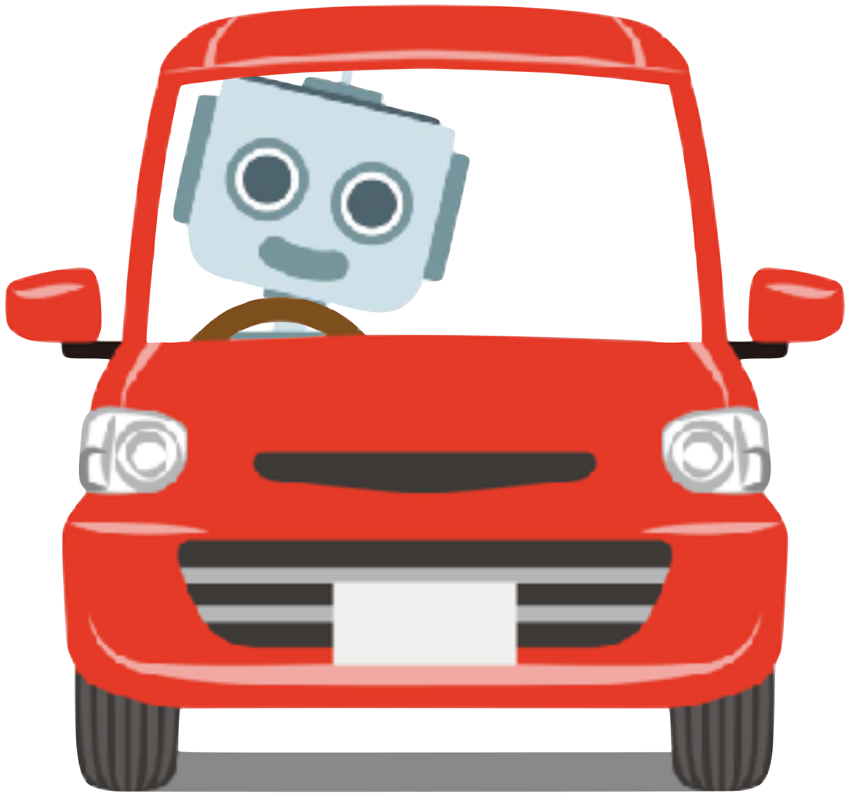
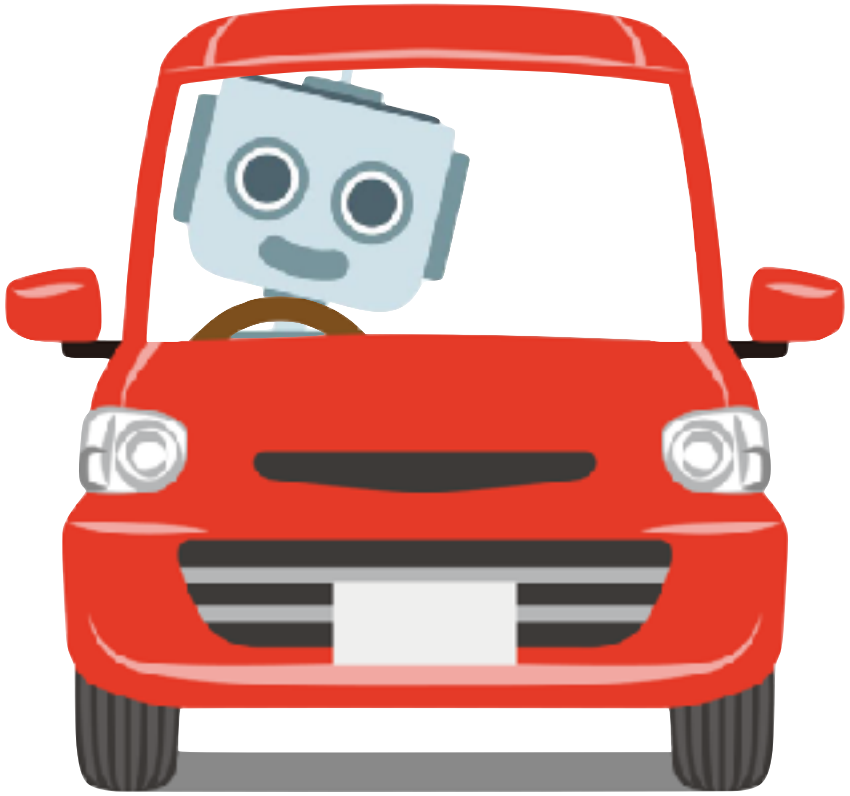
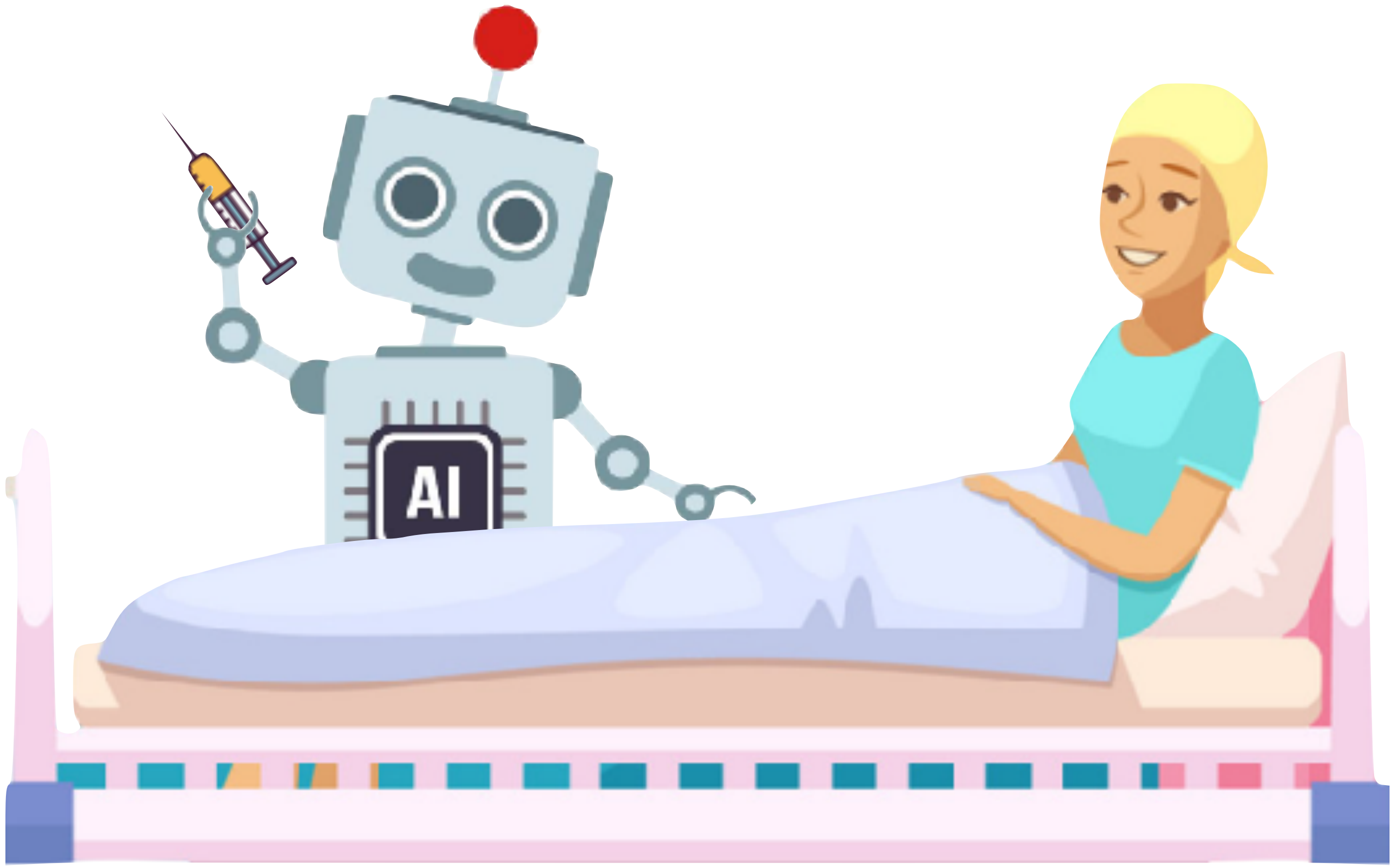
Bayesian Deep Learning:
Motivation and Model Definition

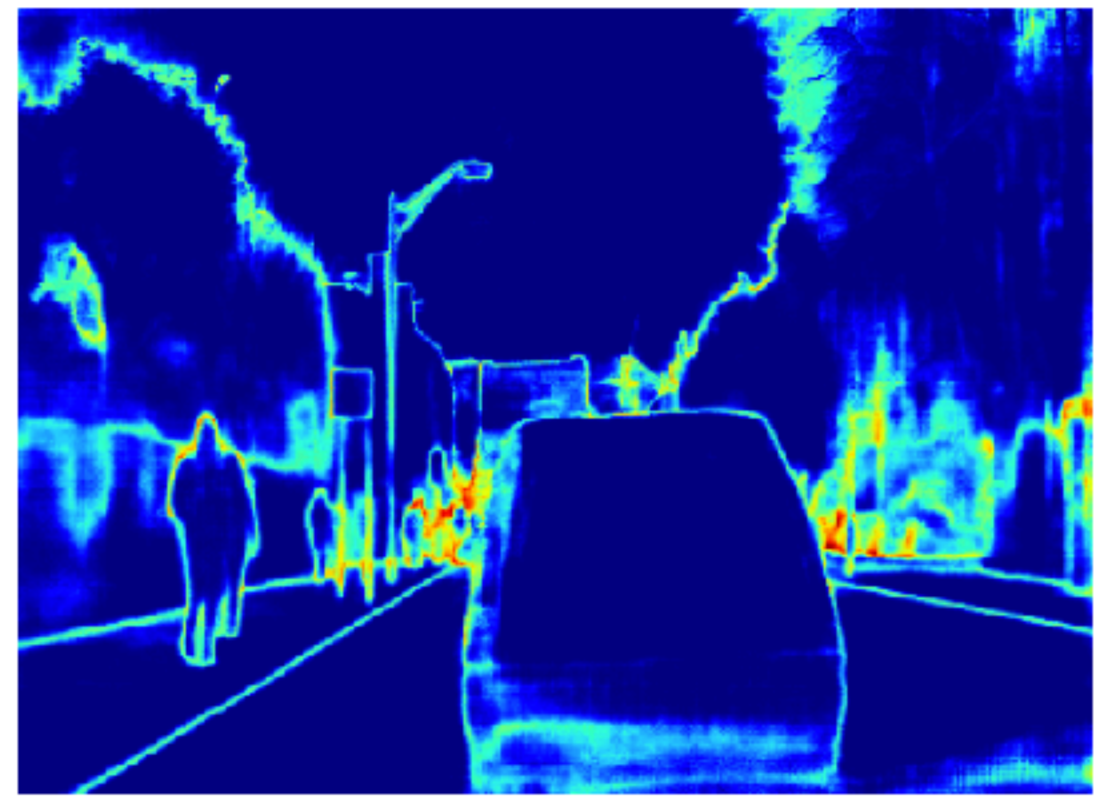
Eric Nalisnick

Deep Learning II,
University of Amsterdam

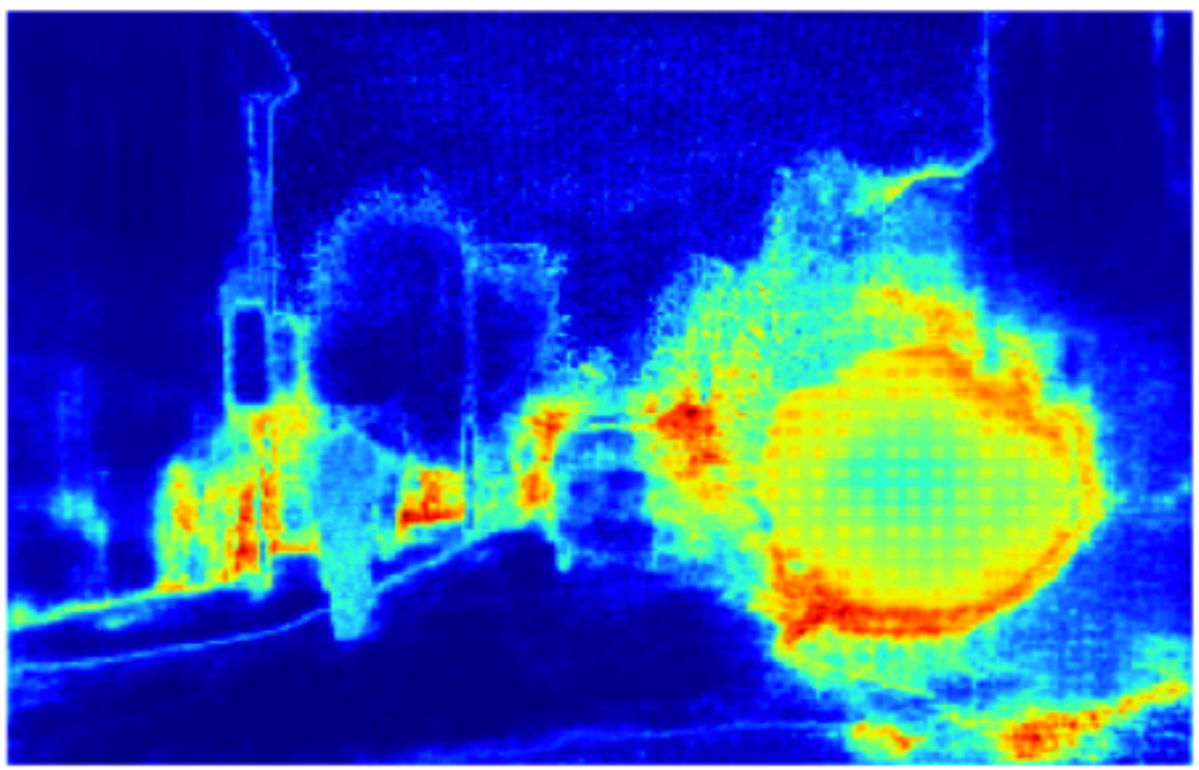





AI

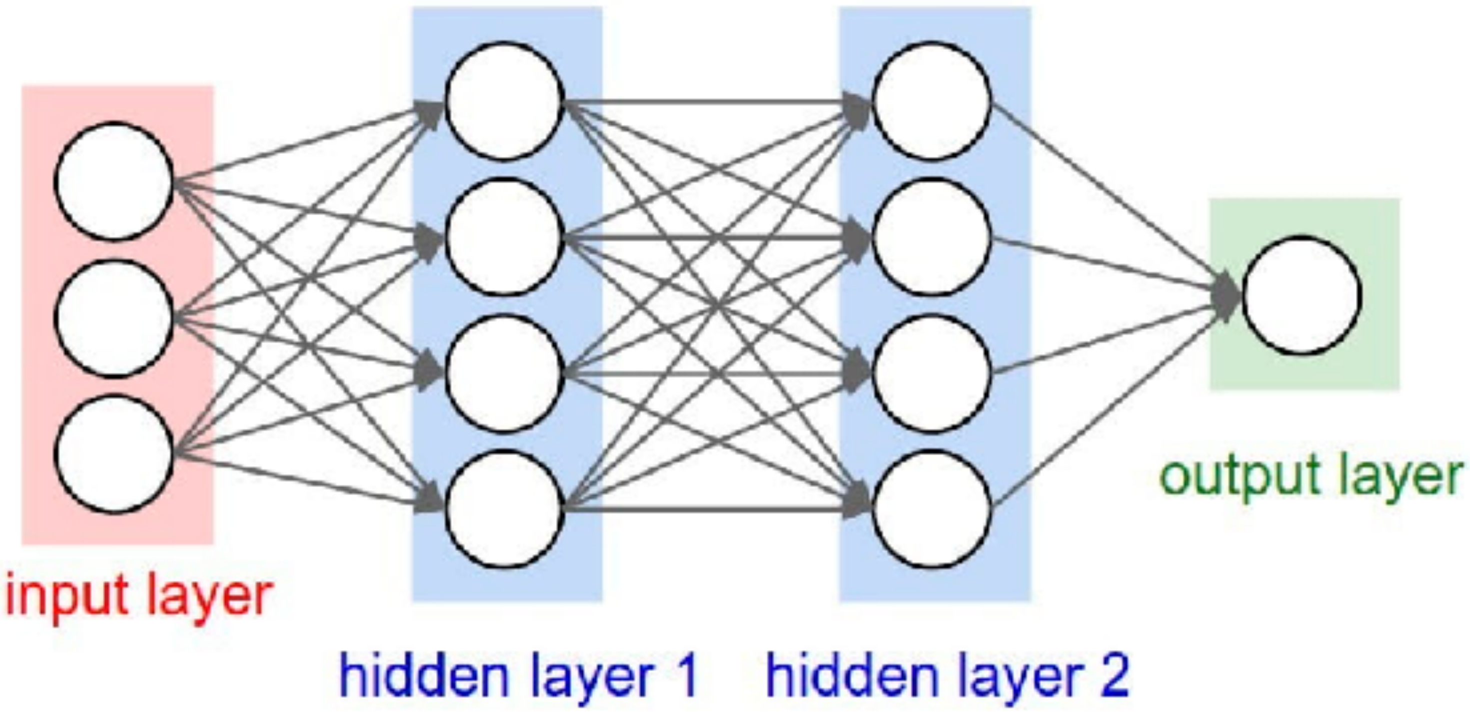
 

Images from Kendall and Gal, "What uncertainties do we need in Bayesian deep learning for computer vision?", *NeurIPS 2017*.

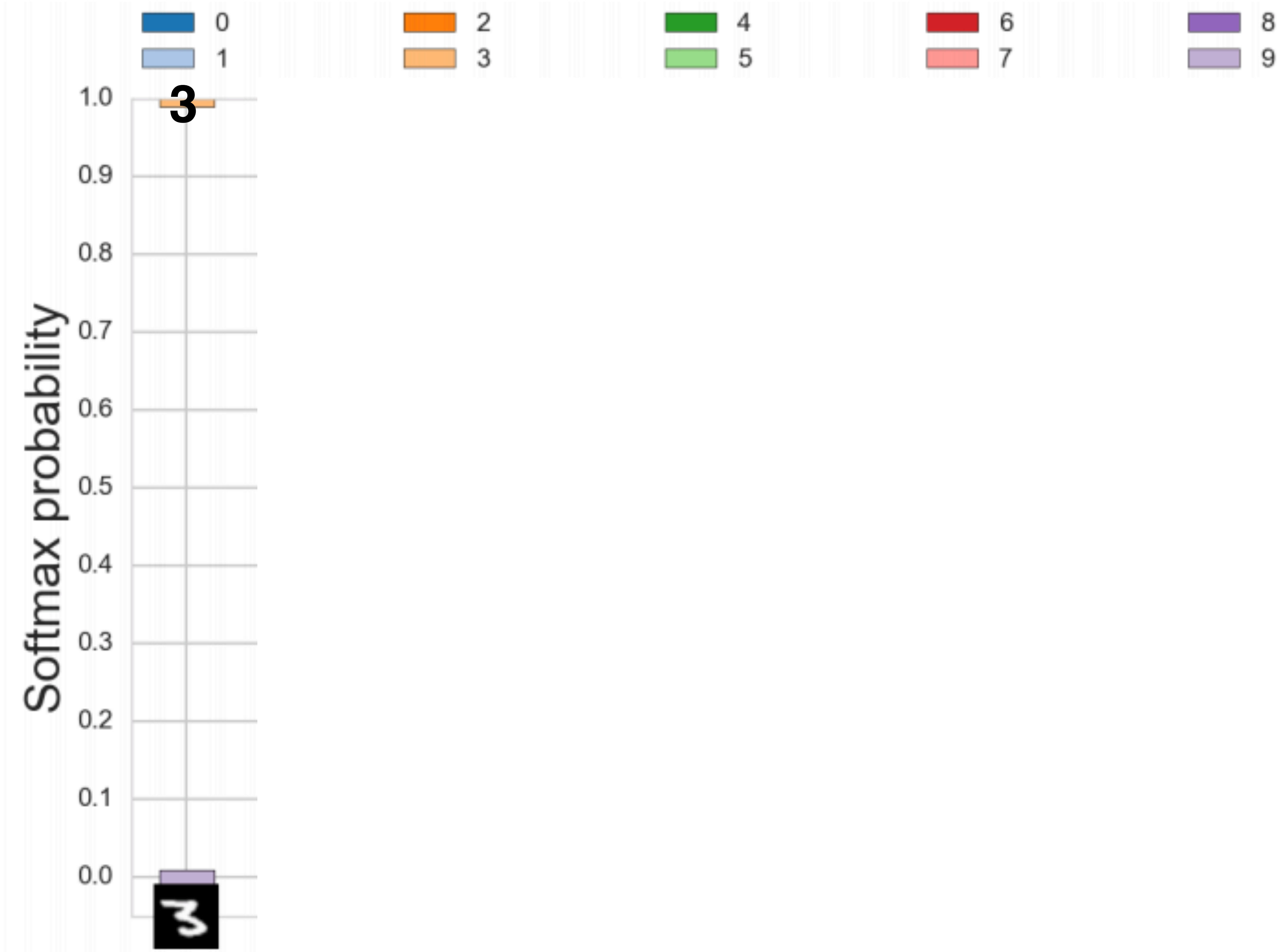
 

Images from Besnier et al., "Learning Uncertainty For Safety-Oriented Semantic Segmentation In Autonomous Driving", *ICIP 2021*.

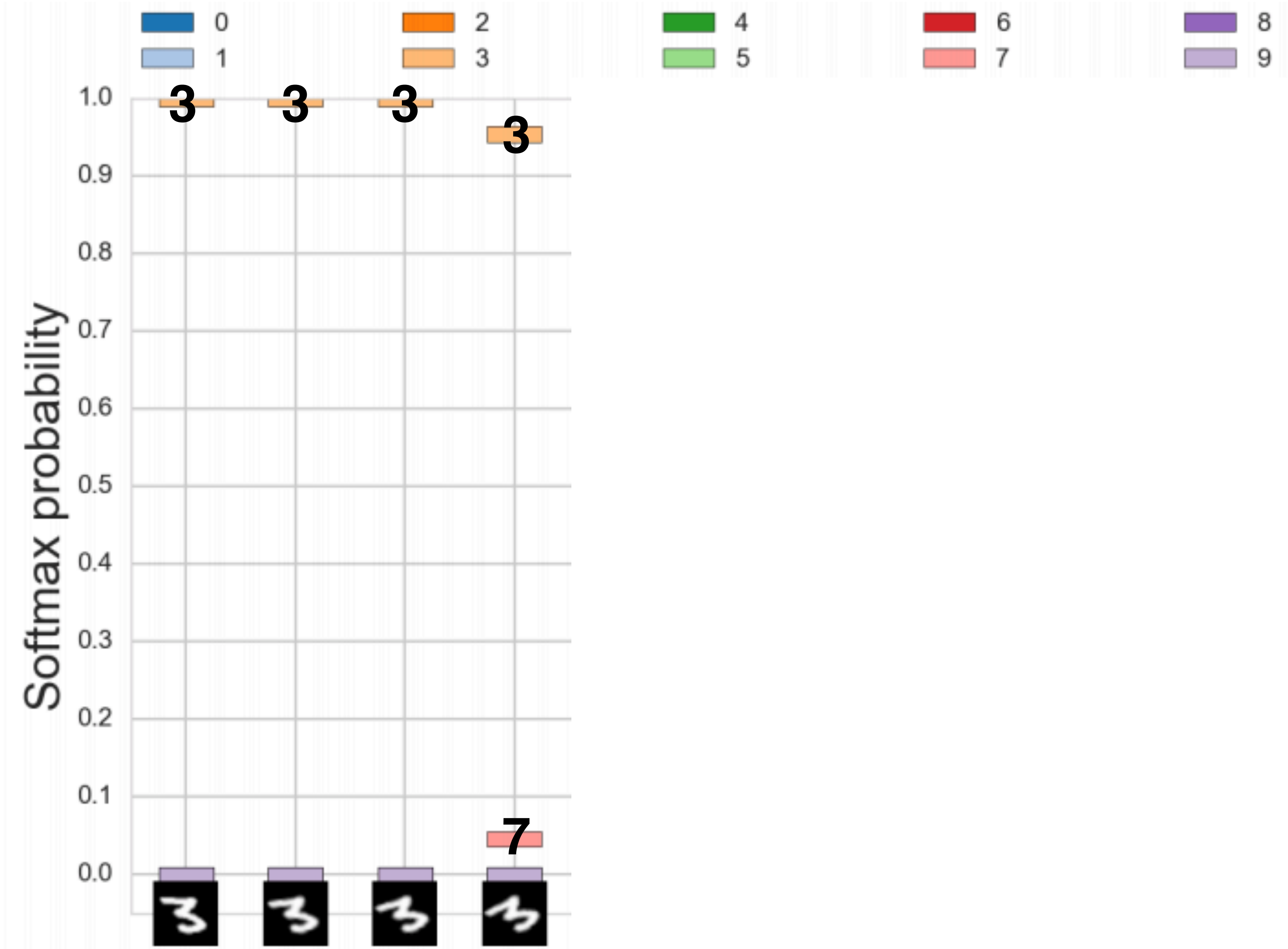
Neural Network



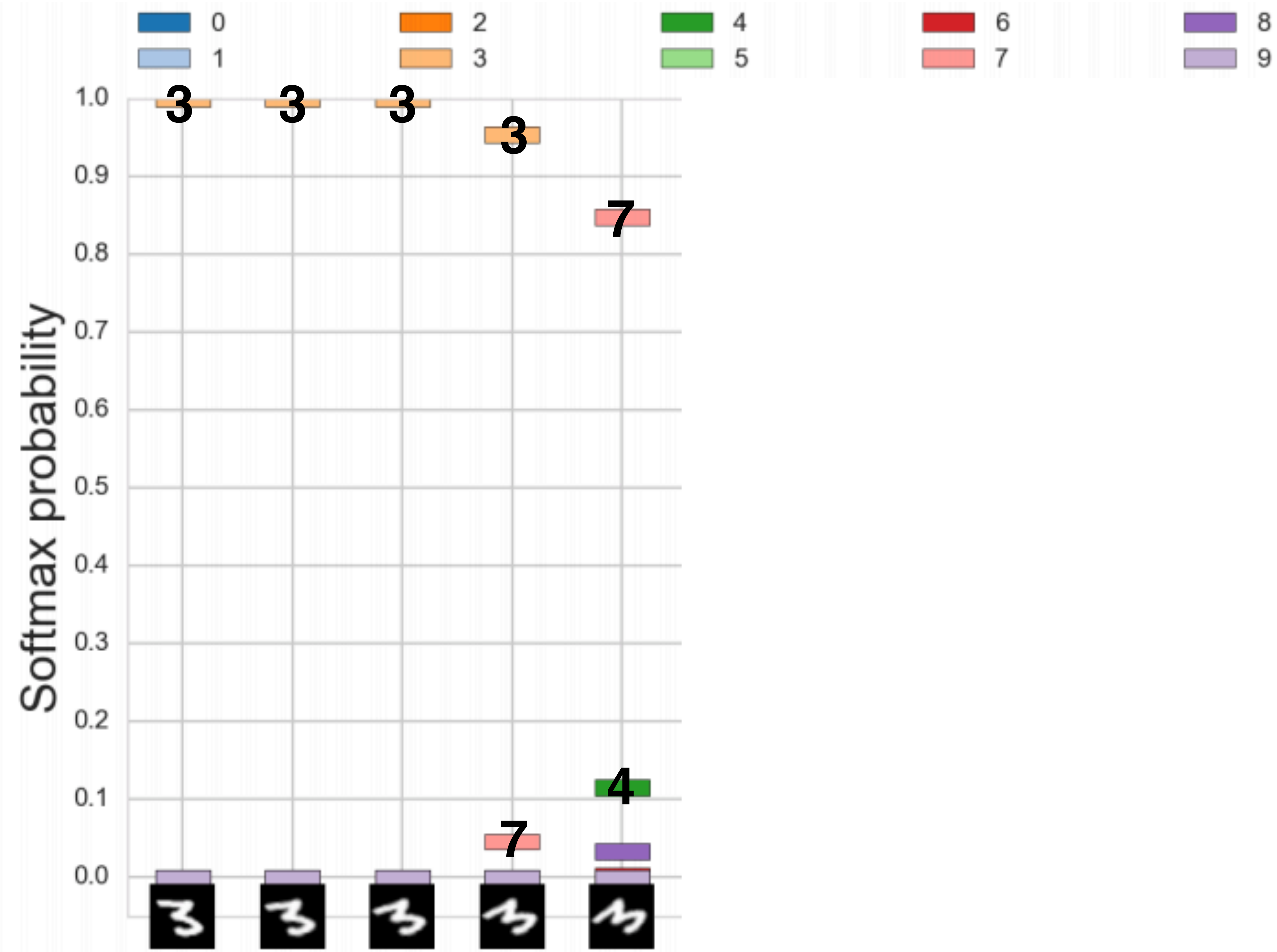
Data set of
hand-written digits



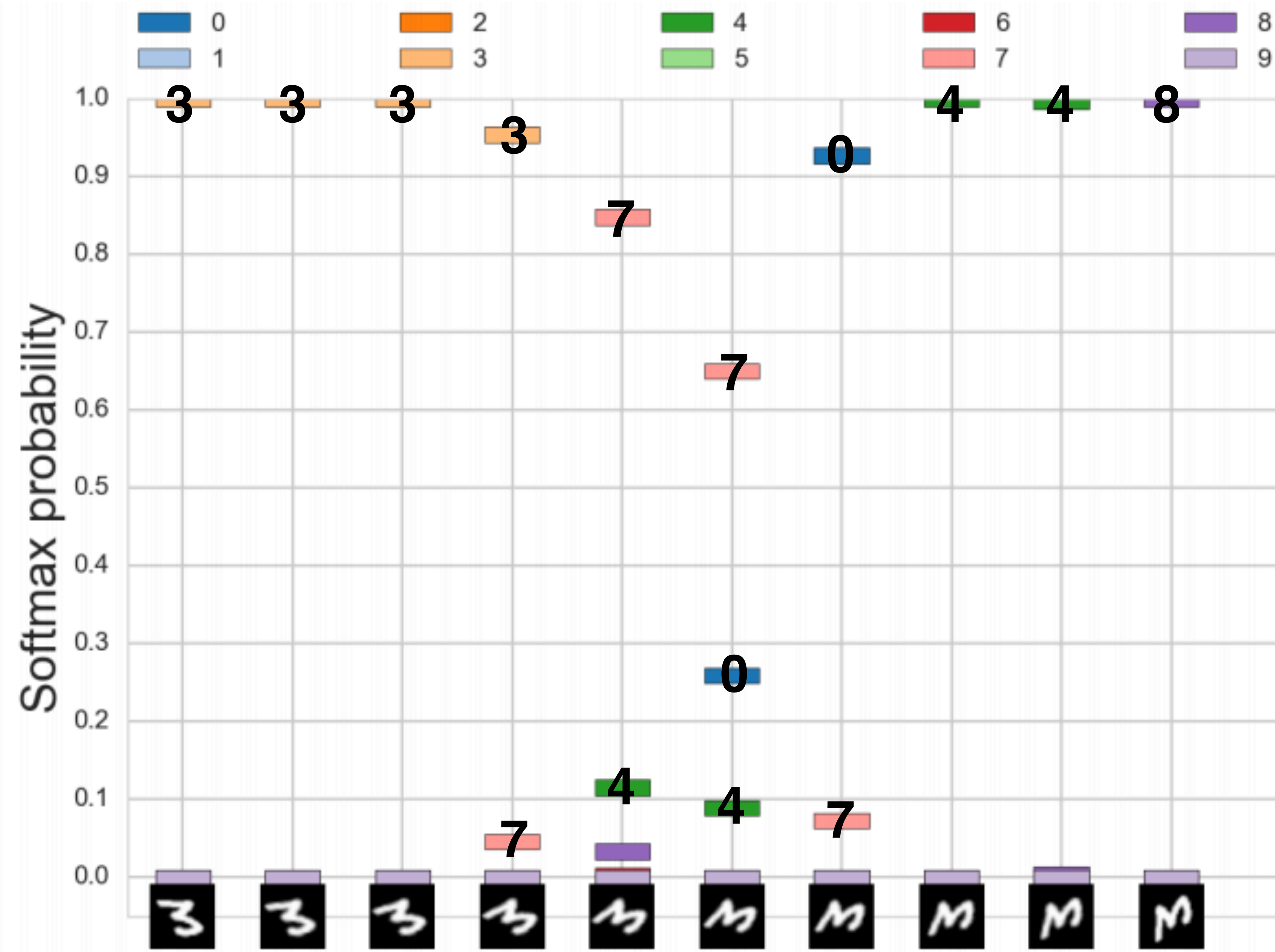
[Louizos & Welling, ICML 2017]



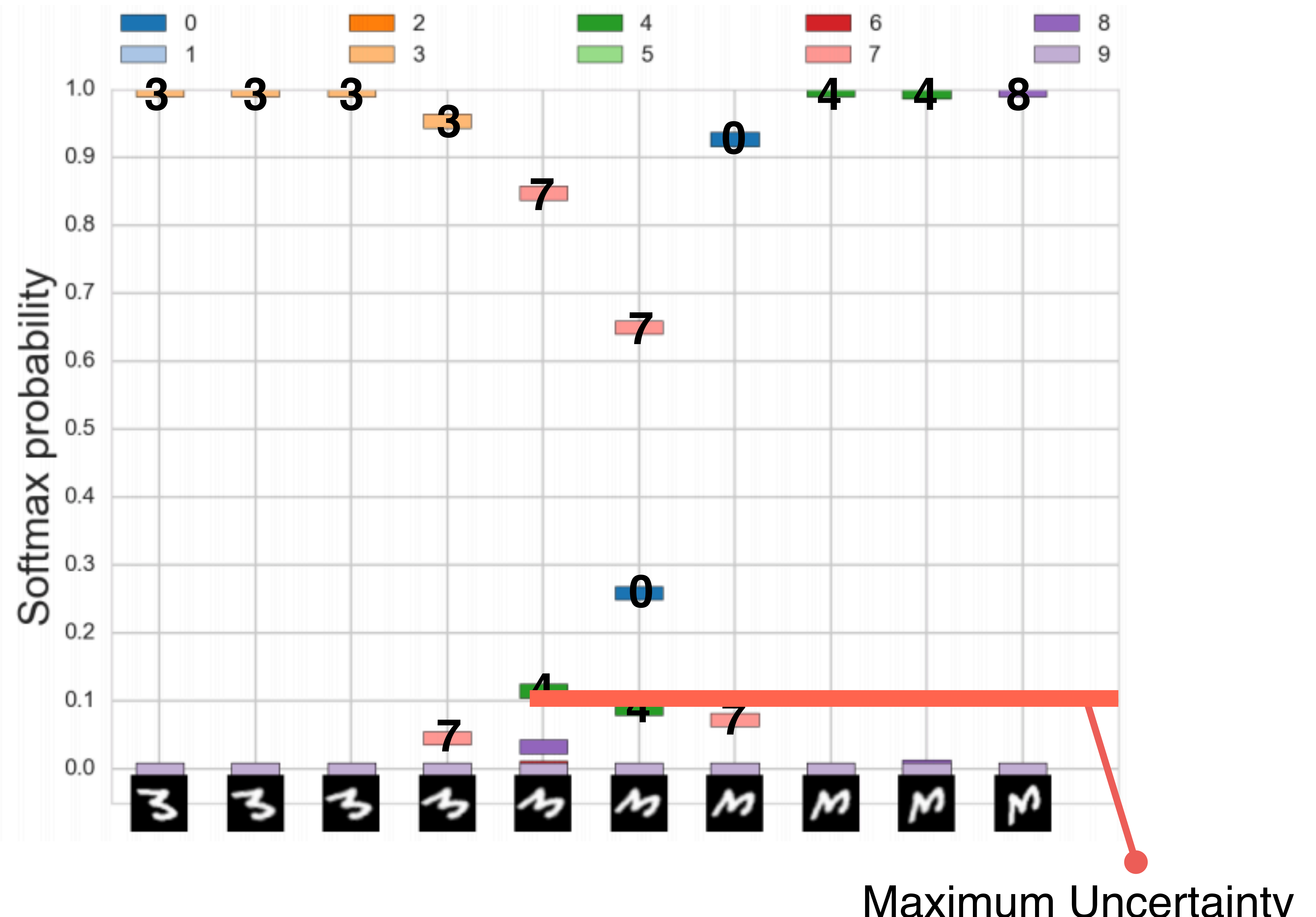
[Louizos & Welling, ICML 2017]



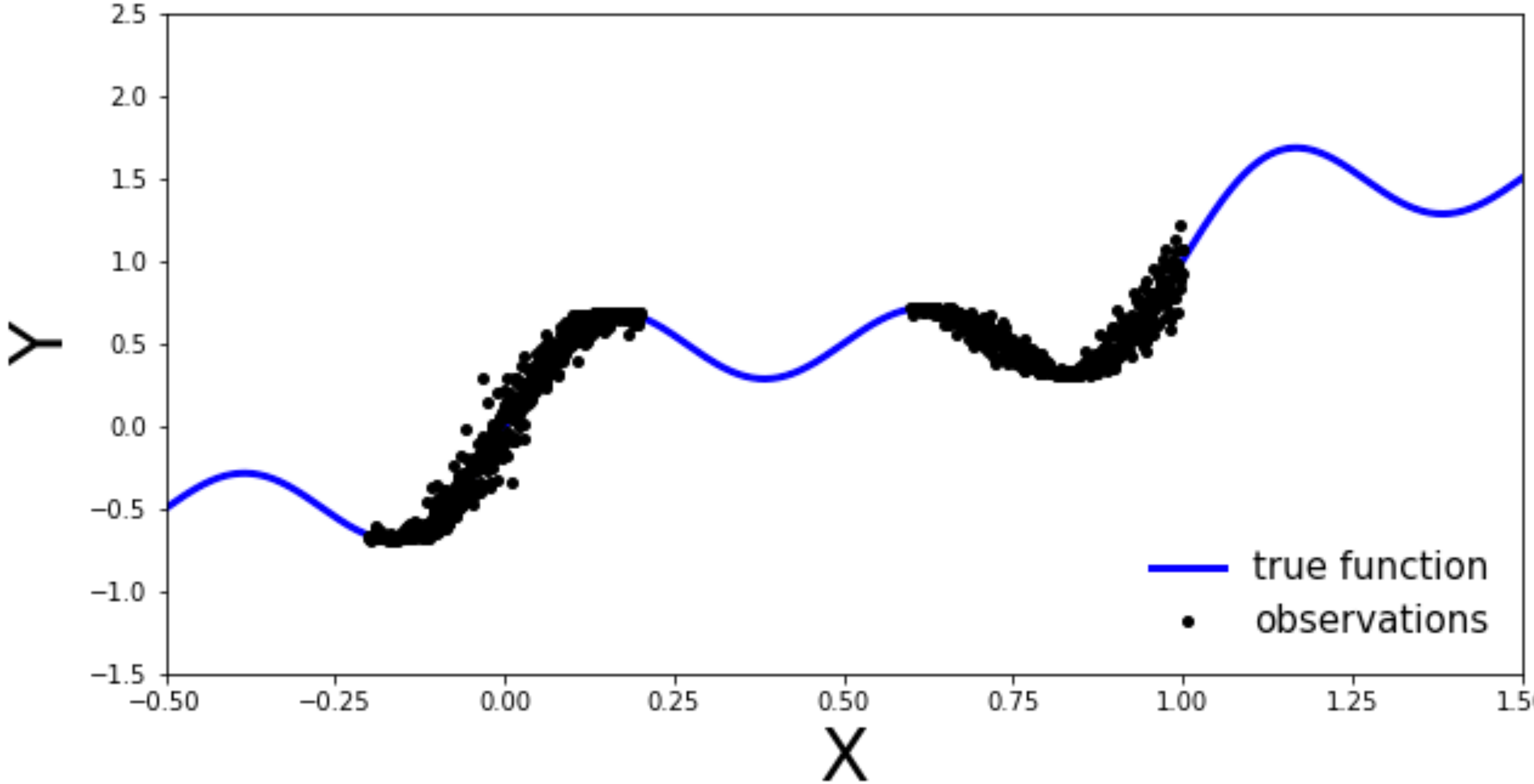
[Louizos & Welling, ICML 2017]

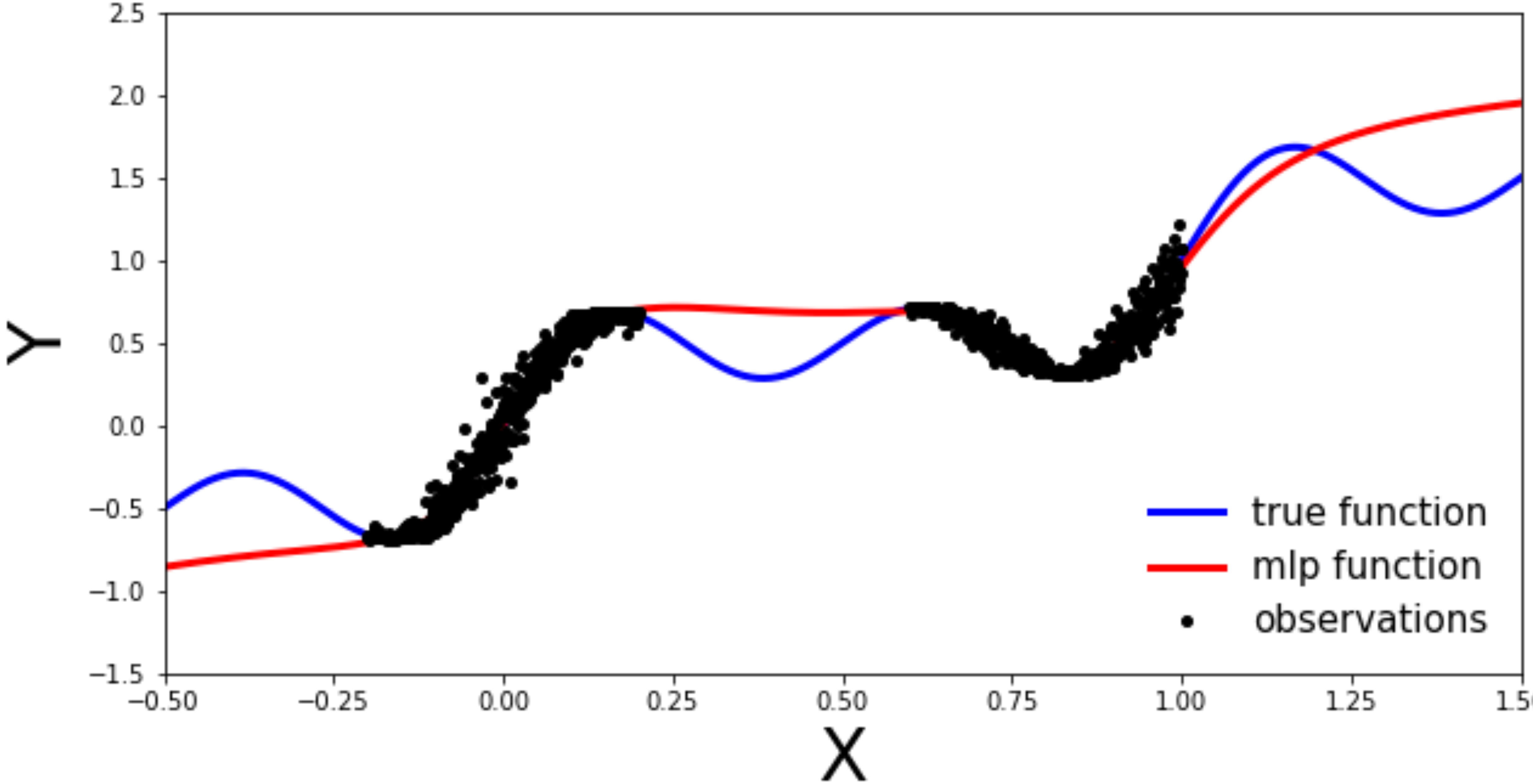


[Louizos & Welling, ICML 2017]

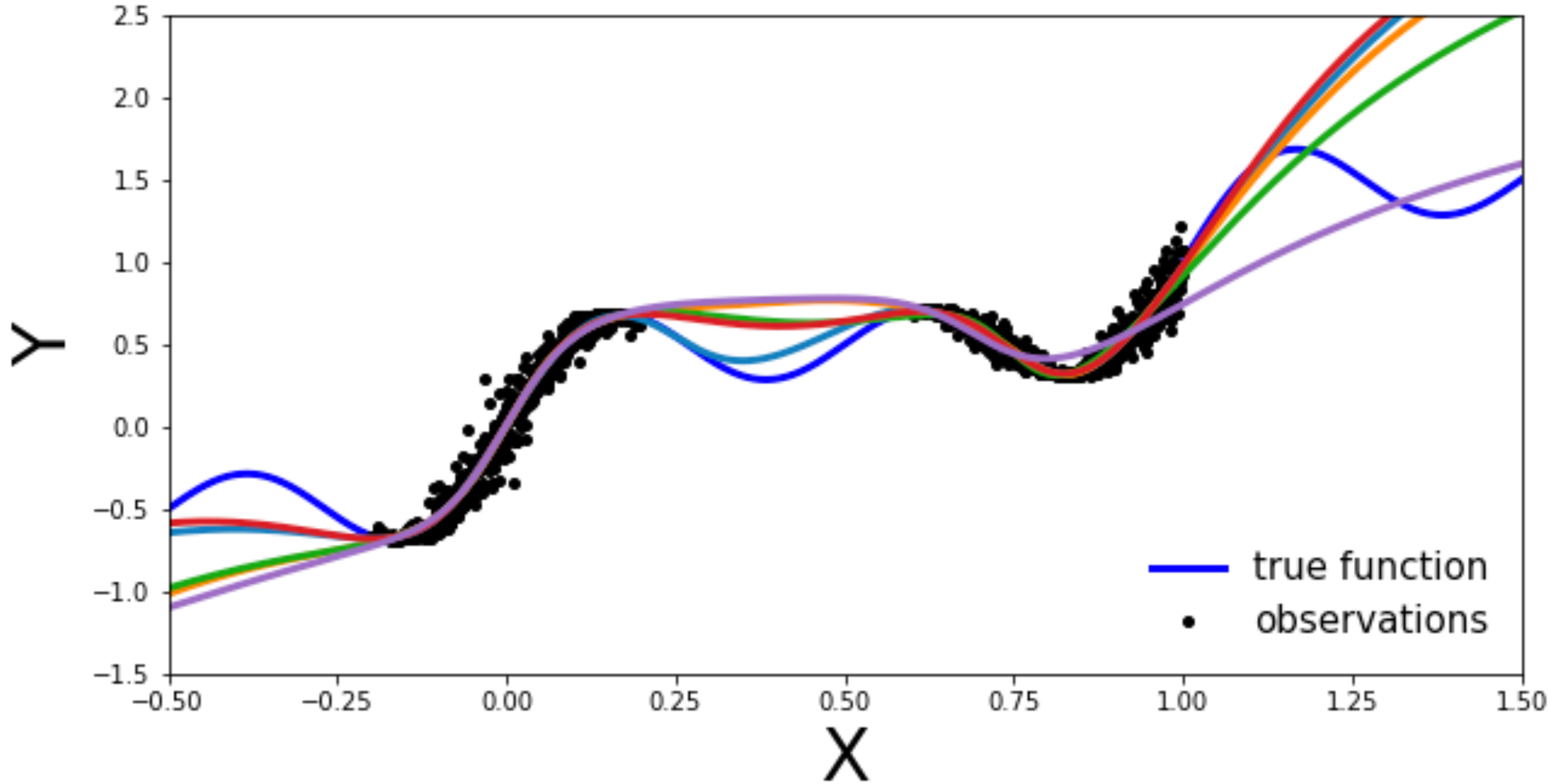


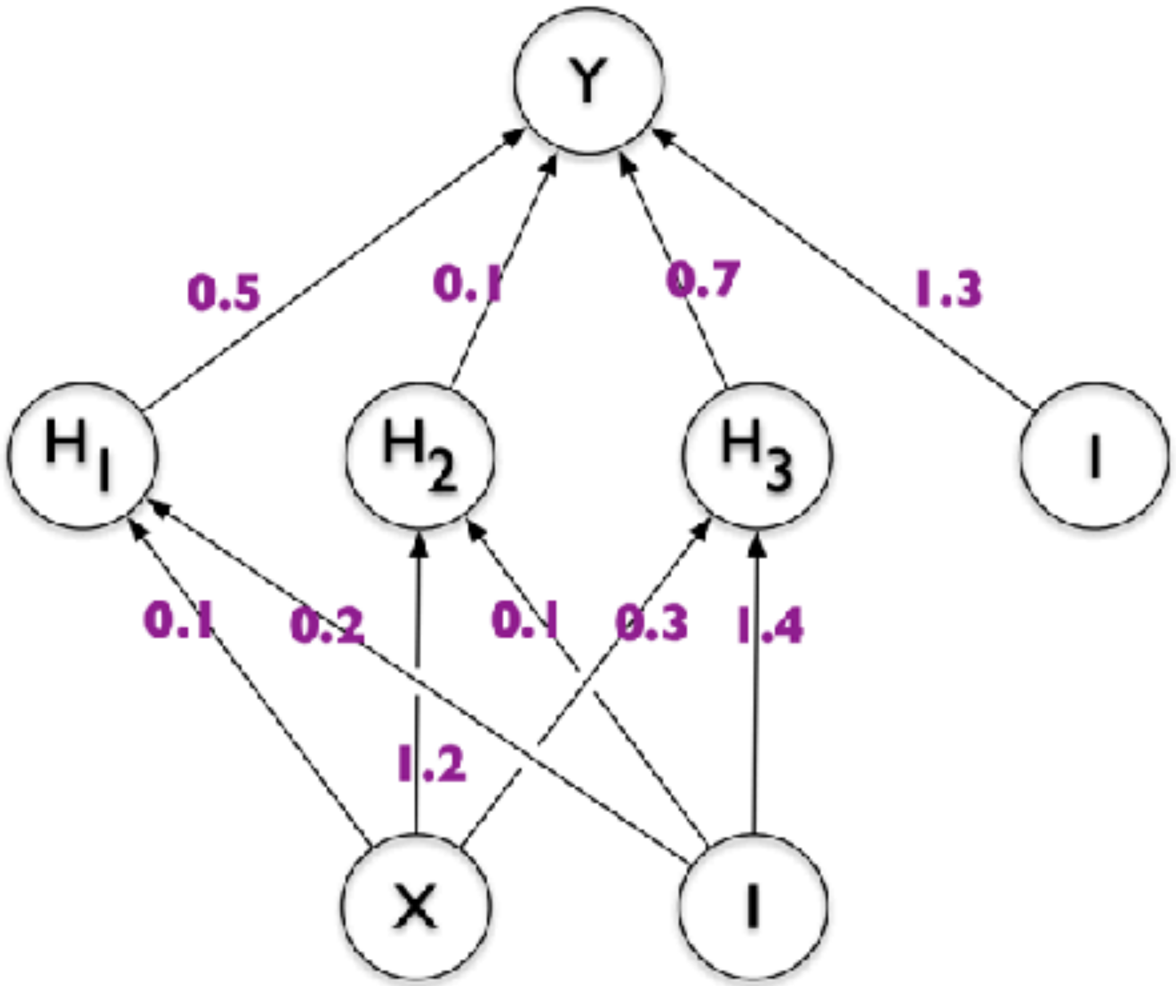
[Louizos & Welling, ICML 2017]


2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
Y
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
1.25
1.50
X
true function
observations



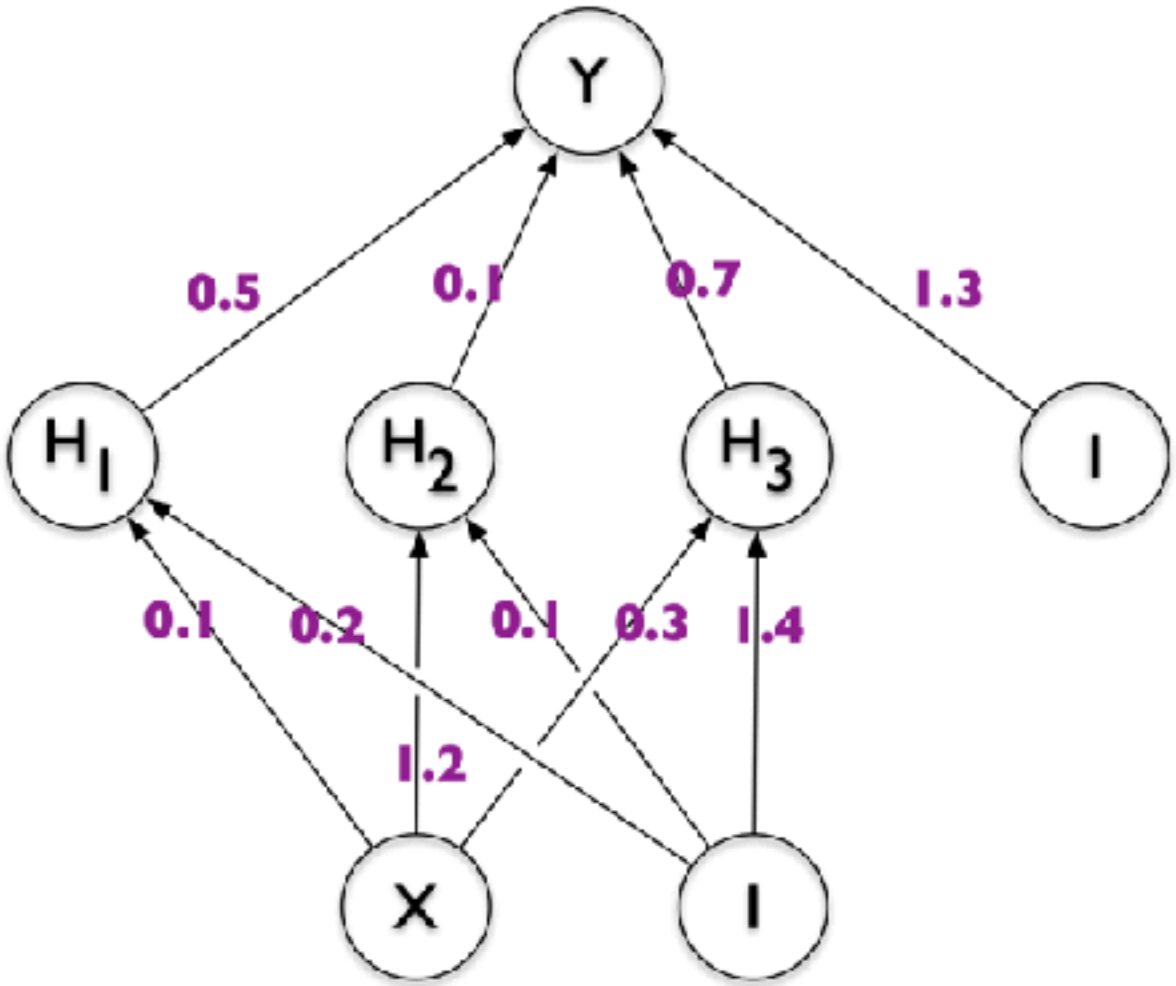
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
Y
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
1.25
1.50
X
true function
mlp function
observations


2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
Y
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
1.25
1.50
X
true function
observations

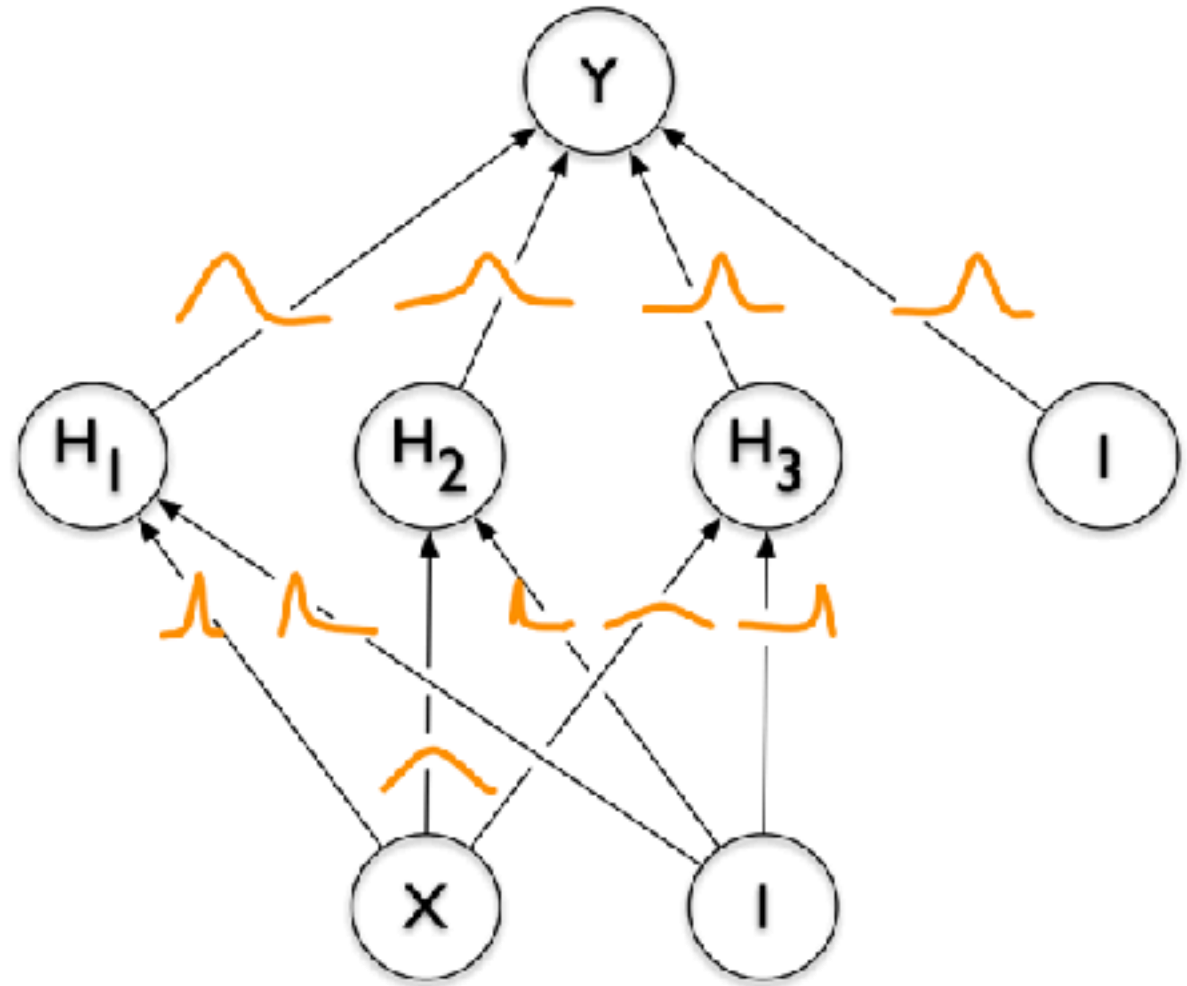


Traditional NN

Images from Blundell et al., “Weight Uncertainty in Neural Networks”, *ICML 2015*.



Traditional NN

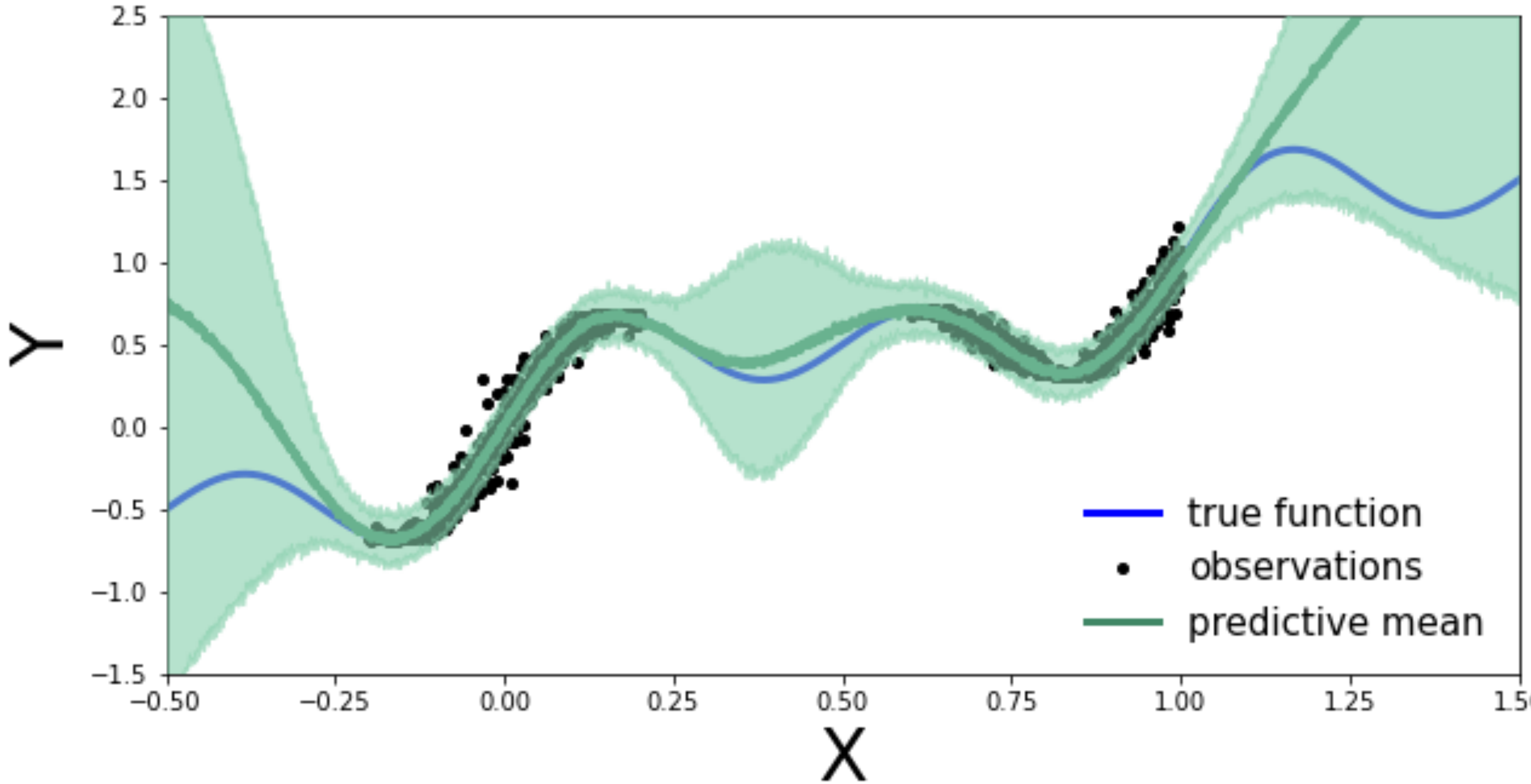


Bayesian NN

Images from Blundell et al., “Weight Uncertainty in Neural Networks”, *ICML 2015*.

$$p(y^* \mid x^*, \mathcal{D}) = \int_w p(y^* \mid x^*, w)\; p(w \mid \mathcal{D})\; dw$$

$$p(y^* \mid x^*, D) = \int_w p(y^* \mid x^*, w)\; p(w \mid D)\; dw$$



Model Definition

Data model

$$\mathsf{y} \sim p\left(\mathsf{y} \,|\, \mathsf{x}, \mathsf{W}_1, \ldots, \mathsf{W}_L\right)$$

Assume NNs are fully-connected, feedforward, unless stated otherwise.

Data model

$$\mathrm{y} \sim p\left(\mathrm{y} \mid \mathrm{x}, \mathrm{W}_1, \ldots, \mathrm{W}_L\right)$$

Assume NNs are fully-connected, feedforward, unless stated otherwise.

For real-valued regression…

$$\mathrm{y} \sim \mathrm{N}\left(\mu = f\left(\mathrm{x}; \mathrm{W}_1, \mathrm{W}_2, \mathrm{W}_3\right), \sigma_0^2\right)$$

Data model

$$\mathsf{y} \sim p\left(\mathsf{y} \,|\, \mathsf{x}, \mathsf{W}_1, \ldots, \mathsf{W}_L\right)$$

Assume NNs are fully-connected, feedforward, unless stated otherwise.

For classification…

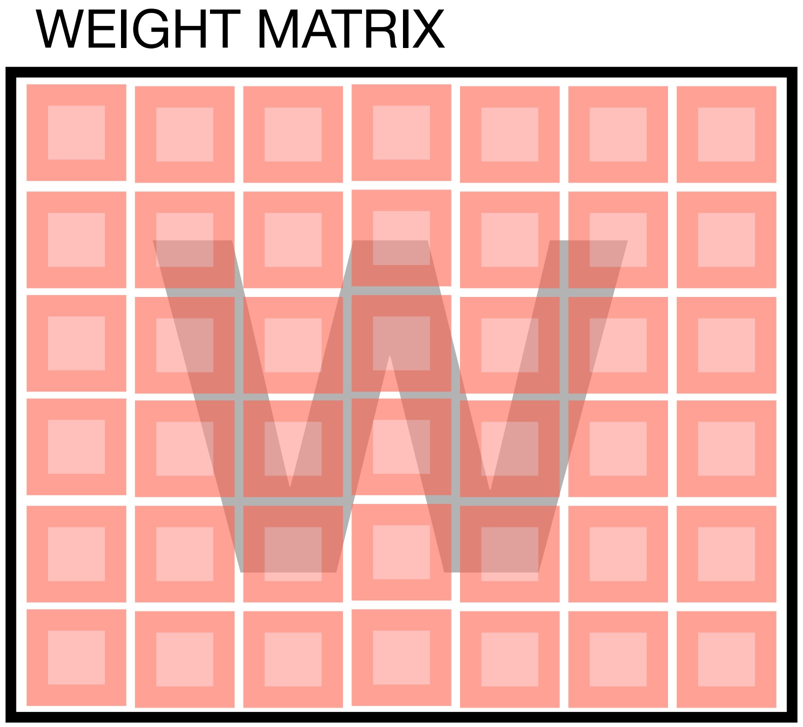
$$\mathsf{y} \sim \mathsf{Categorical}\left(\pi = f\left(\mathsf{x}; \mathsf{W}_1, \mathsf{W}_2, \mathsf{W}_3\right)\right)$$

Data model

$$\mathrm{y} \sim p\left(\mathrm{y} \,|\, \mathrm{x}, \mathrm{W}_1, \ldots, \mathrm{W}_L\right)$$

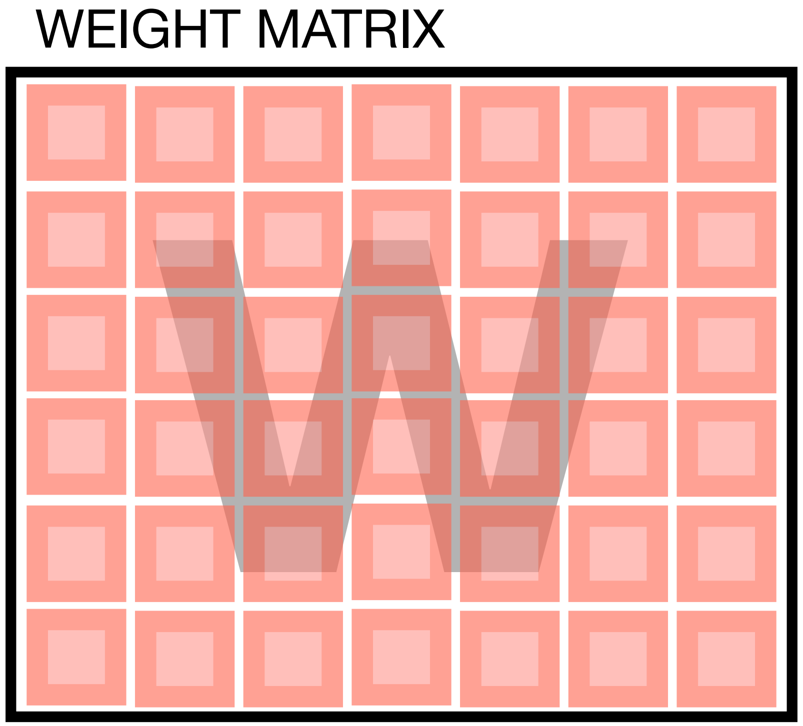
Prior per weight

$$\mathbf{w} \sim p(\mathbf{w})$$



Prior per weight

$$\mathbf{w} \sim \mathrm{N}(0, \sigma^2)$$

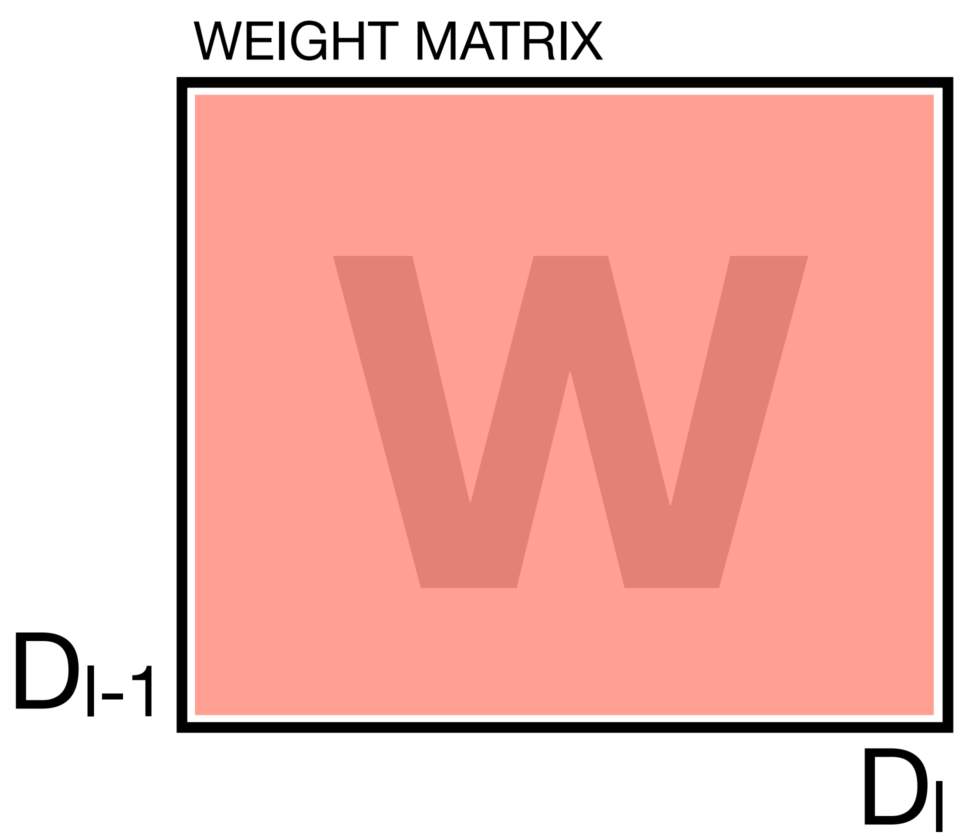


Prior per layer

$$\mathrm{W}_l \sim p(\mathrm{W}_l)$$

Prior per layer

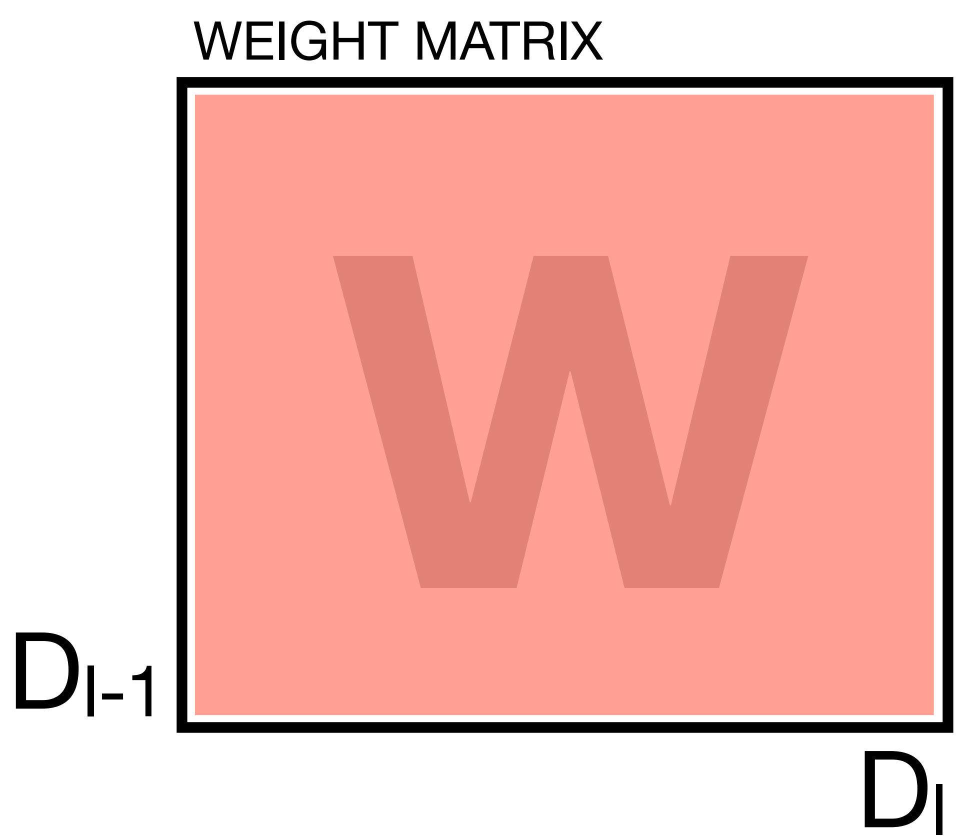
$$\mathbf{W}_l \sim \mathbf{N}(\mathbf{0}, \boldsymbol{\Sigma})$$



Prior per layer

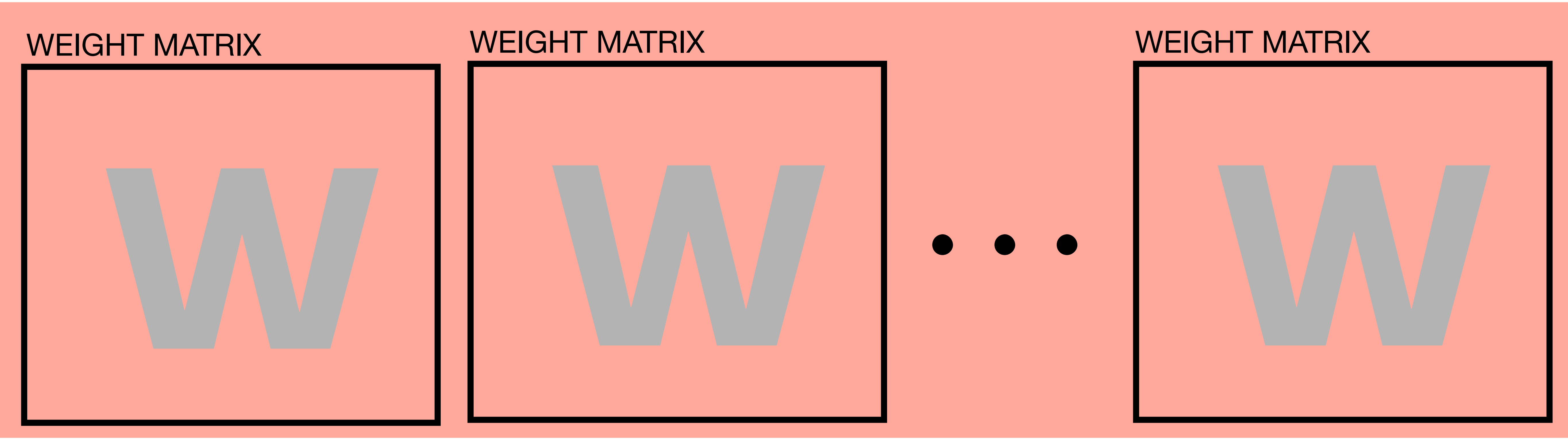
$$W_l \sim N(\mathbf{0}, \Sigma)$$

Size: $(D_{l-1} + D_l)^2$



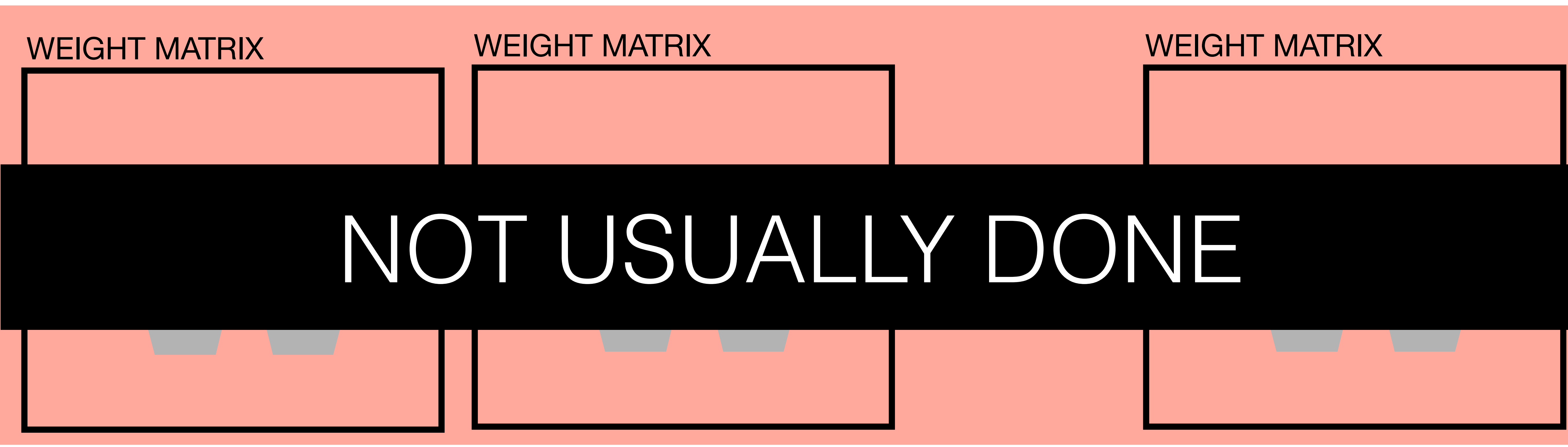
Joint prior

$$\mathrm{W}_1, \ldots, \mathrm{W}_L \sim p\left(\mathrm{W}_1, \ldots, \mathrm{W}_L\right)$$



Joint prior

$$W_1, \ldots, W_L \sim p\left(W_1, \ldots, W_L\right)$$



Joint prior

$$\mathsf{W}_1, \ldots, \mathsf{W}_L \sim p\left(\mathsf{W}_1, \ldots, \mathsf{W}_L\right)$$

$$\mathsf{W}_1, \ldots, \mathsf{W}_L \sim \mathbf{N}(\mathbf{0}, \Sigma)$$

Size: (# total weights)^2

Posterior

$$p\left(\mathsf{W}_1, \ldots, \mathsf{W}_L \,|\, \mathsf{y}, \mathsf{x}\right) =$$

Posterior

$$p\left(\mathsf{W}_1, \ldots, \mathsf{W}_L \,|\, \mathsf{y}, \mathsf{x}\right) =$$

$$\frac{p\left(\mathsf{y} \,|\, \mathsf{x}, \mathsf{W}_1, \ldots, \mathsf{W}_L\right) \ \prod_{l=1}^{L} p(\mathsf{W}_l)}{p\left(\mathsf{y} \,|\, \mathsf{x}\right)}$$

Posterior

$$p\left(\mathsf{W}_1, \ldots, \mathsf{W}_L \,|\, \mathsf{y}, \mathsf{x}\right) =$$

$$\frac{p\left(\mathsf{y} \,|\, \mathsf{x}, \mathsf{W}_1, \ldots, \mathsf{W}_L\right) \ \prod_{l=1}^{L} p(\mathsf{W}_l)}{p\left(\mathsf{y} \,|\, \mathsf{x}\right)}$$

Posterior

$$p\left(\mathsf{W}_1, \ldots, \mathsf{W}_L \,|\, \mathsf{y}, \mathsf{x}\right) =$$

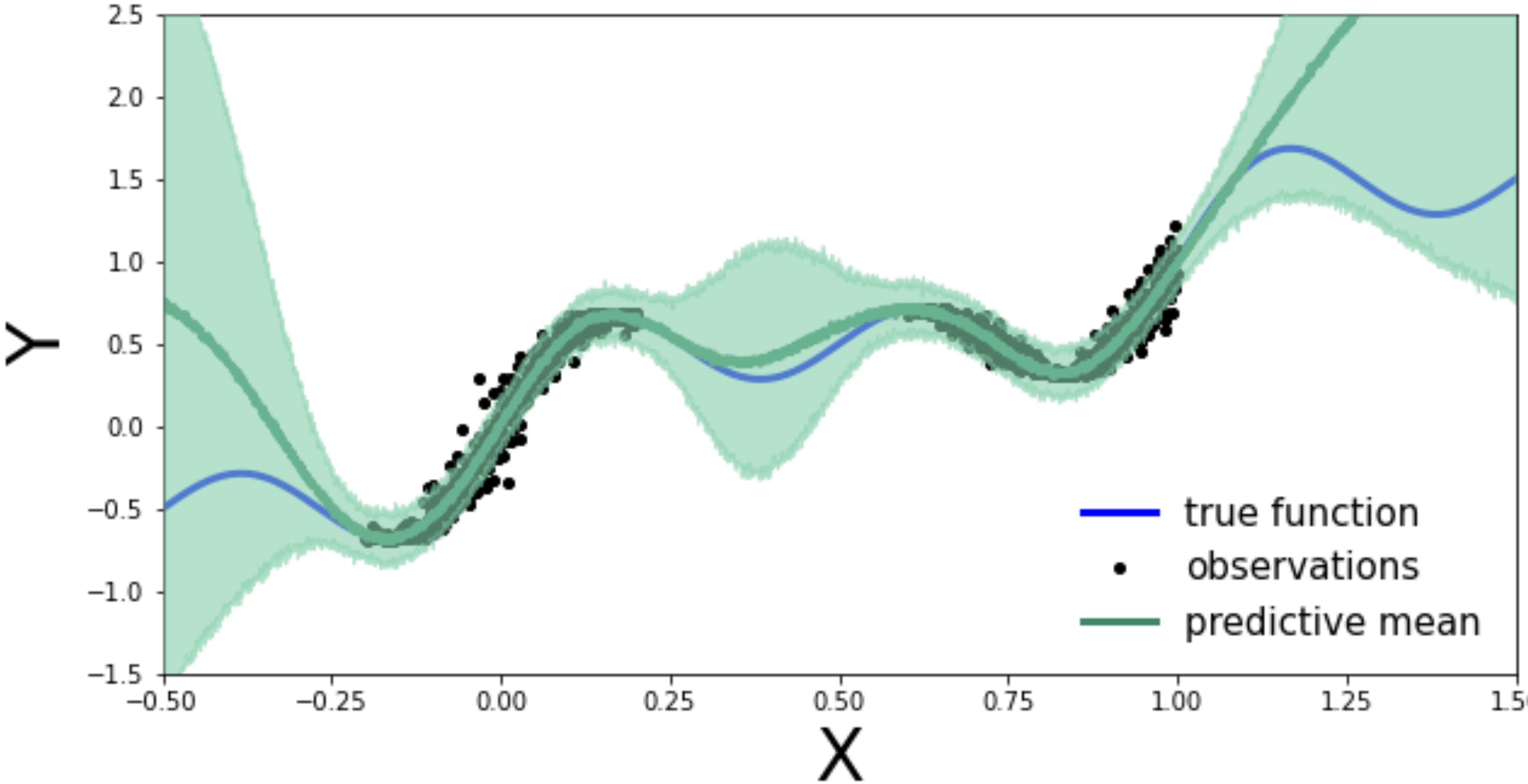
$$\frac{p\left(\mathsf{y} \,|\, \mathsf{x}, \mathsf{W}_1, \ldots, \mathsf{W}_L\right) \; \prod_{l=1}^{L} p(\mathsf{W}_l)}{\boxed{\int_{\mathsf{W}_1, \ldots, \mathsf{W}_L} \mathsf{p}(\mathsf{y} \,|\, \mathsf{x}, \mathsf{W}_1, \ldots, \mathsf{W}_L) \; \prod_l \mathsf{p}(\mathsf{W}_l) \; \mathsf{d}\mathsf{W}_1, \ldots, \mathsf{W}_L}}$$

Posterior Predictive

$$p(y^* \mid x^*, y, x) =$$

$$\int_{W_1,\ldots,W_L} p(y^* \mid x^*, W_1, \ldots, W_L)\ p(W_1, \ldots, W_L \mid y, x)\ dW_1, \ldots, W_L$$

$$p(y^* \mid x^*, D)$$

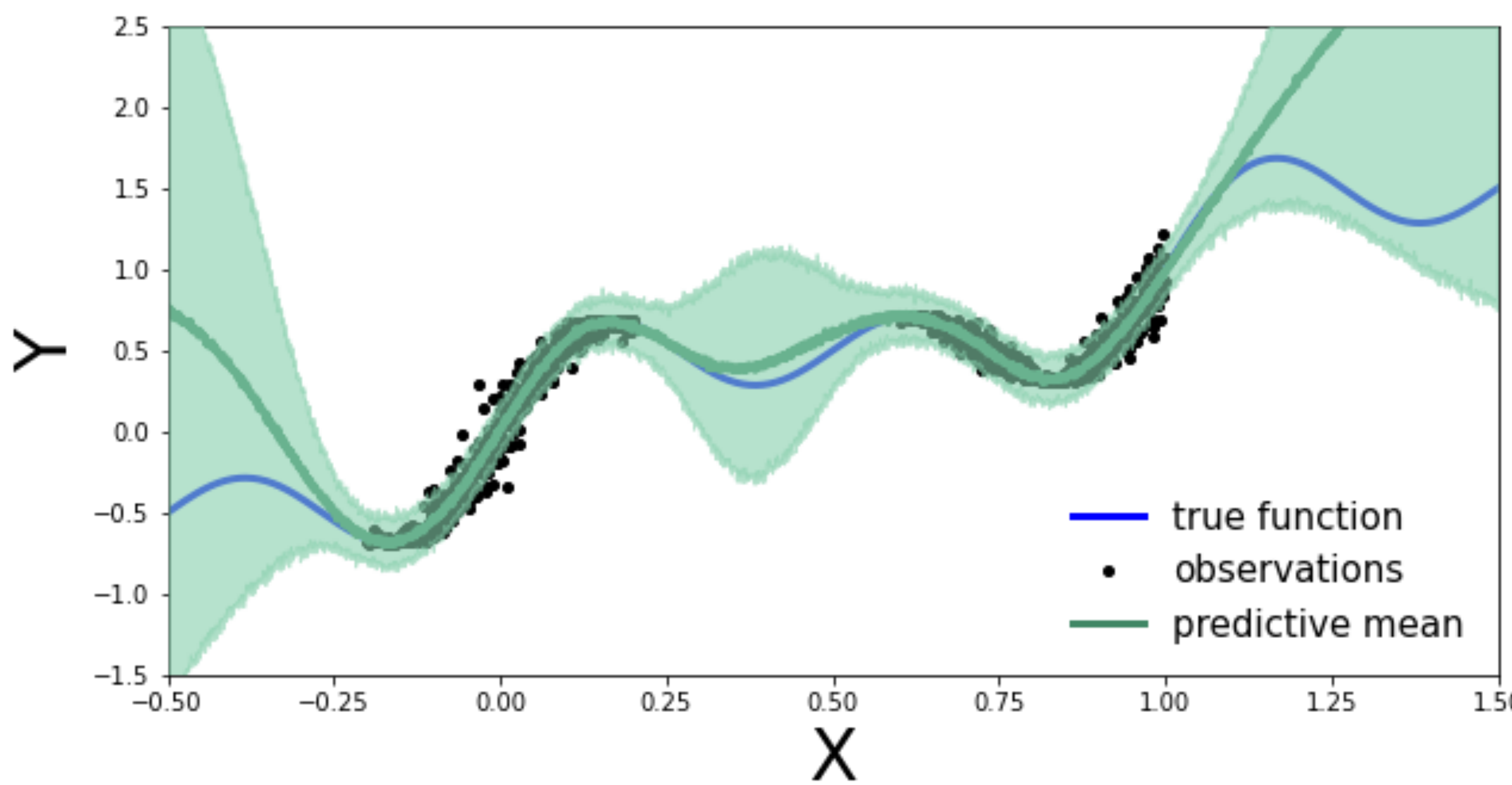


2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
Y
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
1.25
1.50
X
true function
observations
predictive mean

$$\mathrm{w} \sim \mathrm{N}(0,\ \sigma^2 = 5)$$

$$\sigma_0 \sim \mathrm{Gamma}\,(1/2,\ 1)$$

$$\mathrm{y} \sim \mathrm{N}\left(\mu = f\left(\mathrm{x}; \mathrm{W}_1, \mathrm{W}_2, \mathrm{W}_3\right), \sigma_0^2\right)$$



Bayesian Deep Learning:
Priors

Eric Nalisnick

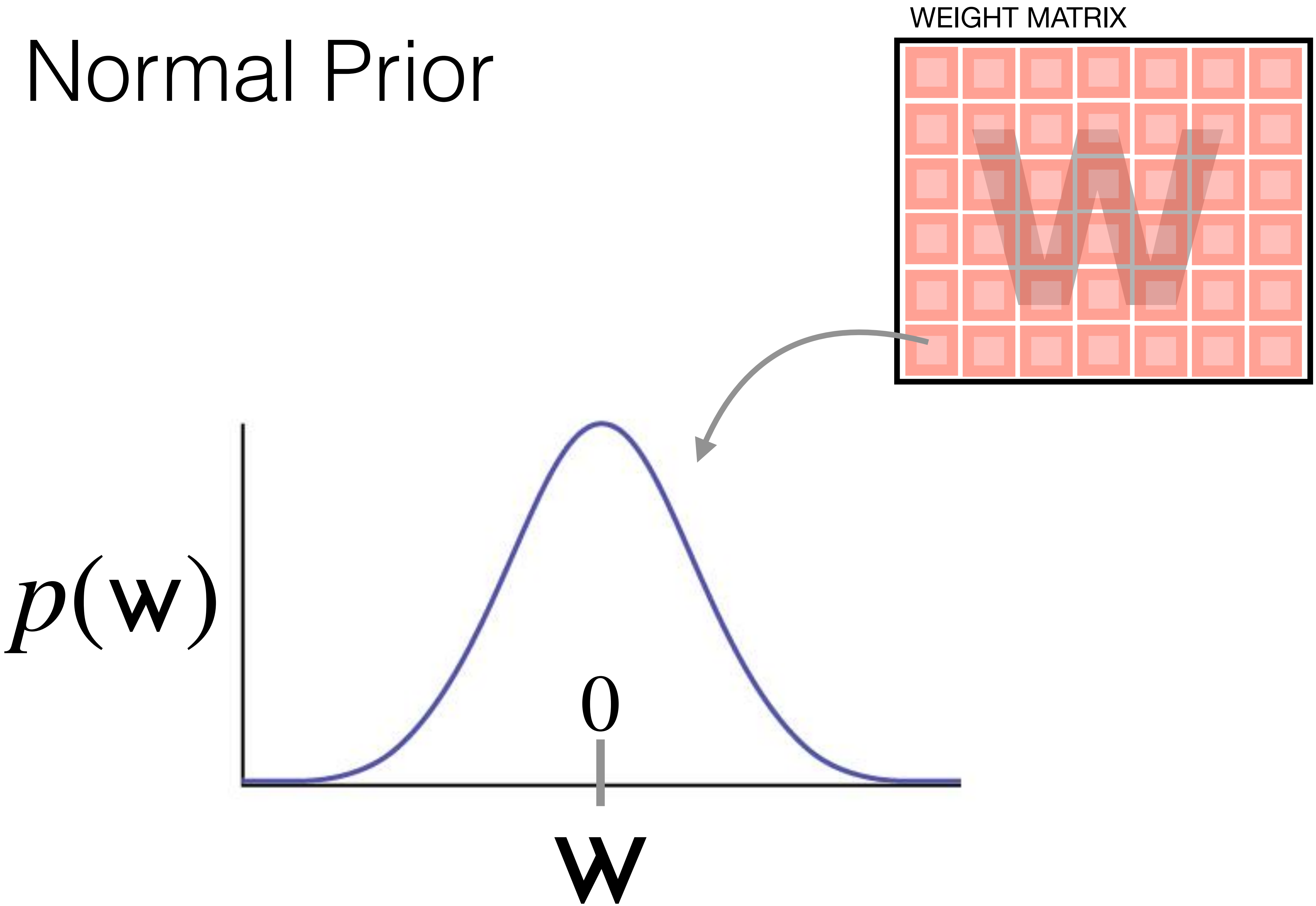
Deep Learning II,
University of Amsterdam

$$\frac{p\left(\mathsf{y} \mid \mathsf{x}, \mathsf{W}_1, \ldots, \mathsf{W}_L\right) \; \prod_{l=1}^{L} p(\mathsf{W}_l)}{p\left(\mathsf{y} \mid \mathsf{x}\right)}$$

Garbage in: *arbitrary priors*
Garbage out: *uncontrollable error bars*

Michael I. Jordan, *MLSS* (2017)

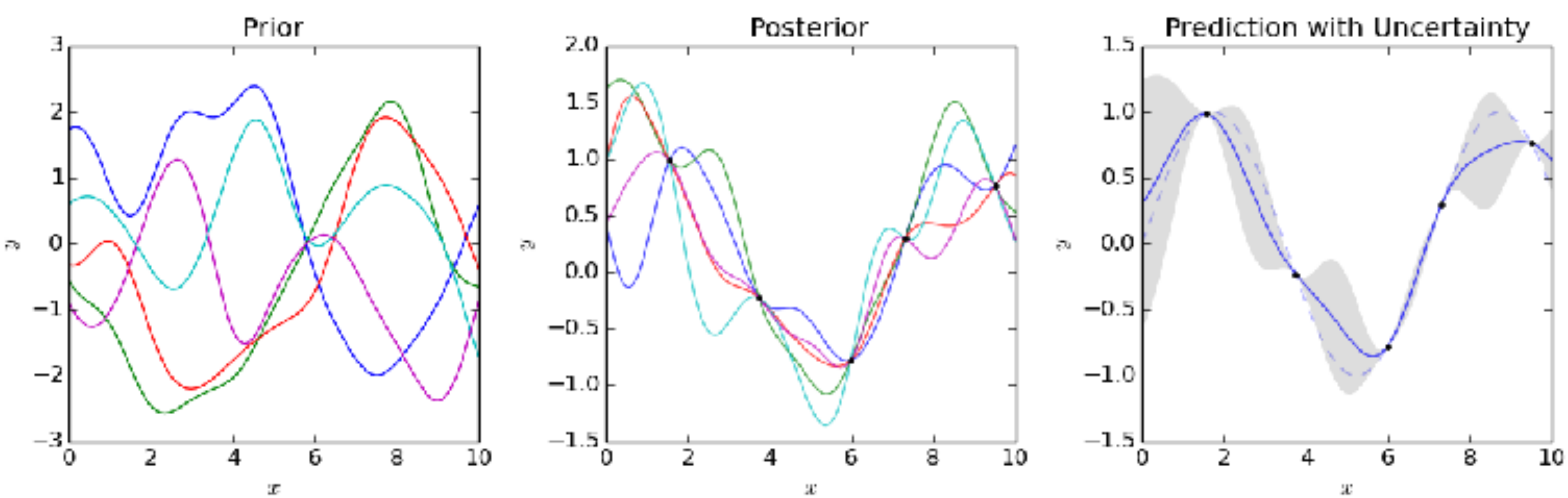
Normal Prior



Normal Prior

As NN becomes infinitely wide, it converges to a *Gaussian process*

$$\mathbf{w} \sim \mathbf{N}(0, \sigma^2/H)$$



https://en.wikipedia.org/wiki/Gaussian_process#/media/File:Gaussian_Process_Regression.png

Normal Prior

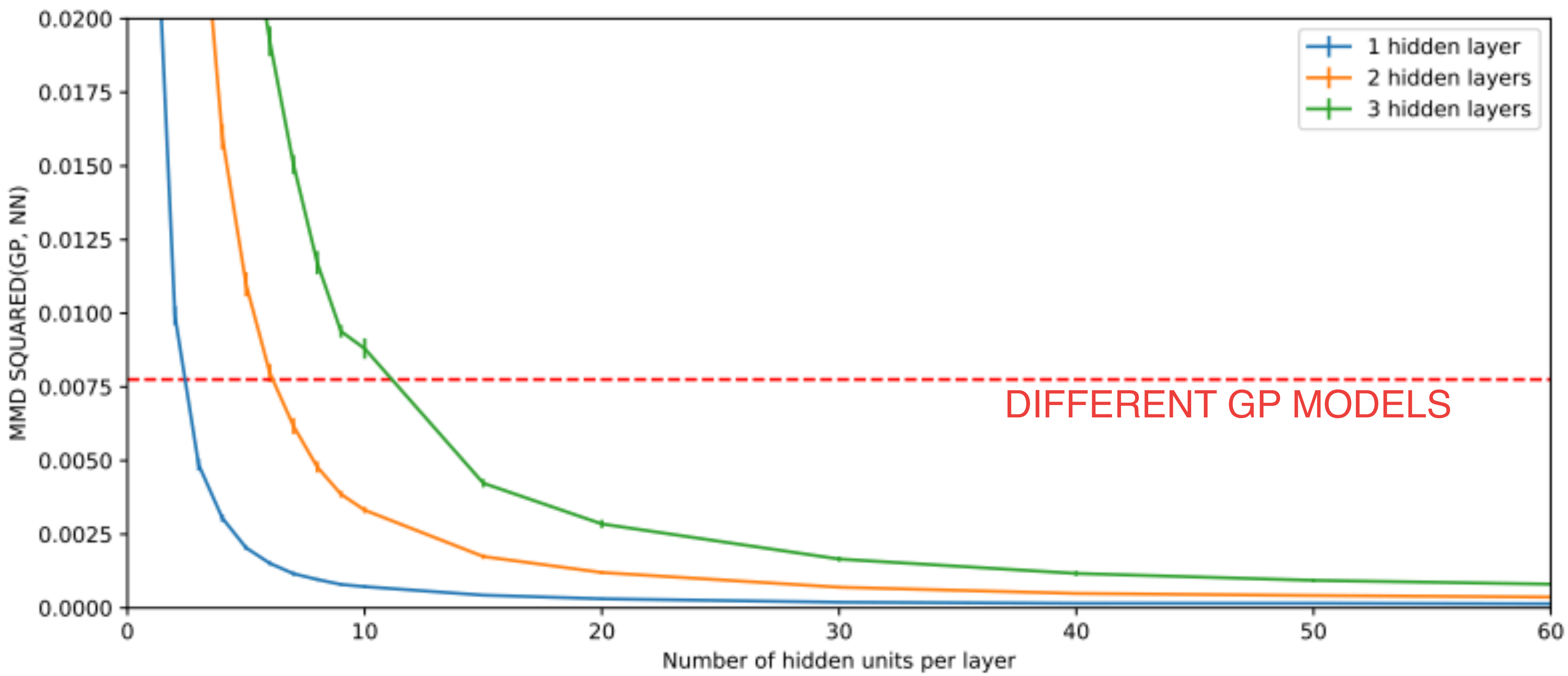
As NN becomes infinitely wide, it converges to a *Gaussian process*

$$\mathbf{w} \sim \mathrm{N}(0, \sigma^2/H)$$

"With Gaussian priors the contributions of individual units are all negligible, and consequently, these units do not represent 'hidden features' that capture important aspects of the data" [Neal, 1995]

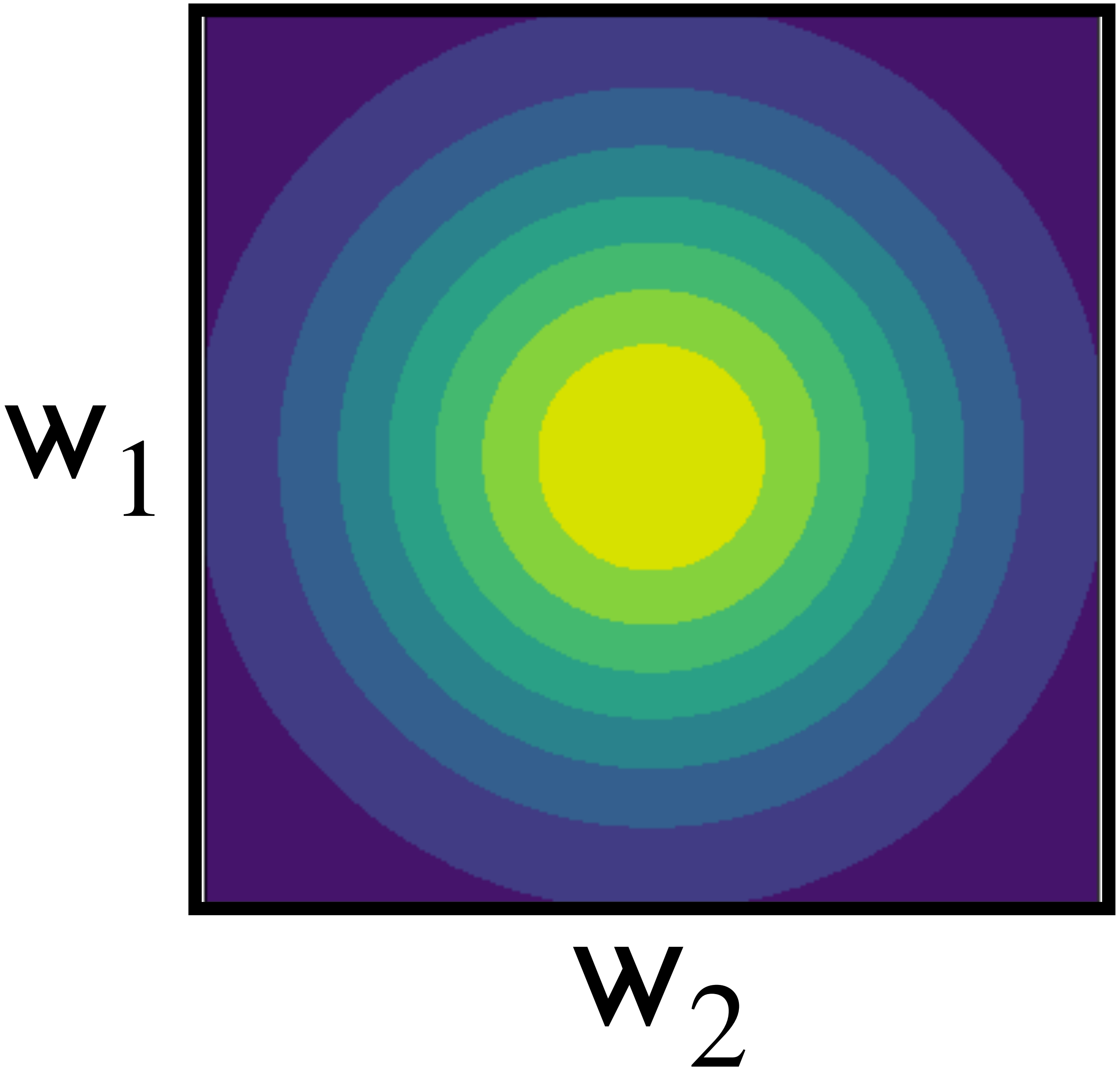
Normal Prior

As NN becomes infinitely wide, it converges to a *Gaussian process*

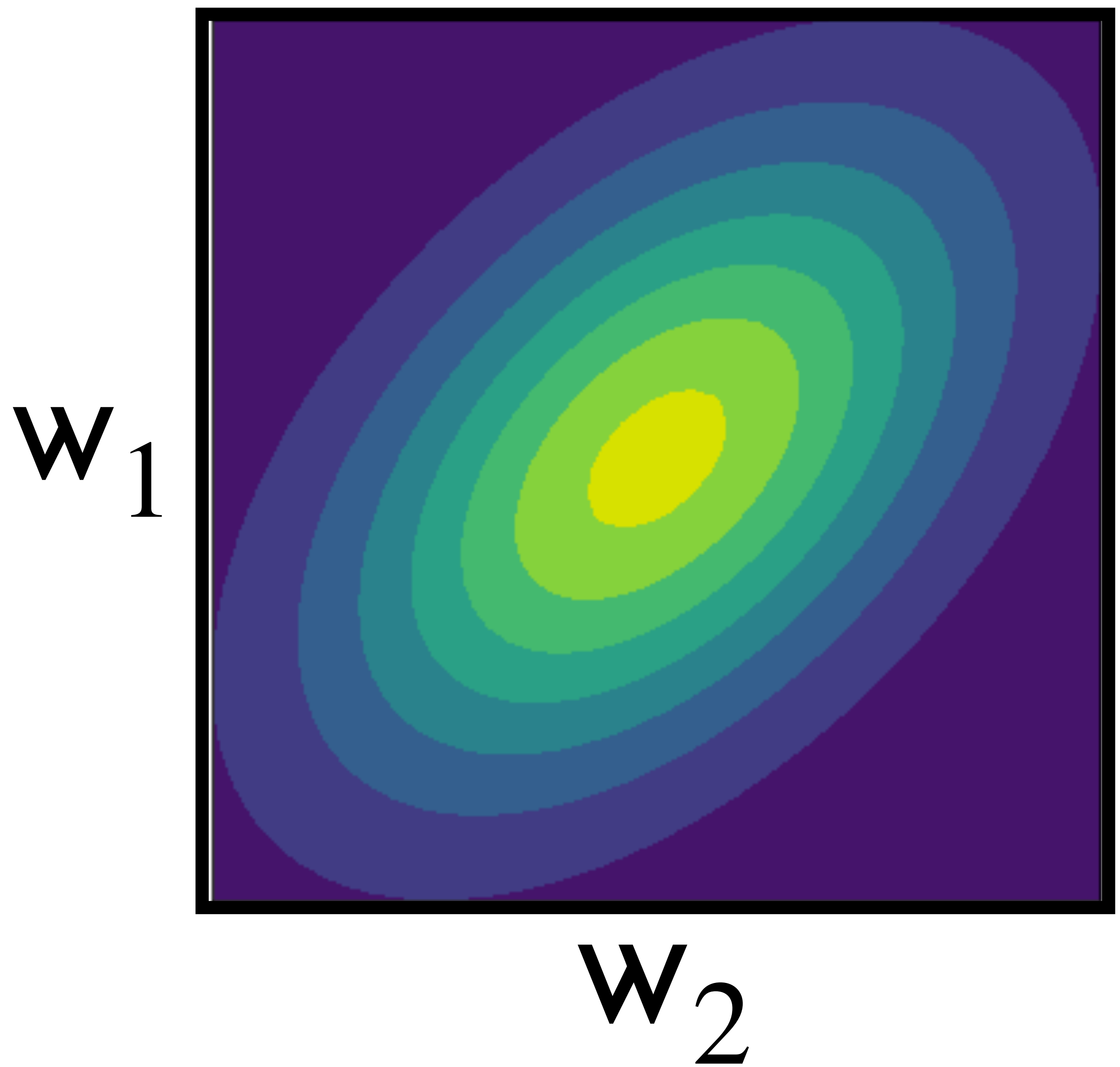


[Matthews et al., 2018]

Multivariate Normal



Multivariate Normal



Multivariate Normal

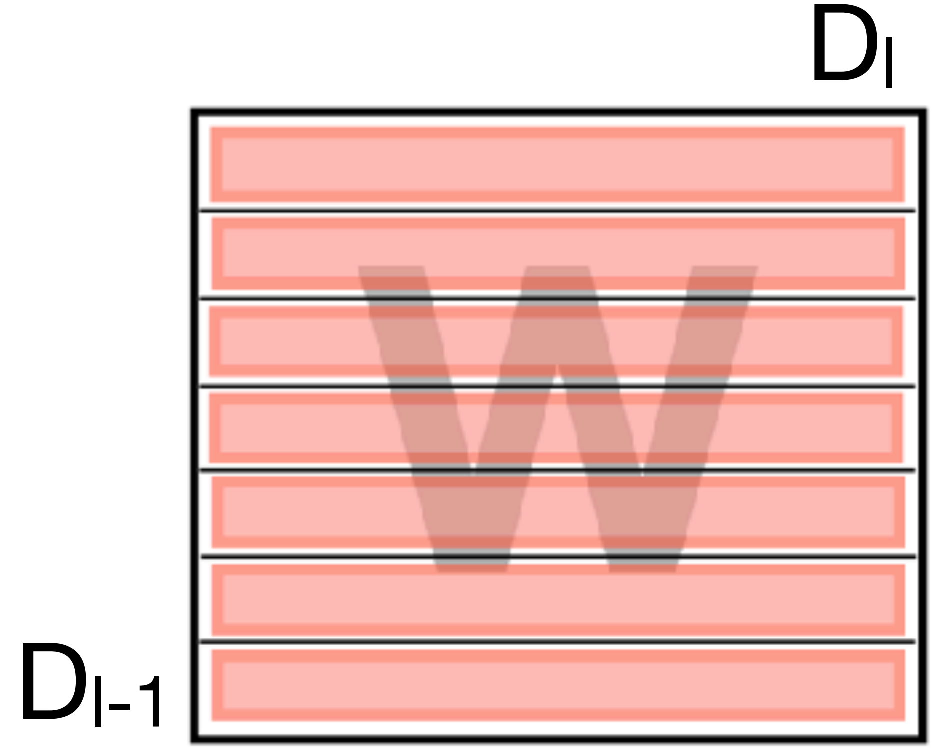
$$\begin{bmatrix} \sigma_1^2 & & & & & \\ & \sigma_2^2 & & & & \\ & & \sigma_3^2 & & & \\ & & & \sigma_4^2 & & \\ & & & & \sigma_5^2 & \\ & & & & & \sigma_6^2 \end{bmatrix}$$

$$\Sigma$$

Multivariate Normal

$$\begin{bmatrix} \sigma_1^2 & & & & & \\ & \sigma_2^2 & & & & \\ & & \sigma_3^2 & & & \\ & & & \sigma_4^2 & & \\ & & & & \sigma_5^2 & \\ & & & & & \sigma_6^2 \end{bmatrix}$$

$$\Sigma$$



$$\mathsf{h}_{l-1}\mathsf{W}_l$$

Hierarchical Priors

$$\tau \sim p(\tau)$$

$$\mathbf{w} \sim p(\mathbf{w} \mid \tau)$$

Hierarchical Priors

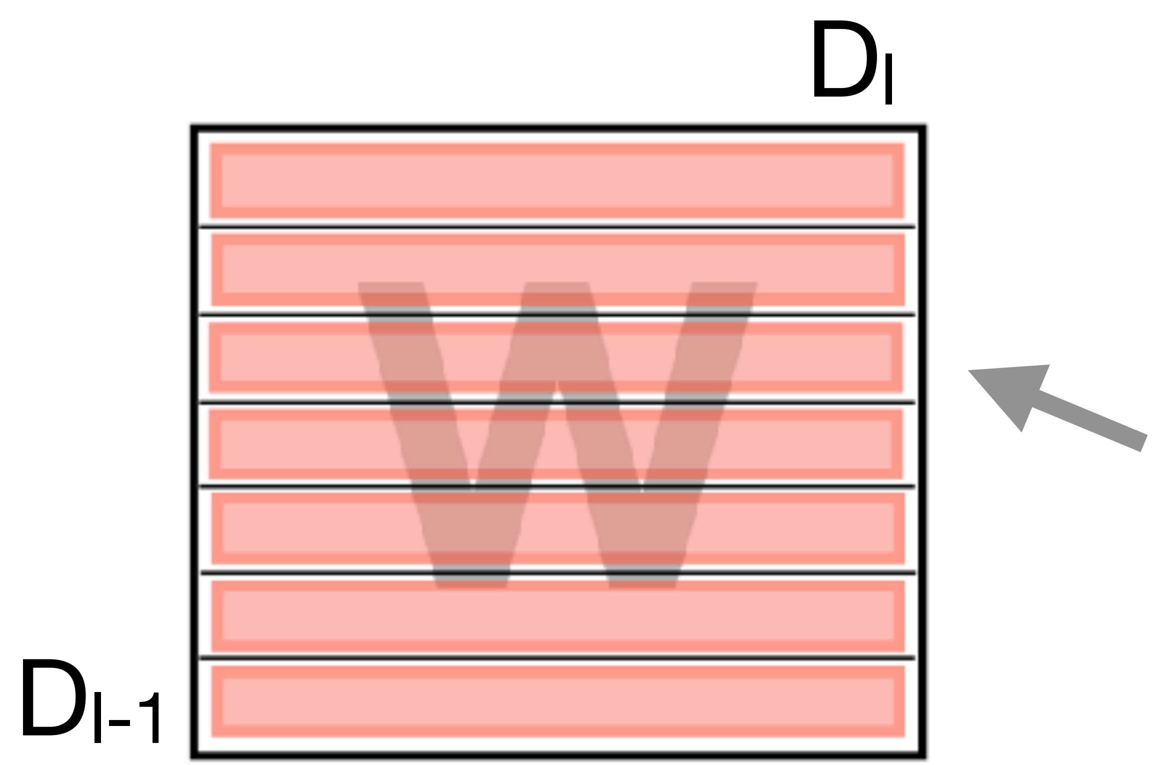
$$\tau \sim p(\tau)$$

$$\mathrm{w} \sim \mathrm{N}(0,\ \tau^2)$$

Hierarchical Priors: Structure

$$\tau_i \sim p(\tau)$$

$$\mathbf{w}_{i,j} \sim \mathrm{N}(0,\ \tau_i^2)$$



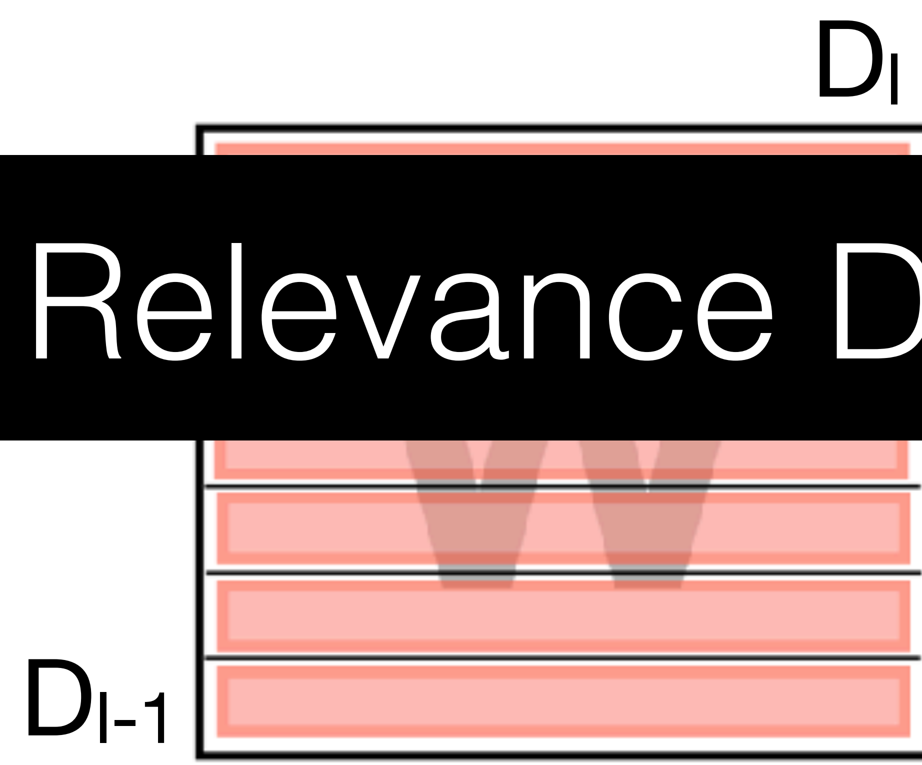
i indexes rows

Hierarchical Priors: Structure

$$\tau_i \sim p(\tau)$$

$$w_{i,j} \sim \mathrm{N}(0,\ \tau_i^2)$$

MacKay, 1994



“Automatic Relevance Determination”

Hierarchical Priors: Heavy-Tails

$$\tau^2 \sim \Gamma^{-1}(\alpha, \beta)$$

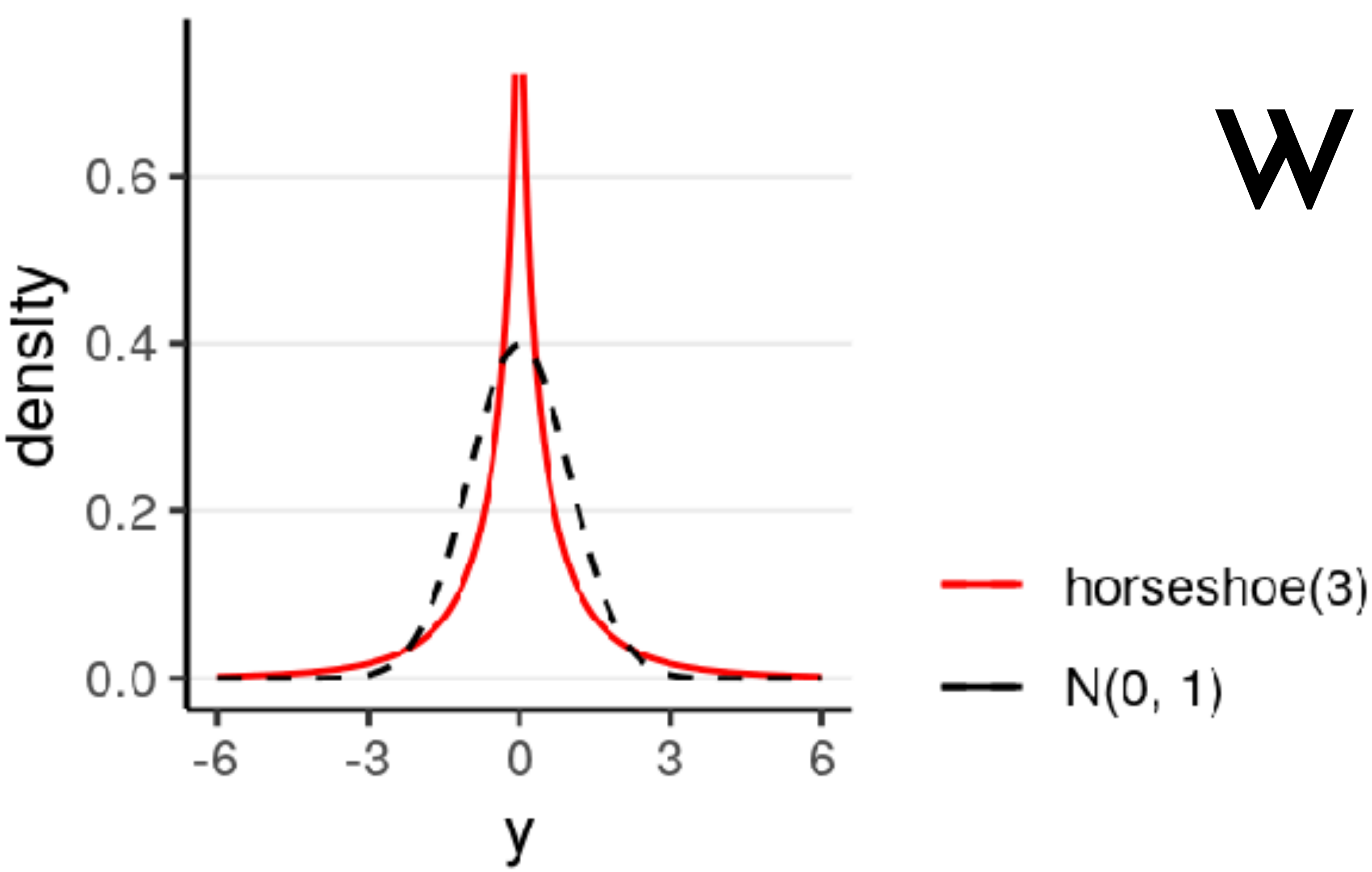
$$\mathsf{w} \sim \mathsf{N}(0, \tau^2)$$

$$\mathsf{t}(\mathsf{w}) = \int_\tau \mathsf{N}(\mathrm{w}; 0, \tau^2)\ \Gamma^{-1}(\tau^2; \alpha, \beta)\ d\tau$$

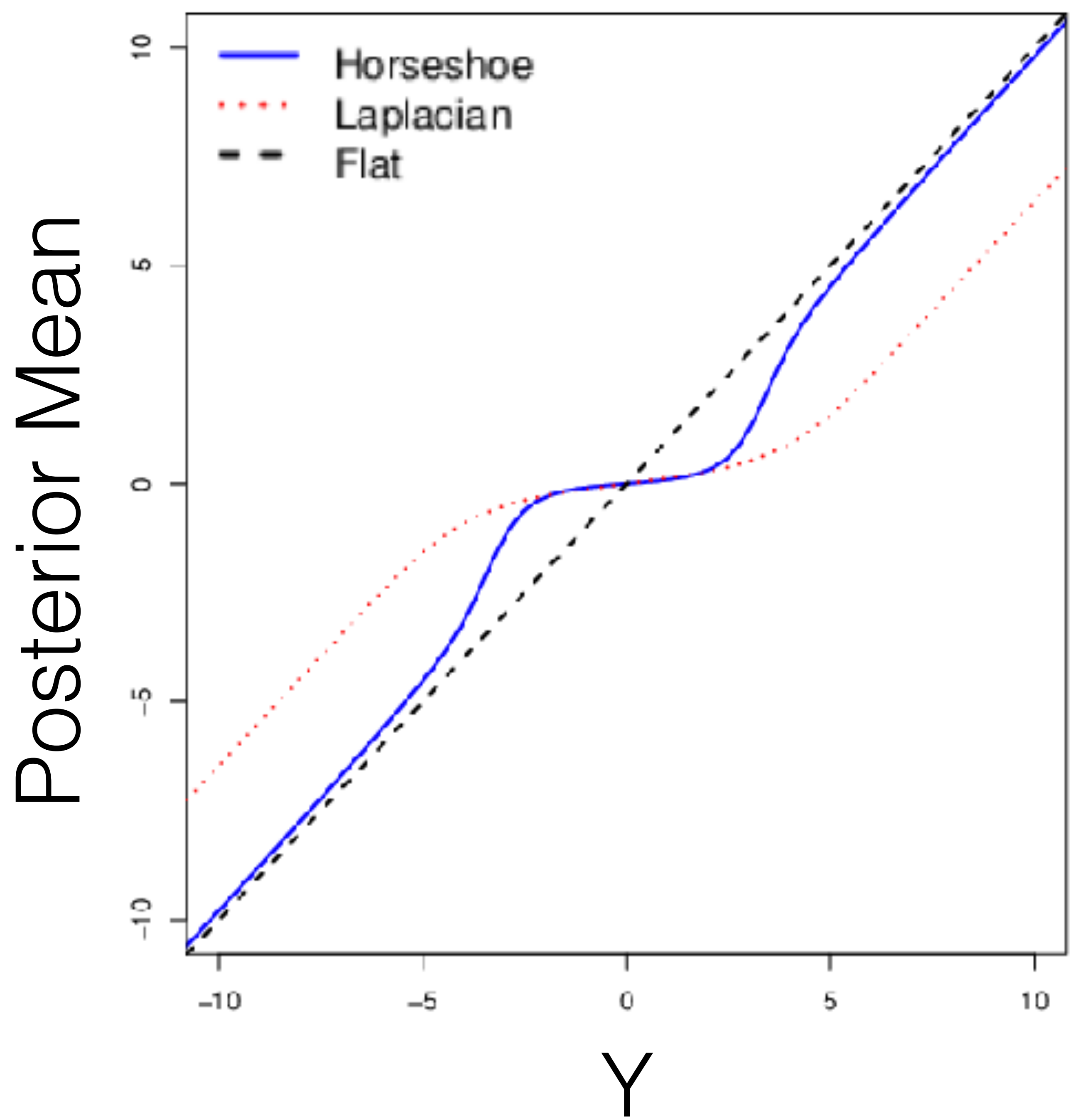
Hierarchical Priors: Heavy-Tails

$$\tau^2 \sim \text{Cauchy}^+(\sigma)$$

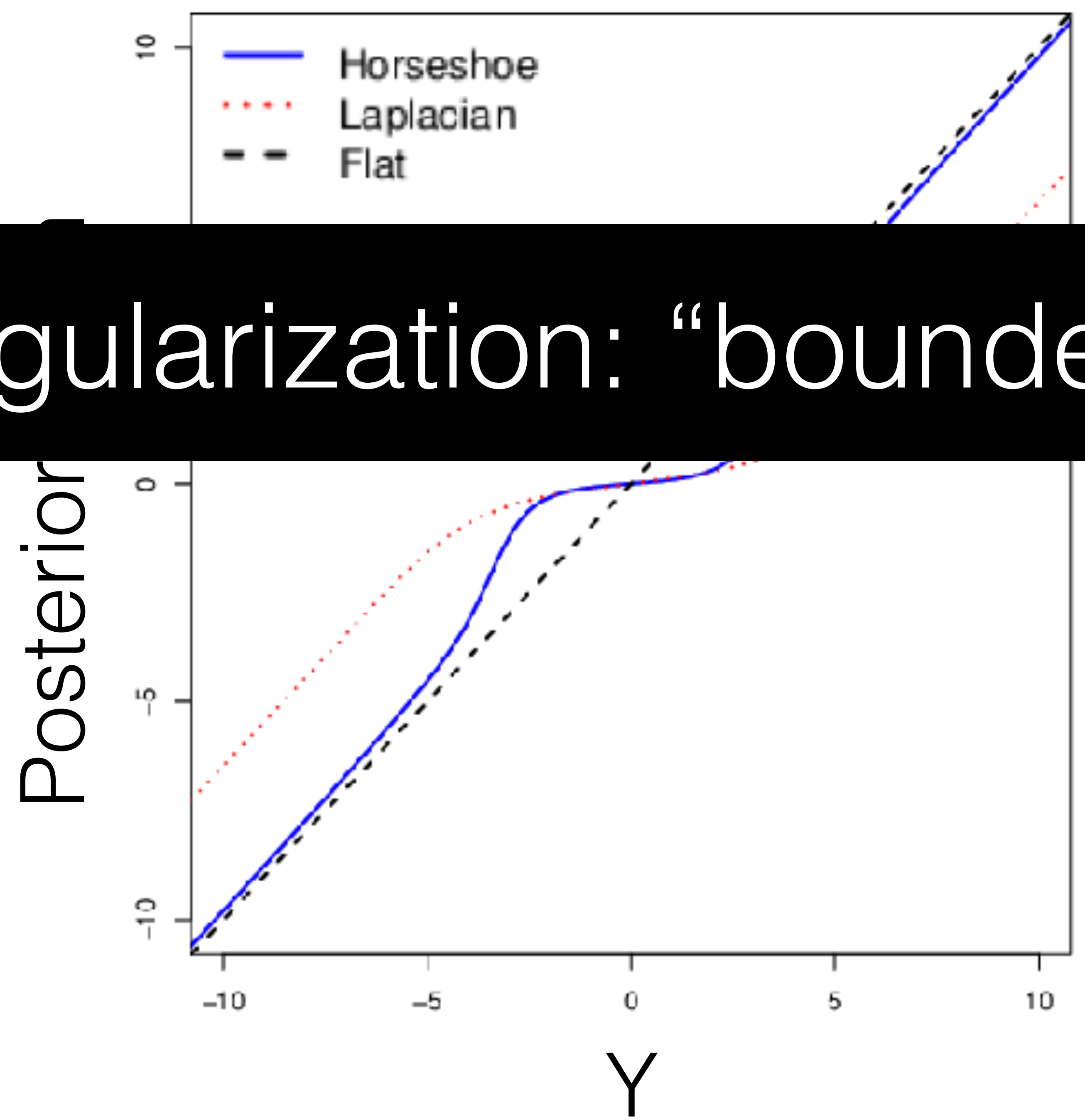
$$\text{w} \sim \text{N}(0, \tau^2)$$



Hierarchical Priors: Heavy-Tails



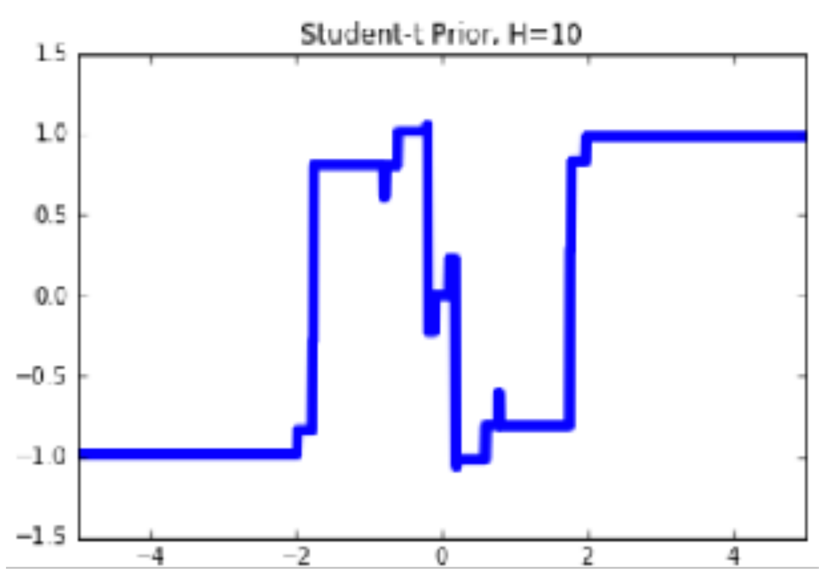
Hierarchical Priors: Heavy-Tails

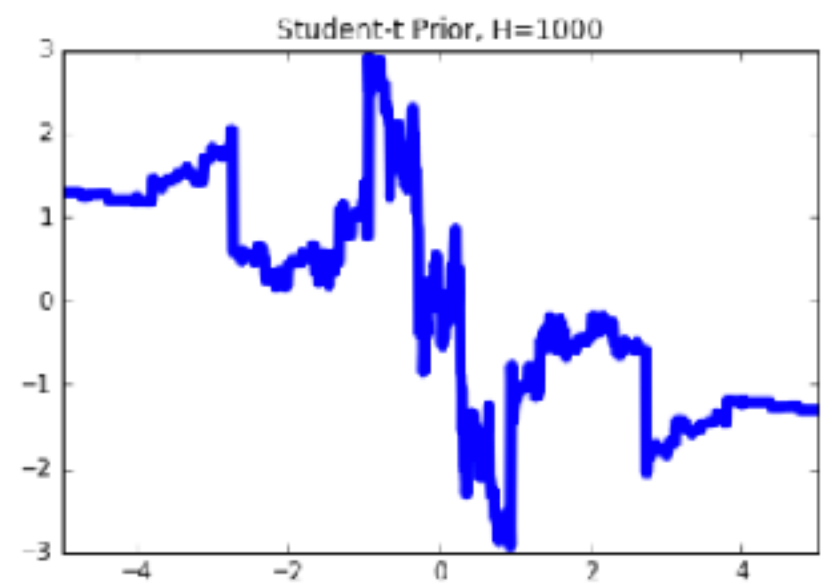


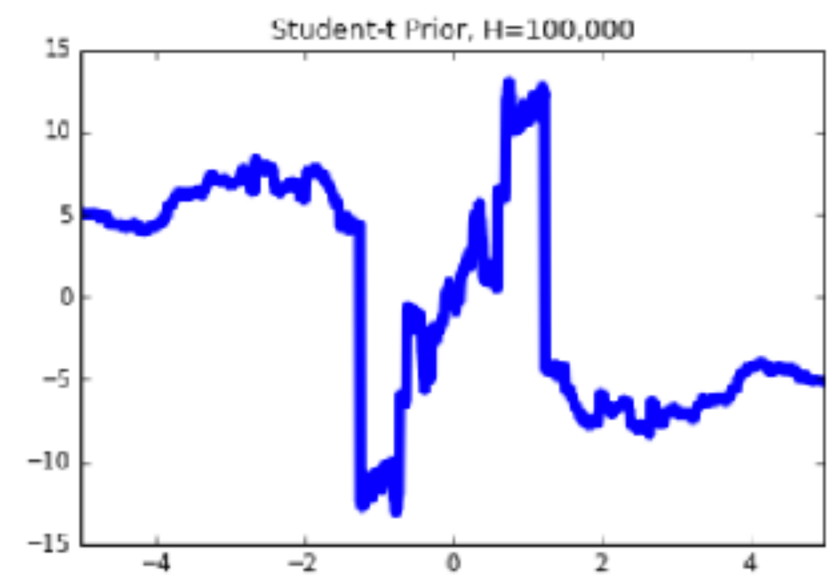
Forget regularization: “bounded Influence”

Hierarchical Priors: Heavy-Tails

Infinitely wide NN no longer converges to a *Gaussian process;* instead a *jump* process.







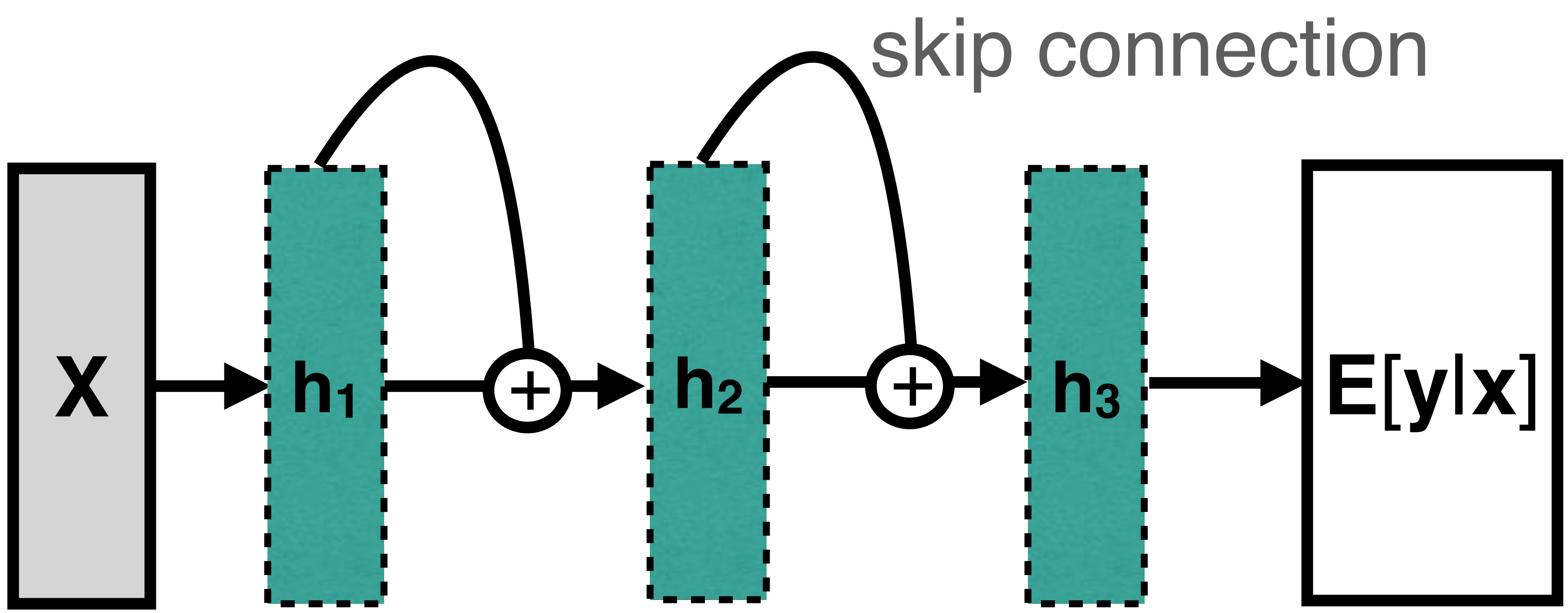
Discrete Priors

$$\mathrm{w} \sim \mathrm{Bernoulli}(\pi)$$

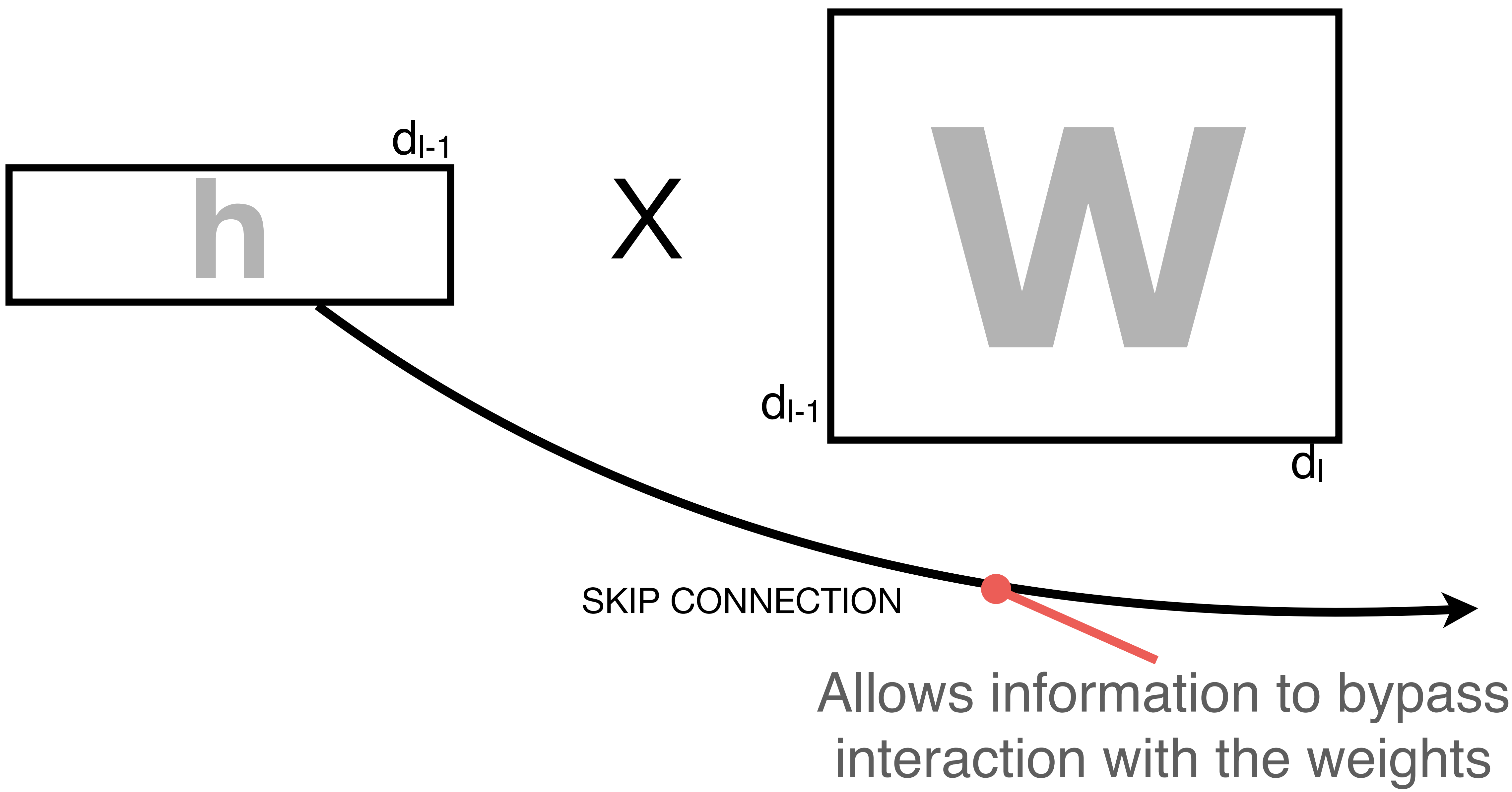
Interesting due to their computational efficiency [Soudry et al., 2014] and biological plausibility [Baldassi et al., 2007].

But no access to gradients.

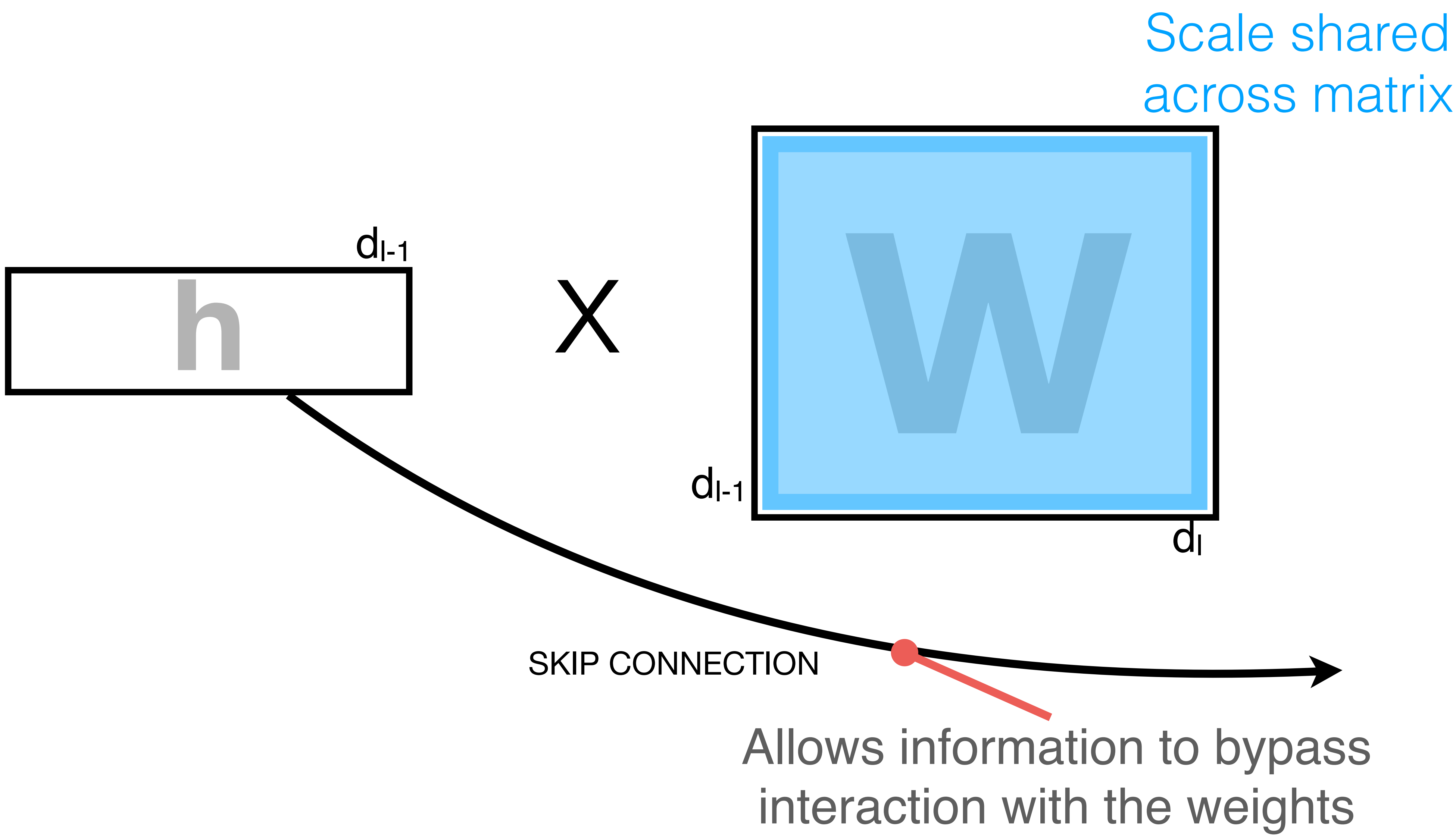
Other Architectures: ResNets



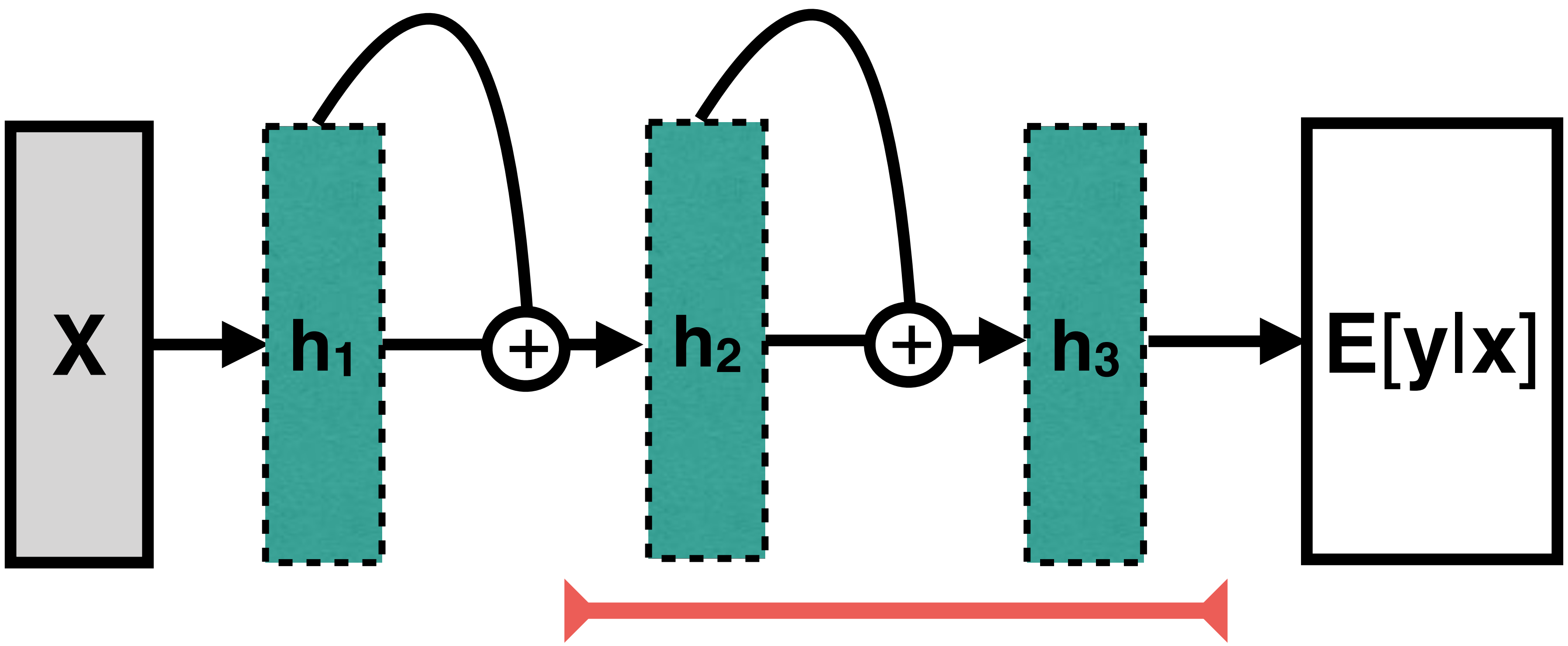
Other Architectures: ResNets



Other Architectures: ResNets

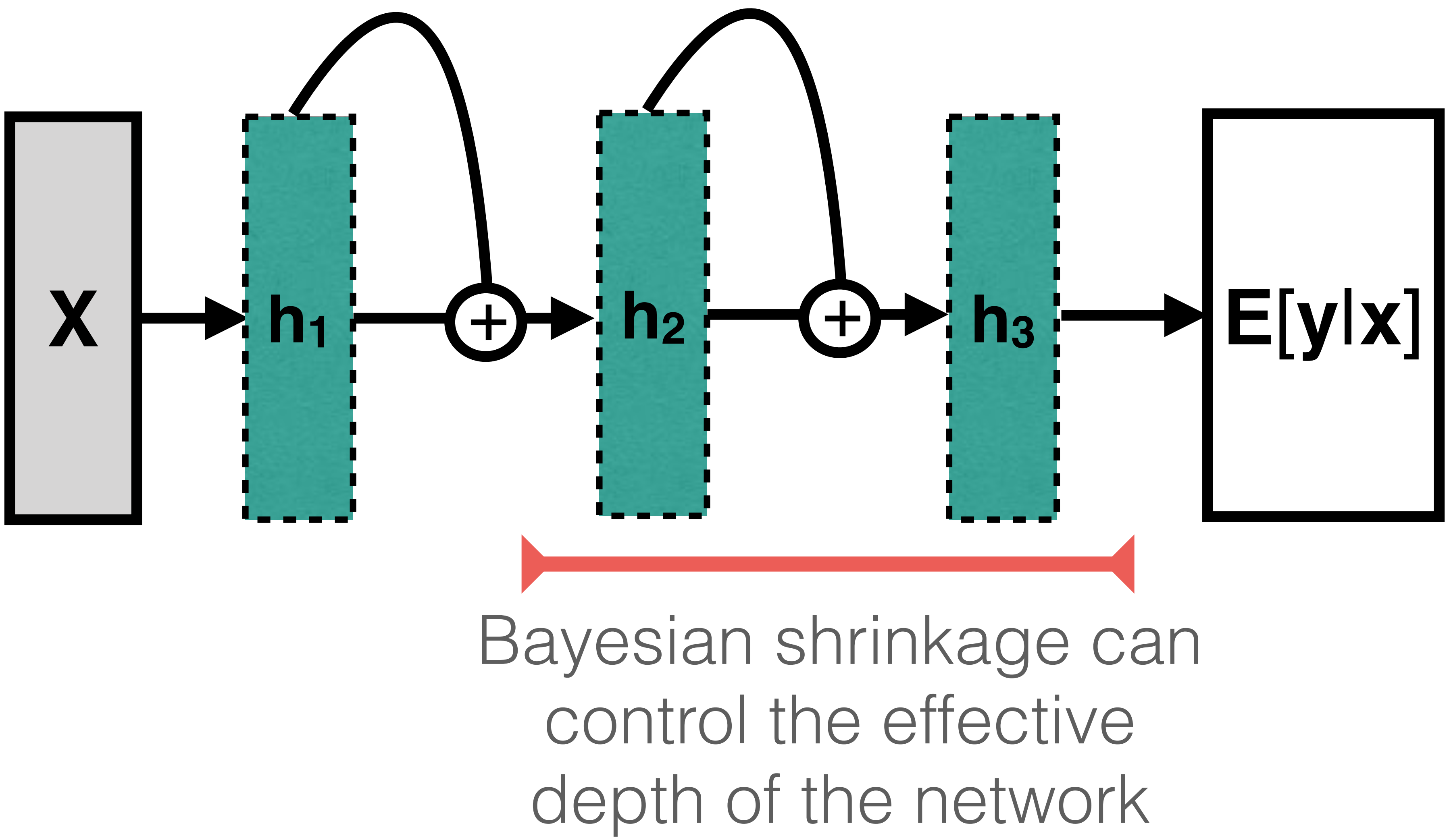


Other Architectures: ResNets



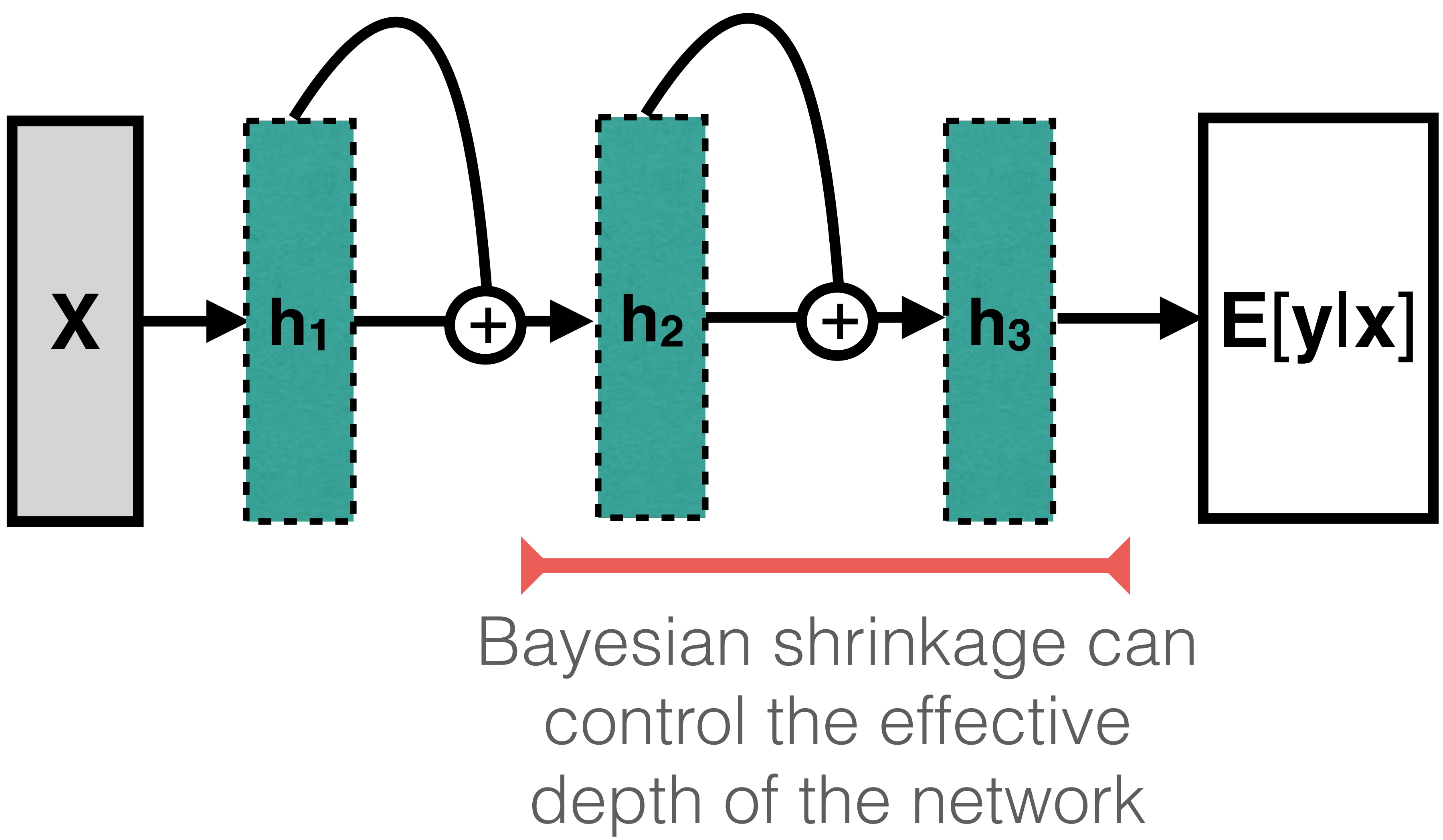
Bayesian shrinkage can
control the effective
depth of the network

Other Architectures: ResNets



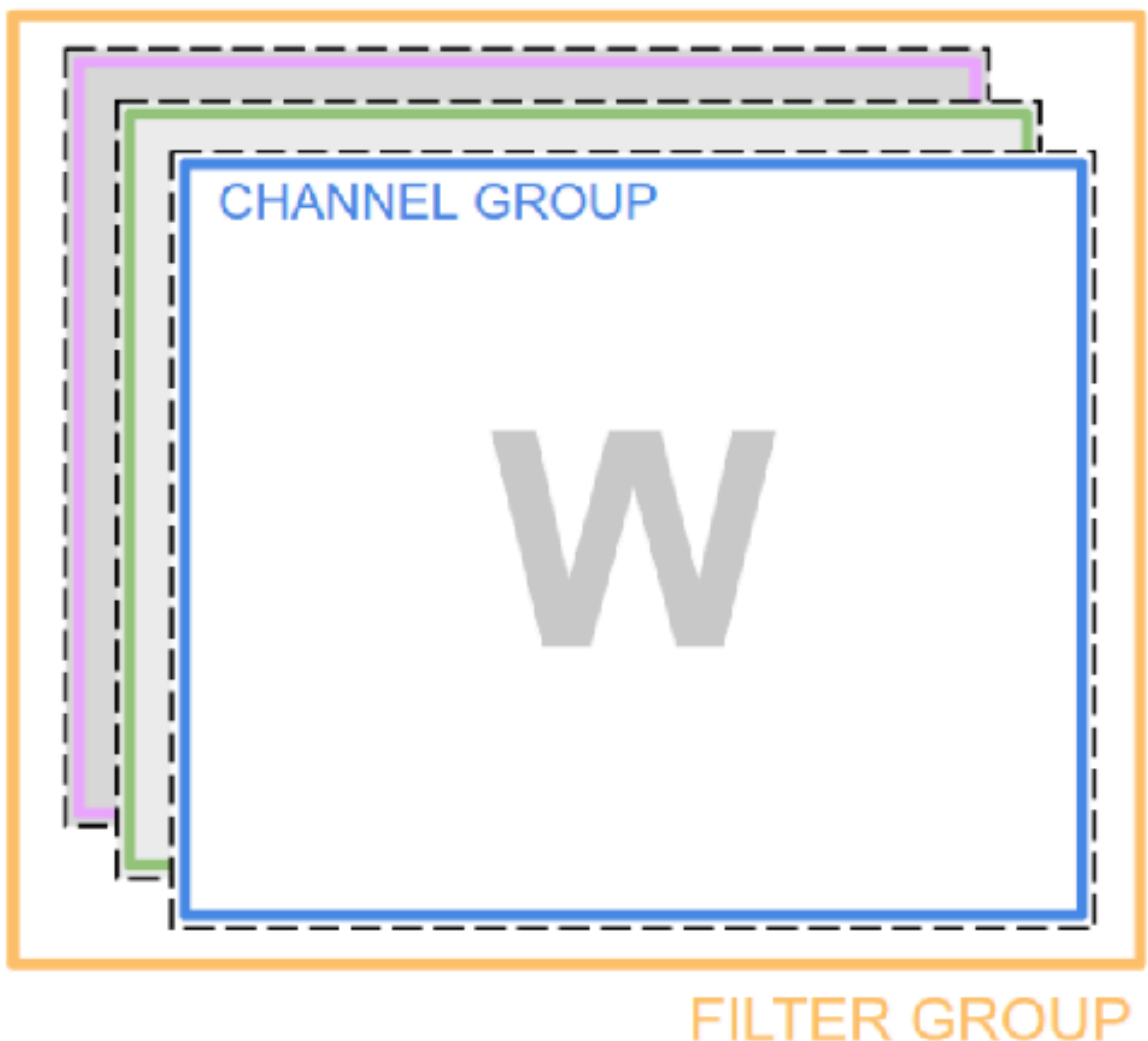
$$\mathbf{w}_{l,i,j} \sim \mathrm{N}(0, \gamma_l^2) \qquad \gamma_l \sim p(\gamma)$$

Other Architectures: ResNets

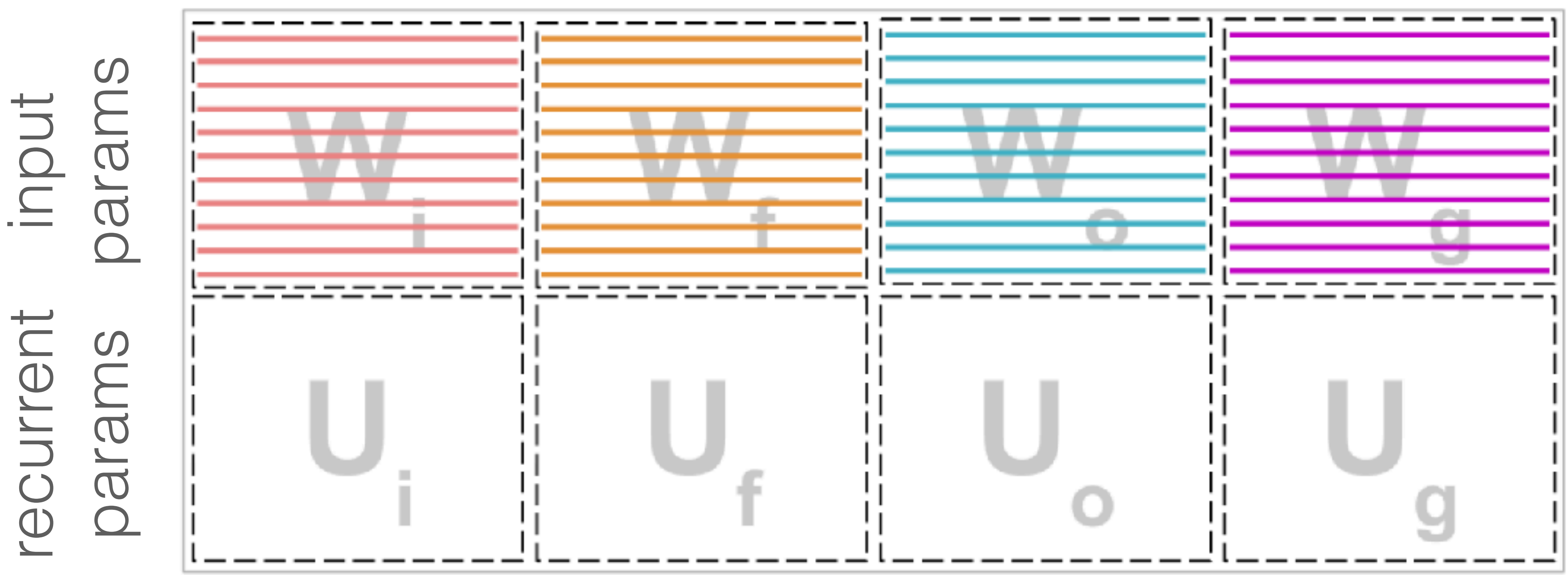


$$\mathrm{w}_{l,i,j} \sim \mathrm{N}(0, \tau_i^2 \gamma_l^2) \quad \gamma_l \sim p(\gamma) \quad \tau_i \sim p(\tau)$$

Other Architectures: ConvNet



Other Architectures: LSTM



Tuning the Prior: Type II MLE

$$p(W; \psi)$$

Tuning the Prior: Type II MLE

$$\mathrm{p}(\mathrm{W};\psi)$$

$$p\left(\mathrm{y} \mid \mathrm{x};\psi\right)$$

Tuning the Prior: Type II MLE

$$\mathrm{p}(\mathrm{W};\psi)$$

$$p\left(\mathrm{y}\,|\,\mathrm{x};\underline{\psi}\right)=\int_{\mathrm{W}}\mathrm{p}(\mathrm{y}\,|\,\mathrm{x},\mathrm{W})\;\mathrm{p}(\mathrm{W};\underline{\psi})\;\mathrm{dW}$$

Tuning the Prior: Type II MLE

$$\mathrm{p}(\mathrm{W}; \psi)$$

$$p\left(\mathrm{y} \mid \mathrm{x}; \underline{\psi}\right) = \int_{\mathrm{W}} \mathrm{p}(\mathrm{y} \mid \mathrm{x}, \mathrm{W})\ \mathrm{p}(\mathrm{W}; \underline{\psi})\ \mathrm{dW}$$

Scalable Marginal Likelihood Estimation for Model Selection in Deep Learning

Alexander Immer [1 2] **Matthias Bauer** [† 3 4] **Vincent Fortuin** [1] **Gunnar Rätsch** [1 2] **Mohammad Emtiyaz Khan** [5]

Summary

- **Normal priors**: easy to implement, correspond to Gaussian process in the infinite limit.
- **Hierarchical priors**: good for inducing structure and heavy-tails.
- **Discrete priors**: efficient but hard to implement.

[Lee, 2004]

Priors for Neural Networks

Herbert K. H. Lee

Department of Applied Mathematics and Statistics

University of California, Santa Cruz

herbie@ams.ucsc.edu

Abstract

Neural networks are commonly used for classification and regression. The Bayesian approach may be employed, but choosing a prior for the parameters presents challenges. This paper reviews several priors in the literature and introduces Jeffreys priors for neural network models. The effect on the posterior is demonstrated through an example.

Key Words: nonparametric classification; nonparametric regression; Bayesian statistics; prior sensitivity

1 Introduction

Neural networks are a popular tool for nonparametric classification and regression. They offer a computationally tractable model that is fully flexible, in the sense of being able to approximate a wide range of functions (such as all continuous functions). Many references on neural networks are available (Bishop, 1995; Fine, 1999; Ripley, 1996). The Bayesian approach is appealing as it allows full accounting for uncertainty in the model and the choice of model (Lee, 2001; Neal, 1996). An important decision in any Bayesian analysis is the choice of prior. The idea is that your prior should reflect your current beliefs (either from previous data

[Lee, 2004]

Chapter 3

Survey of Neural Network Priors

We demand rigidly defined areas of doubt and uncertainty!

Douglas Adams
The Hitchhiker's Guide to the Galaxy

Having covered the basics of Bayesian NNs and strategies for inferring their posterior, I now turn to the focal point of the dissertation: prior distributions for both conditional NNs and density networks. Surprisingly, a broad review of Bayesian NN priors has been performed by only Robinson [2001], which is now considerably out of date. Thus, in this chapter I survey the existing work on NN priors, some of which was performed in the early days of Bayesian NNs and therefore also discussed by Robinson [2001]. However, most of the work is recent, some having been conducted concurrently with my own work to be presented in the coming chapters.

NNs have been applied to a myriad of different problems over the past thirty years, and this of course makes it impossible to discuss every prior ever used for a NN. Instead, I attempt to summarize broad themes from the literature that pertain to core NN methodology. For instance,

[Nalisnick, 2018]

eural Networks

K. H. Lee

Mathematics and Statistics

lifornia, Santa Cruz

rs.ucsc.edu

bstract

ication and regression. The Bayesian approach may be

presents challenges. This paper reviews several priors

neural network models. The effect on the posterior is

rametric regression; Bayesian statistics; prior sensitiv-

Neural networks are a popular tool for nonparametric classification and regression. They offer a computationally tractable model that is fully flexible, in the sense of being able to approximate a wide range of functions (such as all continuous functions). Many references on neural networks are available (Bishop, 1995; Fine, 1999; Ripley, 1996). The Bayesian approach is appealing as it allows full accounting for uncertainty in the model and the choice of model (Lee, 2001; Neal, 1996). An important decision in any Bayesian analysis is the choice of prior. The idea is that your prior should reflect your current beliefs (either from previous data

[Lee, 2004]

Chapter 3

Survey of Neural Network Priors

We demand rigidly defined areas of doubt and uncertainty!

Douglas A

The Hitchhiker's Guide to the G

Having covered the basics of Bayesian NNs and strategies for inferring their posterior, I turn to the focal point of the dissertation: prior distributions for both conditional NNs and sity networks. Surprisingly, a broad review of Bayesian NN priors has been performed by Robinson [2001], which is now considerably out of date. Thus, in this chapter I survey th isting work on NN priors, some of which was performed in the early days of Bayesian NNs therefore also discussed by Robinson [2001]. However, most of the work is recent, some ha been conducted concurrently with my own work to be presented in the coming chapters.

NNs have been applied to a myriad of different problems over the past thirty years, and of course makes it impossible to discuss every prior ever used for a NN. Instead, I attem summarize broad themes from the literature that pertain to core NN methodology. For insta

[Nalisnick, 2018]

eural Networks

K. H. Lee

Neural networks are a popular tool for n tationally tractable model that is fully fl functions (such as all continuous functions Fine, 1999; Ripley, 1996). The Bayesian a the model and the choice of model (Lee, 20 the choice of prior. The idea is that your

Priors in Bayesian Deep Learning: A Review

Vincent Fortuin
Department of Computer Science
ETH Zürich
Zürich, Switzerland
fortuin@inf.ethz.ch

Abstract

While the choice of prior is one of the most critical parts of the Bayesian inference workflow, recent Bayesian deep learning models have often fallen back on vague priors, such as standard Gaussians. In this review, we highlight the importance of prior choices for Bayesian deep learning and present an overview of different priors that have been proposed for (deep) Gaussian processes, variational autoencoders, and Bayesian neural networks. We also outline different methods of learning priors for these models from data. We hope to motivate practitioners in Bayesian deep learning to think more carefully about the prior specification for their models and to provide them with some inspiration in this regard.

1 Introduction

Bayesian models have gained a stable popularity in data analysis [1] and machine learning [2]. Especially in recent years, the interest in combining these models with deep learning has surged[1]. The main idea of Bayesian modeling is to infer a *posterior* distribution over the parameters $\boldsymbol{\theta}$ of the model given some observed data $\mathcal{D}$ using Bayes' theorem [3, 4] as

$$p(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})}{p(\mathcal{D})} = \frac{p(\mathcal{D} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})}{\int p(\mathcal{D} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta}} \tag{1}$$

[Fortuin, 2021]

Bayesian Deep Learning: Posterior Inference

Eric Nalisnick

Amsterdam Machine Learning Lab

Deep Learning II,
University of Amsterdam

$$p\left(\mathsf{W}_1, \ldots, \mathsf{W}_L \,|\, \mathsf{y}, \mathsf{x}\right)$$

Non-identifiability

$$h_1 = f(w_{1,1}x)$$

$$h_2 = f(w_{1,2}x)$$

$$\hat{y} = w_{2,1}h_1 + w_{2,2}h_2$$

Non-identifiability

$$h_1 = f(w_{1,1}x)$$

$$h_2 = f(w_{1,2}x)$$

$$\hat{y} = w_{2,1}h_1 + w_{2,2}h_2$$

Non-identifiability

$$h_1 = f(w_{1,1}x)$$

$$h_2 = f(w_{1,2}x)$$

$$\hat{y} = w_{2,1}h_1 + w_{2,2}h_2$$

Non-identifiability

- Permutation invariance.

- Scale invariance for ReLUs:

$$\mathsf{ReLU}(\mathsf{x}) = (1/\alpha) \cdot \mathsf{ReLU}(\alpha \cdot \mathsf{x}), \quad \forall \alpha > 0$$

Conjugacy?

Not in general…

Conjugacy?

Not in general…

But sometimes for the last layer:

$$\log \mathsf{N}\left(\mathsf{y}\,|\,\mathsf{x}, \mathsf{W}_1, \ldots, \mathsf{W}_L\right) =$$

$$\frac{-1}{2\sigma_0^2}\left(\mathsf{y} - \mathsf{h}_{L-1}\mathsf{W}_L\right)^2 + \ldots$$

Conjugacy?

Not in general…

But sometimes for the last layer.

“neural linear model”

$$\frac{-1}{2\sigma_0^2}\left(\mathsf{y} - \mathsf{h}_{L-1}\mathsf{W}_L\right)^2 + \ldots$$

MAP Estimation

$$p\left(\mathsf{W}_1, \ldots, \mathsf{W}_L \,|\, \mathsf{y}, \mathsf{x}\right) \propto$$

$$\log p\left(\mathsf{y} \,|\, \mathsf{x}, \mathsf{W}_1, \ldots, \mathsf{W}_L\right) + \sum_{l=1}^{L} \log p(\mathsf{W}_l)$$

MAP Estimation

$$p\left(\mathsf{W}_1, \ldots, \mathsf{W}_L \mid \mathsf{y}, \mathsf{x}\right) \propto$$

$$\log p\left(\mathsf{y} \mid \mathsf{x}, \mathsf{W}_1, \ldots, \mathsf{W}_L\right) + \sum_{l=1}^{L} \log p(\mathsf{W}_l)$$

For normal priors…

$$-\sum_{l=1}^{L} \frac{1}{2\sigma_l^2} ||\mathsf{W}_l||_2^2 + \mathsf{const}.$$

MAP Estimation

$$p\left(\mathsf{W}_1, \ldots, \mathsf{W}_L \,|\, \mathsf{y}, \mathsf{x}\right) \propto$$

Equivalent to weight decay

For normal priors…

$$-\sum_{l=1}^{L} \frac{1}{2\sigma_l^2} ||\mathsf{W}_l||_2^2 + \text{const}.$$

MAP Estimation

Caution: MAP estimates have very different characteristics than the true posterior (e.g. sparsity)

On Bayesian classification with Laplace priors

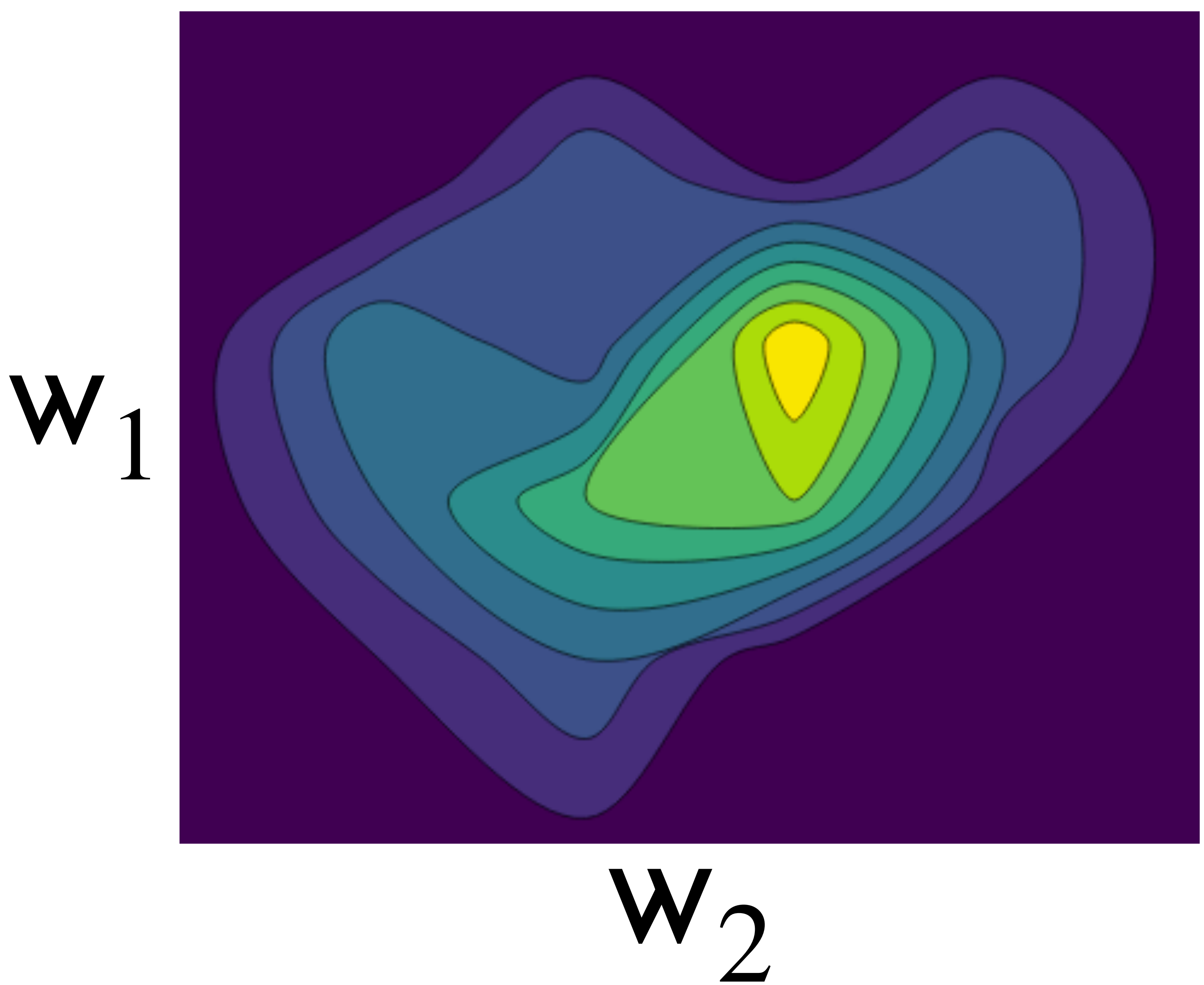
Ata Kabán

School of Computer Science, The University of Birmingham, Birmingham B15 2TT, UK

Received 27 February 2006; received in revised form 20 November 2006
Available online 28 February 2007

Communicated by M. Singh

Markov Chain Monte Carlo (MCMC)



Markov Chain Monte Carlo (MCMC)

$$\mathrm{p}\left(\mathrm{W}_1, \ldots, \mathrm{W}_L \,|\, \mathrm{y}, \mathrm{x}\right) \approx \frac{1}{S} \sum_{s=1}^{S} \delta \left[\hat{\mathrm{W}}_{1,s}, \ldots, \hat{\mathrm{W}}_{L,s} \right]$$

Initialize $\mathbf{w}^0$

For t=1 to T:

:

Initialize $\mathbf{w}^0$

For t=1 to T:

Sample $u \sim \text{Uniform}(0,1)$

:

Initialize $\mathbf{w}^0$

For t=1 to T:

Sample $\quad \mathrm{u} \sim \mathrm{Uniform}(0,1)$

Sample $\quad \mathrm{w}^* \sim \mathrm{q}(\mathrm{w}^* \mid \mathrm{w}^{t-1})$

:

Initialize $\mathbf{w}^0$

For t=1 to T:

Sample $\quad \mathrm{u} \sim \mathrm{Uniform}(0,1)$

Sample $\quad \mathrm{w}^* \sim \mathrm{q}(\mathrm{w}^* \mid \mathrm{w}^{t-1})$

If $\quad \mathrm{u} < \min \left\{ 1, \dfrac{\mathrm{p}(\mathrm{y}, \mathrm{w}^* \mid \mathrm{x}) \; \mathrm{q}(\mathrm{w}^{t-1} \mid \mathrm{w}^*)}{\mathrm{p}(\mathrm{y}, \mathrm{w}^{t-1} \mid \mathrm{x}) \; \mathrm{q}(\mathrm{w}^* \mid \mathrm{w}^{t-1})} \right\}$:

$$\mathrm{w}^t = \mathrm{w}^*$$

Initialize $\mathbf{w}^0$

For t=1 to T:

Sample $\quad u \sim \text{Uniform}(0,1)$

Sample $\quad w^* \sim q(w^* \,|\, w^{t-1})$

If $\quad u < \min\left\{1, \dfrac{p(y, w^* \,|\, x)\; q(w^{t-1} \,|\, w^*)}{p(y, w^{t-1} \,|\, x)\; q(w^* \,|\, w^{t-1})}\right\}$:

$$w^t = w^*$$

Else:

$$w^t = w^{t-1}$$

Hamiltonian Monte Carlo (HMC)

Generate proposal by iterating:

(assuming the identity matrix for the mass)

$$\mathsf{v}^{m+1} = \mathsf{v}^m + \alpha \nabla_{\mathsf{w}} \log \mathsf{p}(\mathsf{w}^m \,|\, \mathsf{y}, \mathsf{x})$$

$$\mathsf{w}^{m+1} = \mathsf{w}^m + \alpha' \cdot \mathsf{v}^m$$

where α and α' are step sizes and $\mathsf{v}^0 \sim \mathsf{N}(0,1)$

Hamiltonian Monte Carlo (HMC)

Generate proposal by iterating:

(assuming the identity matrix for the mass)

$$\mathsf{v}^{m+1} = \mathsf{v}^{m} + \alpha \nabla_{\mathsf{w}} \log \mathsf{p}(\mathsf{w}^{m} \,|\, \mathsf{y}, \mathsf{x})$$

$$\mathsf{w}^{m+1} = \mathsf{w}^{m} + \alpha' \cdot \mathsf{v}^{m}$$

where α and α' are step sizes and $\mathsf{v}^0 \sim \mathsf{N}(0,1)$

Propose: $\mathsf{w}^* = \mathsf{w}^M$

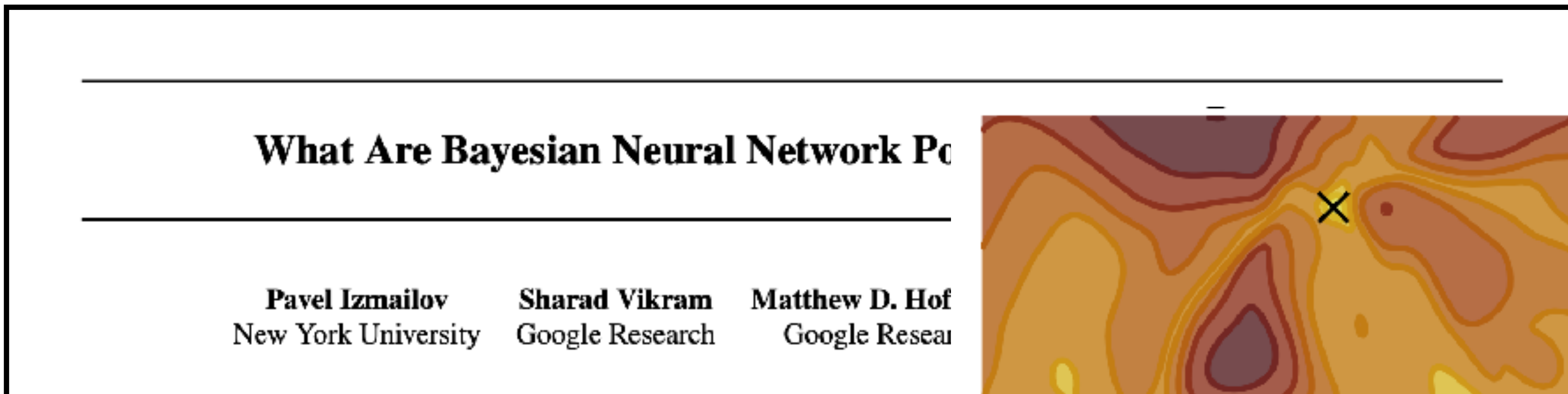
Hamiltonian Monte Carlo (HMC)

What Are Bayesian Neural Network Posteriors Really Like?

Pavel Izmailov New York University
Sharad Vikram Google Research
Matthew D. Hoffman Google Research
Andrew Gordon Wilson New York University

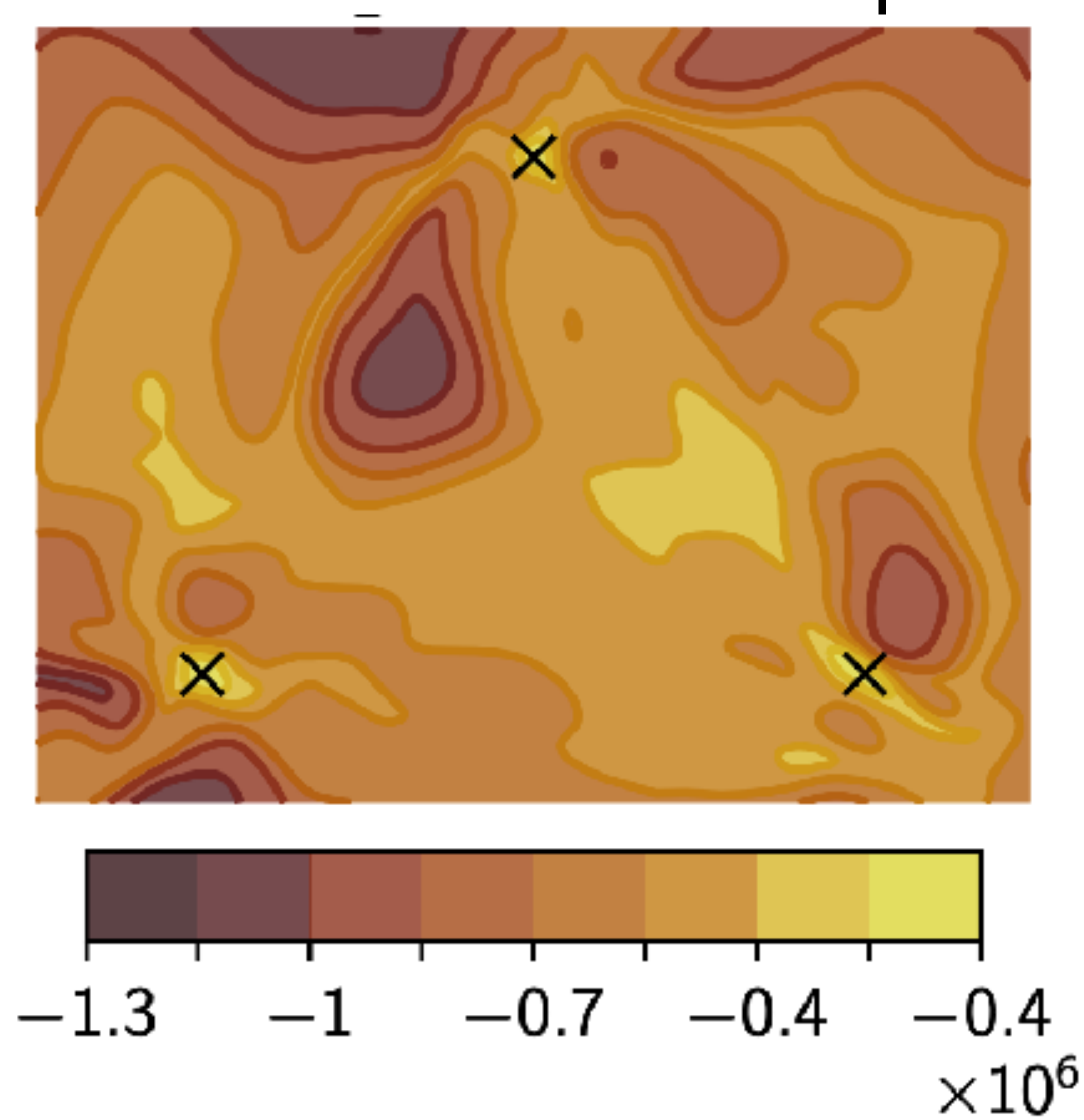
Computation done
on 512 TPUs

Hamiltonian Monte Carlo (HMC)



What Are Bayesian Neural Network Po

Pavel Izmailov New York University
Sharad Vikram Google Research
Matthew D. Hof Google Resea



Computation done on 512 TPUs

HMC to Langevin Dynamics

One step iteration of HMC:

$$\begin{aligned} \mathsf{w}^1 &= \mathsf{w}^0 + \alpha' \cdot \mathsf{v}^1 \\ &= \mathsf{w}^0 + \alpha' \cdot \left(\alpha \nabla_{\mathsf{w}} \log \mathsf{p}(\mathsf{w}^0 \,|\, \mathsf{y}, \mathsf{x}) + \mathsf{v}^0\right) \end{aligned}$$

HMC to Langevin Dynamics

One step iteration of HMC:

$$
\begin{aligned}
\mathsf{w}^1 &= \mathsf{w}^0 + \alpha' \cdot \mathsf{v}^1 \\
&= \mathsf{w}^0 + \alpha' \cdot \left(\alpha \nabla_{\mathsf{w}} \log \mathsf{p}(\mathsf{w}^0 \,|\, \mathsf{y}, \mathsf{x}) + \mathsf{v}^0\right)
\end{aligned}
$$

Langevin Dynamics:

$$
\mathsf{w}^{m+1} = \mathsf{w}^m + \alpha'' \cdot \nabla_{\mathsf{w}} \log \mathsf{p}(\mathsf{w}^m \,|\, \mathsf{y}, \mathsf{x}) + \hat{\mathsf{v}}
$$

$$
\hat{\mathsf{v}} \sim \mathsf{N}(0, \epsilon)
$$

Langevin Dynamics

$$\mathsf{w}^{m+1} = \mathsf{w}^m + \alpha'' \cdot \nabla_{\mathsf{w}} \log \mathsf{p}(\mathsf{w}^m \,|\, \mathsf{y}, \mathsf{x}) + \hat{\mathsf{v}}$$

$$\hat{\mathsf{v}} \sim \mathsf{N}(0,\ \epsilon)$$

- ⊗ “Adjusted": Run accept-reject step
- ⊗ “Unadjusted": Always accept proposal
- ⊗ Can also use stochastic gradients

MCMC for ResNet-20 on CIFAR-10

Metric	HMC (reference)	SGMCMC SGLD	SGHMC	SGHMC CLR	SGHMC CLR-Prec
Accuracy	89.64 ±0.25	89.32 ±0.23	89.38 ±0.32	**89.63 ±0.37**	87.46 ±0.21
Agreement	94.01 ±0.25	91.54 ±0.15	91.98 ±0.35	**92.67 ±0.52**	90.96 ±0.24
Total Var	0.074 ±0.003	0.110 ±0.001	0.109 ±0.001	**0.099 ±0.006**	0.111 ±0.002

Variational Inference

$$p\left(W_1, \ldots, W_L \mid y, x\right) \approx q\left(W_1, \ldots, W_L; \phi\right)$$

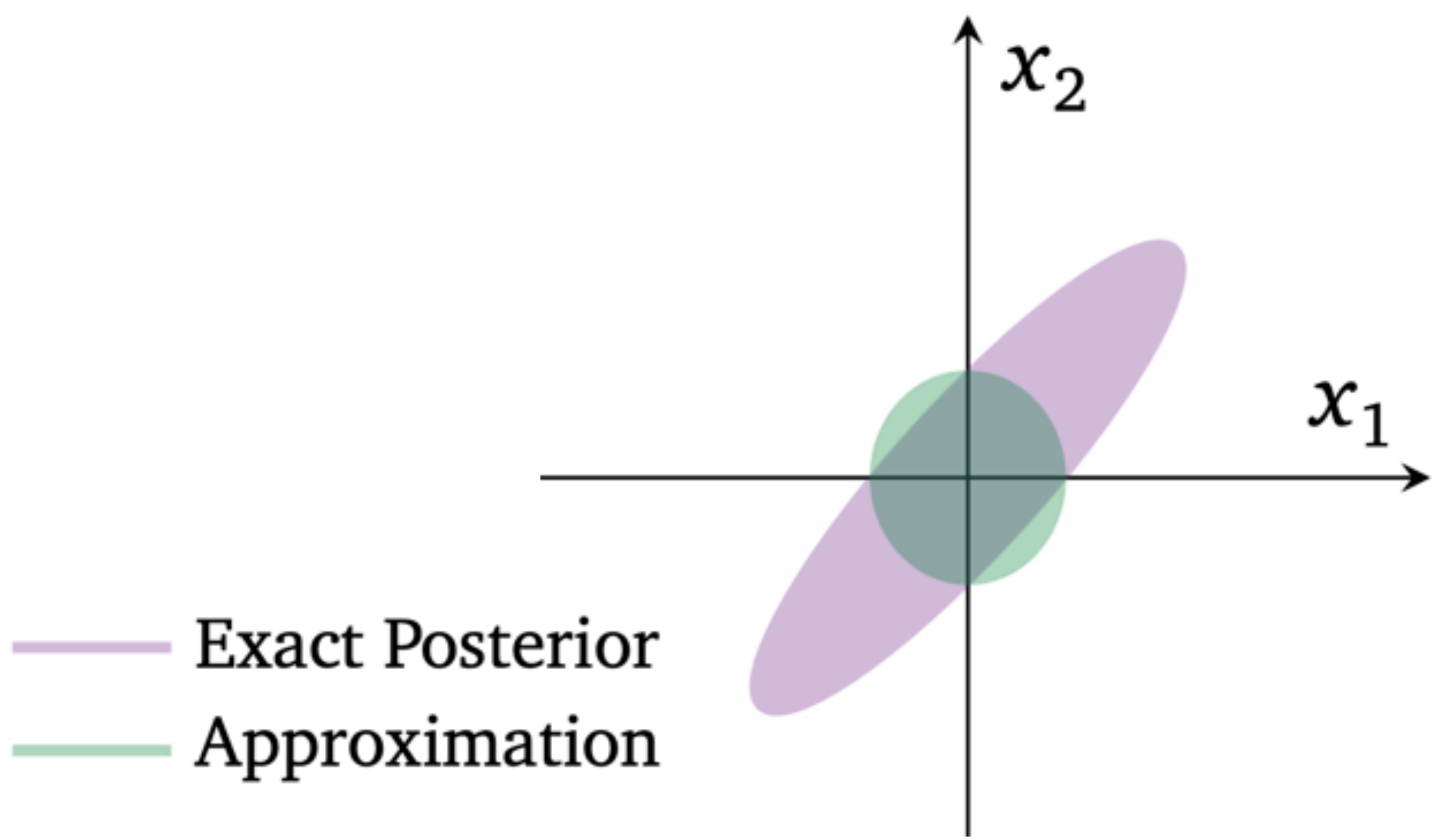


Image from Blei et al., “Variational Inference: A Review for Statisticians,” JASA 2017

We usually need to assume some degree of factorization.

We usually need to assume some degree of factorization.

Over layers:

$$\mathsf{q}\left(\mathsf{W}_1, \ldots, \mathsf{W}_L; \phi\right) = \prod_{l=1}^{L} \mathsf{q}\left(\mathsf{W}_l; \phi_l\right)$$

We usually need to assume some degree of factorization.

Over layers:

$$\mathsf{q}\left(\mathsf{W}_1, \ldots, \mathsf{W}_L; \phi\right) = \prod_{l=1}^{L} \mathsf{q}\left(\mathsf{W}_l; \phi_l\right)$$

Over weights ("mean-field"):

$$= \prod_{l=1}^{L} \prod_{d=1}^{D_l} \mathsf{q}\left(\mathsf{w}_{l,d}; \phi_{l,d}\right)$$

Normals are common, for instance.

Over layers:

$$\mathsf{q}\left(\mathsf{W}_1, \ldots, \mathsf{W}_L; \phi\right) = \prod_{l=1}^{L} \mathsf{N}\left(\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l\right)$$

Over weights (“mean-field”):

$$= \prod_{l=1}^{L} \prod_{d=1}^{D_l} \mathsf{N}\left(\mu_{l,d,}, \sigma^2_{l,d}\right)$$

Optimization Objective

$$\phi^* = \mathrm{argmin}_{\phi}\ \mathbb{D}\left[\mathrm{q}\left(\mathrm{w};\phi\right) || \mathrm{p}\left(\mathrm{w} \mid \mathrm{y}, \mathrm{x}\right)\right]$$

Optimization Objective

$$\phi^* = \text{argmin}_\phi \ \text{KLD}\left[\text{q}\left(\text{w}; \phi\right) || \text{p}\left(\text{w} \,|\, \text{y}, \text{x}\right)\right]$$

Optimization Objective

$$\phi^* = \text{argmin}_\phi \ \text{KLD}\left[\text{q}\left(\text{w};\phi\right) || \text{p}\left(\text{w} \,|\, \text{y}, \text{x}\right)\right]$$

$$= \text{argmin}_\phi \int_{\text{w}} \text{q}\left(\text{w};\phi\right) \log \frac{\text{q}\left(\text{w};\phi\right)}{\text{p}\left(\text{w} \,|\, \text{y}, \text{x}\right)} \, d\text{w}$$

Optimization Objective

$$\mathrm{KLD}\left[\mathrm{q}\left(\mathrm{w};\phi\right)||\mathrm{p}\left(\mathrm{w}\,|\,\mathrm{y},\mathrm{x}\right)\right]=$$

Optimization Objective

$$\mathrm{KLD}\left[q\left(w;\phi\right)||\,p\left(w\,|\,y,x\right)\right] =$$

$$\mathbb{E}_{q_\phi}\left[-\log p\left(y\,|\,x,w\right)\right]+$$

$$\mathrm{KLD}\left[q(w;\phi)\,||\,p(w)\right] + \mathrm{const}\,.$$

Optimization Objective

$$\mathrm{KLD}\left[q\left(w;\phi\right)||\,p\left(w\,|\,y,x\right)\right] =$$

$$\mathbb{E}_{q_\phi}\left[-\log p\left(y\,|\,x,w\right)\right] +$$

$$\mathrm{KLD}\left[q(w;\phi)||\,p(w)\right] + \mathrm{const}\,.$$

Reparameterization Trick

$$\mathbb{E}_{q_\phi}\left[-\log p\left(y \mid x, w\right)\right]$$

Reparameterization Trick

$$\mathbb{E}_{\mathsf{q}_\phi}\left[-\log \mathsf{p}\left(\mathsf{y} \,|\, \mathsf{x}, \mathsf{w}\right)\right]$$

$$= \mathbb{E}_\eta\left[-\log \mathsf{p}\left(\mathsf{y} \,|\, \mathsf{x}, \mathsf{w} = g(\eta;\phi)\right)\right]$$

Reparameterization Trick

$$\boxed{\mathbb{E}_{\mathsf{q}_\phi}\left[-\log \mathsf{p}\left(\mathsf{y} \,|\, \mathsf{x}, \mathsf{w}\right)\right]}$$

$$= \mathbb{E}_{\eta}\left[-\log \mathsf{p}\left(\mathsf{y} \,|\, \mathsf{x}, \mathsf{w} = g(\eta; \phi)\right)\right]$$

$$\approx \frac{1}{S}\sum_{s} -\log \mathsf{p}\left(\mathsf{y} \,|\, \mathsf{x}, \mathsf{w} = g(\hat{\eta}_s; \phi)\right)$$

$$\frac{\partial}{\partial\ \phi}\mathbb{E}_{\mathsf{q}_\phi}\left[-\log \mathsf{p}\left(\mathsf{y} \,|\, \mathsf{x}, \mathsf{w}\right)\right]$$

$$\approx -\frac{1}{S}\sum_{s}\frac{\partial}{\partial\phi}\log \mathsf{p}\left(\mathsf{y} \,|\, \mathsf{x}, \mathsf{w} = g(\hat{\eta}_s;\phi)\right)$$

$$\frac{\partial}{\partial \ \phi} \mathbb{E}_{\mathsf{q}_\phi} \left[-\log \mathsf{p} \left(\mathsf{y} \,|\, \mathsf{x}, \mathsf{w} \right) \right]$$

$$\approx -\frac{1}{S} \sum_s \frac{\partial}{\partial \phi} \log \mathsf{p} \left(\mathsf{y} \,|\, \mathsf{x}, \mathsf{w} = g(\hat{\eta}_s; \phi) \right)$$

Blundell et al., 2015

"Bayes by Backprop"

If $\mathsf{q}(\mathsf{w};\phi) = \mathsf{N}(\mu,\sigma)$:

$$\hat{\mathsf{w}} = g(\hat{\eta};\phi) = \mu + \sigma \cdot \hat{\eta}, \quad \eta \sim \mathsf{N}(0,1)$$

If $\mathsf{q}(\mathsf{w};\phi) = \mathsf{N}(\mu,\sigma)$:

$$\hat{\mathsf{w}} = g(\hat{\eta};\phi) = \mu + \sigma \cdot \hat{\eta}, \quad \eta \sim \mathsf{N}(0,1)$$

$$h \cdot \hat{\mathsf{w}} = h \cdot (\mu + \sigma \cdot \hat{\eta})$$

If $\mathsf{q}(\mathsf{w}; \phi) = \mathsf{N}(\mu, \sigma)$:

$$\hat{\mathsf{w}} = g(\hat{\eta}; \phi) = \mu + \sigma \cdot \hat{\eta}, \quad \eta \sim \mathsf{N}(0,1)$$

$$h \cdot \hat{\mathsf{w}} = h \cdot (\mu + \sigma \cdot \hat{\eta})$$

Or for a general q:

$$\hat{\mathsf{w}} = \mathsf{CDF}_q^{-1}(\hat{\eta}; \phi), \quad \eta \sim \mathsf{Uniform}(0,1)$$

MCMC for ResNet-20 on CIFAR-10

Metric	HMC (reference)	MFVI	SGLD	SGHMC	SGHMC CLR	SGHMC CLR-Prec
			SGMCMC			
Accuracy	89.64 ±0.25	86.45 ±0.27	89.32 ±0.23	89.38 ±0.32	**89.63 ±0.37**	87.46 ±0.21
Agreement	94.01 ±0.25	88.75 ±0.24	91.54 ±0.15	91.98 ±0.35	**92.67 ±0.52**	90.96 ±0.24
Total Var	0.074 ±0.003	0.136 ±0.000	0.110 ±0.001	0.109 ±0.001	**0.099 ±0.006**	0.111 ±0.002

Laplace Approximation

$$p\left(w \mid y, x\right)$$

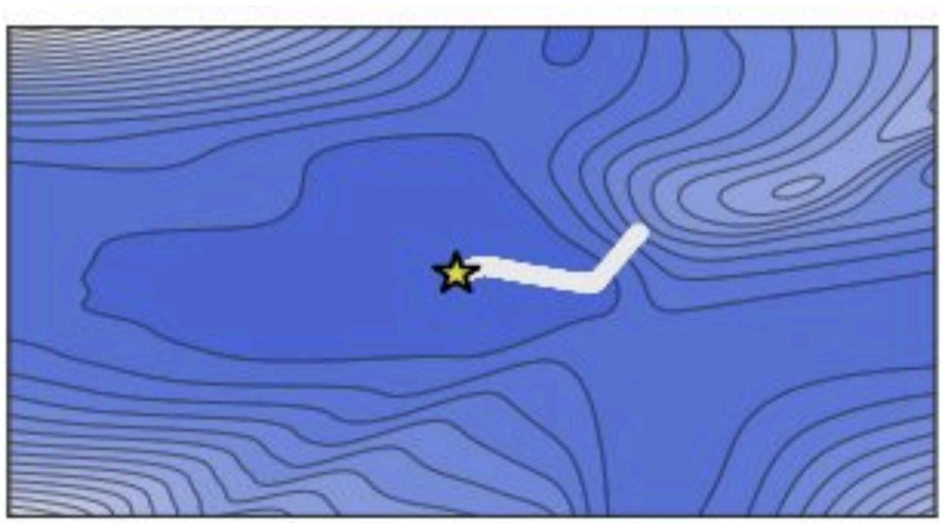
$$\approx N\left(\hat{w}_{\mathrm{MAP}}, \bar{H}^{-1}(\hat{w}_{\mathrm{MAP}})\right)$$

Laplace Approximation

$$p\left(\mathsf{w} \mid \mathsf{y}, \mathsf{x}\right) \approx \mathsf{N}\left(\hat{\mathsf{w}}_{\mathsf{MAP}}, \boxed{\bar{\mathsf{H}}^{-1}(\hat{\mathsf{w}}_{\mathsf{MAP}})}\right)$$

$$\bar{\mathsf{H}}(\mathsf{w}) = -\sum_{n=1}^{N} \frac{\partial^2 \log \mathsf{p}(\mathsf{y}_n, \mathsf{w} \mid \mathsf{x}_n)}{\partial \mathsf{w}^2}$$

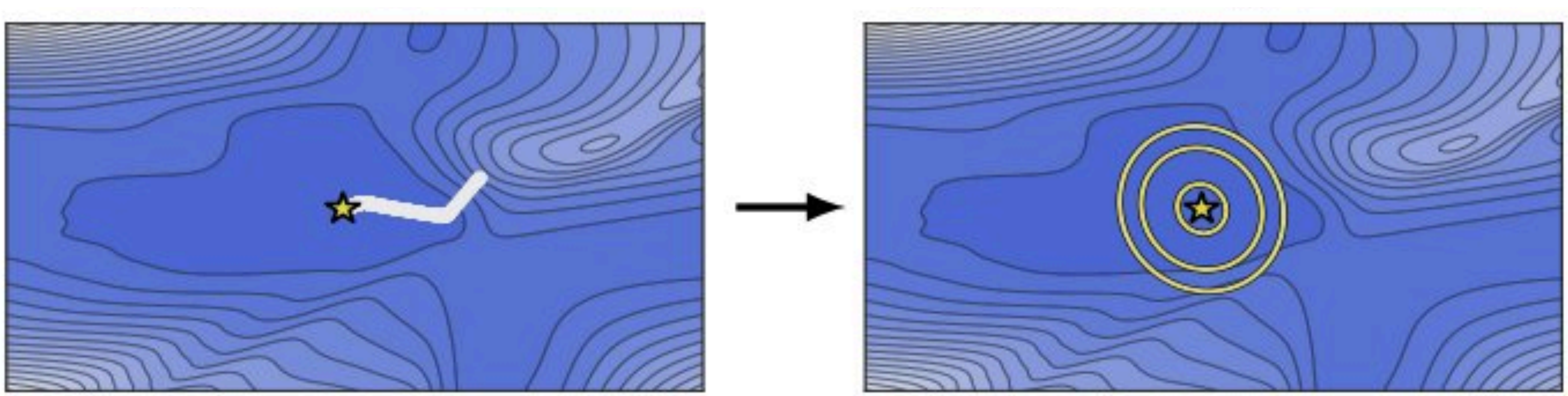
Laplace Approximation



find $\hat{w}_{MAP}$

Images from Alexander Immer: https://twitter.com/a1mmer/status/1454057890864566272

Laplace Approximation



find $\hat{\mathrm{w}}_{\mathrm{MAP}}$ $\quad$ $\mathrm{N}\left(\hat{\mathrm{w}}_{\mathrm{MAP}}, \mathrm{H}^{-1}(\hat{\mathrm{w}}_{\mathrm{MAP}})\right)$

Images from Alexander Immer: https://twitter.com/a1mmer/status/1454057890864566272

Laplace Approximation

- ⊗ Pro: can apply to a pre-trained model by assuming parameters are at the 'MAP'
- ⊗ Con: Hessian matrix can be numerically unstable, need to assume structure (e.g. low-rank, diagonal).

Summary

- **Conjugacy** for last layer (sometimes)
- **MCMC** is possible but will require approximations
- **Variational inference** is practical but usually has inferior performance (compared to MCMC).