PA2 Individual Report

Zhenrui Yue
Computer Science & Engineering
UC San Diego
La Jolla, CA 92093
yuezrhb@gmail.com

1 Maximum Likelihood Estimation

Given the probability function:

$$p(x;\lambda) = \lambda \exp(-\lambda x)$$

To find the maximum likelihood estimate for the parameter λ for n samples, we introduce the log likelihood L:

$$L = \prod_{i=1}^{n} \log(\lambda \exp(-\lambda x_i)) = \prod_{i=1}^{n} \log(\lambda) + (-\lambda x_i) = n\log(\lambda) - \lambda \sum_{i=1}^{n} x_i$$

To maximize L is the same as to minimize $-L$:

$$-L = -n\log(\lambda) + \lambda \sum_{i=1}^{n} x_i$$

We differentiate $-L$ with respect to λ:

$$\frac{\partial(-L)}{\partial\lambda} = -\frac{n}{\lambda} + \sum_{i=1}^{n} x_i$$

$$\frac{\partial^2(-L)}{\partial\lambda^2} = \frac{n}{\lambda^2} > 0$$

Let the first equation above equal 0, we have:

$$\lambda = \frac{n}{\sum_{i=1}^{n} x_i}$$

Since the second derivative is greater than 0, the λ value above proves to be the minimum of $-L$, therefore this is the λ value for maximum likelihood estimation.

2 Multiclass Classification

a) Derivation

Given the cross-entropy loss and softmax function below:

$$E = -\sum_{k} t_k \log(y_k)$$

$$y_i = \frac{e^{a_i}}{\sum e^{a_k}}$$

We first compute the derivative of the softmax function with respect to a_i:

$$\frac{\partial y_i}{\partial a_i} = \frac{e^{a_i}\sum e^{a_k} - e^{a_i}e^{a_i}}{(\sum e^{a_k})^2} = \frac{e^{a_i}}{\sum e^{a_k}} \times \frac{(\sum e^{a_k} - e^{a_i})}{\sum e^{a_k}} = y_i(1-y_i)$$

We also compute the derivative of the softmax function with respect to a_j:

$$\frac{\partial y_i}{\partial a_j} = \frac{-e^{a_i}e^{a_j}}{(\sum e^{a_k})^2} = -\frac{e^{a_i}}{\sum e^{a_k}} \times \frac{e^{a_j}}{\sum e^{a_k}} = -y_i y_j$$

Now we compute the derivative of the cross-entropy loss with respect to a_i:

$$\begin{aligned}\frac{\partial E}{\partial a_i} &= -\sum_k t_k \frac{\partial \log(y_k)}{\partial a_i} \\ &= -\sum_k t_k \frac{\partial \log(y_k)}{\partial y_k} \times \frac{\partial y_k}{\partial a_i} \\ &= -\sum_k \frac{t_k}{y_k} \times \frac{\partial y_k}{\partial a_i}\end{aligned}$$

Plug in the first 2 computed derivatives, we have:

$$\begin{aligned}\frac{\partial E}{\partial a_i} &= -\frac{t_i}{y_i} \times \frac{\partial y_i}{\partial a_i} - \sum_{k \neq i} \frac{t_k}{y_k} \times \frac{\partial y_k}{\partial a_i} \\ &= -t_i(1-y_i) - \sum_{k \neq i} \frac{-t_k y_k y_i}{y_k} \\ &= -t_i + \sum_k t_k y_i = y_i - t_i\end{aligned}$$

Therefore, for the output layer we have (note that notation for output layer is k):

$$\delta_k = -\frac{\partial E}{\partial a_k} = t_k - y_k$$

For the hidden layer we first compute the derivative of $\tanh(x)$:

$$\begin{aligned}\frac{\partial \tanh(x)}{\partial x} &= \frac{(e^x + e^{-x})(e^x + e^{-x}) - (e^x - e^{-x})(e^x - e^{-x})}{(e^x + e^{-x})^2} \\ &= 1 - \frac{(e^x - e^{-x})^2}{(e^x + e^{-x})^2} = 1 - tanh^2(x)\end{aligned}$$

So, we have for the hidden layer, note y_i represents the class of true label:

$$\frac{\partial y_i}{\partial z_j} = \sum_k \frac{\partial y_i}{\partial a_k}\frac{\partial a_k}{\partial z_j} = y_i(1-y_i)w_{ji} - \sum_{k \neq i} y_i y_k w_{jk}$$

$$\begin{aligned}\delta_j &= -\frac{\partial E}{\partial a_j} = -\frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial z_j}\frac{\partial z_i}{\partial a_j} \\ &= \frac{1}{y_i} \times y_i\left(1 - y_i w_{ji} - \sum_k y_k w_{jk}\right)\left(1 - tanh^2(a_j)\right) \\ &= (1 - \sum_k y_k w_{jk})(1 - tanh^2(a_j))\end{aligned}$$

There for the result for the hidden layer is:

$$\delta_j = (1 - \sum_k y_k w_{jk})(1 - tanh^2(a_j))$$

b) Update Rule

For the output layer with n data samples, note $\frac{\partial E^n}{\partial a_k}$ was already computed in (a), and y_i represents the output of the class with true label:

$$w_{jk} = w_{jk} - \alpha \frac{\partial E}{\partial w_{jk}} = w_{jk} - \alpha \frac{\partial E}{\partial y_i} \frac{\partial y_i}{\partial a_k} \frac{\partial a_k}{\partial w_{jk}} = w_{jk} - \alpha \sum_n \frac{\partial E^n}{\partial a_k} \frac{\partial a_k}{\partial w_{jk}}$$

Whereas for the input layer, $\frac{\partial E^n}{\partial a_k}$ could also be plugged in:

$$w_{ij} = w_{ij} - \alpha \frac{\partial E}{\partial w_{jk}} = w_{jk} - \alpha \frac{\partial E}{\partial y_i} (\sum_k \frac{\partial y_i}{\partial a_k} \frac{\partial a_k}{\partial z_j}) \frac{\partial z_j}{\partial a_j} \frac{\partial a_j}{\partial w_{ij}} = w_{jk} - \alpha \sum_n \frac{\partial E^n}{\partial a_k} \frac{\partial a_k}{\partial z_j} \frac{\partial z_j}{\partial a_j} \frac{\partial a_j}{\partial w_{ij}}$$

c) Vectorize computation

For matrix computation we have following equations, with $\boldsymbol{T}$ representing the one-hot coded label matrix, notice that it's matrix multiplication between matrix and matrix (vector) but element-wise computation with functions:

$$E = -\boldsymbol{T} \log(\boldsymbol{y})$$

$$\boldsymbol{y} = \boldsymbol{softmax}(\boldsymbol{a_k})$$

$$\boldsymbol{a_k} = \boldsymbol{W_k^T z_j}$$

$$\boldsymbol{z_j} = \boldsymbol{tanh}(\boldsymbol{a_j})$$

$$\boldsymbol{a_j} = \boldsymbol{W_j^T x}$$

The backpropagation would be in the following order, with y_i representing the calculated probability of the class with the correct label,

$$\frac{\partial E}{\partial \boldsymbol{W_k}} = \frac{\partial E}{\partial \boldsymbol{y}} \frac{\partial \boldsymbol{y}}{\partial \boldsymbol{a_k}} \frac{\partial \boldsymbol{a_k}}{\partial \boldsymbol{W_k}} = -\boldsymbol{T} \frac{1}{y} \frac{\partial \boldsymbol{y}}{\partial \boldsymbol{a_k}} \frac{\partial \boldsymbol{a_k}}{\partial \boldsymbol{W_k}} = -\frac{1}{y_i} \frac{\partial y_i}{\partial \boldsymbol{a_k}} \boldsymbol{z_j}$$