LogitNormal model

Beau Coker

1 Model

Network:

$$f(\mathbf{x}_n; \{s_d\}, \{\beta_k\}, \{\mathbf{w}_k\}, b) = \sum_{k=1}^{K} \beta_k \Phi(\mathbf{x}_n; s_d, \mathbf{w}_k, b) \tag{1}$$

$$\Phi(\mathbf{x}_n; s_d, \mathbf{w}_k, b) = \sqrt{2} \cos\left(\sum_{d=1}^{D} s_d w_{k,d} x_{n,d}\right) \tag{2}$$

In matrix notation:

$$f(X; S, \boldsymbol{\beta}) = \Phi(X; S, W, b)\boldsymbol{\beta} \tag{3}$$

$$\Phi(X; S, W, b) = \sqrt{2} \cos(XSW + \mathbf{b}) \tag{4}$$

$$\tag{5}$$

where $S = \mathrm{diag}(s_1, \ldots, s_D)$ and $W = [\mathbf{w}_1^T, \ldots, \mathbf{w}_K^T]^T$.

Model:

$$\beta_k \sim \mathcal{N}(0, \sigma_\beta^2), \quad k = 1, \ldots, K \tag{6}$$

$$s_d \sim \mathrm{LogitNormal}(\mu_s, \sigma_s^2) \tag{7}$$

$$y_n \mid \mathbf{x}_n, \{\beta_k\}, \{s_d\}, \{\mathbf{w}_k\}, b \sim \mathcal{N}(f(\mathbf{x}_n; \{s_d\}, \{\beta_k\}), \sigma^2), \quad n = 1, \ldots, N \tag{8}$$

$$\tag{9}$$

2 Inference

Algorithm 1: Training time

Input: Neural network with random weights $f(x; \beta, s)$, training data (x, y)

Result: Variational parameters ϕ for $q_\phi(s)$

Initialize ϕ;

for $i = 1 : n_{iter}$ **do**

  `/* Sample output weights` β_i `*/`

  Sample s from variational distribution $q_\phi(s)$;

  Sample β_i from full conditional: $p(\beta \mid x, y, s)$;

  `/* Update variational parameters` ϕ `*/`

  for $j = 1 : n_{grad}$ **do**

    Compute KL divergence $\mathrm{KL}(\phi) := \mathrm{KL}(q_\phi(s), p(s))$;

    for $k = 1 : n_{est}$ **do**

      Sample indicators $s_k = \mathrm{logistic}(\phi_\mu + \phi_\sigma \epsilon)$, where $\epsilon \sim \mathcal{N}(0, I)$;

      Compute function output $f_k(x) = f(x; \beta_i, s_k)$;

    Estimate likelihood term $\mathrm{L}(\phi) := \mathbb{E}_{\phi \sim q}[p(y \mid x, \beta, \phi)] \approx \frac{1}{n_{\text{est}}} \sum_k \mathcal{N}(y; f_k(x), \sigma^2 I)$;

    Compute loss $\mathrm{ELBO}(\phi) = \mathrm{L}(\phi) - \mathrm{KL}(\phi)$;

    Update variational parameters: $\phi \leftarrow \phi + \alpha \nabla_\phi \mathrm{ELBO}(\phi)$;

Algorithm 2: Test time (i.e. sampling from posterior predictive)

Input: Neural network with random weights $f(x; \beta, s)$, variational parameters ϕ, training data (x, y), test input x^*

Result: One sample f^* from posterior predictive at test input x^*

Sample s from variational distribution $q_\phi(s)$;

Sample β from full conditional: $p(\beta \mid x, y, s)$;

Evaluate network $f^* = f(x^*; \beta, s)$

Algorithm 3: Training time Version 2.0

Input: Neural network with random weights $f(x; \beta, s)$, training data (x, y)

Result: Variational parameters ϕ for $q_\phi(s)$

Initialize ϕ;

for $i = 1 : n_{iter}$ **do**

    `/* Update variational parameters` ϕ `*/`

    Compute KL divergence $\mathrm{KL}(\phi) := \mathrm{KL}(q_\phi(s), p(s))$;

    for $k = 1 : n_{est}$ **do**

        Sample indicators $s_k = \mathrm{logistic}(\phi_\mu + \phi_\sigma \epsilon)$, where $\epsilon \sim \mathcal{N}(0, I)$;

        Sample β_k from full conditional: $p(\beta \mid x, y, s_k)$;

        Compute function output $f_k(x) = f(x; \beta_k, s_k)$;

    Estimate likelihood term $\mathrm{L}(\phi) := \mathbb{E}_{\phi \sim q}[p(y \mid x, \beta, \phi)] \approx \frac{1}{n_{\text{est}}} \sum_k \mathcal{N}(y; f_k(x), \sigma^2 I)$;

    Compute loss $\mathrm{ELBO}(\phi) = \mathrm{L}(\phi) - \mathrm{KL}(\phi)$;

    Update variational parameters: $\phi \leftarrow \phi + \alpha \nabla_\phi \mathrm{ELBO}(\phi)$;