T-61.5030 Advanced course in neural computing

Solutions for exercise 4

1. In the univariate case the distribution of the output for expert k is

$$f(d|\mathbf{x},\theta,k) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}(d-y_k)^2\sigma^{-2}} .$$

The multivariate Gaussian distribution is

$$f(\mathbf{d}|\mathbf{x},\boldsymbol{\theta},k) = \frac{1}{(2\pi)^{q/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(\mathbf{d}-\mathbf{y}_k)^T\Sigma^{-1}(\mathbf{d}-\mathbf{y}_k)} ,$$

where q is the dimension of output and Σ is the q-by-q covariance matrix. Otherwise the network does not change.

2. We shall use the maximum likelihood based learning, where the learning algorithm tries to find the parameters which maximise the probability of the data. Equivalently, we can maximise the logarithm of the probability. Then the probability of all samples, which is a product of probabilities of single samples, splits into a sum.

In the stochastic gradient algorithm, the adaptation is done based on one sample. This means that on the average the adaptation is done in the direction of the gradient of the whole data set. Denote the log-likelihood of one sample by

$$l(\mathbf{w},\mathbf{g}) = \ln\left(\frac{1}{(2\pi)^{q/2}}\sum_{k=1}^{K} g_k e^{-\frac{1}{2}||\mathbf{d}-\mathbf{y}_k||^2}\right) .$$

Then

$$\frac{\partial l}{\partial \mathbf{y}_k} = \frac{(\mathbf{d}-\mathbf{y}_k) g_k e^{-\frac{1}{2}||\mathbf{d}-\mathbf{y}_k||^2}}{\sum_{i=1}^{K} g_i e^{-\frac{1}{2}||\mathbf{d}-\mathbf{y}_i||^2}} = h_k(\mathbf{d}-\mathbf{y}_k) ,$$

where

$$h_k = \frac{g_k e^{-\frac{1}{2}||\mathbf{d}-\mathbf{y}_k||^2}}{\sum_{i=1}^{K} g_i e^{-\frac{1}{2}||\mathbf{d}-\mathbf{y}_i||^2}} .$$

Notice that h_k is the posterior probability of the expert k given the data. If there would be only one expert, the gradient would be $\mathbf{d}-\mathbf{y}$ and the back propagation would proceed as usual. This shows that now the learning of each expert is weighted by the probability for that expert to be responsible for the sample.

If the gating weights g_k are given by soft-max $g_k = e^{u_k}/\sum_{i=1}^{K} e^{u_i}$, then

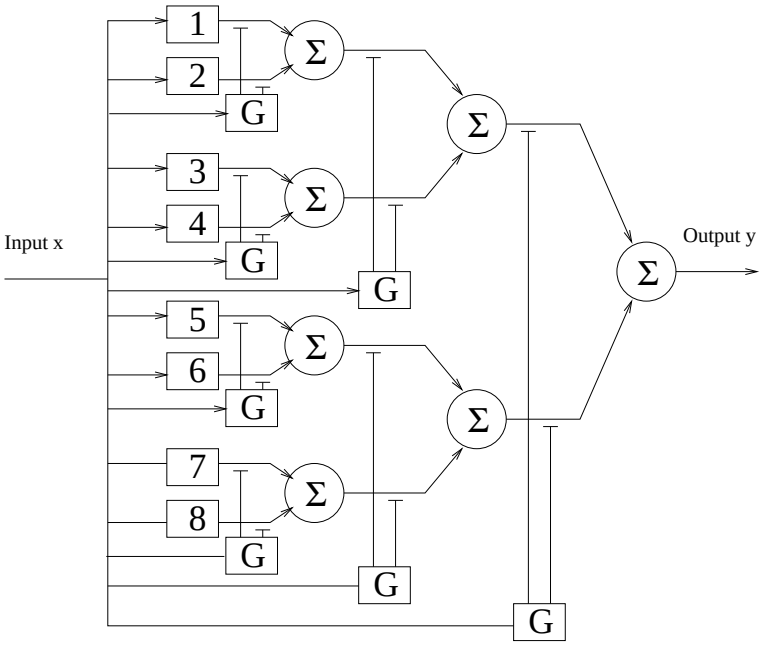
$$l(\mathbf{w},\mathbf{g}) = \ln\left(\frac{1}{(2\pi)^{q/2}}\right) + \ln\left(\sum_{k=1}^{K} e^{u_k} e^{-\frac{1}{2}||\mathbf{d}-\mathbf{y}_k||^2}\right) - \ln\left(\sum_{i=1}^{K} e^{u_i}\right)$$

and

$$\frac{\partial l}{\partial u_k} = \frac{e^{u_k} e^{-\frac{1}{2}||\mathbf{d}-\mathbf{y}_k||^2}}{\sum_{i=1}^{K} e^{u_i} e^{-\frac{1}{2}||\mathbf{d}-\mathbf{y}_i||^2}} - \frac{e^{u_k}}{\sum_{i=1}^{K} e^{u_i}} = h_k - g_k .$$

This means that the learning tries to bring the weights g_k of the experts as close as possible to the posterior probability h_k.

3. (a) Note that Haykin does not make distinction between weighted output and weighted probability of output. Weighted output does not necessarily make sense, but weighted probability of output always does. If the output is tempreature and one expert says 9°C and the other 7°C, it makes sense to say that the temperature would be 8°C. If the network does hand written digit recognition and the output is the digit, combination of experts giving answers 9 and 7 is, of course, not 8 but 9 with 50% probability and 7 with 50% probability. In this course, it is assumed by default that a committee of experts computes weighted probabilities.



(b) Let us use the same notation as in figure 7.11. of Haykin's book. The weights of the gating function closest to the output are g_1 and g_2, the weights from the second closest gating functions are g_{11}, g_{12}, g_{21} and g_{22}, etc. Assuming a single output with unit variance, the probability for the output is

$$f(d|\mathbf{x},\boldsymbol{\theta}) = \frac{1}{\sqrt{2\pi}} \sum_{l=1}^{2} g_l \sum_{k=1}^{2} g_{kl} \sum_{j=1}^{2} g_{jkl} e^{-\frac{1}{2}(d-y_{jkl})^2} .$$

4. The EM algorithm tries to do maximum likelihood estimation by maximising $p(d|\theta)$ w.r.t. θ, where d are observations and θ are the parameters. The problem is that the model gives the probability $p(d, z|\theta)$ but z are unknown. EM algorithm can be used if integration of $p(d, z|\theta)$ over is difficult but integration over $\ln p(d, z|\theta)$ can be done more easily.

At each step the new estimate $\hat{\theta}$ for parameters is done by maximising

$$\hat{\theta}(n+1) = \arg\max_{\theta} Q(\theta, \hat{\theta}(n)) ,$$

where

$$Q(\theta, \hat{\theta}) = \int p(z|d, \hat{\theta}) \ln p(z, d|\theta) dz .$$

We shall prove that this step never decreases $\ln p(d|\hat{\theta})$.

First notice that

$$Q(\theta, \hat{\theta}) = \int p(z|d, \hat{\theta}) \ln(p(d|\theta)p(z|d, \theta))dz = \ln p(d|\theta) + \int p(z|d, \hat{\theta}) \ln p(z|d, \theta)dz \,.$$

From $Q(\hat{\theta}(n+1), \hat{\theta}(n)) \geq Q(\hat{\theta}(n), \hat{\theta}(n))$ it then follows

$$\ln p(d|\hat{\theta}(n+1)) \geq \ln p(d|\hat{\theta}(n)) + \int p(z|d, \hat{\theta}(n)) \ln \frac{p(z|d, \hat{\theta}(n))}{p(z|d, \hat{\theta}(n+1))} dz \geq \ln p(d|\hat{\theta}(n)) \,.$$

This is because the integral in the last equation is the Kullback-Leibler divergence between $p(z|d, \hat{\theta}(n))$ and $p(z|d, \hat{\theta}(n+1))$ and K-L divergence is always non-negative.