U19CS009

Brijesh Rohit

Q-8D Back propogation algorithm makes a backward pass after each forward pass in a network.
This is used to train a neural network using chain rule method.

→ Forward propogation has following eq^ns.

1. $x = a^{(1)}$ → input layer
2. $z^{(2)} = W^{(1)}(x) + b^{(1)}$ → neuron value at hidden layer no. 1
3. $a^{(2)} = f(z^2)$ → activation value at Hidden layer 1
4. $z^{(3)} = W^{(2)}a^{(2)} + b^{(2)}$ → this indicates neuron value at Hidden layer 2
5. $a^{(3)} = f(z^3)$ → this f is for activation at hidden layer 2
6. $s = W^{(3)}a^{(3)}$ → this is output.

Back propogation aims to minimize the cost function

$$C = \text{cost}(s, y).$$

by changing the weights and biases in network, this change is measured by gradients of cost function.

$$\frac{\partial C}{\partial x} = \left[\frac{\partial C}{\partial x_1}, \frac{\partial C}{\partial x_2}, \dots \right]$$

This derivitive measures sensitivity to change & direction, meaning how input x is to be changed is determined by this.

→ using chain rule we get

$$\frac{\partial C}{\partial x} = \frac{\partial C}{\partial z} \cdot \frac{\partial z}{\partial x}$$

$$= \frac{\partial C}{\partial z} \cdot W$$

→ $z = W(x) + b$ (2)

$$\frac{\partial z}{\partial x} = W$$

U19CS009

Brijesh Rohit

→ activation of i^{th} unit in input x ⇒ $x_i = a_i(x)$

→ ——— j^{th} ——— hidden layer

⇒ say $x_j^h = \sum_{i=1}^{I} w_{ji}^h x_i$

→ O/p signal ⇒ $s_j^h = f_j^h(x_j^h)$

from hidden layer.

of j^{th} unit

→ Then we activate unit k in O/p layer.

$$\therefore x_k^{output} = \sum_{j=1}^{J} w_{kj}^{output} s_j^h$$

↓ denotes newron value

→ Then the output of k^{th} unit in output layer

$$s_k^{output} = f_k^{output}(x_k^{output})$$

→ error for k^{th} O/p → $\delta_k^{output} = (b_k - s_k^{output}) f_k^{output}$

→ we update weight as → $w_{kj}^{output}(m+1) = w_{kj}^{output}(m) + \eta \delta_k^{output} s_j^h$

on output layer

→ Error in hidden layer of j^{th} unit

$$\delta_j^h = f_j^h \sum_{k=1}^{K} \delta_k^{output} w_{kj}^{output}$$

→ update weights on hidden layer →

$$w_{ji}^h(m+1) = w_{ji}^h(m) + \eta \delta_j^h a_i$$

→ We apply this given algo one by one, multiple times in unorderly manner, and change values until total error reduces to acceptable value