Statistical Machine Learning (BE4M33SSU)
Lecture 8: Bayesian inference and learning

Czech Technical University in Prague

- Bayesian inference
- Variational Bayesian inference
- Bayesian inference in Deep Learning

When ERM and MLE fail

Empirical risk minimisation:

- The best attainable (Bayes) risk is $R^* = \inf_{h \in \mathcal{Y}^\mathcal{X}} R(h)$
- The best predictor in $\mathcal{H}$ is $h_\mathcal{H} \in \arg\min_{h \in \mathcal{H}} R(h)$
- The predictor h_m learned from $\mathcal{T}^m$ has risk $R(h_m)$

$$\underbrace{\Big(R(h_m) - R^*\Big)}_{\text{excess error}} = \underbrace{\Big(R(h_m) - R(h_\mathcal{H})\Big)}_{\text{estimation error}} + \underbrace{\Big(R(h_\mathcal{H}) - R^*\Big)}_{\text{approximation error}}$$

- Misspecified hypothesis space $\mathcal{H} \Rightarrow$ high approximation error
- Size of $\mathcal{T}^m$ too small $\Rightarrow$ high estimation error

Maximum likelihood estimate: similar

- Misspecified model class $p_\theta(x, y)$, $\theta \in \Theta$
- Size of $\mathcal{T}^m$ too small

Small amount of training data: can we avoid to choose **one** h_m, or to decide for **one** θ^*?

Bayesian inference

Interpret the unknown parameter $\theta \in \Theta$ as a **random** variable

- Model class $p(x, y \mid \theta)$, $\theta \in \Theta$
- Prior distribution $p(\theta)$ on Θ
- Prediction strategy $h\colon \mathcal{X} \to \mathcal{Y}$
- A loss function $\ell\colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$

Given training data $\mathcal{T}^m = \big\{(x^i, y^i) \bigm| i = 1, \ldots, m\big\}$ compute the posterior probability to observe a pair (x, y) by marginalising over $\theta \in \Theta$:

$$p(x, y \mid \mathcal{T}^m) = \frac{1}{p(\mathcal{T}^m)} \int\limits_{\Theta} p(\mathcal{T}^m \mid \theta)\, p(x, y \mid \theta)\, p(\theta)\, d\theta$$

Notice that a point estimate of θ is no longer needed!

Define the Bayes risk of a strategy h by

$$R(h, \mathcal{T}^m) \propto \sum_{x,y} \int\limits_{\Theta} p(\mathcal{T}^m \mid \theta)\, p(x, y \mid \theta)\, p(\theta)\, \ell(y, h(x))\, d\theta$$

Bayesian inference

For 0-1 loss this leads to the predictor

$$h(x, \mathcal{T}^m) = \arg\max_{y \in \mathcal{Y}} \int_{\Theta} \underbrace{p(\theta)\, p(\mathcal{T}^m \mid \theta)}_{\alpha(\theta)}\, p(x, y \mid \theta)\, d\theta = \arg\max_{y \in \mathcal{Y}} \int_{\Theta} \alpha(\theta)\, p(y \mid x, \theta)\, d\theta$$

which means to find the optimal predictor for a **model mixture**.

Notice how the posterior distribution

$$\alpha(\theta) = p(\theta \mid \mathcal{T}^m) \propto p(\mathcal{T}^m \mid \theta)\, p(\theta)$$

interpolates between the situation without any training data, i.e. $m = 0$ and the likelihood of training data for $m \to \infty$.

Bayesian inference

Example 1 (linear regression)

$$y = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + \epsilon \quad \text{with } \epsilon \sim \mathcal{N}(0, \sigma^2)$$

and normal prior for $\boldsymbol{w} \sim \mathcal{N}(0, \sigma_0^2)$. Consequently, we have

$$p(y \mid \boldsymbol{x}, \boldsymbol{w}) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(y - \langle \boldsymbol{w}, \boldsymbol{x} \rangle)^2} \text{ and } p(\boldsymbol{w}) = \frac{1}{\sqrt{2\pi\sigma_0^2}} e^{-\frac{1}{2\sigma_0^2}\|\boldsymbol{w}\|^2}$$

Given training data $\mathcal{T}^m = (\boldsymbol{X}, \boldsymbol{y})$, the posterior distribution for $\boldsymbol{w}$ is Gaussian

$$p(\boldsymbol{w} \mid \mathcal{T}^m) \propto e^{-\frac{1}{2\sigma^2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|^2 - \frac{1}{2\sigma_0^2}\|\boldsymbol{w}\|^2}$$

- MAP estimate gives $\boldsymbol{w}^* = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1})\boldsymbol{X}^T\boldsymbol{y}$, where $\lambda = \sigma^2/\sigma_0^2$.
- if loss $\ell(y, y') = (y - y')^2$ is used, then

$$\boldsymbol{w}^* = \mathbb{E}_{\boldsymbol{w}|\mathcal{T}^m}[\boldsymbol{w}] = \int p(\boldsymbol{w} \mid \mathcal{T}^m)\, \boldsymbol{w}\, d\boldsymbol{w}$$

Variational Bayesian inference

- Computing integrals like

$$\int_{\Theta} p(\mathcal{T}^m \mid \theta)\, p(\theta)\, d\theta$$

  is in most cases not tractable.

- Approximate $p(\theta \mid \mathcal{T}^m)$ by some simple distribution $q_\beta(\theta)$ and find the optimal parameter β by minimising the Kullback Leibler divergence

$$-KL(q_\beta(\theta) \parallel p(\theta \mid \mathcal{T}^m)) = \int_{\Theta} q_\beta(\theta) \log p(\mathcal{T}^m \mid \theta)\, d\theta - KL(q_\beta(\theta) \parallel p(\theta)) + c \to \max_\beta$$

- use $q_\beta(\theta)$ with optimal β for prediction

$$h(x) = \arg\max_y \sum_{y'} \int_{\Theta} q_\beta(\theta)\, p(x, y \mid \theta)\, \ell(y', y)\, d\theta$$

The integrals over θ can be further simplified by sampling from $q_\beta(\theta)$

$$\int_{\Theta} q_\beta(\theta) f(\theta)\, d\theta \approx \frac{1}{m} \sum_{i=1}^{n} f(\theta_i)$$

Variational Bayesian inference

Example 2 Consider the optimisation task

$$\int_\Theta q_\beta(\theta) \log p(\mathcal{T}^m \mid \theta)\, d\theta - KL(q_\beta(\theta) \parallel p(\theta)) \to \max_\beta$$

for following examples

- $p(\theta)$ - uniform, $q_{\theta_0}(\theta) = \delta(\theta - \theta_0)$, i.e. point estimate $\Rightarrow$ $\theta_0 = \arg\max_\theta \log p(\mathcal{T}^m \mid \theta)$ i.e., MLE.
- $p(\theta)$ - $\mathcal{N}(0, \sigma_0^2)$, $q_{\theta_0}(\theta) = \delta(\theta - \theta_0)$, i.e. point estimate $\Rightarrow$

$$\theta_0 = \arg\max_\theta \big[\log p(\mathcal{T}^m \mid \theta) + \lambda \|\theta\|^2\big]$$

- $p(\theta)$ - $\mathcal{N}(0, \sigma_0^2)$, $q_\beta(\theta)$ - $\mathcal{N}(\mu, \sigma^2)$

$$\frac{1}{\sqrt{2\pi\sigma^2}} \int_\Theta e^{-\frac{1}{2\sigma^2}(\theta-\mu)^2} \log p(\mathcal{T}^m \mid \theta)\, d\theta - \frac{1}{2}\Big[\frac{\sigma^2 + \mu^2}{\sigma_0^2} - \ln \sigma\Big] \to \max_{\mu,\sigma}$$

Bayesian inference in Deep Learning

Variational Dropout (Kingma et al., 2015):

- Standard Dropout: randomly switch off neurons (with fixed probability p) during training. At test time – weight node outputs by $(1-p)$.
- Variational Dropout: Assume normal priors and normal posteriors for weights of deep NNs and learn their parameters. At test time: use learned mean values of weights.

Batch Normalisation (Ioffe et al., 2015)

Let a_i denote the activation of a single node in an NN, i.e. $a_i = \sum_j w_{ij} x_j + b_i$.
Re-parametrise weights and bias by

$$a_i = \left(\frac{a_i - \mu_i}{\sigma_i}\right) s_i + d_i,$$

where (μ_i, σ_i^2) is the statistics of a_i over a mini-batch and s_i, d_i are new scale and shift parameters. Do back-prop w.r.t. w'_{ij}, b'_i and s_i, d_i, where

$$w'_{ij} = \frac{w_{ij}}{\sigma_i} \quad \text{and} \quad b'_i = b_i - \mu_i$$

Bayesian inference in Deep Learning

This has the following advantages

- Choose $s_i = 1, d_i = 0$ at initialisation. This means that all nodes of the NN have zero mean and unit variance statistics in the first mini-batch.
- Gradient pre-conditioning improves training speed.
- The re-normalised weights and biases are stochastic (through the stochasticity of mini-batches). This can be interpreted as Bayesian inference and regularises learning.