CSC 578-701 Loop Quiz #2
Lavinia Wang #14737304

Q1: Derive a gradient descent training rule for a single unit with output o, where

$$o = w_0 + w_1 x_1 + w_1 {x_1}^2 + \cdots + w_n x_n + w_n {x_n}^2$$

Solution

First, the error function is defined as: $E(\vec{w}) = \frac{1}{2}\sum_{d\in D}(t_d - o_d)^2$
The update rule is the same, namely: $w_i = w_i + \Delta w_i$

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$

For w_0,

$$\frac{\partial E}{\partial w_0} = \frac{\partial}{\partial w_0}\frac{1}{2}\sum_{d\in D}(t_d - o_d)^2 = \frac{1}{2}\sum_{d\in D}\frac{\partial}{\partial w_0}(t_d - o_d)^2$$
$$= \frac{1}{2}\sum_{d\in D} 2(t_d - o_d)\frac{\partial}{\partial w_0}(t_d - o_d) = \sum_{d\in D}(t_d - o_d)\cdot(-1) = -\sum_{d\in D}(t_d - o_d)$$

Thus,

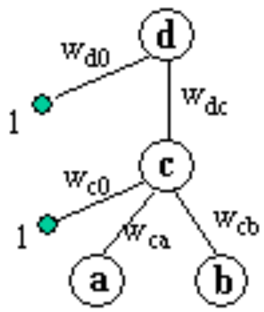
$$\Delta w_0 = \eta \sum_{d\in D}(t_d - o_d)$$

For $w_1, w_2, \dots, w_n$

$$\frac{\partial E}{\partial w_0} = \frac{\partial}{\partial w_0}\frac{1}{2}\sum_{d\in D}(t_d - o_d)^2 = \frac{1}{2}\sum_{d\in D}\frac{\partial}{\partial w_0}(t_d - o_d)^2$$
$$= \frac{1}{2}\sum_{d\in D} 2(t_d - o_d)\frac{\partial}{\partial w_0}(t_d - o_d) = \sum_{d\in D}(t_d - o_d)\cdot(-(x_{id} + {x_{id}}^2))$$

$$\Delta w_i = \eta \sum_{d\in D}(t_d - o_d)(x_{id} + {x_{id}}^2)$$

Q2: Consider a two-layer feedforward ANN with two inputs a and b, one hidden unit c, and one output unit d. This network has five weights $(w_{ca}, w_{cb}, w_{c0}, w_{dc}, w_{d0})$, where w_{x0} represents the threshold weight for unit x. Initialize these weights to the values (.1, .1, .1, .1, .1), then give their values after each of the first two training iterations of the BACKPROPAGATION algorithm. Assume learning rate η = .3, momentum α= 0.9, incremental weight updates, and the following training examples:



a	b	d
1	0	1
0	1	0

With the use of "threshold weight for unit x", you can assume the network as above, where the nodes in green are threshold units and their values are 1.

Solution

Training example 1 <<1, 0>, 1>

Step 1: Propagate forward: compute the activation of the nodes, noting that $a = 1$ and $b = 0$:

$$o_c = \frac{1}{1+e^{-[(w_{c0}\cdot 1)+(w_{ca}\cdot a)+(w_{cb})\cdot b]}} = \frac{1}{1+e^{-[(0.1*1)+(0.1*1)+(0.1)*0]}} = \frac{1}{1+e^{-0.2}} = 0.5498$$

$$o_d = \frac{1}{1+e^{-[(w_{d0}\cdot 1)+(w_{dc}\cdot c)]}} = \frac{1}{1+e^{-[(0.1*1)+(0.1*0.5498)]}} = \frac{1}{1+e^{-0.15498}} = 0.53867$$

Step 2: Propagate backward

First compute the error at each node, noting that $d = 1$:

$$\delta_d = o_d \cdot (1-o_d) \cdot (t_d - o_d) = 0.53867 * (1-0.53867) * (1-0.53867) = 0.1146$$

$$\delta_c = o_c \cdot (1-o_c) \cdot w_{dc} \cdot \delta_d = 0.5498 * (1-0.5498) * 0.1 * 0.1146 = 0.002836$$

Compute the correction terms as follows, noting that $a = 1$, $b = 0$ and $\eta = 0.3$:

$$\Delta w_{d0}(1) = \eta \cdot \delta_d \cdot 1 + \alpha \cdot \Delta w_{d0} \cdot (1-1) = 0.3 * 0.1146 * 1 = 0.03438$$
$$\Delta w_{dc}(1) = \eta \cdot \delta_d \cdot c + \alpha \cdot \Delta w_{dc} \cdot (1-1) = 0.3 * 0.1146 * 0.5498 = 0.0189$$
$$\Delta w_{c0}(1) = \eta \cdot \delta_c \cdot 1 + \alpha \cdot \Delta w_{c0} \cdot (1-1) = 0.3 * 0.002836 * 1 = 0.00085$$
$$\Delta w_{ca}(1) = \eta \cdot \delta_c \cdot a + \alpha \cdot \Delta w_{ca} \cdot (1-1) = 0.3 * 0.002836 * 1 = 0.00085$$
$$\Delta w_{cb}(1) = \eta \cdot \delta_c \cdot b + \alpha \cdot \Delta w_{cb} \cdot (1-1) = 0.3 * 0.002836 * 0 = 0$$

and the new weights become:

$w_{d0} = 0.1 + 0.03438 = 0.13438$

$w_{dc} = 0.1 + 0.0189 = 0.1189$

$w_{c0} = 0.1 + 0.00085 = 0.10085$
$w_{ca} = 0.1 + 0.00085 = 0.10085$
$w_{cb} = 0.1 + 0 = 0.1$

Training example 2 <<0, 1>, 0>

Step 1: Propagate forward: compute the activation of the nodes, noting that $a = 0$ and $b = 1$:

$$o_c = \frac{1}{1 + e^{-[(w_{c0} \cdot 1) + (w_{ca} \cdot a) + (w_{cb}) \cdot b]}} = \frac{1}{1 + e^{-[(0.10085 * 1) + (0.10085 * 0) + (0.1) * 1]}} = \frac{1}{1 + e^{-0.20085}} = 0.55$$

$$o_d = \frac{1}{1 + e^{-[(w_{d0} \cdot 1) + (w_{dc} \cdot c)]}} = \frac{1}{1 + e^{-[(0.13438 * 1) + (0.1189 * 0.55)]}} = \frac{1}{1 + e^{-0.1998}} = 0.5498$$

Step 2: Propagate backward

First compute the error at each node, noting that $d = 0$

$$\delta_d = o_d \cdot (1 - o_d) \cdot (t_d - o_d) = 0.5498 * (1 - 0.5498) * (0 - 0.5498) = -0.1361$$

$$\delta_c = o_c \cdot (1 - o_c) \cdot w_{dc} \cdot \delta_d = 0.55 * (1 - 0.55) * 0.1189 * (-0.1361) = -0.004$$

Compute the correction terms as follows, noting that $a = 0$, $b = 1$, $\eta = 0.3$ and $\alpha = 0.9$:

$\Delta w_{d0}(2) = \eta \cdot \delta_d \cdot 1 + \alpha \cdot \Delta w_{d0} \cdot (2 - 1) = 0.3 * (-0.1361) * 1 + 0.9 * 0.03438 = -0.01$
$\Delta w_{dc}(2) = \eta \cdot \delta_d \cdot c + \alpha \cdot \Delta w_{dc} \cdot (2 - 1) = 0.3 * (-0.1361) * 0.55 + 0.9 * 0.0189 = -0.0055$
$\Delta w_{c0}(2) = \eta \cdot \delta_c \cdot 1 + \alpha \cdot \Delta w_{c0} \cdot (2 - 1) = 0.3 * (-0.004) * 1 + 0.9 * 0.00085 = -0.0004$
$\Delta w_{ca}(2) = \eta \cdot \delta_c \cdot a + \alpha \cdot \Delta w_{ca} \cdot (2 - 1) = 0.3 * (-0.004) * 0 + 0.9 * 0.00085 = 0.00077$
$\Delta w_{cb}(2) = \eta \cdot \delta_c \cdot b + \alpha \cdot \Delta w_{cb} \cdot (2 - 1) = 0.3 * (-0.004) * 1 + 0.9 * 0 = -0.0012$

and the new weights become:
$w_{d0} = 0.13438 - 0.01 = 0.12438$
$w_{dc} = 0.1189 - 0.0055 = 0.1134$
$w_{c0} = 0.10085 - 0.0004 = 0.10045$
$w_{ca} = 0.10085 + 0.00077 = 0.10162$
$w_{cb} = 0.1 - 0.0012 = 0.0988$

CSC 578-701 Loop Quiz #2
Lavinia Wang #14737304

Q3: Revise the BACKPROPAGATION algorithm in Table 4.2 so that it operates on units using the squashing function $tanh$ in place of the sigmoid function. That is, assume the output of a single unit is $o = tanh(\vec{w} \cdot \vec{x})$. Give the weight update rule for output layer weights and hidden layer weights. Hint: $tanh'(x) = 1 - tanh^2(x)$.

Solution

$$E\left(a\big(z(w,x)\big)\right) where\ a = \tan h(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}\ and\ a'(z) = 1 - tanh^2(z)$$

$$\frac{\partial E}{\partial w_i} = \frac{\partial E}{\partial a} \cdot \frac{\partial a}{\partial z} \cdot \frac{\partial z}{\partial w_i} = \sum_{d \in D} (t_d - o_d)(1 - {o_d}^2) \cdot x_i$$

Propagate the input forward through the network:

1. Input the instance $\vec{x}$ to the network and compute the output o_u of every unit *u* in the network.

Propagate the error backward through the network:

2. For each network output unit *k*, calculate its error term δ_k

$$\delta_k \leftarrow (t_d - o_d) \cdot (1 - {o_d}^2)$$

3. For each hidden unit *h*, calculate its error term δ_h

$$\delta_h \leftarrow (1 - {o_h}^2) \sum_{k \in outputs} w_{kh} \delta_h$$

4. Update each network weight w_{ji}

$$w_{ji} \leftarrow w_{ji} + \Delta w_{ji}$$

where

$$\Delta w_{ji} = \eta \delta_j x_{ji}$$