Machine Learning Problem Set 3, Fall 2018

Jianqiao Hao, 2018211670
School of Economics and Management
hjq18@mails.tsinghua.edu.cn

Oct 13th, 2018

1. Note: The matrix representation of the multi-layer network model is with reference to(but not the same as) Neural Networks and Deep Learning - Coursera.

   Proof: Assume that there are h hidden layers in this multi-layer neural network model, and we denote the layer from the input layer to the output layer as layer 0, 1, 2, $\cdots$, $h+1$. Assume that there are n_i nodes in each layer, where $i = 0, 1, 2, \cdots, h+1$. For layer i, we denote $\mathbf{W}^{[i]}$ as

   $$\mathbf{W}^{[i]} = \begin{pmatrix} w_{11}^{[i]} & w_{12}^{[i]} & \dots & w_{1n_{i-1}}^{[i]} & b_1^{[i]} \\ \vdots & & & & \vdots \\ w_{n_i1}^{[i]} & w_{n_i2}^{[i]} & \dots & w_{n_in_{i-1}}^{[i]} & b_{n_i}^{[i]} \end{pmatrix}$$

   which is the augmented weight matrix from layer $i-1$ to layer i. Also Denote $\mathbf{z}^{[i]}$ as the net-value vector of layer i and denote $\mathbf{a}^{[i]}$ to be $f(\mathbf{z}^{[i]})$. For simplicity in the representation, we denote $\mathbf{a}^{[0]}$ to be the input feature $\mathbf{x}$. Since we assume that $f(\cdot)$ is chosen as a linear activation function, we just assume that $f(x) = cx + t$, then we denote $\mathbf{C}^{[i]}$ as an $n_i \times 1$ vector with all element equals c, and $\mathbf{T}^{[i]}$ as an $n_i \times 1$ vector with all element equals t. Then we can calculate each node as below

   $$\begin{aligned} \mathbf{a}^{[0]} &= \mathbf{x} \\ \mathbf{z}^{[1]} &= \mathbf{W}^{[1]}\mathbf{a}^{[0]} = \mathbf{W}^{[1]}\mathbf{x} \\ \mathbf{a}^{[1]} &= \mathbf{C}^{[1]}\mathbf{z}^{[1]} + \mathbf{T}^{[1]} = \mathbf{C}^{[1]}\mathbf{W}^{[1]}\mathbf{x} + \mathbf{T}^{[1]} \\ \mathbf{z}^{[2]} &= \mathbf{W}^{[2]}\mathbf{a}^{[1]} = \mathbf{W}^{[2]}\mathbf{C}^{[1]}\mathbf{W}^{[1]}\mathbf{x} + \mathbf{W}^{[2]}\mathbf{T}^{[1]} \\ \mathbf{a}^{[2]} &= \mathbf{C}^{[2]}\mathbf{z}^{[2]} + \mathbf{T}^{[2]} = \mathbf{C}^{[2]}\mathbf{W}^{[2]}\mathbf{C}^{[1]}\mathbf{W}^{[1]}\mathbf{x} + \mathbf{C}^{[2]}\mathbf{W}^{[2]}\mathbf{T}^{[1]} + \mathbf{T}^{[2]} \\ &\dots \end{aligned}$$

   Therefore, we can write the result of the output layer as

   $$\mathbf{a}^{[h+1]} = \mathbf{C}^{[h+1]}\mathbf{W}^{[h+1]}\mathbf{C}^{[h]}\mathbf{W}^{[h]} \cdots \mathbf{C}^{[1]}\mathbf{W}^{[1]}\mathbf{x} + \theta,$$

   where θ is a vector(if the number of nodes in the output layer is greater than 1) or a scaler(if there is only one node in the output layer) which is independent of $\mathbf{x}$. Therefore, if the activation function is chose to be a linear function for all nodes, then the output is still a linear combination of original feature $\mathbf{x}$, which means the nonlinearity will never be achieved.

2. Solution: Just as the figure showed in the problem, assume that dimension for the four layer is n, n_1, n_2, m respectively. Just following the notations in the lecture slide, if we do the

forward calculation, we could have the output node*

$$y_l = f\Big(\sum_{k=1}^{n_2} w_{kl} f\big(\sum_{j=1}^{n_1} w_{jk} f(\sum_{i=1}^{n} w_{ij} x_i)\big)\Big), \quad l = 1, 2, \cdots, m.$$

We can construct our objective function as

$$J = \frac{1}{2}\sum_{l=1}^{m}(y_l - d_l)^2,$$

where d_l is the target value for each output node.

Then we can calculate the derivative/gradient for the weights from the second hidden layer to the output layer. For each $k = 1, 2, \cdots, n_2$ and $l = 1, 2, \cdots, m$, we have

$$\begin{aligned}\frac{\partial J}{\partial w_{kl}} &= \frac{\partial J}{\partial y_l}\frac{\partial y_l}{\partial net_l}\frac{\partial net_l}{w_{kl}} \\ &= (y_l - d_l) f'(net_l) O_k\end{aligned}$$

where $O_k = f(net_k)$.

So according to the gradient descent, we have

$$\Delta w_{kl} = -\eta \frac{\partial J}{\partial w_{kl}} = \eta(d_l - y_l) f'(net_l) O_k.$$

If we denote $\delta_l = (d_l - y_l) f'(net_l)$, then we can write

$$\Delta w_{kl} = \eta \delta_l O_k.$$

We then go on to calculate the gradients for the weights from the first hidden layer to the output layer. For each $j = 1, 2, \cdots, n_1$ and $k = 1, 2, \cdots, n_2$, we have

$$\begin{aligned}\frac{\partial J}{\partial w_{jk}} &= \frac{\partial J}{\partial O_k}\frac{\partial O_k}{\partial w_{jk}} \\ &= \sum_{l=1}^{m}(y_l - d_l) f'(net_l) w_{kl} f'(net_k) O_j \\ &= -f'(net_k)\sum_{l=1}^{m} w_{kl}\delta_l O_j\end{aligned}$$

We can also denote $\delta_k = f'(net_k)\sum_{l=1}^{m} w_{kl}\delta_l$, then

$$\Delta w_{jk} = \eta \delta_k O_j.$$

Finally, we can calculate the weights between the input layer and the first hidden layer. For each $i = 1, 2, \cdots, n$ and $j = 1, 2, \cdots, n_1$, we have

$$\begin{aligned}\frac{\partial J}{\partial w_{ij}} &= \sum_{l=1}^{m}(y_l - d_l) f'(net_l)\sum_{k=1}^{n_2} w_{kl} f'(net_k) w_{jk} f'(net_j) x_i \\ &= -f'(net_j)\sum_{k=1}^{n_2} w_{jk}\delta_k x_i.\end{aligned}$$

*It seems that the derivation in the lecture ignores the intercept/threshold b and I just follow the derivation in the lecture. The result should also be correct if we just regard $\mathbf{W}$ as the augmented weight matrix and we can also add a constant node into each layer to simulate the b term.

Similarly, if we denote $\delta_j = f'(net_j)\sum_{k=1}^{n_2} w_{jk}\delta_k$, we have

$$\Delta w_{ij} = \eta\delta_j x_i.$$

Based on the calculations above, we could give the pseudo-codes of BP algorithm for four-layer network:

- Step 1: Initialize all the weights with some small random numbers,
- Step 2: For a given training sample $\mathbf{x} = (x_1, x_2, \cdots, x_n)^T$, calculate the output $\mathbf{Y} = (y_1, y_2, \cdots, y_m)^T$,
- Step 3: Gradient descent using desired output $\mathbf{D} = (d_1, d_2, \cdots, d_m)^T$. For the weights between the second hidden layer and the output layer, use the formula

  $$w_{kl}(t+1) = w_{kl}(t)\eta\delta_l O_k,$$

  for the weights between the first hidden layer and the second hidden layer, use the formula

  $$w_{jk}(t+1) = w_{jk}(t)\eta\delta_k O_j,$$

  and for the weights between the input layer and the first hidden layer, use the formula

  $$w_{ij}(t+1) = w_{ij}(t)\eta\delta_j x_i.$$

- Step 4: Loop, go to Step 2 with another training sample until the stop criterion is met.