Statistical Machine Learning (BE4M33SSU)
Lecture 9: EM algorithm; Bayesian learning

Czech Technical University in Prague

- Expectation Maximisation algorithm
- Bayesian inference
- Variational Bayesian inference

1. The Expectation Maximisation Algorithm

Unsupervised generative learning:

- The joint p.d. $p_\theta(x,y)$, $\theta \in \Theta$ is known up to the parameter $\theta \in \Theta$,
- given training data $\mathcal{T}^m = \left\{x^j \in \mathcal{X} \,\middle|\, i = 1,2,\ldots,m\right\}$ i.i.d. generated from p_{θ^*}.

How shall we implement the MLE for θ without knowing class labels y?

$$e_{ML}(\mathcal{T}^m) = \arg\max_{\theta\in\Theta} \frac{1}{m} \sum_{x\in\mathcal{T}^m} \log p_\theta(x) = \arg\max_{\theta\in\Theta} \mathbb{E}_{x\sim\mathcal{T}^m} \left[\log \sum_{y\in\mathcal{Y}} p_\theta(x,y)\right]$$

- If θ is a single parameter or a vector of homogeneous parameters $\Rightarrow$ maximise the log-likelihood directly by gradient ascent (provided it is differentiable in θ).
- If θ is a collection of heterogeneous parameters $\Rightarrow$ apply the **Expectation Maximisation Algorithm** (Schlesinger, 1968, Sundberg, 1974, Dempster, Laird, and Rubin, 1977)

1. The Expectation Maximisation Algorithm

EM algorithm (intuitive idea): Iterate the following two steps

- Given the current parameter estimate $\theta^{(t)}$, compute $\alpha_x(y) := p_{\theta^{(t)}}(y \,|\, x)$ for each $x \in \mathcal{T}^m$ and $y \in \mathcal{Y}$.
- Use this information as "soft" labels and solve the MLE task

$$\theta^{(t+1)} \in \arg\max_{\theta} \sum_{x \in \mathcal{T}^m} \sum_{y \in \mathcal{Y}} \alpha_x(y) \log p_\theta(x, y)$$

Can this really work? Yes it can! Consider the equation $\log p_\theta(x) = \log p_\theta(x, y) - \log p_\theta(y \,|\, x)$ and average it with $\alpha_x(y) = p_{\theta^{(t)}}(y \,|\, x)$

$$\log p_\theta(x) + \underbrace{\sum_{y \in \mathcal{Y}} \alpha_x(y) \log p_\theta(y \,|\, x)}_{g_\theta(x)} = \sum_{y \in \mathcal{Y}} \alpha_x(y) \log p_\theta(x, y)$$

If we choose a new $\theta^{(t+1)}$ that increases the r.h.s., then $\log p_\theta(x)$ will increase because $g_\theta(x)$ will decrease!

1. The Expectation Maximisation Algorithm

EM algorithm (alternative derivation):

- Introduce auxiliary variables $\alpha_x(y) \geqslant 0$, for each $x \in \mathcal{T}^m$, s.t. $\sum_{y \in \mathcal{Y}} \alpha_x(y) = 1$
- Construct a lower bound of the log-likelihood $L(\theta, \mathcal{T}^m) \geqslant L_B(\theta, \alpha, \mathcal{T}^m)$
- Maximise this lower bound by block-wise coordinate ascent.

Construct the bound:

$$L(\theta, \mathcal{T}^m) = \mathbb{E}_{\mathcal{T}^m}\Big[\log \sum_{y \in \mathcal{Y}} p_\theta(x, y)\Big] = \mathbb{E}_{\mathcal{T}^m}\Big[\log \sum_{y \in \mathcal{Y}} \frac{\alpha_x(y)}{\alpha_x(y)} p_\theta(x, y)\Big] \geqslant$$

$$L_B(\theta, \alpha, \mathcal{T}^m) = \mathbb{E}_{\mathcal{T}^m} \sum_{y \in \mathcal{Y}} \Big[\alpha_x(y) \log p_\theta(x, y) - \alpha_x(y) \log \alpha_x(y)\Big]$$

The following equivalent representation shows the difference between $L(\theta, \mathcal{T}^m)$ and $L_B(\theta, \alpha, \mathcal{T}^m)$:

$$L_B(\theta, \alpha, \mathcal{T}^m) = \mathbb{E}_{\mathcal{T}^m}\big[\log p_\theta(x)\big] - \mathbb{E}_{\mathcal{T}^m}\big[D_{KL}(\alpha_x(y) \parallel p_\theta(y \,|\, x))\big]$$

We see that the lower bound is tight if $\alpha_x(y) = p_\theta(y \,|\, x)$ holds $\forall x$ and $\forall y$.

1. The Expectation Maximisation Algorithm

Maximise $L_B(\theta, \alpha, \mathcal{T}^m)$ by block-coordinate ascent:

Start with some $\theta^{(0)}$ and iterate

E-step Fix the current $\theta^{(t)}$, maximise $L_B(\theta^{(t)}, \alpha, \mathcal{T}^m)$ w.r.t. α-s. This gives

$$\alpha_x^{(t)}(y) = p_{\theta^{(t)}}(y \mid x).$$

M-step Fix the current $\alpha^{(t)}$ and maximise $L_B(\theta, \alpha^{(t)}, \mathcal{T}^m)$ w.r.t. θ.

$$\theta^{(t+1)} = \arg\max_{\theta \in \Theta} \mathbb{E}_{\mathcal{T}^m}\Big[\sum_{y \in \mathcal{Y}} \alpha_x^{(t)}(y) \log p_\theta(x, y)\Big]$$

This is equivalent to solving the MLE for annotated training data.

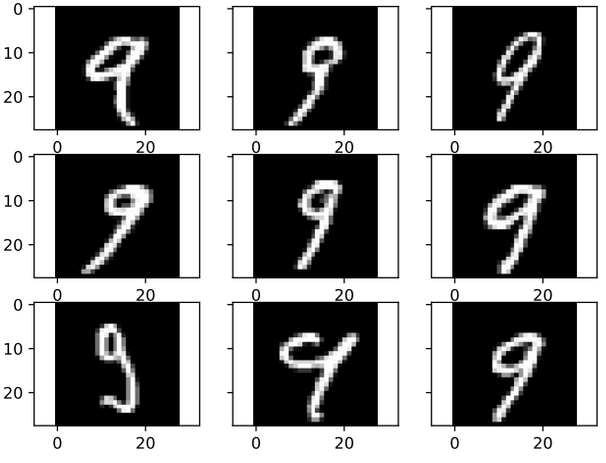
Claims:

- The sequence of likelihood values $L(\theta^{(t)}, \mathcal{T}^m)$, $t = 1, 2, \ldots$ is increasing, and the sequence $\alpha^{(t)}$, $t = 1, 2, \ldots$ is convergent (under mild assumptions).
- There is no guarantee that the EM algorithm converges to a global maximum.
- It is important to use a proper initialisation.

1. The Expectation Maximisation Algorithm

Example 1. We want to learn a simple model for different writing styles of digits

- $x = \{x_i \mid i \in D\}$ image on the pixel domain $D \in \mathbb{Z}^2$,
- $k \in K$ latent variable (mode indicator for the writing style),
- we assume a Naive Bayes model



$$p(x,k) = p(k) \prod_{i \in D} p(x_i \mid k)$$

Learning problem: Given i.i.d. training data $\mathcal{T}^m = \left\{x^j \,\middle|\, j = 1,2,\ldots,m\right\}$, estimate the mode probabilities $p(k)$ and the grey value probabilities $p(x_i \mid k)$ for each image pixel $i \in D$ given the mode indicator $k \in K$.

Applying the EM algorithm: Start with some model and iterate the following steps

E-step Given the current model estimate $p^{(t)}(x,k)$, compute the posterior mode probabilities for each image x in the training data $\mathcal{T}^m$, i.e. $\alpha_x^{(t)}(k) := p^{(t)}(k \mid x)$.

M-step Re-estimate the model by solving

$$\mathbb{E}_{\mathcal{T}^m}\Big[\sum_{k \in K} \alpha_x^{(t)}(k)\big[\log p(k) + \sum_{i \in D} \log p(x_i \mid k)\big]\Big] \to \max_p$$

1. The Expectation Maximisation Algorithm

Notice how the optimisation task in the M-step decomposes into independent small optimisation task. This gives the simple updates

$$p(k) = \mathbb{E}_{\mathcal{T}^m}\left[\alpha_x^{(t)}(k)\right]$$

$$p(x_i = b \mid k) = \frac{\mathbb{E}_{\mathcal{T}^m}\left[\alpha_x^{(t)}(k) \;\middle|\; x_i = b\right]}{\mathbb{E}_{\mathcal{T}^m}\left[\alpha_x^{(t)}(k)\right]}$$

Additional reading:

Schlesinger, Hlavac, Ten Lectures on Statistical and Structural Pattern Recognition, Chapter 6, Kluwer 2002 (also available in Czech)

Thomas P. Minka, Expectation-Maximization as lower bound maximization, 1998 (short tutorial, available on the internet)

2. Bayesian Inference

Motivation:

- Both, ERM and generative learning by MLE are consistent under the respective regularity assumptions. Their estimation errors $R(h_m) - R(h_{\mathcal{H}})$ and $\|\theta_m - \theta^*\|$ are small in the limit of large training data sizes m.
- On the other hand, their estimates h_m and θ_m can deviate substantially from the respective optimal predictor/model in case of small training data sizes.
- Models should be based on our knowledge about the problem. We do not want to restrict the complexity of the model $p_\theta(x, y)$, $\theta \in \Theta$ just because we have only a small amount of training data.
- Deciding for a single model $\theta_m = e_{ML}(\mathcal{T}^m)$ might be sub-optimal in such situations.

2. Bayesian inference

Bayesian inference:

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

- Data distribution: parametric family of models $p(x, y \,|\, \theta)$, $\theta \in \Theta$,
- Prior distribution $p(\theta)$ on Θ.

The prior distribution $p(\theta)$ and i.i.d. training data $\mathcal{T}^m = \big\{(x_i, y_i) \,\big|\, i = 1, \ldots, m\big\}$ define a *posterior parameter distribution* $p(\theta \,|\, \mathcal{T}^m)$, given by

$$p(\theta \,|\, \mathcal{T}^m) = \frac{p(\theta) p(\mathcal{T}^m \,|\, \theta)}{p(\mathcal{T}^m)} \quad \text{with} \quad p(\mathcal{T}^m \,|\, \theta) = \prod_{i=1}^{m} p(x^i, y^i \,|\, \theta).$$

The probability $p(\mathcal{T}^m)$ is obtained by integrating over θ, i.e. $p(\mathcal{T}^m) = \int p(\theta) p(\mathcal{T}^m \,|\, \theta)\, d\theta$ and does not depend on θ.

Notice that the posterior distribution $p(\theta \,|\, \mathcal{T}^m) \propto p(\mathcal{T}^m \,|\, \theta)\, p(\theta)$ interpolates between the situation without any training data, i.e. $m = 0$ and the likelihood of training data for $m \to \infty$.

2. Bayesian inference

Let us use $p(\theta \,|\, \mathcal{T}^m)$, but decide for a particular value of θ for given training data $\mathcal{T}^m$,

$$\theta_m = \operatorname*{arg\,max}_{\theta \in \Theta} p(\theta \,|\, \mathcal{T}^m) = \operatorname*{arg\,max}_{\theta \in \Theta} p(\mathcal{T}^m \,|\, \theta)\, p(\theta) = \operatorname*{arg\,max}_{\theta \in \Theta} \sum_{(x,y) \in \mathcal{T}^m} \log p(x, y \,|\, \theta) + \log p(\theta)$$

This results in an ML estimate with an additional regulariser

$$\theta_m = \operatorname*{arg\,max}_{\theta \in \Theta} \Big[\frac{1}{m} \sum_{(x,y) \in \mathcal{T}^m} \log p(x, y \,|\, \theta) + \frac{1}{m} \log p(\theta) \Big]$$

Example 2. We want to learn a DNN classifier with squashing activation functions (e.g. tanh or sigmoid). Assuming a Gaussian prior for the network weights, i.e. $w \sim \mathcal{N}(0, \sigma)$, we get the learning objective

$$\frac{1}{m} \sum_{(x,y) \in \mathcal{T}^m} \log p(y \,|\, x; w) - \frac{1}{2m\sigma^2} \|w\|^2 \to \max_w$$

This enforces a considerable fraction of neurons to have small weights and thus also small activations. They will therefore operate in a quasi linear regime.

2. Bayesian inference

A more powerful approach uses the posterior distribution $p(\theta \,|\, \mathcal{T}^m) \propto p(\mathcal{T}^m \,|\, \theta)\, p(\theta)$ to construct model mixtures and predictors. Consider the posterior probability to observe a pair (x, y) by marginalising over $\theta \in \Theta$:

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

This is a mixture of distributions with mixture weights $\alpha_m(\theta) \propto p(\mathcal{T}^m \,|\, \theta)\, p(\theta)$.

The Bayes optimal predictor w.r.t. 0/1 loss for this model mixture is

$$h(x, \mathcal{T}^m) = \operatorname*{arg\,max}_{y \in \mathcal{Y}} \int_\Theta \underbrace{p(\theta)\, p(\mathcal{T}^m \,|\, \theta)}_{\alpha_m(\theta)}\, p(x, y \,|\, \theta)\, d\theta = \operatorname*{arg\,max}_{y \in \mathcal{Y}} \int_\Theta \alpha_m(\theta)\, p(x, y \,|\, \theta)\, d\theta$$

Notice:

- the mixture weights $\alpha_m(\theta)$ interpolate between the situation without any training data, i.e. $m = 0$ and the likelihood of training data for $m \to \infty$.
- similar approaches for ERM lead to *Ensembling* methods (see lectures 12,13).

3. Variational Bayesian inference

Variational Bayesian inference:

Computing the integral $\int_\Theta p(\theta \,|\, \mathcal{T}^m)\, p(x, y \,|\, \theta)\, d\theta$ is in most cases not tractable.

We can approximate $p(\theta \,|\, \mathcal{T}^m)$ by some simple distribution $q_\varphi(\theta)$, $\varphi \in \Phi$ and try find the optimal parameter φ by minimising the Kullback-Leibler divergence

$$D_{KL}(q_\varphi(\theta) \,\|\, p(\theta \,|\, \mathcal{T}^m)) = D_{KL}(q_\varphi(\theta) \,\|\, p(\theta)) - \int_\Theta q_\varphi(\theta) \log p(\mathcal{T}^m \,|\, \theta)\, d\theta + c \to \min_\varphi$$

Then we use $q_\varphi(\theta)$ for constructing the model mixture and predictor (e.g. for 0/1 loss)

$$h(x) = \arg\max_y \int_\Theta q_\varphi(\theta)\, p(x, y \,|\, \theta)\, d\theta$$

The remaining integral over θ can be often further simplified by sampling $\theta_i \sim q_\varphi(\theta)$, i.e.

$$\int_\Theta q_\varphi(\theta) p(x, y \,|\, \theta)\, d\theta \approx \frac{1}{m} \sum_{i=1}^{n} p(x, y \,|\, \theta_i)$$

3. Variational Bayesian inference

Example 3 (Bayesian inference for a single neuron)**.** Let us consider a single neuron modelling class probabilities for $y = \pm 1$

$$p(y \,|\, x; w) = \sigma\big(y\langle w, x\rangle\big),$$

where $\sigma()$ denotes the sigmoid function. We assume the prior probability for the neuron weights $p(w) = \mathcal{N}(w; 0, \mathbb{I})$.

Given a training set $\mathcal{T}^m = \{(x^i, y^i) \,|\, i = 1, \ldots, m\}$, the posterior weight distribution is

$$p(w \,|\, \mathcal{T}^m) \propto p(w) \prod_{(x,y)\in\mathcal{T}^m} p(y \,|\, x; w)$$

We will approximate it by a normal distribution $q_\mu(w) = \mathcal{N}(w; \mu, \mathbb{I})$. We must solve

$$\int_{\mathbb{R}^n} q_\mu(w) \sum_{(x,y)\in\mathcal{T}^m} \log\sigma\big(y\langle w, x\rangle\big)\, dw - D_{KL}(q_\mu(w) \,\|\, p(w)) \to \max_\mu$$

3. Variational Bayesian inference

The KL-divergence can be computed and differentiated in closed form.

Let us discuss computing the gradient of the first term

$$\int_{\mathbb{R}^n} q_\mu(w) \sum_{(x,y)\in\mathcal{T}^m} \log\sigma\big(y\langle w,x\rangle\big)\,dw \overset{w=v-\mu}{=} \int_{\mathbb{R}^n} q_0(v) \sum_{(x,y)\in\mathcal{T}^m} \log\sigma\big(y\langle v-\mu,x\rangle\big)\,dv$$

We can use a stochastic gradient estimator by

1. sample $v_i \sim q_0(v)$
2. draw a mini-batch $\mathcal{B}$ from training data and estimate the gradient by

$$g = \nabla_\mu \sum_{(x,y)\in\mathcal{B}} \log\sigma\big(y\langle v_i-\mu,x\rangle\big)$$