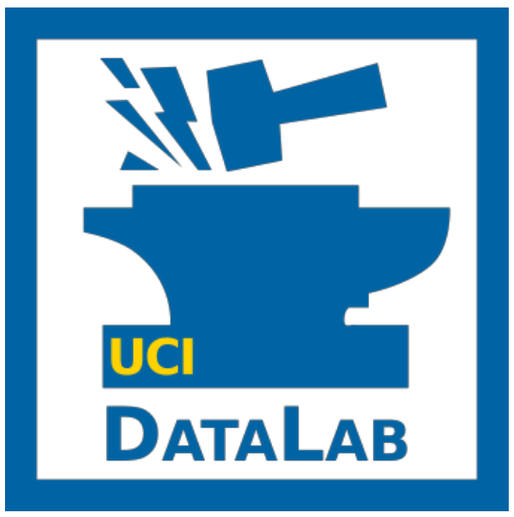
On Priors for Bayesian Neural Networks

Eric Nalisnick

PhD Defense
University of California, Irvine



First-Author Publications and Preprints

WORKING PAPERS / PREPRINTS

E. Nalisnick, D. Tran, D. Blei, and P. Smyth. ***Deep Learning: A Synthesis from Probabilistic Foundations.*** 2018.

E. Nalisnick and P. Smyth. ***Averaging and Combining Variational Models with Stein Particle Descent.*** 2018.

E. Nalisnick, A. Anandkumar, and P. Smyth. ***A Scale Mixture Perspective of Multiplicative Noise in Neural Networks.*** 2015 / 2018.

PUBLICATIONS DURING PHD

E. Nalisnick and P. Smyth. ***Learning Priors for Invariance.*** AISTATS 2018.

E. Nalisnick and P. Smyth. ***Learning Approximately Objective Priors.*** UAI 2017.

E. Nalisnick and P. Smyth. ***Stick-Breaking Variational Autoencoders.*** ICLR 2017.

E. Nalisnick, B. Mitra, N. Craswell, and R. Caruana. ***Improving Document Ranking with Dual Word Embeddings.*** WWW 2016.

PRE-UCI

E. Nalisnick and H. Baird. ***Character-to-Character Sentiment Analysis in Shakespeare's Plays.*** ACL 2013.

E. Nalisnick and H. Baird. ***Extracting Sentiment Networks from Shakespeare's Plays.*** ICDAR 2013.

First-Author Publications and Preprints

WORKING PAPERS / PREPRINTS

E. Nalisnick, D. Tran, D. Blei, and P. Smyth. ***Deep Learning: A Synthesis from Probabilistic Foundations.*** 2018.

E. Nalisnick and P. Smyth. ***Averaging and Combining Variational Models with Stein Particle Descent.*** 2018.

E. Nalisnick, A. Anandkumar, and P. Smyth. ***A Scale Mixture Perspective of Multiplicative Noise in Neural Networks.*** 2015 / 2018.

PUBLICATIONS DURING PHD

E. Nalisnick and P. Smyth. ***Learning Priors for Invariance.*** AISTATS 2018.

E. Nalisnick and P. Smyth. ***Learning Approximately Objective Priors.*** UAI 2017.

E. Nalisnick and P. Smyth. ***Stick-Breaking Variational Autoencoders.*** ICLR 2017.

E. Nalisnick, B. Mitra, N. Craswell, and R. Caruana. ***Improving Document Ranking with Dual Word Embeddings.*** WWW 2016.

PRE-UCI

E. Nalisnick and H. Baird. ***Character-to-Character Sentiment Analysis in Shakespeare's Plays.*** ACL 2013.

E. Nalisnick and H. Baird. ***Extracting Sentiment Networks from Shakespeare's Plays.*** ICDAR 2013.

First-Author Publications and Preprints

WORKING PAPERS / PREPRINTS

E. Nalisnick, D. Tran, D. Blei, and P. Smyth. ***Deep Learning: A Synthesis from Probabilistic Foundations.*** 2018.

E. Nalisnick and P. Smyth. ***Averaging and Combining Variational Models with Stein Particle Descent.*** 2018.

E. Nalisnick, A. Anandkumar, and P. Smyth. ***A Scale Mixture Perspective of Multiplicative Noise in Neural Networks.*** 2015 / 2018.

PUBLICATIONS DURING PHD

E. Nalisnick and P. Smyth. ***Learning Priors for Invariance.*** AISTATS 2018.

E. Nalisnick and P. Smyth. ***Learning Approximately Objective Priors.*** UAI 2017.

E. Nalisnick and P. Smyth. ***Stick-Breaking Variational Autoencoders.*** ICLR 2017.

E. Nalisnick, B. Mitra, N. Craswell, and R. Caruana. ***Improving Document Ranking with Dual Word Embeddings.*** WWW 2016.

PRE-UCI

E. Nalisnick and H. Baird. ***Character-to-Character Sentiment Analysis in Shakespeare's Plays.*** ACL 2013.

E. Nalisnick and H. Baird. ***Extracting Sentiment Networks from Shakespeare's Plays.*** ICDAR 2013.

Outline

1 **Motivation**

2 **Bayesian Neural Networks and Their Priors**

3 **Approximating Objective Priors**

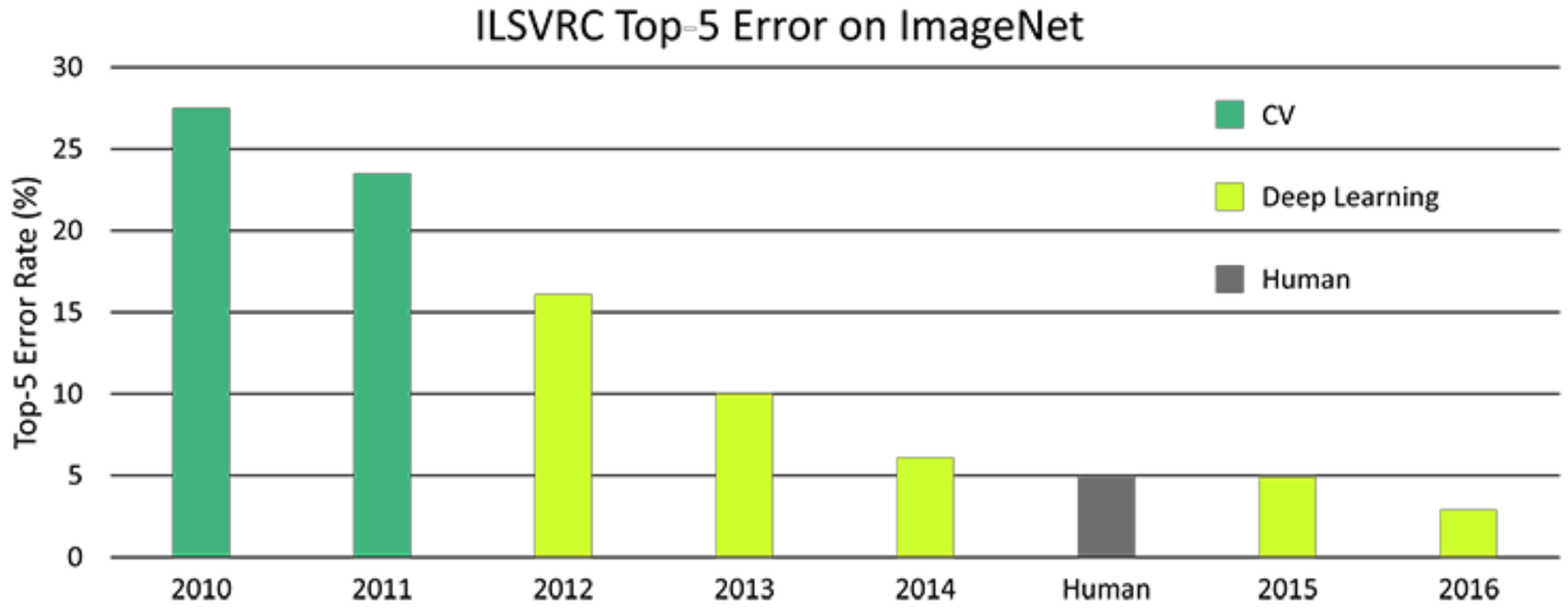
4 **Nonparametric Density Networks**

5 **Future Directions and Conclusions**

Motivation for Bayesian Deep Learning

Deep Learning Results

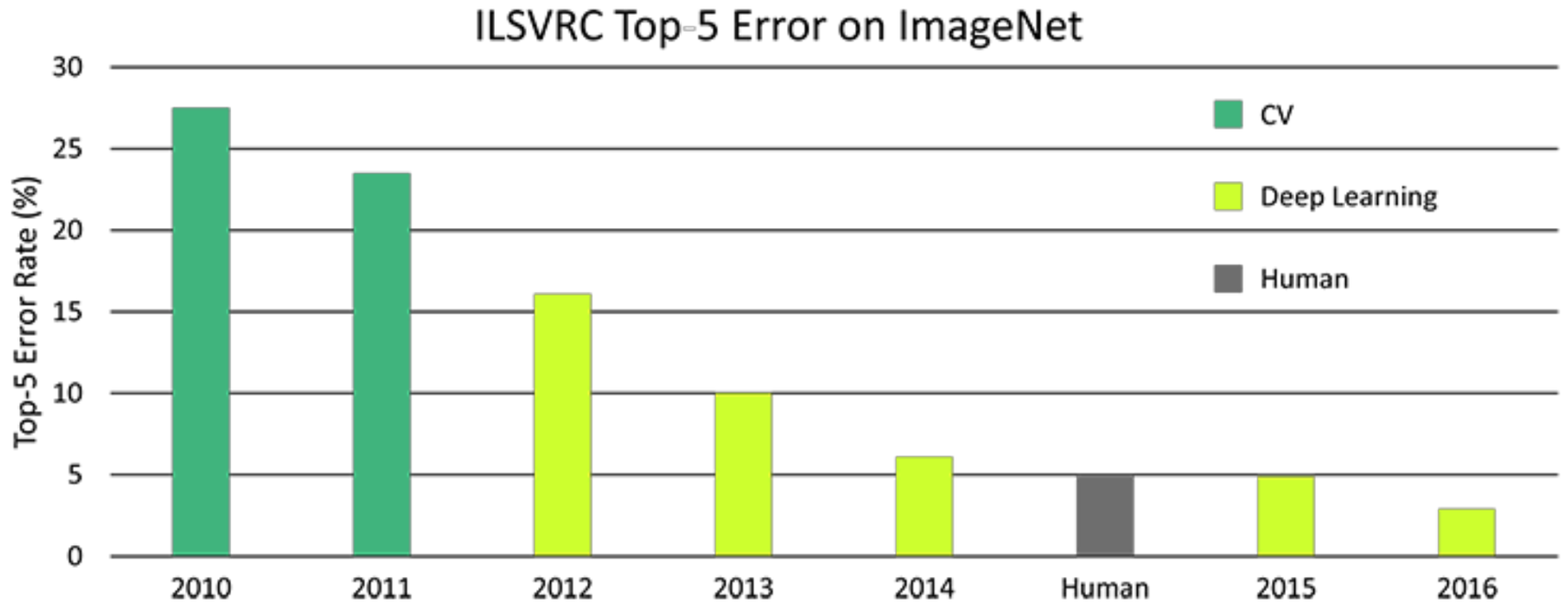
Computer Vision: Results on ImageNet object classification dataset.



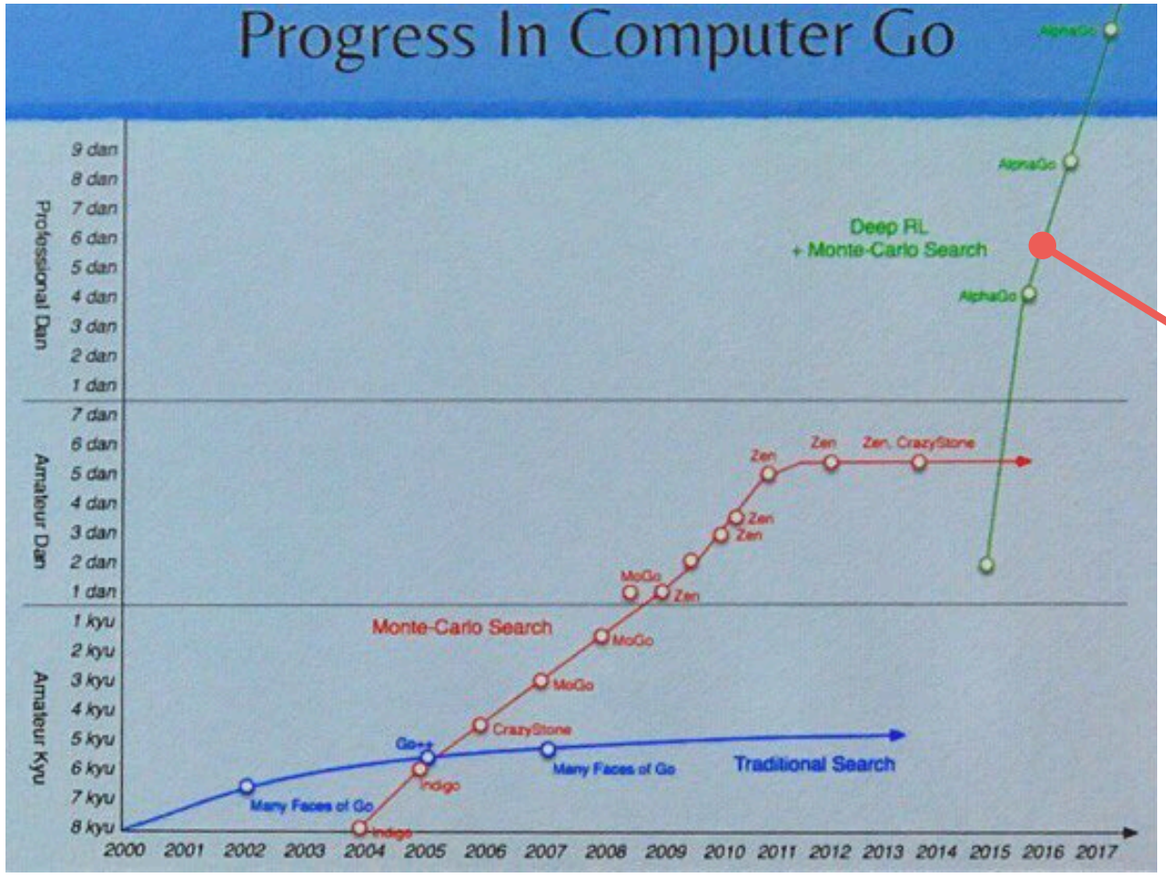
https://www.dsiac.org/sites/default/files/journals/as-figure-1.png

Deep Learning Results

Computer Vision: Results on ImageNet object classification dataset.



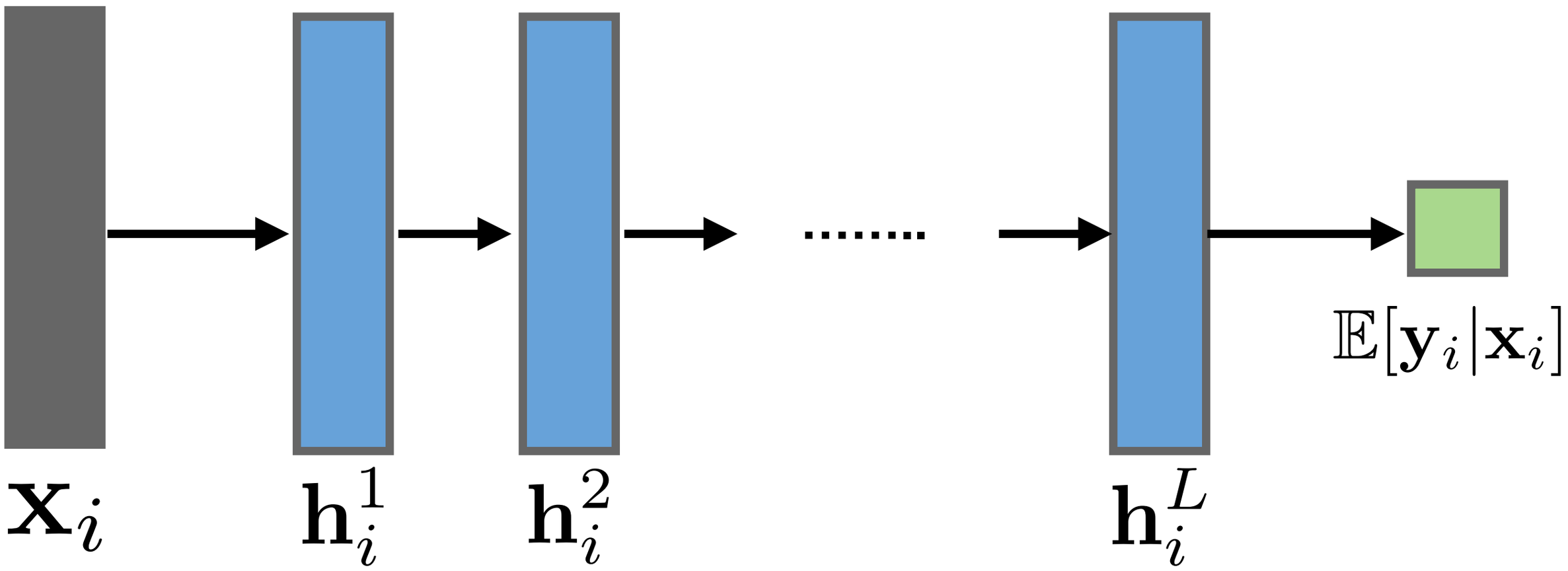
https://www.dsiac.org/sites/default/files/journals/as-figure-1.png



Reinforcement Learning: Results in playing Go.

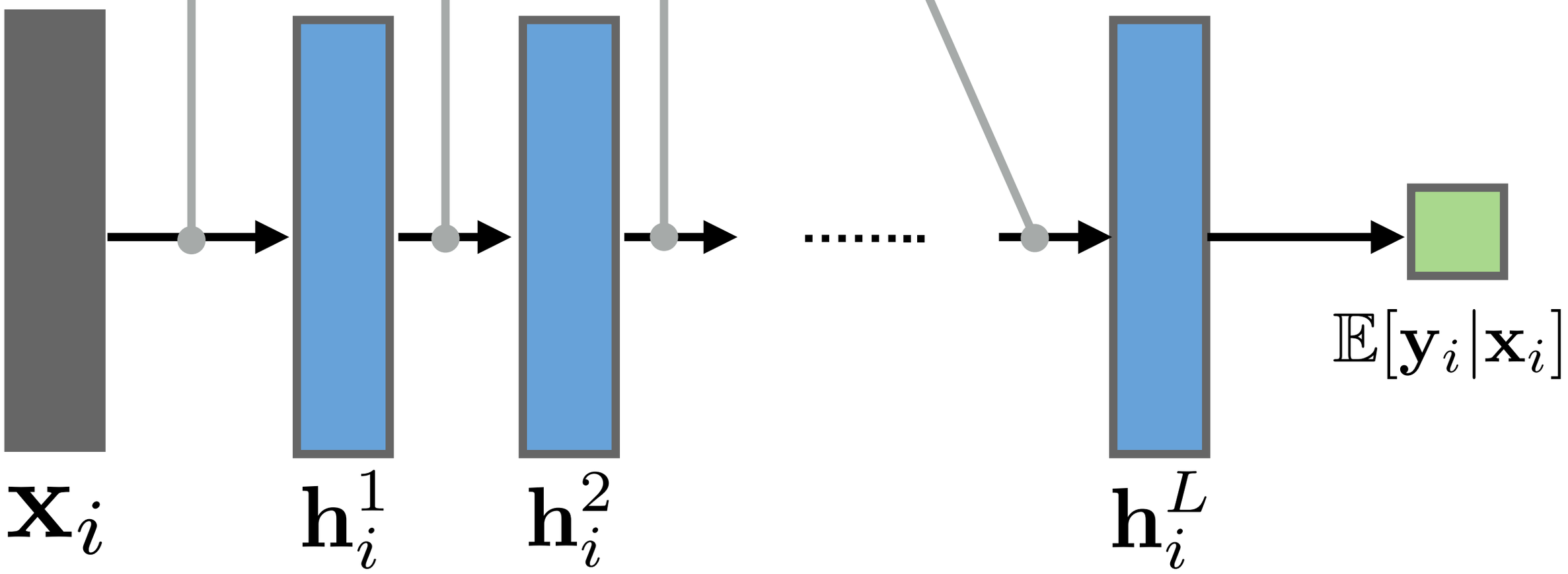
DEEP NEURAL NETWORKS

Neural Networks



Neural Networks

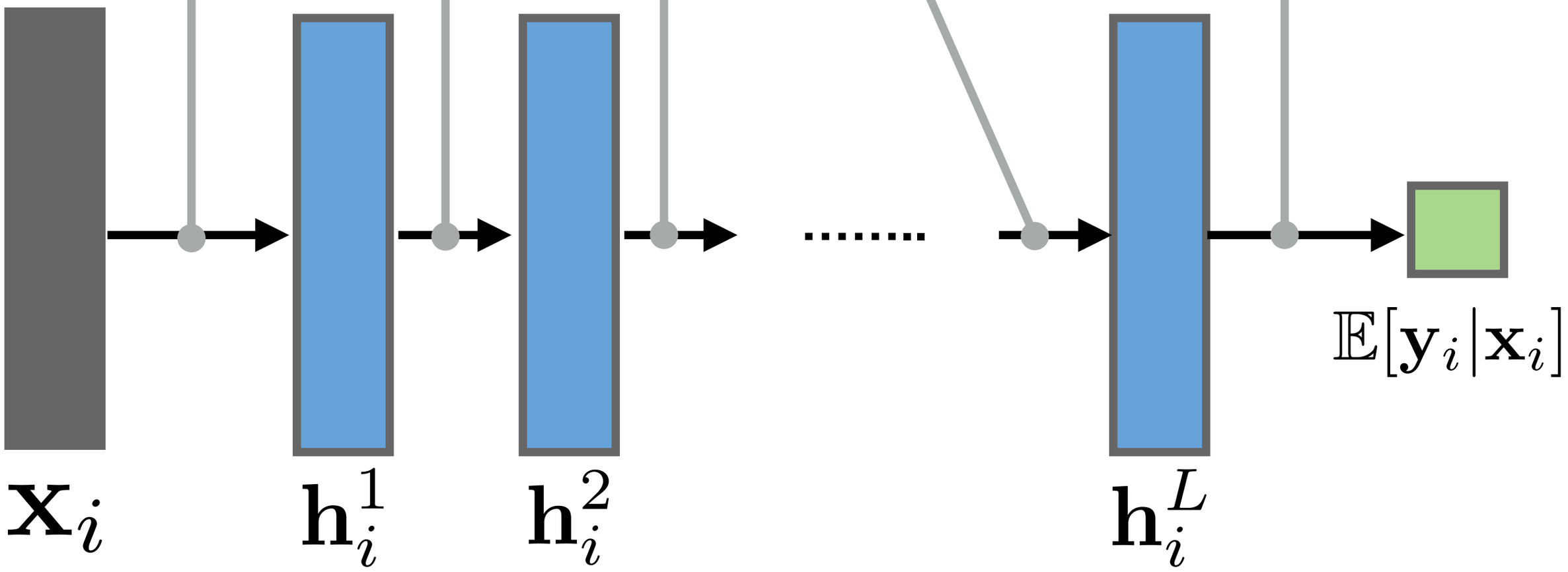
$$\mathbf{h}_i^l = f_l(\mathbf{h}_i^{l-1}\mathbf{W}_l + \mathbf{b}_l), \quad \mathbf{h}_i^0 = \mathbf{x}_i$$



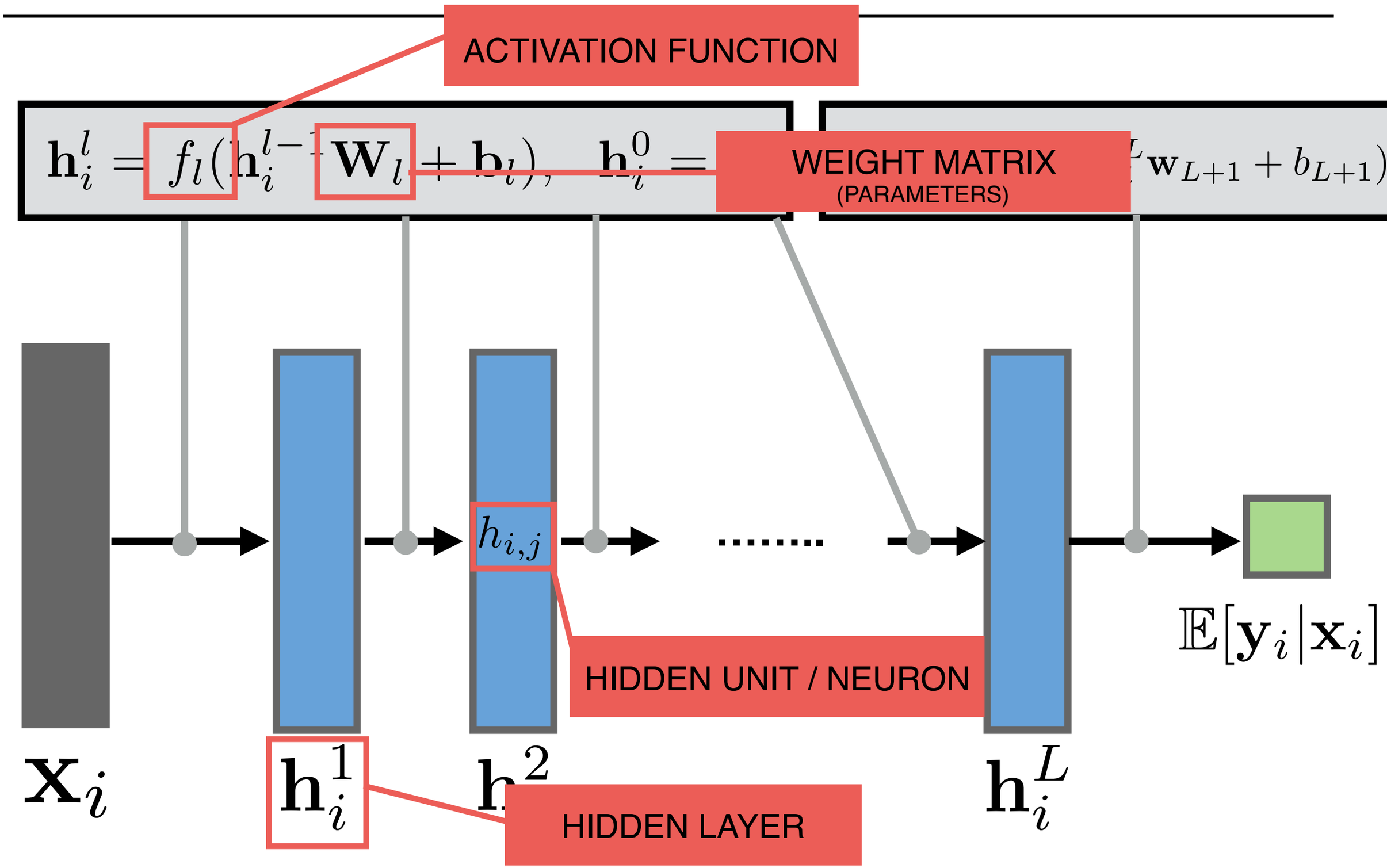
Neural Networks

$$\mathbf{h}_i^l = f_l(\mathbf{h}_i^{l-1}\mathbf{W}_l + \mathbf{b}_l), \quad \mathbf{h}_i^0 = \mathbf{x}_i$$

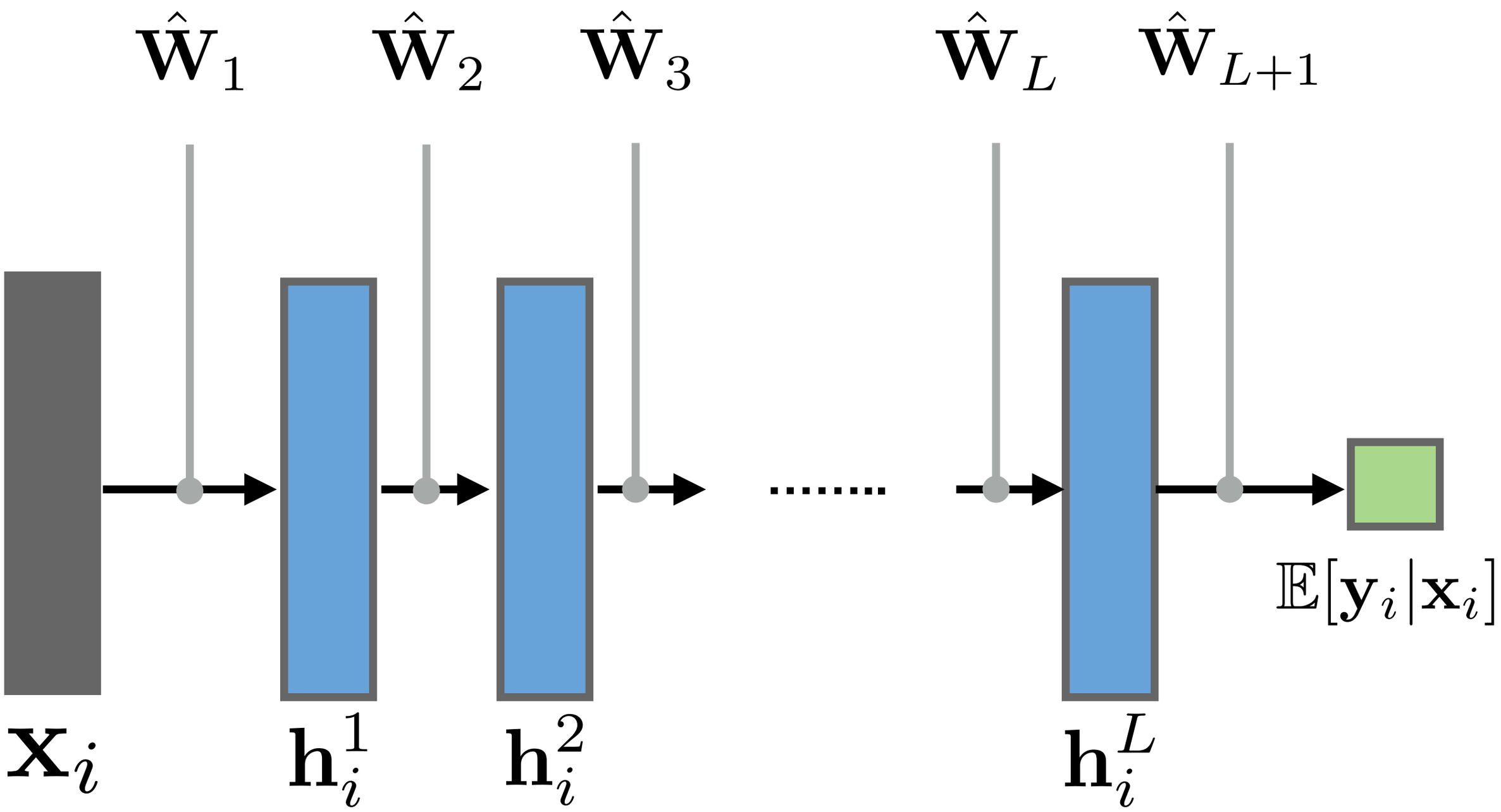
$$\mathbb{E}[y_i|\mathbf{x}_i] = g^{-1}(\mathbf{h}_i^L \mathbf{w}_{L+1} + b_{L+1})$$



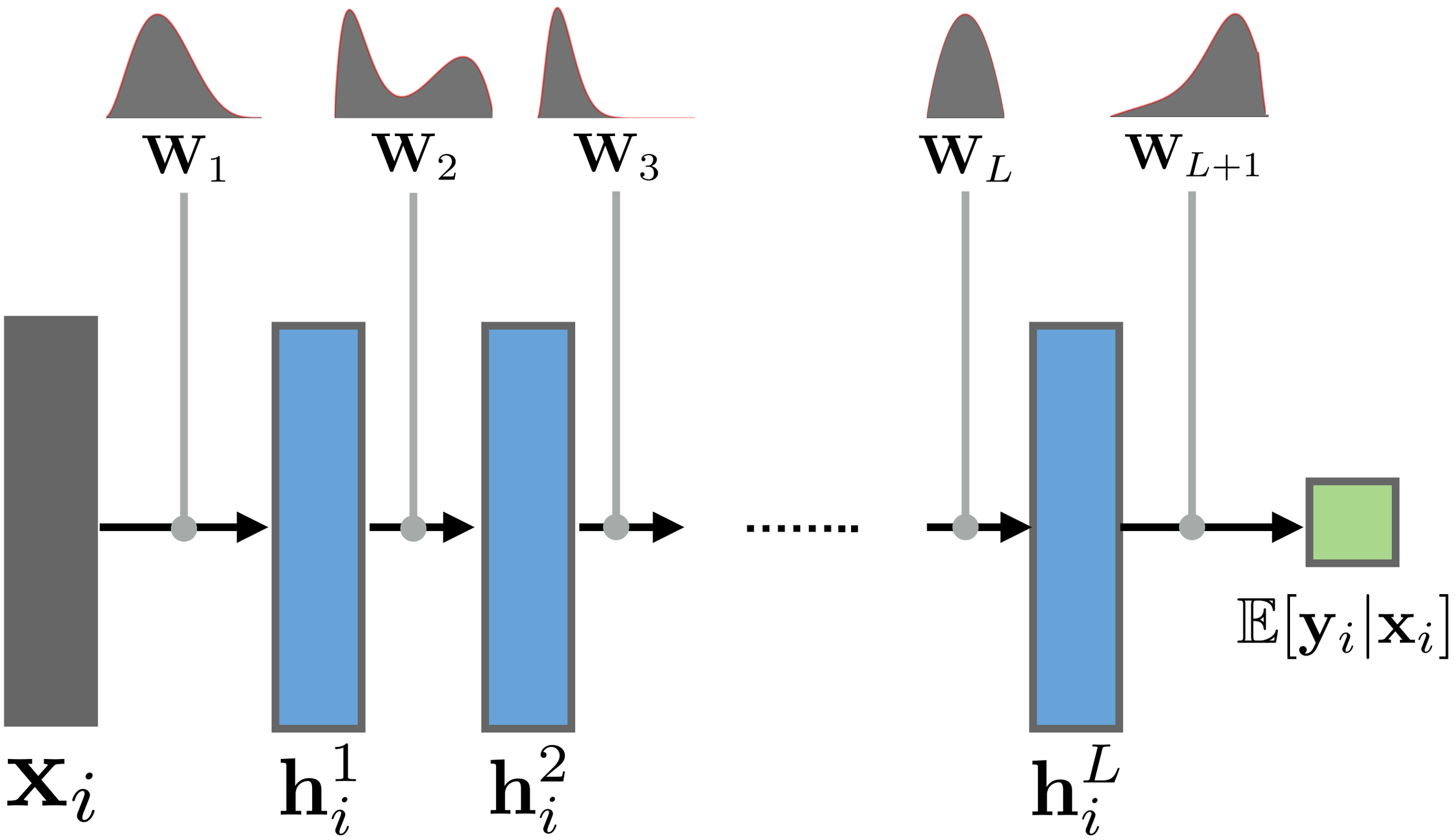
Neural Networks



Neural Networks



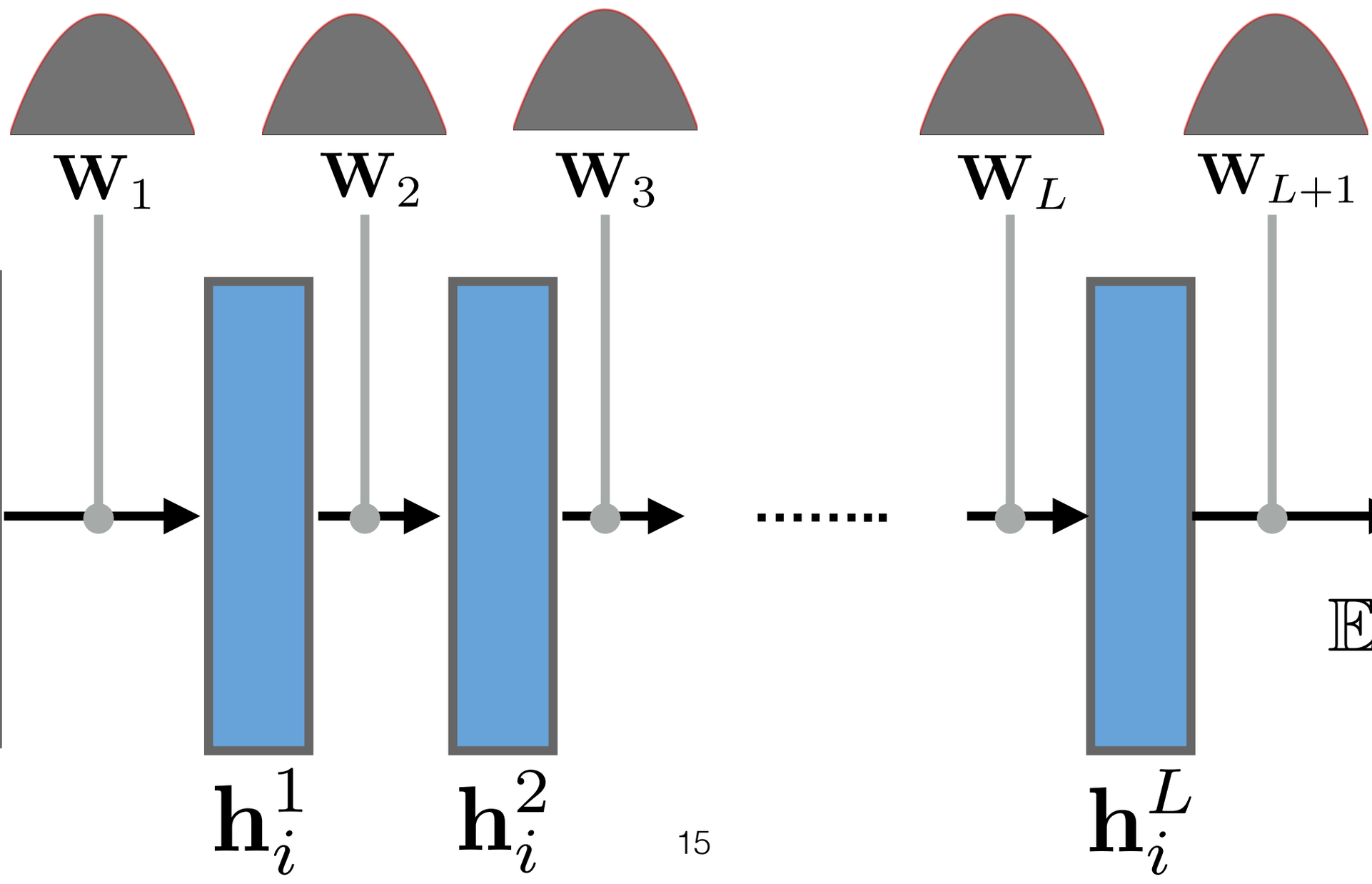
Neural Networks



Bayesian Neural Networks

STEP #1: $p(\{\mathbf{W}_l\}_{l=1}^{L+1}) = \prod_{l=1}^{L+1} p(\mathbf{W}_l)$

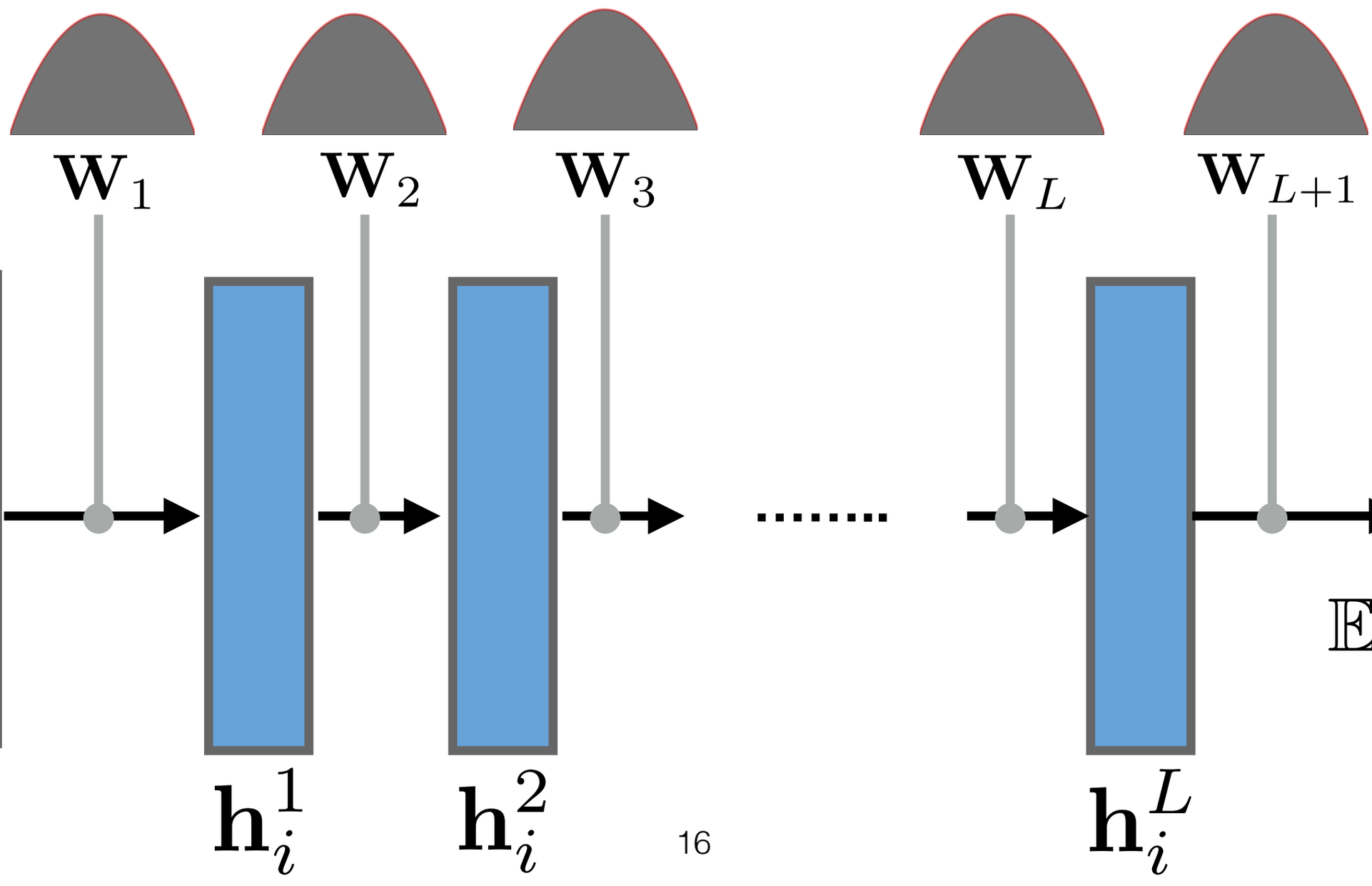
PRIOR



Bayesian Neural Networks

STEP #2: $p(\mathbf{y}|\mathbf{X}, \{\mathbf{W}_l\}_{l=1}^{L+1}) \prod_{l=1}^{L+1} p(\mathbf{W}_l)$

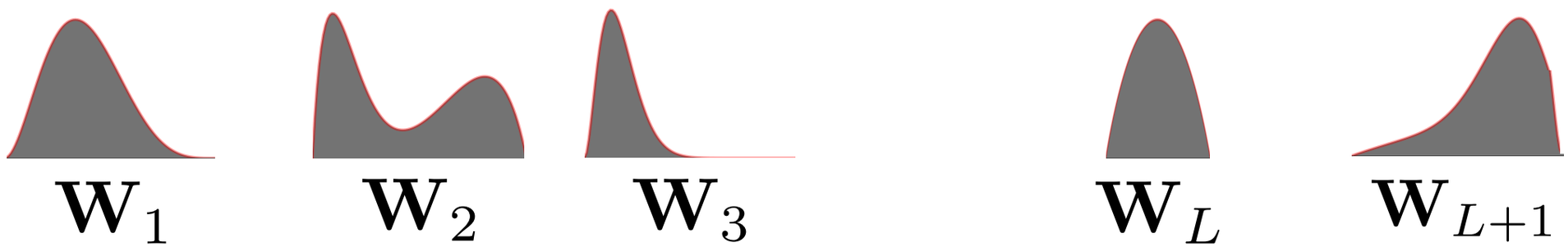
LIKELIHOOD

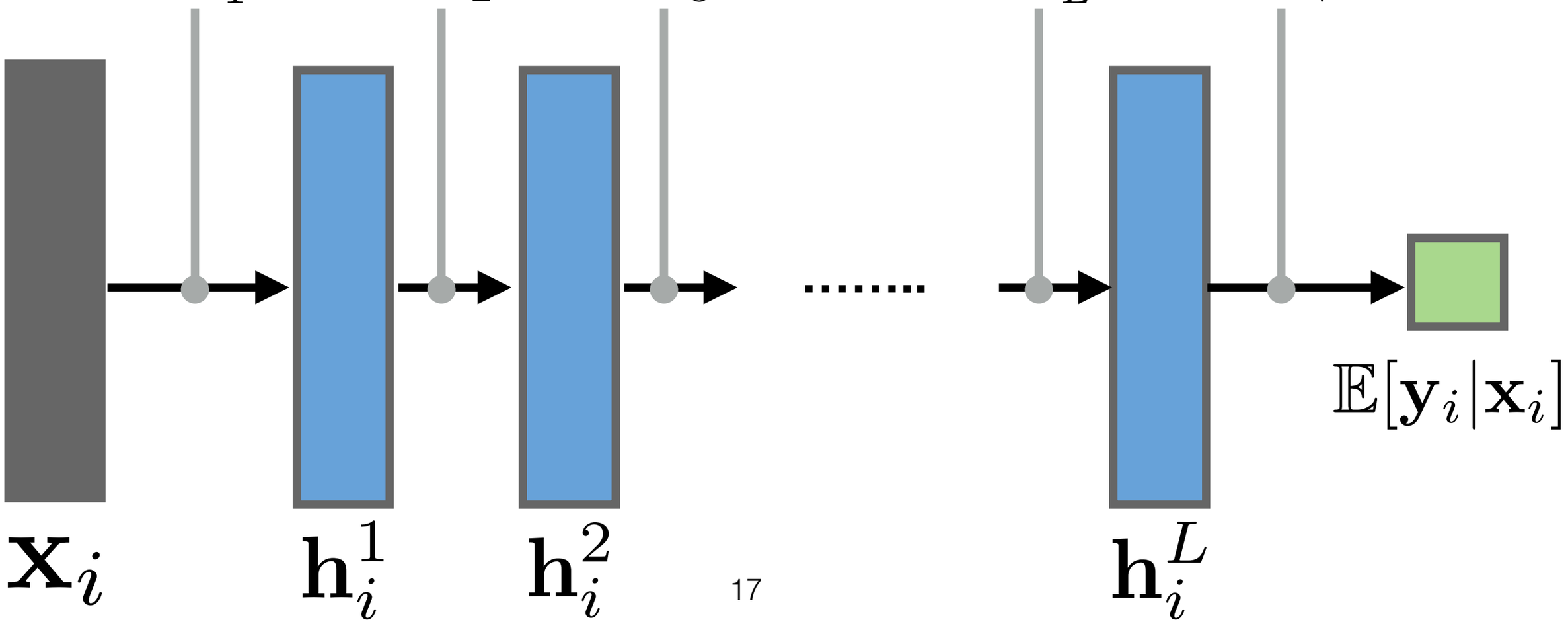


Bayesian Neural Networks

STEP #3: $p(\{\mathbf{W}_l\}_{l=1}^{L+1}|\mathbf{y},\mathbf{X})$

POSTERIOR

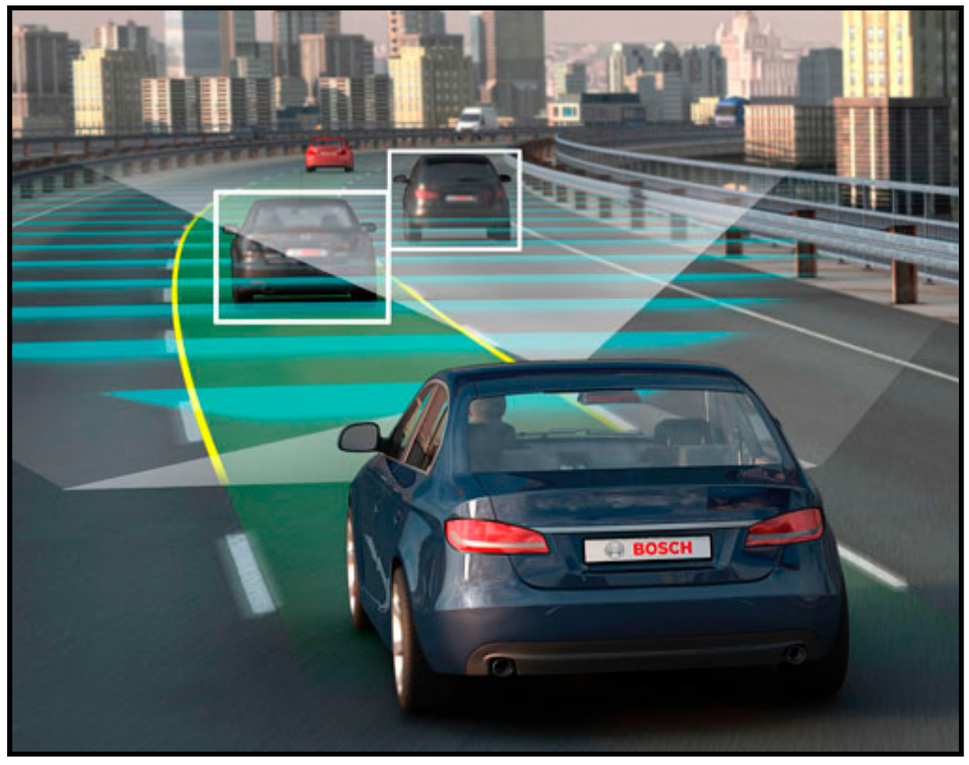




Benefits of Bayesian Neural Networks

Benefits of Bayesian Neural Networks

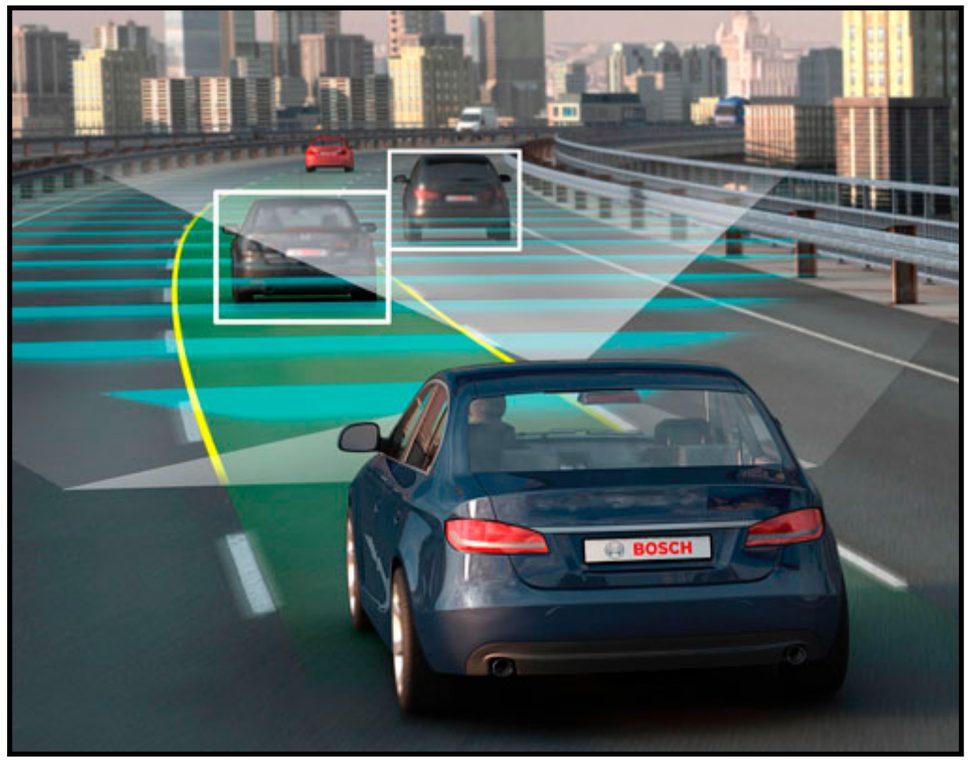
Model Uncertainty for Safe Automation.



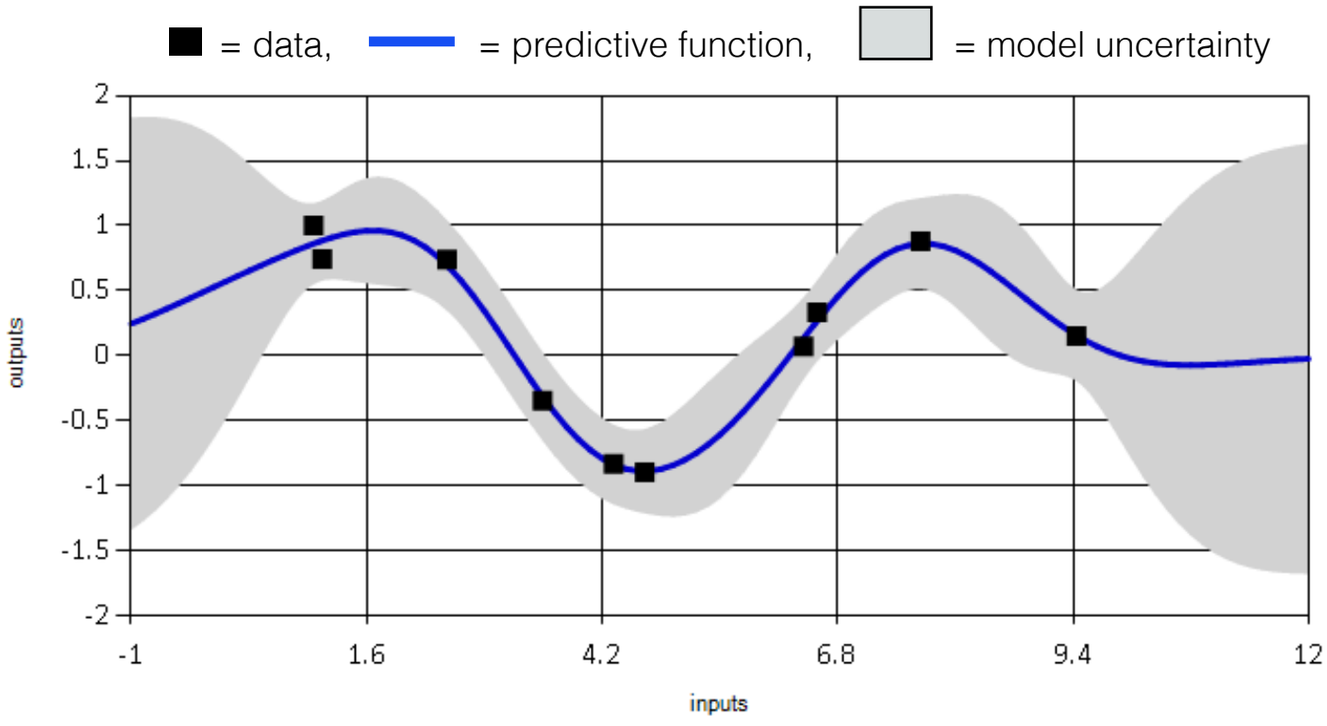
https://www.motorauthority.com/news/1081957_bosch-gives-update-on-its-autonomous-car-development-video

Benefits of Bayesian Neural Networks

Model Uncertainty for Safe Automation.



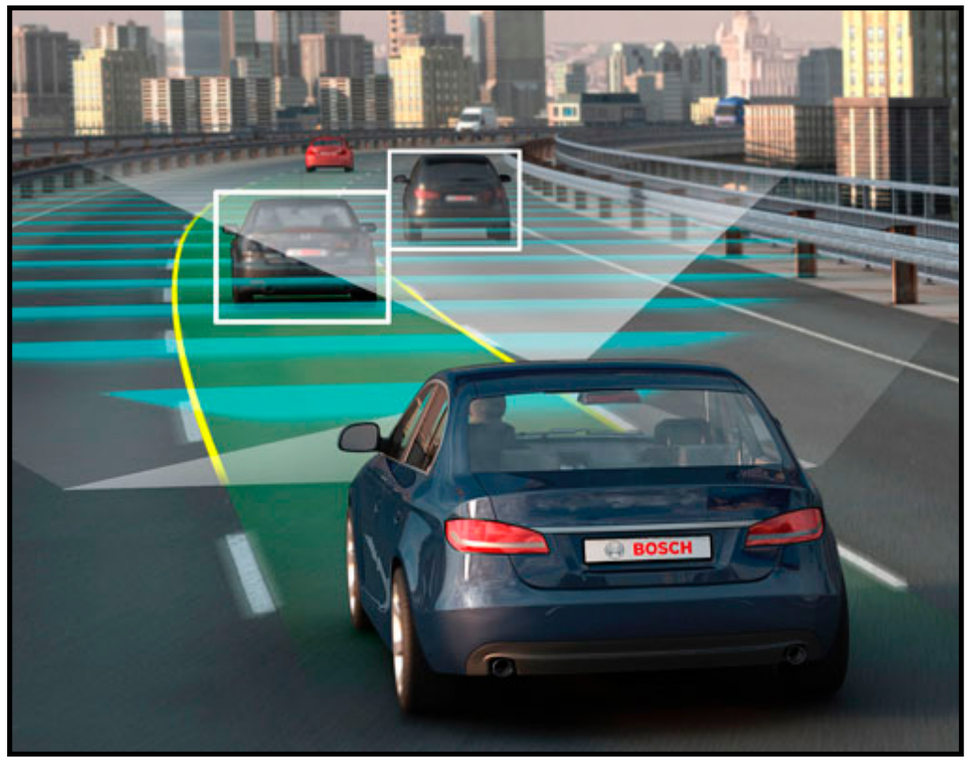
https://www.motorauthority.com/news/1081957_bosch-gives-update-on-its-autonomous-car-development-video



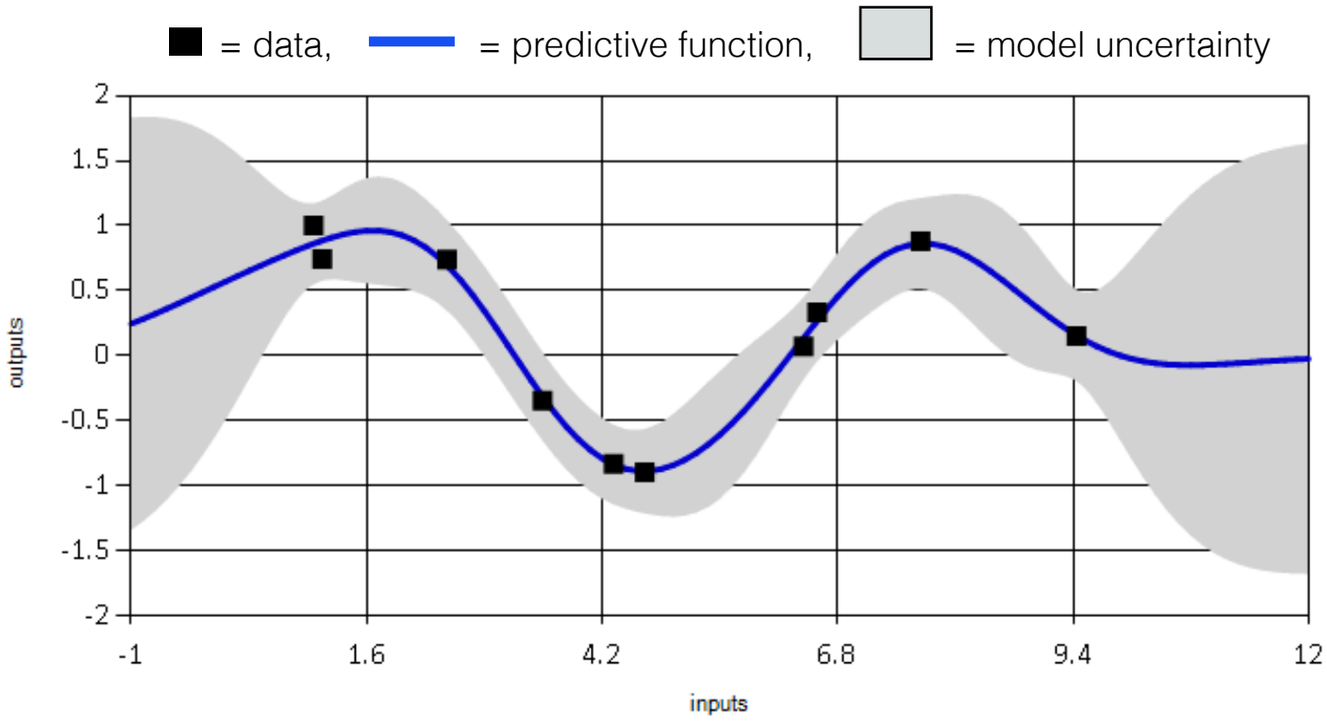
Posterior Predictive Distribution

Benefits of Bayesian Neural Networks

Model Uncertainty for Safe Automation.



https://www.motorauthority.com/news/1081957_bosch-gives-update-on-its-autonomous-car-development-video



Posterior Predictive Distribution

$$p(y^*|\mathbf{x}^*, \mathbf{y}, \mathbf{X}) = \int_{\mathbf{W}} p(y^*|\mathbf{x}^*, \mathbf{W}) p(\mathbf{W}|\mathbf{y}, \mathbf{X}) \, d\mathbf{W}$$

LIKELIHOOD OF NEW DATA

POSTERIOR FOR TRAINING DATA

Benefits of Bayesian Neural Networks

Coherent Framework for Online / Sequential Learning.

Benefits of Bayesian Neural Networks

Coherent Framework for Online / Sequential Learning.

BAYESIAN UPDATING

$$p\left(\mathbf{W}|\{\mathbf{y}_t, \mathbf{y}_{t+1}\}, \{\mathbf{X}_t, \mathbf{X}_{t+1}\}\right)$$

$$\propto p(\mathbf{y}_{t+1}|\mathbf{X}_{t+1}, \mathbf{W}) p(\mathbf{W}|\mathbf{y}_t, \mathbf{X}_t)$$

LIKELIHOOD OF NEW DATA

POSTERIOR
SERVES AS PRIOR

Benefits of Bayesian Neural Networks

Coherent Framework for Online / Sequential Learning.

BAYESIAN UPDATING

$$p\left(\mathbf{W}|\{\mathbf{y}_t, \mathbf{y}_{t+1}\}, \{\mathbf{X}_t, \mathbf{X}_{t+1}\}\right)$$

$$\propto p(\mathbf{y}_{t+1}|\mathbf{X}_{t+1}, \mathbf{W}) p(\mathbf{W}|\mathbf{y}_t, \mathbf{X}_t)$$

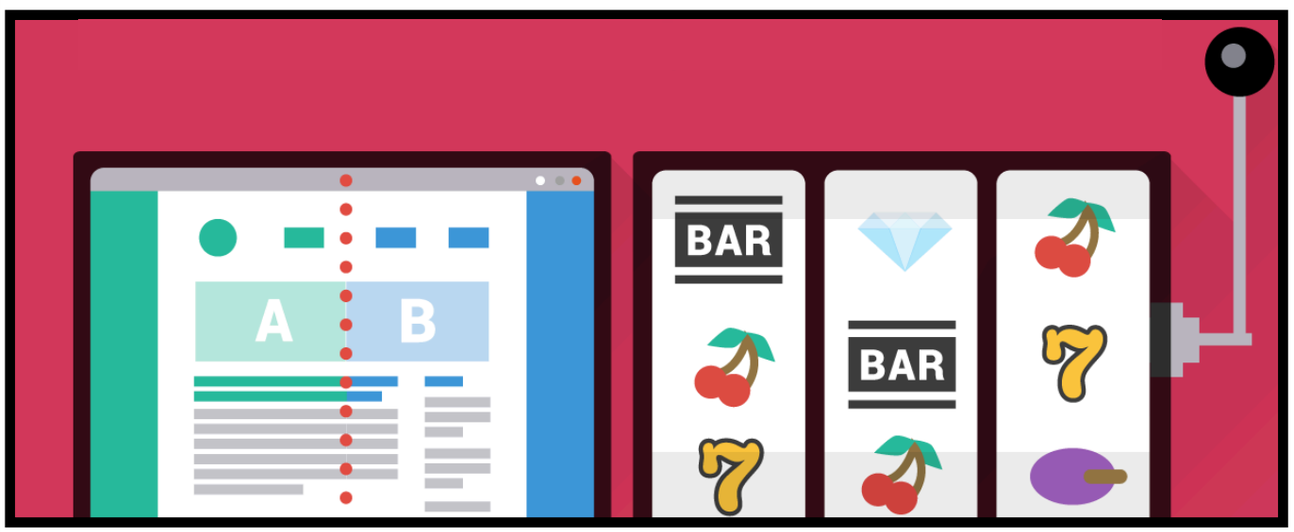
LIKELIHOOD OF NEW DATA

POSTERIOR SERVES AS PRIOR

THOMPSON SAMPLING FOR EXPLORE/EXPLOIT

$$\underset{\text{action}}{\text{argmax}} \; \mathbb{E}[\text{reward}_{T+1}|\text{action}_{T+1}, \hat{\mathbf{W}}]$$

$$\hat{\mathbf{W}} \sim p(\mathbf{W}|\{(\text{reward}_t, \text{action}_t)\}_{t=0}^{T})$$



https://twenga-solutions.com/en/insights/wp-content/uploads/sites/71/2015/05/AB-testing-vs-multi-armed-bandit-algorithms.png

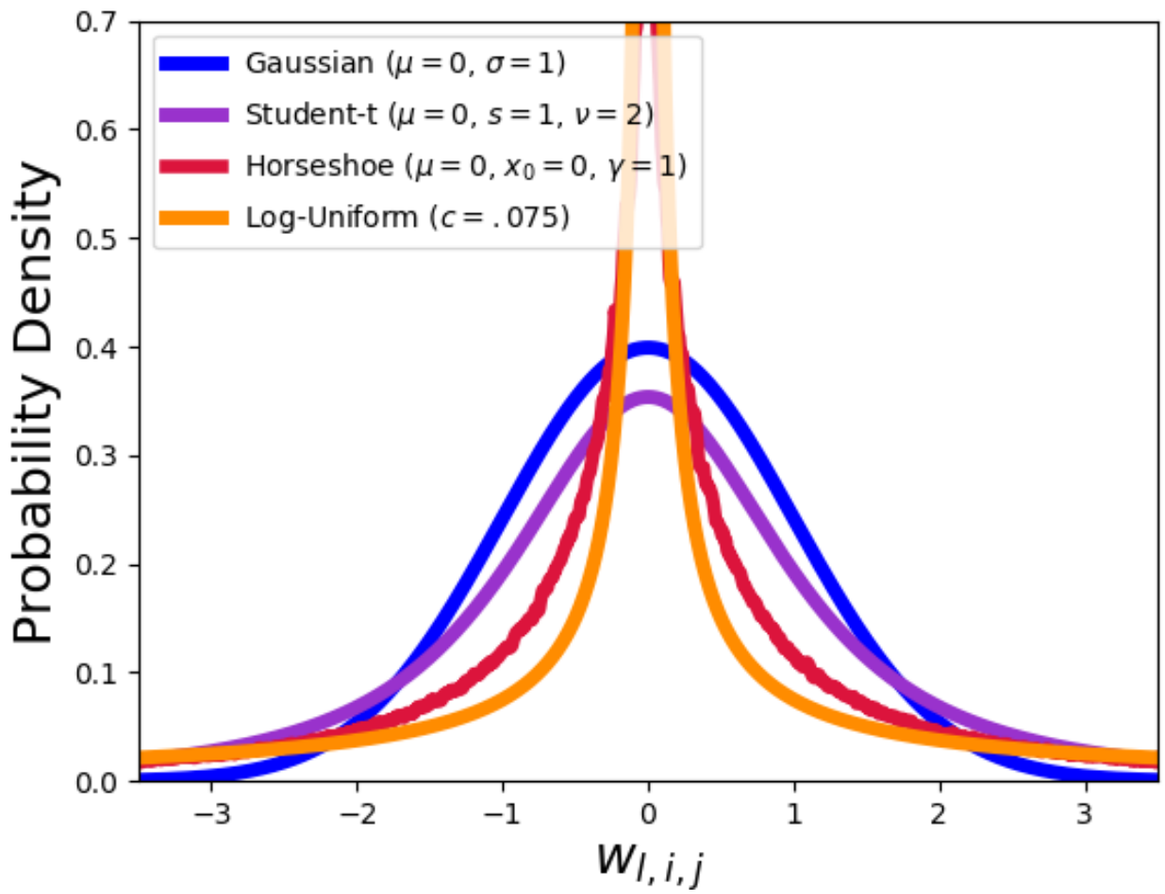
Benefits of Bayesian Neural Networks

Regularization and Model Selection to Ease NN's Data Hunger.

Benefits of Bayesian Neural Networks

Regularization and Model Selection to Ease NN's Data Hunger.

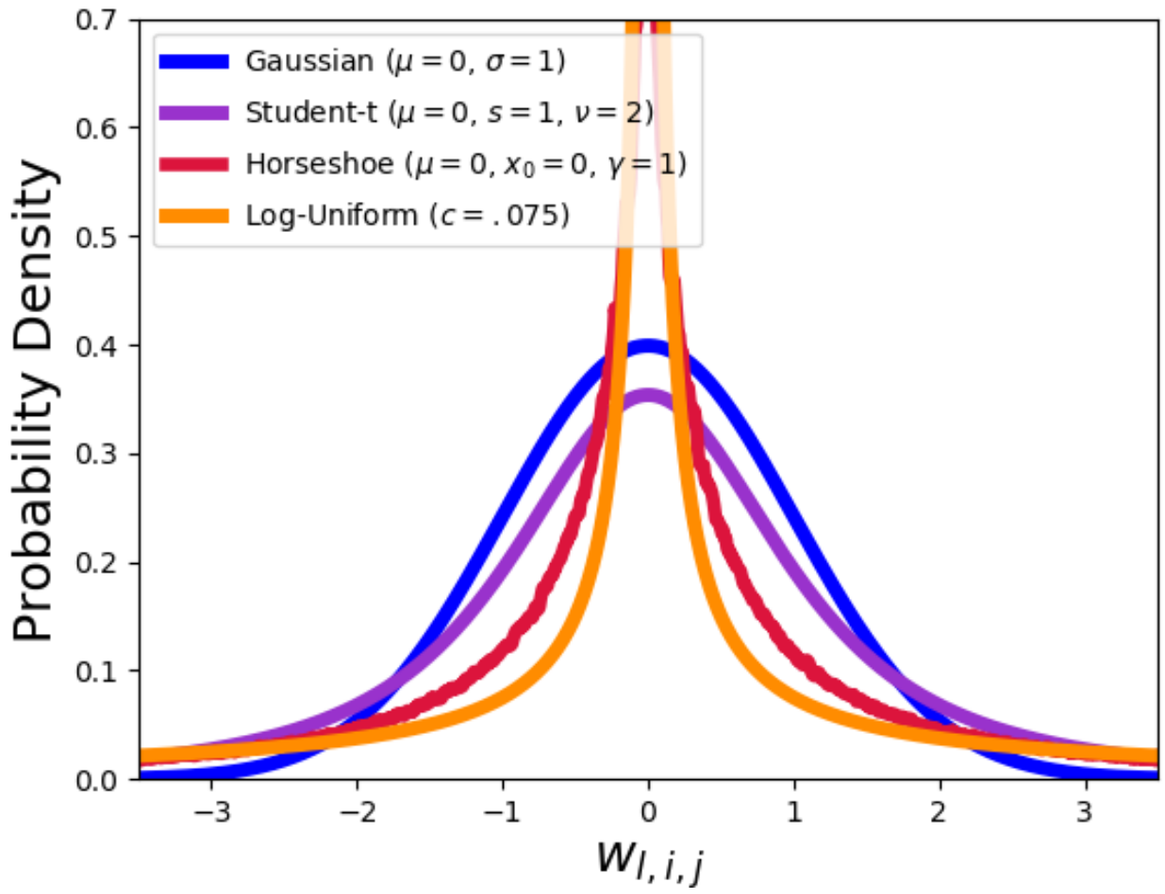
SPARSITY-INDUCING PRIORS



Benefits of Bayesian Neural Networks

Regularization and Model Selection to Ease NN's Data Hunger.

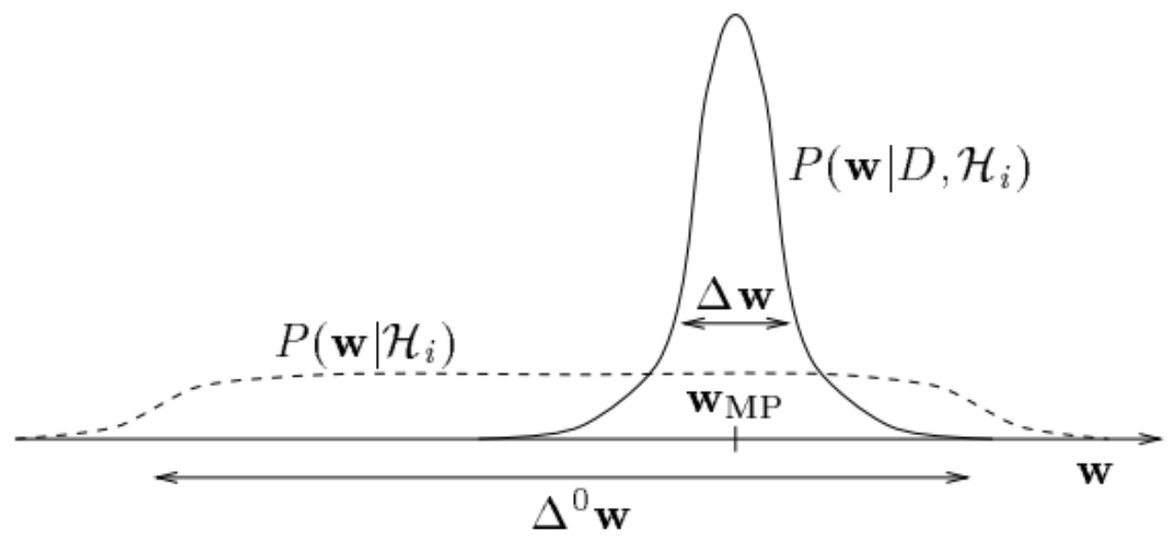
SPARSITY-INDUCING PRIORS



BAYESIAN EVIDENCE FRAMEWORK

$$P(D|\mathcal{H}_i) \simeq \underbrace{P(D|\mathbf{w}_{\mathrm{MP}}, \mathcal{H}_i)}_{\text{Best fit likelihood}} \underbrace{P(\mathbf{w}_{\mathrm{MP}}|\mathcal{H}_i)\,\Delta\mathbf{w}}_{\text{Occam factor}}$$

$$\text{Evidence} \simeq \text{Best fit likelihood} \quad \text{Occam factor}$$



[MacKay, 1991]

Great…so why aren’t Bayesian neural networks more prevalent?

Great…so why aren’t Bayesian neural networks more prevalent?

- **Computational and Memory Cost:** Obtaining distributions instead of point estimates is intrinsically more costly.

Great…so why aren't Bayesian neural networks more prevalent?

- **Computational and Memory Cost:** Obtaining distributions instead of point estimates is intrinsically more costly.
  - → **Computation**: A fully Bayesian treatment requires solving high-dimensional integration problems, which are especially intractable to neural networks.

Great…so why aren't Bayesian neural networks more prevalent?

- **Computational and Memory Cost:** Obtaining distributions instead of point estimates is intrinsically more costly.
  - **Computation**: A fully Bayesian treatment requires solving high-dimensional integration problems, which are especially intractable to neural networks.
  - **Memory**: Even coarse posterior approximations (e.g. fully-factorized Gaussian) usually require at least 2x the number of parameters.

Great…so why aren't Bayesian neural networks more prevalent?

- [ ] **Computational and Memory Cost:** Obtaining distributions instead of point estimates is intrinsically more costly.

- [ ] **Coarse Posterior Approximations:** Usually variational inference is needed to approximate the posterior.

Great…so why aren't Bayesian neural networks more prevalent?

- **Computational and Memory Cost:** Obtaining distributions instead of point estimates is intrinsically more costly.

- **Coarse Posterior Approximations:** Usually variational inference is needed to approximate the posterior.
  - ↳ **Aggressive Factorization**: For the sake of tractability, need to factorize the posterior across layers (and usually further).

$$p(\{\mathbf{W}_l\}_{l=1}^{L+1}|\mathbf{y},\mathbf{X}) \approx \prod_{l=1}^{L+1} q(\mathbf{W}_l)$$

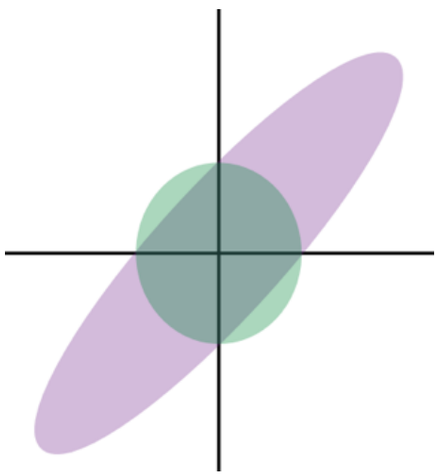
Great…so why aren't Bayesian neural networks more prevalent?

- **Computational and Memory Cost:** Obtaining distributions instead of point estimates is intrinsically more costly.

- **Coarse Posterior Approximations:** Usually variational inference is needed to approximate the posterior.
  - → **Aggressive Factorization**: For the sake of tractability, need to factorize the posterior across layers (and usually further).

$$p(\{\mathbf{W}_l\}_{l=1}^{L+1}|\mathbf{y},\mathbf{X}) \approx \prod_{l=1}^{L+1} q(\mathbf{W}_l)$$

$$\approx \prod_{l=1}^{L+1}\prod_{j=1}^{R_l}\prod_{k=1}^{C_l} q(w_{l,j,k})$$



Exact Posterior

Mean-field Approximation [Blei et al., 2017]

Great…so why aren't Bayesian neural networks more prevalent?

- [ ] **Computational and Memory Cost:** Obtaining distributions instead of point estimates is intrinsically more costly.

- [ ] **Coarse Posterior Approximations:** Usually variational inference is needed to approximate the posterior.

- [ ] **Arbitrary and Difficult to Specify Priors:** NN weights have no intuitive interpretation (in comparison to, say, a Bernoulli model).

Great…so why aren't Bayesian neural networks more prevalent?

- [ ] **Computational and Memory Cost:** Obtaining distributions instead of point estimates is intrinsically more costly.

- [ ] **Coarse Posterior Approximations:** Usually variational inference is needed to approximate the posterior.

- [ ] **Arbitrary and Difficult to Specify Priors:** NN weights have no intuitive interpretation (in comparison to, say, a Bernoulli model).

  ↳ **Unintended Effects**: Priors on weights can have unintended effects on the distribution over output functions [Neal, 1994].

Great…so why aren't Bayesian neural networks more prevalent?

- [ ] **Computational and Memory Cost:** Obtaining distributions instead of point estimates is intrinsically more costly.

- [ ] **Coarse Posterior Approximations:** Usually variational inference is needed to approximate the posterior.

- [ ] **Arbitrary and Difficult to Specify Priors:** NN weights have no intuitive interpretation (in comparison to, say, a Bernoulli model).

  → **Unintended Effects**: Priors on weights can have unintended effects on the distribution over output functions [Neal, 1994].

  → *Garbage in: arbitrary priors*
  Garbage out: uncontrollable error bars
  ~ Michael Jordan on Bayesian Deep Learning, MLSS 2017

Great…so why aren't Bayesian neural networks more prevalent?

- **Computational and Memory Cost:** Obtaining distributions instead of point estimates is intrinsically more costly.

- **Coarse Posterior Approximations:** Usually variational inference is needed to approximate the posterior.

- **Arbitrary and Difficult to Specify Priors:** NN weights have no intuitive interpretation (in comparison to, say, a Bernoulli model).

  → **Unintended Effects**: Priors on weights can have unintended effects on the distribution over output functions [Neal, 1994].

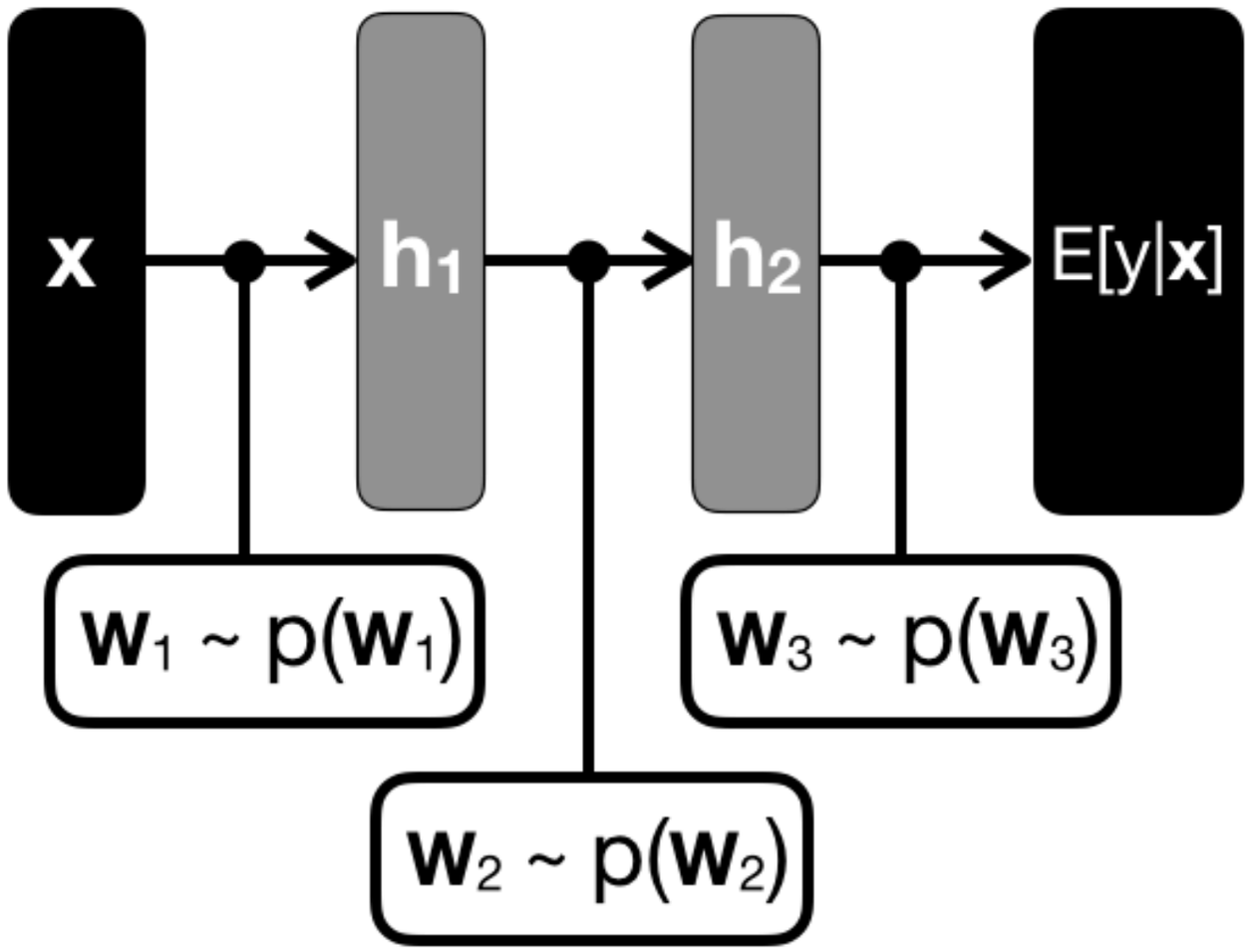
  → *Garbage in: arbitrary priors*
  Garbage out: uncontrollable error bars
  ~ Michael Jordan on Bayesian Deep Learning, MLSS 2017

Bayesian Neural Networks and Their Priors

Bayesian Neural Networks

CONDITIONAL MODEL (SUPERVISED)

[Buntine & Weigend, 1991; MacKay, 1991; Seung et al., 1991]

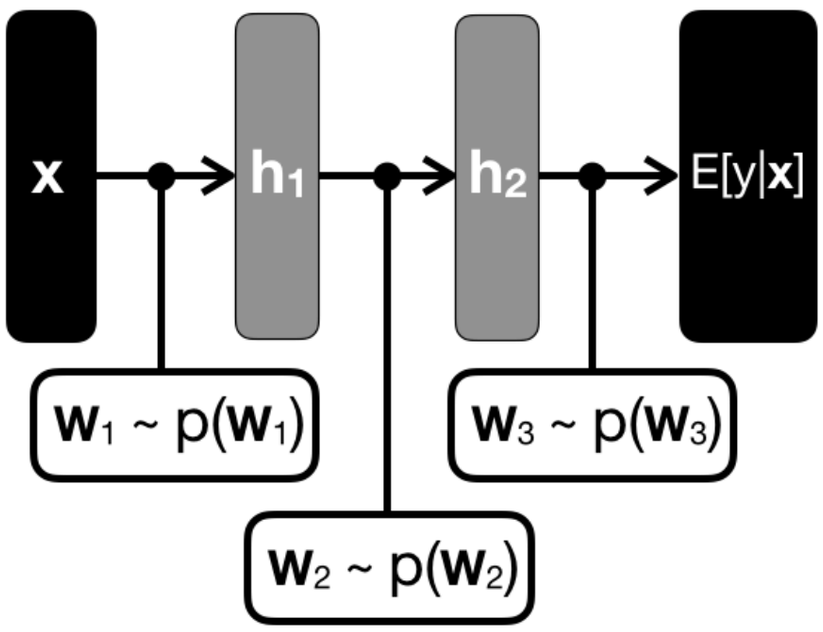


Bayesian Neural Networks

CONDITIONAL MODEL (SUPERVISED)

[Buntine & Weigend, 1991; MacKay, 1991; Seung et al., 1991]

$$p(\{\mathbf{W}_l\}_{l=1}^{L+1}|\mathbf{y},\mathbf{X}) = \frac{p(\mathbf{y}|\mathbf{X},\{\mathbf{W}_l\}_{l=1}^{L+1})\ \prod_{l=1}^{L+1} p(\mathbf{W}_l)}{p(\mathbf{y}|\mathbf{X})}$$

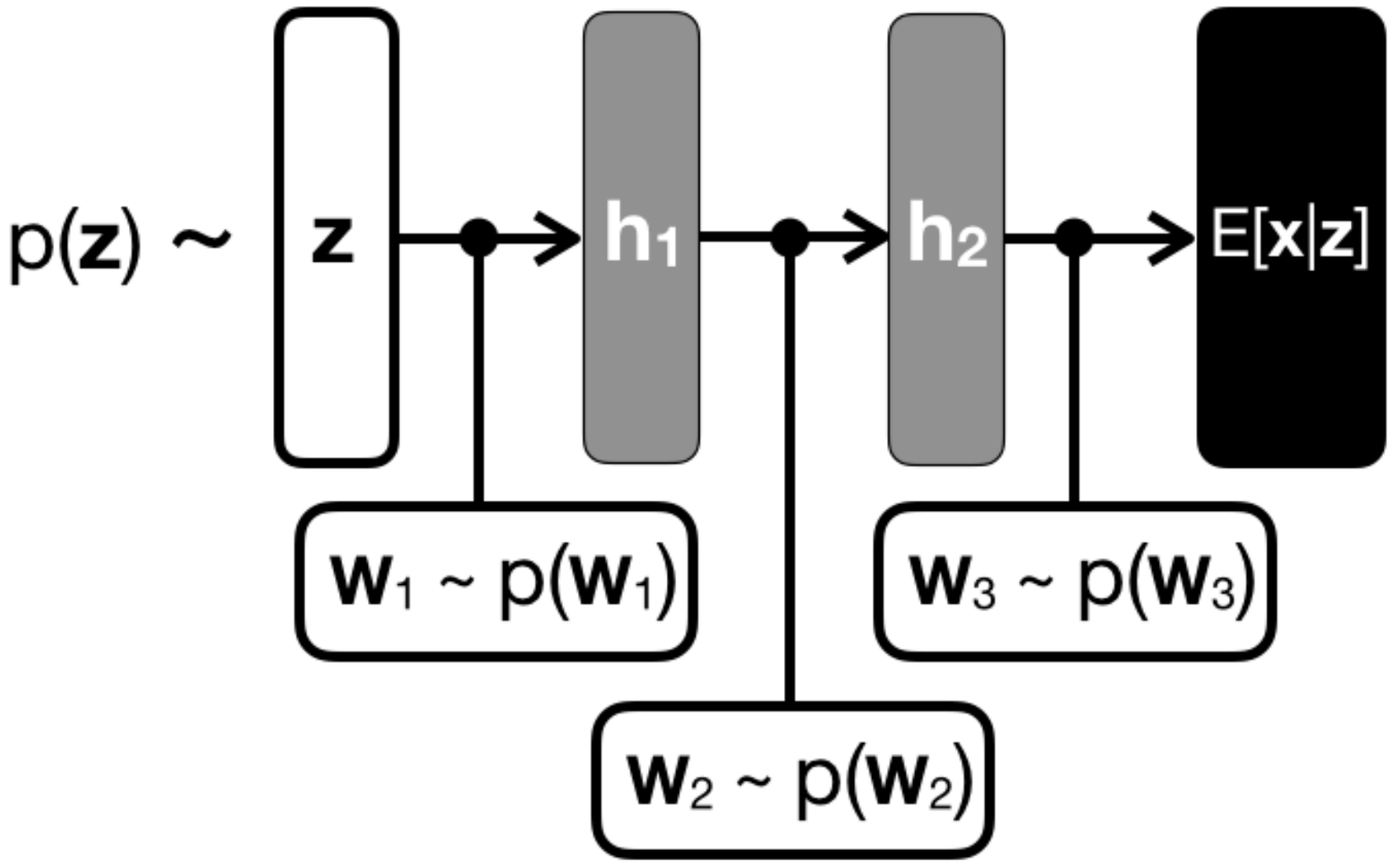


$$p(\mathbf{y}|\mathbf{X}) = \int_{\mathbf{W}} p(\mathbf{y}|\mathbf{X},\{\mathbf{W}_l\}_{l=1}^{L+1})\ \prod_{l=1}^{L+1} p(\mathbf{W}_l)\ d\mathbf{W}$$

Bayesian Neural Networks

DENSITY NETWORK (UNSUPERVISED)

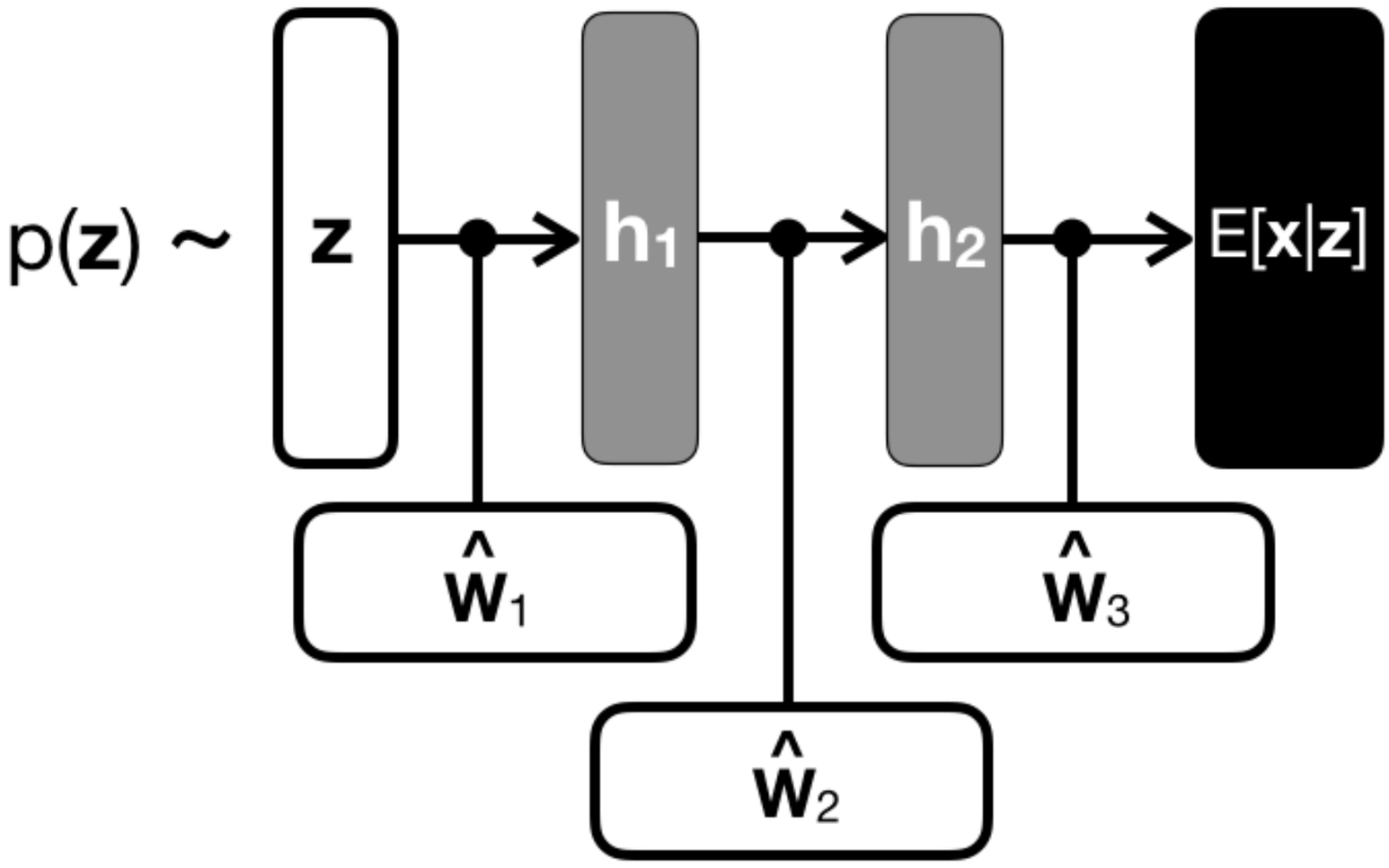
[Amemiya, 1993; MacKay, 1999]



Bayesian Neural Networks

DENSITY NETWORK (UNSUPERVISED)

[Amemiya, 1993; MacKay, 1999]



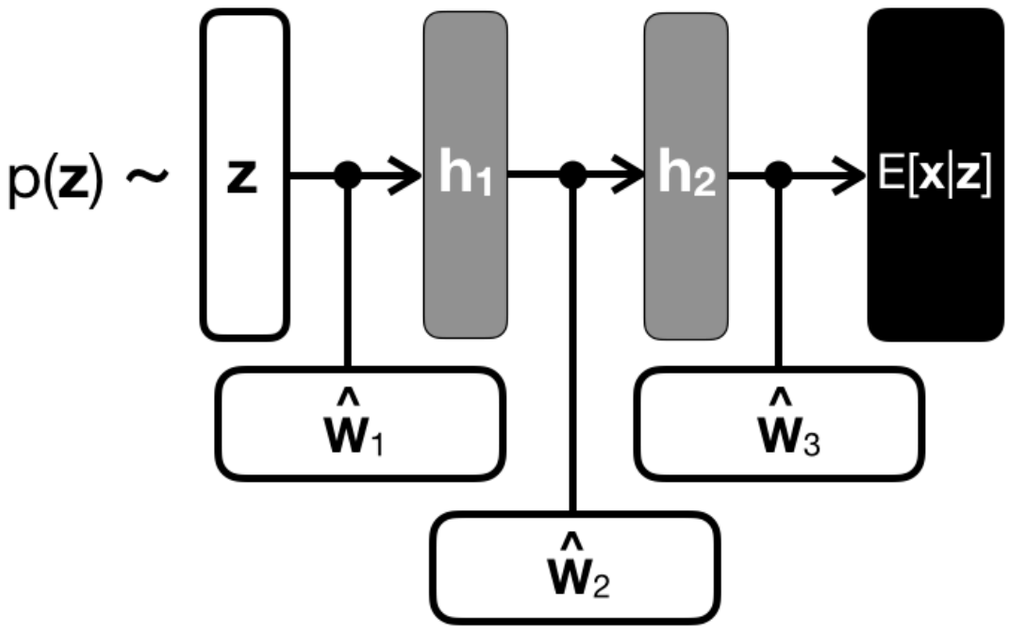
Bayesian Neural Networks

DENSITY NETWORK (UNSUPERVISED)

[Amemiya, 1993; MacKay, 1999]

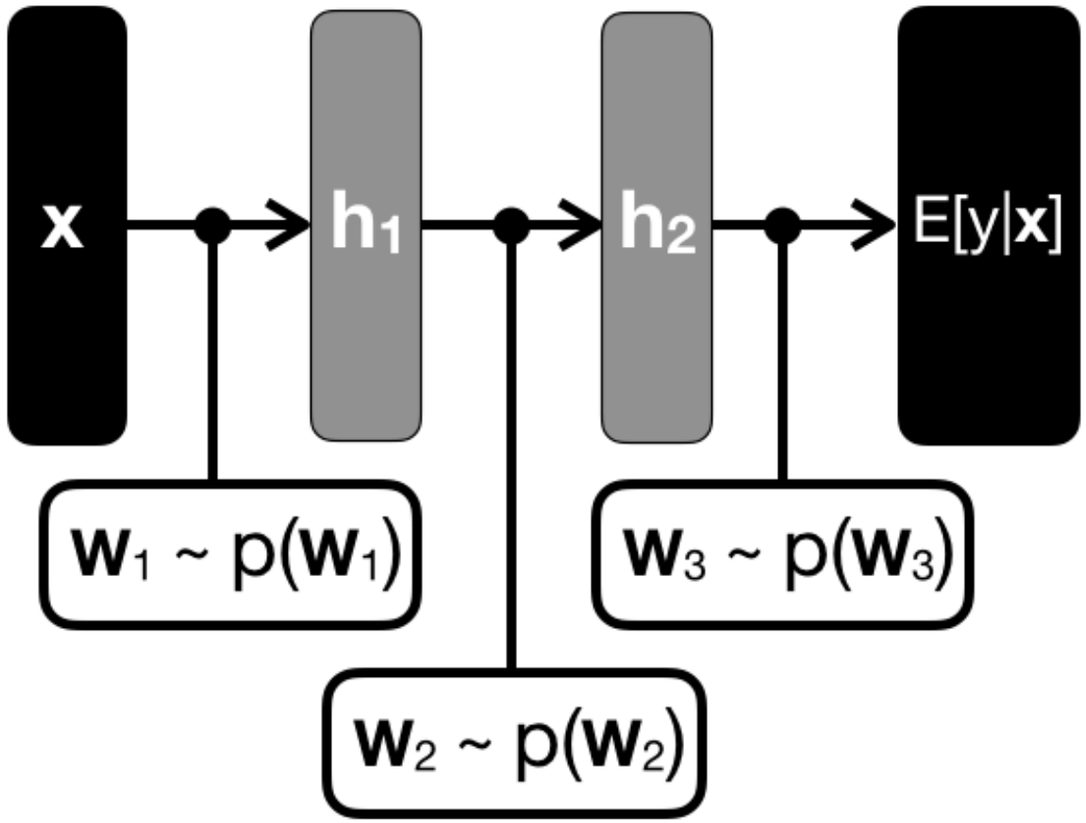
$$p(\{\mathbf{z}_i\}_{i=1}^N|\mathbf{X};\{\mathbf{W}_l\}_{l=1}^{L+1}) = \frac{\prod_{i=1}^N p(\mathbf{x}_i|\mathbf{z}_i;\{\mathbf{W}_l\}_{l=1}^{L+1})p(\mathbf{z}_i)}{p(\mathbf{X};\{\mathbf{W}_l\}_{l=1}^{L+1})}$$

$$p(\mathbf{X};\{\mathbf{W}_l\}_{l=1}^{L+1}) = \prod_{i=1}^N \int_{\mathbf{z}_i} p(\mathbf{x}_i|\mathbf{z}_i;\{\mathbf{W}_l\}_{l=1}^{L+1})p(\mathbf{z}_i)\ d\mathbf{z}_i$$

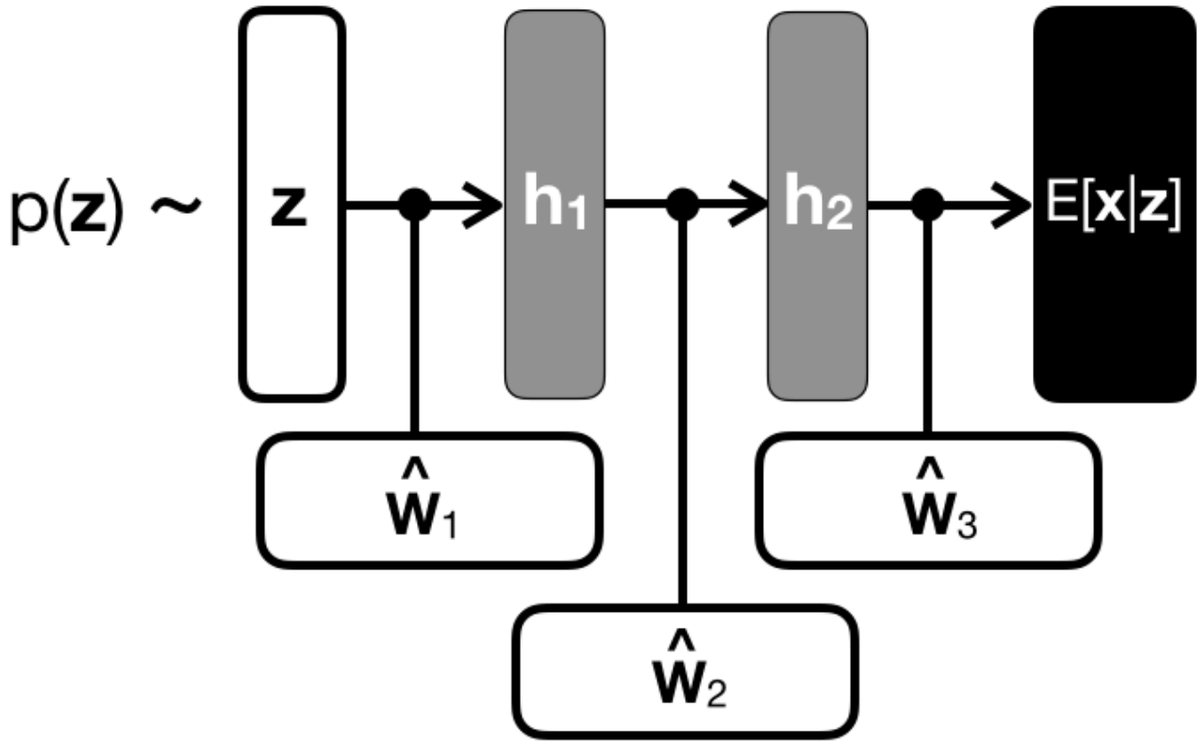


Bayesian Neural Networks

CONDITIONAL MODEL

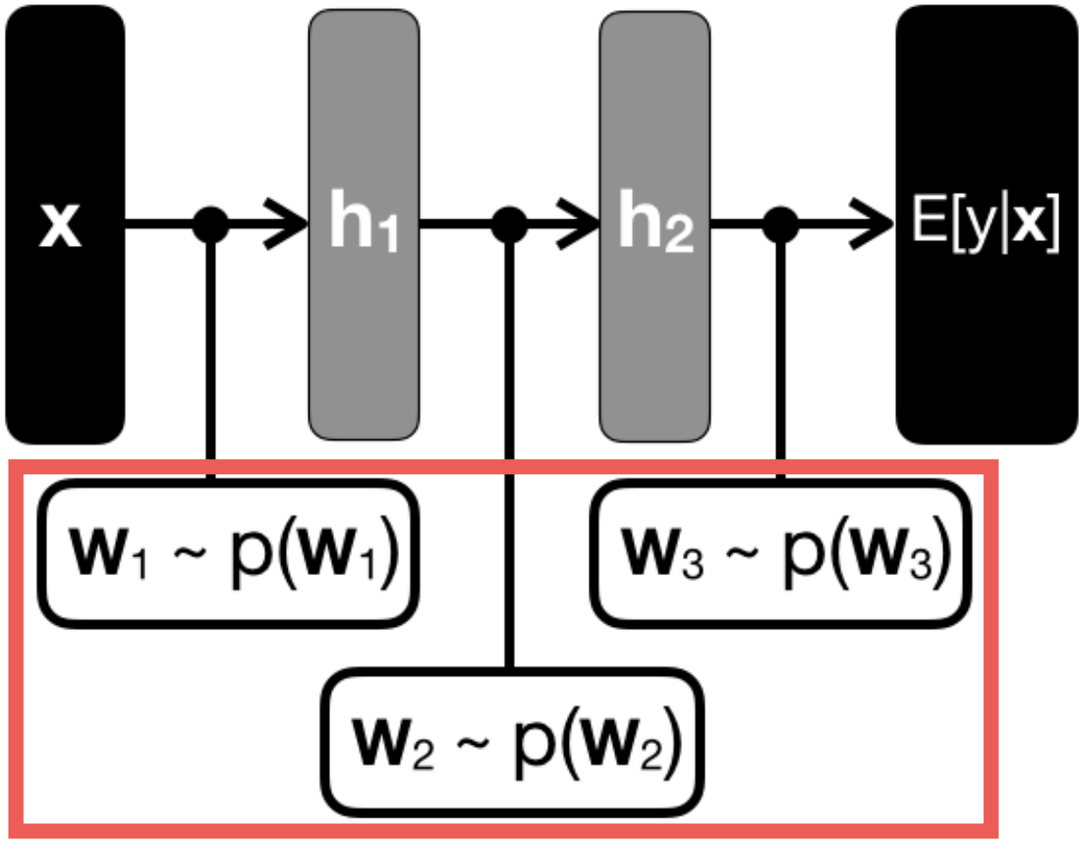


DENSITY NETWORK

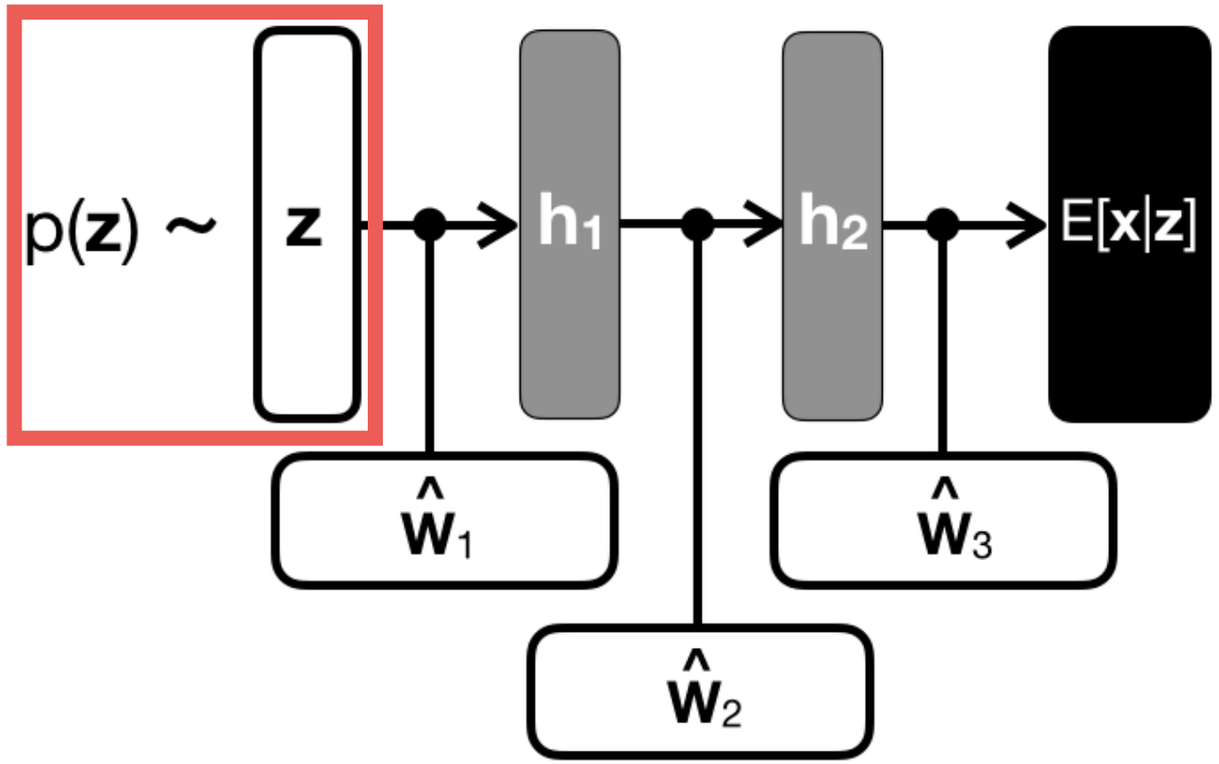


Bayesian Neural Networks

CONDITIONAL MODEL



DENSITY NETWORK



Priors for Neural Networks

A SPECTRUM OF PRIORS

NONPARAMETRIC

Infinite dimensional priors that give the model adaptive capacity. Examples: Dirichlet process, Indian Buffet process.

INDUCTIVE BIAS

Priors that give the model some inductive bias that improves performance. Example: sparsity.

OBJECTIVE

SUBJECTIVE

OBJECTIVE

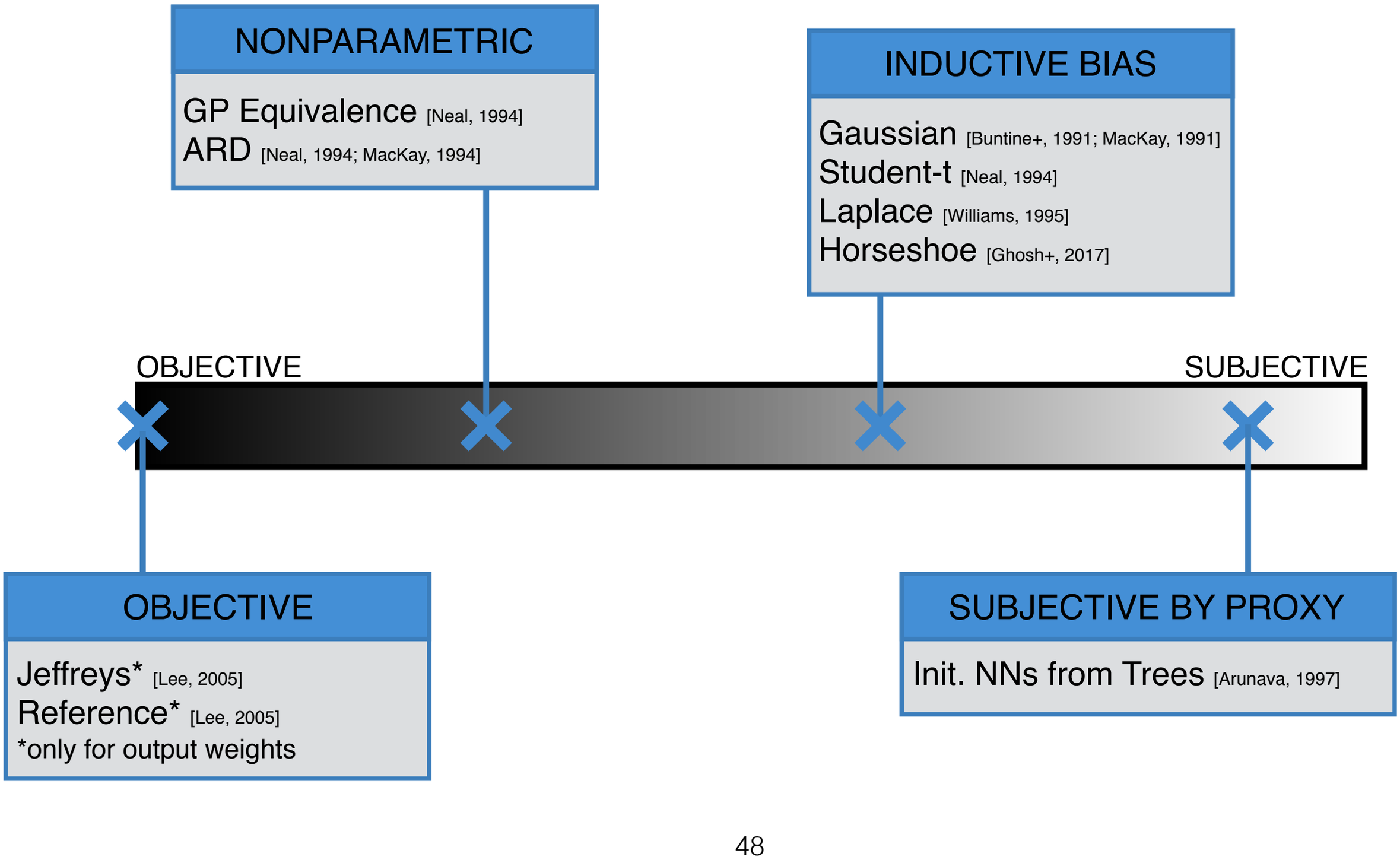
Priors that 'follow the data,' perhaps having frequentist properties. Examples: diffuse, Jeffreys, reference, empirical Bayes.

SUBJECTIVE

Priors determined by previous experiments or expert knowledge.

Priors for Neural Networks

CONDITIONAL MODEL: p(**W**)



Priors for Neural Networks

DENSITY NETWORK: p(**z**)

NONPARAMETRIC

?

INDUCTIVE BIAS

Gaussian [MacKay, 1999; Kingma+, 2014; Rezende+, 2014]
Laplace [Kingma, unpublished]
Scale Mixture [Kingma, unpublished]
Discrete [Neal, 1992]
Structured [Johnson+, 2016]

OBJECTIVE ← → SUBJECTIVE

OBJECTIVE

Parametrized [Chen+, 2017; Makhzani+, 2015]
Empirical/Parametrized [Tomczak+, 2018; Higgins+, 2017]

SUBJECTIVE

Context Free Grammar [Kushner+, 2017]

Priors for Neural Networks

CONDITIONAL MODEL: p(**W**)

NONPARAMETRIC

GP Equivalence [Neal, 1994]
ARD [Neal, 1994; MacKay, 1994]

OBJECTIVE

Jeffreys* [Lee, 2005]
Reference* [Lee, 2005]
*only for output weights

INDUCTIVE BIAS

Gaussian [Buntine+, 1991; MacKay, 1991]
Student-t [Neal, 1994]
Laplace [Williams, 1995]
Horseshoe [Ghosh+, 2017]

SUBJECTIVE BY PROXY

Init. NNs from Trees [Arunava, 1997]

OBJECTIVE

SUBJECTIVE

OBJECTIVE

Parametrized [Chen+, 2017; Makhzani+, 2015]
Empirical/Parametrized [Tomczak+, 2018; Higgins+, 2017]

NONPARAMETRIC

?

INDUCTIVE BIAS

Gaussian [MacKay, 1999; Kingma+, 2014; Rezende+, 2014]
Laplace [Kingma, unpublished]
Scale Mixture [Kingma, unpublished]
Discrete [Neal, 1992]
Structured [Johnson+, 2016]

SUBJECTIVE

Context Free Grammar [Kushner+, 2017]

DENSITY NETWORK: p(**z**)

CONTRIBUTION

Approximating Objective Priors

CHAPTER #5 [Nalisnick & Smyth, UAI 2017]

OBJECTIVE

Priors that ‘follow the data,’ perhaps having frequentist properties. Examples: diffuse, Jeffreys, **reference**, empirical Bayes.

Objective Priors

Reference Priors [Bernardo, 1979] are objective priors defined as:

$$p^*(\boldsymbol{\theta}) = \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}}\ \mathbb{I}[\boldsymbol{\theta}, \mathbf{X}]$$

Objective Priors

Reference Priors [Bernardo, 1979] are objective priors defined as:

$$
\begin{aligned}
p^*(\boldsymbol{\theta}) &= \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}}\ \mathbb{I}[\boldsymbol{\theta}, \mathbf{X}] \\
&= \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}} \int_{\mathbf{X}} p(\mathbf{X})\ \mathrm{KLD}[p(\boldsymbol{\theta}|\mathbf{X})||p(\boldsymbol{\theta})]\ d\mathbf{X}
\end{aligned}
$$

Objective Priors

Reference Priors [Bernardo, 1979] are objective priors defined as:

$$p^*(\boldsymbol{\theta}) = \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}}\ \mathbb{I}[\boldsymbol{\theta}, \mathbf{X}]$$

$$= \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}} \int_{\mathbf{X}} p(\mathbf{X})\ \mathrm{KLD}[p(\boldsymbol{\theta}|\mathbf{X})||p(\boldsymbol{\theta})]\ d\mathbf{X}$$

- [ ] **Independent** of any dataset for generative models.

Objective Priors

Reference Priors [Bernardo, 1979] are objective priors defined as:

$$p^*(\boldsymbol{\theta}) = \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}}\ \mathbb{I}[\boldsymbol{\theta}, \mathbf{X}]$$

$$= \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}} \int_{\mathbf{X}} p(\mathbf{X})\ \mathrm{KLD}[p(\boldsymbol{\theta}|\mathbf{X})||p(\boldsymbol{\theta})]\ d\mathbf{X}$$

- [ ] **Independent** of any dataset for generative models.
- [ ] Need posterior in closed-form and therefore difficult to derive.

Objective Priors

Reference Priors [Bernardo, 1979] are objective priors defined as:

$$p^*(\boldsymbol{\theta}) = \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}}\ \mathbb{I}[\boldsymbol{\theta}, \mathbf{X}]$$

$$= \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}} \int_{\mathbf{X}} p(\mathbf{X})\ \mathrm{KLD}[p(\boldsymbol{\theta}|\mathbf{X})||p(\boldsymbol{\theta})]\ d\mathbf{X}$$

- **Independent** of any dataset for generative models.
- Need posterior in closed-form and therefore difficult to derive.
- Termed *reference* since they serve as a reference against which to compare other priors.

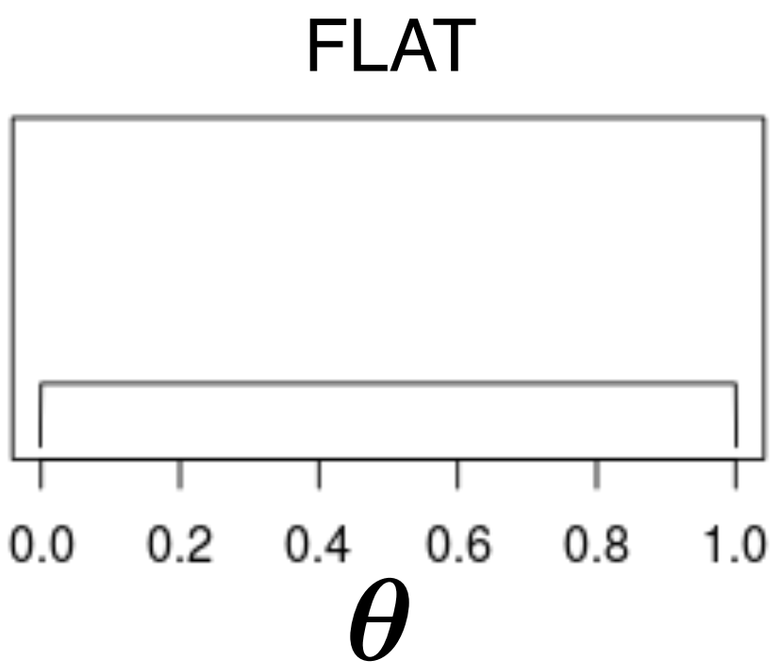
Objective Priors

Reference Priors [Bernardo, 1979] are objective priors defined as:

$$p^*(\boldsymbol{\theta}) = \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}}\ \mathbb{I}[\boldsymbol{\theta}, \mathbf{X}]$$

$$= \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}} \int_{\mathbf{X}} p(\mathbf{X})\ \mathrm{KLD}[p(\boldsymbol{\theta}|\mathbf{X})||p(\boldsymbol{\theta})]\ d\mathbf{X}$$

Bernoulli Priors:



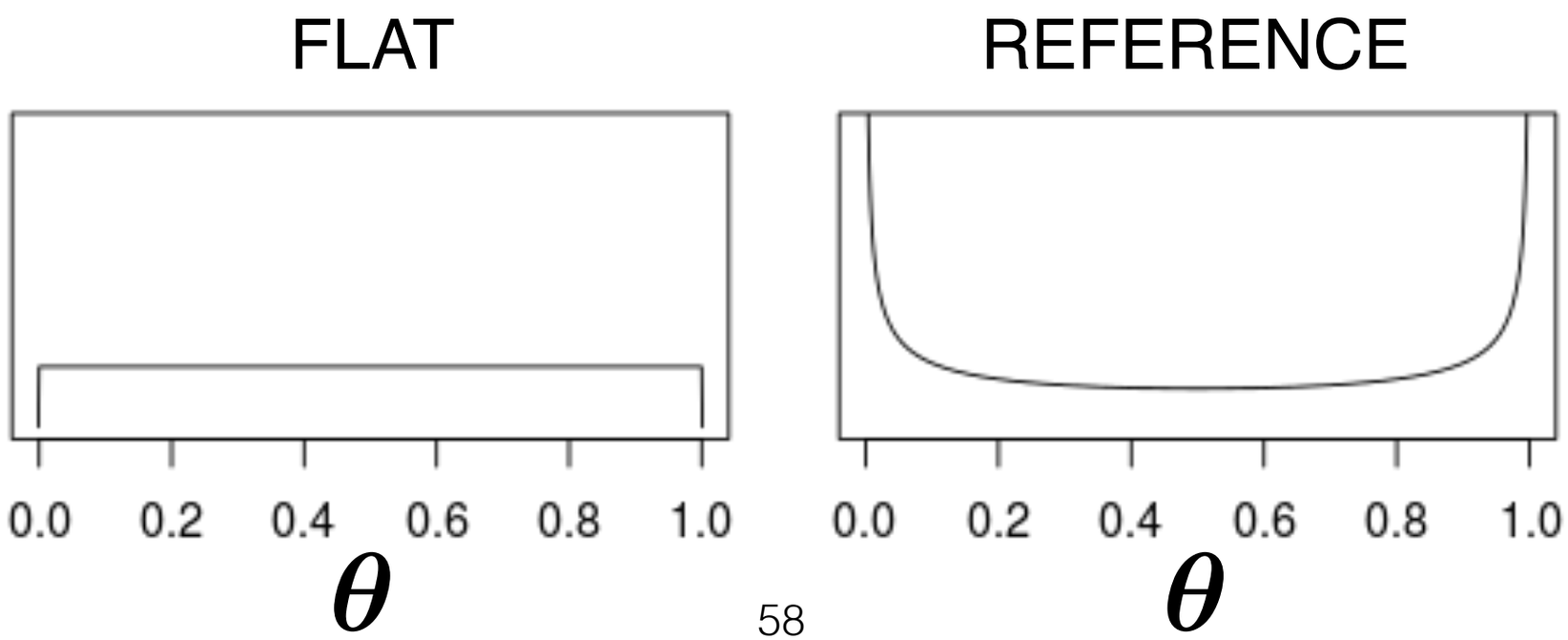
Objective Priors

Reference Priors [Bernardo, 1979] are objective priors defined as:

$$p^*(\boldsymbol{\theta}) = \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}}\ \mathbb{I}[\boldsymbol{\theta}, \mathbf{X}]$$

$$= \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}} \int_{\mathbf{X}} p(\mathbf{X})\ \mathrm{KLD}[p(\boldsymbol{\theta}|\mathbf{X})||p(\boldsymbol{\theta})]\ d\mathbf{X}$$

Bernoulli Priors:



Previous Work

- **Markov Chain Monte Carlo** [Lafferty & Wasserman, 2001]**:** Given a likelihood function, use MCMC to draw samples from the reference prior.

$$\{p(\mathbf{x}|\boldsymbol{\theta})\} \longrightarrow \{\hat{\boldsymbol{\theta}}_1, \ldots, \hat{\boldsymbol{\theta}}_S\}$$

Previous Work

- **Markov Chain Monte Carlo** [Lafferty & Wasserman, 2001]**:** Given a likelihood function, use MCMC to draw samples from the reference prior.
  - **Drawback**: Have to perform a second step of modeling in order to evaluate the prior's density.

$$\{p(\mathbf{x}|\boldsymbol{\theta})\} \longrightarrow \{\hat{\boldsymbol{\theta}}_1, \ldots, \hat{\boldsymbol{\theta}}_S\}$$

Previous Work

- **Markov Chain Monte Carlo** [Lafferty & Wasserman, 2001]**:** Given a likelihood function, use MCMC to draw samples from the reference prior.
  - **Drawback**: Have to perform a second step of modeling in order to evaluate the prior's density.

$$\{p(\mathbf{x}|\boldsymbol{\theta})\} \longrightarrow \{\hat{\boldsymbol{\theta}}_1, \ldots, \hat{\boldsymbol{\theta}}_S\}$$

- **Numerical Algorithm** [Berger et al., 2009]**:** Given a user-specified parameter value and likelihood function, it can compute the reference prior density at that point.

$$\{p(\mathbf{x}|\boldsymbol{\theta}), \boldsymbol{\theta}_0\} \longrightarrow p^*(\boldsymbol{\theta}_0)$$

Previous Work

- **Markov Chain Monte Carlo** [Lafferty & Wasserman, 2001]**:** Given a likelihood function, use MCMC to draw samples from the reference prior.

  ↳ **Drawback**: Have to perform a second step of modeling in order to evaluate the prior's density.

$$\{p(\mathbf{x}|\boldsymbol{\theta})\} \longrightarrow \{\hat{\boldsymbol{\theta}}_1, \ldots, \hat{\boldsymbol{\theta}}_S\}$$

- **Numerical Algorithm** [Berger et al., 2009]**:** Given a user-specified parameter value and likelihood function, it can compute the reference prior density at that point.

  ↳ **Drawback**: Have to choose the evaluation points yourself, which is very hard to do in high dimensions.

$$\{p(\mathbf{x}|\boldsymbol{\theta}), \boldsymbol{\theta}_0\} \longrightarrow p^*(\boldsymbol{\theta}_0)$$

Proposed Method

- **Parametric Approximation:** Posit a parametric family and optimize the parameters to find an approximation.

$$p^*(\boldsymbol{\theta}) \approx p(\boldsymbol{\theta}; \boldsymbol{\lambda}^*)$$

Proposed Method

- **Parametric Approximation:** Posit a parametric family and optimize the parameters to find an approximation.

$$p^*(\boldsymbol{\theta}) \approx p(\boldsymbol{\theta}; \boldsymbol{\lambda}^*)$$

- **Lower Bound Optimization:** We derive a lower bound on the mutual information and optimize it w.r.t. $\boldsymbol{\lambda}$.

$$\mathbb{I}[\boldsymbol{\theta}, \mathbf{X}] \geq \mathbb{E}_{\boldsymbol{\theta}_{\boldsymbol{\lambda}}, \mathbf{x}} \left[\log p(\mathbf{X}|\boldsymbol{\theta}) - \frac{1}{1+\alpha} \log \mathbb{E}_{\boldsymbol{\theta}_{\boldsymbol{\lambda}}} \left[p(\mathbf{X}|\boldsymbol{\theta})^{1+\alpha} \right] \right]$$

$$\text{for } \alpha > 0$$

Proposed Method

- **Parametric Approximation:** Posit a parametric family and optimize the parameters to find an approximation.

$$p^*(\boldsymbol{\theta}) \approx p(\boldsymbol{\theta}; \boldsymbol{\lambda}^*)$$

- **Lower Bound Optimization:** We derive a lower bound on the mutual information and optimize it w.r.t. $\boldsymbol{\lambda}$.

$$\mathbb{I}[\boldsymbol{\theta}, \mathbf{X}] \geq \mathbb{E}_{\boldsymbol{\theta}_{\boldsymbol{\lambda}}, \mathbf{X}} \left[\log p(\mathbf{X}|\boldsymbol{\theta}) - \frac{1}{1+\alpha} \log \mathbb{E}_{\boldsymbol{\theta}_{\boldsymbol{\lambda}}} \left[p(\mathbf{X}|\boldsymbol{\theta})^{1+\alpha} \right] \right]$$

$$\text{for } \alpha > 0$$

$$\approx \mathbb{E}_{\boldsymbol{\theta}_{\boldsymbol{\lambda}}, \mathbf{X}} \left[\log p(\mathbf{X}|\boldsymbol{\theta}) - \max_s \log p(\mathbf{X}|\hat{\boldsymbol{\theta}}_s) \right]$$

[Li & Turner, 2016]

Proposed Method

Lower Bound Optimization: We derive a lower bound on the mutual information and optimize it w.r.t. $\boldsymbol{\lambda}$.

$$\{p(\mathbf{x}|\boldsymbol{\theta}), p(\boldsymbol{\theta};\boldsymbol{\lambda})\} \longrightarrow p(\boldsymbol{\theta};\boldsymbol{\lambda}^*)$$

BENEFITS

Proposed Method

Lower Bound Optimization: We derive a lower bound on the mutual information and optimize it w.r.t. $\boldsymbol{\lambda}$.

$$\{p(\mathbf{x}|\boldsymbol{\theta}), p(\boldsymbol{\theta};\boldsymbol{\lambda})\} \longrightarrow p(\boldsymbol{\theta};\boldsymbol{\lambda}^*)$$

BENEFITS

Gaussian mean: $p^*(\boldsymbol{\mu}) \propto 1$

Proposed Method

Lower Bound Optimization: We derive a lower bound on the mutual information and optimize it w.r.t. $\boldsymbol{\lambda}$.

$$\{p(\mathbf{x}|\boldsymbol{\theta}), p(\boldsymbol{\theta};\boldsymbol{\lambda})\} \longrightarrow p(\boldsymbol{\theta};\boldsymbol{\lambda}^*)$$

BENEFITS

Gaussian mean: $p^*(\boldsymbol{\mu}) \propto 1$

- **Conjugate Reference Priors**: Can pick a prior family that is conjugate and then find the member that is nearest to a reference prior.

Proposed Method

Lower Bound Optimization: We derive a lower bound on the mutual information and optimize it w.r.t. $\boldsymbol{\lambda}$.

$$\{p(\mathbf{x}|\boldsymbol{\theta}), p(\boldsymbol{\theta};\boldsymbol{\lambda})\} \longrightarrow p(\boldsymbol{\theta};\boldsymbol{\lambda}^*)$$

BENEFITS

Gaussian mean: $p^*(\boldsymbol{\mu}) \propto 1$

- **Conjugate Reference Priors**: Can pick a prior family that is conjugate and then find the member that is nearest to a reference prior.
- **Proper Reference Priors**: If concerned b/c a reference prior is improper, can use an approximation that is proper.

Proposed Method

Lower Bound Optimization: We derive a lower bound on the mutual information and optimize it w.r.t. $\boldsymbol{\lambda}$.

$$\{p(\mathbf{x}|\boldsymbol{\theta}), p(\boldsymbol{\theta};\boldsymbol{\lambda})\} \longrightarrow p(\boldsymbol{\theta};\boldsymbol{\lambda}^*)$$

BENEFITS

- **Comparison to Lafferty & Wasserman's [2001] MCMC**: No need to perform a second modeling step to evaluate the prior.

- **Comparison to Berger et al.'s [2009] Algorithm**: No need to specify evaluation points.

Proposed Method

Drawback?: Need to specify the variational family.

Proposed Method

Drawback?: Need to specify the variational family.

Notice that the objective doesn't require evaluation of the prior, only expectations / samples:

$$\mathbb{I}[\boldsymbol{\theta}, \mathbf{X}] \geq \mathbb{E}_{\boldsymbol{\theta}_{\boldsymbol{\lambda}}, \mathbf{x}} \left[\log p(\mathbf{X}|\boldsymbol{\theta}) - \frac{1}{1+\alpha} \log \mathbb{E}_{\boldsymbol{\theta}_{\boldsymbol{\lambda}}} \left[p(\mathbf{X}|\boldsymbol{\theta})^{1+\alpha} \right] \right]$$

Proposed Method

Drawback?: Need to specify the variational family.

Notice that the objective doesn't require evaluation of the prior, only expectations / samples:

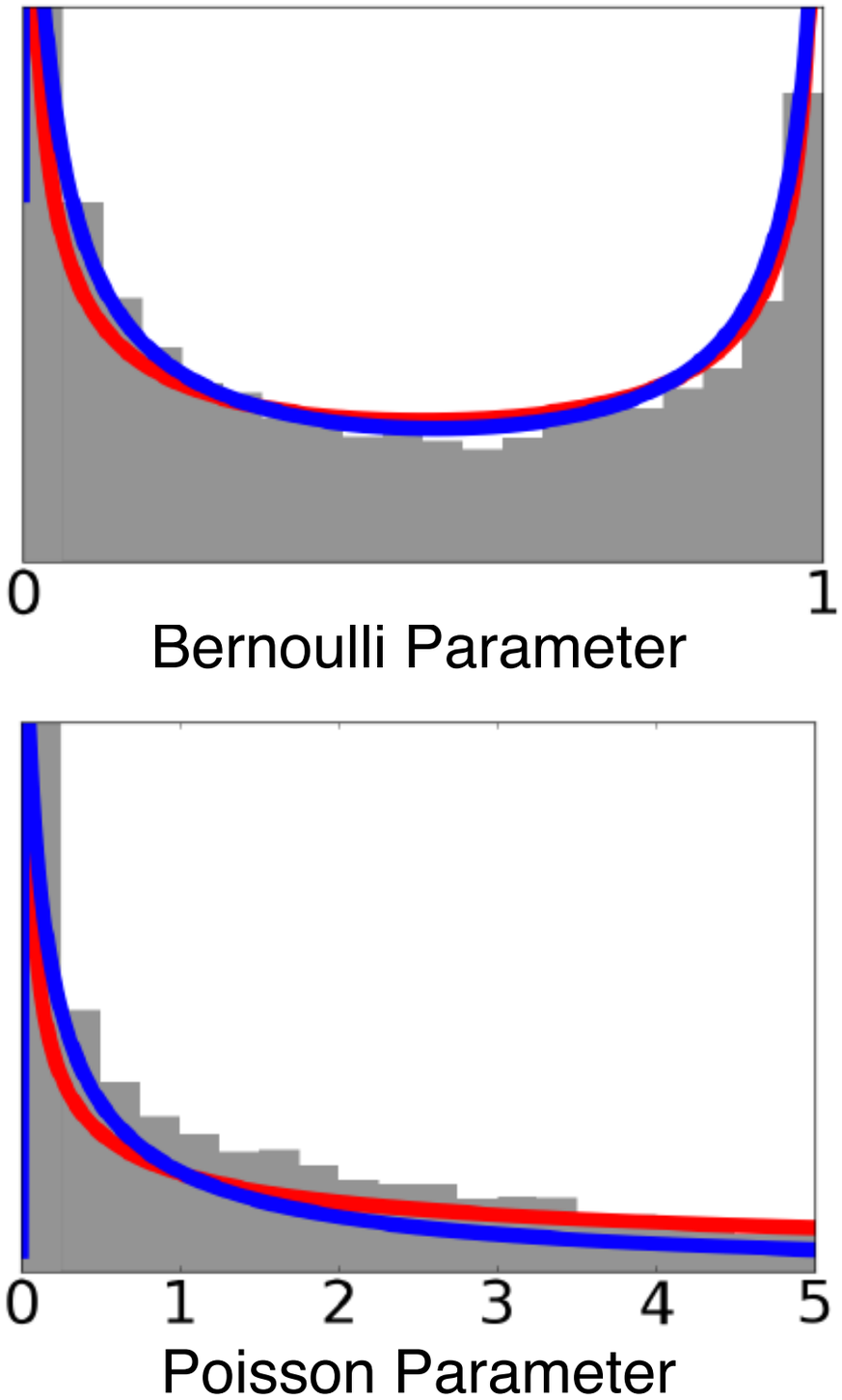
$$\mathbb{I}[\boldsymbol{\theta}, \mathbf{X}] \geq \mathbb{E}_{\boldsymbol{\theta}_{\boldsymbol{\lambda}}, \mathbf{x}} \left[\log p(\mathbf{X}|\boldsymbol{\theta}) - \frac{1}{1+\alpha} \log \mathbb{E}_{\boldsymbol{\theta}_{\boldsymbol{\lambda}}} \left[p(\mathbf{X}|\boldsymbol{\theta})^{1+\alpha} \right] \right]$$

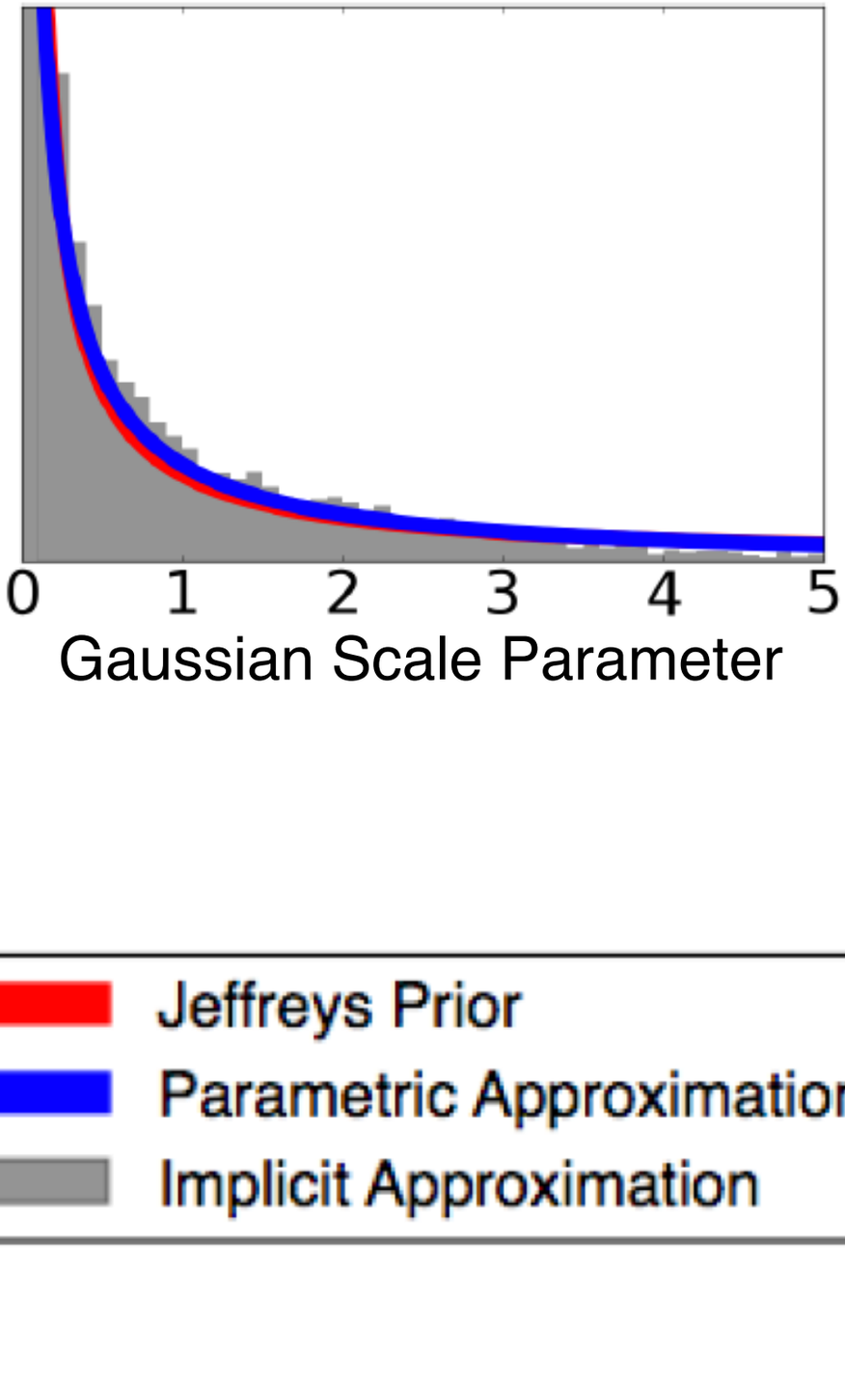
Thus, we can use an **implicit prior**:

$$\hat{\boldsymbol{\theta}} = f(\hat{\boldsymbol{\epsilon}}; \boldsymbol{\lambda}), \quad \hat{\boldsymbol{\epsilon}} \sim p_0(\boldsymbol{\epsilon})$$

This results in *f* being a functional sampler, similar to the MCMC approach.

Recovering Jeffreys Priors





See paper / dissertation for two-sample tests comparing approximation methods.

Finding the Density Net's Reference Prior

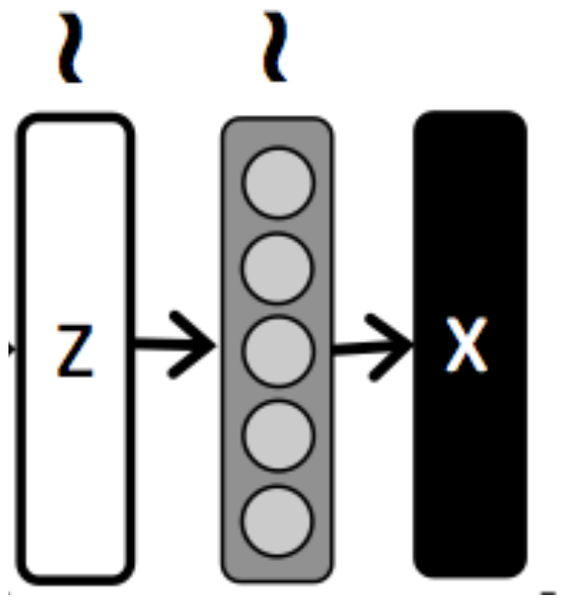
"…to improve our variational bounds we should improve our priors and not just the encoder and decoder….perhaps we should investigate multimodal priors…"

M. Hoffman & M. Johnson. "ELBO Surgery". NIPS 2016 Workshop on *Advances in Approx. Bayesian Inference*.

Finding the Density Net's Reference Prior

ALGORITHM

$p(\mathbf{z};\boldsymbol{\lambda})$ $p(\mathbf{W})$



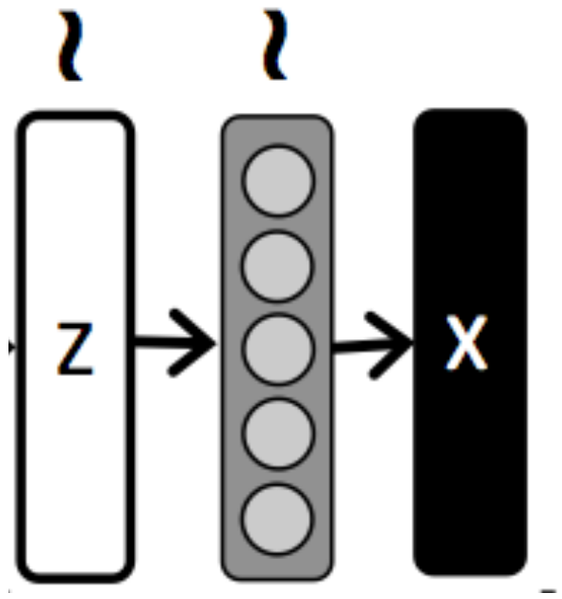
DENSITY NET

Finding the Density Net's Reference Prior

ALGORITHM

STEP #1: sample $\hat{\mathbf{z}}_0 \sim p(\mathbf{z}; \boldsymbol{\lambda})$

$p(\mathbf{z}; \boldsymbol{\lambda})$ $p(\mathbf{W})$



DENSITY NET

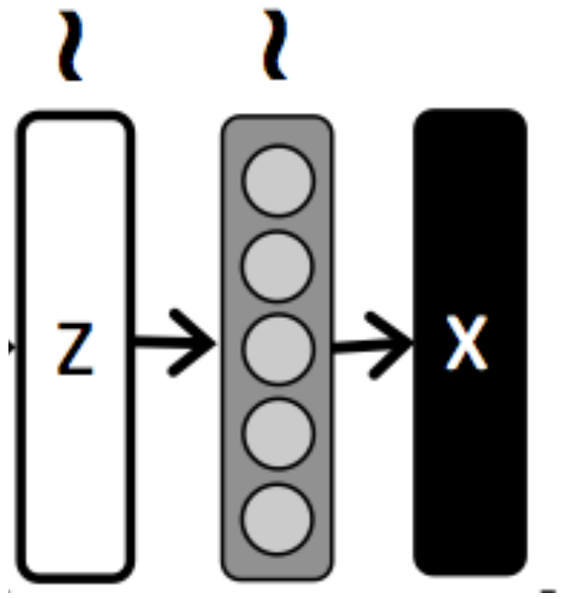
Finding the Density Net's Reference Prior

ALGORITHM

STEP #1: sample $\hat{\mathbf{z}}_0 \sim p(\mathbf{z}; \boldsymbol{\lambda})$

STEP #2: sample $\hat{\mathbf{W}} \sim p(\mathbf{W})$

$p(\mathbf{z}; \boldsymbol{\lambda})$ $p(\mathbf{W})$



DENSITY NET

Finding the Density Net's Reference Prior

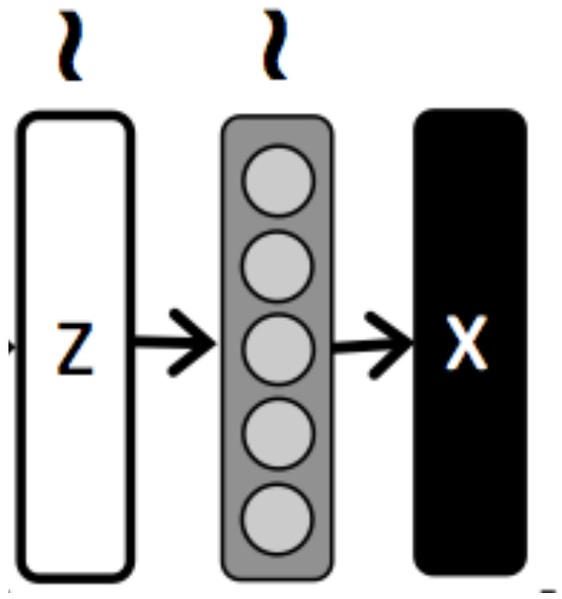
ALGORITHM

STEP #1: sample $\hat{\mathbf{z}}_0 \sim p(\mathbf{z}; \boldsymbol{\lambda})$

STEP #2: sample $\hat{\mathbf{W}} \sim p(\mathbf{W})$

STEP #3: sample $\hat{\mathbf{X}} = \{\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_N\} \sim p(\mathbf{x}|\hat{\mathbf{z}}_0, \hat{\mathbf{W}})$

$p(\mathbf{z}; \boldsymbol{\lambda})$ $p(\mathbf{W})$



DENSITY NET

Finding the Density Net's Reference Prior

ALGORITHM

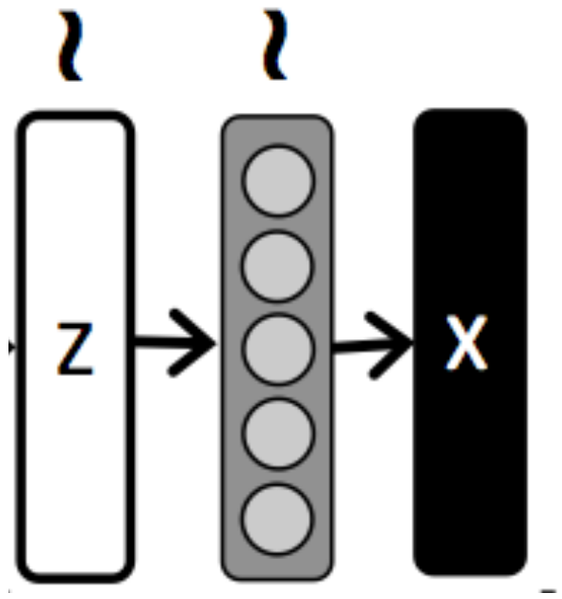
STEP #1: sample $\hat{\mathbf{z}}_0 \sim p(\mathbf{z}; \boldsymbol{\lambda})$

STEP #2: sample $\hat{\mathbf{W}} \sim p(\mathbf{W})$

STEP #3: sample $\hat{\mathbf{X}} = \{\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_N\} \sim p(\mathbf{x}|\hat{\mathbf{z}}_0, \hat{\mathbf{W}})$

STEP #4: sample $\{\hat{\mathbf{z}}_1, \ldots, \hat{\mathbf{z}}_S\} \sim p(\mathbf{z}; \boldsymbol{\lambda})$

$p(\mathbf{z}; \boldsymbol{\lambda})$ $p(\mathbf{W})$



DENSITY NET

Finding the Density Net's Reference Prior

ALGORITHM

STEP #1: sample $\hat{\mathbf{z}}_0 \sim p(\mathbf{z}; \boldsymbol{\lambda})$

STEP #2: sample $\hat{\mathbf{W}} \sim p(\mathbf{W})$

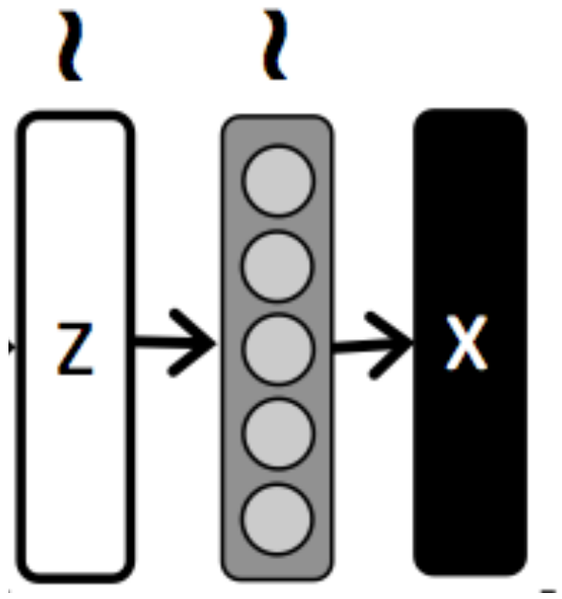
STEP #3: sample $\hat{\mathbf{X}} = \{\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_N\} \sim p(\mathbf{x}|\hat{\mathbf{z}}_0, \hat{\mathbf{W}})$

STEP #4: sample $\{\hat{\mathbf{z}}_1, \ldots, \hat{\mathbf{z}}_S\} \sim p(\mathbf{z}; \boldsymbol{\lambda})$

STEP #5: maximize w.r.t. $\boldsymbol{\lambda}$

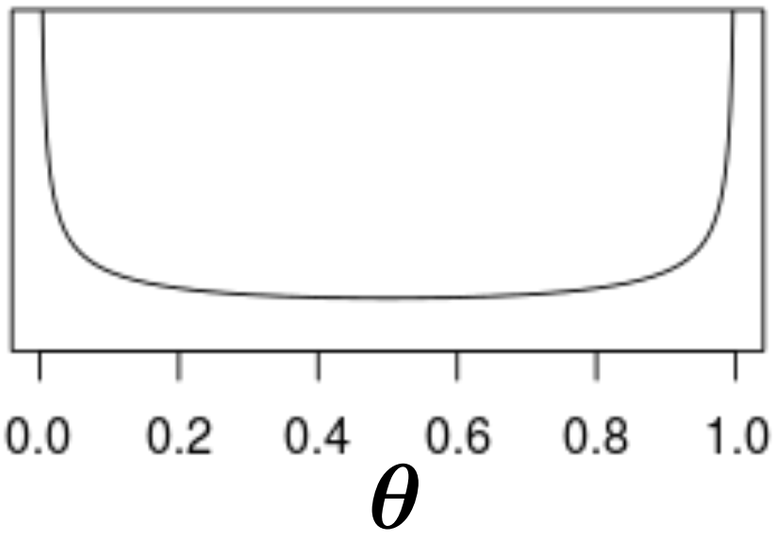
$$\mathcal{J}(\boldsymbol{\lambda}) = \log p(\hat{\mathbf{X}}|\hat{\mathbf{z}}_0, \hat{\mathbf{W}}) - \max_s \log p(\hat{\mathbf{X}}|\hat{\mathbf{z}}_s, \hat{\mathbf{W}})$$

$p(\mathbf{z}; \boldsymbol{\lambda})$ $p(\mathbf{W})$



DENSITY NET

Finding the Density Net's Reference Prior



ALGORITHM

STEP #1: sample $\hat{\mathbf{z}}_0 \sim p(\mathbf{z}; \boldsymbol{\lambda})$

STEP #2: sample $\hat{\mathbf{W}} \sim p(\mathbf{W})$

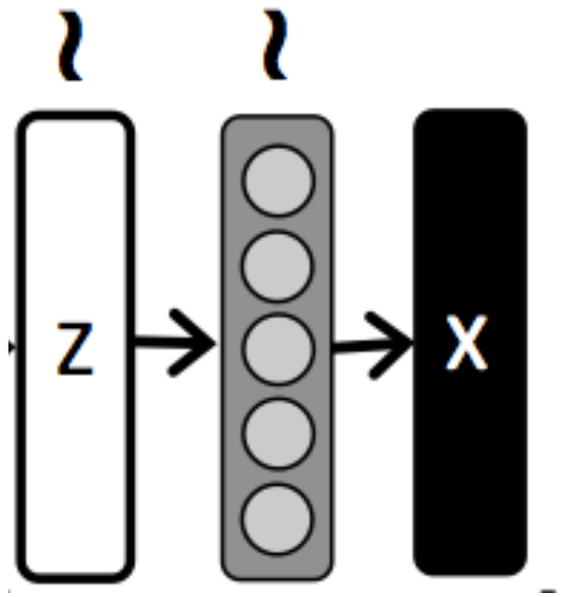
STEP #3: sample $\hat{\mathbf{X}} = \{\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_N\} \sim p(\mathbf{x}|\hat{\mathbf{z}}_0, \hat{\mathbf{W}})$

STEP #4: sample $\{\hat{\mathbf{z}}_1, \ldots, \hat{\mathbf{z}}_S\} \sim p(\mathbf{z}; \boldsymbol{\lambda})$

STEP #5: maximize w.r.t. $\boldsymbol{\lambda}$

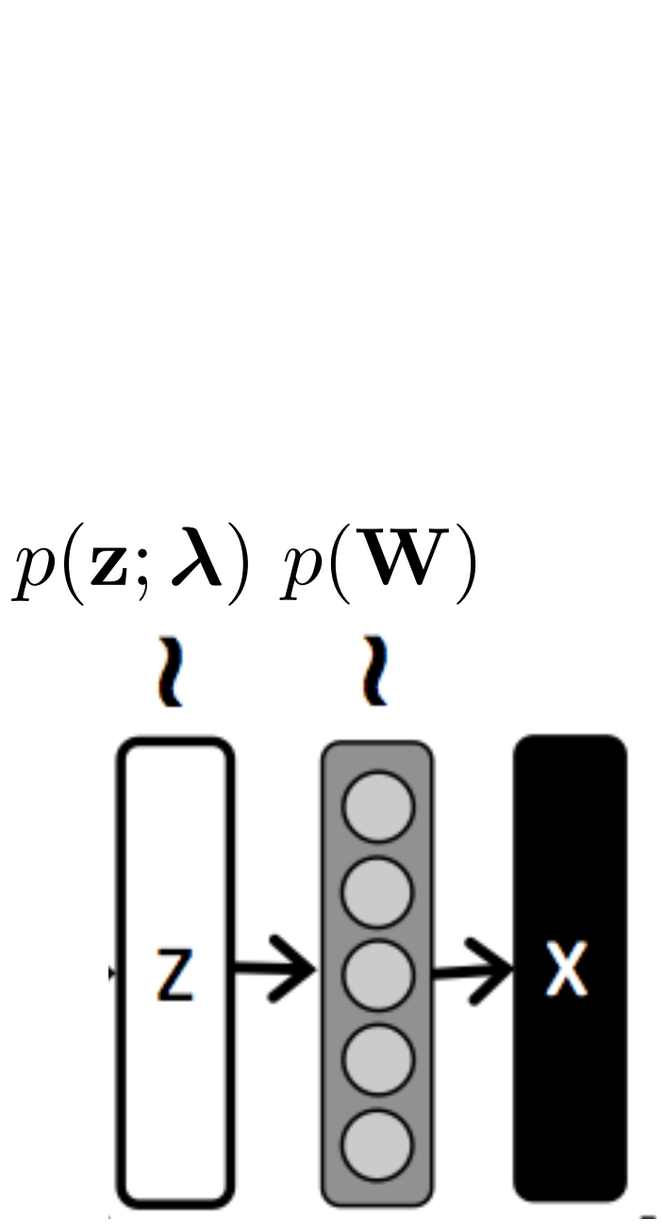
$$\mathcal{J}(\boldsymbol{\lambda}) = \log p(\hat{\mathbf{X}}|\hat{\mathbf{z}}_0, \hat{\mathbf{W}}) - \max_s \log p(\hat{\mathbf{X}}|\hat{\mathbf{z}}_s, \hat{\mathbf{W}})$$

$p(\mathbf{z}; \boldsymbol{\lambda})$ $p(\mathbf{W})$



DENSITY NET

Finding the Density Net's Reference Prior



DENSITY NET

ALGORITHM

STEP #1: sample $\hat{\mathbf{z}}_0 \sim p(\mathbf{z};\boldsymbol{\lambda})$

STEP #2: sample $\hat{\mathbf{W}} \sim p(\mathbf{W})$

STEP #3: sample $\hat{\mathbf{X}} = \{\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_N\} \sim p(\mathbf{x}|\hat{\mathbf{z}}_0, \hat{\mathbf{W}})$

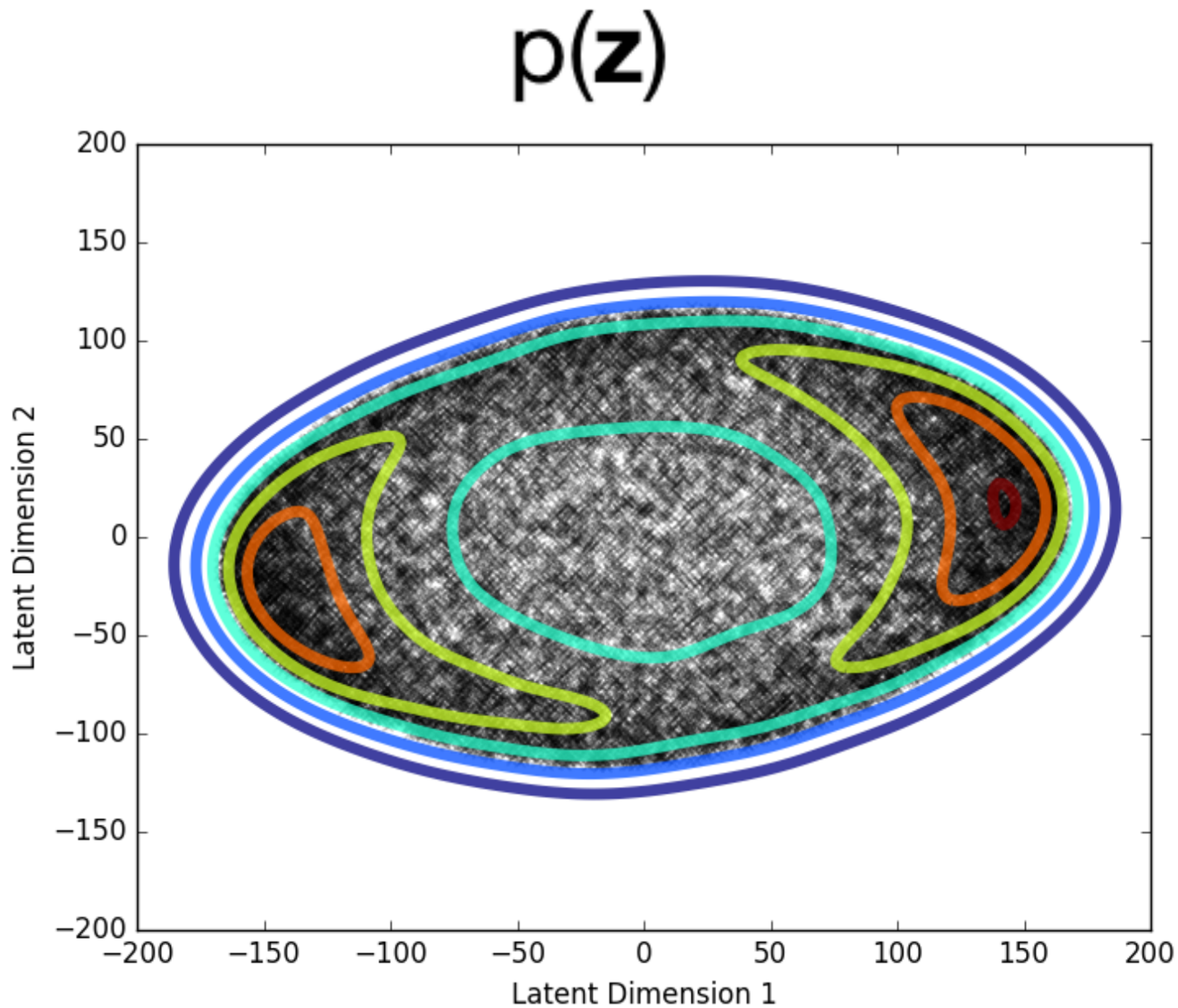
STEP #4: sample $\{\hat{\mathbf{z}}_1, \ldots, \hat{\mathbf{z}}_S\} \sim p(\mathbf{z};\boldsymbol{\lambda})$

STEP #5: maximize w.r.t. $\boldsymbol{\lambda}$

$$\mathcal{J}(\boldsymbol{\lambda}) = \log p(\hat{\mathbf{X}}|\hat{\mathbf{z}}_0, \hat{\mathbf{W}}) - \max_s \log p(\hat{\mathbf{X}}|\hat{\mathbf{z}}_s, \hat{\mathbf{W}})$$

STEP #6: repeat

Finding the Density Net's Reference Prior



Improves performance in low dimensions but no gains in high dimensions (25+).

Priors for Neural Networks

CONDITIONAL MODEL: p(**W**)

NONPARAMETRIC

OBJECTIVE

Jeffreys* [Lee, 2005]
Reference* [Lee, 2005]
Reference Approx.

INDUCTIVE BIAS

SUBJECTIVE BY PROXY

OBJECTIVE

SUBJECTIVE

OBJECTIVE

Parametrized [Chen+, 2017; Makhzani+, 2015]
Empirical/Parametrized [Tomczak+, 2018; Higgins+, 2017]
Reference Approx. **[Nalisnick & Smyth, 2017]**

INDUCTIVE BIAS

SUBJECTIVE

NONPARAMETRIC

DENSITY NETWORK: p(**z**)

CONTRIBUTION

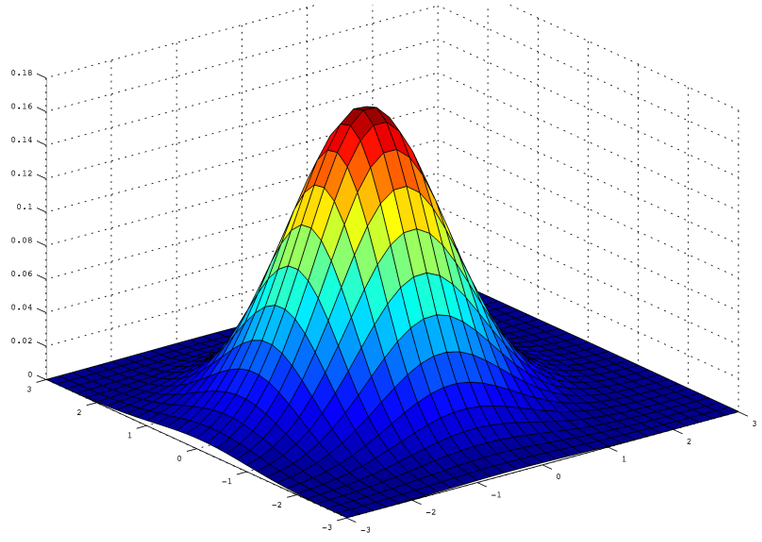
Nonparametric Priors

CHAPTER #7 [Nalisnick & Smyth, ICLR 2017]

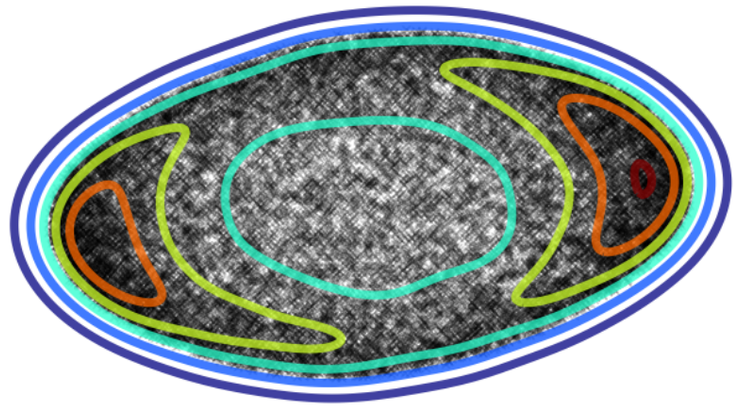
NONPARAMETRIC

Infinite dimensional priors that give the model adaptive capacity. Examples: **Dirichlet process,** Indian Buffet process.

Nonparametric Priors for Density Nets



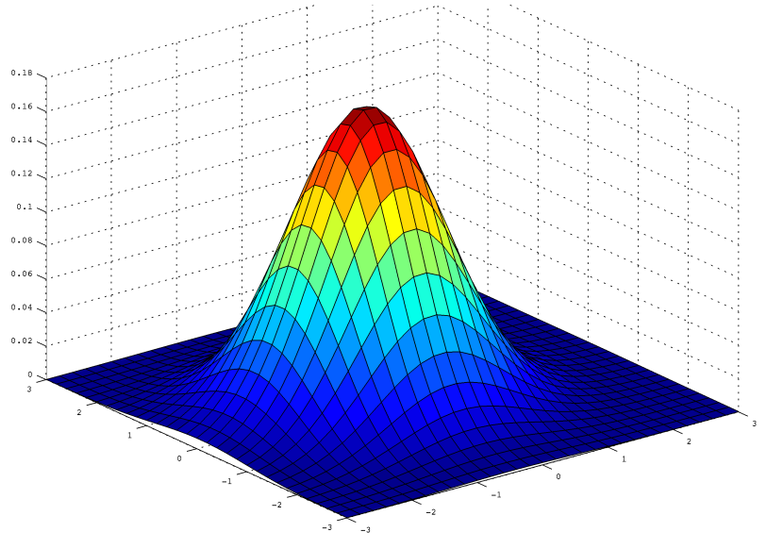
GAUSSIAN PRIOR



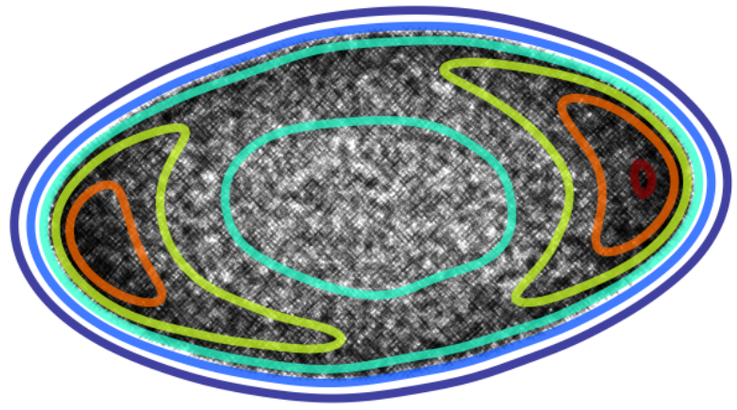
REFERENCE PRIOR

FINITE DIMENSIONAL

Nonparametric Priors for Density Nets



GAUSSIAN PRIOR

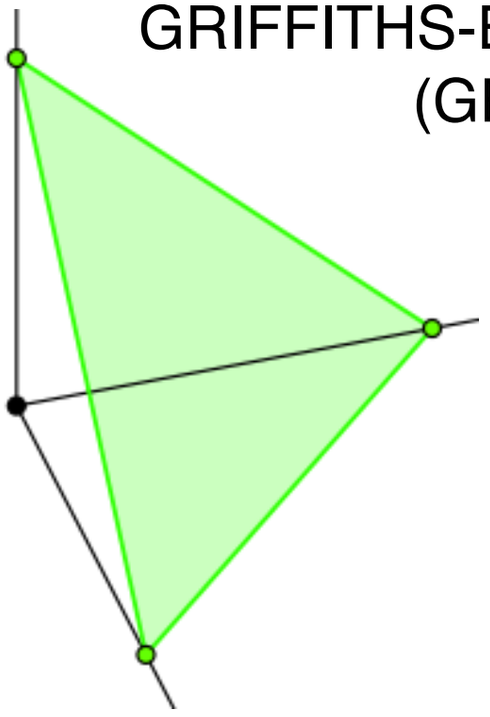


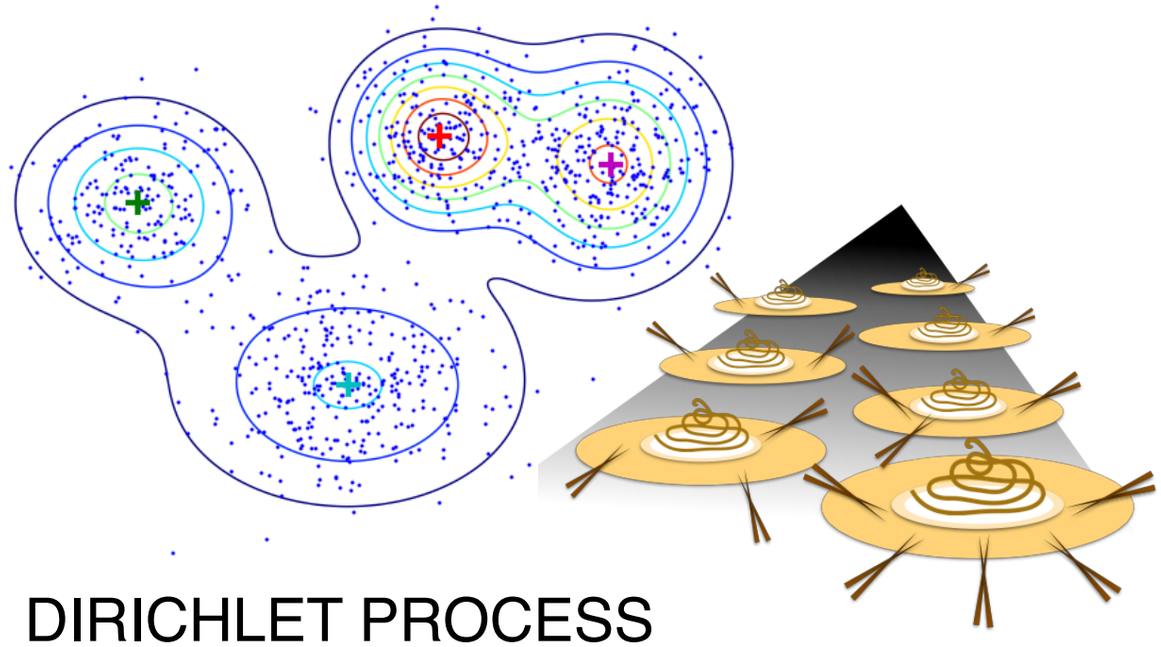
REFERENCE PRIOR

FINITE DIMENSIONAL

INFINITE DIMENSIONAL

GRIFFITHS-ENGEN-McCLOSKEY
(GEM) PRIOR

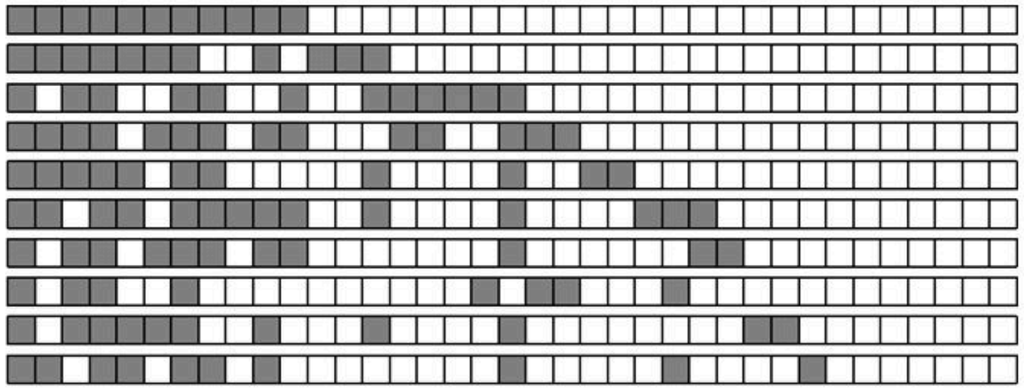




DIRICHLET PROCESS
MIXTURE PRIOR

Nonparametric Priors for Density Nets

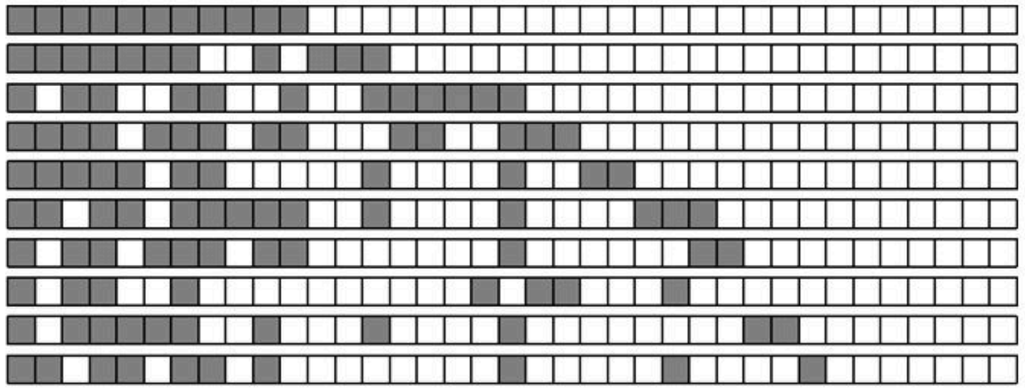
What about the IBP, the natural nonparametric prior for latent feature models?



INDIAN BUFFET PROCESS

Nonparametric Priors for Density Nets

What about the IBP, the natural nonparametric prior for latent feature models?



INDIAN BUFFET PROCESS

Structured Variational Autoencoders for the Beta-Bernoulli Process

Rachit Singh* **Jeffrey Ling*** **Finale Doshi-Velez**
Harvard University
{rachitsingh@college,jling@college,finale@seas}.harvard.edu

Abstract

Beta-Bernoulli processes, also known as Indian buffet processes, are nonparametric priors that allow generative models to automatically infer the number of features in datasets. However, inference for these models proves to be challenging, often relying on specific forms of the likelihood for computational tractability.

We propose to amortize inference using a variational autoencoder trained via gradient descent, allowing for arbitrary likelihood models. Our model extends previously considered mean field variational methods with a structured posterior and new developments in the training of discrete variable VAEs. We experimentally demonstrate a Beta-Bernoulli process VAE that learns decomposable latent features and allows for scalable inference of arbitrary likelihoods on large datasets.

Training

Two methods to compute backpropagation gradients:

- Black box variational inference (BBVI): directly backpropagate with score estimator
- Replace non-reparameterizable variables in variational approximation:
  - Bernoulli $\rightarrow$ Gumbel softmax (Maddison et al. 2017, Jang et al. 2017)
  - Beta $\rightarrow$ Kumaraswamy (Nalisnick & Smyth 2017)

The Dirichlet Process

$$G \sim \mathrm{DP}(\alpha, H)$$

CONCENTRATION

BASE MEASURE

The Dirichlet Process

$$G \sim \mathrm{DP}(\alpha, H)$$

CONCENTRATION

BASE MEASURE

BASE MEASURE

$$\boldsymbol{\mu}_k \sim H(\boldsymbol{\mu})$$

The Dirichlet Process

$$G \sim \mathrm{DP}(\alpha, H)$$

CONCENTRATION

BASE MEASURE

BASE MEASURE

$$\boldsymbol{\mu}_k \sim H(\boldsymbol{\mu})$$

WEIGHTS

$$\{\pi_1, \ldots, \pi_k, \ldots\} = \mathrm{GEM}(\alpha)$$

$$\pi_k = v_k \prod_{j=1}^{k} (1 - v_j)$$

$$v_k \sim \mathrm{Beta}(1, \alpha)$$

The Dirichlet Process

$$G \sim \mathrm{DP}(\alpha, H)$$

CONCENTRATION

BASE MEASURE

BASE MEASURE

$$\boldsymbol{\mu}_k \sim H(\boldsymbol{\mu})$$

WEIGHTS

$$\{\pi_1, \ldots, \pi_k, \ldots\} = \mathrm{GEM}(\alpha)$$

$$\pi_k = v_k \prod_{j=1}^{k} (1 - v_j)$$

$$v_k \sim \mathrm{Beta}(1, \alpha)$$

DISTRIBUTION

$$G(\boldsymbol{\mu}) = \sum_{k=1}^{\infty} \pi_k \delta[\boldsymbol{\mu}_k]$$

The Dirichlet Process

$$G \sim \mathrm{DP}(\alpha, H)$$

CONCENTRATION

BASE MEASURE

The Dirichlet Process

$$G \sim \mathrm{DP}(\alpha, H)$$

CONCENTRATION

BASE MEASURE

$$\hat{\boldsymbol{\mu}} \sim H(\boldsymbol{\mu})$$

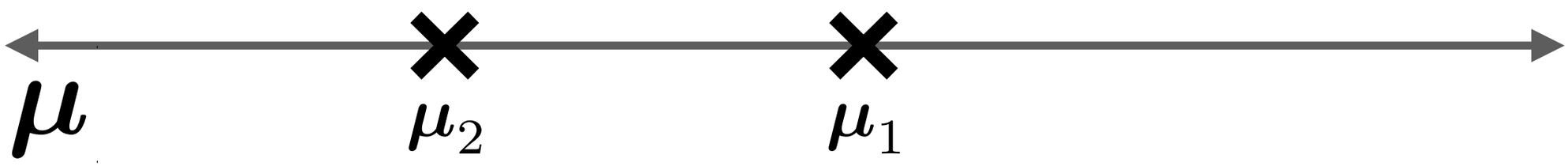
The Dirichlet Process

$$G \sim \mathrm{DP}(\alpha, H)$$

CONCENTRATION

BASE MEASURE

$$\hat{\boldsymbol{\mu}} \sim H(\boldsymbol{\mu})$$



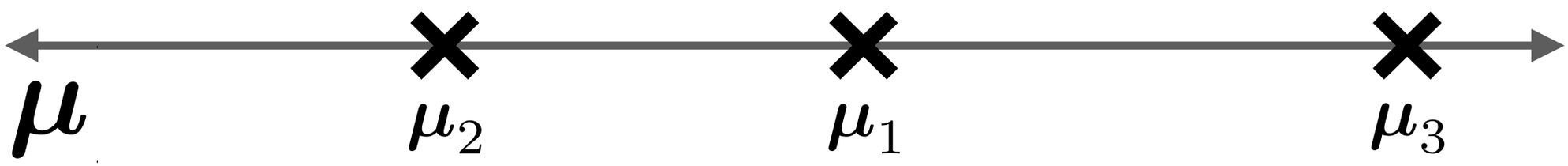
The Dirichlet Process

$$G \sim \mathrm{DP}(\alpha, H)$$

CONCENTRATION

BASE MEASURE

$$\hat{\boldsymbol{\mu}} \sim H(\boldsymbol{\mu})$$

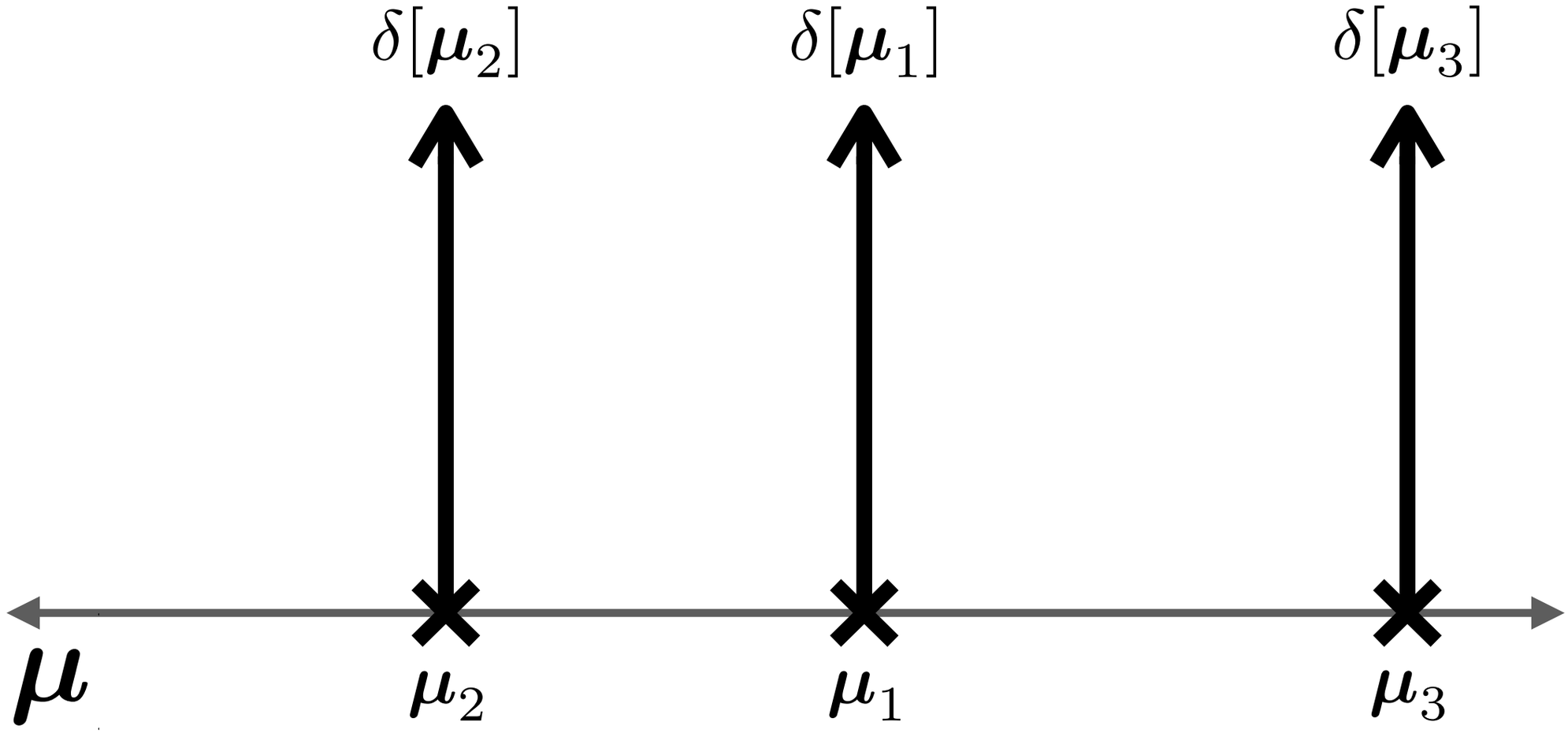


The Dirichlet Process

$$G \sim \mathrm{DP}(\alpha, H)$$

CONCENTRATION

BASE MEASURE



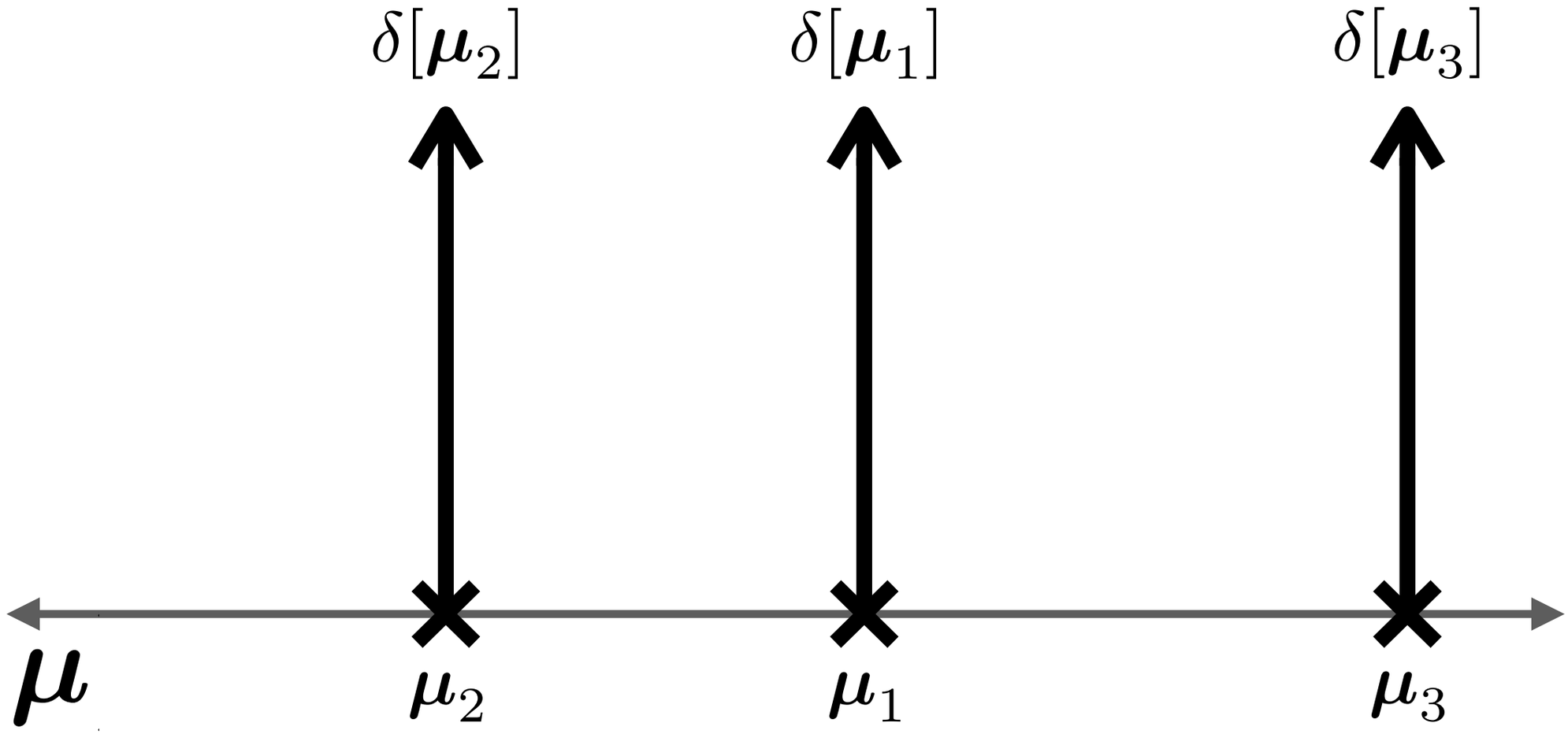
The Dirichlet Process

$$G \sim \mathrm{DP}(\alpha, H)$$

CONCENTRATION

BASE MEASURE

$$\{\hat{\pi}_1, \ldots, \hat{\pi}_3, \ldots\} \sim \mathrm{GEM}(\alpha)$$



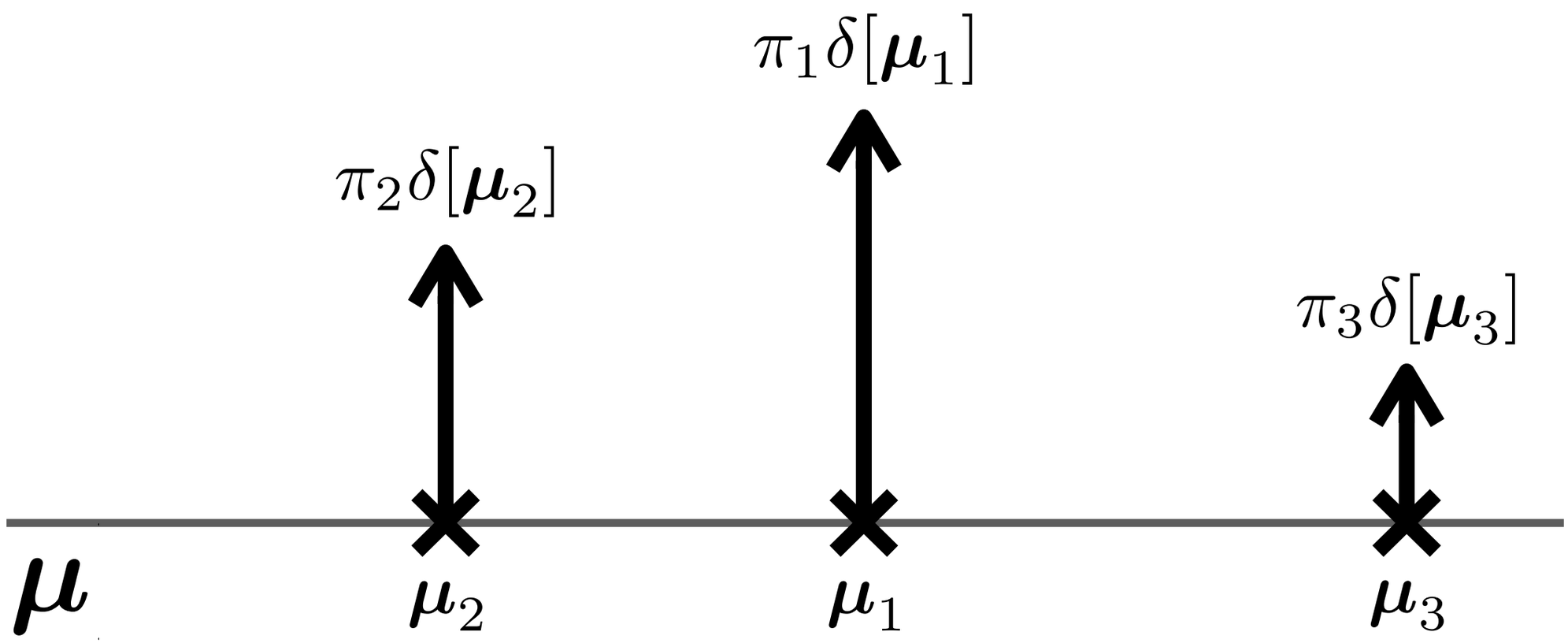
The Dirichlet Process

$$G \sim \mathrm{DP}(\alpha, H)$$

CONCENTRATION

BASE MEASURE

$$\{\hat{\pi}_1, \ldots, \hat{\pi}_3, \ldots\} \sim \mathrm{GEM}(\alpha)$$



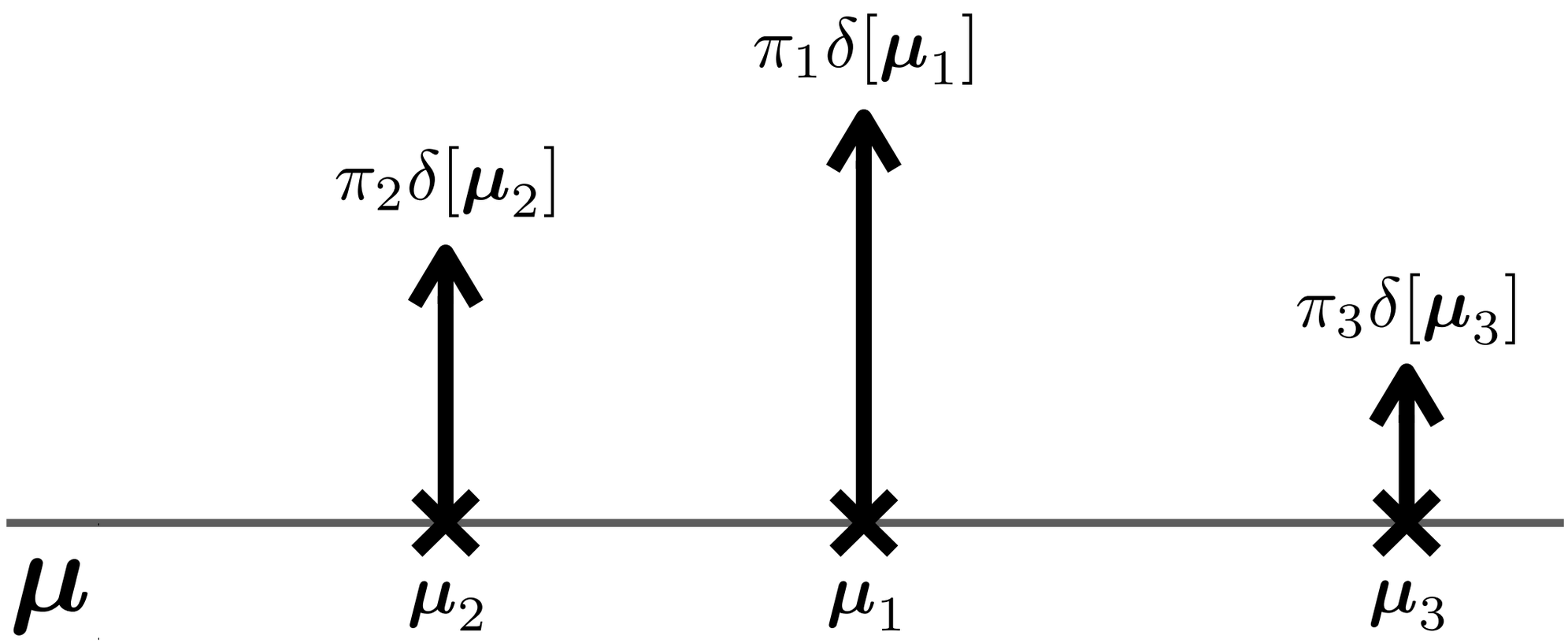
The Dirichlet Process

$$G \sim \mathrm{DP}(\alpha, H)$$

CONCENTRATION

BASE MEASURE

$$\hat{\boldsymbol{\mu}} \sim G(\boldsymbol{\mu})$$



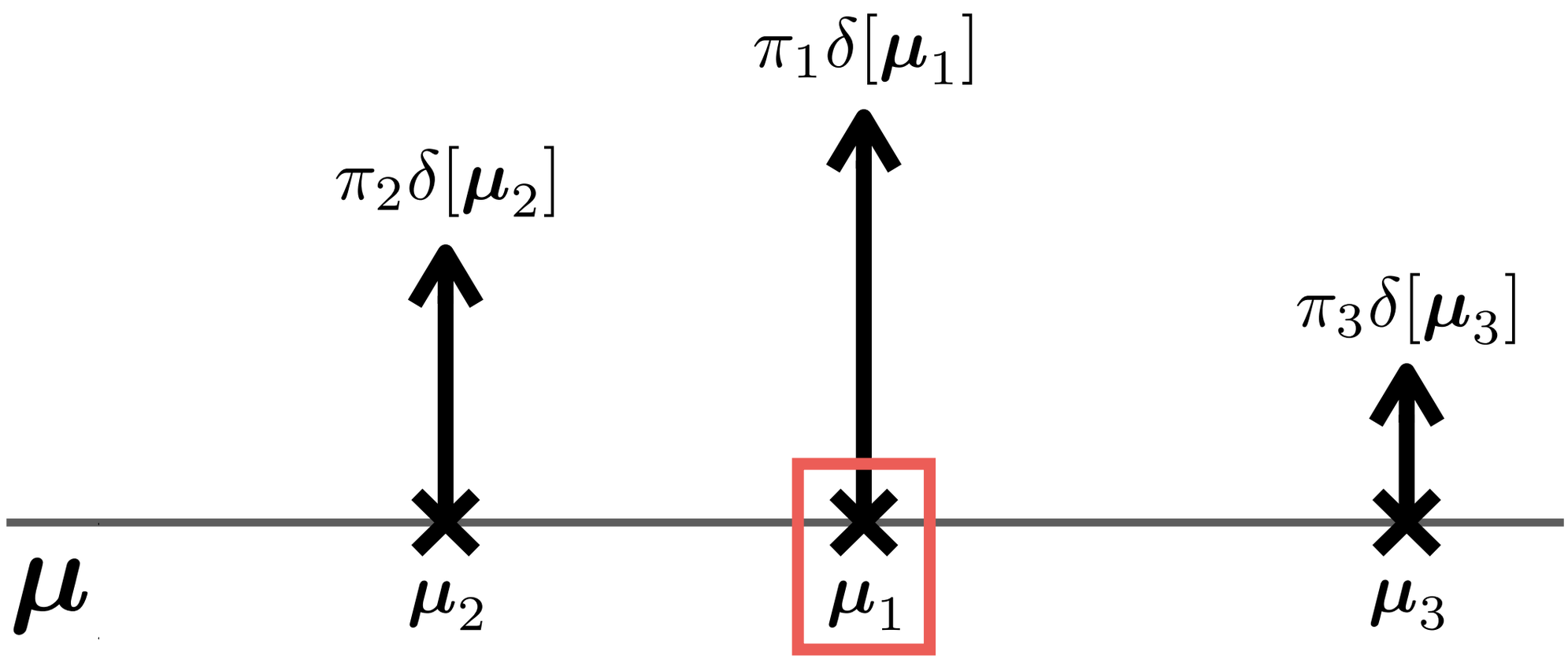
The Dirichlet Process

$$G \sim \mathrm{DP}(\alpha, H)$$

CONCENTRATION

BASE MEASURE

$$\hat{\boldsymbol{\mu}} \sim G(\boldsymbol{\mu})$$



The Dirichlet Process Mixture Model

$$G \sim \mathrm{DP}(\alpha, H)$$

The Dirichlet Process Mixture Model

$$G \sim \mathrm{DP}(\alpha, H) \qquad \mathbf{z}|G \sim \int p(\mathbf{z}|\boldsymbol{\mu})G(\boldsymbol{\mu})$$

KERNEL
(EMISSION DISTRIBUTION)

The Dirichlet Process Mixture Model

$$G \sim \mathrm{DP}(\alpha, H) \qquad \mathbf{z}|G \sim \int p(\mathbf{z}|\boldsymbol{\mu})G(\boldsymbol{\mu})$$

KERNEL
(EMISSION DISTRIBUTION)

COLLAPSING OUT G, WE HAVE…

$$\mathbf{z}|\{\boldsymbol{\mu}_k\}, \{\pi_k\} \sim \sum_{k=1}^{\infty} \pi_k p(\mathbf{z}|\boldsymbol{\mu}_k)$$

$$\{\pi_k\} \sim \mathrm{GEM}(\alpha)$$

$$\{\boldsymbol{\mu}_k\} \sim H(\boldsymbol{\mu})$$

The Dirichlet Process Mixture Model

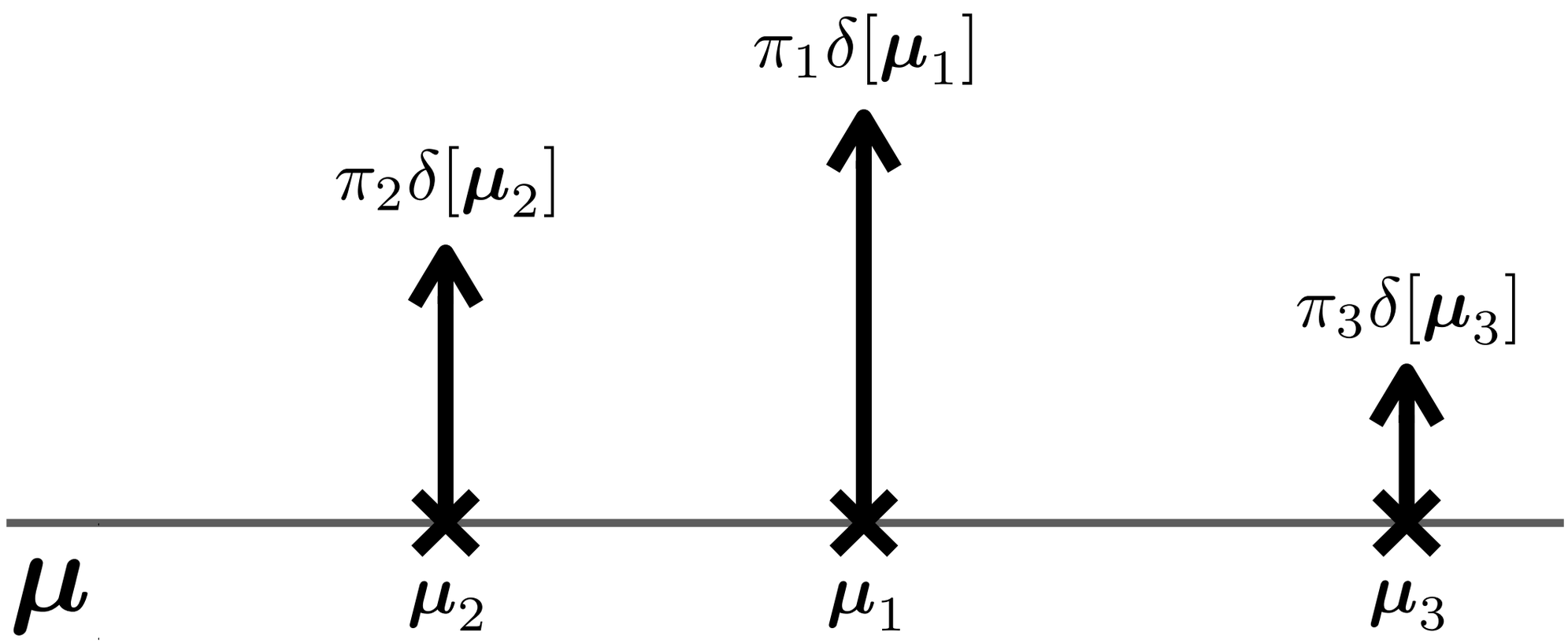
$$G \sim \mathrm{DP}(\alpha, H) \qquad \mathbf{z}|G \sim \int p(\mathbf{z}|\boldsymbol{\mu})G(\boldsymbol{\mu})$$

COLLAPSING OUT G, WE HAVE…

$$\mathbf{z} \sim \mathrm{DPMM}(p, \alpha, H) \Bigg| \begin{aligned} \mathbf{z}|\{\boldsymbol{\mu}_k\}, \{\pi_k\} &\sim \sum_{k=1}^{\infty} \pi_k p(\mathbf{z}|\boldsymbol{\mu}_k) \\ \{\pi_k\} &\sim \mathrm{GEM}(\alpha) \\ \{\boldsymbol{\mu}_k\} &\sim H(\boldsymbol{\mu}) \end{aligned}$$

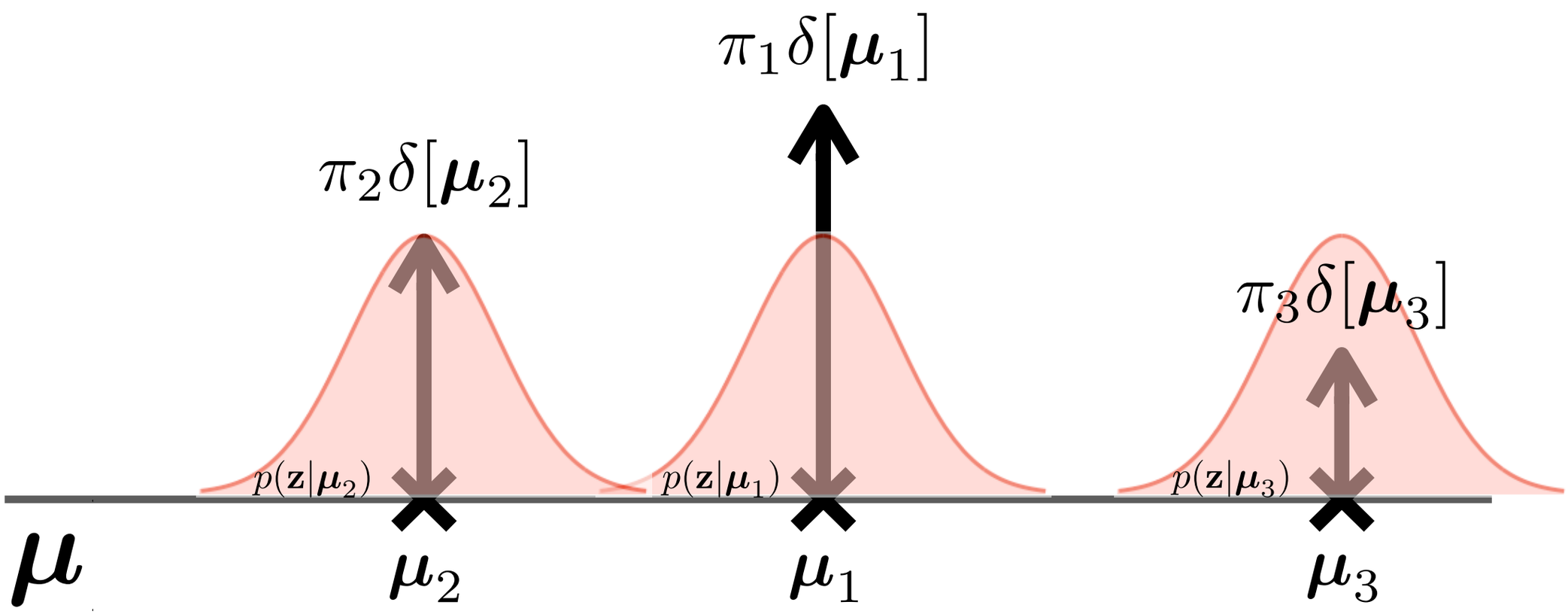
The Dirichlet Process Mixture Model

$$\mathbf{z} \sim \mathrm{DPMM}(p, \alpha, H)$$



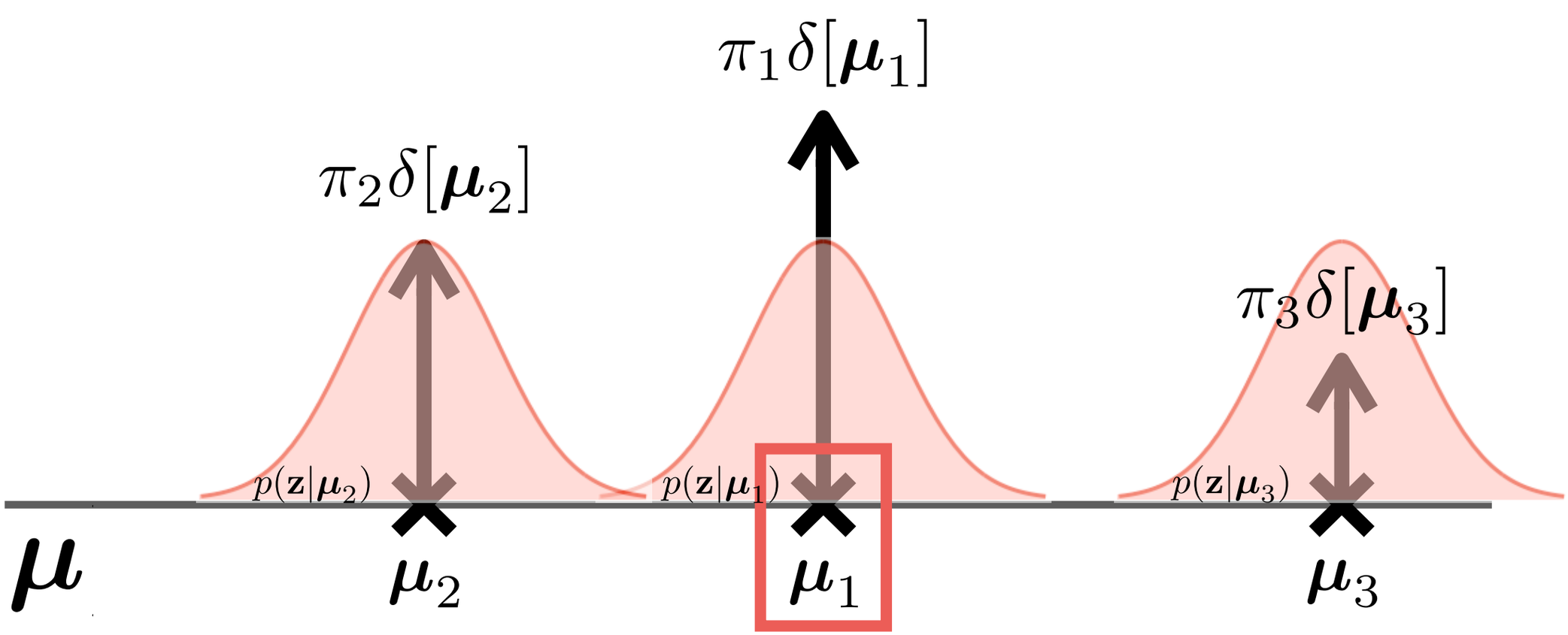
The Dirichlet Process Mixture Model

$$\mathbf{z} \sim \mathrm{DPMM}(p, \alpha, H)$$



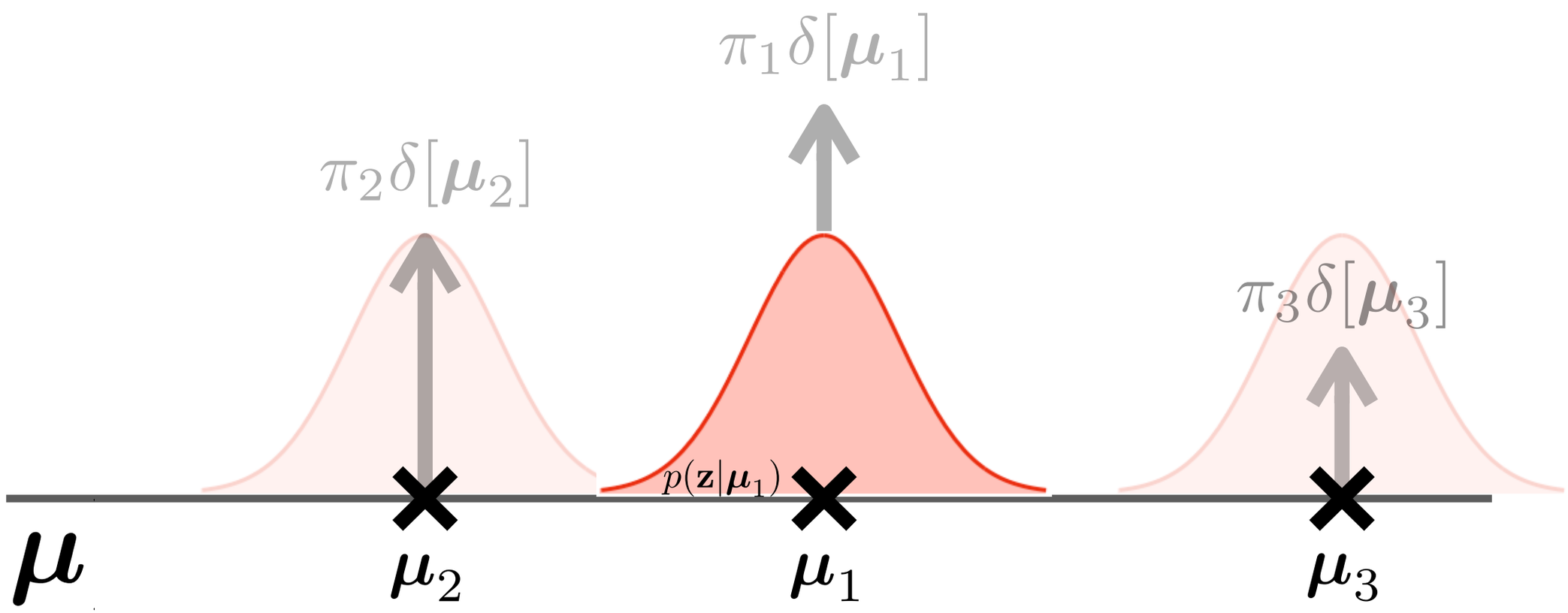
The Dirichlet Process Mixture Model

$$\mathbf{z} \sim \mathrm{DPMM}(p, \alpha, H)$$



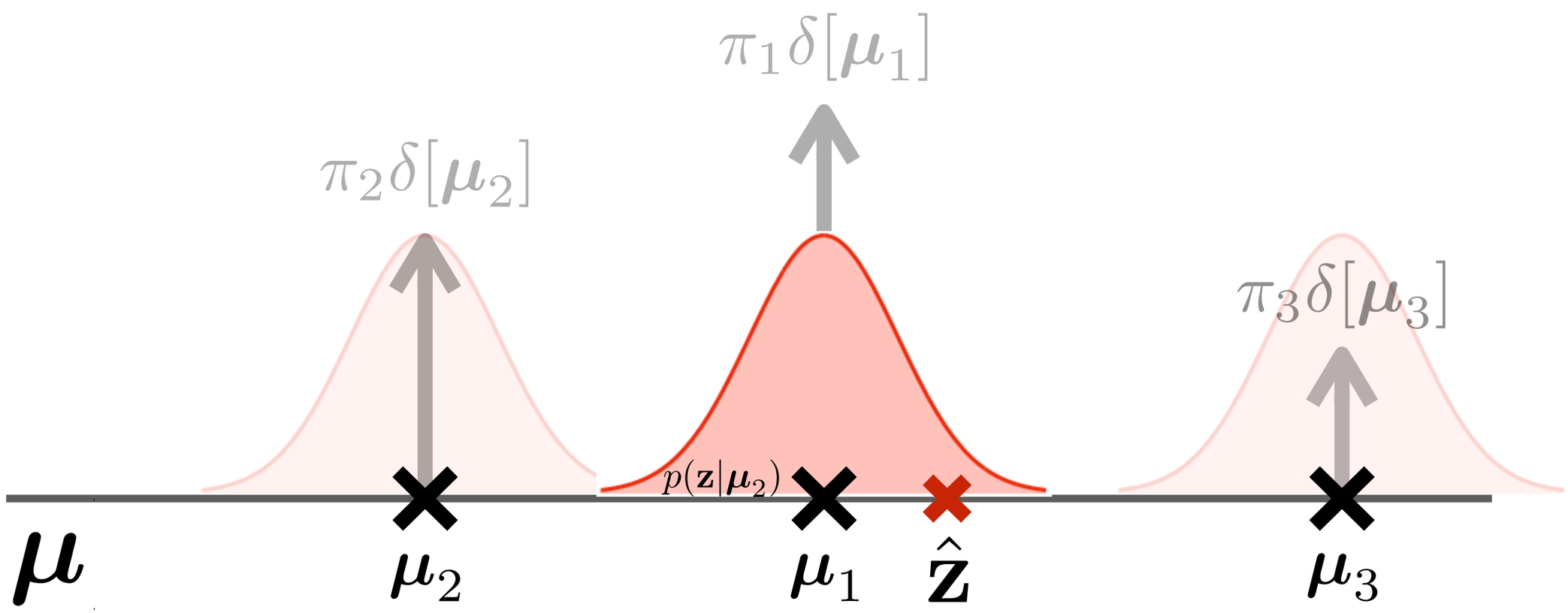
The Dirichlet Process Mixture Model

$$\mathbf{z} \sim \mathrm{DPMM}(p, \alpha, H)$$



The Dirichlet Process Mixture Model

$$\mathbf{z} \sim \mathrm{DPMM}(p, \alpha, H)$$



Nonparametric Density Networks

Nonparametric Density Networks

STICK-BREAKING
DENSITY NETWORK

$$\boldsymbol{\pi}_i \sim \mathrm{GEM}(\alpha)$$

$$\mathbf{x}_i \sim p\left(\mathbf{x}_i | \boldsymbol{\pi}_i; \{\mathbf{W}\}_{l=1}^{L+1}\right)$$

Adaptive Width: Due to order bias, can think of the network as having adaptive width, using enough latent variables as the data needs.

Nonparametric Density Networks

STICK-BREAKING DENSITY NETWORK

$$\boldsymbol{\pi}_i \sim \mathrm{GEM}(\alpha)$$

$$\mathbf{x}_i \sim p(\mathbf{x}_i|\boldsymbol{\pi}_i; \{\mathbf{W}\}_{l=1}^{L+1})$$

Adaptive Width: Due to order bias, can think of the network as having adaptive width, using enough latent variables as the data needs.

LATENT DIRICHLET PROCESS DENSITY NETWORK

$$\mathbf{z}_i \sim \mathrm{DPMM}(N(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \alpha, N(\boldsymbol{\mu}))$$

$$\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{z}_i; \{\mathbf{W}\}_{l=1}^{L+1})$$

Adaptive Complexity: Can draw the latent representation from as complex a distribution as needed.

Amortized Variational Inference

Poondi Kumaraswamy
(1930-1988)

Gradient-Based Inference: Want to use an **inference model** to predict the latent variables and thus need to take gradients through samples. The Beta distribution does not allow this so we use the Beta-like **Kumaraswamy distribution** as the posterior on stick segments

Amortized Variational Inference

Poondi Kumaraswamy
(1930-1988)

Gradient-Based Inference: Want to use an **inference model** to predict the latent variables and thus need to take gradients through samples. The Beta distribution does not allow this so we use the Beta-like **Kumaraswamy distribution** as the posterior on stick segments

STICK-BREAKING VAE

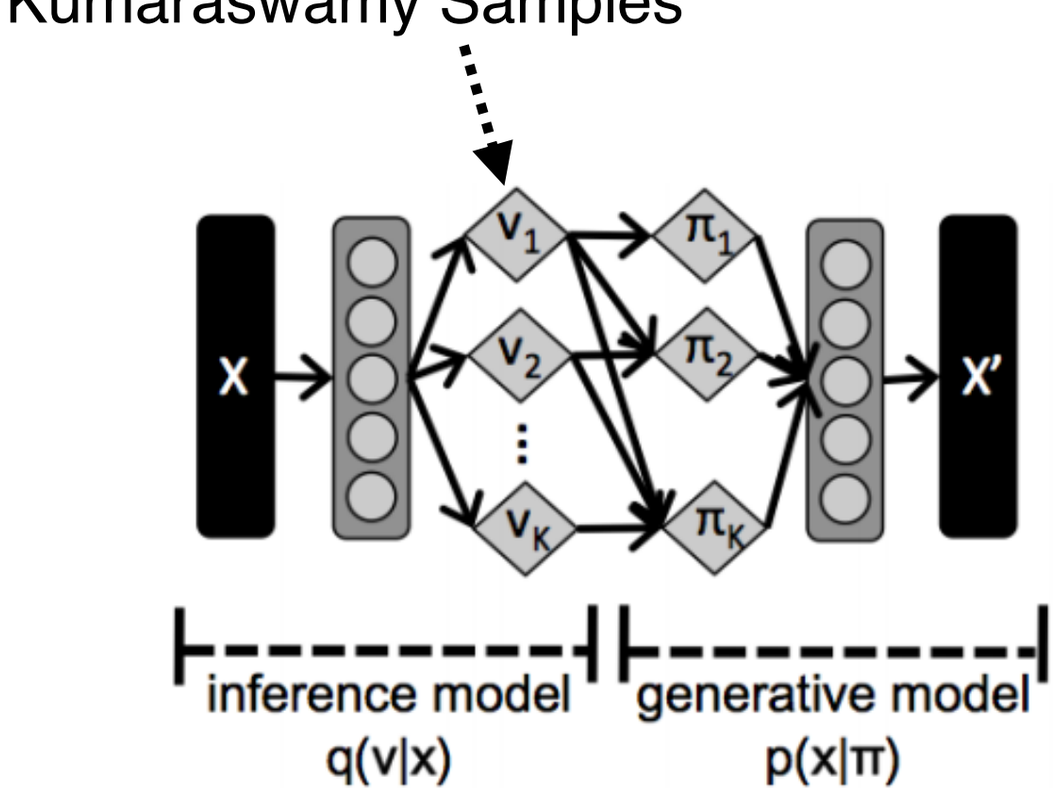
Truncated posterior;
not necessary but learns faster

Amortized Variational Inference

Poondi Kumaraswamy
(1930-1988)

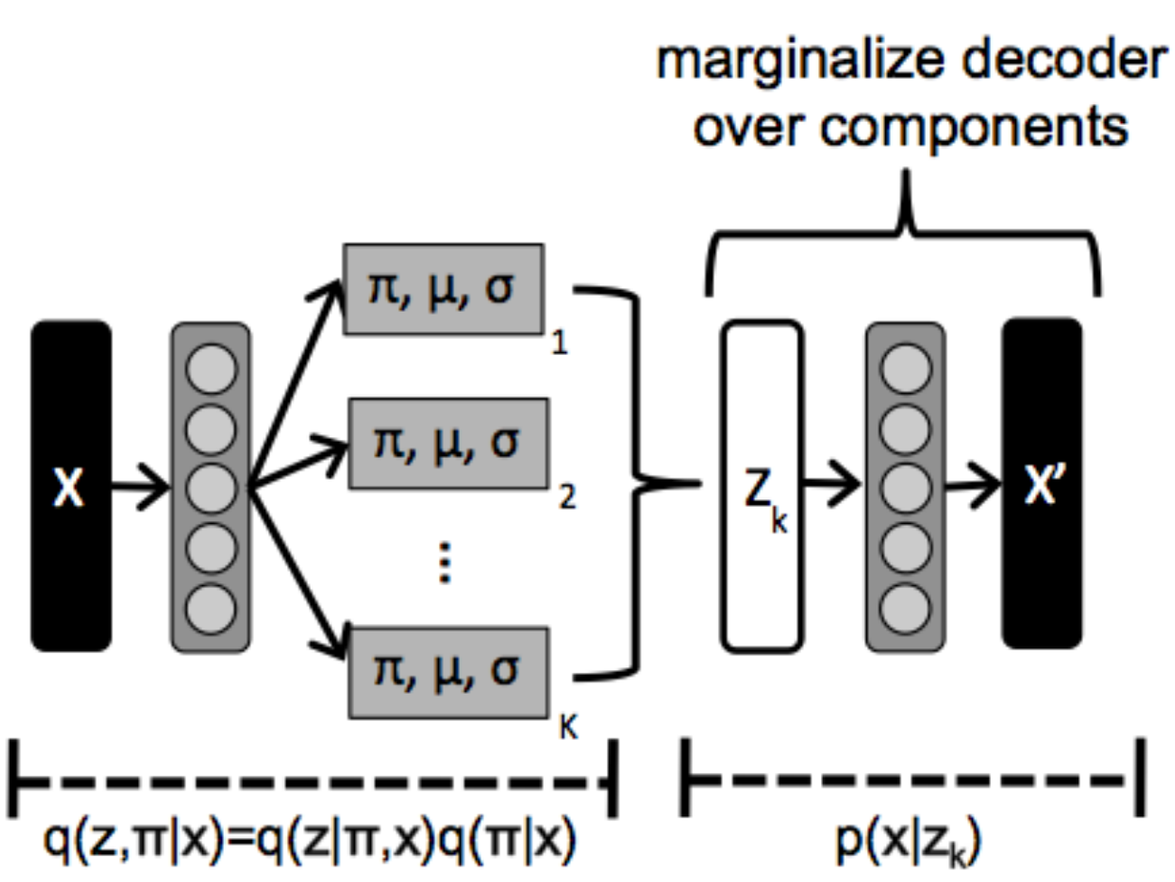
Gradient-Based Inference: Want to use an **inference model** to predict the latent variables and thus need to take gradients through samples. The Beta distribution does not allow this so we use the Beta-like **Kumaraswamy distribution** as the posterior on stick segments

STICK-BREAKING VAE



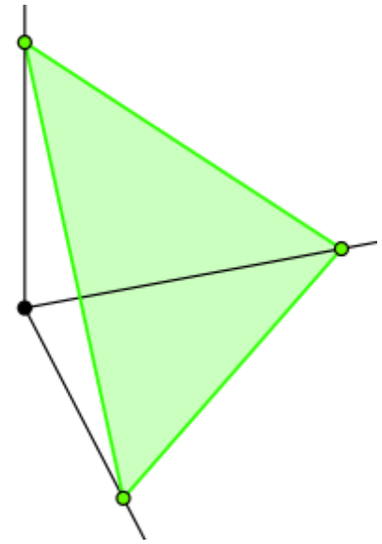
Truncated posterior;
not necessary but learns faster

LATENT DIRICHLET PROCESS VAE

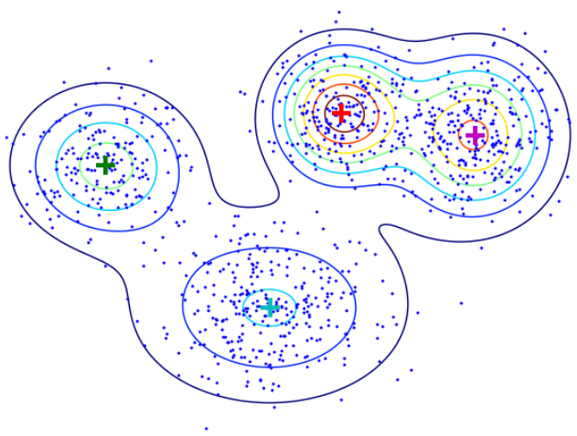


Results: MNIST Samples

STICK-BREAKING VAE

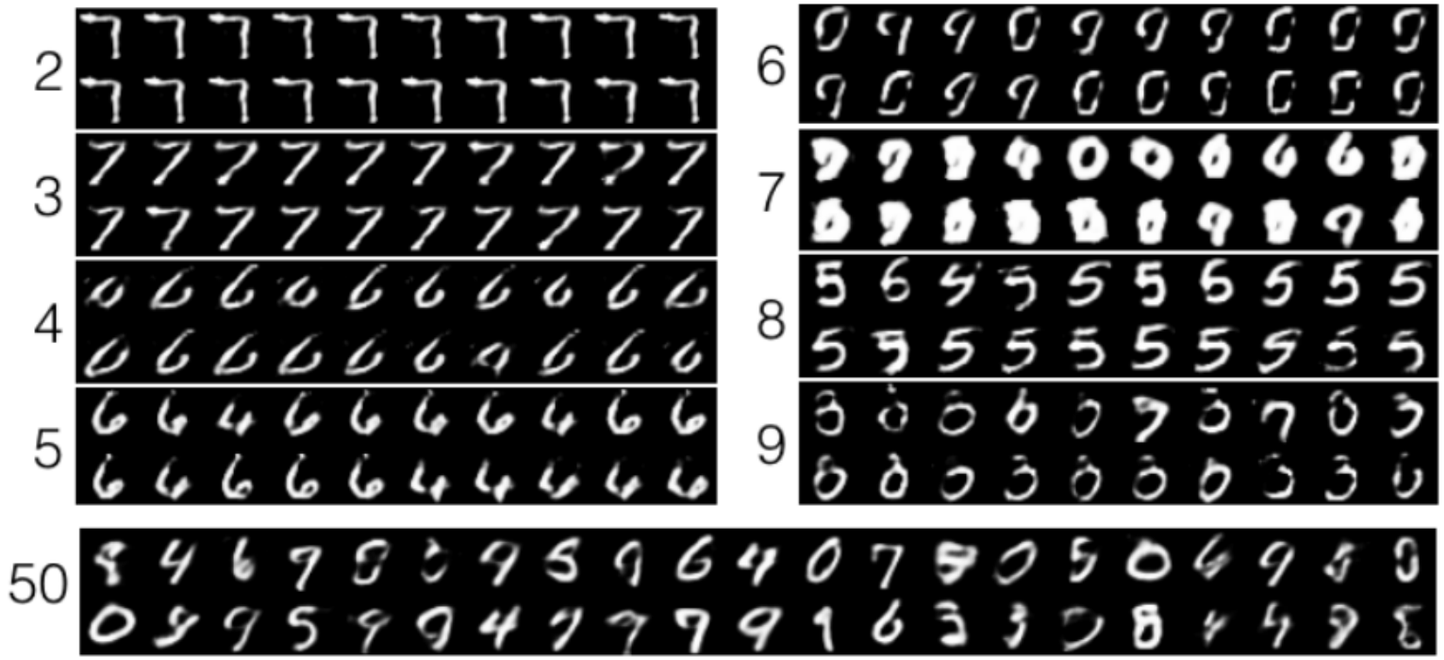


DPMM VAE



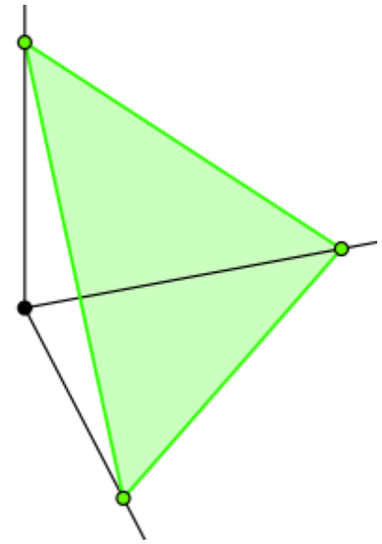
Results: MNIST Samples

DIMENSIONALITY

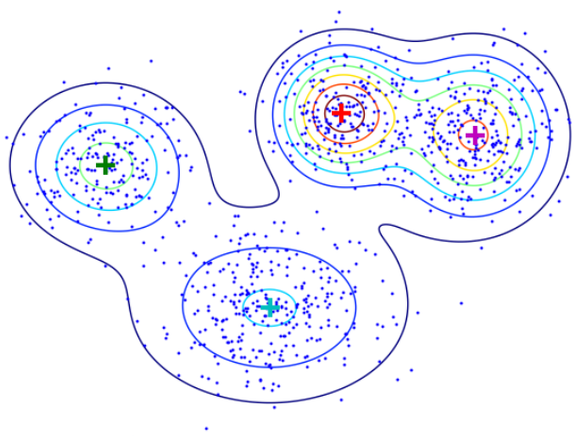


Truncation level of 50. Beta(1,5) prior.

STICK-BREAKING VAE

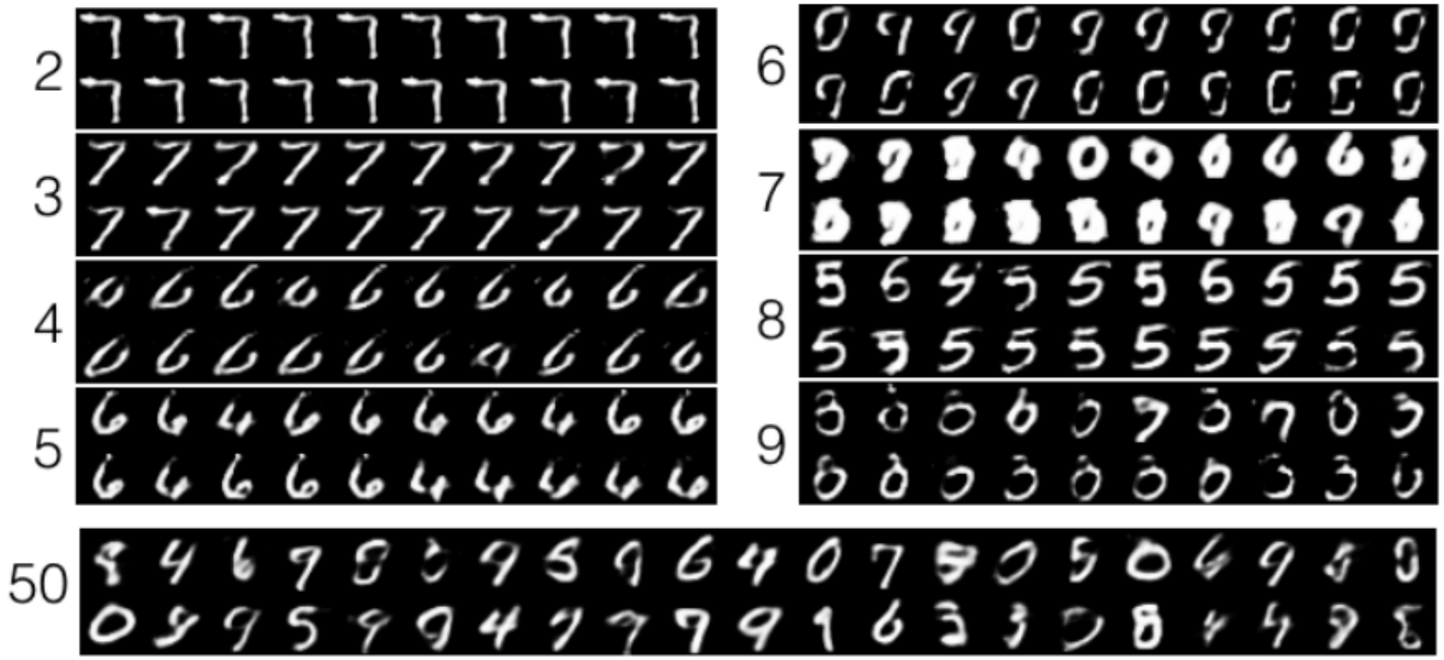


DPMM VAE

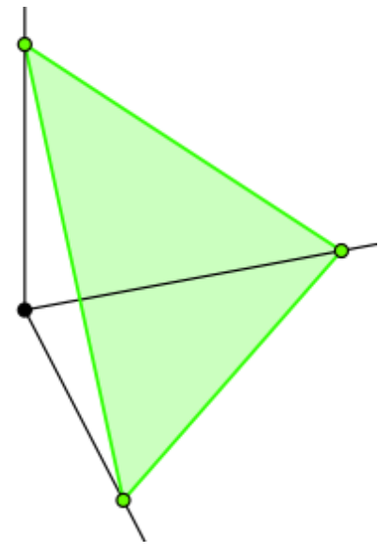


Results: MNIST Samples

DIMENSIONALITY

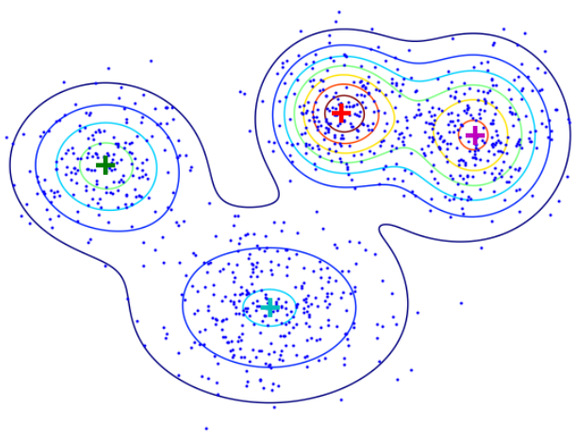


STICK-BREAKING VAE



Truncation level of 50. Beta(1,5) prior.

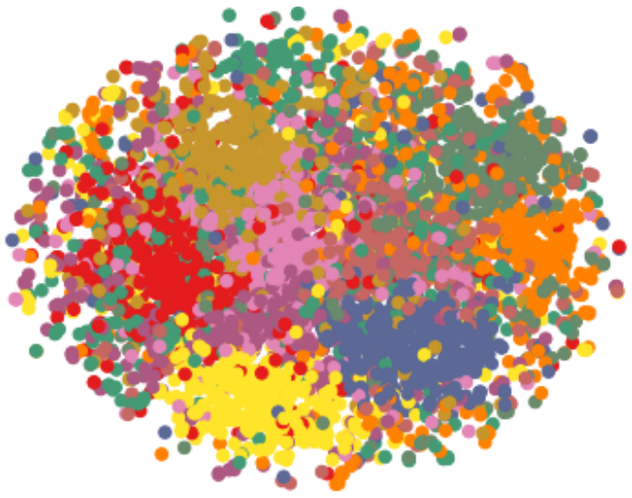
DPMM VAE



Samples from Component #1

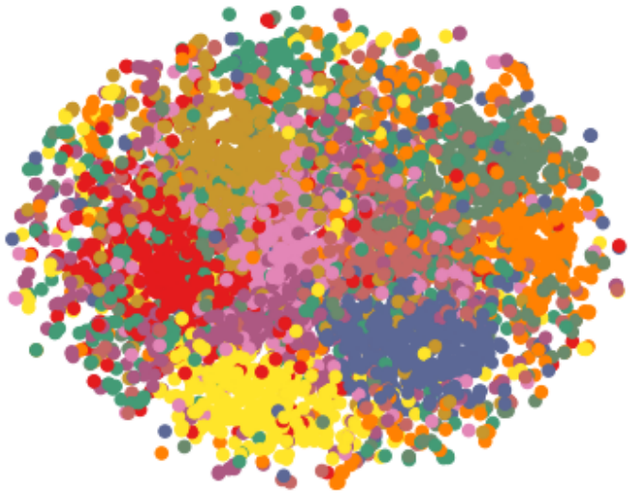
Samples from Component #5

Results: MNIST Latent Space



Gaussian Latent Space (t-SNE)

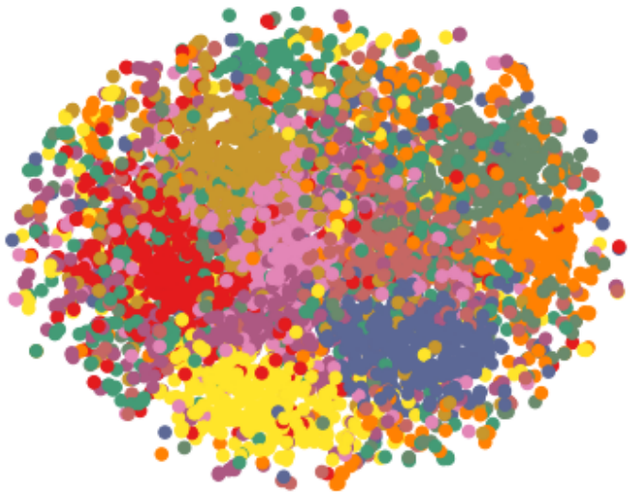
Results: MNIST Latent Space



Gaussian Latent Space (t-SNE)

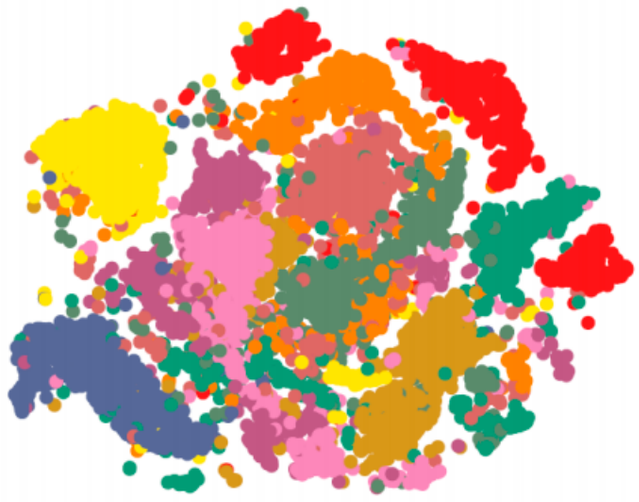
GEM Latent Space (t-SNE)

Results: MNIST Latent Space



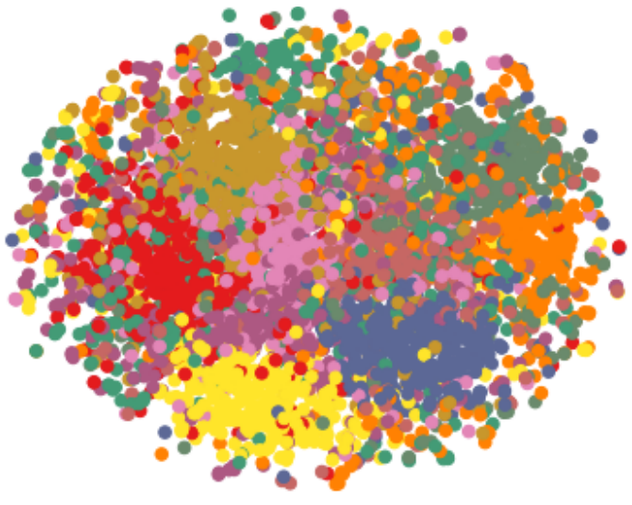
Gaussian Latent Space (t-SNE)

GEM Latent Space (t-SNE)

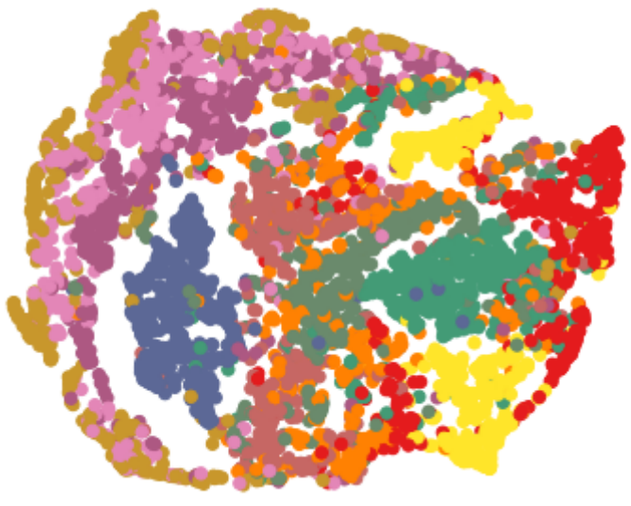


DPMM Latent Space (t-SNE)

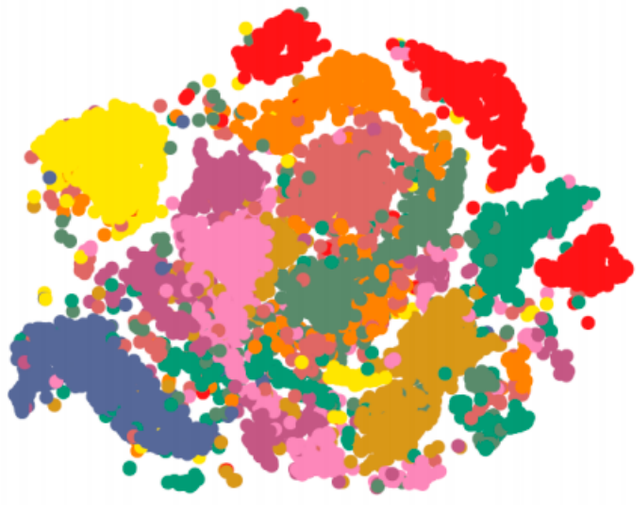
Results: MNIST Latent Space



Gaussian Latent Space (t-SNE)



GEM Latent Space (t-SNE)

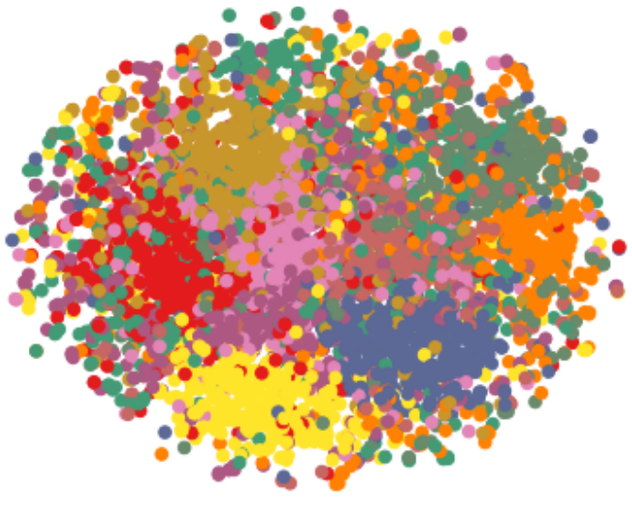


DPMM Latent Space (t-SNE)

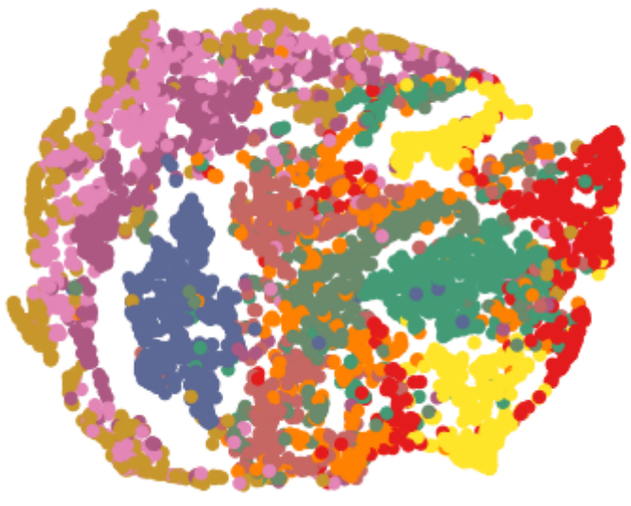
	k=3	k=5	k=10
SB-VAE	9.34	8.65	8.90
DLGMM	9.14	8.38	8.42
Gauss VAE	28.4	20.96	15.33
GMVAE[6]	—	8.96	—

Error Rate of k-Nearest Neighbor Classifier on Latent Space

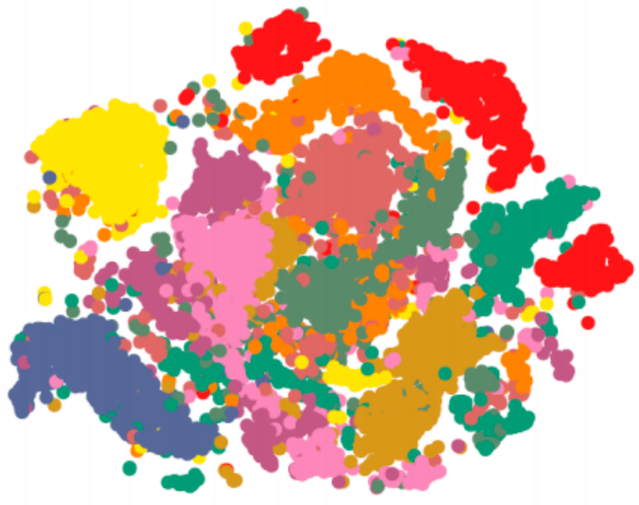
Results: MNIST Latent Space



Gaussian Latent Space (t-SNE)



GEM Latent Space (t-SNE)



DPMM Latent Space (t-SNE)

	k=3	k=5	k=10
SB-VAE	9.34	8.65	8.90
DLGMM	9.14	8.38	8.42
Gauss VAE	28.4	20.96	15.33
GMVAE[6]	—	8.96	—

Error Rate of k-Nearest Neighbor Classifier on Latent Space

Results: Likelihood and Semi-Supervised

HELD-OUT LIKELIHOOD

SEMI-SUPERVISED CLASSIFICATION

Results: Likelihood and Semi-Supervised

HELD-OUT LIKELIHOOD

	$-\log p_{\boldsymbol{\theta}}(\mathbf{x}_i)$	
	MNIST	OMNIGLOT
DLGMM (500d-3x25s)	**96.50**	123.50
DLDPMM (500d-17tx25s)	96.91	123.76
Gauss-VAE (500d-25s)	96.80	**119.18**
SB-VAE (500d-25t)	98.01	–

Negative log likelihood on test set (importance sampled)

SEMI-SUPERVISED CLASSIFICATION

Results: Likelihood and Semi-Supervised

HELD-OUT LIKELIHOOD

	$-\log p_{\boldsymbol{\theta}}(\mathbf{x}_i)$	
	MNIST	OMNIGLOT
DLGMM (500d-3x25s)	**96.50**	123.50
DLDPMM (500d-17tx25s)	96.91	123.76
Gauss-VAE (500d-25s)	96.80	**119.18**
SB-VAE (500d-25t)	98.01	–

Negative log likelihood on test set (importance sampled)

SEMI-SUPERVISED CLASSIFICATION

	MNIST (N=45,000)			**SVHN** (N=65,000)		
	10%	5%	1%	10%	5%	1%
SB-DGM	4.86$_{\pm.14}$	5.29$_{\pm.39}$	**7.34**$_{\pm.47}$	**32.08**$_{\pm4.00}$	**37.07**$_{\pm5.22}$	**61.37**$_{\pm3.60}$
Gauss-DGM	**3.95**$_{\pm.15}$	**4.74**$_{\pm.43}$	11.55$_{\pm2.28}$	36.08$_{\pm1.49}$	48.75$_{\pm1.47}$	69.58$_{\pm1.64}$
kNN	6.13$_{\pm.13}$	7.66$_{\pm.10}$	15.27$_{\pm.76}$	64.81$_{\pm.34}$	68.94$_{\pm.47}$	76.64$_{\pm.54}$

Error rate on test set when training with X% labeled.

Nonparametric version of (Kingma et al., NIPS 2014)'s M2 model

Priors for Neural Networks

CONDITIONAL MODEL: p(**W**)

NONPARAMETRIC

GP Equivalence [Neal, 1994]
ARD [Neal, 1994; MacKay, 1994]

INDUCTIVE BIAS

SUBJECTIVE BY PROXY

OBJECTIVE

OBJECTIVE

SUBJECTIVE

OBJECTIVE

INDUCTIVE BIAS

NONPARAMETRIC

Stick-Breaking [Nalisnick & Smyth, 2017]
DP-Mixture [Nalisnick et al., 2016]

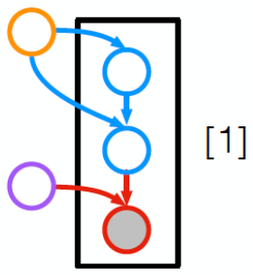
SUBJECTIVE

DENSITY NETWORK: p(**z**)

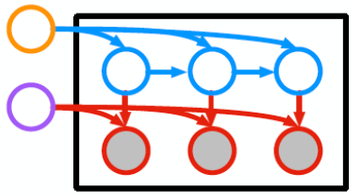
Future Directions and Summary

Future Directions

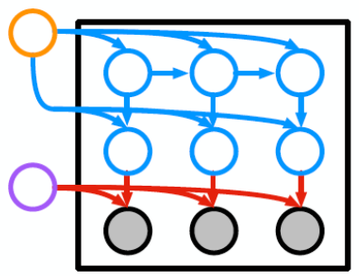
- **Structured Priors / Models:** [Johnson et al., 2016] demonstrated how to combine density networks and latent graphical models.


[1]

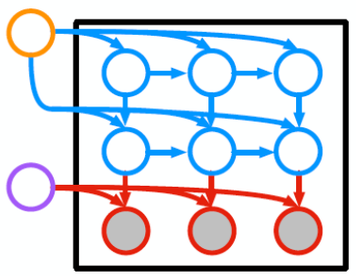
Gaussian mixture model


[2]

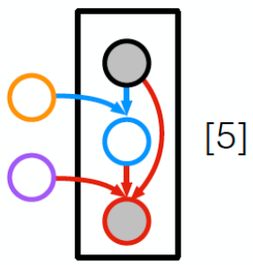
Linear dynamical system


[3]

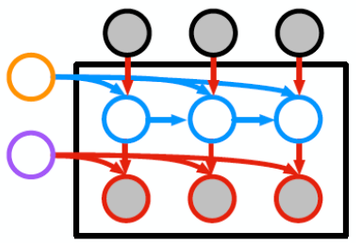
Hidden Markov model


[4]

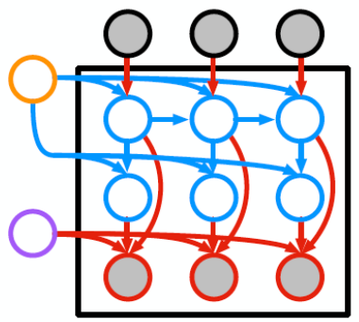
Switching LDS


[5]

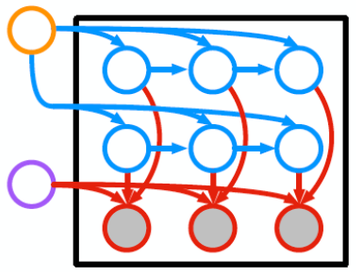
Mixture of Experts


[2]

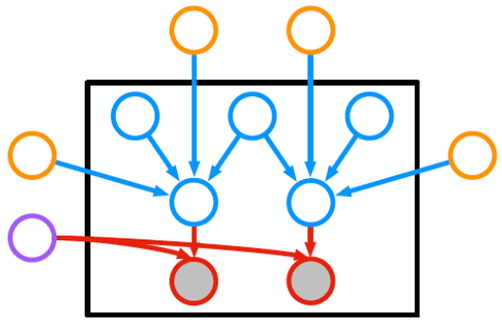
Driven LDS


[6]

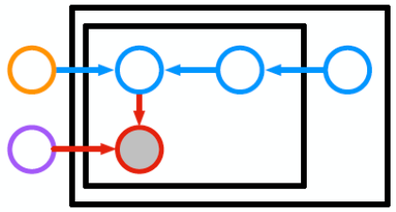
IO-HMM


[7]

Factorial HMM


[8,9]

Canonical correlations analysis


[10]

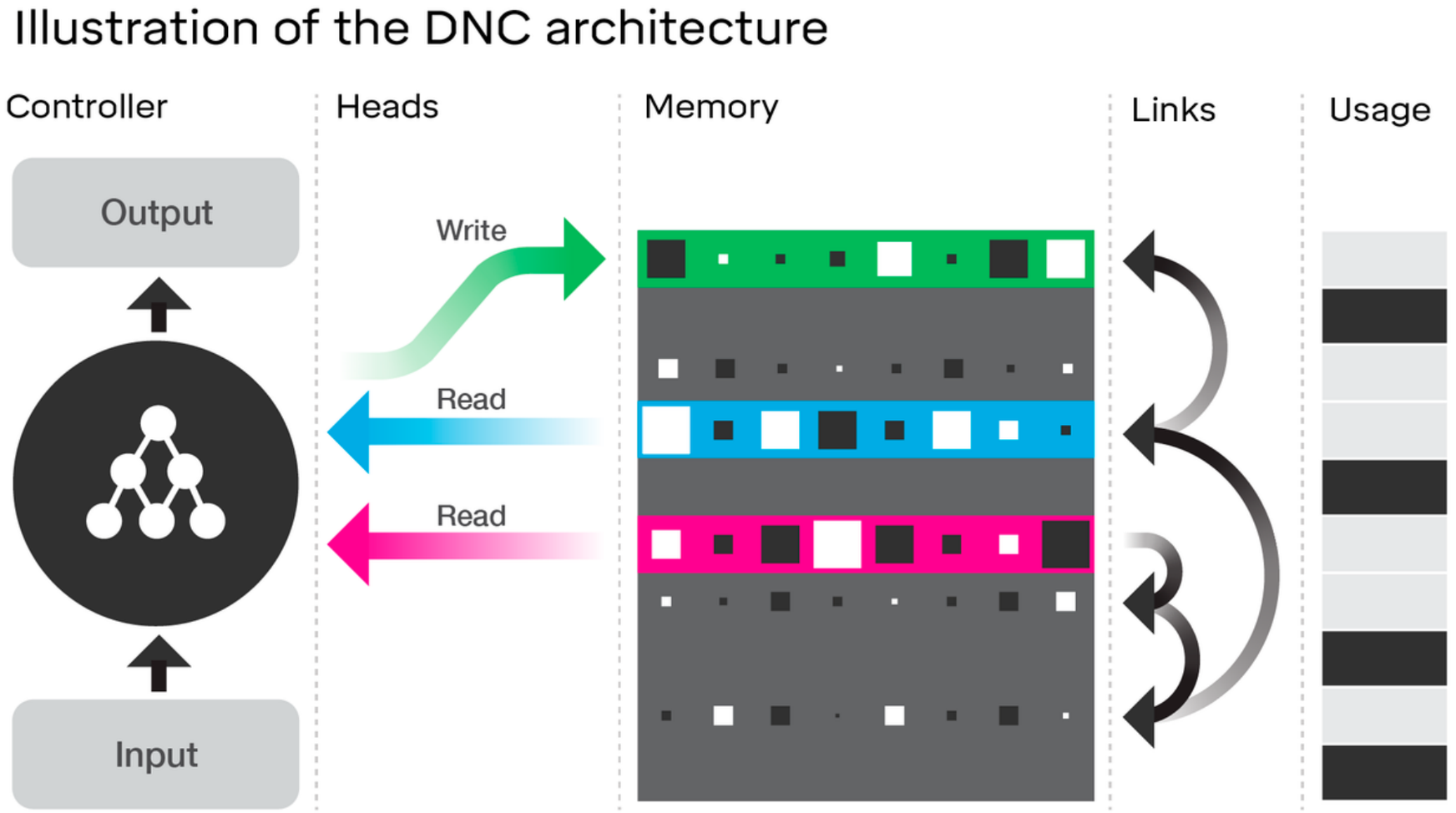
admixture / LDA / NMF


= Neural Network

From M. Johnson's DL summer school slides

Future Directions

- ‘**Neural Nonparametrics’:** Recent deep learning innovations such as *differentiable memory* [Graves et al., 2014] and *adaptive computation* [Graves, 2016] have the character of nonparametrics. I think they can be formalized with traditional BNP priors.



Summary

- [ ] **Bayesian neural networks** have the potential to quantify uncertainty, learn in sequential settings, and ease NN's hunger for data.

- [ ] **Choice of prior is important:** They endow the model with characteristics such as the ability to follow the data and grow in complexity.

- [ ] **Contributions presented:** Extended and explored *reference priors* and *nonparametric priors* for density networks.

Many Thanks to…

MY COMMITTEE



MY RESEARCH GROUP

UCI DATALAB

BREN:ICS
INFORMATION AND COMPUTER SCIENCES
UCIRVINE | UNIVERSITY of CALIFORNIA

Appendix

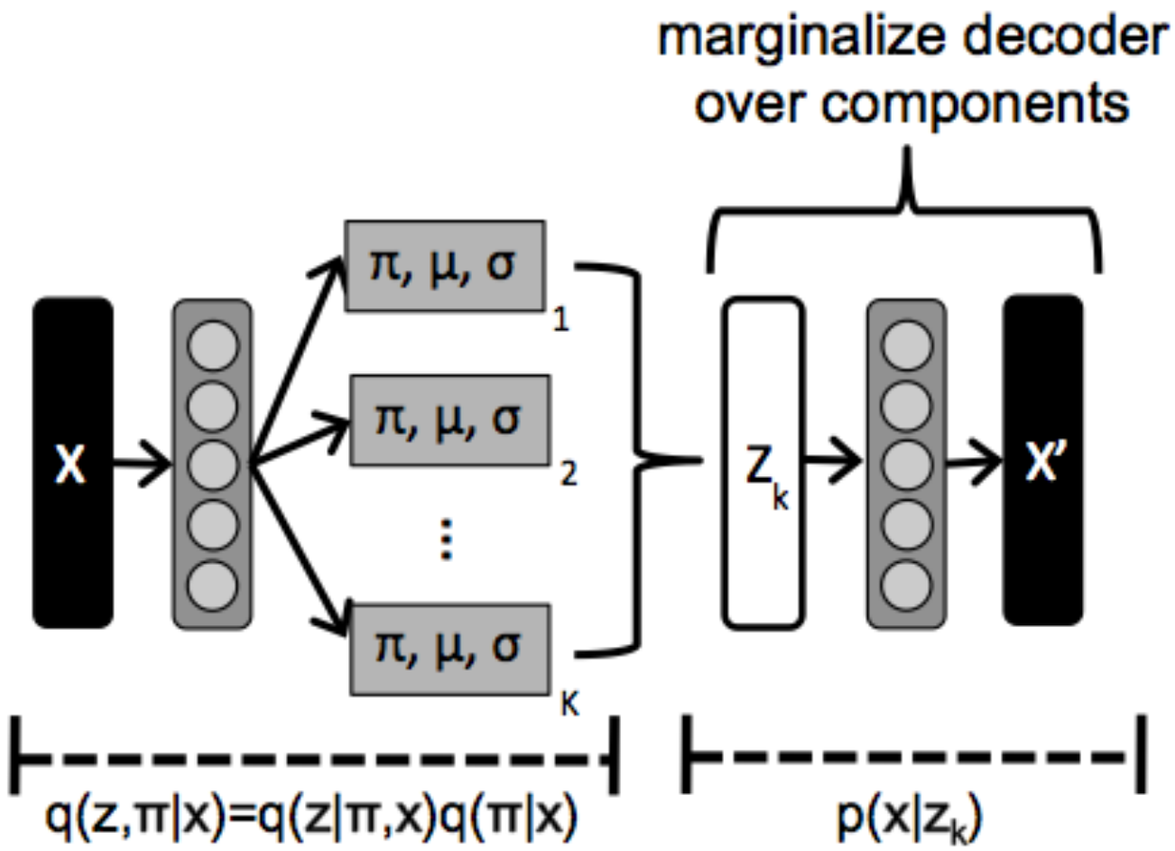
Approximating Reference Priors

Can re-write the Reference prior objective as:

$$
\begin{aligned}
p^*(\boldsymbol{\theta}) &= \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}} \int_{\mathcal{D}} p(\mathcal{D}) \int_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathcal{D}) \log \frac{p(\boldsymbol{\theta}|\mathcal{D})}{p(\boldsymbol{\theta})} d\boldsymbol{\theta} d\mathcal{D} \\
&= \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}} \int_{\boldsymbol{\theta}} p(\boldsymbol{\theta}) \int_{\mathcal{D}} p(\mathcal{D}|\boldsymbol{\theta}) \log \frac{p(\mathcal{D}|\boldsymbol{\theta})}{p(\mathcal{D})} d\mathcal{D} d\boldsymbol{\theta} \\
&= \underset{p(\boldsymbol{\theta})}{\operatorname{argmax}} \mathbb{E}_{p(\boldsymbol{\theta})} KLD[p(\mathcal{D}|\boldsymbol{\theta}) \,||\, p(\mathcal{D})].
\end{aligned}
$$

MODEL #2: Dirichlet Process Variational Autoencoder

(Nalisnick et al., 2016)



$$\mathcal{L}_{\text{SGVB}} = \sum_k \mu_{\pi_k} [\frac{1}{S} \sum_s \log p_{\boldsymbol{\theta}}(\mathbf{x}_i | \hat{\mathbf{z}}_{i,k,s}) + \mathbb{E}_{q_k}[\log p(\mathbf{z}_i)]]$$

$$- \text{KLD}[q(\boldsymbol{\pi}_k|\mathbf{x}_i)||p(\boldsymbol{\pi}_k)] - \frac{1}{S} \sum_s \log \sum_k \hat{\pi}_{i,k,s} q(\hat{\mathbf{z}}_{i,k,s}; \boldsymbol{\phi}_k)$$