Neural Computation : Important Equations

John A. Bullinaria - 2015

To study Neural Networks effectively, one has to deal with quite a lot of mathematics. The examination will not require you to carry out complex mathematical derivations, but you should be able to understand and explain the following important equations from the lectures, and be able to remember and write down those marked with a "":*

Basic Neuron Equation

$$out_j = f\left(\sum_{i=1}^{n} w_{ji}\, in_i - \theta_j\right) \qquad *$$

Sigmoid/Logistic Activation Function

$$\text{Sigmoid}(x) = \frac{1}{1+e^{-x}} \qquad *$$

Sum Squared Error and Cross Entropy Cost Functions

$$E_{SSE}(\{w_{kl}\}) = \tfrac{1}{2}\sum_p \sum_j \left(targ_j^p - out_j(in_i^p)\right)^2 \qquad *$$

$$E_{CE}(\{w_{ikl}\}) = -\sum_p \sum_j \left[targ_j^p.\log\left(out_j\left(in_i^p\right)\right) + \left(1-targ_j^p\right).\log\left(1-out_j\left(in_i^p\right)\right)\right]$$

Gradient Descent – General Weight Update Equation

$$\Delta w_{kl} = -\eta \frac{\partial E(\{w_{ij}\})}{\partial w_{kl}} \qquad *$$

Gradient Descent Updates for Single Layer Perceptron

$$\Delta w_{kl} = \eta \sum_p \left(targ_l - out_l\right).in_k \qquad *$$

Back-propagation with Momentum

$$delta_k^{(N)} = \left(targ_k - out_k^{(N)}\right)$$

$$delta_k^{(n)} = \left(\sum_k delta_k^{(n+1)}.w_{lk}^{(n+1)}\right).f'\left(\sum_j out_j^{(n-1)} w_{jk}^{(n)}\right)$$

$$\Delta w_{kl}^{(n)}(t) = \eta \sum_p delta_l^{(n)}(t).out_k^{(n-1)}(t) + \alpha.\Delta w_{kl}^{(n)}(t-1)$$

Regularization and Weight Decay/Sharing

$$E_{Reg} = E_{SSE/CE} + \lambda\Omega \qquad *$$

$$\Omega_{WeightDecay} = \tfrac{1}{2} \sum_{j,i,m} (w_{ji}^{(m)})^2 \qquad *$$

$$\Omega_{SoftWeightSharing} = -\sum_{k,l,m} \ln\left(\sum_{j=1}^{M} \frac{\alpha_j}{\sqrt{2\pi\sigma_j^2}} \exp\left\{ -\frac{(w_{kl}^{(m)} - \mu_j)^2}{2\sigma_j^2} \right\} \right)$$

Bias and Variance Decomposition

$$\mathcal{E}_D\left[\left(\mathcal{E}[y \mid x_i] - net(x_i, W, D) \right)^2 \right]$$

$$= \left(\mathcal{E}_D\left[net(x_i, W, D) \right] - \mathcal{E}[y \mid x_i] \right)^2 + \mathcal{E}_D\left[\left(net(x_i, W, D) - \mathcal{E}_D\left[net(x_i, W, D) \right] \right)^2 \right]$$

Radial Basis Function Networks

$$y_k(\mathbf{x}) = \sum_{j=0}^{M} w_{kj} \phi_j(\mathbf{x}) \qquad \phi_j(\mathbf{x}) = \exp\left(-\frac{\left\| \mathbf{x} - \boldsymbol{\mu}_j \right\|^2}{2\sigma_j^2} \right) \qquad *$$

K-Means Clustering / Vector Quantization

$$J = \sum_{j=1}^{K} \sum_{p \in S_j} \left\| \mathbf{x}^p - \boldsymbol{\mu}_j \right\|^2 \qquad \boldsymbol{\mu}_j = \frac{1}{N_j} \sum_{p \in S_j} \mathbf{x}^p \qquad *$$

Kohonen Networks – Discriminant Function and Weight Update Equation

$$d_j(\mathbf{x}) = \sum_{i=1}^{D} (x_i - w_{ji})^2 \qquad *$$

$$T_{j,I(\mathbf{x})}(t) = \exp(-S_{j,I(\mathbf{x})}^2 / 2\sigma^2(t)) \qquad \sigma(t) = \sigma_0 \exp(-t/\tau_\sigma) \qquad *$$

$$\Delta w_{ji} = \eta(t).T_{j,I(\mathbf{x})}(t).(x_i - w_{ji}) \qquad \eta(t) = \eta_0 \exp(-t/\tau_\eta) \qquad *$$

Mixtures of Experts – Gated Combination and Softmax

$$y(p) = \sum_{i=1}^{K} g_i(\mathbf{x}(p)) y_i(p) \qquad *$$

$$g_i(p) = \exp\left(\sum_j a_{ij} x_j(p) \right) \Bigg/ \sum_{l=1}^{K} \exp\left(\sum_j a_{lj} x_j(p) \right) \qquad *$$