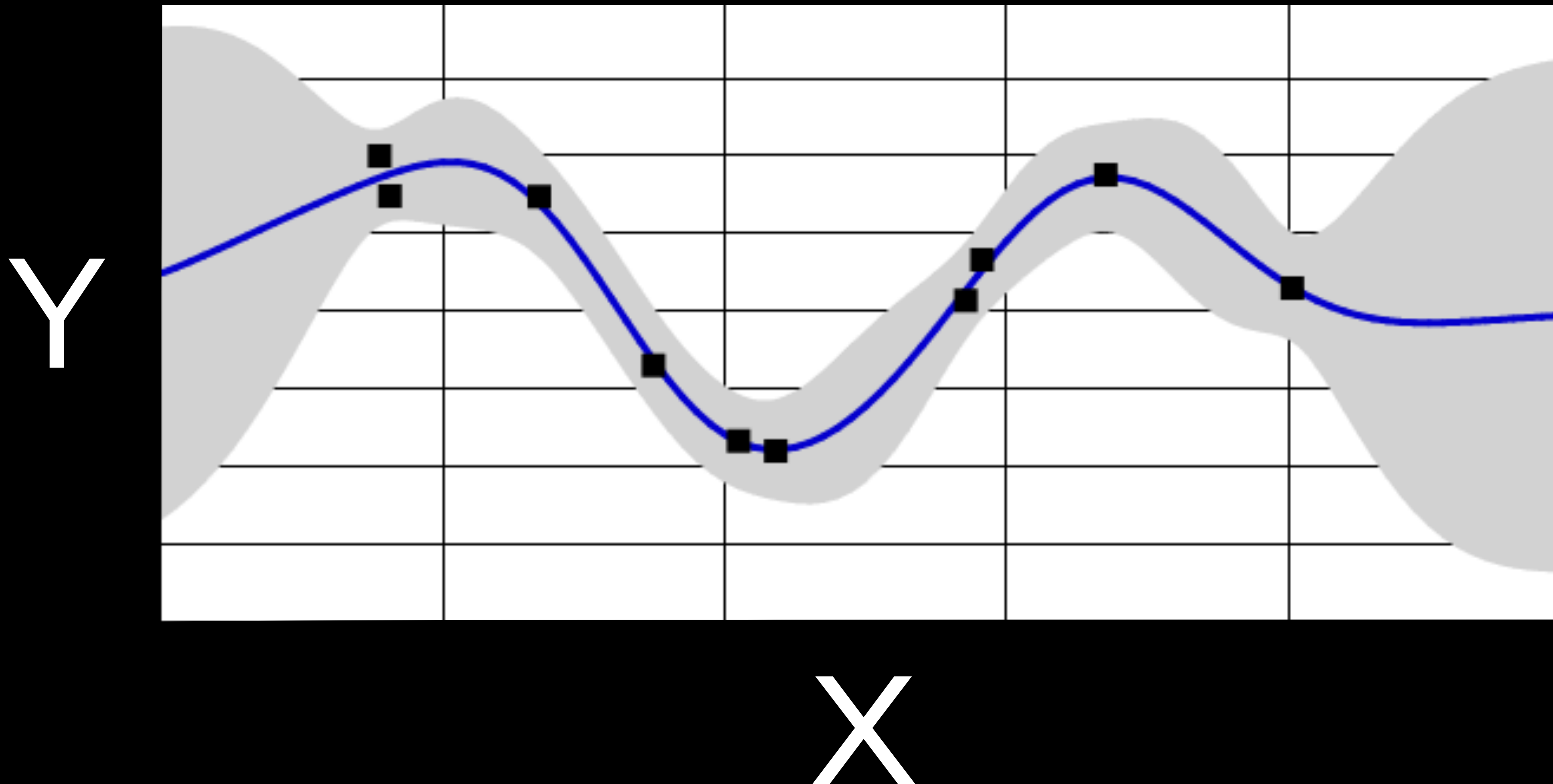
The Boons of Being Less Bayesian

a study of partially stochastic neural networks

Eric Nalisnick

Johns Hopkins University



Are stochastic parameters always useful?

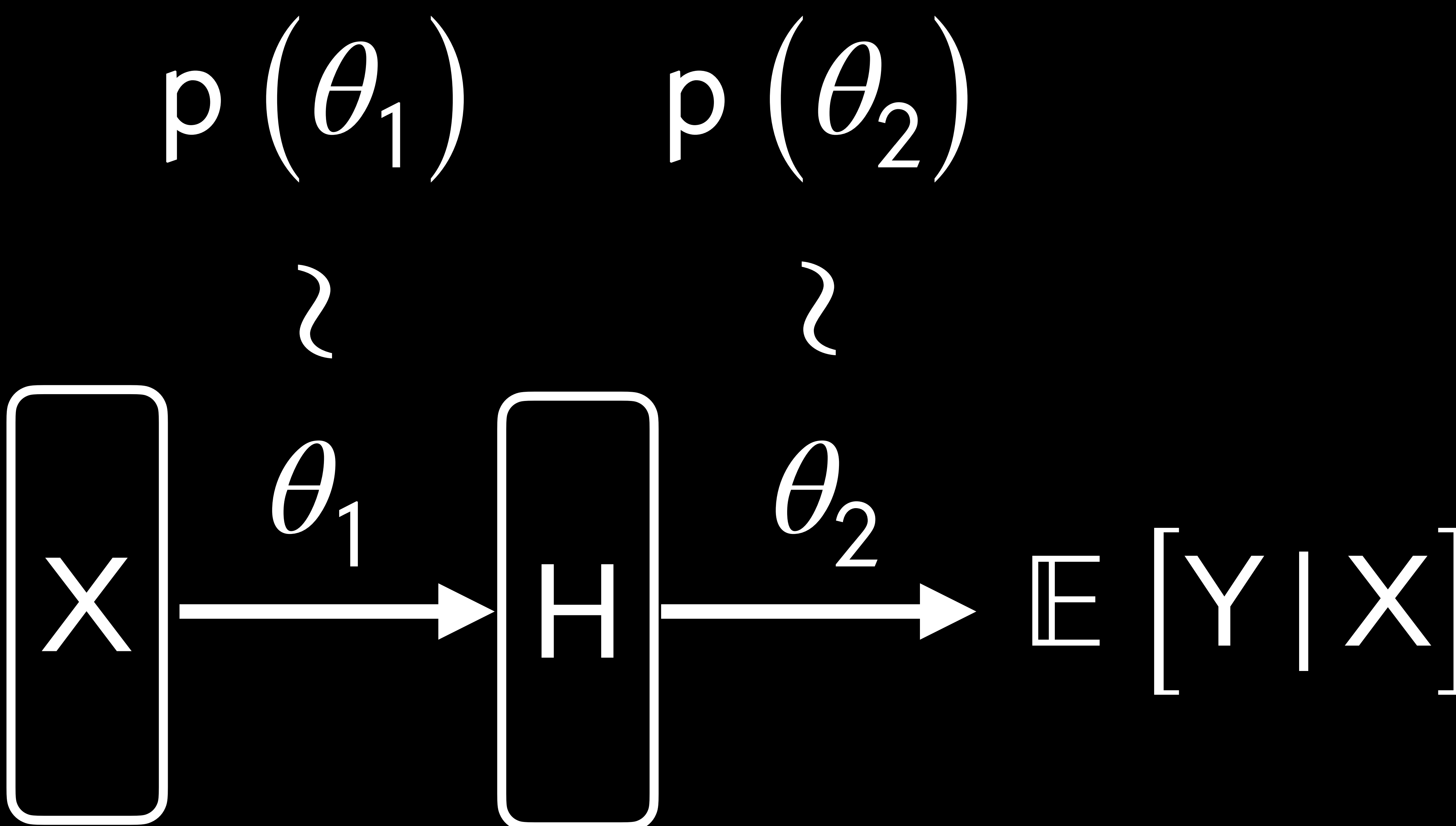
Outline

- ⊗ background
- ⊗ subnetwork inference algorithm
- ⊗ subnetworks are all you need

Outline

- ⊗ background
- ⊗ subnetwork inference algorithm
- ⊗ subnetworks are all you need

Neural Network



Bayes Rule

$$p\left(\theta_1, \ldots, \theta_L \,|\, \mathfrak{D}\right) =$$

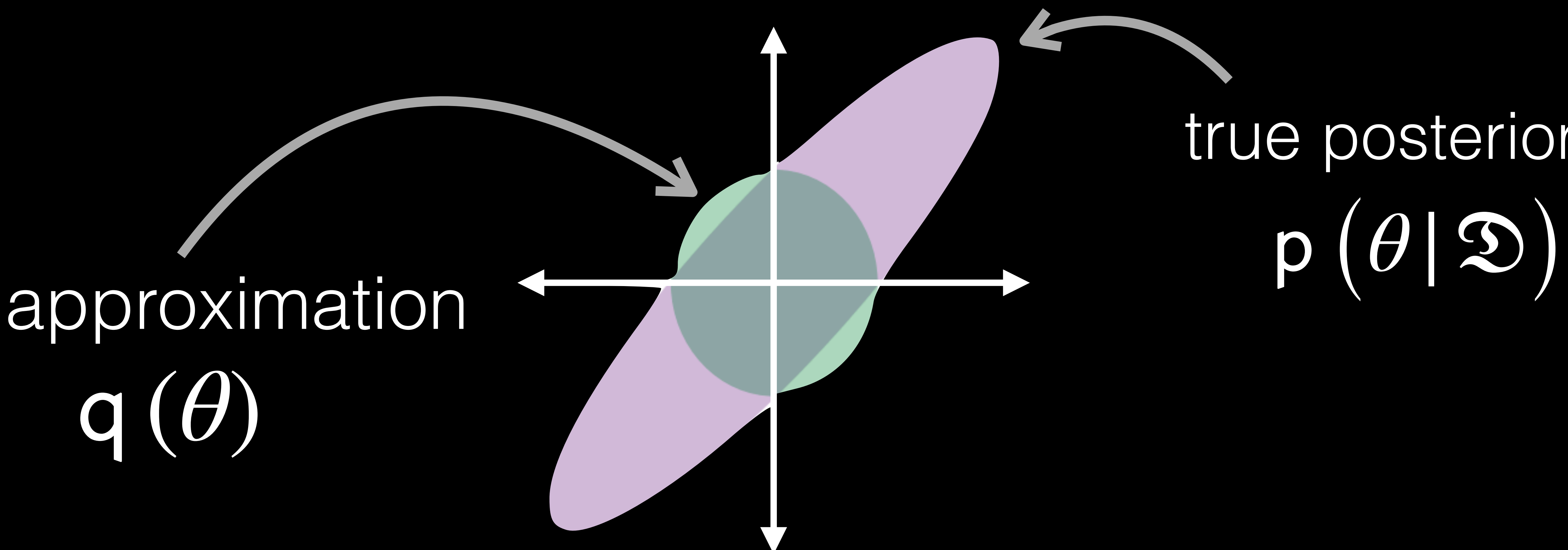
$$\frac{\prod_l p\left(\theta_l\right) \;\; \prod_{n=1}^{N} p\left(y_n \,|\, x_n, \theta_1, \cdots, \theta_L\right)}{p(\mathfrak{D})}$$

Bayes Rule

$$\cancel{p\left(\theta_1, \ldots, \theta_L \mid \mathfrak{D}\right)} =$$

$$\frac{\prod_l p\left(\theta_l\right) \ \prod_{n=1}^{N} p\left(y_n \mid x_n, \theta_1, \cdots, \theta_L\right)}{\cancel{p(\mathfrak{D})}}$$

Variational Inference



Mean Field Assumption

$$\mathrm{p}\left(\theta_1, \ldots, \theta_\mathrm{L} \mid \mathfrak{D}\right) \approx \mathrm{q}\left(\theta_1, \ldots, \theta_\mathrm{L}\right)$$

Mean Field Assumption

$$p\left(\theta_1, \ldots, \theta_L \mid \mathfrak{D}\right) \approx q\left(\theta_1, \ldots, \theta_L\right)$$

$$= \prod_{l}^{L} q\left(\theta_l\right)$$

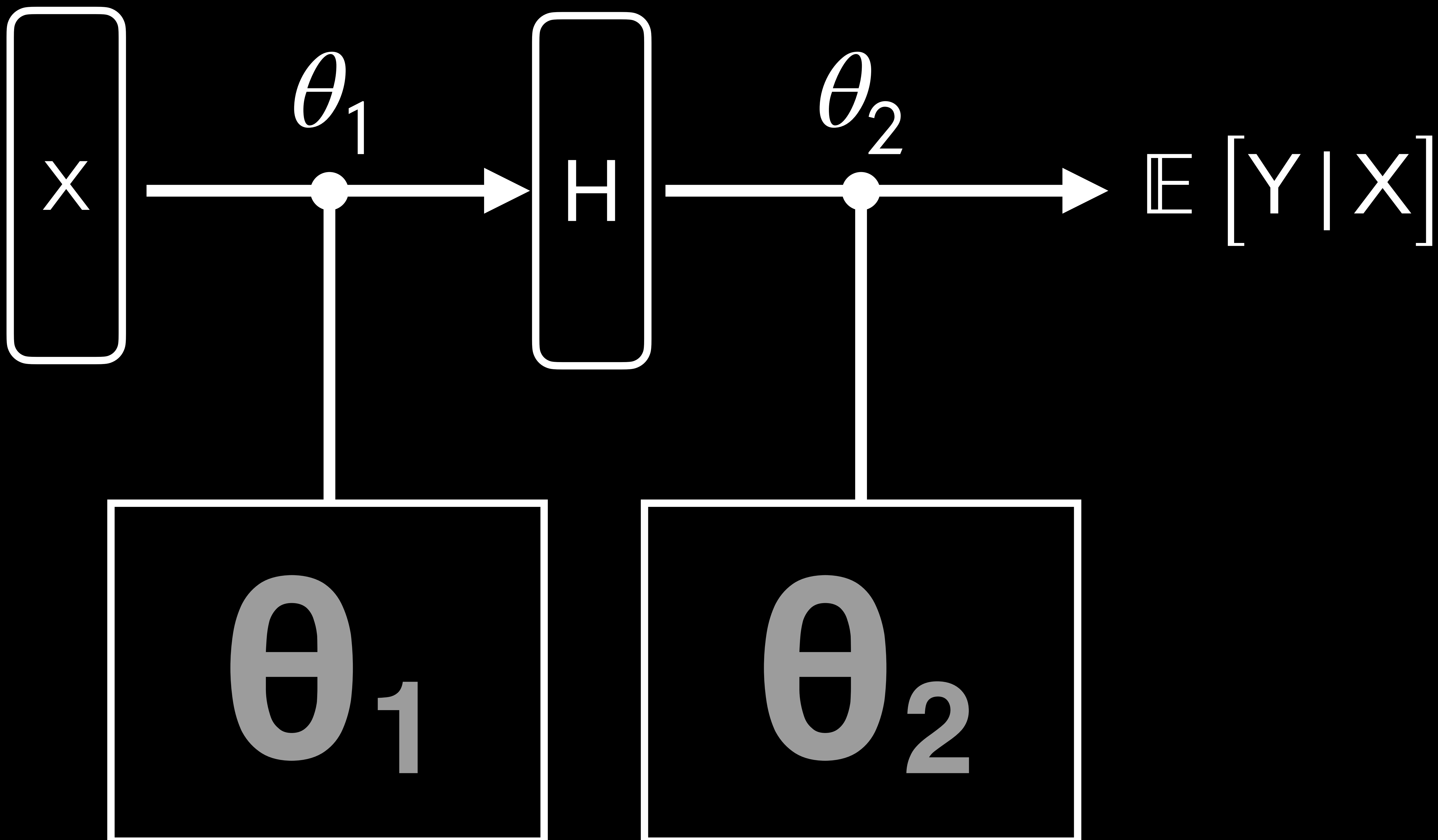
factorize over layers

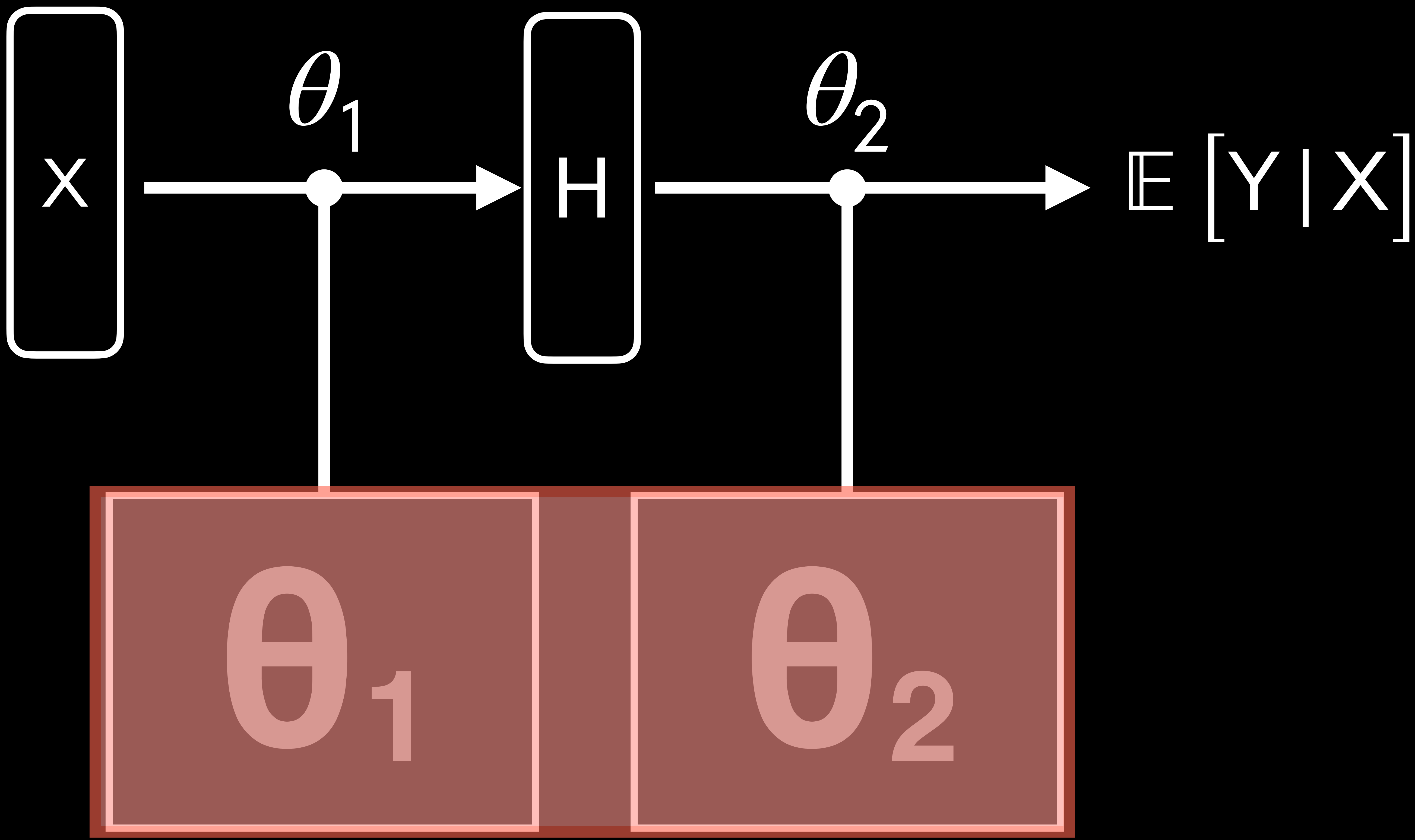
Mean Field Assumption

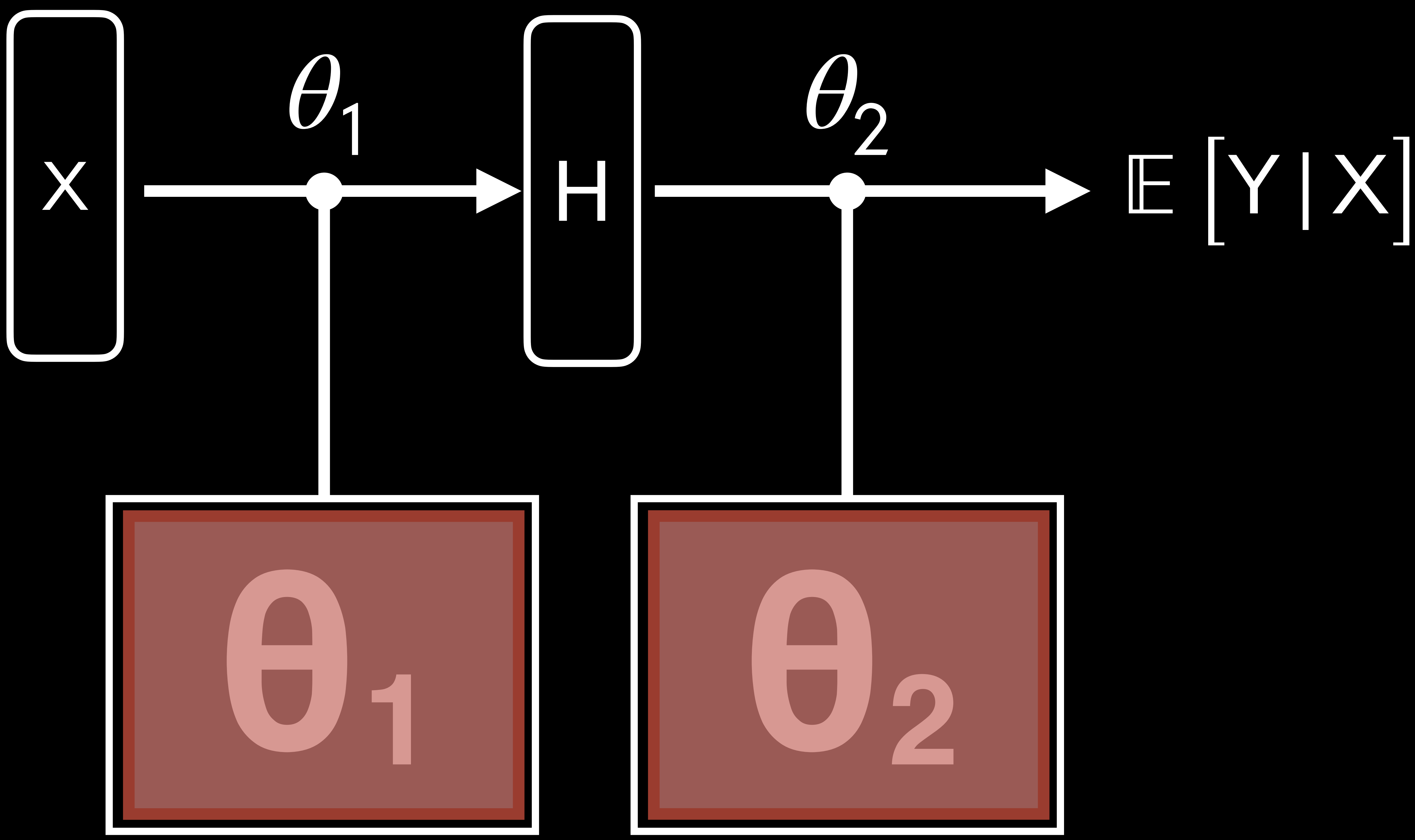
$$\mathrm{p}\left(\theta_1, \ldots, \theta_L \mid \mathfrak{D}\right) \approx \mathrm{q}\left(\theta_1, \ldots, \theta_L\right)$$

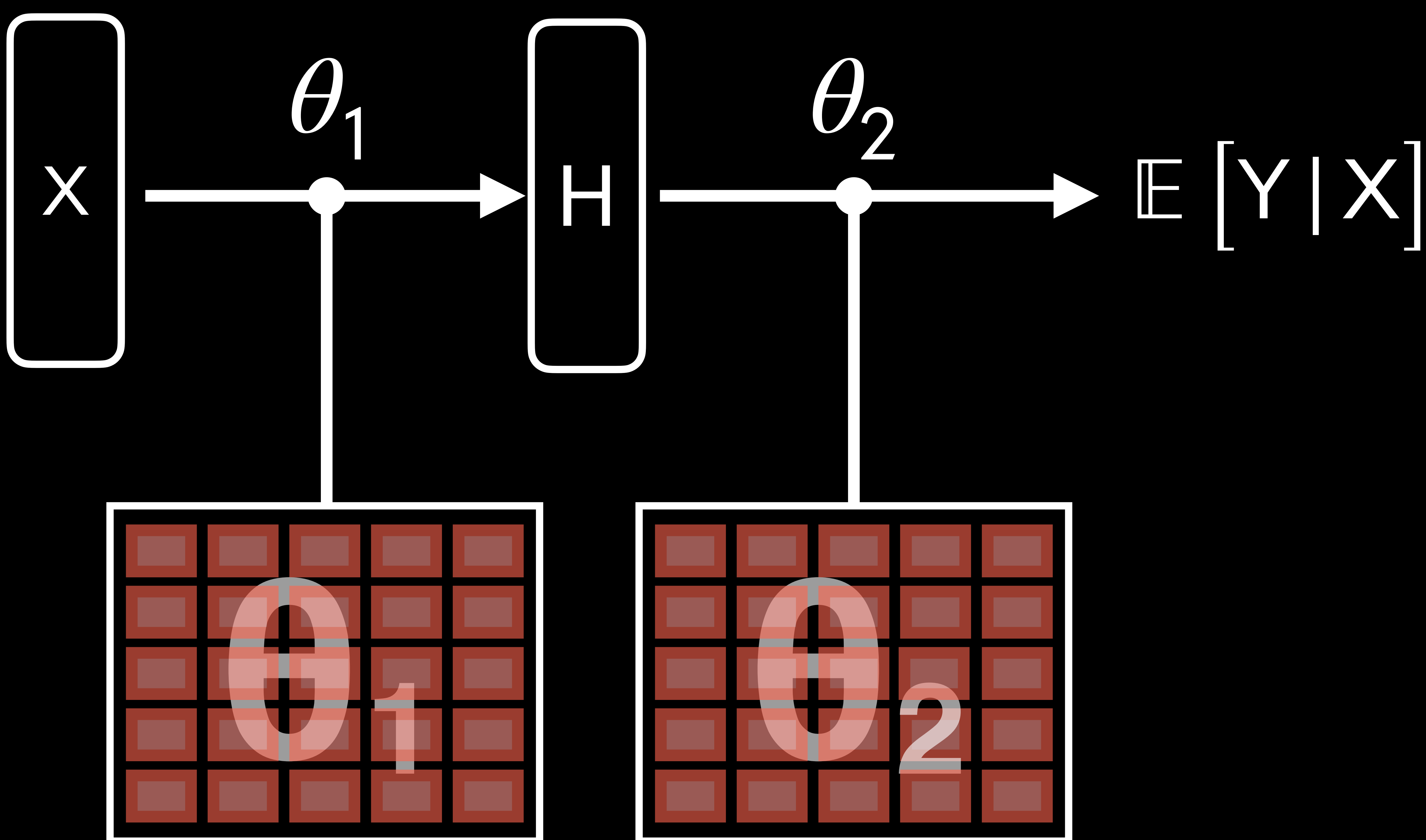
$$= \prod_{l}^{L} \mathrm{q}\left(\theta_l\right) \qquad = \prod_{l}^{L} \prod_{d}^{D_l} \mathrm{q}\left(\theta_{l,d}\right)$$

factorize over layers   factorize within layers


X
θ_1
H
θ_2
$\mathbb{E}[Y|X]$
θ_1
θ_2


X
θ_1
H
θ_2
$\mathbb{E}\left[Y|X\right]$
θ_1
θ_2


X
θ_1
H
θ_2
$\mathbb{E}[Y|X]$
θ_1
θ_2


X
θ_1
H
θ_2
$\mathbb{E}\left[Y|X\right]$
Θ_1
Θ_2

Outline

- ⊗ background
- ⊗ subnetwork inference algorithm
- ⊗ subnetworks are all you need

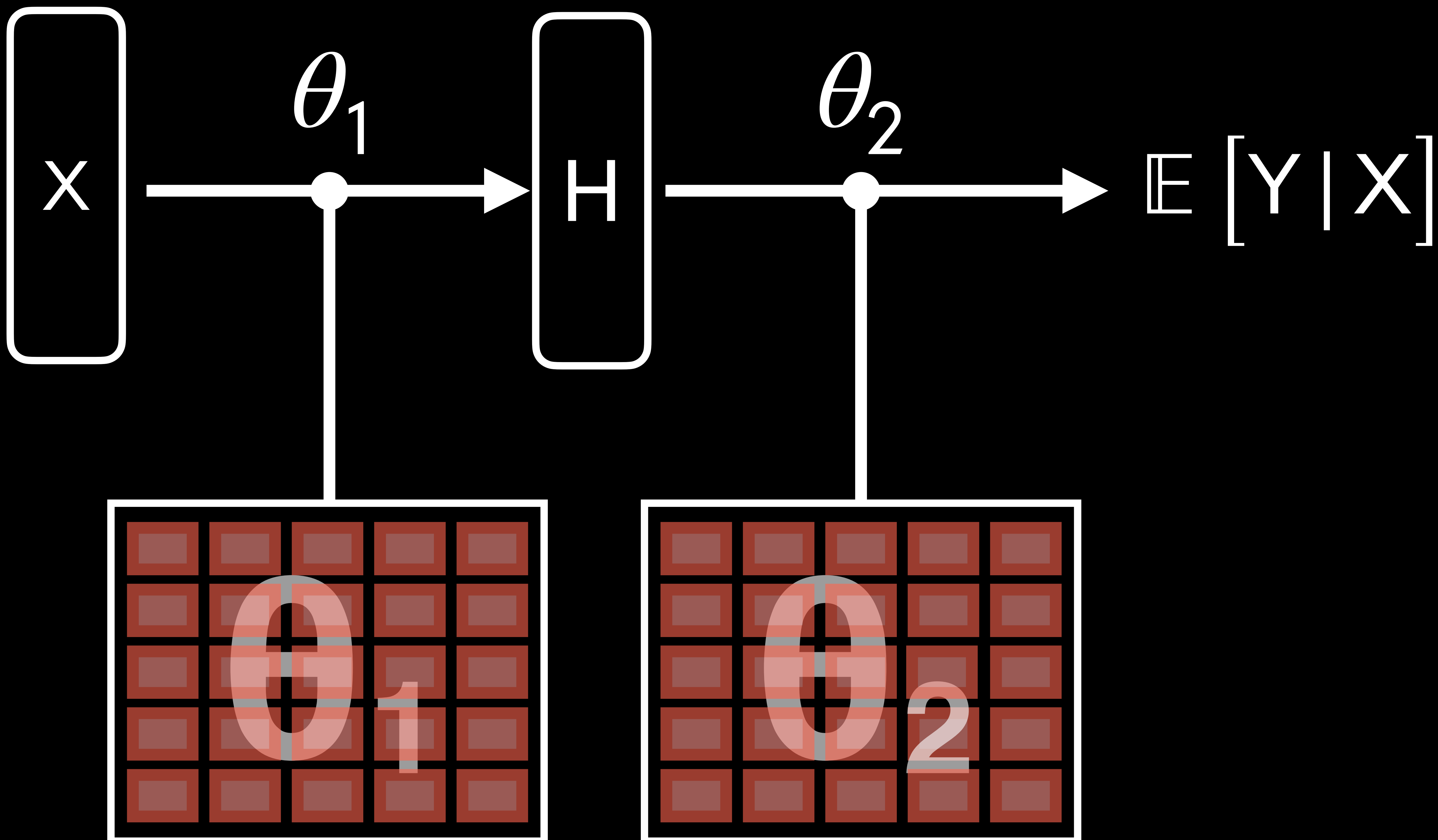
Outline

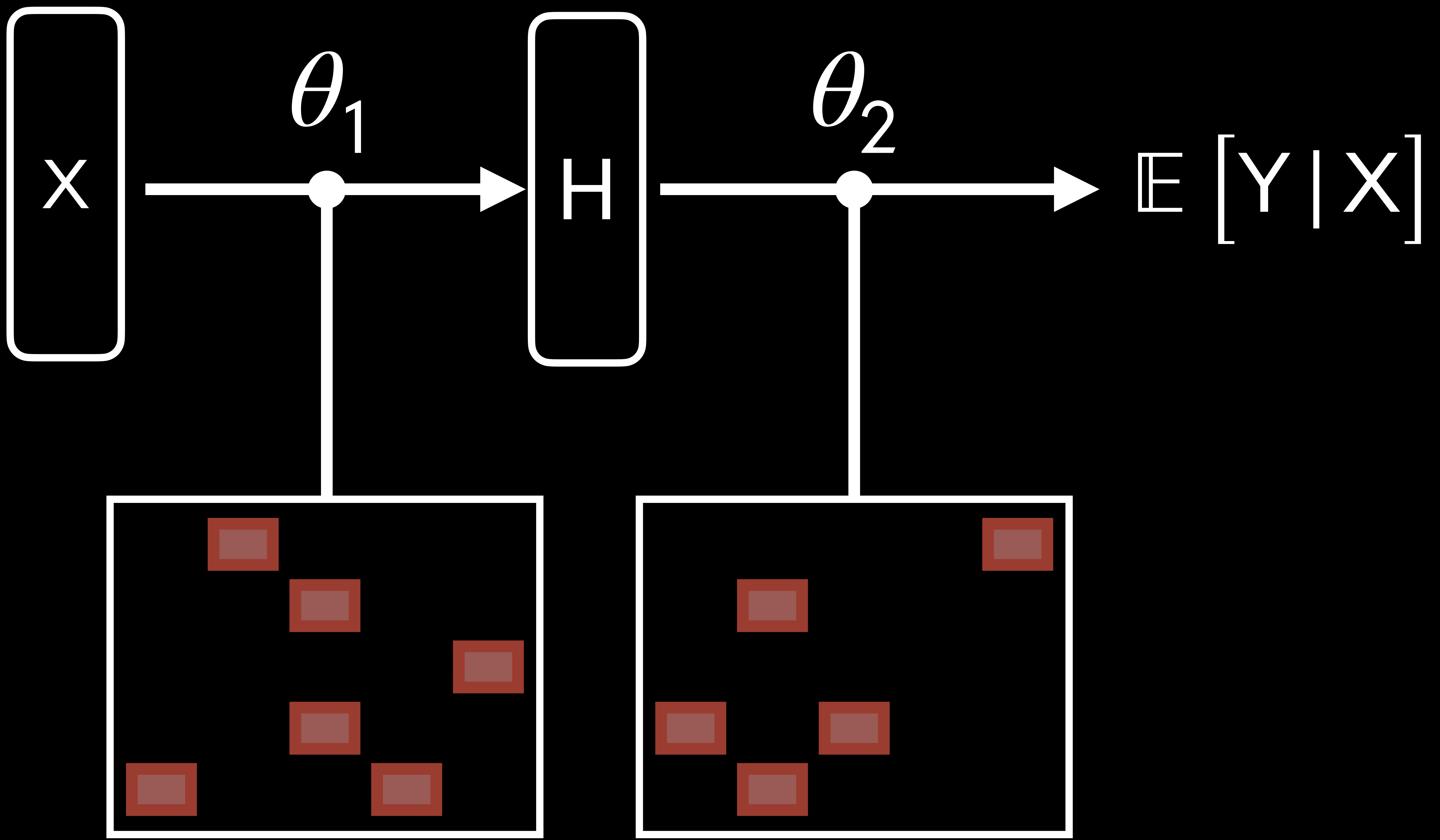
- ⊗ background
- ⊗ subnetwork inference algorithm
- ⊗ subnetworks are all you need

Lottery Ticket Hypothesis

Lottery Ticket Hypothesis: *feed-forward networks contain subnetworks ("winning tickets") that—when trained in isolation—reach test accuracy comparable to the original network.*

[Frankle & Carbin, ICLR 2019 Best Paper]


X
θ_1
H
θ_2
$\mathbb{E}\left[Y|X\right]$
Θ_1
Θ_2


X
θ_1
H
θ_2
$\mathbb{E}[Y|X]$

Lottery Ticket Hypothesis for BNNs

Can the posterior distribution over all weights be represented as a posterior over a subnetwork? (in terms of inducing equivalent predictive distributions)

Lottery Ticket Hypothesis for BNNs

$$\mathrm{p}\left(\theta_{1},\ldots,\theta_{\mathrm{L}} \,|\, \mathfrak{D}\right) \stackrel{?}{=}$$

$$\mathrm{p}\left(\left\{\theta_{\mathrm{s}} \,|\, \mathrm{s} \in \mathbb{S}\right\} \,|\, \mathfrak{D}\right) \cdot \prod_{r \in \mathbb{R}} \delta\left[\theta_{r} - \bar{\theta}_{r}\right]$$

Problem

We can't simply find the true, complete posterior and prune it, analogously to how Frankle & Carbin [2019] find their subnetworks.

So we gave up on investigating a LTH for BNNs…but I'll return to this topic later.

Subnetwork Variational Inference

$$\mathsf{p}\left(\theta_1, \ldots, \theta_\mathsf{L} \,|\, \mathfrak{D}\right) \stackrel{?}{=}$$

$$\mathsf{p}\left(\left\{\theta_\mathsf{s} \,|\, \mathsf{s} \in \mathbb{S}\right\} \,|\, \mathfrak{D}\right) \cdot \prod_{r \in \mathbb{R}} \delta\left[\theta_r - \bar{\theta}_r\right]$$

$$\approx \; \mathsf{q}\left(\left\{\theta_\mathsf{s} \,|\, \mathsf{s} \in \tilde{\mathbb{S}}\right\}\right) \cdot \prod_{r \in \tilde{\mathbb{R}}} \delta\left[\theta_r - \bar{\theta}_r\right]$$

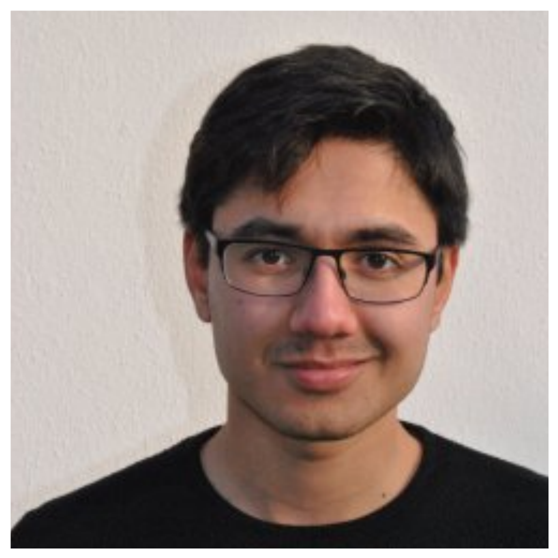
Subnetwork Variational Inference

Can we have a posterior approximation whose structure is data- / learning- driven?

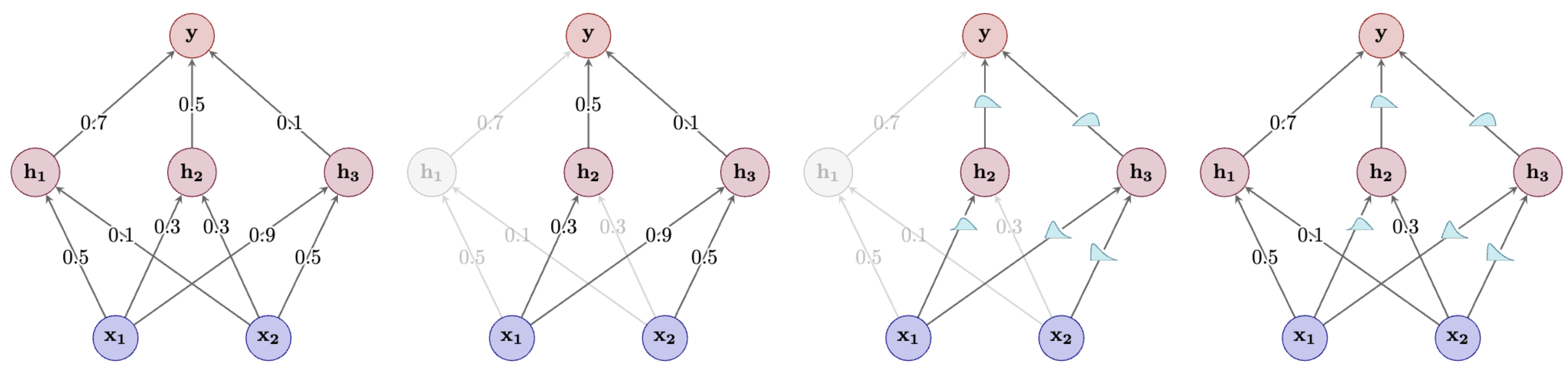
Is it better to perform high-quality, expensive inference over a few parameters than poor, cheap inference over many parameters?

Bayesian Deep Learning via Subnetwork Inference

Erik Daxberger [1,2] Eric Nalisnick [*,3] James Urquhart Allingham [*,1] Javier Antorán [*,1]
José Miguel Hernández-Lobato [1,4,5]

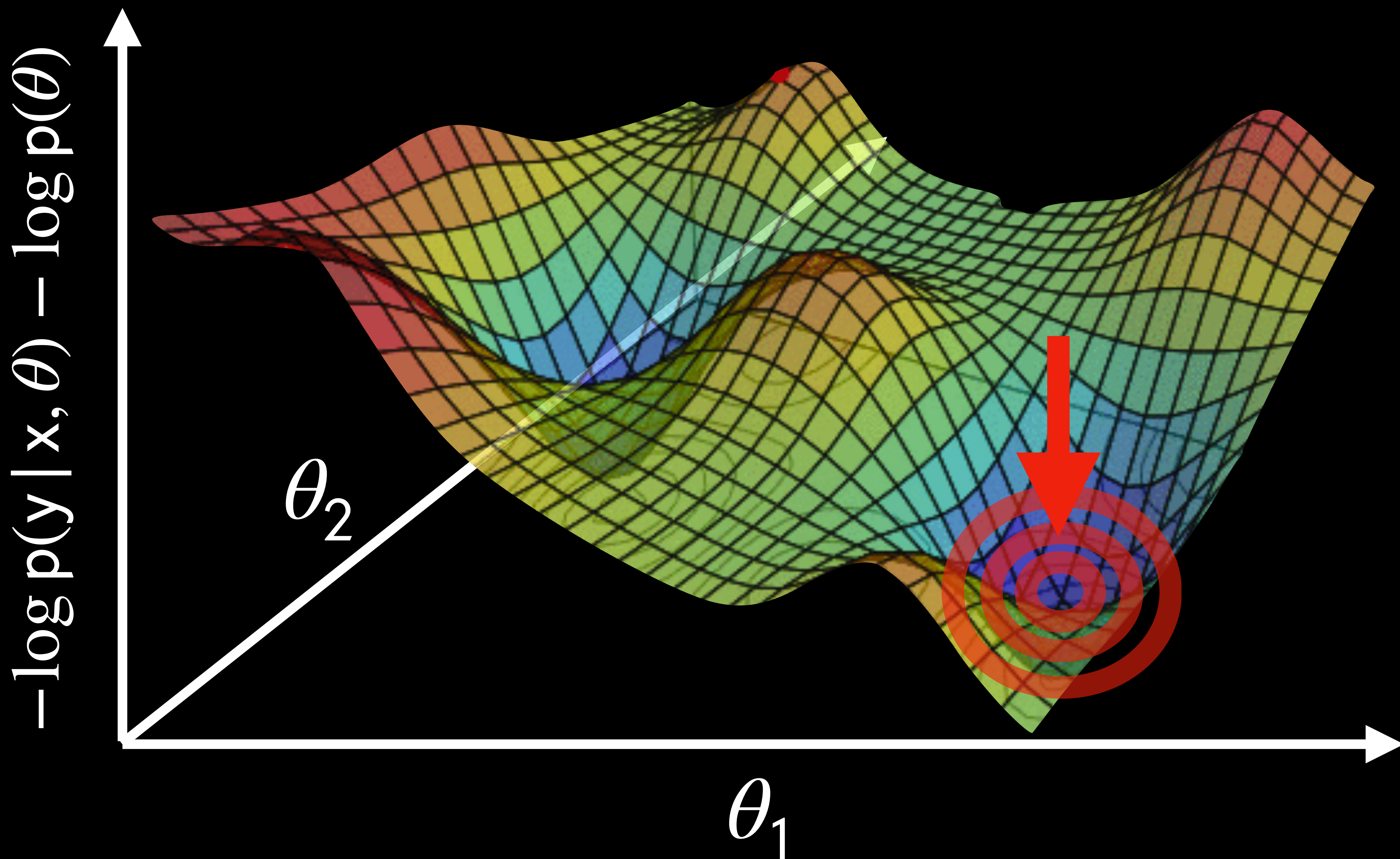


Erik Daxberger

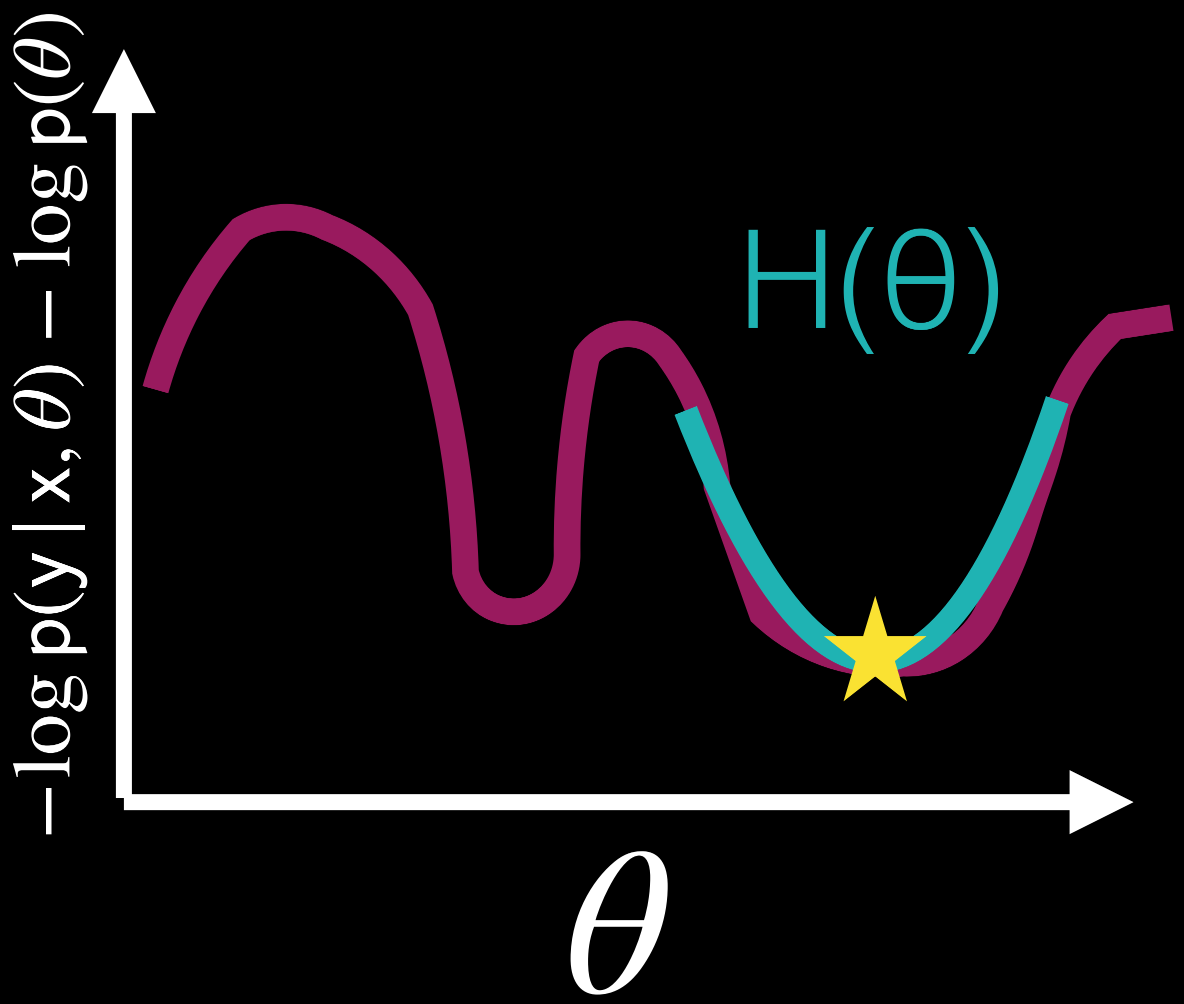


(a) Point Estimation (b) Subnetwork Selection (c) Bayesian Inference (d) Prediction

Laplace Approximation



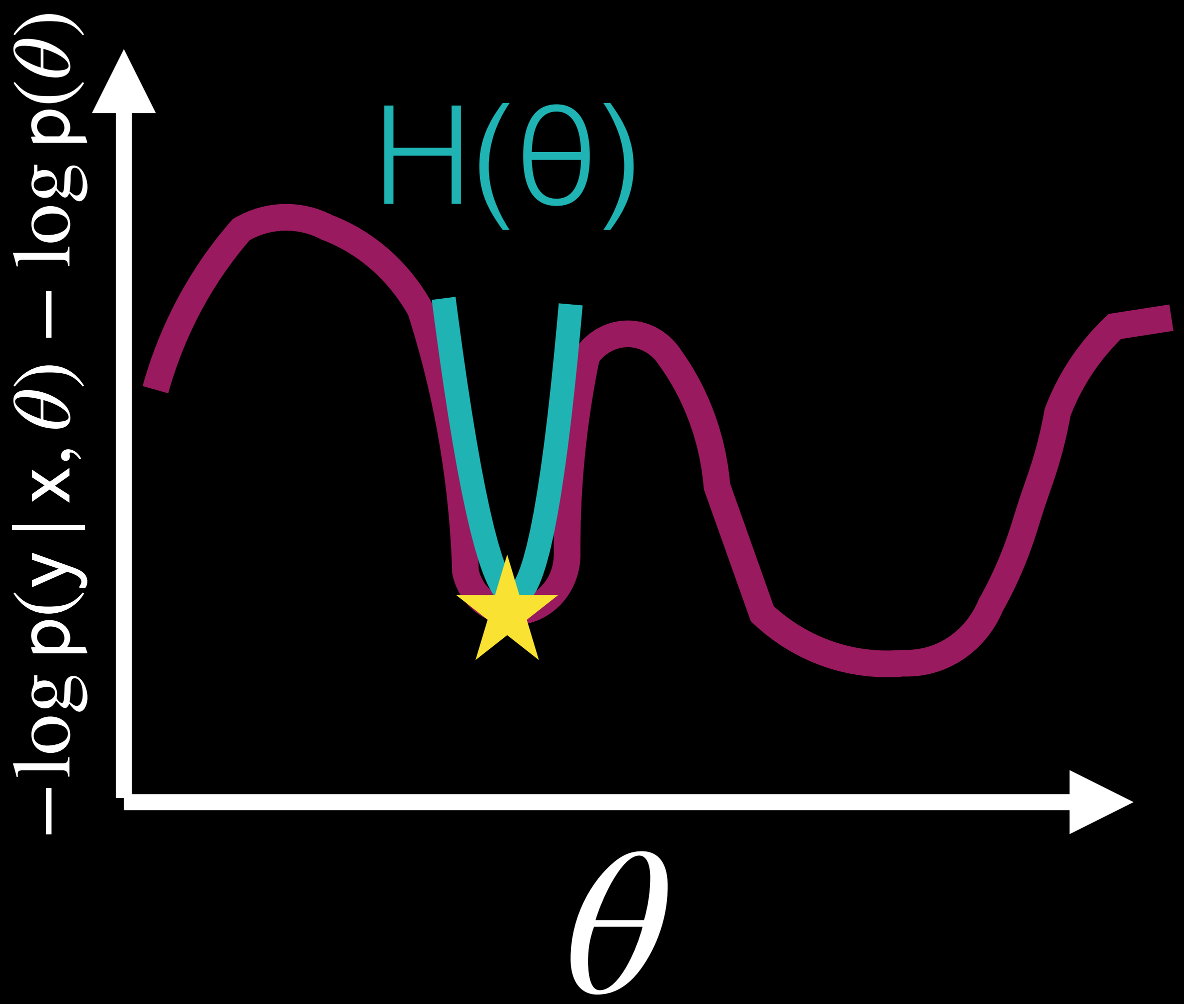
Laplace Approximation



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

small curvature,
large posterior variance

Laplace Approximation



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

large curvature,
small posterior variance

Subnetwork Laplace Approximation

1. Find MAP estimate for all weights
2. Select subnetwork via heuristic
3. Construct the Laplace approximation, with a full covariance matrix, over the subnetwork
4. Compute predictive distribution as usual with the posterior from step #3.

Subnetwork Laplace Approximation

1. Find MAP estimate for all weights
2. Select subnetwork via heuristic
3. Construct the Laplace approximation, with a full covariance matrix, over the subnetwork
4. Compute predictive distribution as usual with the posterior from step #3.

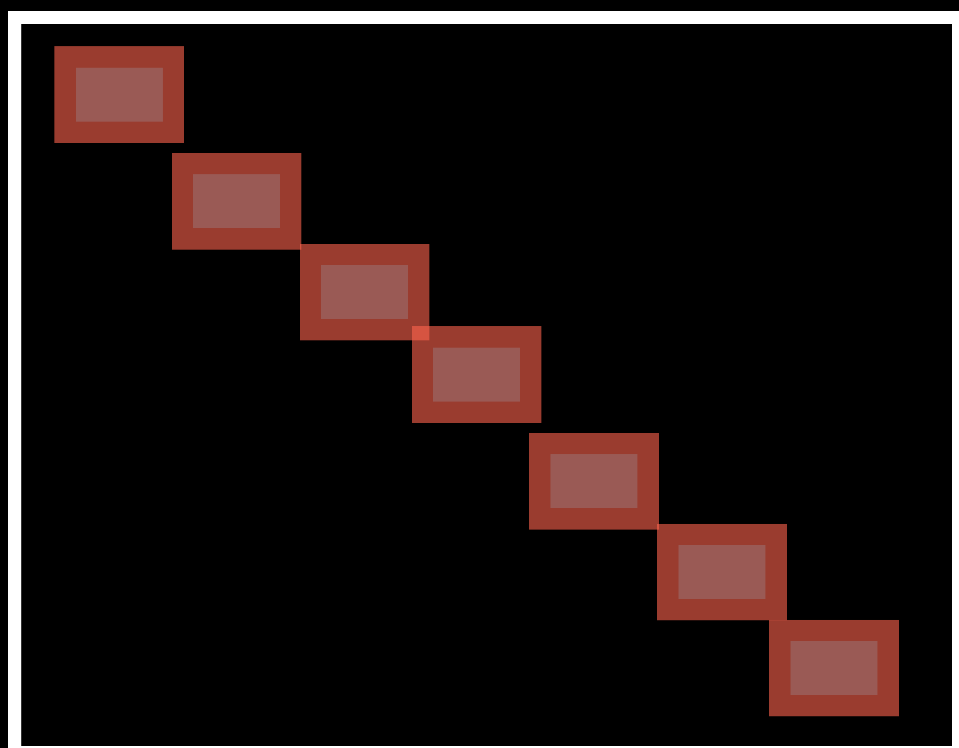
Subnetwork Laplace Approximation

1. Find MAP estimate for all weights
2. Select subnetwork via heuristic
3. Construct the Laplace approximation, with a full covariance matrix, over the subnetwork
4. Compute predictive distribution as usual with the posterior from step #3.

Subnetwork Selection

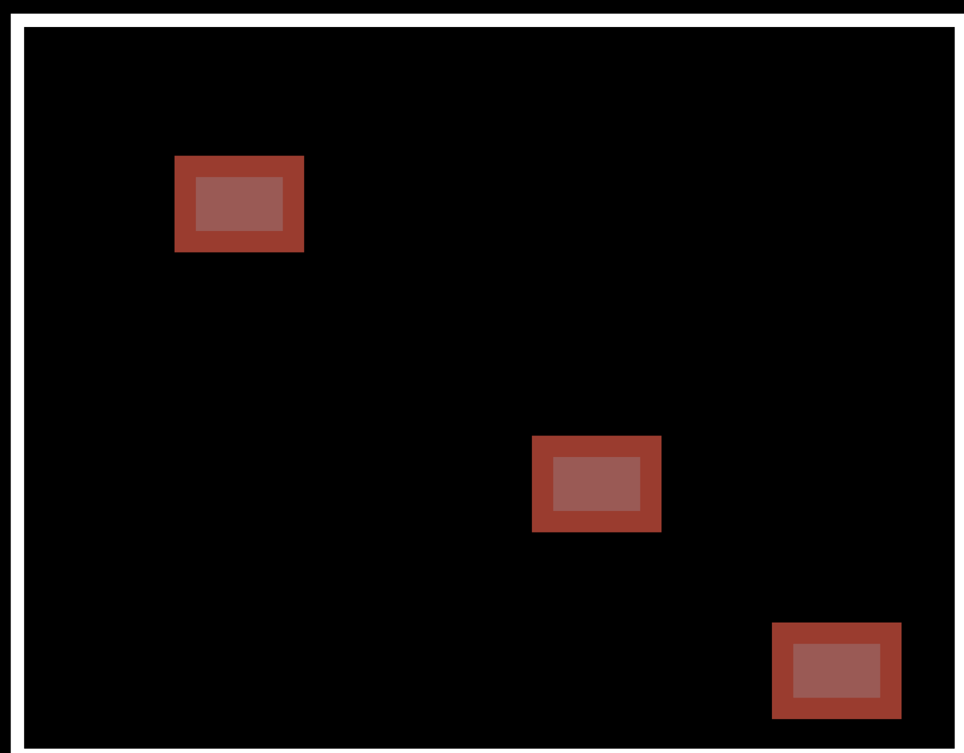
$$\mathsf{H}^{-1}\left(\hat{\theta}_{\mathsf{MAP}}\right)$$

Subnetwork Selection



$$\mathrm{H}^{-1}\left(\hat{\theta}_{\mathrm{MAP}}\right)$$

Subnetwork Selection



$$\mathrm{H}^{-1}\left(\hat{\theta}_{\mathrm{MAP}}\right)$$

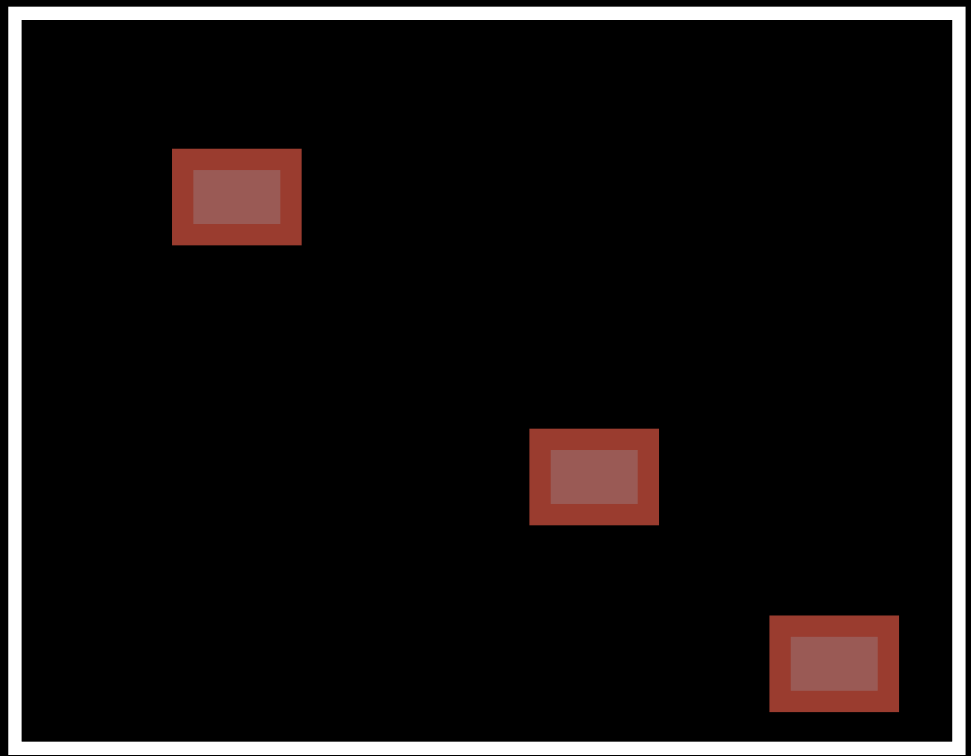
Subnetwork Laplace Approximation

1. Find MAP estimate for all weights
2. Select subnetwork via heuristic
3. Construct the Laplace approximation, with a full covariance matrix, over the subnetwork
4. Compute predictive distribution as usual with the posterior from step #3.

Subnetwork Laplace Approximation

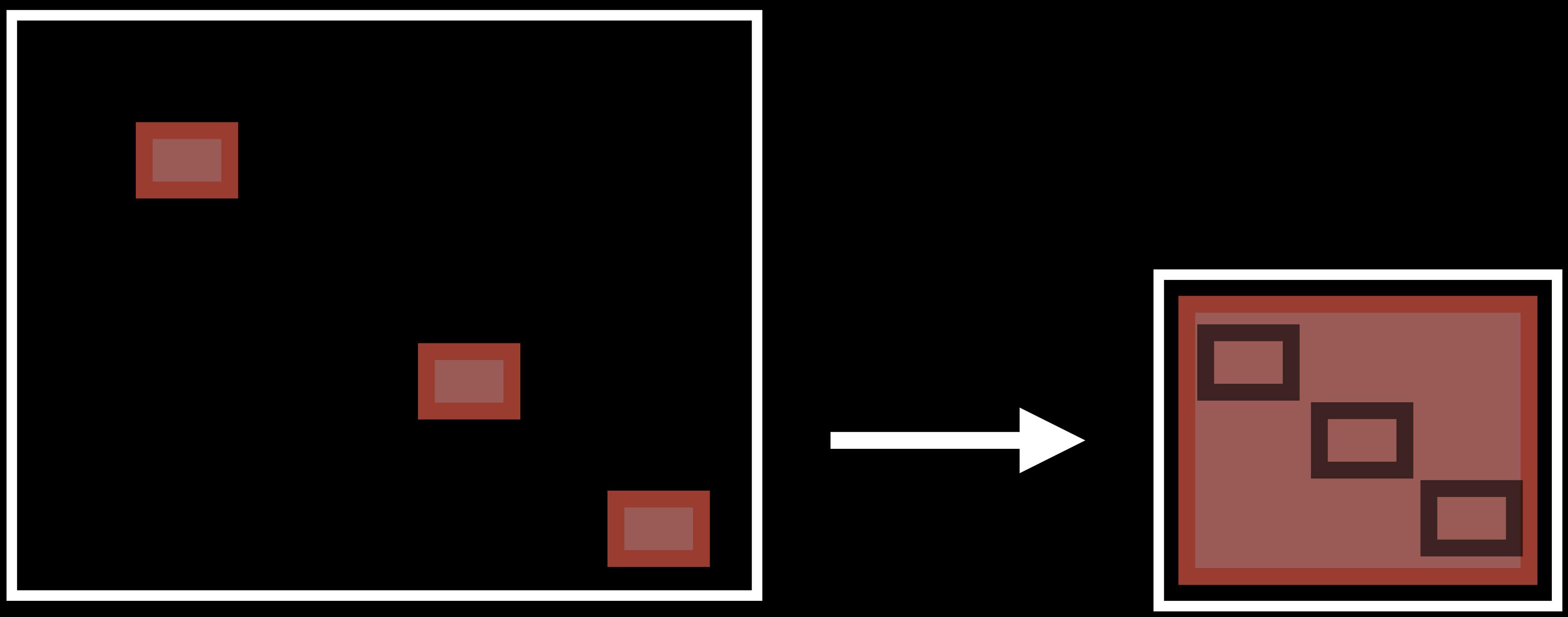
1. Find MAP estimate for all weights
2. Select subnetwork via heuristic
3. Construct the Laplace approximation, with a full covariance matrix, over the subnetwork
4. Compute predictive distribution as usual with the posterior from step #3.

Posterior Construction



$$\mathsf{H}^{-1}\left(\hat{\theta}_{\mathsf{MAP}}\right)$$

Posterior Construction



$$\mathsf{H}^{-1}\left(\hat{\theta}_{\mathsf{MAP}}\right) \qquad \mathsf{H}^{-1}\left(\hat{\theta}_{\mathbb{S}}\right)$$

Posterior Construction

$$\mathrm{p}(\theta|\mathfrak{D}) \approx$$

$$\mathrm{N}\left(\hat{\theta}_{\mathbb{S}}, \mathrm{H}^{-1}(\hat{\theta}_{\mathbb{S}})\right) \cdot \prod_{\mathrm{r}\in\mathbb{R}} \delta\left[\theta_{\mathrm{r}} - \hat{\theta}_{\mathrm{r}}\right]$$

Subnetwork Laplace Approximation

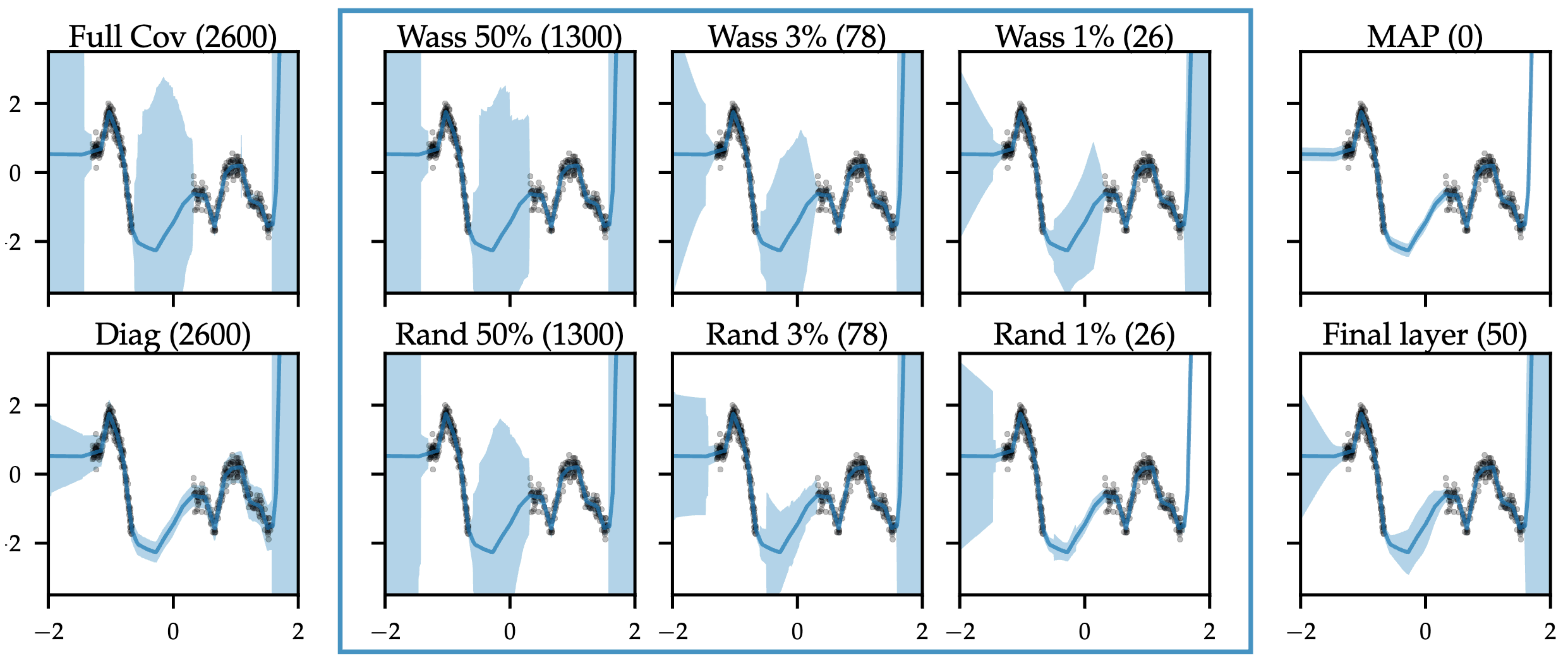
1. Find MAP estimate for all weights
2. Select subnetwork via heuristic
3. Construct the Laplace approximation, with a full covariance matrix, over the subnetwork
4. Compute predictive distribution as usual with the posterior from step #3.

Subnetwork Laplace Approximation

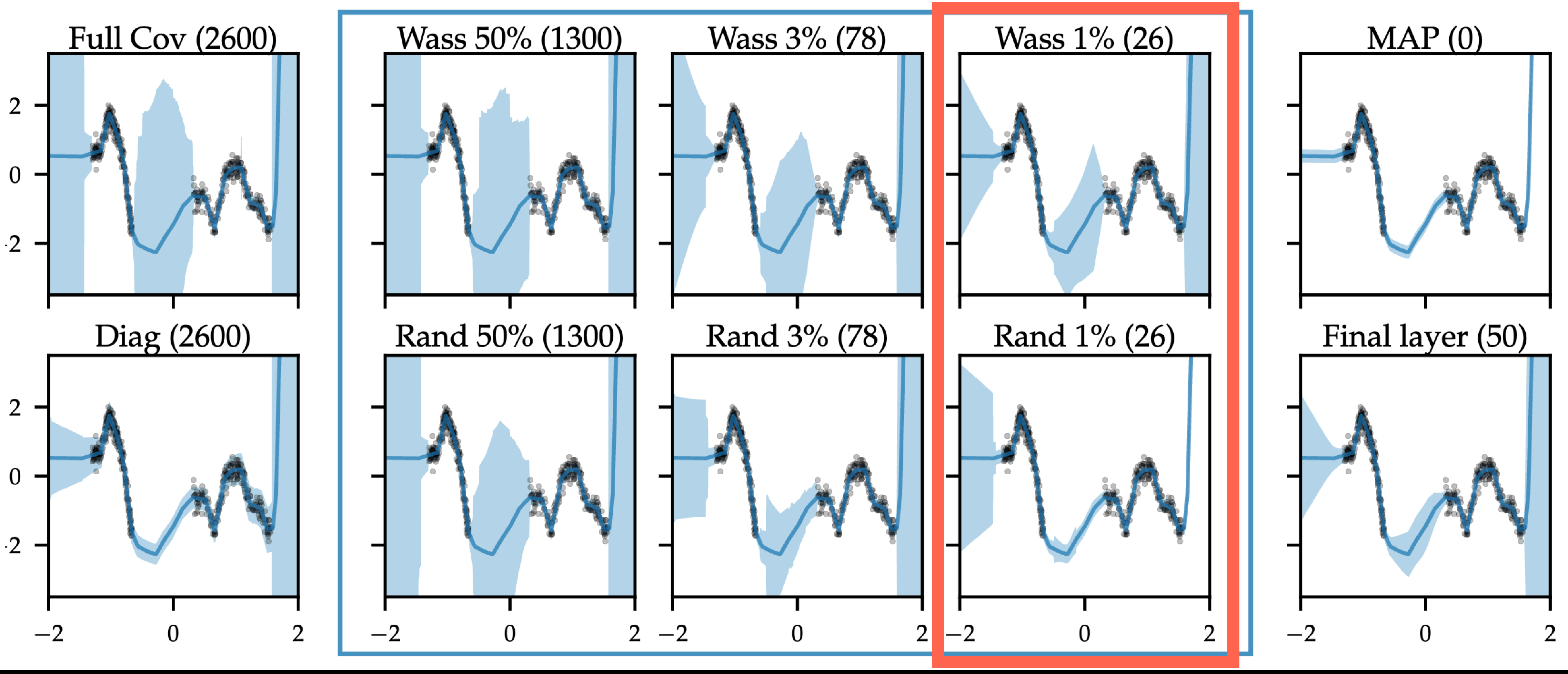
1. Find MAP estimate for all weights
2. Select subnetwork via heuristic
3. Construct the Laplace approximation, with a full covariance matrix, over the subnetwork
4. Compute predictive distribution as usual with the posterior from step #3.

Simulation Results

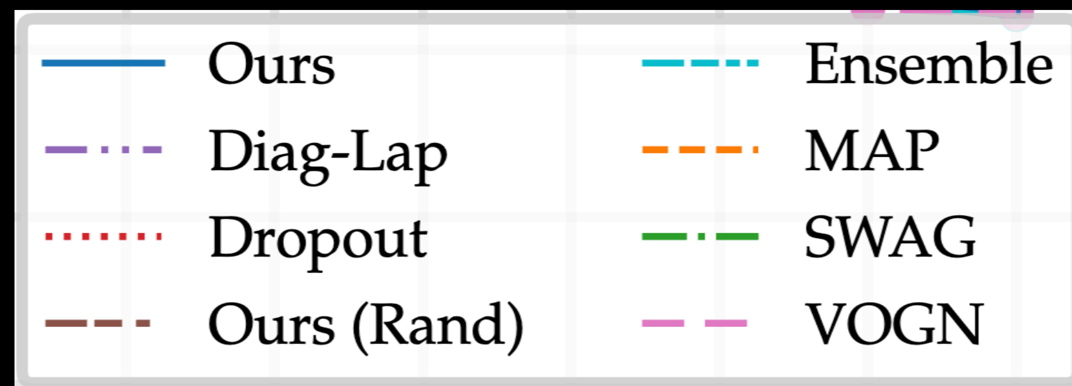
Simulation Results



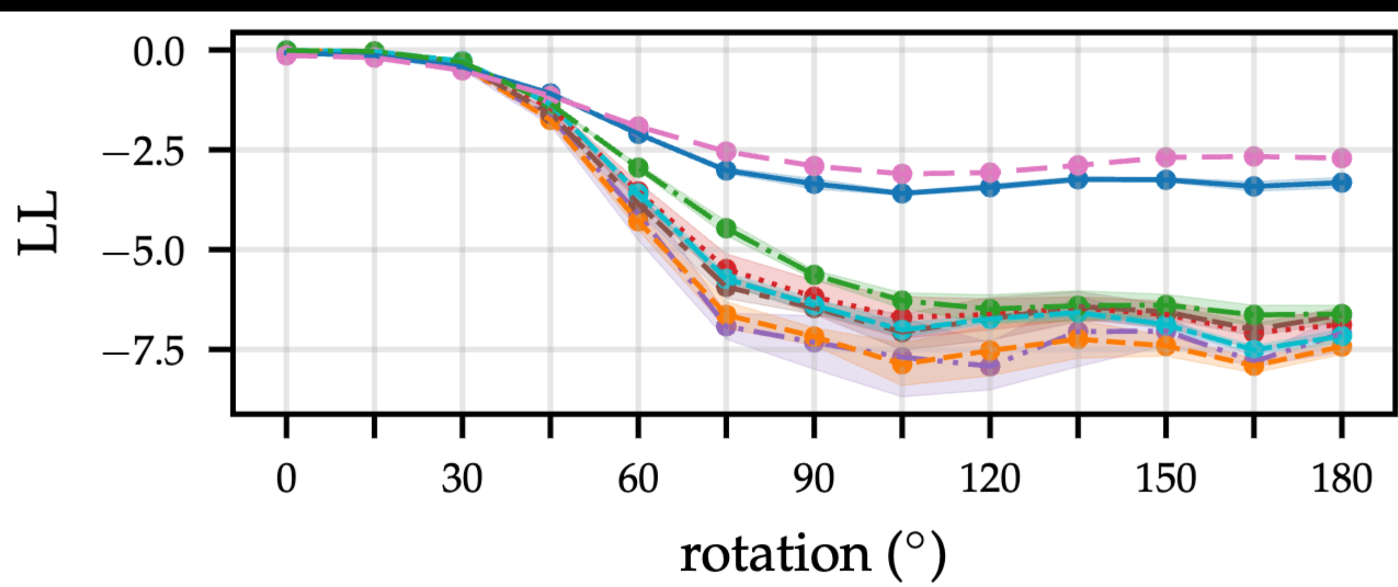
Simulation Results



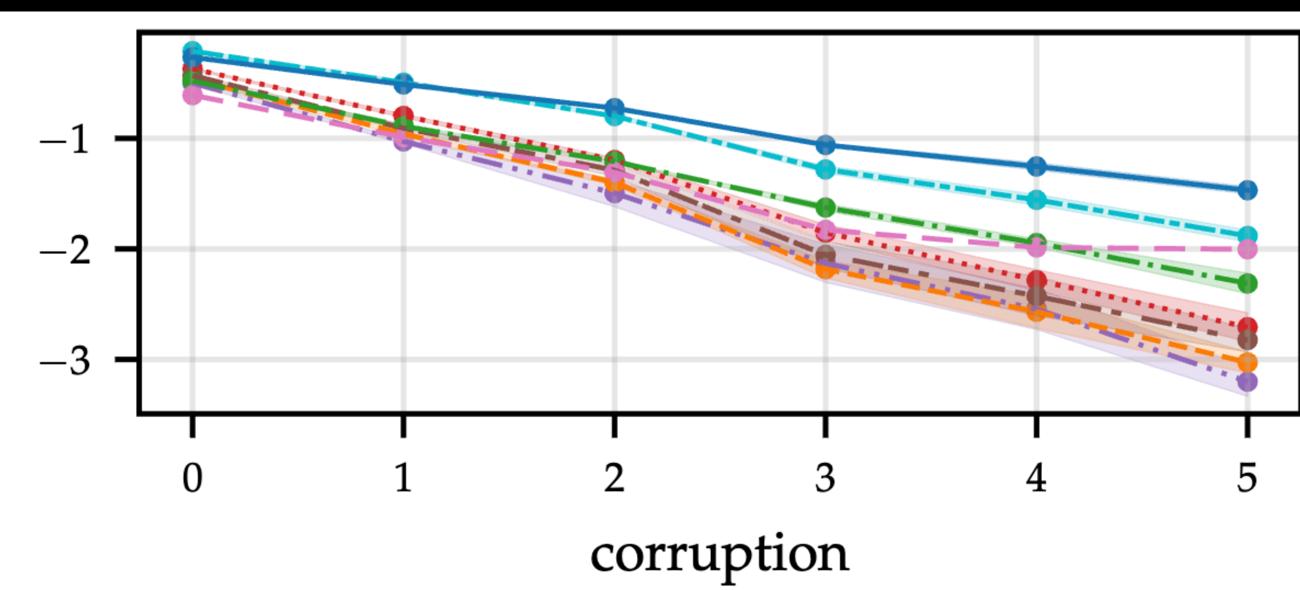
Robustness Results



Rotated MNIST



Corrupted CIFAR-10



Outline

- ⊗ background
- ⊗ subnetwork inference algorithm
- ⊗ subnetworks are all you need

Outline

- ⊗ background
- ⊗ subnetwork inference algorithm
- ⊗ subnetworks are all you need

Do Bayesian Neural Networks Need To Be Fully Stochastic?

Mrinank Sharma University of Oxford
Sebastian Farquhar University of Oxford
Eric Nalisnick University of Amsterdam
Tom Rainforth University of Oxford

Mrinank Sharma

$$\mathrm{p}\left(\theta_1, \ldots, \theta_\mathrm{L} \,|\, \mathfrak{D}\right) \stackrel{?}{=} \mathrm{p}\left(\left\{\theta_\mathrm{s} \,|\, \mathrm{s} \in \mathbb{S}\right\} \,|\, \mathfrak{D}\right) \cdot \prod_{r \in \mathbb{R}} \delta\left[\theta_r - \bar{\theta}_r\right]$$

Do Bayesian Neural Networks Need To Be Fully Stochastic?

Mrinank Sharma
University of Oxford

Sebastian Farquhar
University of Oxford

Eric Nalisnick
University of Amsterdam

Tom Rainforth
University of Oxford

Mrinank Sharma

$$p\left(\theta_1, \ldots, \theta_L \mid \mathfrak{D}\right) \stackrel{?}{=} p\left(\left\{\theta_s \mid s \in \mathbb{S}\right\} \mid \mathfrak{D}\right) \cdot \prod_{r \in \mathbb{R}} \delta\left[\theta_r - \bar{\theta}_r\right]$$

Do Bayesian Neural Networks Need To Be Fully Stochastic?

Mrinank Sharma University of Oxford

Sebastian Farquhar University of Oxford

Eric Nalisnick University of Amsterdam

Tom Rainforth University of Oxford

Mrinank Sharma

$$\mathrm{p}\left(\theta_1, \ldots, \theta_{\mathrm{L}} \mid \mathfrak{D}\right) \stackrel{?}{=} \mathrm{p}\left(\left\{\theta_{\mathrm{s}} \mid \mathrm{s} \in \mathbb{S}\right\} \mid \mathfrak{D}\right) \cdot \prod_{r \in \mathbb{R}} \delta\left[\theta_r - \bar{\theta}_r\right]$$

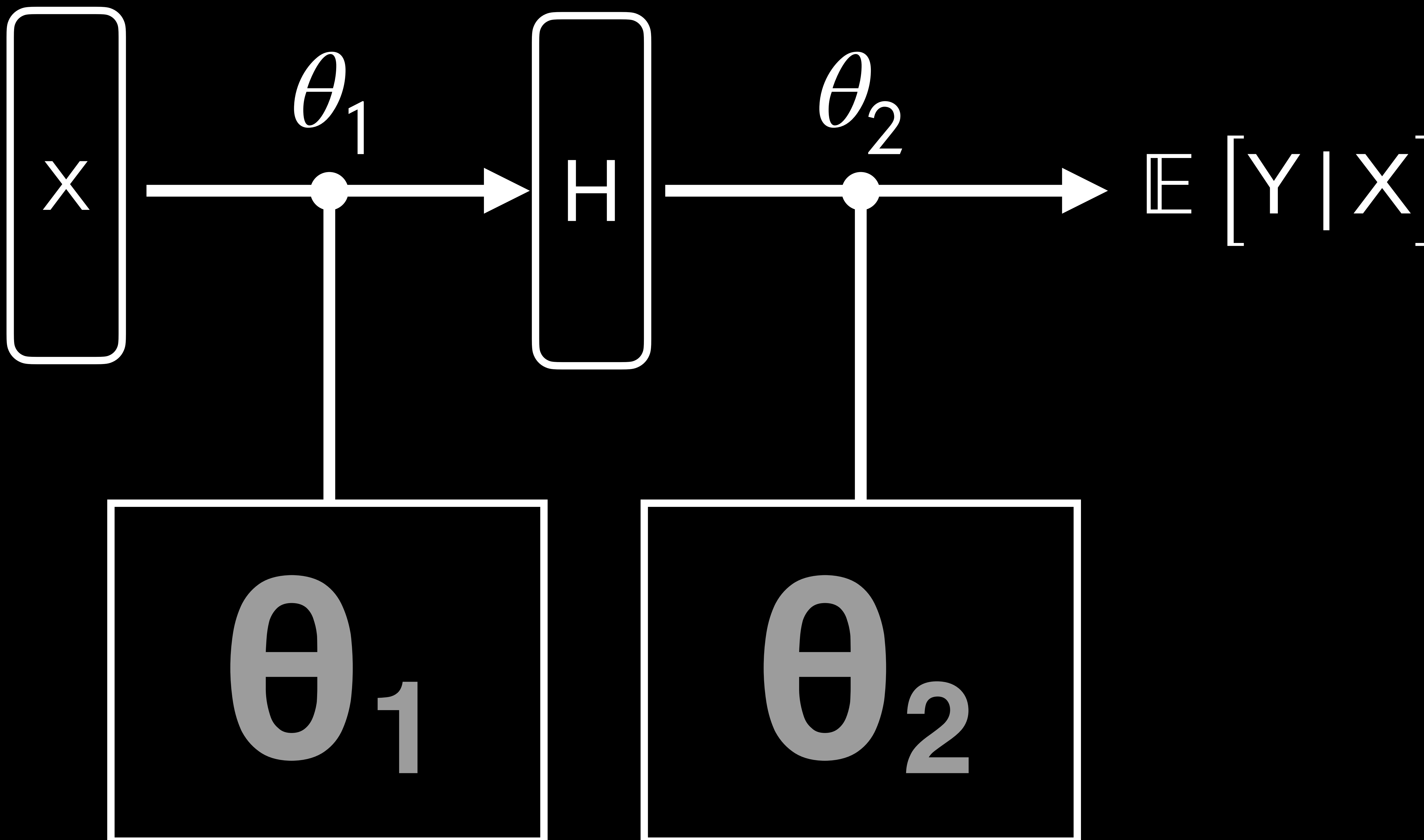
Theoretical Result

Informal: a multi-layer perceptron, with at least one hidden layer, can represent arbitrary predictive distributions, as long as there is at least one hidden layer between its stochastic variable(s) and output.

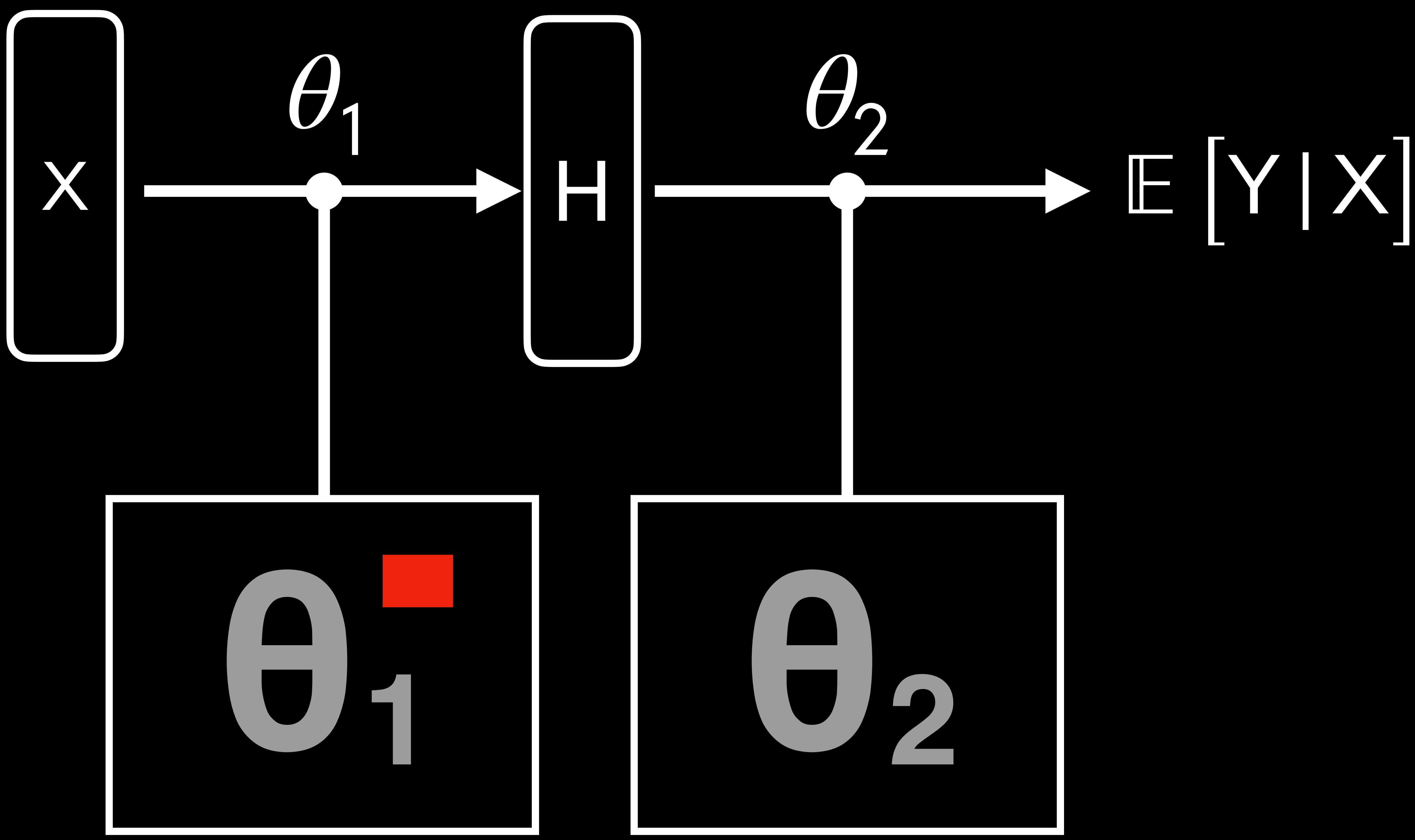
Theoretical Result

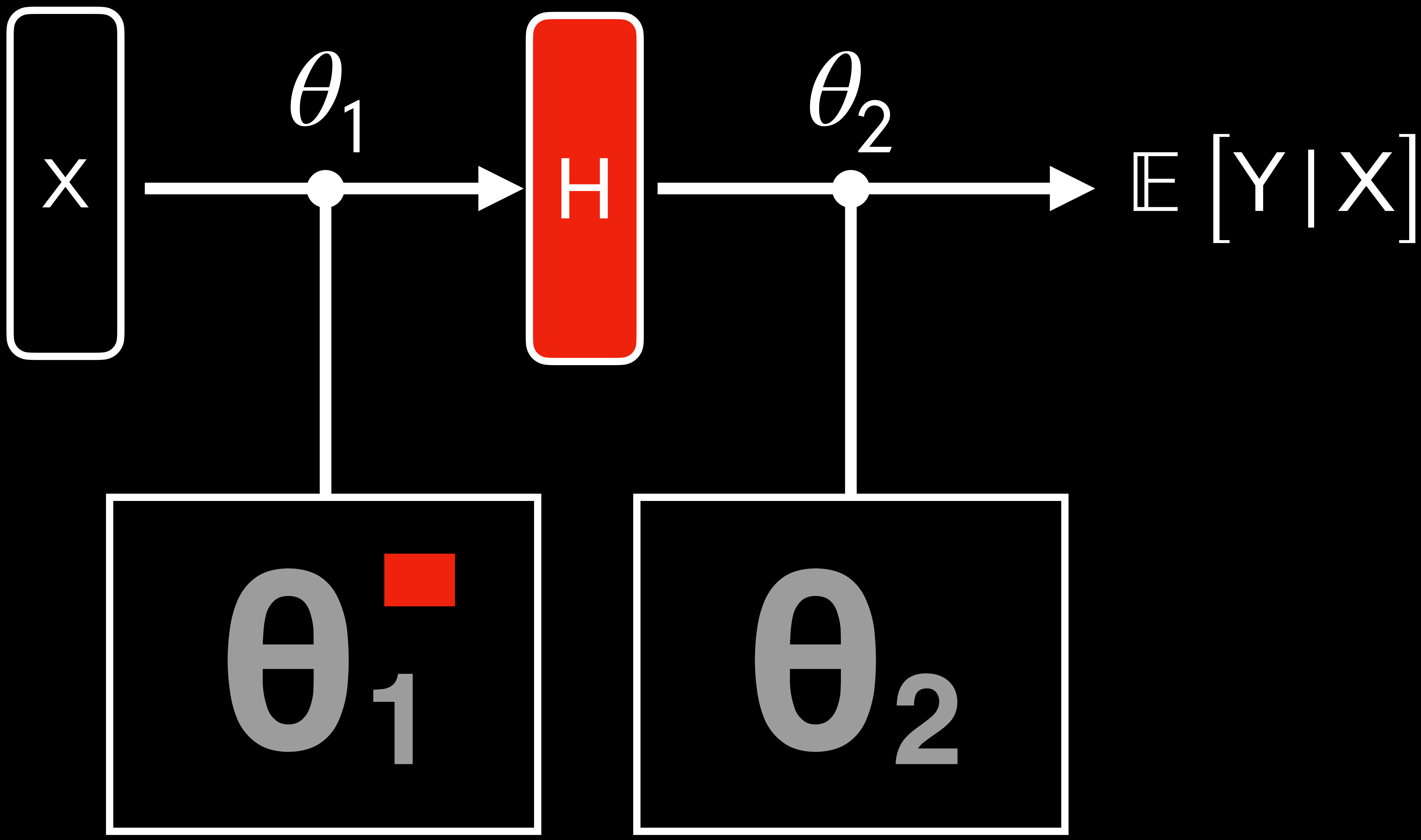
Informal: a multi-layer perceptron, with at least one hidden layer, can represent arbitrary predictive distributions, as long as there is at least one hidden layer between its stochastic variable(s) and output.

Proof sketch: Combine the *noise outsourcing lemma* [Austin, 2012] with the *universal approximation theorem* [Leshno, 1993]

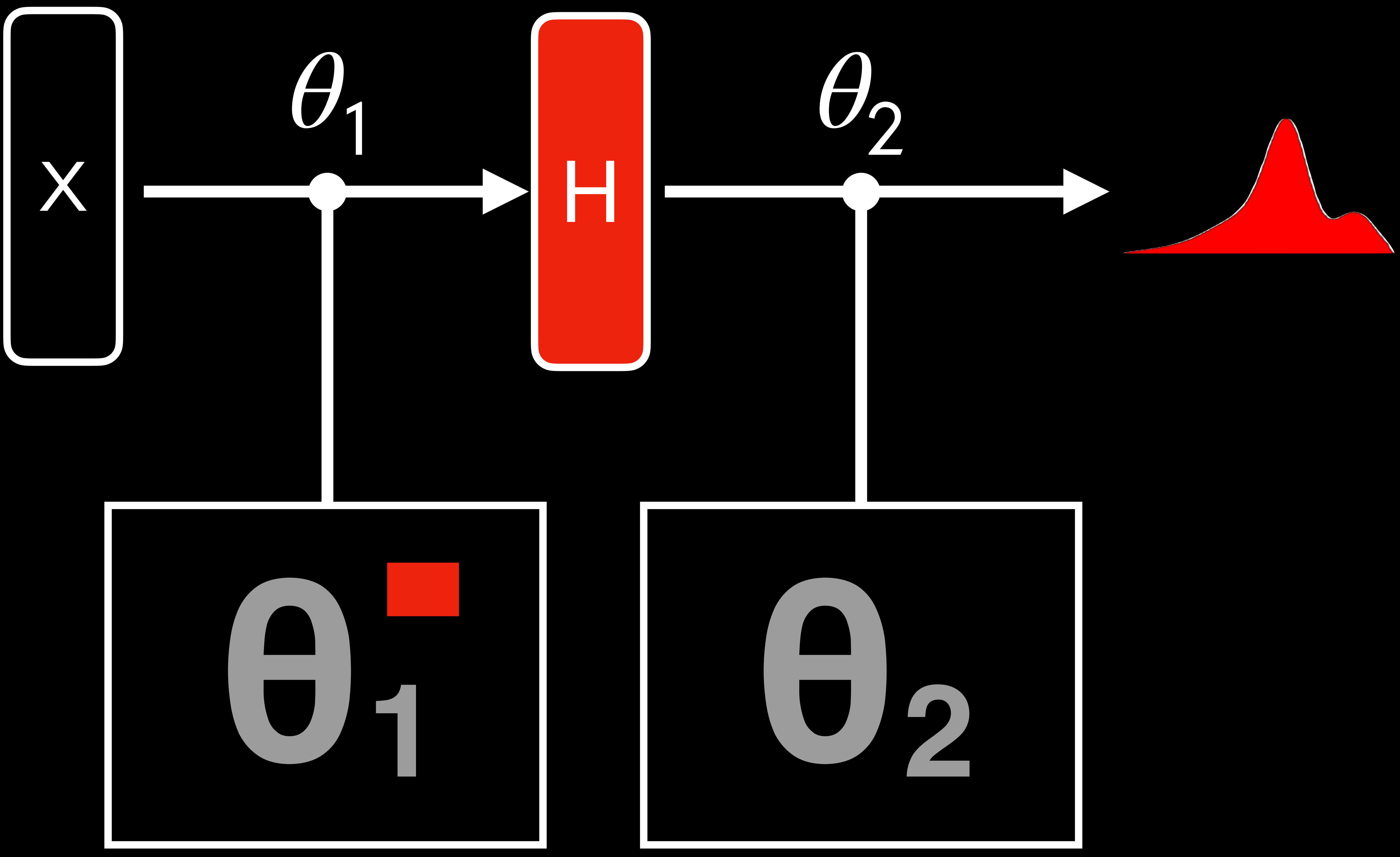


X
θ_1
H
θ_2
$\mathbb{E}[Y|X]$
θ_1
θ_2


X
θ_1
H
θ_2
$\mathbb{E}[Y|X]$
θ_1
θ_2



X
θ1
H
θ2
E [Y | X]
θ1
θ2



X
θ_1
H
θ_2
θ_1
θ_2

Consequence

Doing posterior inference for many / all of a BNN's parameters is overkill!

Unfortunately, the theory is too blunt to give any more advice about how many stochastic variables to use and where to place them (except for not in the last layer).

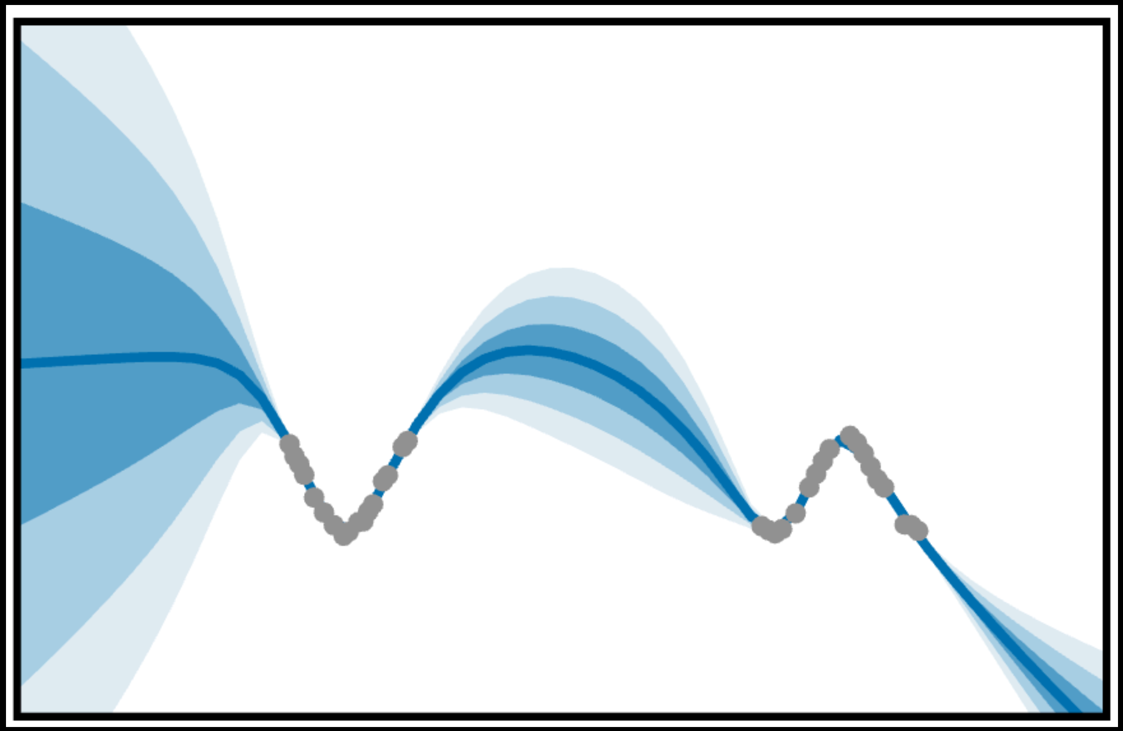
Experimental Results

Do we ever see a systematic benefit to having more stochastic variables?

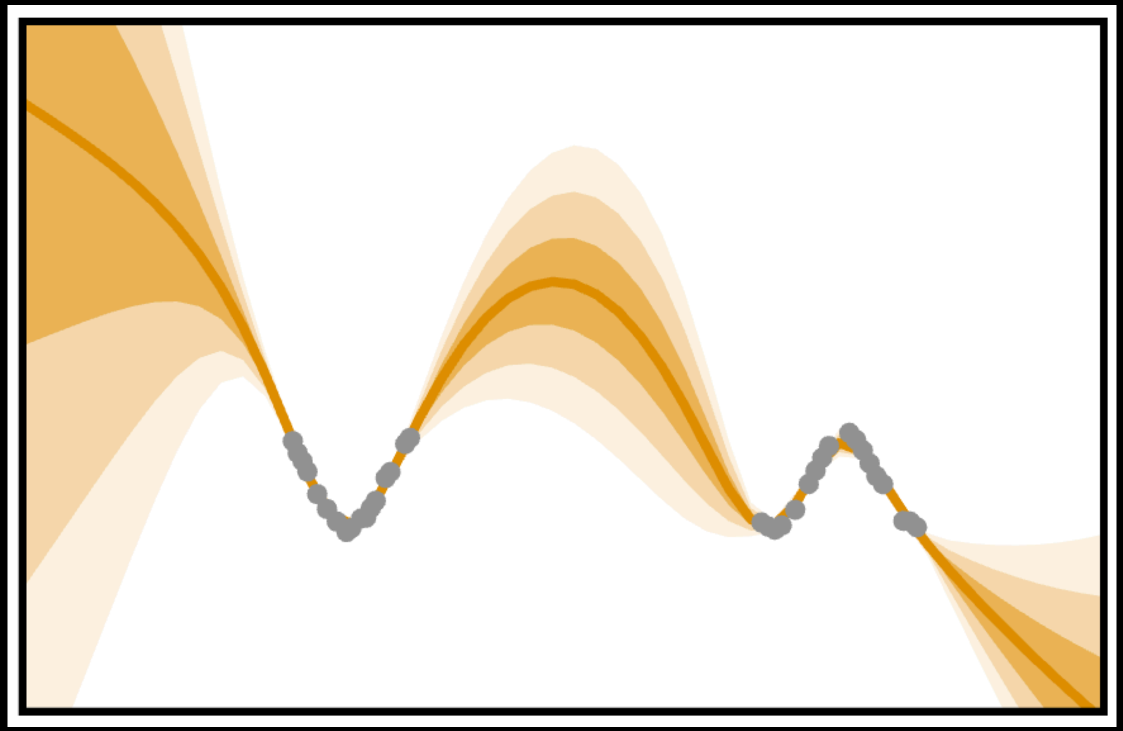
Experimental Results

Do we ever see a systematic benefit to having more stochastic variables?

Predictive Distributions



HMC on All Layers

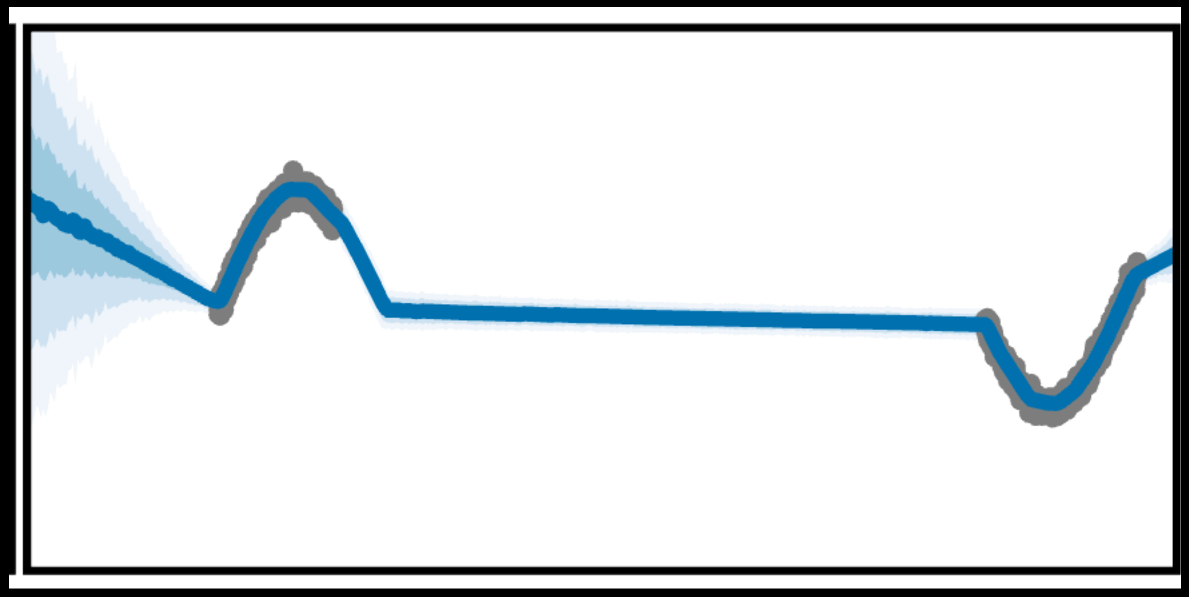


HMC on 1st Layer

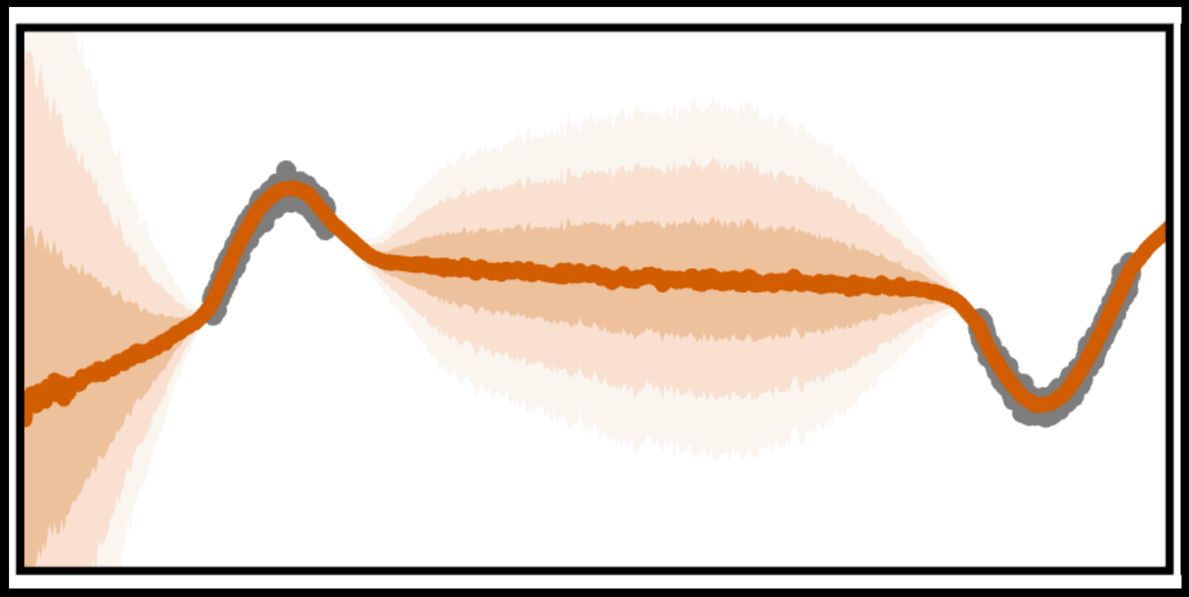
Experimental Results

Do we ever see a systematic benefit to having more stochastic variables?

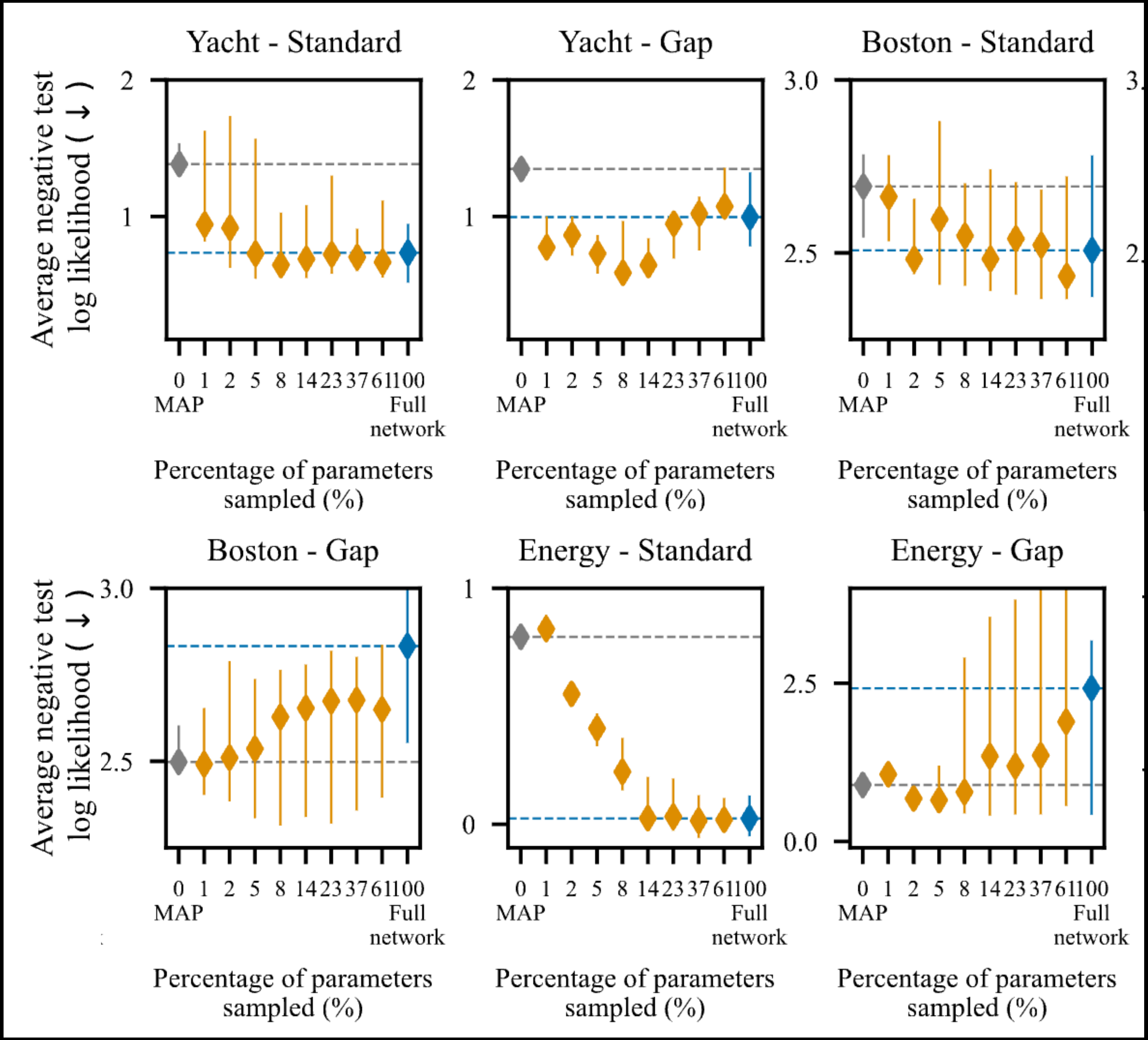
Predictive Distributions



MF-VI on All Layers



MF-VI on Last Layer


Yacht - Standard
Yacht - Gap
Boston - Standard
Boston - Gap
Energy - Standard
Energy - Gap
Average negative test log likelihood (↓)
Percentage of parameters sampled (%)
0 1 2 5 8 14 23 37 61 100
MAP
Full network
2
1
3.0
2.5
0
0.0

	CIFAR10		CIFAR100	
Model	Acc (%)	NLL	Acc (%)	NLL
Deterministic	95.61 ±0.01	0.187 ±0.001	79.33 ±0.45	0.862 ±0.014
Fully stochastic	94.69 ±0.07	0.214 ±0.002	77.68 ±0.29	0.944 ±0.002
Input layer stochastic	**95.70** ±0.08	0.187 ±0.002	**79.49** ±0.15	0.861 ±0.021
Output layer stochastic	95.60 ±0.05	0.189 ±0.001	78.92 ±0.34	0.933 ±0.010
Output layer and last block stochastic	95.59 ±0.08	**0.168** ±0.0005	79.00 ±0.091	**0.834** ±0.0007

Outline

- ⊗ background
- ⊗ subnetwork inference algorithm
- ⊗ subnetworks are all you need

Outline

- ⊗ background
- ⊗ subnetwork inference algorithm
- ⊗ subnetworks are all you need

Conclusions

1. Subnetwork inference is justified both experimentally and theoretically
2. Open problems:
   1. Better methods for choosing the subnetwork
   2. Principled, unified inference algorithms for models with stochastic and deterministic parameters