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

Czech Technical University in Prague

- Unsupervised generative learning
- Expectation Maximisation algorithm
- Bayesian inference
- Variational Bayesian inference

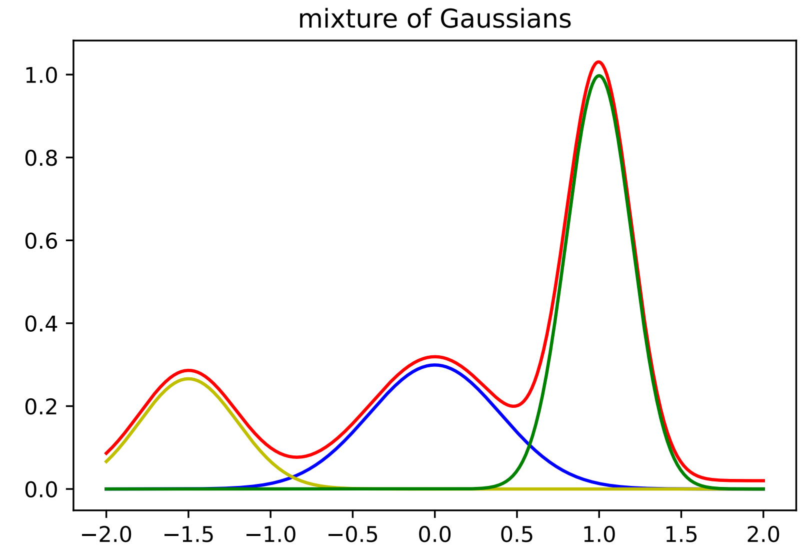
Unsupervised generative learning

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

Can we estimate the parameter θ without ever seeing the hidden states y?

Example 1 (Mixture of Gaussians)**.**
We observe data $x \in \mathbb{R}$ generated from a mixture of k Gaussians

$$p(x) = \sum_{i=1}^{k} \alpha_i \frac{1}{\sqrt{2\pi}\sigma_i} e^{\frac{(x-\mu_i)^2}{2\sigma_i^2}}$$



Can we estimate the parameters α_i, μ_i, σ_i from given training data $\mathcal{T}^m = \left\{ x^j \in \mathcal{X} \;\middle|\; j = 1, \ldots, m \right\}$?

Unsupervised generative learning

Example 2 (Generating handwritten digits)**.**
Our training set consists of images of handwritten digits (MNIST). We want to design and train a model for generating such images. We consider a model

$$p(x,z) = p_\theta(x \,|\, z)p(z),$$

where $x \in \mathbb{R}^{h\times w}$ is an image and $z \in \mathbb{R}^n$ is a vector of latent variables encoding shapes and writing styles. We fix a simple prior distribution $p(z)$ on the latent space, e.g. $\mathcal{N}(0,\mathbb{I})$, and a parametric model $p_\theta(x \,|\, z)$, e.g. $\mathcal{N}\big(\mu(z,\theta),\sigma^2\mathbb{I}\big)$, where $\mu(z,\theta)$ is a parametrised mapping $z \in \mathbb{R}^n \mapsto x \in \mathbb{R}^{h\times w}$.
Can we estimate the parameter θ without ever seeing the latent states z?

Unsupervised generative learning

Given a parametric family of distributions $p_\theta(x, y)$, $\theta \in \Theta$ and a training set $\mathcal{T}^m = \{x^j \in \mathcal{X} \mid i = 1, \ldots, m\}$, we want to estimate the model parameter θ by the maximum likelihood estimator

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

- 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)

The Expectation Maximisation Algorithm

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

- Given the current parameter estimate $\theta^{(t)}$, compute the hidden state probabilities $\alpha_x(y) := p_{\theta^{(t)}}(y \mid x)$ 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 $\log p_\theta(x) = \log p_\theta(x, y) - \log p_\theta(y \mid x)$ for a single training example x and average it with $\alpha(y) = p_{\theta^{(t)}}(y \mid x)$

$$\underbrace{\log p_\theta(x)}_{f(\theta)} = \underbrace{\sum_{y \in \mathcal{Y}} \alpha(y) \log p_\theta(x, y)}_{g(\theta)} - \underbrace{\sum_{y \in \mathcal{Y}} \alpha(y) \log p_\theta(y \mid x)}_{h(\theta)}$$

We notice that the function $h(\theta)$ has its global maximum at $\theta^{(t)}$.

The Expectation Maximisation Algorithm

By denoting $h_t = h(\theta^{(t)})$ and rewriting the equality

$$f(\theta) = \underbrace{\big[g(\theta) - h_t\big]}_{\widetilde{g}(\theta)} - \underbrace{\big[h(\theta) - h_t\big]}_{\widetilde{h}(\theta)},$$

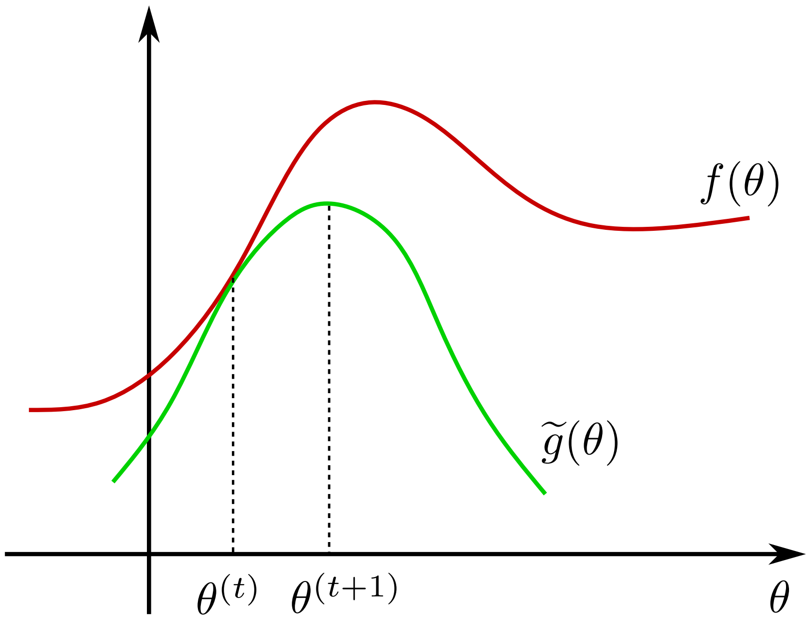
we see:

- $\widetilde{h}(\theta)$ has global maximum $\widetilde{h}(\theta^{(t)}) = 0$.
- $\widetilde{g}(\theta)$ lower bounds $f(\theta)$
- $f(\theta^{(t)}) = \widetilde{g}(\theta^{(t)})$ and their gradients in this point coincide.

Given the current estimate $\theta^{(t)}$ we define $\alpha(y) = p_{\theta^{(t)}}(y \,|\, x)$ and

$$g(\theta) = \sum_{y \in \mathcal{Y}} \alpha(y) \log p_\theta(x, y).$$

We maximise $g(\theta)$ instead of $f(\theta) = \log p_\theta(x)$. It is guaranteed that $\log p_{\theta^{(t+1)}}(x) \geqslant \log p_{\theta^{(t)}}(x)$ holds for the maximiser $\theta^{(t+1)}$ of $g(\theta)$.



The Expectation Maximisation Algorithm

Start with a suitably chosen $\theta^{(0)}$ and iterate the following steps until convergence

E-step Fix the current $\theta^{(t)}$ and compute

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

M-step Fix the current $\alpha^{(t)}$, use them as "soft" labels and solve the MLE task.

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

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

Claims:

- The sequence of likelihood values $L(\theta^{(t)}) = \frac{1}{m}\sum_{x \in \mathcal{T}^m} \log p_{\theta^{(t)}}(x)$, $t = 1, 2, \ldots$ is increasing.
- The sequence of $\alpha_x^{(t)}$, $t = 1, 2, \ldots$ is convergent.

There is **no guarantee** that the EM algorithm converges to a global maximum of the log-likelihood. This underlines the importance of a suitable initialisation.

The Expectation Maximisation Algorithm

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)

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.

Idea: Given training data $\mathcal{T}^m = \{(x_j, y_j) \mid j = 1, 2, \ldots, m\}$, decide for a **weighted mixture** of models

$$p(x, y) = \sum_{k=1}^{K} \alpha_k(\mathcal{T}^m)\, p_{\theta_k}(x, y)$$

and use it as predictive distribution.

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 $\mathcal{T}^m = \big\{(x_j, y_j) \;\big|\; j = 1, \ldots, m\big\}$ 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 $m \to \infty$.

Bayesian inference: MAP decision

Let us use $p(\theta \,|\, \mathcal{T}^m)$, but decide for a single value of θ by using the MAP criterion,

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

This results in an ML estimate with an additional regulariser

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

Example 3. 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 semi-linear regime.

Bayesian inference

The Bayesian 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 $\theta \in \Theta$:

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

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

$$h(x, \mathcal{T}^m) = \arg\max_{y \in \mathcal{Y}} \int\limits_{\Theta} \underbrace{p(\theta)\, p(\mathcal{T}^m \,|\, \theta)}_{\alpha_m(\theta)}\, p(x, y \,|\, \theta)\, d\theta = \arg\max_{y \in \mathcal{Y}} \int\limits_{\Theta} \alpha_m(\theta)\, p(x, y \,|\, \theta)\, d\theta$$

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).

Variational Bayesian inference:

Computing the integral $\int\limits_{\Theta} p(\theta \,|\, \mathcal{T}^m)\, p(x, y \,|\, \theta)\, d\theta$ 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)

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

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

$$\int_\Theta q_\varphi(\theta) p(x, y \,|\, \theta)\, d\theta \approx \frac{1}{K} \sum_{k=1}^{K} p(x, y \,|\, \theta_k)$$

Variational Bayesian inference

Example 4 (Variational 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 $p(w)$ as $w \sim \mathcal{N}(0, \mathbb{I})$.

Given a training set $\mathcal{T}^m = \{(x^i, y^i) \mid i = 1, \ldots, m\}$, 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 $q_\mu(w)$ as $w \sim \mathcal{N}(\mu, \mathbb{I})$. We must solve

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

Variational Bayesian inference

The KL-divergence can be computed in closed form (see seminar).

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 $v_i \sim \mathcal{N}(0,\mathbb{I})$
2. draw a mini-batch $\mathcal{B}$ from training data and estimate the gradient by

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

Let $q_{\mu_*}(w)$, i.e. $w \sim \mathcal{N}(\mu_*,\mathbb{I})$ denote the optimal approximate of the posterior distribution. The predictive distribution is then

$$p(x,y) \propto \int_{\mathbb{R}^n} q_{\mu_*}(w)\sigma\big(y\langle w,x\rangle\big)\,dw = \int_{\mathbb{R}^n} \frac{e^{-\frac{(w-\mu_*)^2}{2}}}{1+e^{-y\langle w,x\rangle}}\,dw$$

Appendix: Alternative derivation of the EM Algorithm

- 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 $\alpha_x(y) = p_\theta(y \mid x)$ holds $\forall x$ and $\forall y$.