Neural Network II

Chul Min Yeum
Assistant Professor
Civil and Environmental Engineering
University of Waterloo, Canada

CIVE 497 – CIVE 700: Smart Structure Technology

Last updated: 2021-04-05

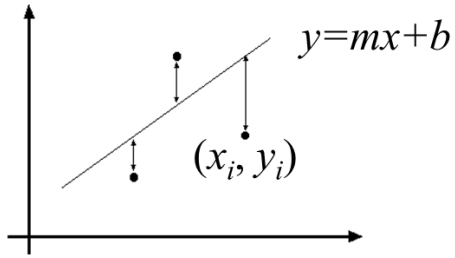
UNIVERSITY OF WATERLOO
FACULTY OF ENGINEERING

Recall: Linear Regression

Data (measurement): $(x_1, y_1), \ldots, (x_n, y_n)$

Model: Line $f(x_i, m, b) = mx_i + b$

Task: Find (m, b)



Minimize $E = J(m, b) = \sum_{i=1}^{n} (y_i - f(x_i, m, b))^2 = \sum_{i=1}^{n} (y_i - mx_i - b)^2$

$$J(\theta) = J(\theta^1, \theta^2) = \sum_{i=1}^{n} (y_i - \theta^1 x_i - \theta^2)^2$$

$$\frac{\partial J(\theta^1, \theta^2)}{\partial \theta^1} = -2 \sum_{i=1}^{n} [y_i - \theta^1 x_i - \theta^2] x_i \qquad \theta_{j+1}^1 \leftarrow \theta_j^1 - \alpha \frac{\partial}{\partial \theta_j^1} J(\theta)$$

$$\frac{\partial J(\theta^1, \theta^2)}{\partial \theta^2} = -2 \sum_{i=1}^{n} [y_i - \theta^1 x_i - \theta^2] \qquad \theta_{j+1}^2 \leftarrow \theta_j^2 - \alpha \frac{\partial}{\partial \theta_j^2} J(\theta)$$

Recall: Linear Regression (Continue)

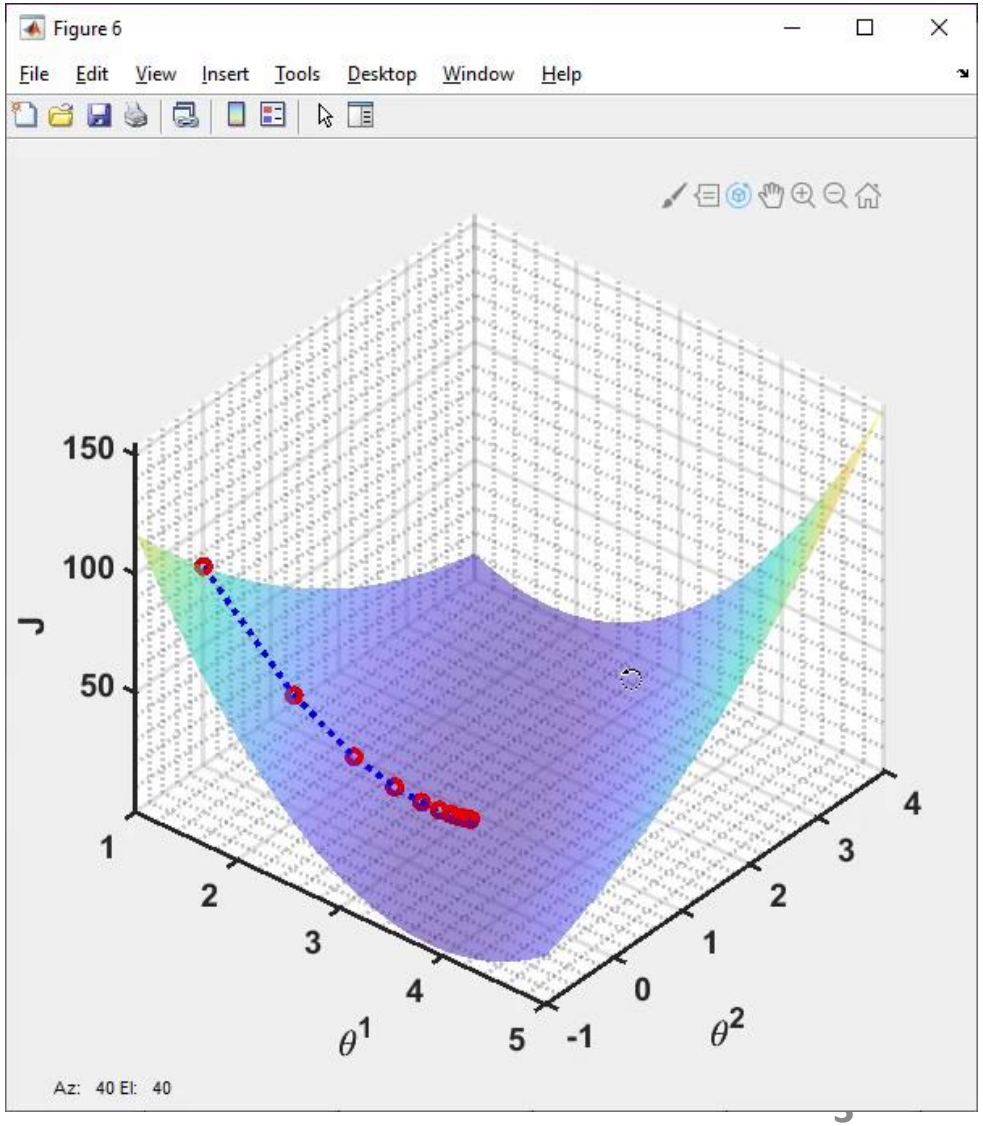
Gradient Descent

$$\theta_{j+1} \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

Repeat until convergence

$$\frac{\partial J(\theta^1, \theta^2)}{\partial \theta^1} = -2 \sum_{i=1}^{n} [y_i - \theta^1 x_i - \theta^2] x_i \qquad \theta_{j+1}^1 \leftarrow \theta_j^1 - \alpha \frac{\partial}{\partial \theta_j^1} J(\theta)$$

$$\frac{\partial J(\theta^1, \theta^2)}{\partial \theta^2} = -2 \sum_{i=1}^{n} [y_i - \theta^1 x_i - \theta^2] \qquad \theta_{j+1}^2 \leftarrow \theta_j^2 - \alpha \frac{\partial}{\partial \theta_j^2} J(\theta)$$



Backward Pass

Our goal with backpropagation is **to update each of the weights in the network** so that they cause the actual output to be closer the target output, thereby minimizing the error for each output neuron and the network as a whole.

$$\theta_{j+1} \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

How to find $\frac{\partial}{\partial \theta_j} J(\theta)$ to update the parameter θ ?

Chain Rule

Given a multivariable function $f(x, y)$, and two single variable functions $x(t)$ and $y(t)$, here's what the multivariable chain rule says:

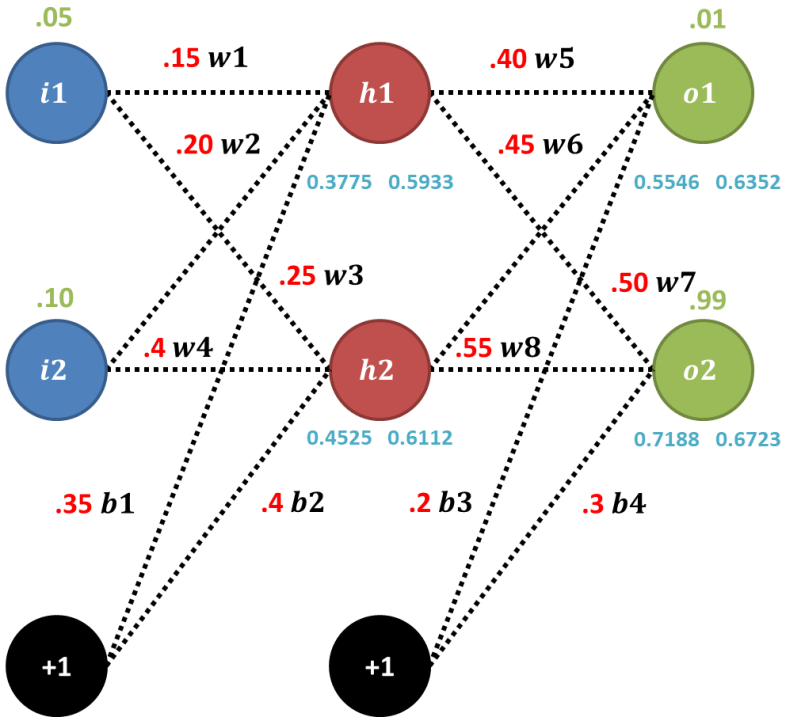
$$\underbrace{\frac{d}{dt} f(x(t), y(t))}_{\text{Derivative of composition function}} = \frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial y}\frac{dy}{dt}$$

The single variable chain rule tells you how to take the derivative of the composition of two functions:

$$\frac{d}{dt} f(g(t)) = \frac{df}{dg}\frac{dg}{dt} = f'(g(t))g'(t)$$

https://www.khanacademy.org/math/multivariable-calculus/multivariable-derivatives/differentiating-vector-valued-functions/a/multivariable-chain-rule-simple-version

Backpropagation (w_5)



$$\theta_{j+1} \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

$$w_5 \leftarrow w_5 - \alpha \frac{\partial E}{\partial w_5}$$

$$E = J(\mathbf{w}, \mathbf{b})$$

$$= (o_1 - f(i_1, i_2, \mathbf{w}, \mathbf{b})_1)^2 + (o_2 - f(i_1, i_2, \mathbf{w}, \mathbf{b})_2)^2$$

$$= (o_1 - out_{o1})^2 + (o_2 - out_{o2})^2$$

$$out_{o1} = f(net_{o1}) = \frac{1}{1 + e^{-net_{o1}}}$$

$$net_{o1} = w_5 out_{h1} + w_6 out_{h2} + b_3$$

$$\frac{\partial E}{\partial w_5} = \frac{\partial E}{\partial out_{o1}} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial net_{o1}}{\partial w_5} + \frac{\partial E}{\partial out_{o2}} \frac{\partial out_{o2}}{\partial net_{o2}} \frac{\partial net_{o2}}{\partial w_5}$$

0

$$= \frac{\partial E}{\partial out_{o1}} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial net_{o1}}{\partial w_5}$$

Backpropagation (w_5)

$$\frac{\partial E}{\partial w_5} = \frac{\partial E}{\partial out_{o1}} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial net_{o1}}{\partial w_5}$$

$$E = J(\mathbf{w}, \mathbf{b})$$
$$= (o_1 - f(i_1, i_2, \mathbf{w}, \mathbf{b})_1)^2$$
$$+ (o_2 - f(i_1, i_2, \mathbf{w}, \mathbf{b})_2)^2$$
$$= (o_1 - out_{o1})^2 + (o_2 - out_{o2})^2$$

$$out_{o1} = f(net_{o1}) = \frac{1}{1 + e^{-net_{o1}}}$$

$$net_{o1} = w_5 out_{h1} + w_6 out_{h2} + b_3$$

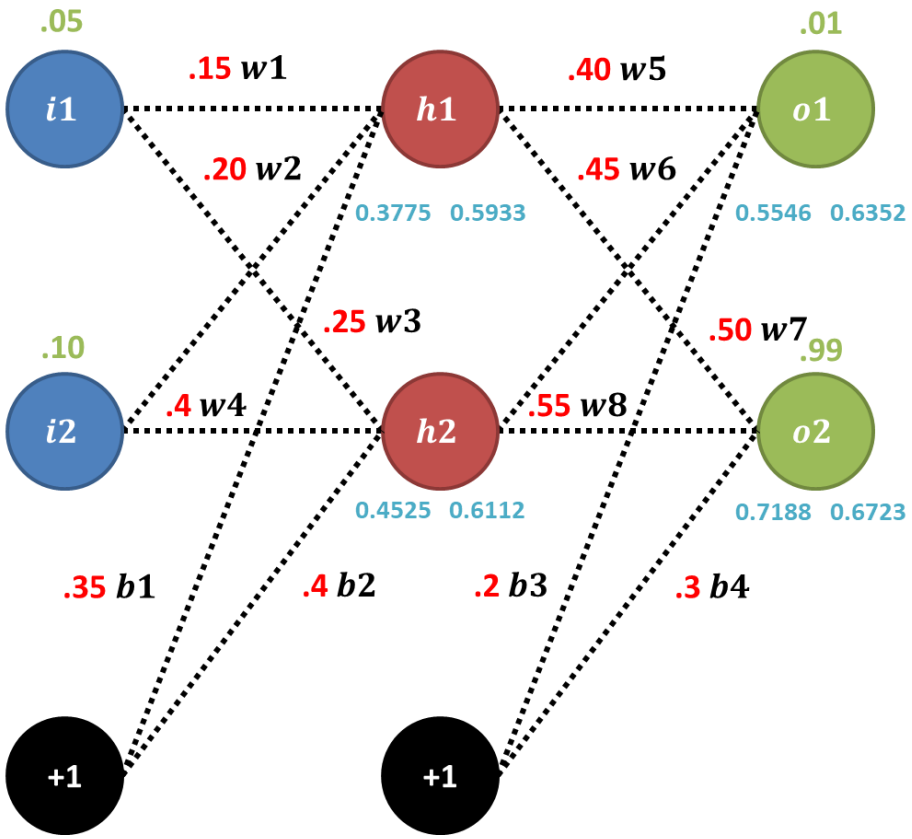
$$\frac{\partial E}{\partial out_{o1}} = -2(o_1 - out_{o1})$$

$$\frac{\partial out_{o1}}{\partial net_{o1}} = f(net_{o1}) * f(-net_{o1})$$

$$\frac{\partial net_{o1}}{\partial w_5} = out_{h1}$$

$$f(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{1 + e^x},$$
$$\frac{\mathrm{d}}{\mathrm{d}x} f(x) = \frac{e^x \cdot (1 + e^x) - e^x \cdot e^x}{(1 + e^x)^2} = \frac{e^x}{(1 + e^x)^2} = f(x)(1 - f(x)) = f(x)f(-x).$$

Backpropagation (w_5)



$$\theta_{j+1} \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

$$w_5 \leftarrow w_5 - \alpha \frac{\partial E}{\partial w_5}$$

$$\frac{\partial E}{\partial out_{o1}} = -2(o_1 - out_{o1}) = -2(0.01 - 0.6352)$$
$$= 1.2504$$

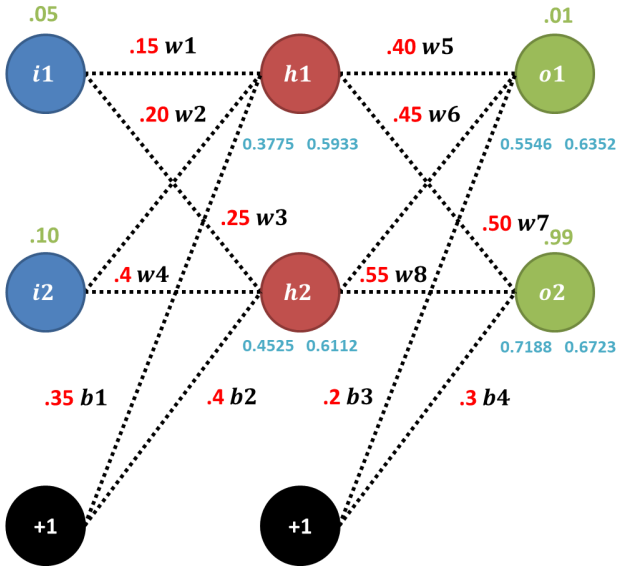
$$\frac{\partial out_{o1}}{\partial net_{o1}} = f(net_{o1}) * f(-net_{o1})$$
$$= f(0.5546) * f(-0.5546) = 0.2317$$

$$\frac{\partial net_{o1}}{\partial w_5} = out_{h1} = 0.5933$$

$$\frac{\partial E}{\partial w_5} = \frac{\partial E}{\partial out_{o1}} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial net_{o1}}{\partial w_5}$$

$$= 1.2504 * 0.2317 * 0.5933 = 0.1719$$

Backpropagation (w_1)



$$E = J(\mathbf{w}, \mathbf{b}) = (o_1 - out_{o1})^2 + (o_2 - out_{o2})^2$$

$$out_{o1} = f(net_{o1}) = \frac{1}{1 + e^{-net_{o1}}} \qquad net_{o2} = w_7 out_{h1} + w_8 out_{h2} + b_4$$

$$net_{o1} = w_5 out_{h1} + w_6 out_{h2} + b_3 \qquad out_{o2} = f(net_{o2}) = \frac{1}{1 + e^{-net_{o2}}}$$

$$net_{h1} = w_1 i_1 + w_2 i_2 + b_1 \qquad net_{h2} = w_3 i_1 + w_4 i_2 + b_2$$

$$out_{h2} = f(net_{h2}) = \frac{1}{1 + e^{-net_{h2}}}$$

$$out_{h1} = f(net_{h1}) = \frac{1}{1 + e^{-net_{h1}}}$$

$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial out_{o1}} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial net_{o1}}{\partial w_1} + \frac{\partial E}{\partial out_{o2}} \frac{\partial out_{o2}}{\partial net_{o2}} \frac{\partial net_{o2}}{\partial w_1}$$

$$= -2(o_1 - out_{o1}) * f(net_{o1}) * f(-net_{o1}) * \frac{\partial net_{o1}}{\partial w_1} + -2(o_2 - out_{o2}) * f(net_{o2}) * f(-net_{o2}) * \frac{\partial net_{o2}}{\partial w_1}$$

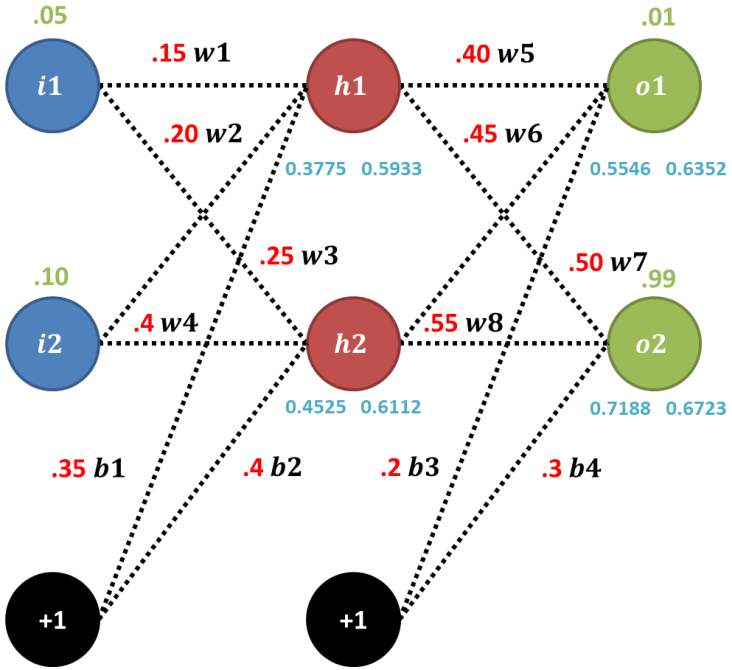
$$\frac{\partial net_{o1}}{\partial w_1} = \frac{\partial net_{o1}}{\partial out_{h1}} \frac{\partial out_{h1}}{\partial net_{h1}} \frac{\partial net_{h1}}{\partial w_1} + \frac{\partial net_{o1}}{\partial out_{h2}} \frac{\partial out_{h2}}{\partial net_{h2}} \frac{\partial net_{h2}}{\partial w_1} = \frac{\partial net_{o1}}{\partial out_{h1}} \frac{\partial out_{h1}}{\partial net_{h1}} \frac{\partial net_{h1}}{\partial w_1} = w_5 * f(net_{h1}) * f(-net_{h1}) * i_1$$

$$\frac{\partial net_{o2}}{\partial w_1} = \frac{\partial net_{o2}}{\partial out_{h1}} \frac{\partial out_{h1}}{\partial net_{h1}} \frac{\partial net_{h1}}{\partial w_1} + \frac{\partial net_{o2}}{\partial out_{h2}} \frac{\partial out_{h2}}{\partial net_{h2}} \frac{\partial net_{h2}}{\partial w_1} = \frac{\partial net_{o2}}{\partial out_{h1}} \frac{\partial out_{h1}}{\partial net_{h1}} \frac{\partial net_{h1}}{\partial w_1} = w_7 * f(net_{h1}) * f(-net_{h1}) * i_1$$

0

0

Backpropagation (w_1)



$$\theta^{j+1} \leftarrow \theta^{j} - \alpha \frac{\partial}{\partial \theta^{j}} J(\theta)$$

$$w_1 \leftarrow w_1 - \alpha \frac{\partial E}{\partial w_1}$$

$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial out_{o1}} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial net_{o1}}{\partial w_1} + \frac{\partial E}{\partial out_{o2}} \frac{\partial out_{o2}}{\partial net_{o2}} \frac{\partial net_{o2}}{\partial w_1}$$

$$= -2(o_1 - out_{o1}) * f(net_{o1}) * f(-net_{o1}) * \frac{\partial net_{o1}}{\partial w_1} +$$

$$-2(o_2 - out_{o2}) * f(net_{o2}) * f(-net_{o2}) * \frac{\partial net_{o2}}{\partial w_1}$$

$$= -2(0.01 - 0.6352) * f(0.5546) * f(-0.5546) * 0.0048 - 2(0.99 - 0.6723) * f(0.7188)$$

$$* f(-0.7188) * 0.006 = 5.5090e - 04$$

$$\frac{\partial net_{o1}}{\partial w_1} = \frac{\partial net_{o1}}{\partial out_{h1}} \frac{\partial out_{h1}}{\partial net_{h1}} \frac{\partial net_{h1}}{\partial w_1} + \frac{\partial net_{o1}}{\partial out_{h2}} \frac{\partial out_{h2}}{\partial net_{h2}} \frac{\partial net_{h2}}{\partial w_1} = \frac{\partial net_{o1}}{\partial out_{h1}} \frac{\partial out_{h1}}{\partial net_{h1}} \frac{\partial net_{h1}}{\partial w_1}$$

$$= w_5 * f(net_{h1}) * f(-net_{h1}) * i_1 = 0.4 * f(0.3775) * f(-0.3775) * 0.05 = 0.0048$$

$$\frac{\partial net_{o2}}{\partial w_1} = \frac{\partial net_{o2}}{\partial out_{h1}} \frac{\partial out_{h1}}{\partial net_{h1}} \frac{\partial net_{h1}}{\partial w_1} + \frac{\partial net_{o2}}{\partial out_{h2}} \frac{\partial out_{h2}}{\partial net_{h2}} \frac{\partial net_{h2}}{\partial w_1} = \frac{\partial net_{o2}}{\partial out_{h1}} \frac{\partial out_{h1}}{\partial net_{h1}} \frac{\partial net_{h1}}{\partial w_1}$$

$$= w_7 * f(net_{h1}) * f(-net_{h1}) * i_1 = 0.5 * f(0.3775) * f(-0.3775) * 0.05 = 0.006$$

Efficient Computation Forward and Backward Propagation

$$\frac{\partial E}{\partial w_5} = \frac{\partial E}{\partial out_{o1}} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial net_{o1}}{\partial w_5} + \frac{\partial E}{\partial out_{o2}} \frac{\partial out_{o2}}{\partial net_{o2}} \frac{\partial net_{o2}}{\partial w_5} = \frac{\partial E}{\partial out_{o1}} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial net_{o1}}{\partial w_5}$$

$$\frac{\partial E}{\partial w_7} = \frac{\partial E}{\partial out_{o1}} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial net_{o1}}{\partial w_7} + \frac{\partial E}{\partial out_{o2}} \frac{\partial out_{o2}}{\partial net_{o2}} \frac{\partial net_{o2}}{\partial w_7} = \frac{\partial E}{\partial out_{o2}} \frac{\partial out_{o2}}{\partial net_{o2}} \frac{\partial net_{o2}}{\partial w_7}$$

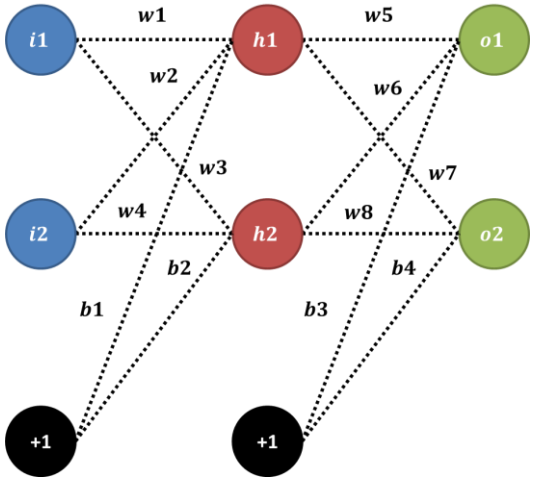
$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial out_{o1}} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial net_{o1}}{\partial w_1} + \frac{\partial E}{\partial out_{o2}} \frac{\partial out_{o2}}{\partial net_{o2}} \frac{\partial net_{o2}}{\partial w_1}$$

$$= \frac{\partial E}{\partial out_{o1}} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial net_{o1}}{\partial out_{h1}} \frac{\partial out_{h1}}{\partial net_{h1}} \frac{\partial net_{h1}}{\partial w_1} + \frac{\partial E}{\partial out_{o2}} \frac{\partial out_{o2}}{\partial net_{o2}} \frac{\partial net_{o2}}{\partial out_{h1}} \frac{\partial out_{h1}}{\partial net_{h1}} \frac{\partial net_{h1}}{\partial w_1}$$

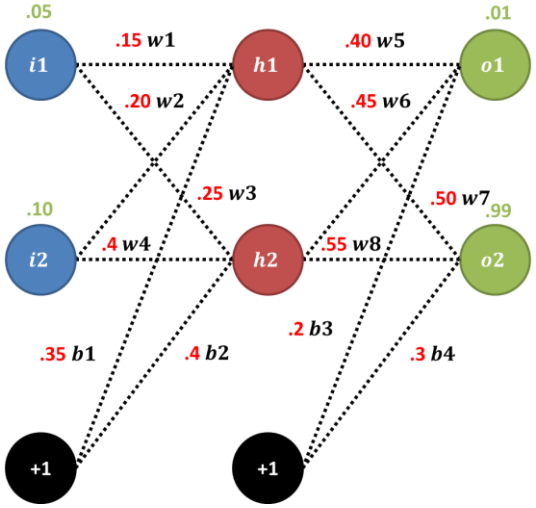
$$\frac{\partial E}{\partial w_4} = \frac{\partial E}{\partial out_{o1}} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial net_{o1}}{\partial w_4} + \frac{\partial E}{\partial out_{o2}} \frac{\partial out_{o2}}{\partial net_{o2}} \frac{\partial net_{o2}}{\partial w_4}$$

$$= \frac{\partial E}{\partial out_{o1}} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial net_{o1}}{\partial out_{h2}} \frac{\partial out_{h2}}{\partial net_{h2}} \frac{\partial net_{h2}}{\partial w_4} + \frac{\partial E}{\partial out_{o2}} \frac{\partial out_{o2}}{\partial net_{o2}} \frac{\partial net_{o2}}{\partial out_{h2}} \frac{\partial out_{h2}}{\partial net_{h2}} \frac{\partial net_{h2}}{\partial w_4}$$

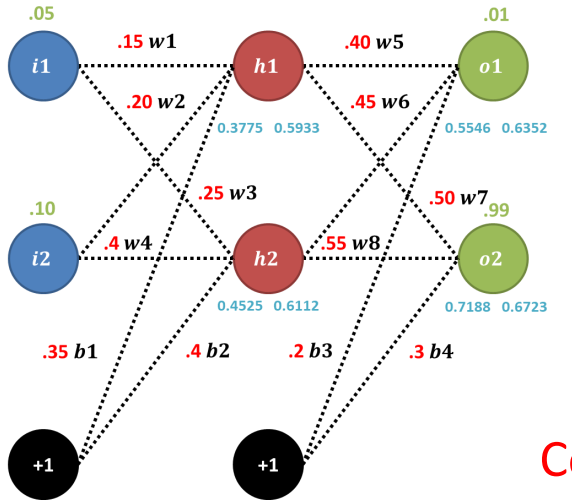
Summary of Neural Network



S1. Design Neural Network

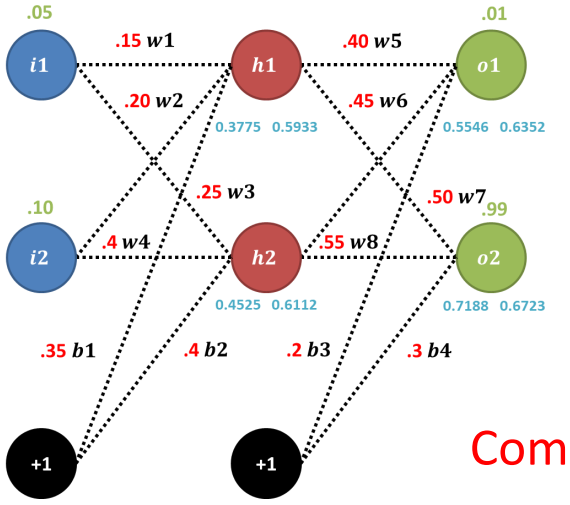


S2. Initialization of NN



S3. Forward Propagation

Compute $J(\theta)$



S4. Backward Propagation

Compute $\frac{\partial}{\partial \theta_j} J(\theta)$

$$\theta_{j+1} \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

S5. Update NN

$$\theta_{j+1} \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

Batch (Vanilla) Gradient Descent

$$\theta_{j+1} \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(x^i, y^j; \theta)$$

Stochastic Gradient Descent

$$\theta_{j+1} \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(x^{\{i:i+b\}}, y^{\{i:i+b\}}; \theta)$$

Mini-batch Gradient Descent

Review of the Tutorial

Forward and Back-propagation- Math Simplified

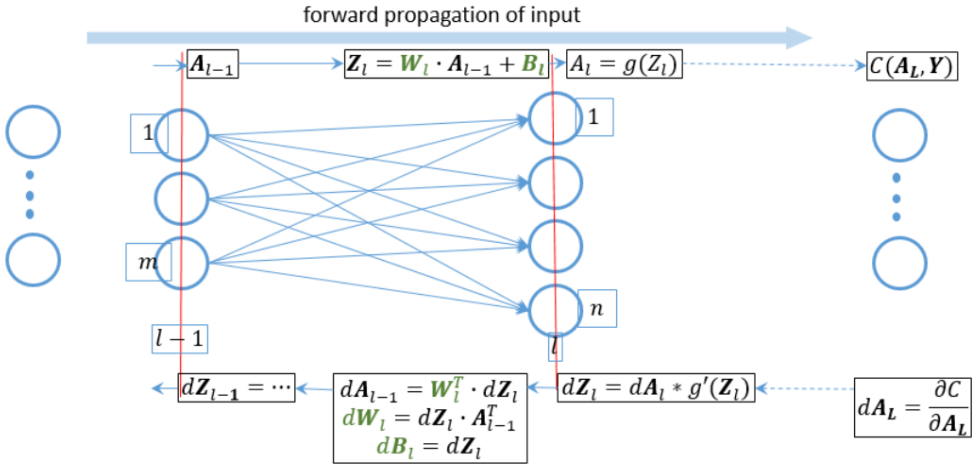
Author: DebPanigrahi (https://github.com/DebPanigrahi/Machine-Learning/blob/master/back_prop.ipynb)
Revise: Chul Min Yeum (cmyeum@uwaterloo.ca) (Mar 27, 2024)

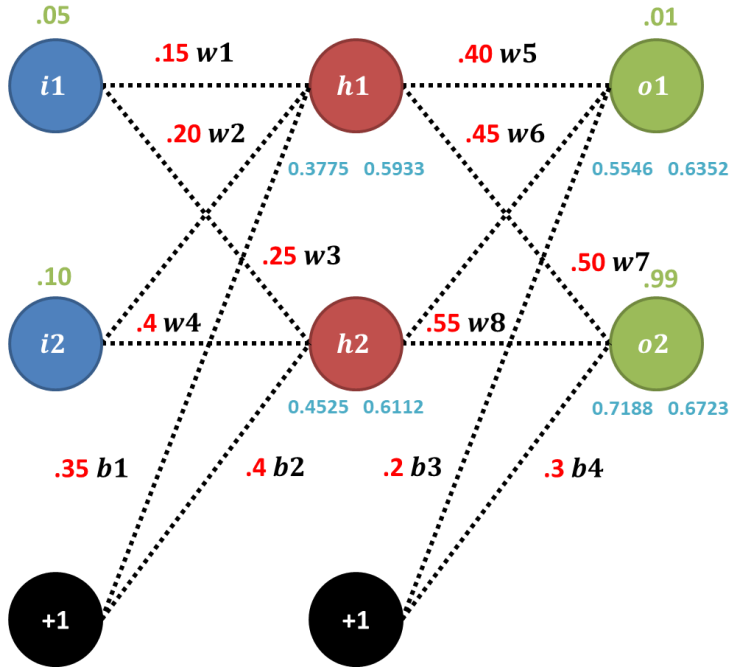
There is already a lot of coverage on this topic on the internet, but some skip the math due to wide audience, others just complicate it using complex mathematical notations and words. In order to drive the NN one may not care what's under the hood, but to do more than that you may need to know what's under the hood (sometimes down to the components) without the jargons. Below I derive some basic math to compute the updates needed for back-propagation. The notations are influenced by fast.ai (Deep Learning) program at USF and Deep Learning specialization course in Coursera. Hope you find it helpful. After all the functions are linear with few more variables, how hard can it get? All we need is familiarity with neural network and high-school math.

Forward propagation is nothing but applying a series of functions on an input vector X with resulting output of each is also a vector. For a neural network generating a logical output between 0 and 1 with 1 hidden layer the function can be represented as $\sigma(f_2(g_1(f_1(X)))$. More generally for a network with $L-1$ hidden layers; $g_L(f_L(...g_l(f_l(..f_1(X)..))...))$. And activation at layer l can be represented as $A_l = g_l(f_l(A_{l-1}))$ assuming its a linear layer followed by a non-linear activation function. Chain rule states that the derivative of g_l with respect to A_{l-1} can be represented as a function of derivative of g_l with respect to f_l: $\frac{\partial g_l}{\partial A_{l-1}} = \frac{\partial f_l}{\partial A_{l-1}} \cdot \frac{\partial g_l}{\partial f_l}$. And we say error $\frac{\partial g_l}{\partial f_l}$ in layer l is now "back-propagated" to layer $l-1$.

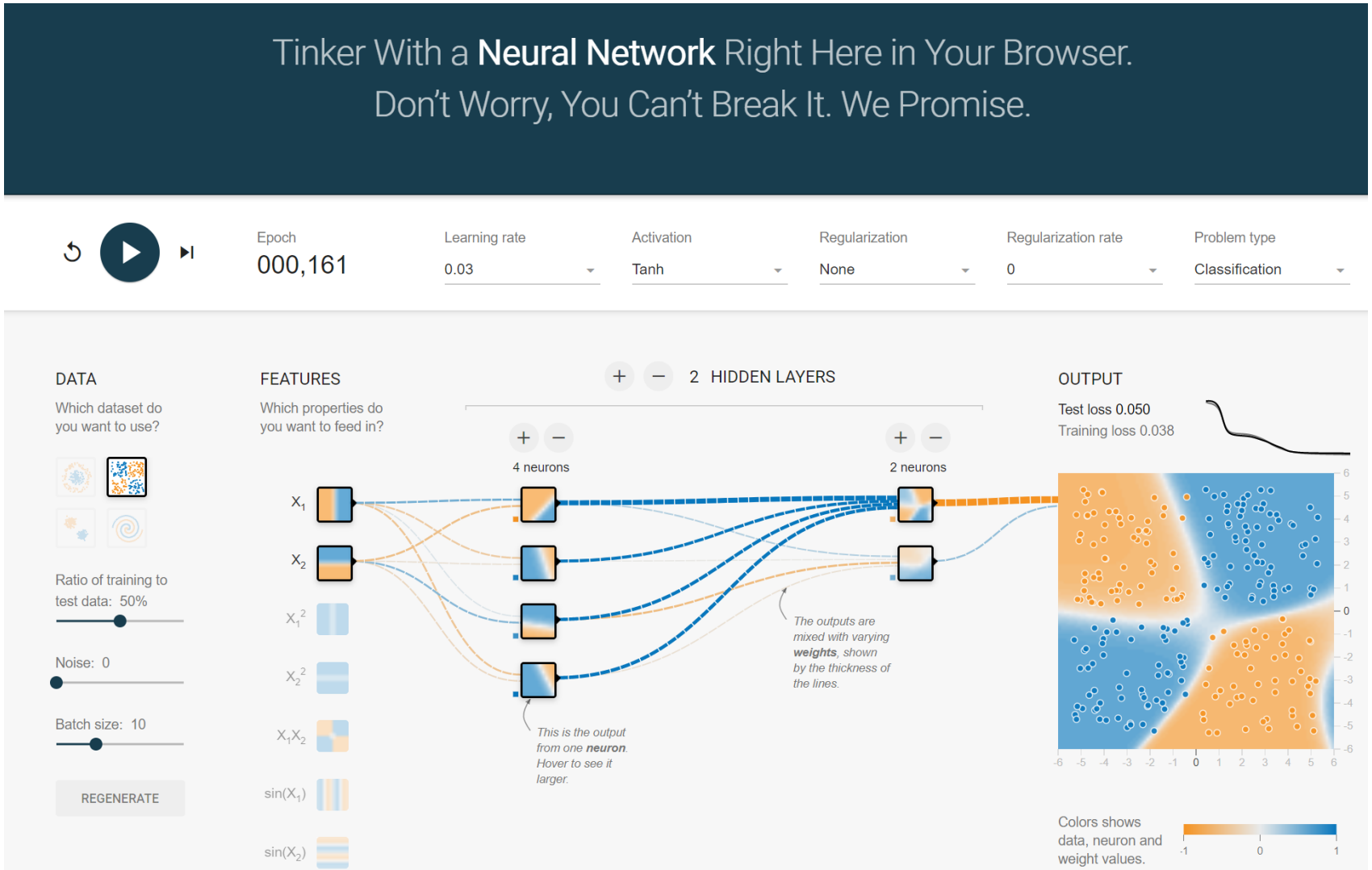
Below is a neural network with 2 adjacent (linear+activation) layers $l-1$ and l with m and n neurons respectively. Then the forward propagation through linear layer is $Z_l = W_l \cdot A_{l-1} + B_l$; where W_l is an $n \times m$ dimensional weight matrix mapping a m dimensional vector A_{l-1} to n dimensional vector Z_l with bias B_l. And through the activation layer it is $g_l(Z_l)$ which is a non-linear function like relu.

$$W = \begin{bmatrix} w_{11} & \dots & w_{1m} \\ \vdots & \ddots & \vdots \\ w_{n1} & \dots & w_{nm} \end{bmatrix}_{n\times m} \quad A_{l-1} = \begin{bmatrix} a_{(l-1)1} \\ \vdots \\ a_{(l-1)m} \end{bmatrix}_{m\times 1} \quad b = \begin{bmatrix} b_{l1} \\ \vdots \\ b_{ln} \end{bmatrix}_{n\times 1} \quad Z = \begin{bmatrix} z_{l1} \\ \vdots \\ z_{ln} \end{bmatrix}_{n\times 1}$$





Neural Network Playground



http://playground.tensorflow.org/

Binary Classification (Circle)