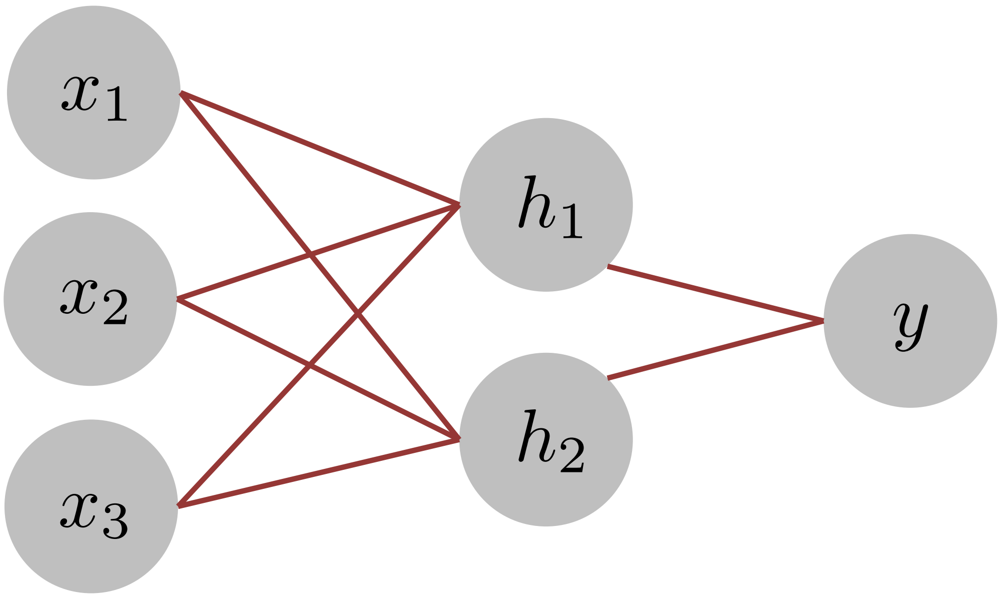
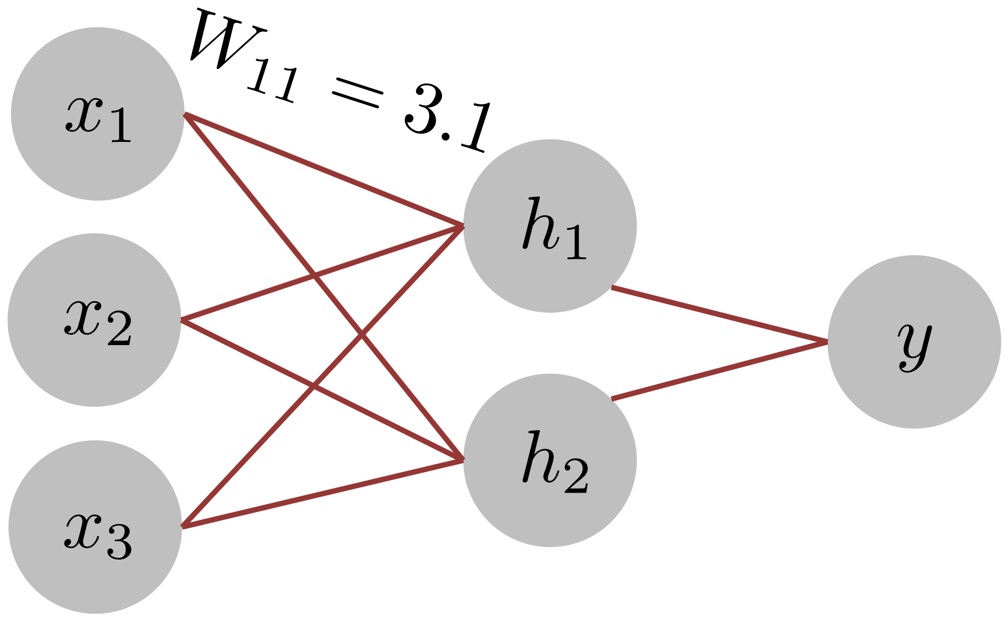
Bayesian Neural Networks

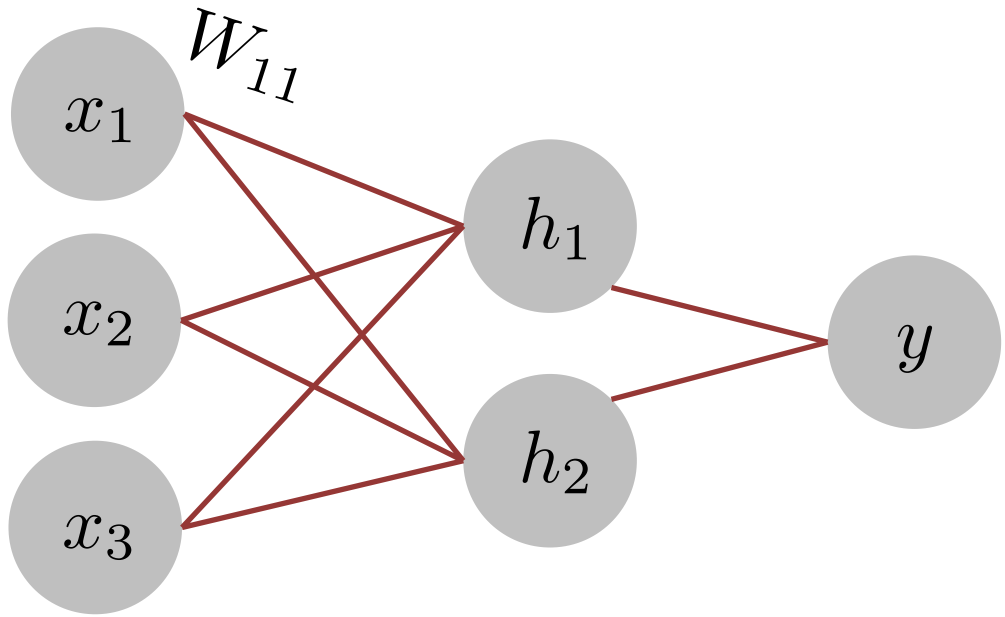
Bayesian Neural Networks

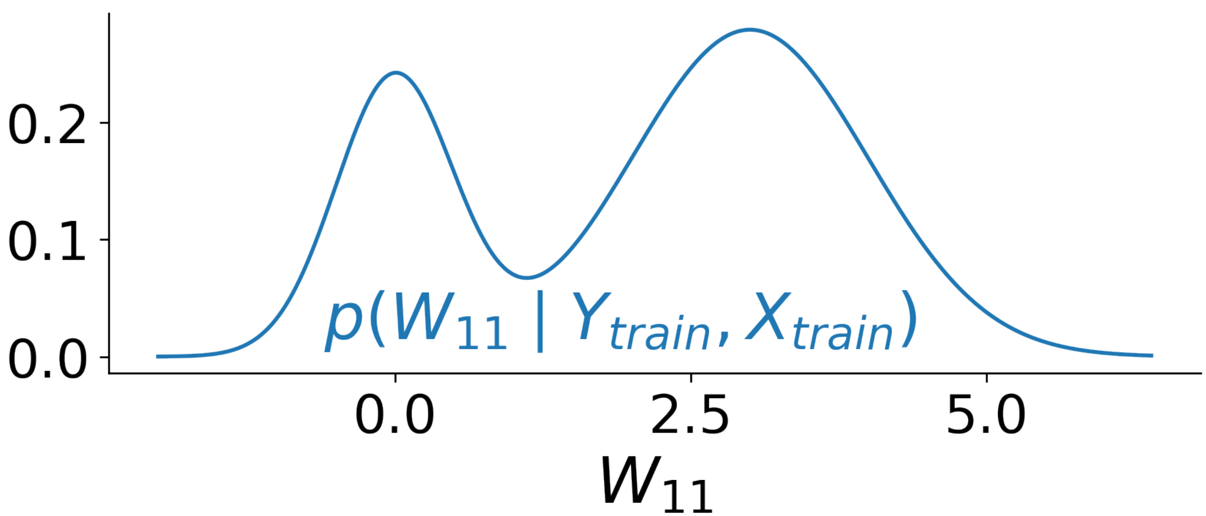


Bayesian Neural Networks



Bayesian Neural Networks





Bayesian Neural Networks

$$p(y \mid x, Y_{\mathrm{train}}, X_{\mathrm{train}})$$

Bayesian Neural Networks

$$p(y \mid x, Y_{\text{train}}, X_{\text{train}})$$
$$= \int p(y \mid x, w)\, p(w \mid Y_{\text{train}}, X_{\text{train}}) dw$$

Bayesian Neural Networks

$$p(y \mid x, Y_{\text{train}}, X_{\text{train}})$$

$$= \int \underbrace{p(y \mid x, w)}_{\text{NN output}} p(w \mid Y_{\text{train}}, X_{\text{train}}) dw$$

Bayesian Neural Networks

$$
\begin{aligned}
&p(y \mid x, Y_{\text{train}}, X_{\text{train}}) \\
&= \int p(y \mid x, w)\, p(w \mid Y_{\text{train}}, X_{\text{train}}) dw \\
&= \mathbb{E}_{p(w \mid Y_{\text{train}}, X_{\text{train}})} p(y \mid x, w)
\end{aligned}
$$

Bayesian Neural Networks

$$p(y \mid x, Y_{\text{train}}, X_{\text{train}})$$
$$= \int p(y \mid x, w)\, p(w \mid Y_{\text{train}}, X_{\text{train}}) dw$$
$$= \mathbb{E}_{p(w \mid Y_{\text{train}}, X_{\text{train}})} p(y \mid x, w)$$

$$p(w \mid Y_{\text{train}}, X_{\text{train}}) \sim \{\text{Gibbs}\}?$$

Bayesian Neural Networks

$$
\begin{aligned}
&p(y \mid x, Y_{\text{train}}, X_{\text{train}}) \\
&= \int p(y \mid x, w)\, p(w \mid Y_{\text{train}}, X_{\text{train}}) dw \\
&= \mathbb{E}_{p(w \mid Y_{\text{train}}, X_{\text{train}})} p(y \mid x, w)
\end{aligned}
$$

$$
p(w \mid Y_{\text{train}}, X_{\text{train}}) \sim \{\text{Gibbs}\}?
$$

$$
p(w \mid Y_{\text{train}}, X_{\text{train}}) = \frac{p(Y_{\text{train}} \mid X_{\text{train}}, w) p(w)}{Z}
$$

Bayesian Neural Networks

$$p(y \mid x, Y_{\mathrm{train}}, X_{\mathrm{train}})$$

$$= \int p(y \mid x, w)\, p(w \mid Y_{\mathrm{train}}, X_{\mathrm{train}}) dw$$

$$= \mathbb{E}_{p(w \mid Y_{\mathrm{train}}, X_{\mathrm{train}})} p(y \mid x, w)$$

$$p(w \mid Y_{\mathrm{train}}, X_{\mathrm{train}}) \sim \{\mathrm{Gibbs}\}?$$

$$p(w \mid Y_{\mathrm{train}}, X_{\mathrm{train}}) = \frac{p(Y_{\mathrm{train}} \mid X_{\mathrm{train}}, w) p(w)}{Z}$$

Depends on the whole dataset!

Langevin Monte Carlo

Gibbs and Metropolis Hastings can't do mini-batches

Langevin Monte Carlo

Say we want to sample from $p(w \mid D)$

Langevin Monte Carlo

Say we want to sample from $p(w \mid D)$

Start from w^0

Langevin Monte Carlo

Say we want to sample from $p(w \mid D)$

Start from w^0

For k = 1, …

$$w^{k+1} = w^k + \varepsilon \nabla \log p(w^k \mid D) + \eta^k,$$

$$\eta^k \sim \mathcal{N}(0, 2\varepsilon I)$$

Langevin Monte Carlo

Say we want to sample from $p(w \mid D)$

Start from w^0

For k = 1, …

$$w^{k+1} = \underbrace{\boxed{w^k + \varepsilon \nabla \log p(w^k \mid D)}}_{\text{Gradient ascent}} + \eta^k,$$

$$\eta^k \sim \mathcal{N}(0, 2\varepsilon I)$$

Langevin Monte Carlo

Say we want to sample from $p(w \mid D)$

Start from w^0

For k = 1, …

$$w^{k+1} = w^k + \varepsilon \nabla \log p(w^k \mid D) + \eta^k,$$

$$= w^k + \varepsilon \nabla \left(\log p(w^k) + \sum_{i=1}^{N} \log p(y_i \mid x_i, w^k) \right) + \eta^k$$

$$\eta^k \sim \mathcal{N}(0, 2\varepsilon I)$$

Langevin Monte Carlo

Say we want to sample from $p(w \mid D)$

Start from w^0

For k = 1, …

$$w^{k+1} = w^k + \varepsilon \nabla \log p(w^k \mid D) + \eta^k,$$

$$= w^k + \varepsilon \nabla \left(\log p(w^k) + \sum_{i=1}^{N} \log p(y_i \mid x_i, w^k) \right) + \eta^k$$

Weight decay $-C\|w^k\|^2$

Usual cross entropy

$$\eta^k \sim \mathcal{N}(0, 2\varepsilon I)$$

Langevin Monte Carlo

- Initialize weights w^0

Langevin Monte Carlo

- Initialize weights w^0

- Do say 100 iterations with usual SGD, but add Gaussian noise $\eta^k \sim \mathcal{N}(0, 2\varepsilon I)$ to each update

Langevin Monte Carlo

- Initialize weights w^0
- Do say 100 iterations with usual SGD, but add Gaussian noise $\eta^k \sim \mathcal{N}(0, 2\varepsilon I)$ to each update
- After 100 hundred epochs decide that Markov Chain converged and start collecting weights values

Langevin Monte Carlo

- Initialize weights w^0
- Do say 100 iterations with usual SGD, but add Gaussian noise $\eta^k \sim \mathcal{N}(0, 2\varepsilon I)$ to each update
- After 100 hundred epochs decide that Markov Chain converged and start collecting weights values
- For a new object predict compute average prediction of CNNs with weights $w^{100}, w^{101}, \ldots, w^{200}$

Langevin Monte Carlo

- Initialize weights w^0

- Do say 100 iterations with usual SGD, but add Gaussian noise $\eta^k \sim \mathcal{N}(0, 2\varepsilon I)$ to each update

- After 100 hundred epochs decide that Markov Chain converged and start collecting weights values

- ~~For a new object predict compute average prediction of CNNs with weights~~ $\cancel{w^{100}, w^{101}, \dots, w^{200}}$

- Train another CNN to mimic the ensemble [Balan, Anoop Korattikara, et al. "Bayesian dark knowledge." *Advances in Neural Information Processing Systems*. 2015.]