

# Lecture 10: Bayesian Deep Learning

## Efstratios Gavves

# Announcement

---

## Original schedule

- Tue, Oct 9: Lecture 11
- Fri, Oct 12: Lecture 12
- Thu, Oct 18: Lecture 13
- Fri, Oct 19: No lecture, instead poster presentations in the afternoon

## New schedule

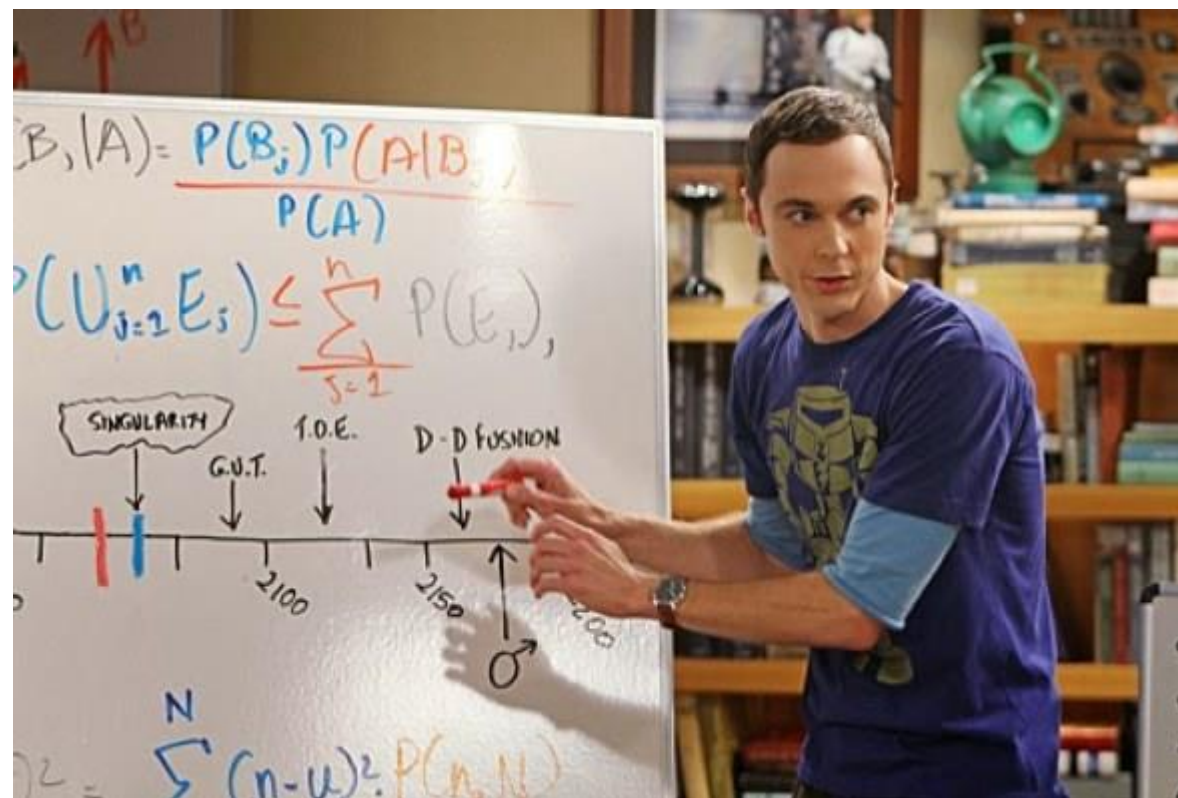
- Tue, Oct 9: **No lecture**
- Fri, Oct 12: Lecture 11
- Thu, Oct 18: Lecture 12
- Fri, Oct 19: Lecture 13 in the morning, poster presentations in the afternoon

# Lecture overview

---

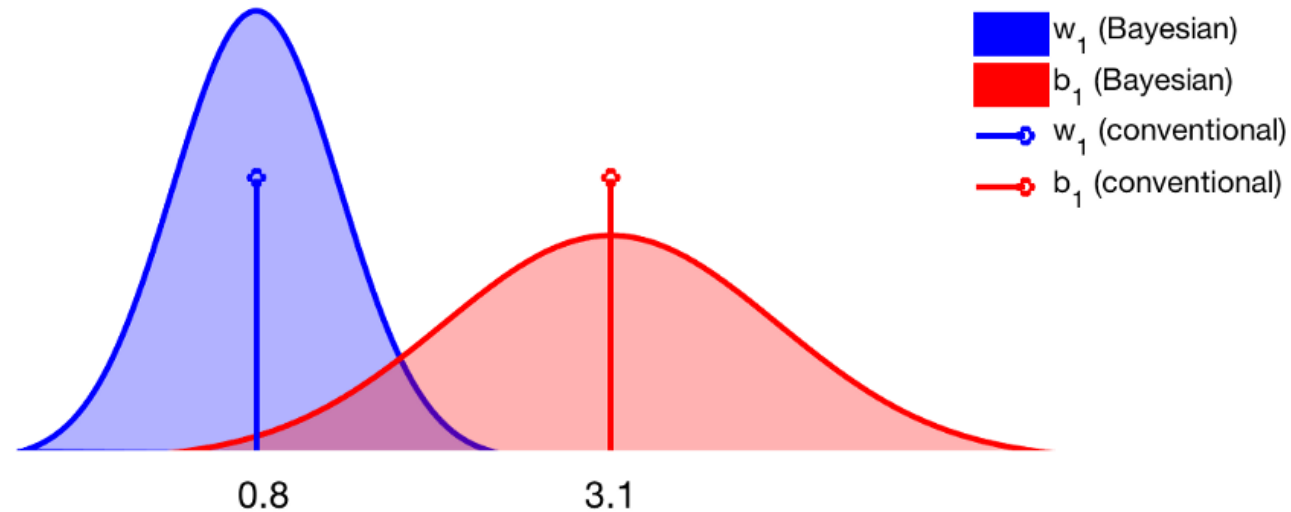
- Why Bayesian Deep Learning?
- Types of uncertainty
- Bayesian Neural Networks
- Backprop by Bayes
- MC Dropout

# Bayesian modelling



# The Bayesian approach

- Conventional Machine Learning → single optimal value per weight
- Bayesian Machine Learning → a distribution per latent variable/weight



# Benefits of being Bayesian

---

# Benefits of being Bayesian

---

- Ensemble modelling → better accuracies
- Uncertainty estimates → control our predictions
- Sparsity and model compression
- Active Learning
- Distributed Learning
- And more ...



# Why uncertainty?

- Machine predictions can get embarrassing quite quickly
- Would be nice to have a mechanism to control uncertainty in the world





# Types of Uncertainty

---

- Epistemic uncertainty
  - Captures our ignorance regarding which of all possible models from a class of models generated the data we have
  - By increasing the amount of data, epistemic uncertainty can be explained away
  - Why?

# Types of Uncertainty

- Epistemic uncertainty

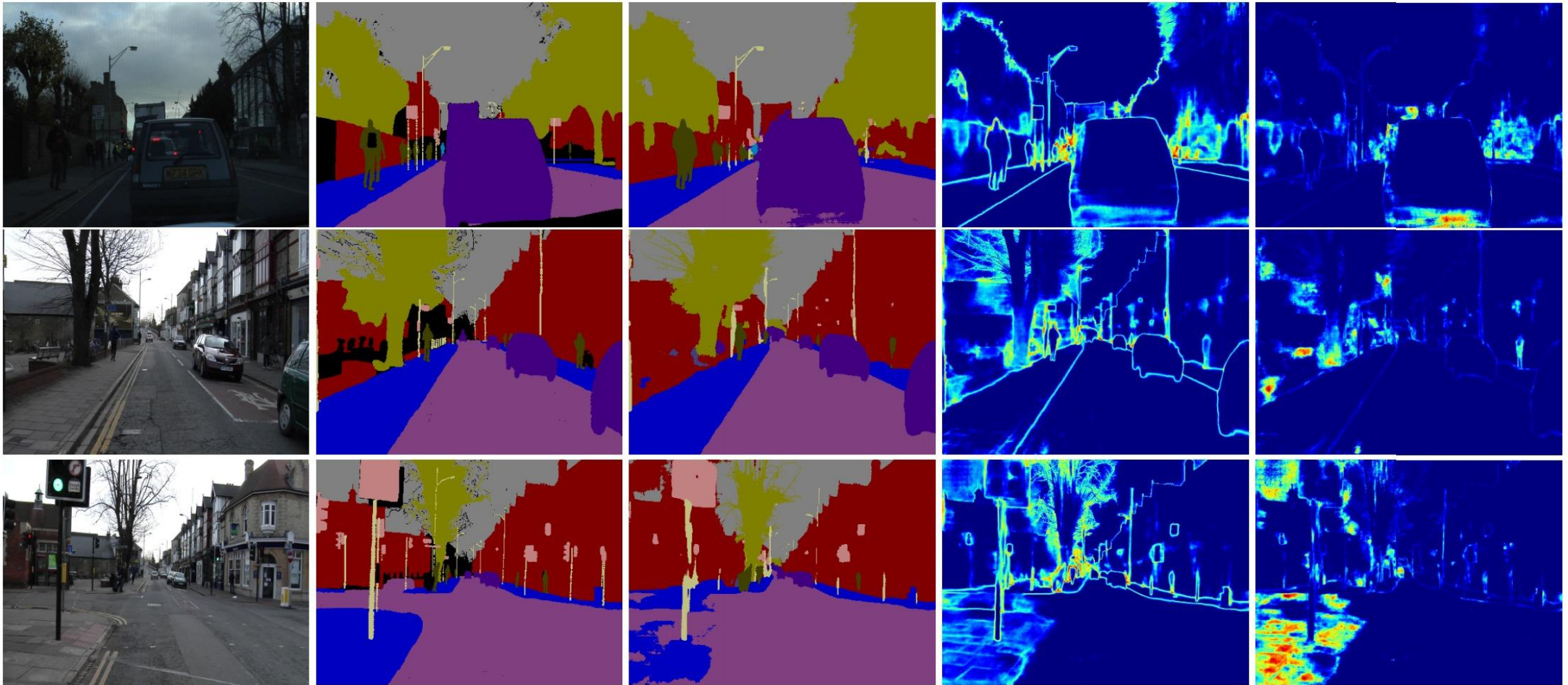
- Captures our ignorance regarding which of all possible models from a class of models generated the data we have
- By increasing the amount of data, epistemic uncertainty can be explained away
- **Why?** The more data we have the fewer are the possible models that could in fact generate all the data

- Aleatoric uncertainty

- Uncertainty due to the nature of the data.
- If we predict depth from images, for instance, highly specular surfaces make it very hard to predict depth. Or if we detect objects, severe occlusions make it very difficult to predict the object class and the precise bounding box
- Better features reduce aleatoric uncertainty

- Predictive Uncertainty = Epistemic uncertainty + Aleatoric uncertainty

# Types of Uncertainty



(a) Input Image

(b) Ground Truth

(c) Semantic Segmentation

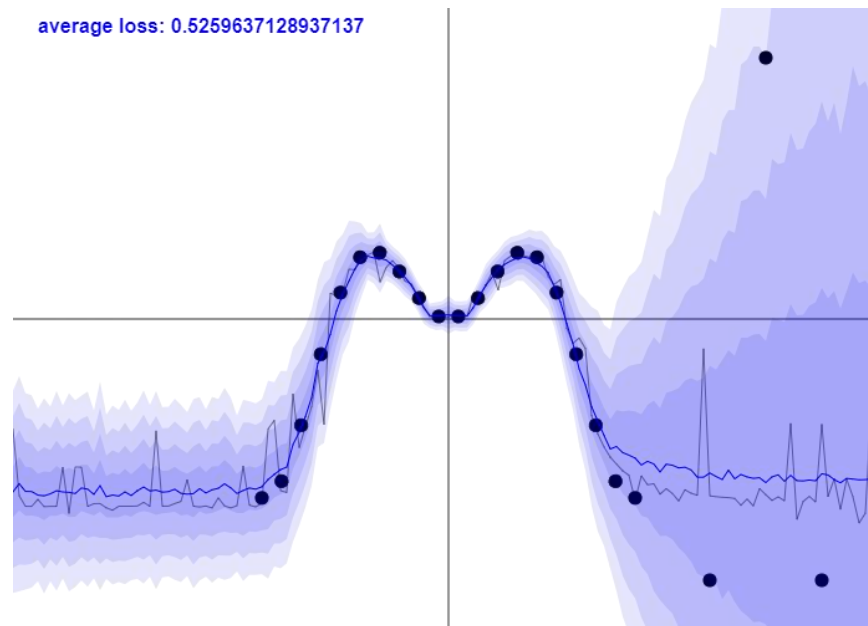
(d) Aleatoric Uncertainty

(e) Epistemic Uncertainty

# Epistemic uncertainty

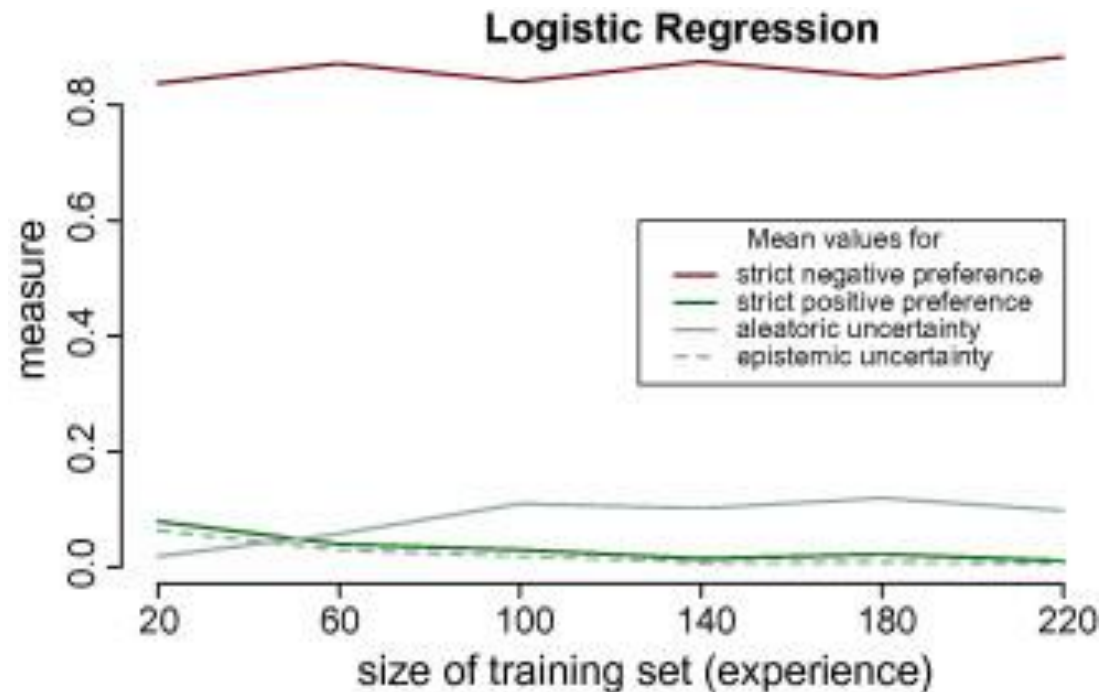
- Important to consider modelling when
  - we have safety-critical applications
  - the datasets are small

Should I give the operate or give a drug?



# Aleatoric uncertainty

- Important to consider modelling when
  - Large datasets → small epistemic uncertainty
  - Real-time → aleatoric models can be deterministic (no Monte Carlo sampling needed)



# Data-dependent aleatoric uncertainty

---

- Also called heteroscedastic aleatoric uncertainty
- The uncertainty is in the raw inputs
- Data-dependent aleatoric uncertainty can be one of the model outputs

# Task-dependent aleatoric uncertainty

---

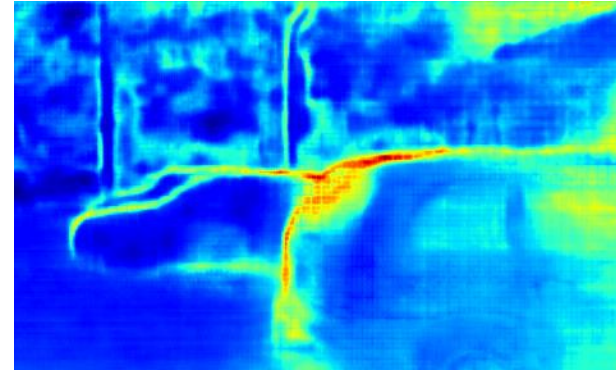
- Also called homoscedastic aleatoric uncertainty
- It is not a model output, it relates to the uncertainty that a particular task might cause
  - For instance, for the task of depth estimation predicting depth around the edges is very hard → thus uncertain
- When having multiple tasks task-dependent aleatoric uncertainty may be reduced
  - For instance?



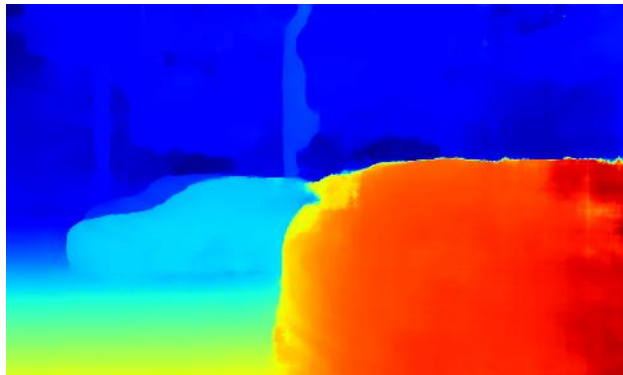
# Task-dependent aleatoric uncertainty



Input



Uncertainty

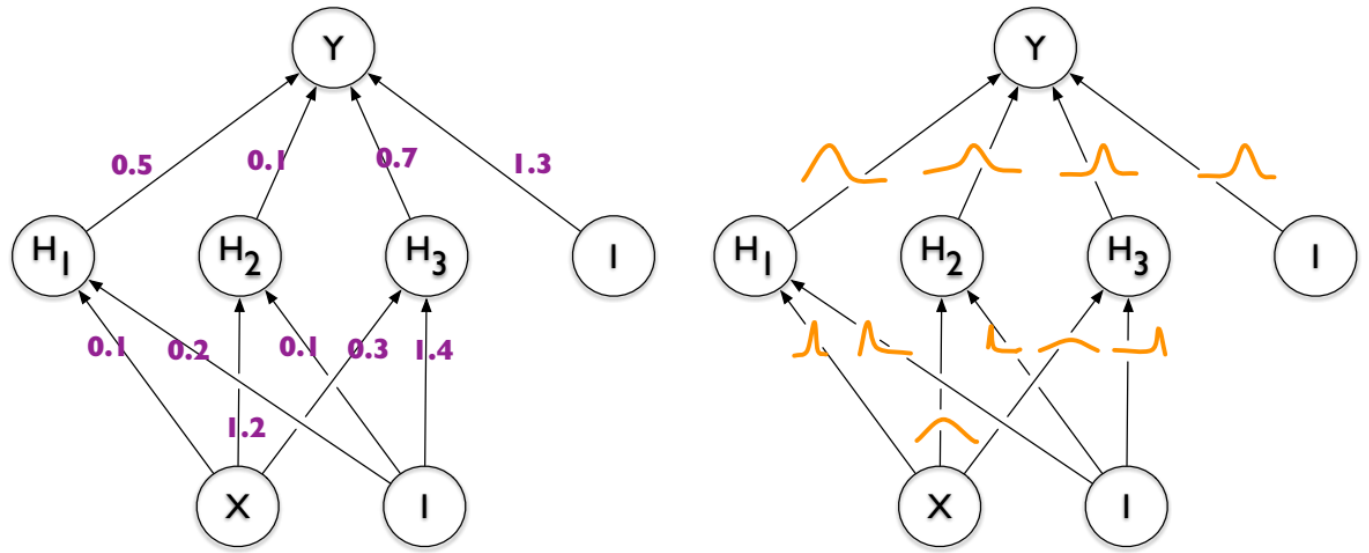


Depth Prediction



Edge prediction as second task?

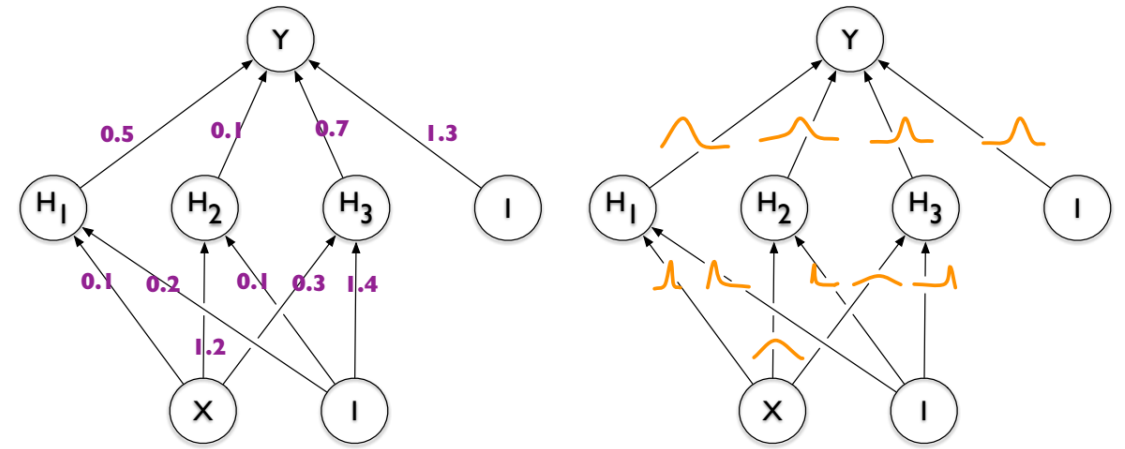
# Bayesian Modelling Variational Inference



*Figure 1.* Left: each weight has a fixed value, as provided by classical backpropagation. Right: each weight is assigned a distribution, as provided by Bayes by Backprop.

# Bayesian Deep Learning

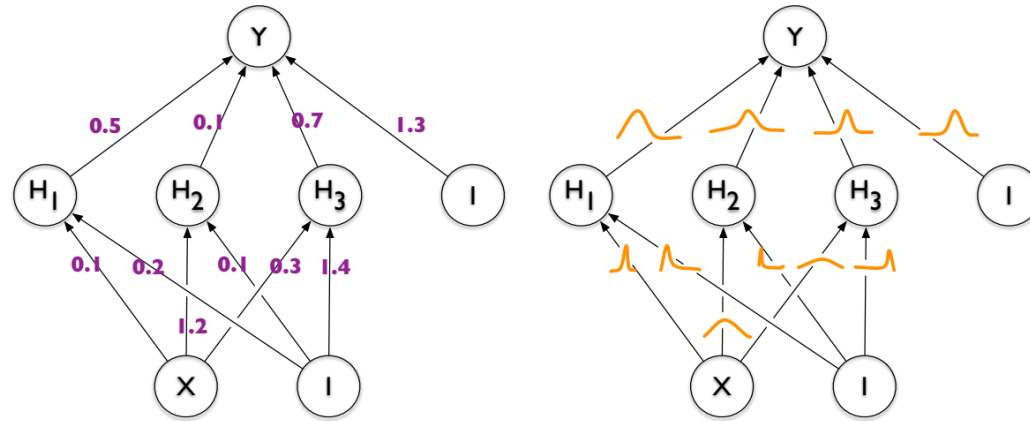
- Deep learning provides powerful feature learners from raw data
  - But they cannot model uncertainty
- Bayesian learning provides meaningful uncertainty estimates
  - But they often rely on methods that are not scalable, e.g. Gaussian Processes
- Bayesian Deep Learning combines the best of two worlds
  - Hierarchical representation power
  - Outputs complex multi-modal distributions



*Figure 1.* Left: each weight has a fixed value, as provided by classical backpropagation. Right: each weight is assigned a distribution, as provided by Bayes by Backprop.

# Bayesian Deep Learning: Goal?

- Deep Networks: filters & architecture
- Standard Deep Networks → single optimal value per filter
- A Bayesian approach associates a distribution per latent variable/filter



*Figure 1.* Left: each weight has a fixed value, as provided by classical backpropagation. Right: each weight is assigned a distribution, as provided by Bayes by Backprop.

# Modelling data-dependent aleatoric uncertainty

- We add a variance term per data point to our loss function

$$\mathcal{L} = \frac{\|y_i - \hat{y}_i\|^2}{2\sigma_i^2} + \log \sigma_i$$

- What is the role of  $2\sigma_i^2$ ?
  - When the nominator becomes large, the network may choose to shrink the loss by increasing the output variance  $\sigma_i$
- But then what about  $\log \sigma_i$ ?
  - Without it the network will always tend to return high variance

A. Kendal, Y. Gal, What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision, NIPS 2017

# Modelling task-dependent aleatoric uncertainty

- Similar to the data-dependent uncertainty

$$\mathcal{L} = \frac{\|y_i - \hat{y}_i\|^2}{2\sigma^2} + \log \sigma$$

- The only difference is that now the variance is a learnable parameter shared by all task data points
- One can use task-dependent uncertainties to weigh multiple tasks



# Results

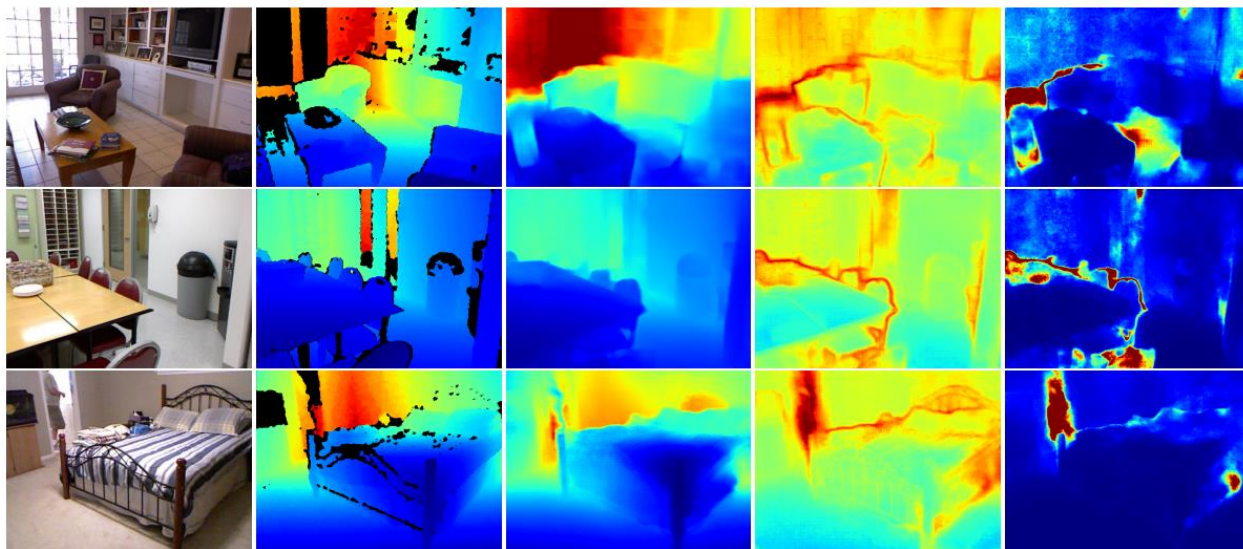


Figure 5: NYUv2 Depth results. From left: input image, ground truth, depth regression, aleatoric uncertainty, and epistemic uncertainty.

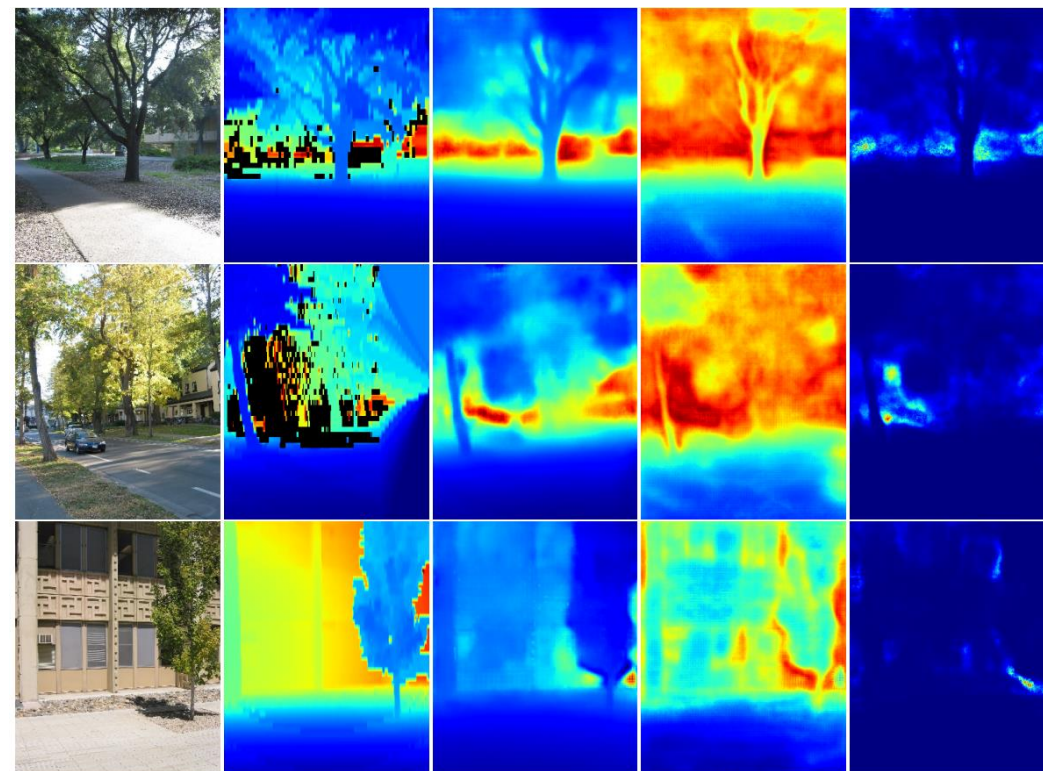


Figure 6: Qualitative results on the Make3D depth regression dataset. Left to right: input image, ground truth, depth prediction, aleatoric uncertainty, epistemic uncertainty. Make3D does not provide labels for depth greater than 70m, therefore these distances dominate the epistemic uncertainty signal. Aleatoric uncertainty is prevalent around depth edges or distant points.



# Modelling epistemic uncertainty

---

- Epistemic uncertainty is harder to model

$$p(w|x, y) = \frac{p(x, y|w)p(w)}{\int_w p(x, y|w)p(w) dw}$$

- Computing the posterior densities is usually intractable for complex functions like neural networks

# Epistemic uncertainty: Monte Carlo (MC) Dropout!

---

- Variational Inference assumes a (approximate) posterior distribution to approximate the true posterior
- Dropout turns on or off neurons based on probability distribution (Bernoulli)
- The Bernoulli distribution can be used as the variational distribution → MC Dropout

Y. Gal, Z. Ghahramani, Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning, MLR 2016

# Bayesian Neural Networks as (approximate) Gaussian Processes

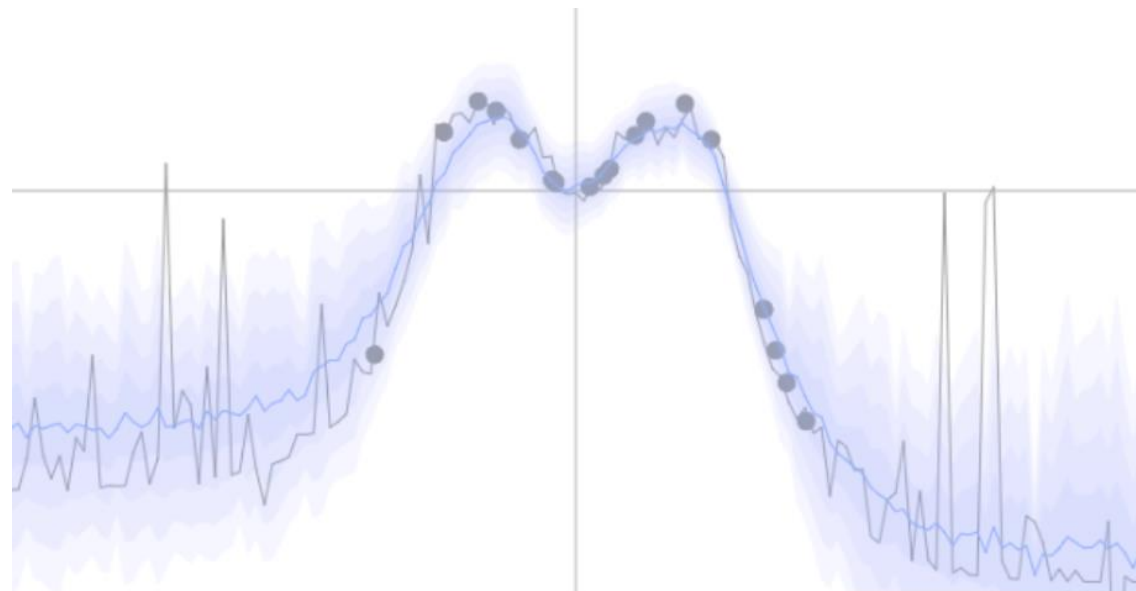
- Expected model output described by
  - Predictive mean  $\mathbb{E}(\mathbf{y}^*)$
  - Predictive variance  $\text{Var}(\mathbf{y}^*)$
- Starting from a Gaussian Process and deriving a variational approximation, one arrives at a Dropout Neural Network
- The model precision is equivalent to  $\tau = \frac{l^2 p}{2N\lambda}$ 
  - $l$  is the length-scale: large for high-frequency data, small for low-frequency data
  - $p$  the dropout survival rate
  - $\lambda$  is the learning rate

# Predictive mean and variance in MC Dropout Deep Nets

$$\tau = \frac{l^2 p}{2N\lambda}$$

$$\mathbb{E}(y^*) \approx \frac{1}{T} \sum_{t=1}^T \hat{y}_t^*(x^*)$$

$$\text{Var}(y^*) \approx \tau^{-1} \mathbf{I}_D + \frac{1}{T} \sum_{t=1}^T \hat{y}_t^*(x^*)^T \hat{y}_t^*(x^*) - \mathbb{E}(y^*)^T \mathbb{E}(y^*)$$



[Demo](#)

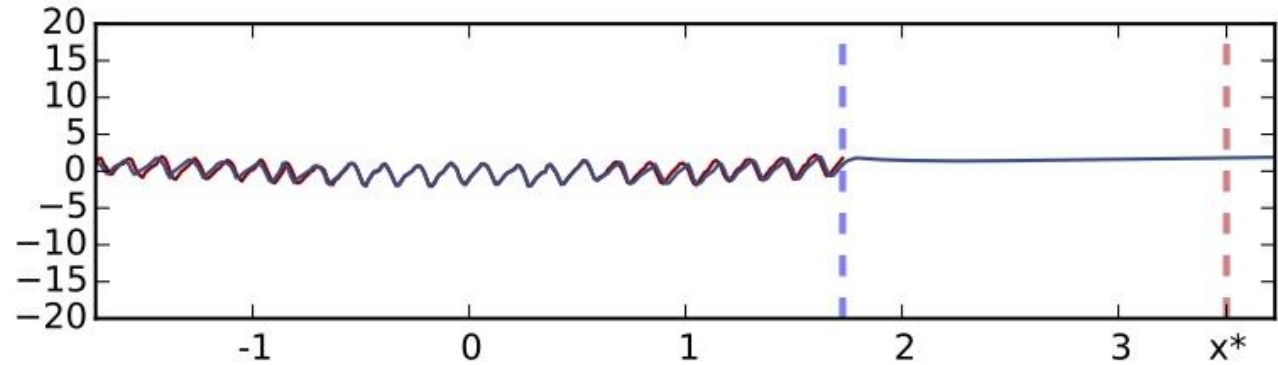
# Dropout for Bayesian Uncertainty in practice

---

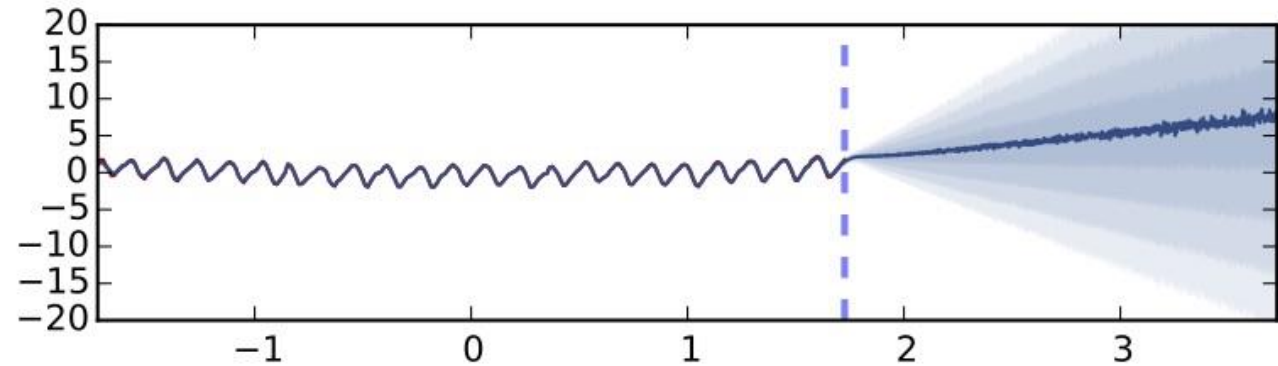
- Use dropout in all layers both during training and testing
- At test time repeat dropout 10 times and look at **mean** and **sample variance**
- Pros: Very easy to train
- Pros: Easy to convert a standard network to a Bayesian Network
- Pros: No need for an inference network  $q_w(\varphi)$
- Cons: Requires weight sampling also during testing → expensive

# Example

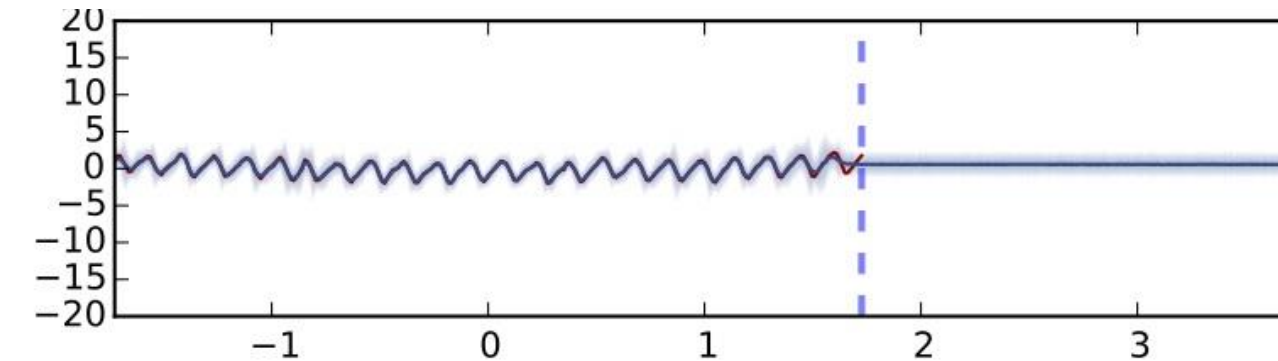
Prediction in a 5-layer ReLU neural network with dropout



Using 100-trial MC dropout



Using 100-trial MC dropout with **tanh** nonlinearity



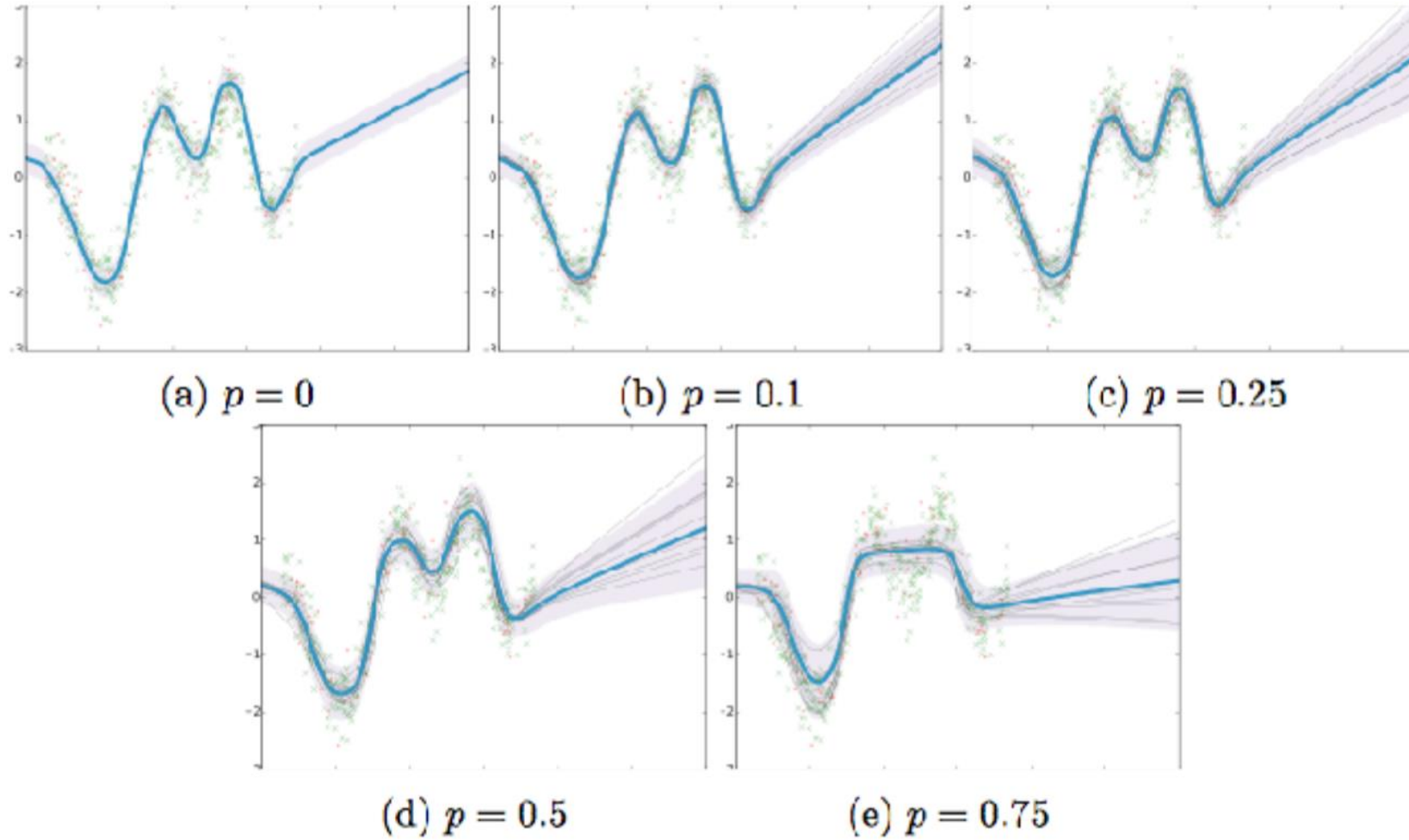
# Tricks of the trade

---

- Over-parameterized models give better uncertainty estimates, as they capture a bigger class of data
- Large models need higher dropout rates for meaningful uncertainty
  - Large models tend to push  $p \rightarrow 0.5$
  - For smaller models lower dropout rates reduce uncertainty estimates



# MC Dropout rates



# Bayes by Backprop

- Start from a Deep Network with a distribution on its weights
- Similar to VAE, we only need to minimize the negative ELBO

$$\mathcal{L} = KL(q(w|\theta)||p(w)) - \int_w q(w|\theta) \log p(x|w) dw$$

- An approximation is to sample a single weight value  $w_s$  from our posterior  $q(w|\theta)$ , e.g., a Gaussian, then compute the MC ELBO

$$\mathcal{L} = \log q(w_s|\theta) - \log p(w_s) - \log p(x|w_s)$$

C. Blundell, J. Cornebise, K. Kavukcuoglu, D. Wierstra, Weight Uncertainty in Neural Networks, ICML 2015

# Bayes by Backprop

---

- Assume a Gaussian variational posterior on the weights
- Each weight is then parameterized as

$$w = \mu + \varepsilon \circ \sigma$$

where  $\sigma$  is  $\rho$ -parameterized by the softplus

$$\sigma = \log(1 + \exp(\rho))$$

- Why?

# Bayes by Backprop

- Assume a Gaussian variational posterior on the weights
- Each weight is then parameterized as

$$w = \mu + \varepsilon \circ \sigma$$

where  $\sigma$  is  $\rho$ -parameterized by the softplus

$$\sigma = \log(1 + \exp(\rho))$$

- **Why?** With this parameterization the standard deviation is always positive
- Then we optimize the ELBO
- In the end we learn an ensemble of networks, since we can sample as many weights as we want

# Bayes by Backprop - Algorithm

1. Sample  $\varepsilon \sim N(0, 1)$
2. Set  $w = \mu + \varepsilon \circ \log(1 + \exp(\rho))$
3. Set  $\theta = \{\mu, \rho\}$
4. Let  $\mathcal{L}(w, \theta) = \log q(w|\theta) - \log p(w)p(x|w)$
5. Calculate gradients

$$\begin{aligned}\nabla_{\mu} &= \frac{\partial \mathcal{L}}{\partial w} \frac{\partial w}{\partial \mu} + \frac{\partial \mathcal{L}}{\partial \mu} \\ \nabla_{\rho} &= \frac{\partial \mathcal{L}}{\partial w} \frac{\partial w}{1 + \exp(-\rho)} + \frac{\partial \mathcal{L}}{\partial \rho}\end{aligned}$$

7. Last, update the variational parameters

$$\begin{aligned}\mu_{t+1} &= \mu_t - \eta_t \nabla_{\mu} \\ \rho_{t+1} &= \rho_t - \eta_t \nabla_{\rho}\end{aligned}$$

# Bayes by Backprop: Results

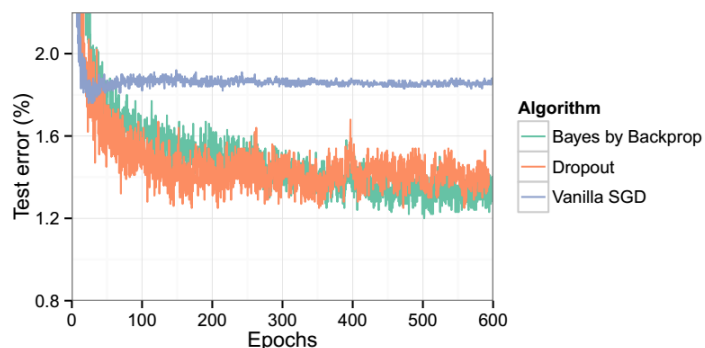


Figure 2. Test error on MNIST as training progresses.

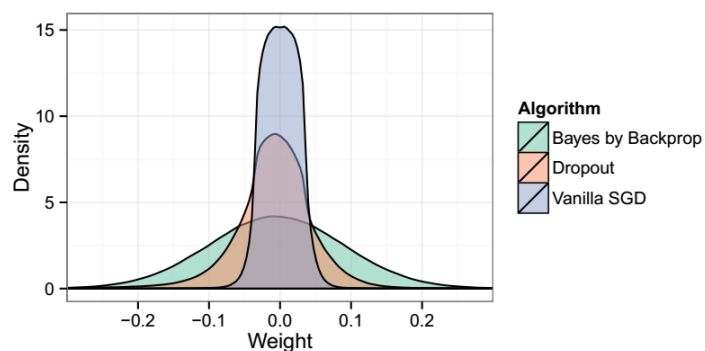


Figure 3. Histogram of the trained weights of the neural network, for Dropout, plain SGD, and samples from Bayes by Backprop.

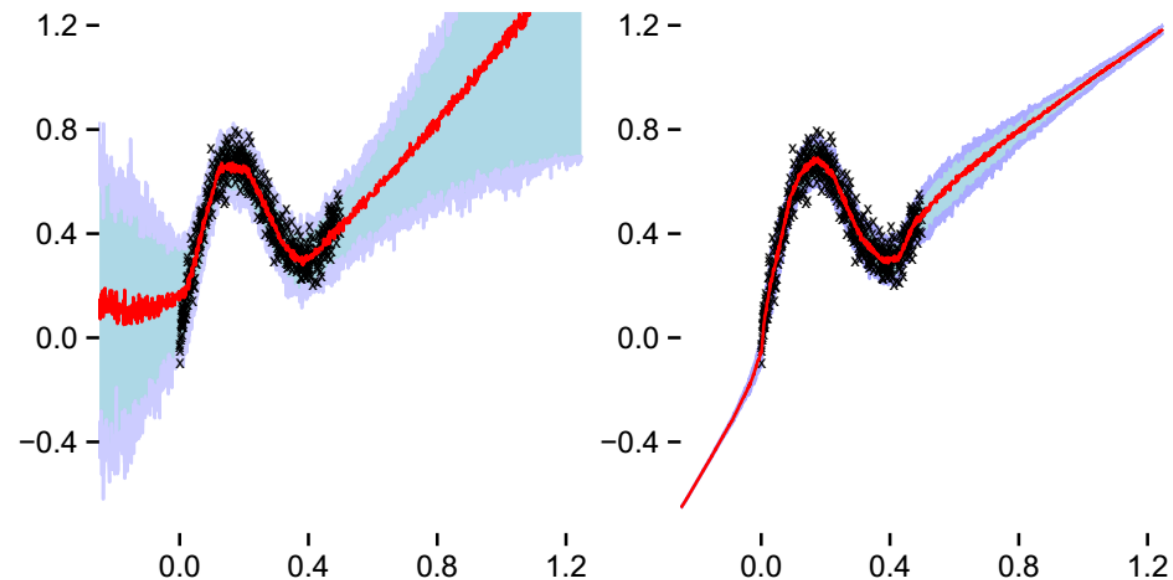


Figure 5. Regression of noisy data with interquartile ranges. Black crosses are training samples. Red lines are median predictions. Blue/purple region is interquartile range. Left: Bayes by Backprop neural network, Right: standard neural network.

# Bayesian Neural Network Compression

- Revisit connection between minimum description length and variational inference
- Minimum Description Length: best model uses the minimum number of bits to communicate the model complexity  $\mathcal{L}^C$  and the model error  $\mathcal{L}^E$

$$\mathcal{L}(\varphi) = \underbrace{\mathbb{E}_{q_w(\varphi)}[\log p(D|w)]}_{\mathcal{L}^E} + \underbrace{\mathbb{E}_{q_w(\varphi)}[\log p(w)] + H(q_w(\varphi))}_{\mathcal{L}^C}$$

- Use sparsity-inducing priors for groups of weights  $\rightarrow$  prune weights that are not necessary for the model

C. Louizos, K. Ullrich, M. Welling, Bayesian Compression for Deep Learning, NIPS 2017



# Bayesian Neural Network Compression

- Define the prior over weights

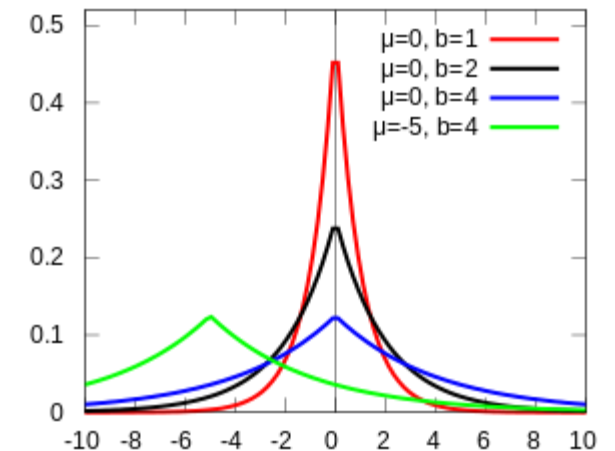
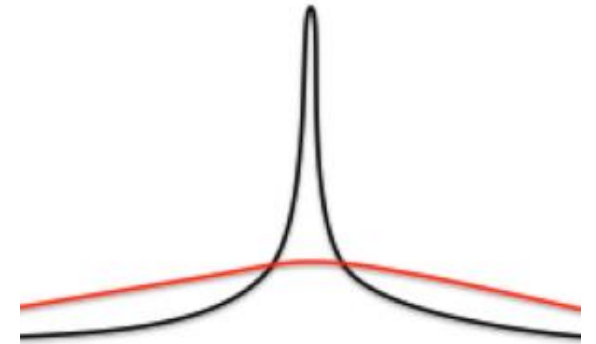
$$z \sim p(z)$$
$$w \sim N(w; 0, z^2)$$

- The **scales** of the weight prior **have a prior** themselves

- Goal: by treating the scales as random variables the marginal  $p(w)$  can be set to have heavy tails  $\rightarrow$  more density near 0

- Several distributions possible to serve as priors

Spike-and-slab  
distribution



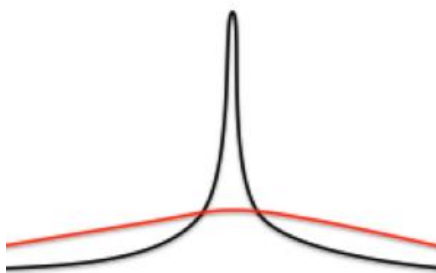
Laplace distribution  
(Lasso)

# Sparse-inducing distributions

## Spike-and-slab distribution

Mixture of a very spiky and a very broad Gaussian

Or a mixture of a  $\delta$ -spike at 0, and a slab on the real line

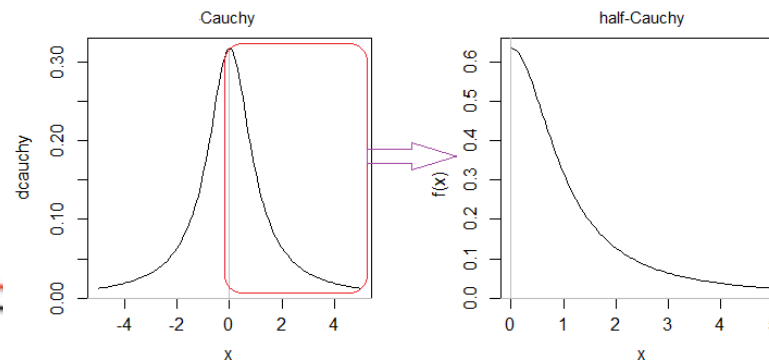
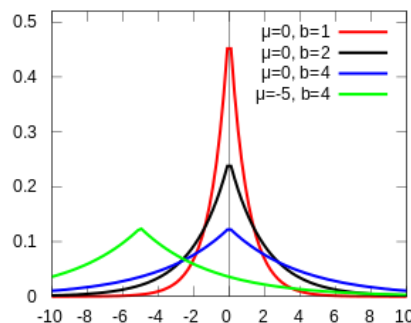


This would lead to large number of possible models:  $2^M$  for  $M$  parameters

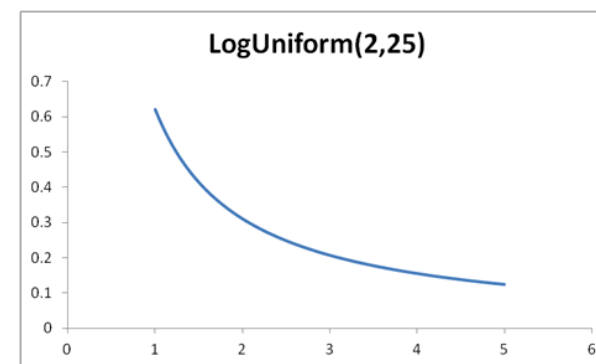
## Laplace distribution (Lasso)

$$p(z^2; \lambda) = \exp(-\lambda |z|)$$

Lasso focuses on shrinking the larger values



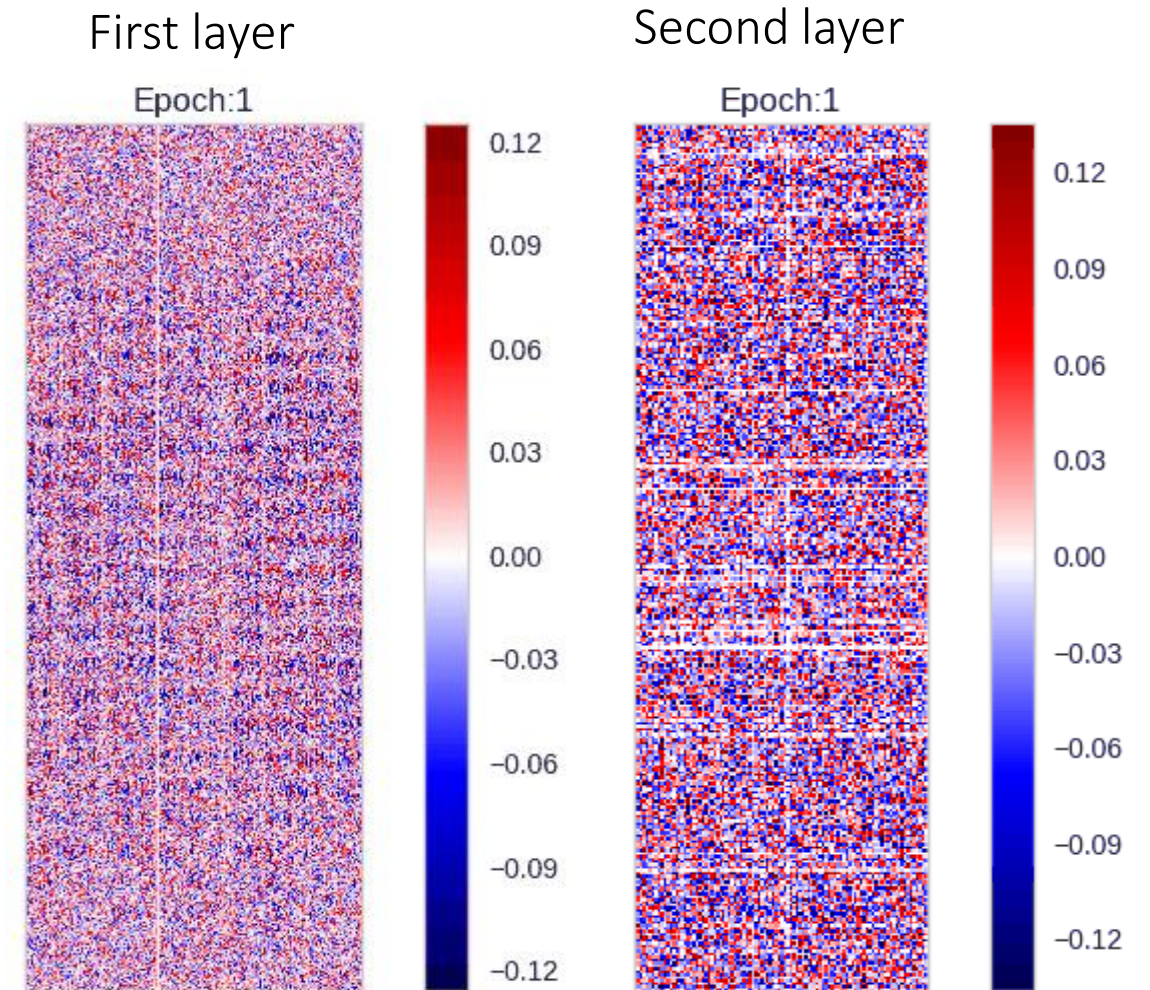
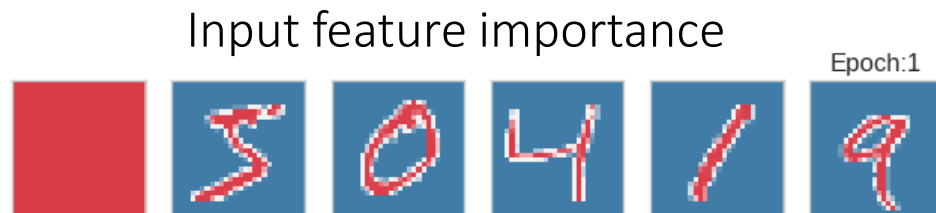
## Half-Cauchy



## Log-Uniform

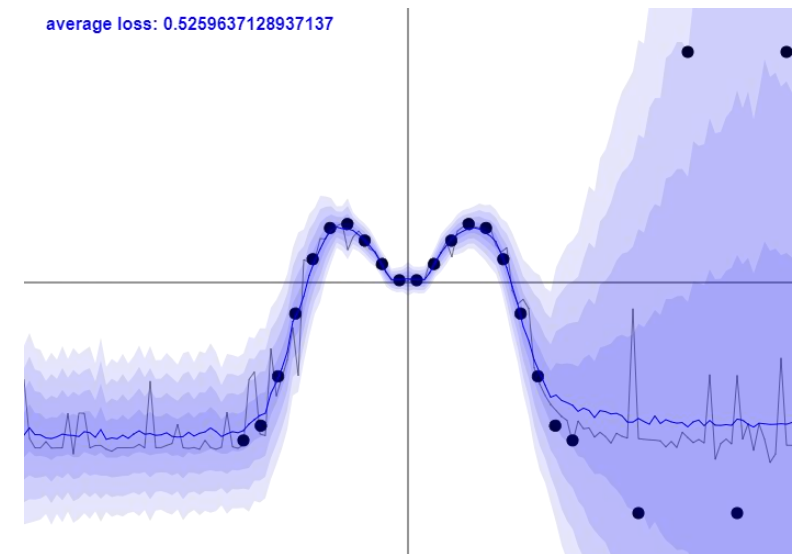
# Bayesian Neural Network Compression

- 700x compression
- 50x speed up



# Some open questions

- Hard to model epistemic uncertainty real-time
  - Typically, Monte Carlo approximations are required
  - Efficiency and uncertainty is needed for robotics, self-driving, health AI, etc
- No benchmarks to fairly evaluate
- Inference techniques are still not good enough



# Summary

- Why Bayesian Deep Learning?
- Types of uncertainty
- Bayesian Neural Networks
- Backprop by Bayes
- MC Dropout