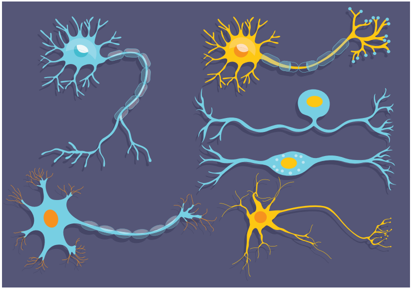
Face Recognition - Day 2

Probe 2018

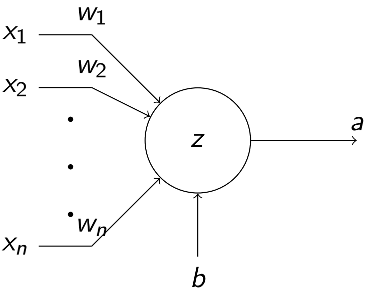
Department of Electronics and Communication Engineering
National Institute of Technology, Tiruchirappalli

Artificial Neural Networks

- An algorithm for learning the complicated functional mapping between the input and the output parameters.
- Inspired by biological neural networks.

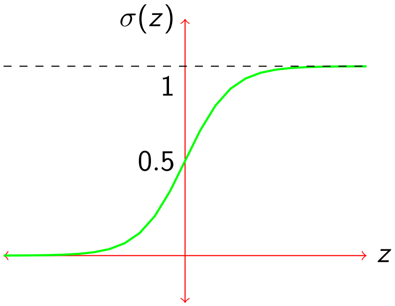


Perceptron



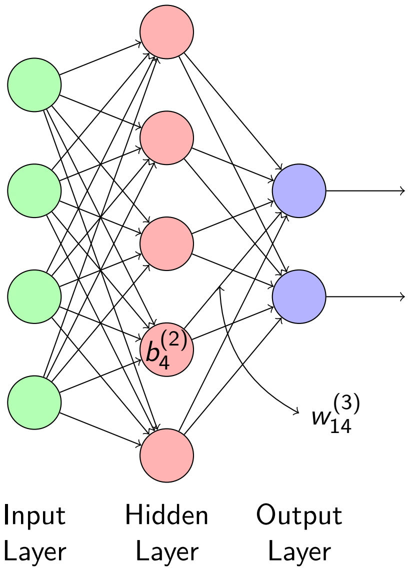
- Mimicks the behaviour of a single neuron.
- $z = \sum_{k=1}^{n} x_k w_k + b$
- $a = \sigma(z)$

Sigmoid Function

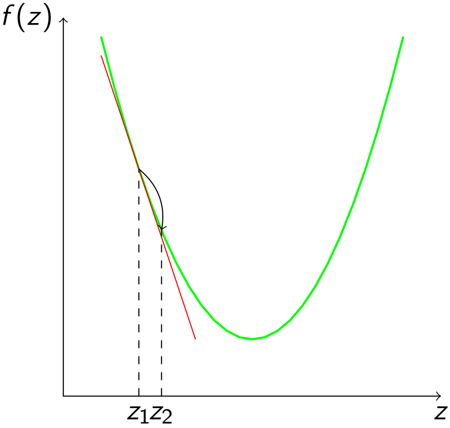


$$\sigma(z) = \frac{1}{1 + e^{(-z)}}$$

Architecture

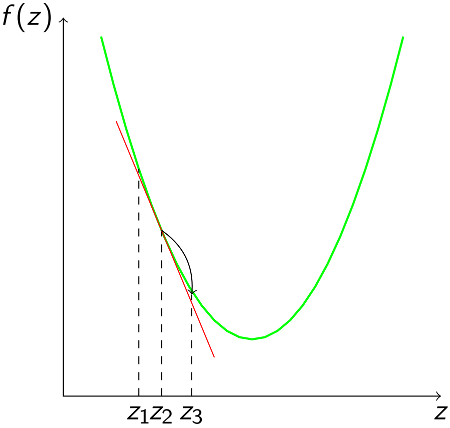


Gradient Descent



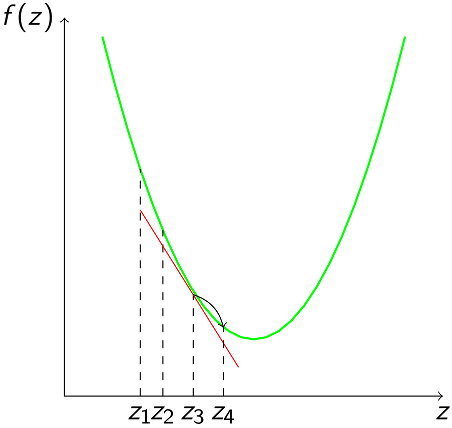
$$z_{t+1}^{i} = z_{t}^{i} - \alpha \frac{\partial f(z)}{\partial z_{t}^{i}}$$

Gradient Descent



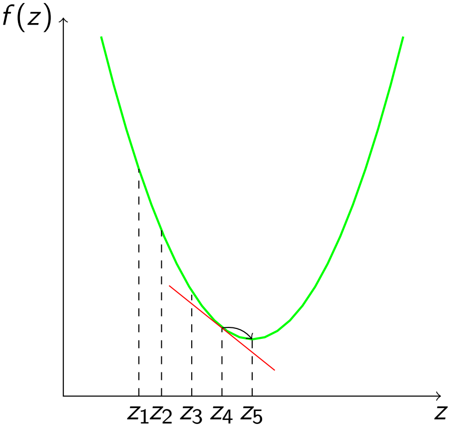
$$z_{t+1}^{i} = z_t^{i} - \alpha \frac{\partial f(z)}{\partial z_t^{i}}$$

Gradient Descent



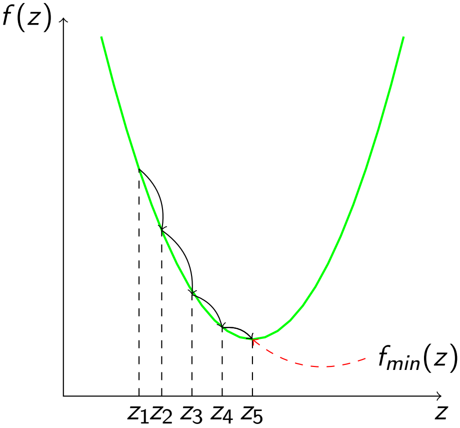
$$z_{t+1}^{i} = z_t^{i} - \alpha \frac{\partial f(z)}{\partial z_t^{i}}$$

Gradient Descent



$$z_{t+1}^{i} = z_{t}^{i} - \alpha \frac{\partial f(z)}{\partial z_{t}^{i}}$$

Gradient Descent



$$z_{t+1}^i = z_t^i - \alpha \frac{\partial f(z)}{\partial z_t^i}$$

In our case the function to be minimized is called the cost function, given by,

$$C = \frac{1}{2}\left\|\mathbf{a}^{(L)} - \mathbf{y}\right\|_2^2$$

Or,

$$C = \frac{1}{2}\sum_{k=1}^{n_L}(a_k^{(L)} - y_k)^2$$

Backpropagation

- Special case of older and more general technique called **automatic differentiation**.
- 1986 - **David Rumelhart**, **Geoffrey Hinton**, **Ronald Williams**.
- An algorithm for efficiently computing the **partial derivatives** required for performing gradient descent.
- Gives detailed insights into how changing weights and biases changes the overall behaviour of the network.

Notations

- $w_{jk}^{(l)}$ - weight corresponding to the connection going from k^{th} neuron in the $(l-1)^{th}$ layer to the j^{th} neuron in the l^{th} layer.
- $b_j^{(l)}$ - bias corresponding to the j^{th} neuron in the l^{th} layer.
- $a_j^{(l)}$ - activation of the j^{th} neuron in the l^{th} layer.

Notations

Activation vector,

$$\mathbf{a}^{(l)} = \begin{bmatrix} a_1^{(l)} \\ a_2^{(l)} \\ \vdots \\ a_n^{(l)} \end{bmatrix} \tag{1}$$

Weights matrix,

$$\mathbf{W}^{(l)} = \begin{bmatrix} w_{11}^{(l)} & \dots & w_{1m}^{(l)} \\ \vdots & \ddots & \vdots \\ w_{n1}^{(l)} & \dots & w_{nm}^{(l)} \end{bmatrix} \tag{2}$$

We introduce another paramter,

$$\mathbf{z}^{(l)} = \mathbf{W}^{(l)}\mathbf{a}^{(l-1)} + \mathbf{b}^{(l)}$$

where,

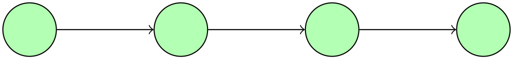
$$z_j^{(l)} = \sum_{k=1}^{n} w_{jk}^{(l)} a_k^{(l-1)} + b_j^{(l)}$$

Hence,

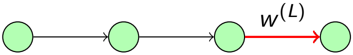
$$\mathbf{a}^{(l)} = \sigma(\mathbf{z}^l)$$

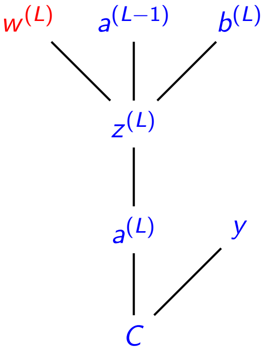
Updating Model Parameters

Let us start with a simple model,



$$C(w_1, b_1, w_2, b_2, w_3, b_3)$$



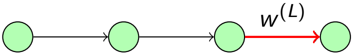


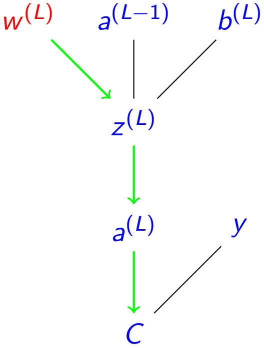
$$C(\dots) = \frac{1}{2}(a^{(L)} - y)^2$$

we need,

$$\frac{\partial C}{\partial w^{(L)}}, \frac{\partial C}{\partial b^{(L)}}$$

The derivative measures how sensitive the cost is to the model parameters.

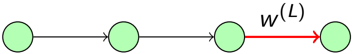




C depends on a which in turn depends on z which inturn depends on w.

By applying the chain rule we get,

$$\frac{\partial C_0}{\partial w^{(L)}} = \frac{\partial C_0}{\partial a^{(L)}} \frac{\partial a^{(L)}}{\partial z^{(L)}} \frac{\partial z^{(L)}}{\partial w^{(L)}}$$



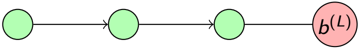
$$C_0 = \frac{1}{2}(a^{(L)} - y)^2$$
$$z^{(L)} = w^L a^{(L-1)} + b^{(L)}$$
$$a^{(L)} = \sigma(z^{(L)})$$

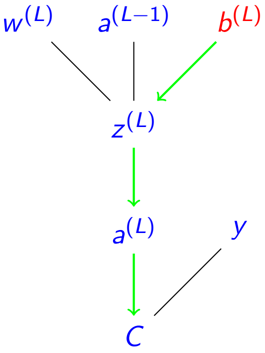
$$\frac{\partial C_0}{\partial a^{(L)}} = (a^{(L)} - y)$$

$$\frac{\partial a^{(L)}}{\partial z^{(L)}} = \sigma'(z^{(L)})$$

$$\frac{\partial z^{(L)}}{\partial w^{(L)}} = a^{(L-1)}$$

$$\begin{aligned}\frac{\partial C_0}{\partial w^{(L)}} &= \frac{\partial C_0}{\partial a^{(L)}} \frac{\partial a^{(L)}}{\partial z^{(L)}} \frac{\partial z^{(L)}}{\partial w^{(L)}} \\ &= (a^{(L)} - y)\sigma'(z^{(L)})a^{(L-1)}\end{aligned}$$

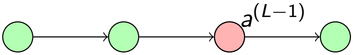


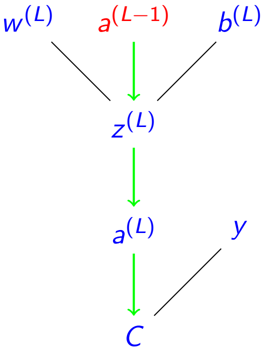


$$C_0 = \tfrac{1}{2}(a^{(L)} - y)^2$$
$$z^{(L)} = w^L a^{(L-1)} + b^{(L)}$$
$$a^{(L)} = \sigma(z^{(L)})$$

Similarly,
$$\frac{\partial z^{(L)}}{\partial b^{(L)}} = 1$$

$$\begin{aligned}\frac{\partial C_0}{\partial b^{(L)}} &= \frac{\partial C_0}{\partial a^{(L)}}\frac{\partial a^{(L)}}{\partial z^{(L)}}\frac{\partial z^{(L)}}{\partial b^{(L)}} \\ &= (a^{(L)} - y)\sigma'(z^{(L)})\end{aligned}$$



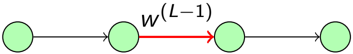


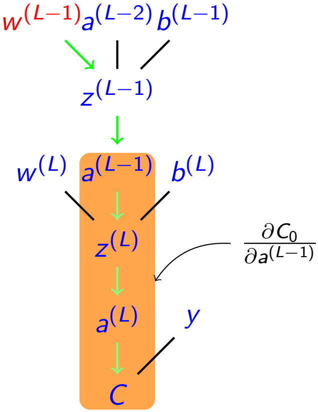
$$C_0 = \frac{1}{2}(a^{(L)} - y)^2$$
$$z^{(L)} = w^L a^{(L-1)} + b^{(L)}$$
$$a^{(L)} = \sigma(z^{(L)})$$

Similarly,

$$\frac{\partial z^{(L)}}{\partial a^{(L-1)}} = w^{(L)}$$

$$\frac{\partial C_0}{\partial a^{(L-1)}} = \frac{\partial C_0}{\partial a^{(L)}} \frac{\partial a^{(L)}}{\partial z^{(L)}} \frac{\partial z^{(L)}}{\partial a^{(L-1)}}$$
$$= (a^{(L)} - y)\sigma'(z^{(L)}) w^{(L)}$$





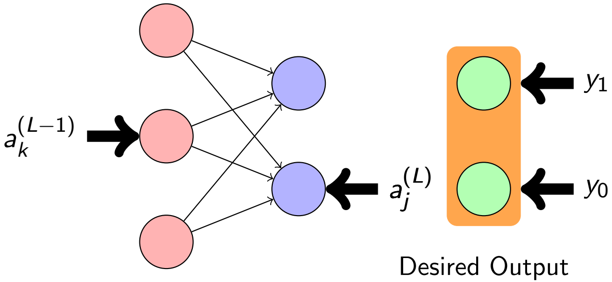
By applying the chain rule we get,

$$\frac{\partial C_0}{\partial w^{(L-1)}} = \frac{\partial C_0}{\partial a^{(L-1)}} \frac{\partial a^{(L-1)}}{\partial z^{(L-1)}} \frac{\partial z^{(L-1)}}{\partial w^{(L-1)}}$$

$\frac{\partial C_0}{\partial a^{(L-1)}}$ is computed at the previous stage.

But isn't this model over simplified?

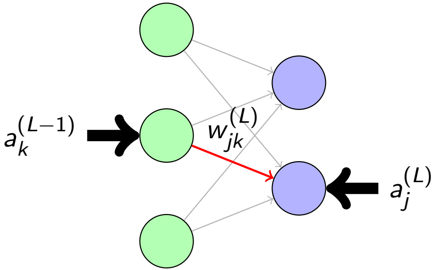
Not that much Actually!

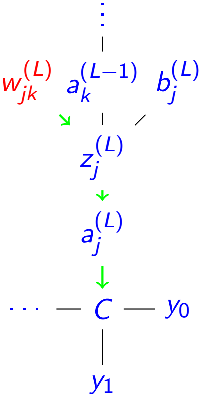


$$C_0 = \frac{1}{2}\sum_{j=0}^{n_L-1}(a_j^{(L)} - y_j)^2$$

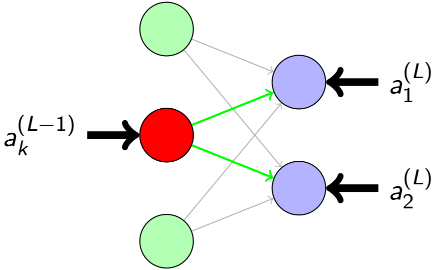
$$z_j^{(l)} = \sum_k w_{jk}^{(l)} a_k^{(l-1)} + b_j^{(l)}$$

$$a_j^{(l)} = \sigma(z_j^{(l)})$$





$$\frac{\partial C_0}{\partial w_{jk}^{(L)}} = \frac{\partial C_0}{\partial a_j^{(L)}} \frac{\partial a_j^{(L)}}{\partial z_j^{(L)}} \frac{\partial z_j^{(L)}}{\partial w_{jk}^{(L)}}$$



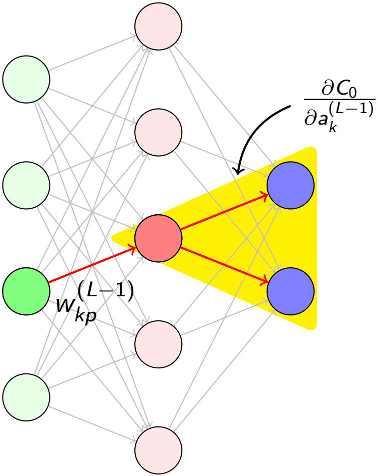
Contribution due to the top path,

$$\frac{\partial C_0}{\partial a_1^{(L)}} \frac{\partial a_1^{(L)}}{\partial z_1^{(L)}} \frac{\partial z_1^{(L)}}{\partial a_k^{(L-1)}}$$

Contribution from the bottom path,

$$\frac{\partial C_0}{\partial a_2^{(L)}} \frac{\partial a_2^{(L)}}{\partial z_2^{(L)}} \frac{\partial z_2^{(L)}}{\partial a_k^{(L-1)}}$$

$$\frac{\partial C_0}{\partial a_k^{(L-1)}} = \sum_{j=0}^{n_L-1} \frac{\partial C_0}{\partial a_j^{(L)}} \frac{\partial a_j^{(L)}}{\partial z_j^{(L)}} \frac{\partial z_j^{(L)}}{\partial a_k^{(L-1)}}$$



$\frac{\partial C_0}{\partial a_k^{(L-1)}}$ is computed in the previous stage.

Therefore,

$$\frac{\partial C_0}{\partial w_{kp}^{(L-1)}} = \frac{\partial C_0}{\partial a_k^{(L-1)}} \frac{\partial a_k^{(L-1)}}{\partial z_k^{(L-1)}} \frac{\partial z_k^{(L-1)}}{\partial w_{kp}^{(L-1)}}$$

In general,

$$\frac{\partial C_0}{\partial a_j^{(l)}} = \begin{cases} \sum_{k=0}^{n_l-1} \frac{\partial C_0}{\partial a_k^{(l+1)}} \frac{\partial a_k^{(l+1)}}{\partial z_k^{(l+1)}} \frac{\partial z_k^{(l+1)}}{\partial a_j^{(l)}} & l \neq L \\ a_j^{(L)} - y_j & l = L \end{cases}$$

and,

$$\frac{\partial C_0}{\partial w_{jk}^{(l)}} = \frac{\partial C_0}{\partial a_j^{(l)}} \frac{\partial a_j^{(l)}}{\partial z_j^{(l)}} \frac{\partial z_j^{(l)}}{\partial w_{jk}^{(l)}}$$

$$\frac{\partial C_0}{\partial a_j^{(l)}} = \begin{cases} \sum_{k=0}^{n_l-1} \sigma'(z_k^{(l+1)}) w_{kj}^{(l+1)} \frac{\partial C_0}{\partial a_k^{(l+1)}} & l \neq L \\ a_j^{(L)} - y_j & l = L \end{cases}$$

$$\frac{\partial C_0}{\partial w_{jk}^{(l)}} = \sigma'(z_j^{(l)}) a_k^{(l-1)} \frac{\partial C_0}{\partial a_j^{(l)}}$$

Vectorized Implementation

1.

$$\nabla_{\mathbf{a}^{(l)}} C_0 = \begin{cases} \sum_{k=0}^{n_l-1} \sigma'(\mathbf{z}^{(l+1)}) \odot (\mathbf{W}^{(l+1),T})\nabla_{\mathbf{a}^{(l+1)}} C_0 & l \neq L \\ \mathbf{a}^{(L)} - \mathbf{y} & l = L \end{cases}$$

2.

$$\nabla_{\mathbf{W}^{(l)}} C_0 = \sigma'(\mathbf{z}^{(l)}) \odot (\nabla_{\mathbf{a}^{(l)}} C_0)\mathbf{a}^{(l-1),T}$$

$$\mathbf{W}_{t+1}^{(l)} = \mathbf{W}_t^{(l)} + \alpha \nabla_{\mathbf{W}_t^{(l)}} C_0$$

$\odot$ - denotes elementwise product.