

Sorbonne Université, Computer Science Master Données, Apprentissage et Connaissances (DAC) Bayesian Deep Learning

Nicolas Thome
Sorbonne Université
ISIR Lab - Machine Learning Team (MLIA)



Outline

Beyond Bayesian Linear Regression

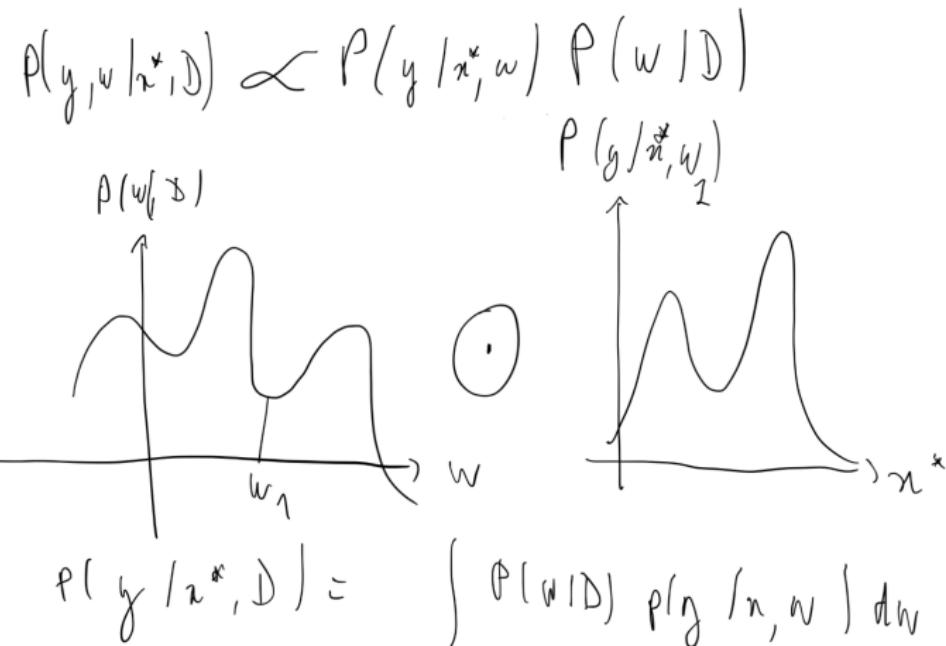
Bayesian Logistic Regression

Bayesian Neural Networks

Monte Carlo Dropout

Beyond Bayesian Linear Regression

- Posterior distribution for parameters w : $p(w|X, Y) \propto p(Y|X, w)p(w)$
- Predictive distribution $p(y|x^*, \mathcal{D}) = \int p(y|x^*, w)p(w|\mathcal{D})dw$, $(X, Y) := \mathcal{D}$



Beyond Bayesian Linear Regression

- Posterior distribution for parameters w : $p(w|X, Y) \propto p(Y|X, w)p(w)$
- Predictive distribution $p(y^*|x^*, \mathcal{D}) = \int p(y^*|x^*, w)p(w|\mathcal{D})dw$, $(X, Y) := \mathcal{D}$
- **Closed form for posterior $p(w|\mathcal{D})$ and predictive distribution $p(y^*|x^*, \mathcal{D})$: more the exception than the rule!**
- Slightly more complicated models : no closed form solution
 - ▶ Bayesian Logistic Regression
 - ▶ Simplest linear classification model
 - ▶ Likelihood not Gaussian
 - ▶ Neural network with one hidden layer in general
 - ▶ No closed form for regression and classification
 - ▶ And of course deep neural networks

Approximate Inference

No analytical expression for posterior $p(w|\mathcal{D})$ and $p(y^*|x^*, \mathcal{D})$ in general

⇒ Approximation needed!

- Gaussian approximation for $p(w|\mathcal{D})$
 - ▶ Ex: Laplace approximation [MacKay, 1992]
 - ▶ Historically used for bayesian logistic regression
- Monte Carlo methods: sampling to directly evaluate integral $p(y^*|x^*, \mathcal{D})$
 - ▶ Metropolis-Hastings, Hamiltonian Monte Carlo [Neal, 1996], Expectation propagation [Hernandez-Lobato and Adams, 2015, Jyläniemi et al., 2014]
- Variational inference [Hinton and van Camp, 1993, Graves, 2011, Blundell et al., 2015]: convert integration into optimization
 - ▶ Minimize KL divergence between $p(w|\mathcal{D})$ and a proposed parametric function

Outline

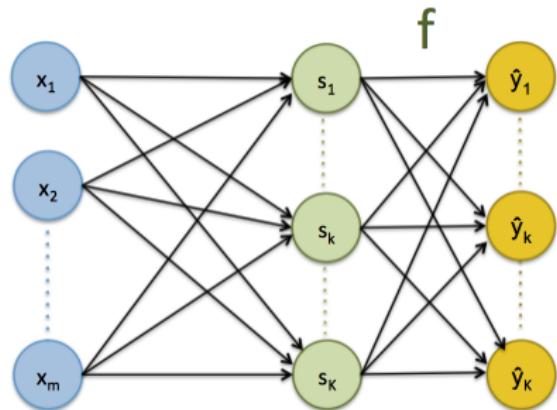
Beyond Bayesian Linear Regression

Bayesian Logistic Regression

Bayesian Neural Networks

Monte Carlo Dropout

Bayesian Logistic Regression (BLR)



- $s_i = Wx_i$
- Multi-class: $p(y_i|x_i, w) = \hat{y}_i$
 - ▶ $\hat{y}_{i,k} = \frac{\exp(s_i)}{\sum_k \exp(s_k)}$
- Binary case: $p(y_i = 1|x_i, w) = \sigma(s_i)$
 - ▶ σ sigmoid
 - ▶ $p(y_i = -1|x_i, w) = 1 - \sigma(s_i)$

$$p(w|X, Y) \propto p(Y|X, w)p(w)$$

- $p(Y|X, w) = \prod_{i=1}^N p(y_i = 1|x_i, w)$ not Gaussian anymore!
- ⇒ no closed-form on posterior distribution $p(w|X, Y)!$

Bayesian Logistic Regression training (MAP)

$$\begin{aligned}\mathbf{w}_{\text{MAP}} &= \arg \max_{\mathbf{w}} p(\mathbf{X}, \mathbf{Y} | \mathbf{w}) p(\mathbf{w}) = \arg \max_{\mathbf{w}} \prod_{n=1}^N p(y_n | \mathbf{x}_n, \mathbf{w}) p(\mathbf{w}) \\ &= \arg \min_{\mathbf{w}} \sum_{n=1}^N -\log(p(y_n | \mathbf{x}_n, \mathbf{w})) - \log(p(\mathbf{w}))\end{aligned}$$

- Gaussian prior: $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mathbf{0}, \sigma_0^2 \mathbf{I})$;
- MAP BLR training with binary prediction:

$$\mathbf{w}_{\text{MAP}} = \arg \min_{\mathbf{w}} \sum_{n=1}^N \left(-y_n \log \sigma(\mathbf{w}^T \mathbf{x}_n + b) - (1 - y_n) \log(1 - \sigma(\mathbf{w}^T \mathbf{x}_n + b)) + \frac{1}{2\sigma_0^2} \|\mathbf{w}\|_2^2 \right)$$

- Again: Gaussian prior \Leftrightarrow weight decay

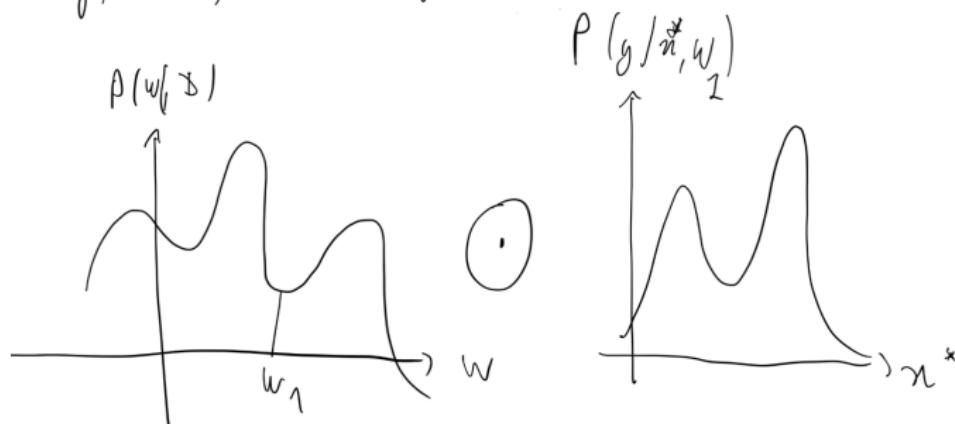
Bayesian Logistic Regression training (MAP)

$$\mathbf{w}_{\text{MAP}} = \arg \min_{\mathbf{w}} \sum_{n=1}^N \left(-y_n \log \sigma(\mathbf{w}^T \mathbf{x}_n + b) - (1-y_n) \log(1-\sigma(\mathbf{w}^T \mathbf{x}_n + b)) + \frac{1}{2\sigma_0^2} \|\mathbf{w}\|_2^2 \right)$$

- \mathbf{w}_{MAP} with gradient descent
- Recap: we want to estimate predictive distribution:

$$p(y=1|\mathbf{x}^*, \mathcal{D}) = \int p(y=1|\mathbf{x}, \mathbf{w}) p(\mathbf{w}|\mathcal{D}) d\mathbf{w}$$

$$p(y, w | \mathcal{D}) \propto p(y | \mathbf{x}^*, w) p(w | \mathcal{D})$$



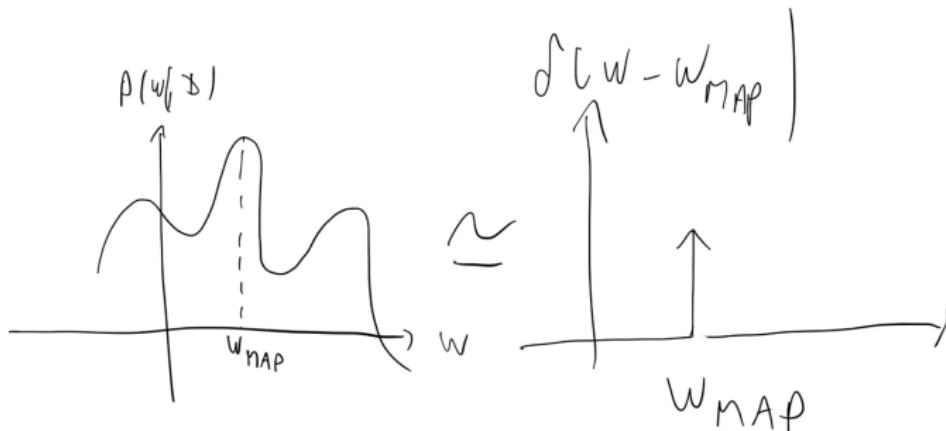
Bayesian Logistic Regression training (MAP)

$$\mathbf{w}_{\text{MAP}} = \arg \min_{\mathbf{w}} \sum_{n=1}^N \left(-y_n \log \sigma(\mathbf{w}^T \mathbf{x}_n + b) - (1-y_n) \log(1-\sigma(\mathbf{w}^T \mathbf{x}_n + b)) + \frac{1}{2\sigma_0^2} \|\mathbf{w}\|_2^2 \right)$$

- \mathbf{w}_{MAP} with gradient descent
- Recap: we want to estimate predictive distribution:

$$p(y=1|\mathbf{x}^*, \mathcal{D}) = \int p(y=1|\mathbf{x}, \mathbf{w}) p(\mathbf{w}|\mathcal{D}) d\mathbf{w}$$

- ▶ Need full posterior distribution $p(\mathbf{w}|\mathbf{X}, \mathbf{Y})$, but posterior intractable
- ▶ $p(\mathbf{w}|\mathbf{X}, \mathbf{Y}) \approx \delta(\mathbf{w} - \mathbf{w}_{\text{MAP}}) \Rightarrow p(y=1|\mathbf{x}^*, \mathcal{D}) \approx p(y=1|\mathbf{x}, \mathbf{w}_{\text{MAP}})$



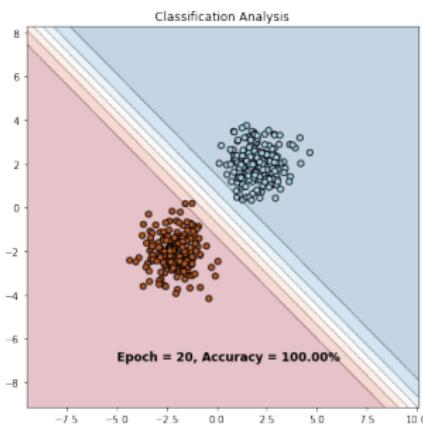
Bayesian Logistic Regression training (MAP)

$$\mathbf{w}_{\text{MAP}} = \arg \min_{\mathbf{w}} \sum_{n=1}^N \left(-y_n \log \sigma(\mathbf{w}^T \mathbf{x}_n + b) - (1-y_n) \log(1-\sigma(\mathbf{w}^T \mathbf{x}_n + b)) + \frac{1}{2\sigma_0^2} \|\mathbf{w}\|_2^2 \right)$$

- \mathbf{w}_{MAP} with gradient descent
- Recap: we want to estimate predictive distribution:

$$p(y=1|\mathbf{x}^*, \mathcal{D}) = \int p(y=1|\mathbf{x}, \mathbf{w}) p(\mathbf{w}|\mathcal{D}) d\mathbf{w}$$

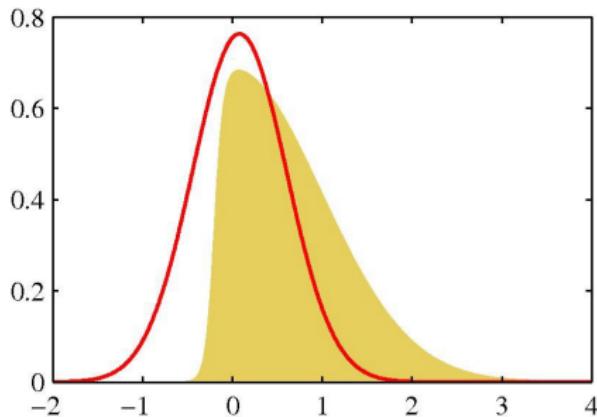
- ▶ Need full posterior distribution $p(\mathbf{w}|\mathbf{X}, \mathbf{Y})$, but posterior intractable
- ▶ $p(\mathbf{w}|\mathbf{X}, \mathbf{Y}) \approx \delta(\mathbf{w} - \mathbf{w}_{\text{MAP}}) \Rightarrow p(y=1|\mathbf{x}^*, \mathcal{D}) \approx p(y=1|\mathbf{x}, \mathbf{w}_{\text{MAP}})$



- $p(\mathbf{w}|\mathbf{X}, \mathbf{Y}) \approx \delta(\mathbf{w} - \mathbf{w}_{\text{MAP}})$: very coarse approximation:
- Uncertainty does not increase far from training data
- Need for more accurate approximations

Laplace Approximation for $p(w|X, Y)$

- Approximate $p(w|X, Y)$ by a normal distribution $q(w) = \mathcal{N}(w; \mu, \Sigma)$
- Fit the mean μ of $q(w)$ to the mode of $p(w|X, Y)$
 - ▶ Mode of $p(w)$ $\Rightarrow \nabla_w p(w) = 0$
 - ▶ In practice, maximize log posterior (e.g. gradient ascent) \Rightarrow MAP: $\mu = w_{MAP}$
- Fit the inverse covariance Σ^{-1} of $q(w)$ to the Hessian of $p(w|X, Y)$ at $\mu = w_{MAP}$: $\Sigma^{-1} = \nabla \nabla_w p(w|X, Y)|_{w=w_{MAP}}$



from [Bishop, 2006]

- Laplace limitation: approximation at a single value of $p(w|X, Y)$, ignores global properties

Predictive Distribution $p(y^*|x^*, \mathcal{D})$ for BLR

RECAP, with $\mathcal{D} = X, Y$:

$$p(y^*|x^*, \mathcal{D}) = \int p(y^*|x^*, w)p(w|\mathcal{D})dw$$

- Posterior approximation by normal $p(w|\mathcal{D}) \approx q(w) = \mathcal{N}(w; \mu, \Sigma)$:

$$p(y^*|x^*, \mathcal{D}) \approx \int p(y^*|x^*, w)q(w)dw$$

- However, likelihood $p(y^*|x^*, \mathcal{D})$ still not Gaussian
⇒ **Intractable posterior distribution** $p(y^*|x^*, \mathcal{D})$!

Predictive Distribution $p(y^*|x^*, \mathcal{D})$ for BLR

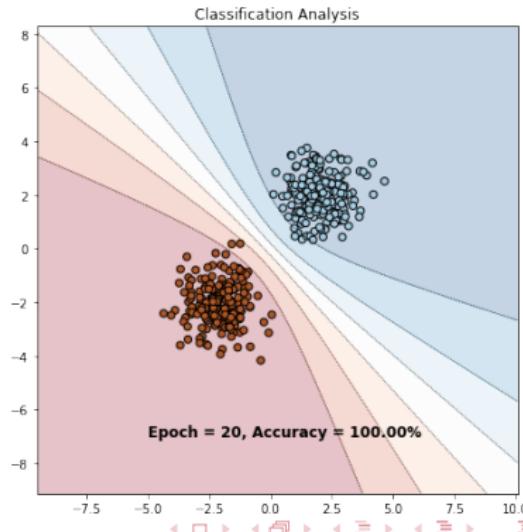
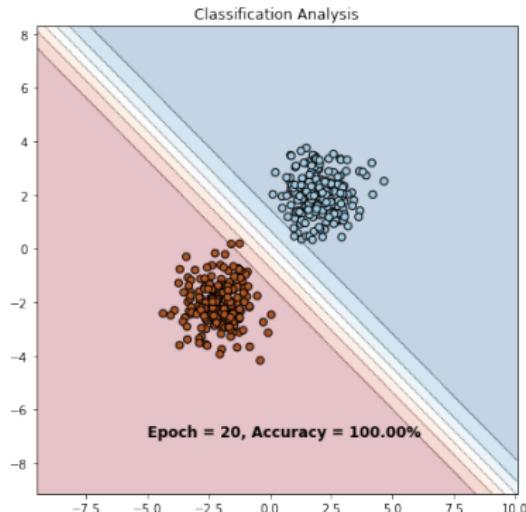
- Option 1: use Monte Carlo (MC) sampling

- ▶ Binary case (simple x-class extension): $p(y^* = 1|x^*, w) = \sigma(w^T x^*)$, σ sigmoid

$$p(y^* = 1|x^*, \mathcal{D}) \approx \sum_{s=1}^S \sigma((w^s)^T x^*) \quad w^s \sim q(w) = \mathcal{N}(w; \mu, \Sigma)$$

- ▶ Easy to sample from Gaussian $q(w)$

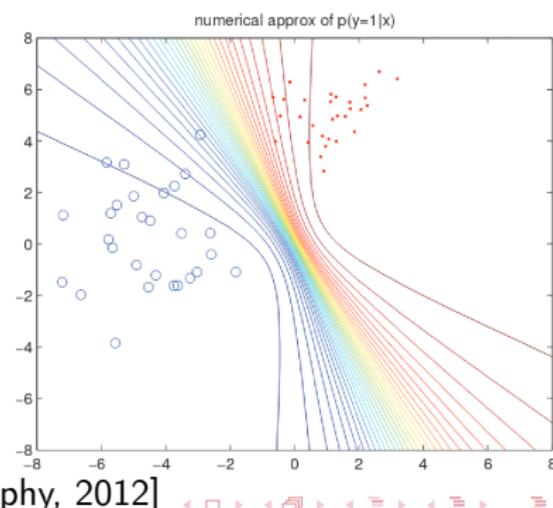
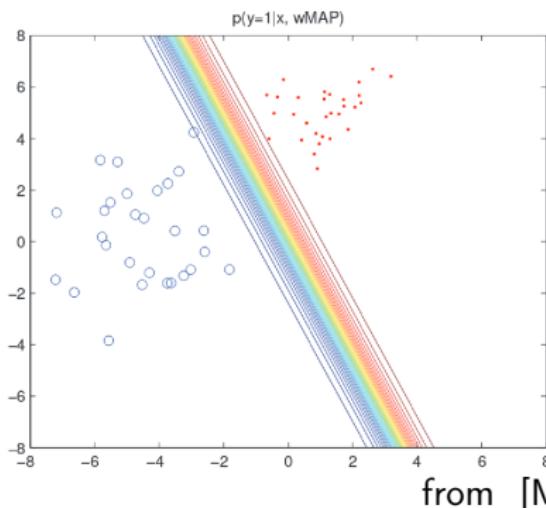
- **Practical session:** MAP solution for LR vs BLR (Laplace & MC sampling)



Predictive Distribution $p(y^*|x^*, \mathcal{D})$ for BLR

- $p(y^*|x^*, \mathcal{D}) \approx \int p(y^*|x^*, w)q(w)dw$ intractable
- **Option 2:** (binary case): $p(y^*|x^*, \mathcal{D}) \approx \int \sigma(w^T x^*)q(w)dw ; q(w) = \mathcal{N}(w; \mu, \Sigma)$
- Convolution of sigmoid with Gaussian still intractable
 - ▶ Approximate $\sigma(w^T x^*)$ by probit: $\sigma(a) \approx \Phi(\lambda a), \lambda^2 = \pi/8$
 - ▶ Convolution of probit with Gaussian \Rightarrow probit:

$$p(y^*|x^*, \mathcal{D}) \approx \int \Phi\left(\lambda w^T x^*\right) \mathcal{N}(w; \mu, \Sigma) dw = \Phi\left(\frac{\mu_f}{\sqrt{\lambda^{-2} + \sigma_f^2}}\right) \quad \mu_f = \mu^T x^* \quad \sigma_f^2 = x^{*T} \Sigma x^*$$



Outline

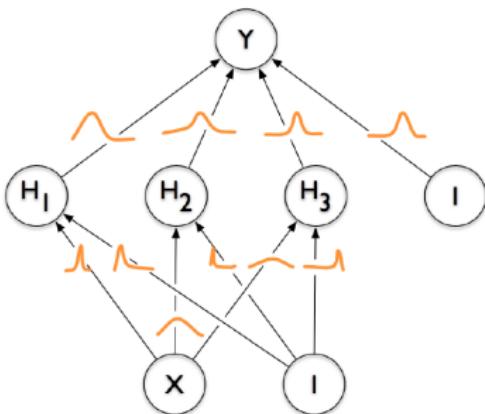
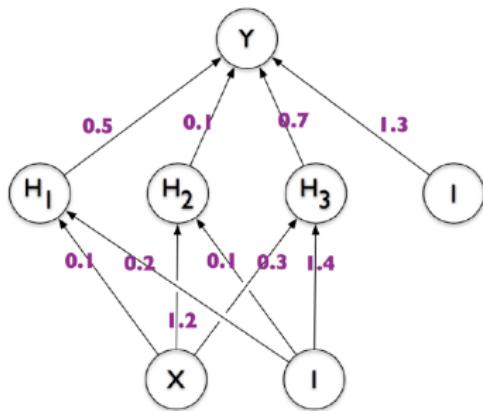
Beyond Bayesian Linear Regression

Bayesian Logistic Regression

Bayesian Neural Networks

Monte Carlo Dropout

Bayesian Neural Networks (BNN)



Credit: [Blundell et al., 2015]

- Standard NN: $y_i = f^w(x_i)$, **Bayesian NN:** $p(y_i|x_i, \mathcal{D})$
 - ▶ Define prior over weights $p(w)$, e.g. $p(w) = \mathcal{N}(w|0, \alpha^{-1}\mathcal{I})$ (point estimate for bias)
 - ▶ In practice, typically separate variance $\sigma^2 = \alpha^{-1}$ for each layer
 - ▶ Define likelihood, $p(y_i|x_i, w)$, e.g. for regression $p(y_i|x_i, w) = \mathcal{N}(y_i; f^w(x_i), \beta^{-1})$
 - ▶ **Goal: compute posterior** $p(w|X, Y) = \prod_{i=1}^N p(w|x_i, y_i, \beta) \propto p(w) \prod_{i=1}^N p(y_i|x_i, w)$

Bayesian Neural Networks (BNN)

$$p(w|X, Y) \propto p(w) \prod_{i=1}^N p(y_i|x_i, w)$$

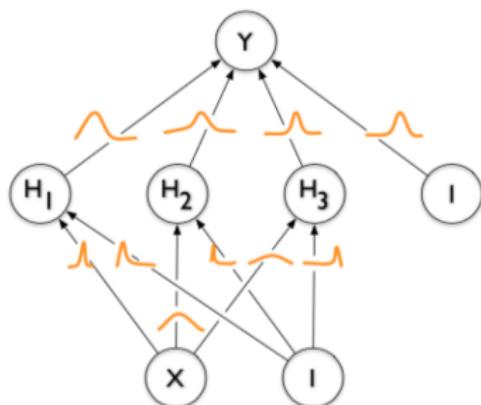
- With Bayesian Neural networks, even with:

- Gaussian prior $p(w) = \mathcal{N}(w|0, \alpha^{-1}\mathcal{I})$
- Gaussian likelihood, e.g. regression $p(y_i|x_i, w) = \mathcal{N}(y_i; f^w(x_i), \beta^{-1})$
- Posterior $p(w|X, Y, \beta) \propto p(w) \prod_{i=1}^N p(y_i|x_i, w)$ is NOT Gaussian!!

- Non-linear dependence of $f^w(x)$ on w !

- RECAP:

- $p(x) = \mathcal{N}(x|\mu_x, \Sigma_x)$
- $p(y|x) = \mathcal{N}(y|Ax + b, \Sigma_y)$
 - Linear dependence $Ax + b$ required
- Then: $p(x|y) = \mathcal{N}(x|\mu_{x|y}, \Sigma_{x|y})$
 - Not true for BNNs!



Credit: [Blundell et al., 2015]

Posterior Inference: MCMC Carlo Sampling

The true predictive distribution $p(y^*|x^*, \mathcal{D})$ cannot be evaluated analytically

$$p(y^*|x^*, \mathcal{D}) = \int p(y^*|x^*, w)p(w|\mathcal{D})dw$$

- Monte Carlo estimation of the integral:

$$p(y^*|x^*, \mathcal{D}) \approx \frac{1}{S} \sum_{s=1}^S p(y^*|x^*, w^s) \quad w^s \sim p(w|\mathcal{D})$$

- Can't sample exactly from $p(w|\mathcal{D})$, **BUT approximate sampling using Markov chain Monte Carlo (MCMC) possible !**
 - ▶ Metropolis-Hastings (MH), Hamiltonian Monte Carlo (HMC) [Neal, 1996]
- **Works well, accurate posterior inference in BNNs**
- **Main drawback: does not scale to large datasets**
 - ▶ Computing likelihood for MH/HMC acceptance step requires the whole dataset

Variational Inference (VI)

The true posterior $p(w|X, Y)$ cannot usually be evaluated analytically

- Defining an **approximating variational distribution** $q_\theta(w)$, parameterized by θ
- Minimizing its KL divergence with the true posterior:**

$$KL(q_\theta(w) \| p(w|X, Y)) = \int q_\theta(w) \log \frac{q_\theta(w)}{p(w|X, Y)} dw$$

- Computing approximate predictive distribution:** $p(y^*|x^*, X, Y) \Leftarrow q_{\theta^*}(w)$:

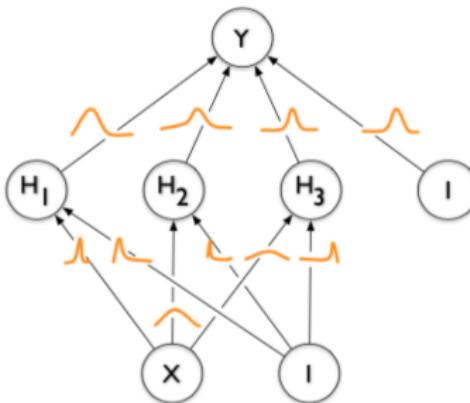
$$p(y^*|x^*, X, Y) \approx \int p(y^*|x^*, w) q_{\theta^*}(w) dw$$

Variational Inference (VI)

- Recap: BNN prior, e.g. Gaussian: $p(w) = \mathcal{N}(w|0, \alpha^{-1}\mathcal{I})$
- **Variational approximate posterior** $q_\theta(w)$, e.g. fully factorized Gaussian:

$$q_\theta(\cdot) = \mathcal{N}(\cdot|\theta) = \mathcal{N}(\cdot|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^D \mathcal{N}(w_i|\mu_i, \sigma_i)$$

- Each weight of the network w_j has its own mean μ_j and variance σ_j
 - ▶ $\theta = \{(\mu_j, \sigma_j)\}_{j \in \{1; D\}}$: **variational parameters**



Credit: [Blundell et al., 2015]

Variational Inference (VI): ELBO

$$\begin{aligned} KL(q_{\theta}(w) \| p(w|X, Y)) &= \int q_{\theta}(w) \log \frac{q_{\theta}(w)}{p(w|X, Y)} dw = - \int q_{\theta}(w) \log \frac{p(w|X, Y)}{q_{\theta}(w)} dw \\ &= - \int q_{\theta}(w) \log \frac{p(Y|X, w)p(w)}{q_{\theta}(w)p(Y|X)} dw \\ &= - \int q_{\theta}(w) \log p(Y|X, w) dw + \int q_{\theta}(w) \log \frac{q_{\theta}(w)}{p(w)} + \log p(Y|X) \\ &= - \int q_{\theta}(w) \log p(Y|X, w) dw + KL(q_{\theta}(w) \| p(w)) + \log p(Y|X) \end{aligned}$$

- $\Rightarrow KL(q_{\theta}(w) \| p(w|X, Y)) = -\mathcal{L}_{VI}(X, Y, \theta) + \log p(Y|X)$

- ▶ $\mathcal{L}_{VI}(X, Y, \theta)$: Evidence Lower Bound (ELBO)

$$\mathcal{L}_{VI}(\theta) = \int q_{\theta}(w) \log p(Y|X, w) dw - KL(q_{\theta}(w) \| p(w))$$

- ▶ $\mathcal{L}_{VI}(\theta) = \log p(Y|X) - KL(q_{\theta}(w) \| p(w|X, Y)) \leq \log p(Y|X)$

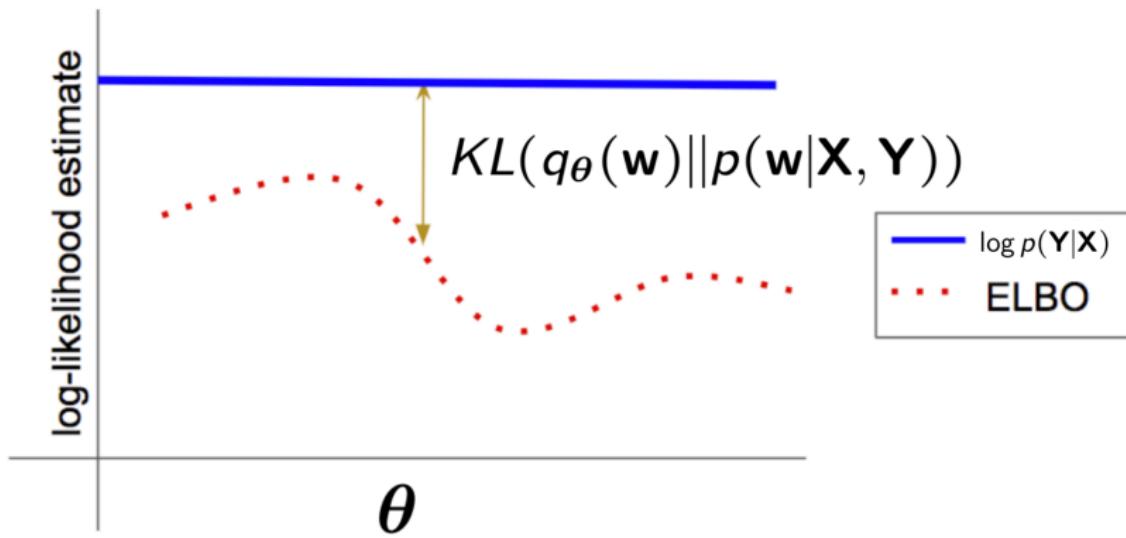
$$p(w|X, Y) = \frac{p(Y|X, w)p(w)}{p(Y|X)}$$

ELBO Illustration

- $\mathcal{L}_{VI}(\mathbf{X}, \mathbf{Y}, \boldsymbol{\theta})$: Evidence Lower Bound (ELBO)

$$\mathcal{L}_{VI}(\boldsymbol{\theta}) = \int q_{\boldsymbol{\theta}}(\mathbf{w}) \log p(\mathbf{Y}|\mathbf{X}, \mathbf{w}) d\mathbf{w} - KL(q_{\boldsymbol{\theta}}(\mathbf{w}) || p(\mathbf{w}))$$

- $\mathcal{L}_{VI}(\boldsymbol{\theta}) = \log p(\mathbf{Y}|\mathbf{X}) - KL(q_{\boldsymbol{\theta}}(\mathbf{w}) || p(\mathbf{w}|\mathbf{X}, \mathbf{Y})) \leq \log p(\mathbf{Y}|\mathbf{X})$

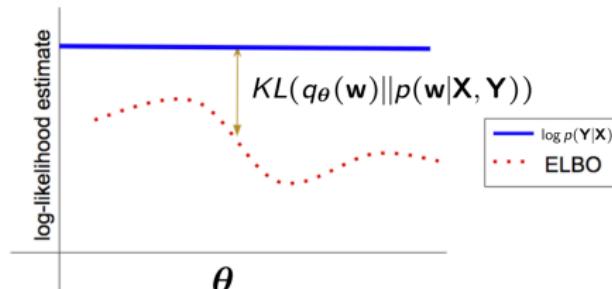


Variational Inference (VI): ELBO

- $\mathcal{L}_{VI}(\theta) = \log p(Y|X) - KL(q_\theta(w)||p(w|X, Y))$
- \Rightarrow Minimizing $KL(q_\theta(w)||p(w|X, Y)) \Leftrightarrow$ maximizing $\mathcal{L}_{VI}(\theta)$ w.r.t $q_\theta(w)$:

$$\begin{aligned}\mathcal{L}_{VI}(\theta) &= \int q_\theta(w) \log p(Y|X, w) dw - KL(q_\theta(w)||p(w)) \leq \log p(Y|X) \\ &= \mathbb{E}_{q_\theta(w)}[\log p(Y|X, w)] - KL(q_\theta(w)||p(w)) \\ &= \sum_{i=1}^N \int q_\theta(w) \log p(y_i|f^w(x_i)) dw - KL(q_\theta(w)||p(w))\end{aligned}$$

- **Expected log likelihood** $\mathbb{E}_{q_\theta(w)}[\log p(Y|X, w)]$: max $\Leftrightarrow q_\theta(w)$ explain data well
- **Prior KL** $KL(q_\theta(w)||p(w))$: min $\Leftrightarrow q_\theta(w)$ as close as possible to $p(w)$ prior



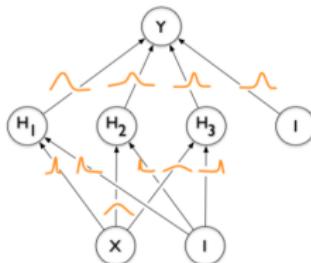
Variational Inference: Training

- Variational Bayesian NN training: computing derivates of \mathcal{L}_{VI} w.r.t variational parameters θ

$$\mathcal{L}_{VI}(\theta) = \sum_{i=1}^N \int q_\theta(w) \log p(y_i | f^w(x_i)) dw - KL(q_\theta(w) || p(w))$$

- RECAP: approximate variational posterior, e.g. fully factorize Gaussian:

$$q_\theta(w) = \prod_{i=1}^D \mathcal{N}(w_j | \mu_j, \sigma_j)$$



- **Prior KL** $KL(q_\theta(w) || p(w))$: often can be integrated analytically, e.g. with Gaussian functions for prior $p(w)$ and posterior approximation $q_\theta(w)$
- **Expected log likelihood** $\mathbb{E}_{q_\theta(w)}[\log p(Y|X, w)]$: no close-form solution in general, requires tractable calculations over the entire dataset
⇒ estimation by sampling

Stochastic Variational Inference (VI): Training

$$\mathcal{L}_{VI}(\theta) = \sum_{i=1}^N \int q_\theta(w) \log p(y_i | f^w(x_i)) dw - KL(q_\theta(w) || p(w))$$

- **Scalable gradient computation: batch sampling** [Graves, 2011]
 - ▶ \mathcal{L}_{VI} linearly decomposes into training examples, unbiased gradient estimator
- **Modern solutions: approximate integral with MC integration** $\hat{w}_i \sim q_\theta(w)$
 - ▶ Sample $\log p(y_i | f^{\hat{w}_i}(x_i))$, $\hat{w}_i \sim q_\theta(w)$
$$\mathcal{L}_{VI}(\theta) = \sum_{i \in S} \log p(y_i | f^{\hat{w}_i}(x_i)) - KL(q_\theta(w) || p(w))$$
- Issue: computing gradient wrt variational parameters $\frac{\partial}{\partial \theta} \log p(y_i | f^{\hat{w}_i}(x_i))$
- **Problem: sampling $\hat{w}_i \sim q_\theta(w)$ depends on variational parameters θ**
 - ▶ Solution, easy cases: re-parametrization $w = g(\theta, \epsilon)$
 - ▶ Where g deterministic and ϵ independent of θ - As in VAE [Kingma and Welling, 2014]
 - ▶ Crucial point: sampling fully in ϵ , independent of θ
 - ▶ Gaussian ex: $\theta = \{\theta_j\}$, $\theta_j = (\mu_j, \sigma_j)$: $\hat{w}_j \sim \mathcal{N}(w_j | \mu_j, \sigma_j)$
 - ▶ $w_j = g((\mu_j, \sigma_j), \epsilon_j) = \mu_j + \sigma_j \epsilon_j$: $\hat{w}_j \sim \mathcal{N}(w_j | \mu_j, \sigma_j) \Leftrightarrow \hat{\epsilon}_j \sim \mathcal{N}(\epsilon_j | 0, 1)$

$$\mathcal{L}_{VI}(\theta) = \sum_{i \in S} \log p(y_i | f^{g(\theta, \hat{\epsilon}_i)}(x_i)) - KL(q_\theta(w) || p(w))$$

Stochastic Variational Inference (VI): Training

Algorithm 1 Minimise divergence between $q_\theta(\omega)$ and $p(\omega|X, Y)$

- 1: Given dataset \mathbf{X}, \mathbf{Y} ,
 - 2: Define learning rate schedule η ,
 - 3: Initialise parameters θ randomly.
 - 4: **repeat**
 - 5: Sample M random variables $\hat{\epsilon}_i \sim p(\epsilon)$, S a random subset of $\{1, \dots, N\}$ of size M .
 - 6: Calculate stochastic derivative estimator w.r.t. θ :
$$\widehat{\Delta\theta} \leftarrow -\frac{N}{M} \sum_{i \in S} \frac{\partial}{\partial\theta} \log p(\mathbf{y}_i | \mathbf{f}^{g(\theta, \hat{\epsilon}_i)}(\mathbf{x}_i)) + \frac{\partial}{\partial\theta} \text{KL}(q_\theta(\omega) || p(\omega)).$$
 - 7: Update θ :
$$\theta \leftarrow \theta + \eta \widehat{\Delta\theta}.$$
 - 8: **until** θ has converged.
-

Bayesian Neural Networks: Predictive Distribution

- Gaussian approximation of posterior $q_\theta(w) \approx p(w|X, Y)$ e.g. VI or Laplace
- Predictive distribution: $p(y^*|x^*, \mathcal{D}) \approx \int p(y^*|x^*, w)q_\theta(w)dw$
- **Even with $q_\theta(w)$ Gaussian, no closed-form for $p(y^*|x^*, \mathcal{D})!$**
 - ▶ Again due to non-linear dependence of $y^*(x^*, w)$ wrt w

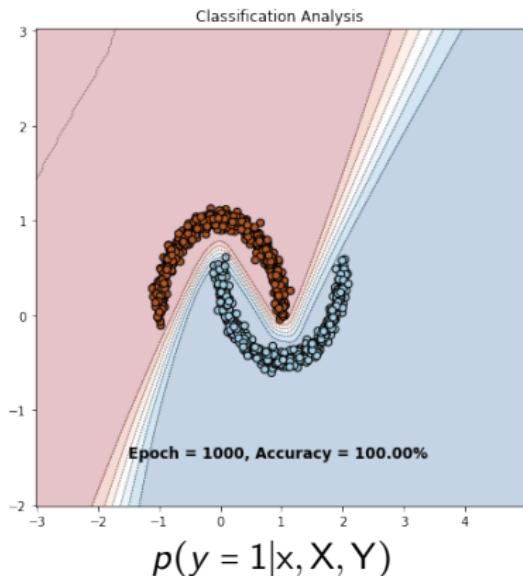
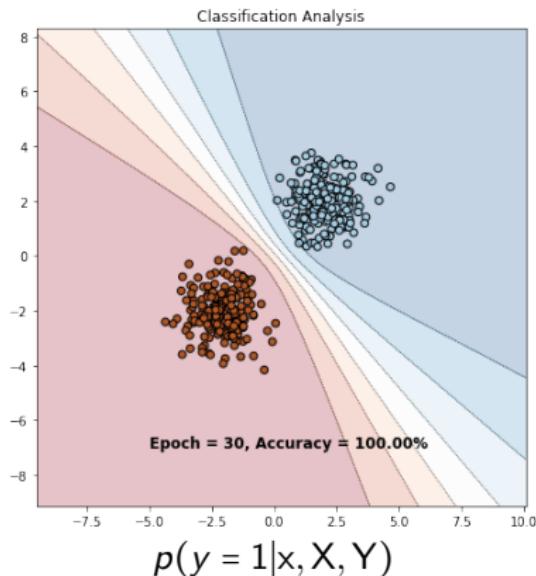
Solutions:

1. MC sampling, easy to sample from $q_\theta(w)$
 2. Perform Taylor expansion of $y^*(x^*, w)$ around w_{MAP} for regression^a:
$$y(x, w) \approx y(x, w_{MAP}) + \frac{\partial y}{\partial w} \Big|_{w=w_{MAP}} (w - w_{MAP})$$
- $p(y^*|x^*, w) \approx \mathcal{N}(y^*|\mu_y; \beta^{-1})$, $\mu_y = y(x, w_{MAP}) + \frac{\partial y}{\partial w} \Big|_{w=w_{MAP}} (w - w_{MAP})$
 - Closed form solution for $p(y^*|x^*, \mathcal{D})$
 - ▶ $p(y^*|x^*, \mathcal{D}) = \mathcal{N}(y^*|y(x, w_{MAP}); \sigma^2(x))$
 - ▶ $\sigma^2(x) = \beta^{-1} + \mathbf{g}^T \mathbf{A}^{-1} \mathbf{g}$, \mathbf{g} gradient and \mathbf{A} Hessian at w_{MAP}

^aFor classification, logit Taylor expansion, see 5.7.1 in [Bishop, 2006]

Practical session

- Own implementation of VI for:
 - ▶ Bayesian logistic regression
 - ▶ Neural network for non-linear classification (moons)
- Option: pyro <https://pyro.ai/> ⇒ stochastic process sampling, posterior approximation (VI), predictive distribution with MC sampling



Outline

Beyond Bayesian Linear Regression

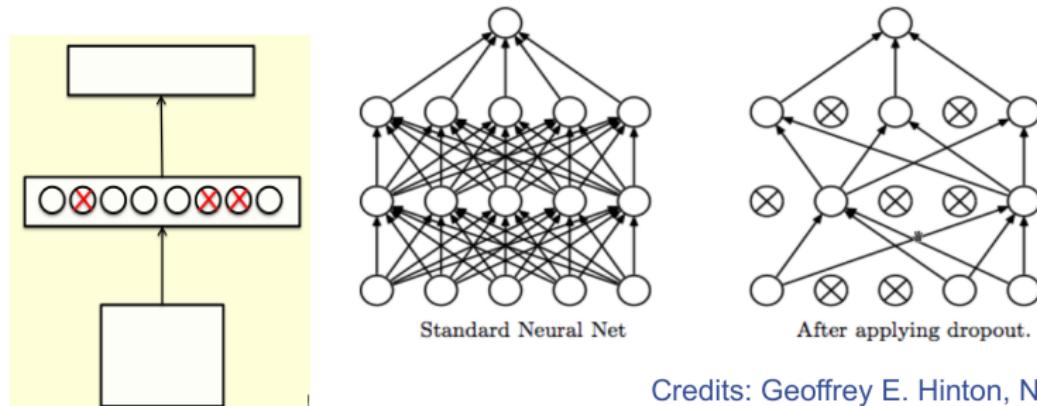
Bayesian Logistic Regression

Bayesian Neural Networks

Monte Carlo Dropout

Dropout [Hinton et al., 2012]

- Randomly omit each hidden unit with probability p , e.g. $p = 0.5$
- **Regularization technique**, limits over-fitting (better generalization)
 - ▶ Prevent co-adaptation
 - ▶ May be viewed as averaging over many NN
 - ▶ Slower convergence



Credits: Geoffrey E. Hinton, NIPS 2012

Dropout as a variational inference [Gal, 2016]

- Input $x \in \mathbb{R}^{D_a}$, latent vector $h \in \mathbb{R}^L$
 - ▶ First layer: $h = \sigma(xW_1)$, σ non-linearity
- **Dropout sampling:** in input : $x \odot \hat{\varepsilon}$
 - ▶ $\hat{\varepsilon} = \{\hat{\varepsilon}_i^1\}_{i \in \{1; D\}}$ $\varepsilon_i \sim \text{Bernoulli}(1 - p)$
 - ▶ First layer: $h = \sigma((x \odot \hat{\varepsilon})W_1)$
 - ▶ $(x \text{ diag}(\hat{\varepsilon}))W_1 = x(\text{ diag}(\hat{\varepsilon})W_1) = x\hat{W}_1$
 - ▶ Randomly setting to 0 rows of W_1 (size $((D, L))$ with probability p

$$\begin{array}{c} \text{Input } x \in \mathbb{R}^{D_a} \\ \text{Weight matrix } W_1 \in \mathbb{R}^{D \times L} \\ \text{Output } h \in \mathbb{R}^L \end{array}$$
$$x \odot \hat{\varepsilon} \quad \left(\begin{array}{c} w_1 \\ \vdots \\ w_D \end{array} \right) \quad \left(\begin{array}{c} \hat{\varepsilon}_1 \\ \ddots \\ 0 \end{array} \right) \quad \left(\begin{array}{c} \hat{\varepsilon}_D \end{array} \right) \quad = \quad \left(\begin{array}{c} h_1 \\ \vdots \\ h_L \end{array} \right)$$

^adimension (1,D)

Dropout as a variational inference [Gal, 2016]

- **Illustration: dropout for a 2 layer NN** (1 hidden), $\epsilon_i \sim \text{Bernoulli}(1 - p_i)$:

$$\blacktriangleright h_1 = \sigma(\hat{x}W_1) = \sigma(x\hat{W}_1), \hat{W}_1 = \text{diag}(\epsilon_1)W_1$$

$$\blacktriangleright \hat{y} := f^{\hat{W}_1, \hat{W}_2}(x) = \hat{h}_1 W_2 = h_1 \hat{W}_2, \hat{W}_2 = \text{diag}(\epsilon_2)W_2 - \hat{W} = \{\hat{W}_1; \hat{W}_2\}$$

- **MC Dropout sampling:** $\frac{1}{S} \sum_{s \in S} p(y_i | f^{\hat{W}}(x_i)) \approx \int p(y^* | f^W(x^*)) q(W) dw$

► ∀ layer $l \in \{1; L\}$, W_l random variable: $W_l \sim q(W_l) = g(M_l, \epsilon_l) = \text{diag}(\epsilon_l)M_l$
► $\epsilon_{l,i} \sim \text{Bernoulli}(1 - p_l)$, M_l deterministic parameters

$$\blacktriangleright q(W) = \prod_{l=1}^M q(W_l)$$

- **Big result** (see next): **training NN with dropout \Leftrightarrow training BNN with variational posterior approximation $q_M(W)$** (and some prior $p(W)$)

► $M = \{M_l\}_{l \in \{1; L\}}$ variational parameters

- **MC dropout:** sampling several passes with dropout \Leftrightarrow **performing MC approximate inference with variational posterior $q_M(W)$**

$$\frac{1}{S} \sum_{s \in S} p(y_i | f^{\hat{W}}(x_i)) \approx \int p(y^* | f^W(x^*)) q(W) dw \approx p(y^* | x^*, X, Y)$$

Dropout as a variational inference [Gal, 2016]

Big result: proof sketch for a 2 layer NN (regression)

- **Prediction with dropout:** $\hat{y} = \sigma(x\hat{W}_1)\hat{W}_2 =: f^{\hat{W}_1, \hat{W}_2}(x)$

$$\blacktriangleright \forall I \in \{1; 2\} : \hat{W}_I = \text{diag}(\hat{\epsilon}_I)M_I - \hat{W} = \{\hat{W}_1; \hat{W}_2\}$$

- **Training for regression,** $\hat{\mathcal{L}}_{\text{dropout}}$ objective function:

$$\hat{\mathcal{L}}_{\text{dropout}}(M_1, M_2) = \frac{1}{M} \sum_{i \in S} \|f^{\hat{W}}(x_i) - y_i\|^2 + \lambda_1 \|M_1\|^2 + \lambda_2 \|M_2\|^2$$

- With Gaussian likelihood: $p(y_i | f^{\hat{W}}(x_i)) = \mathcal{N}(y, f^{\hat{W}}(x), \tau^{-1}I)$, we have:

$$\|f^{\hat{W}}(x_i) - y_i\|^2 = -\frac{1}{\tau} \log p(y_i | f^{g(M, \hat{\epsilon}^i)}(x))$$

- $\hat{\mathcal{L}}_{\text{dropout}}$ rewrites as follows:

$$\boxed{\hat{\mathcal{L}}_{\text{dropout}}(M_1, M_2) = \frac{1}{M\tau} \sum_{i \in S} \log p(y_i | f^{g(M, \hat{\epsilon}^i)}(x)) + \lambda_1 \|M_1\|^2 + \lambda_2 \|M_2\|^2} \quad (1)$$

Dropout as a variational inference [Gal, 2016]

- Big similarity between $\hat{\mathcal{L}}_{dropout}$ in Eq (1) and algo 1!
- Same algorithms if:

$$\frac{\partial}{\partial \mathbf{M}} KL(q(\mathbf{W}) || p(\mathbf{W})) = \frac{\partial}{\partial \mathbf{M}} N\tau(\lambda_1 \|\mathbf{M}_1\|^2 + \lambda_2 \|\mathbf{M}_2\|^2)$$

[Gal and Ghahramani, 2016] showed that this can be fulfilled for:

- $p(\mathbf{W}) = \prod_I p(\mathbf{W}_I) = \prod_I \mathcal{MN}(\mathbf{W}_I; \mathbf{0}; I/I_I^2, I)$ (prior factorized over layers)
- $q(\mathbf{W}_I) = \text{diag}(\hat{\varepsilon}_I)\mathbf{M}_I, \varepsilon_{I,i} \sim \text{Bernoulli}(1 - p_i), q(\mathbf{W}) = \prod_I q(\mathbf{W}_I)$
 - ▶ Approximated by a mixture of two Gaussians with small std and one component fixed at zero
$$q_{\theta_{i,k}}(w_{i,k}) = (1 - p_i)\mathcal{N}(w_{i,k}; m_{i,k}; \sigma^2 I) + p_i\mathcal{N}(w_{i,k}; \mathbf{0}; \sigma^2 I)$$

⇒ A neural network with dropout can be interpreted as a variational Bayesian approximation

Model uncertainty

Predictive prediction with variational inference approximated with:

$$\begin{aligned} p(y^*|x^*, X, Y) &= \int p(y^*|f^W(x^*)) p(W|X, Y) dW \\ &\approx \int p(y^*|f^W(x^*)) q(W) dW := q_{W^*}(y^*|x^*) \end{aligned}$$

⇒ Estimate $p(y^*|x^*, X, Y)$ by MC sampling of $p(y^*|f^{\hat{W}}(x^*))$, $\hat{W} \sim q(W)$

- $W = \{W_i\}_{i=1}^L$ our set of random variables
- $f^W(x^*)$ our model's stochastic output
- $q_{W^*}(W)$ our optimum of variational distribution

Model uncertainty in regression

We will perform moment-matching and estimate the first two moments of the predictive distribution empirically.

Proposition

Given $p(y^*|f^W(x^*) = \mathcal{N}(y^*; f^W(x^*); \tau^{-1}I)$ for some $\tau > 0$,
 $\mathbb{E}_{q_{w^*}(y^*|x^*)}[y^*]$ can be estimated with the unbiased estimator

$$\widetilde{\mathbb{E}}[y^*] := \frac{1}{T} \sum_{t=1}^T f^{\hat{W}_t}(x^*) \xrightarrow{T \rightarrow \infty} \mathbb{E}_{q_{w^*}(y^*|x^*)}[y^*]$$

with $\hat{W}_t \sim q(W)$

⇒ equivalent to **performing T stochastic forward passes through the network and averaging the results.**

Model uncertainty in regression

Proposition

Given $p(y^*|f^W(x^*) = \mathcal{N}(y^*; f^W(x^*); \tau^{-1}I)$ for some $\tau > 0$, $\mathbb{E}_{q_w^*(y^*|x^*)}[(y^*)^T(y^*)]$ can be estimated with the unbiased estimator, with $\hat{W}_t \sim q(W)$:

$$\begin{aligned}\widetilde{\mathbb{E}}[(y^*)^T(y^*)] &:= \tau^{-1}I + \frac{1}{T} \sum_{t=1}^T f^{\hat{W}_t}(x^*)^T f^{\hat{W}_t}(x^*) \\ &\xrightarrow{T \rightarrow \infty} \mathbb{E}_{q_w^*(y^*|x^*)}[(y^*)^T(y^*)]\end{aligned}$$

Corollary

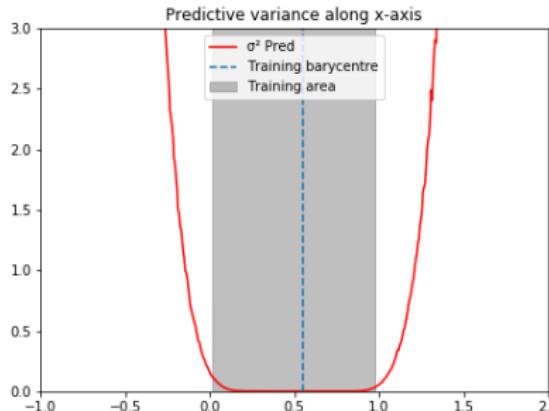
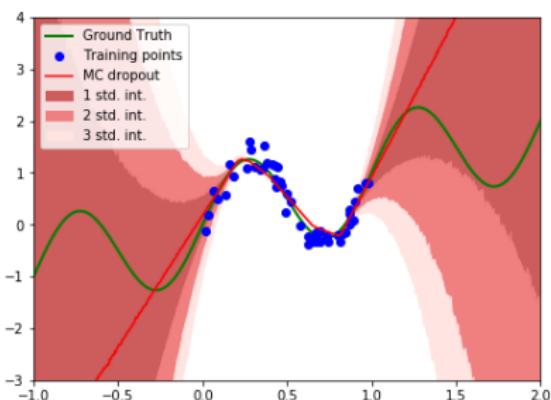
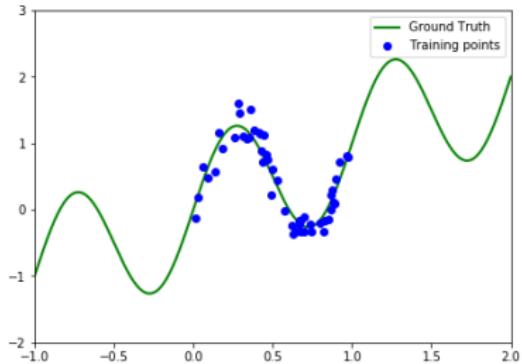
$\text{Var}_{q_w^*(y^*|x^*)}[(y^*)]$ can be estimated with the unbiased estimator

$$\begin{aligned}\widetilde{\text{Var}}[(y^*)] &:= \tau^{-1}I + \frac{1}{T} \sum_{t=1}^T f^{\hat{W}_t}(x^*)^T f^{\hat{W}_t}(x^*) - \frac{1}{T} \left(\sum_{t=1}^T f^{\hat{W}_t} \right)^T \left(\sum_{t=1}^T f^{\hat{W}_t} \right) \\ &\xrightarrow{T \rightarrow \infty} \text{Var}_{q_w^*(y^*|x^*)}[y^*]\end{aligned}$$

⇒ sample variance of T stochastic forward passes through the NN + the inverse model precision

Application: MC dropout for predictive distribution

- MC dropout for regression: $\sin(x) + x$

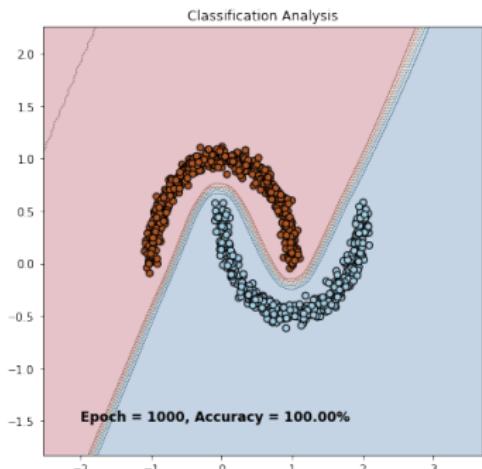


Application: MC dropout for predictive distribution

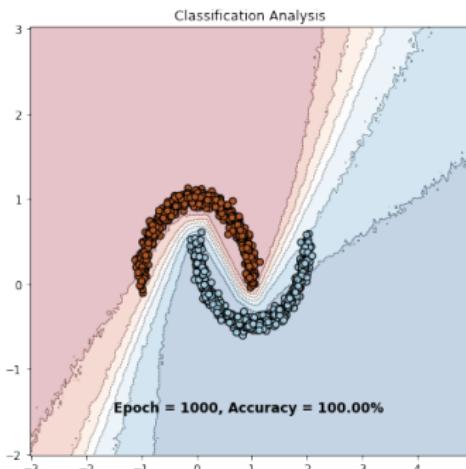
- MC dropout for non-linear classification

- As for BLR: $p(y^* = 1|x^*, \mathcal{D}) \approx \sum_{s=1}^S f_{w^s}(x^*)$, $w^s \sim q(W)$

Deterministic NN



Bayesian NN (MC dropout)



References |

- [Bishop, 2006] Bishop, C. M. (2006).
Pattern Recognition and Machine Learning (Information Science and Statistics).
Springer-Verlag, Berlin, Heidelberg.
- [Blundell et al., 2015] Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015).
Weight uncertainty in neural network.
In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1613–1622, Lille, France. PMLR.
- [Gal, 2016] Gal, Y. (2016).
Uncertainty in Deep Learning.
PhD thesis, University of Cambridge.
- [Gal and Ghahramani, 2016] Gal, Y. and Ghahramani, Z. (2016).
Dropout as a bayesian approximation: Representing model uncertainty in deep learning.
In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 1050–1059. JMLR.org.
- [Graves, 2011] Graves, A. (2011).
Practical variational inference for neural networks.
In Shawe-Taylor, J., Zemel, R. S., Bartlett, P. L., Pereira, F., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 24*, pages 2348–2356. Curran Associates, Inc.
- [Hernandez-Lobato and Adams, 2015] Hernandez-Lobato, J. M. and Adams, R. (2015).
Probabilistic backpropagation for scalable learning of bayesian neural networks.
In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1861–1869, Lille, France. PMLR.
- [Hinton et al., 2012] Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2012).
Improving neural networks by preventing co-adaptation of feature detectors.
CoRR, abs/1207.0580.
- [Hinton and van Camp, 1993] Hinton, G. E. and van Camp, D. (1993).
Keeping the neural networks simple by minimizing the description length of the weights.
In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, COLT '93, pages 5–13, New York, NY, USA. ACM.

References II

- [Jylänki et al., 2014] Jylänki, P., Nummenmaa, A., and Vehtari, A. (2014).
Expectation propagation for neural networks with sparsity-promoting priors.
Journal of Machine Learning Research, 15:1849–1901.
- [Kingma and Welling, 2014] Kingma, D. P. and Welling, M. (2014).
Auto-encoding variational bayes.
In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.
- [MacKay, 1992] MacKay, D. J. C. (1992).
A practical bayesian framework for backpropagation networks.
Neural Comput., 4(3):448–472.
- [Murphy, 2012] Murphy, K. P. (2012).
Machine Learning: A Probabilistic Perspective.
The MIT Press.
- [Neal, 1996] Neal, R. M. (1996).
Bayesian Learning for Neural Networks.
Springer-Verlag, Berlin, Heidelberg.