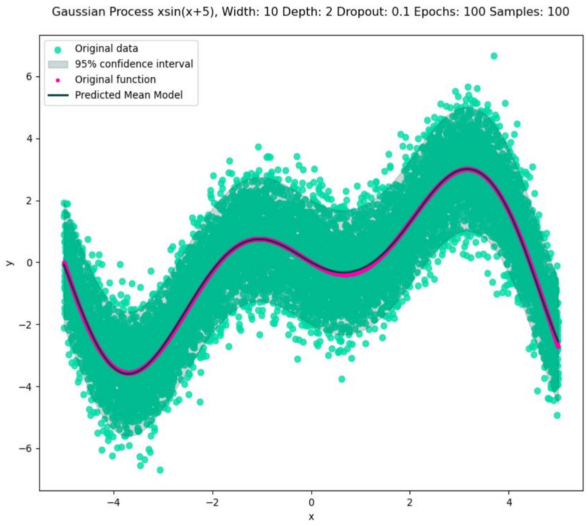
Graphs comparison



BNN Variational



BNN MCMC



Gaussian Process

Comparison of results

Type of Model	Variational BNN	MCMC BNN	Gaussian Process
Training Time	169	469	199
MSE	17.17	14.80	2.97
Coverage	0.414	0.9468	0.9562
Epochs	10000	50 (50 burn in)	100
Width	100	20	N/A
Depth	2	2	N/A

Discussion of Findings

$$y = x \times sin(x+5)$$

with 10,000 data points between -5 and +5

- Found that the form of Bayesian neural network (BNNs) using variational inference which I used couldn’t capture the variation in the data with uncertainty but BNNs using Markov Chain Monte Carlo (MCMC) could capture it
- BNNs using MCMC took longer to run than Gaussian processes which achieved similar accuracy
- MCMC in practice may run into issues as it will only be successful if the model converges
- Gaussian process was more accurate, faster and gave a smoother result