Lecture 6

Artificial Neural Network

Dr. Le Huu Ton

Hanoi, 2017

Outline

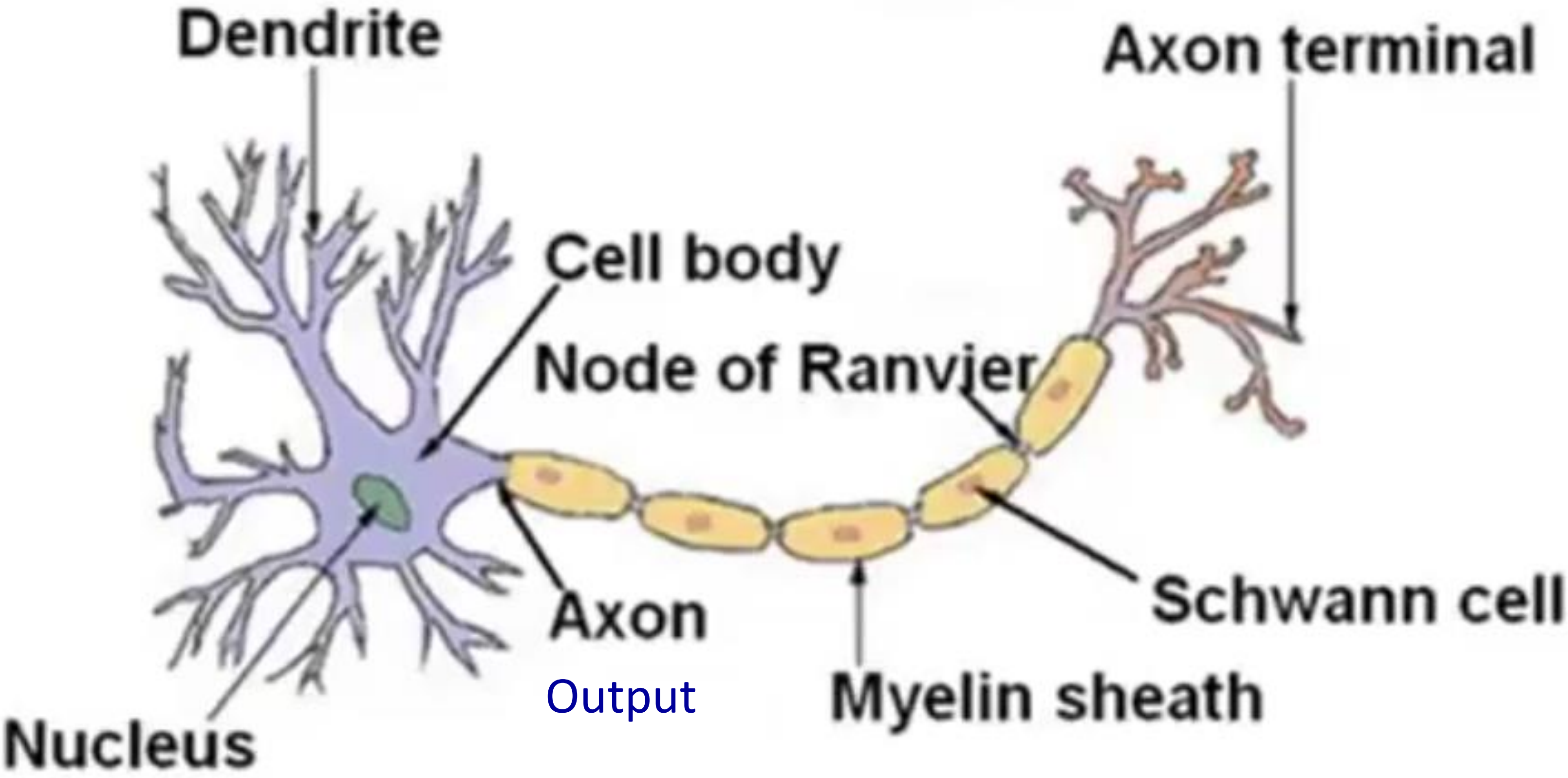
- Artificial Neural Network
- Back Propagation Gradient Descent

Outline

- Artificial Neural Network
- Back Propagation Gradient Descent

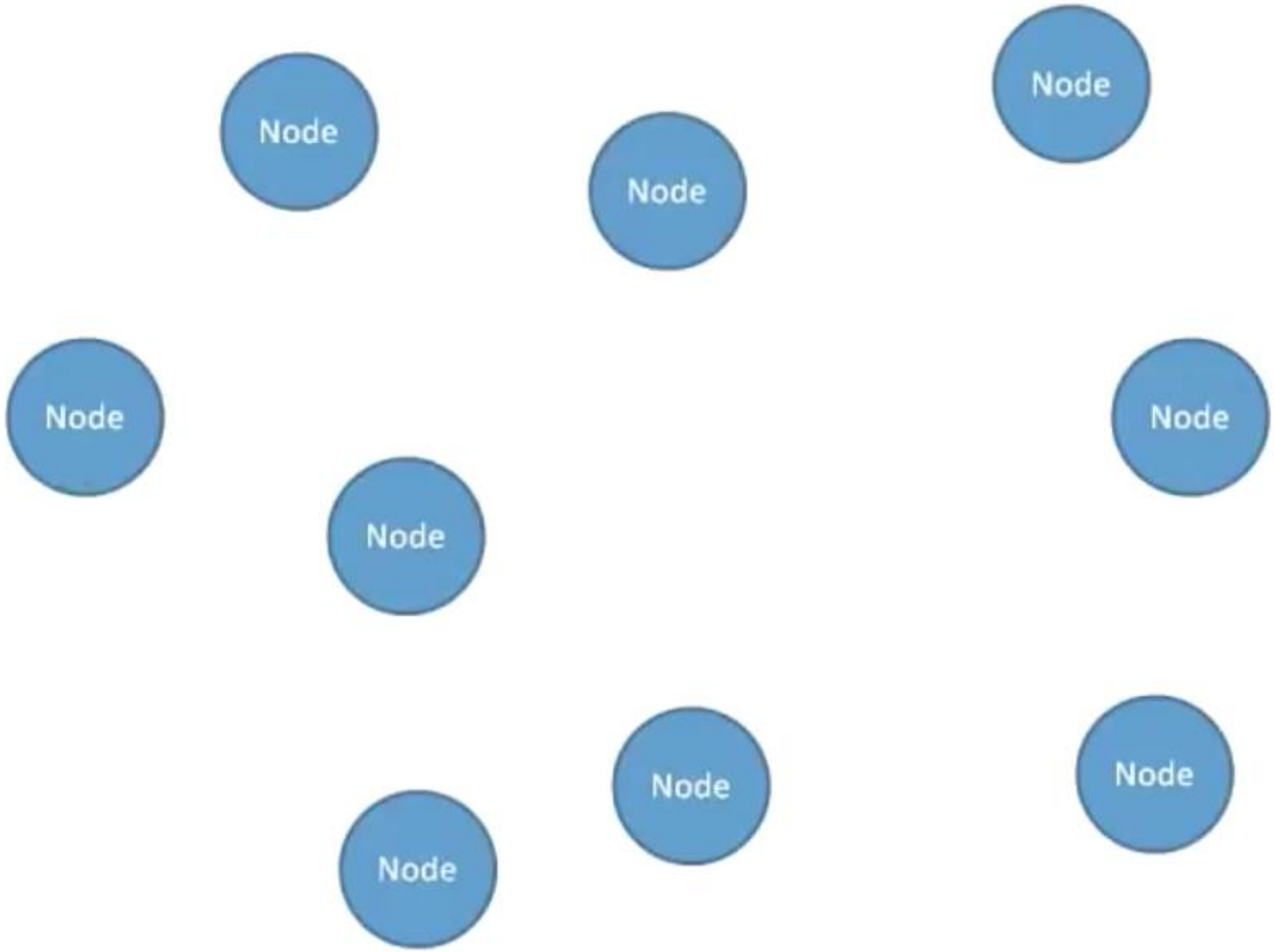
Artificial Neural Network

Input

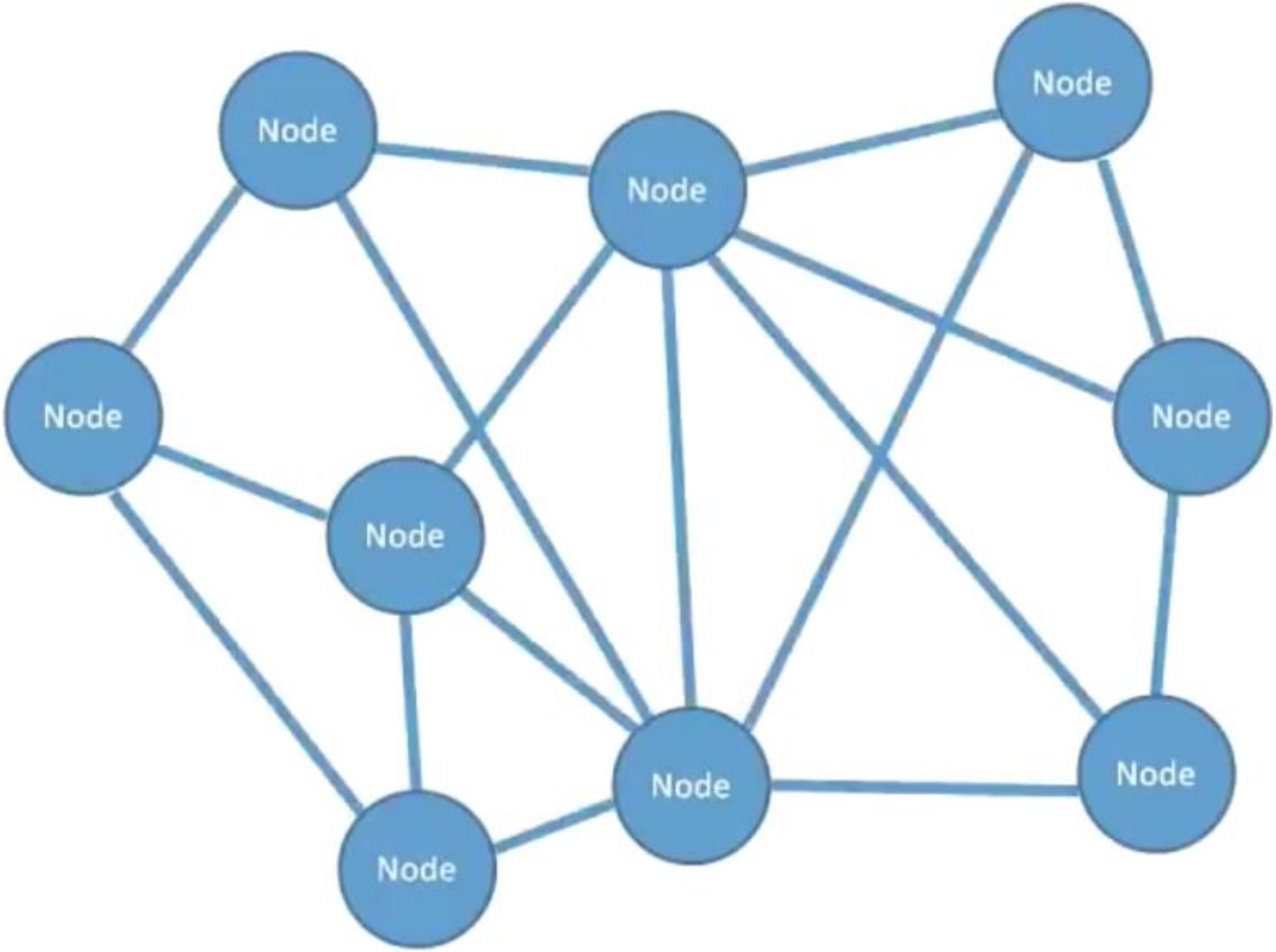


Output

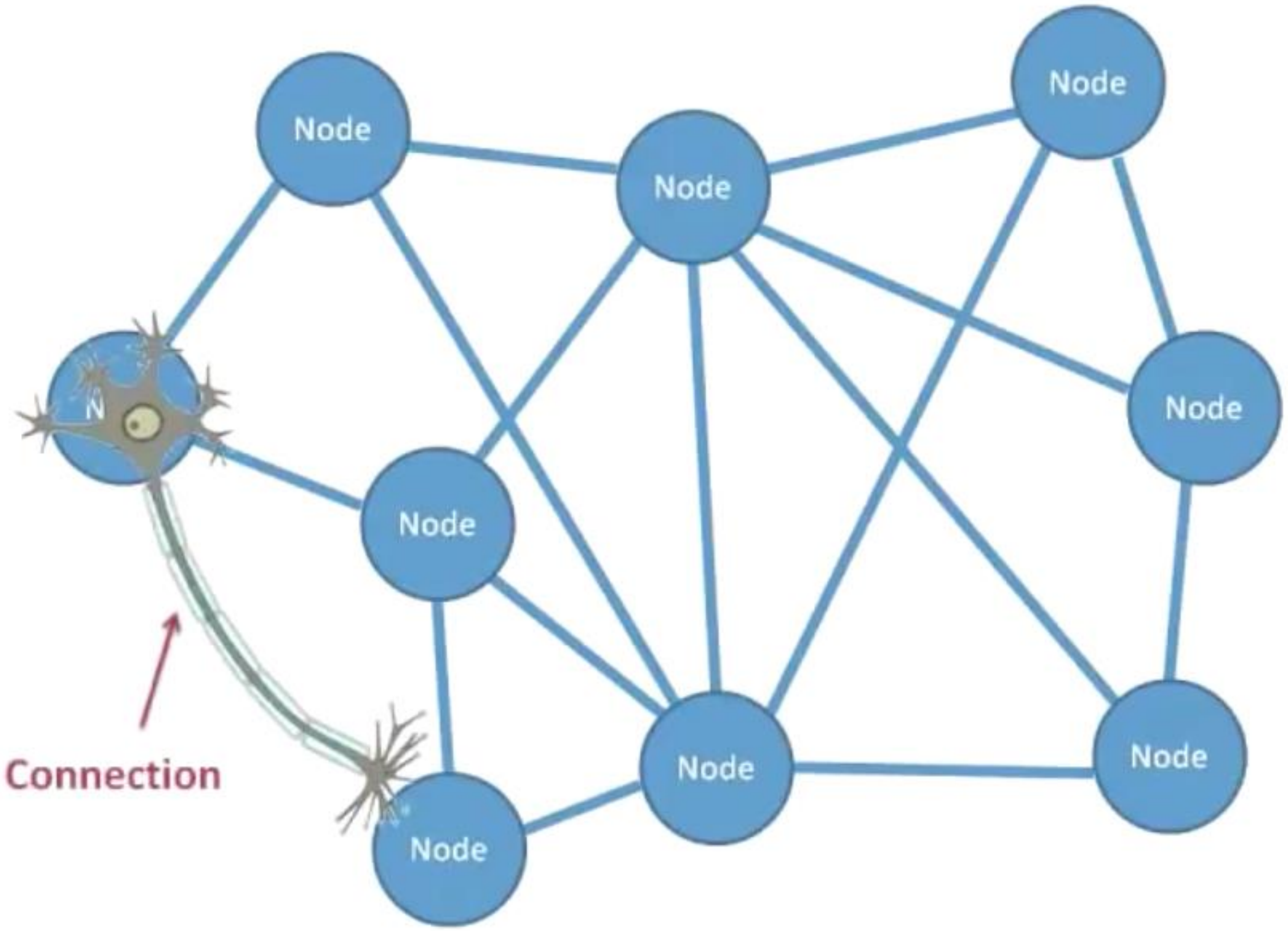
Artificial Neural Network



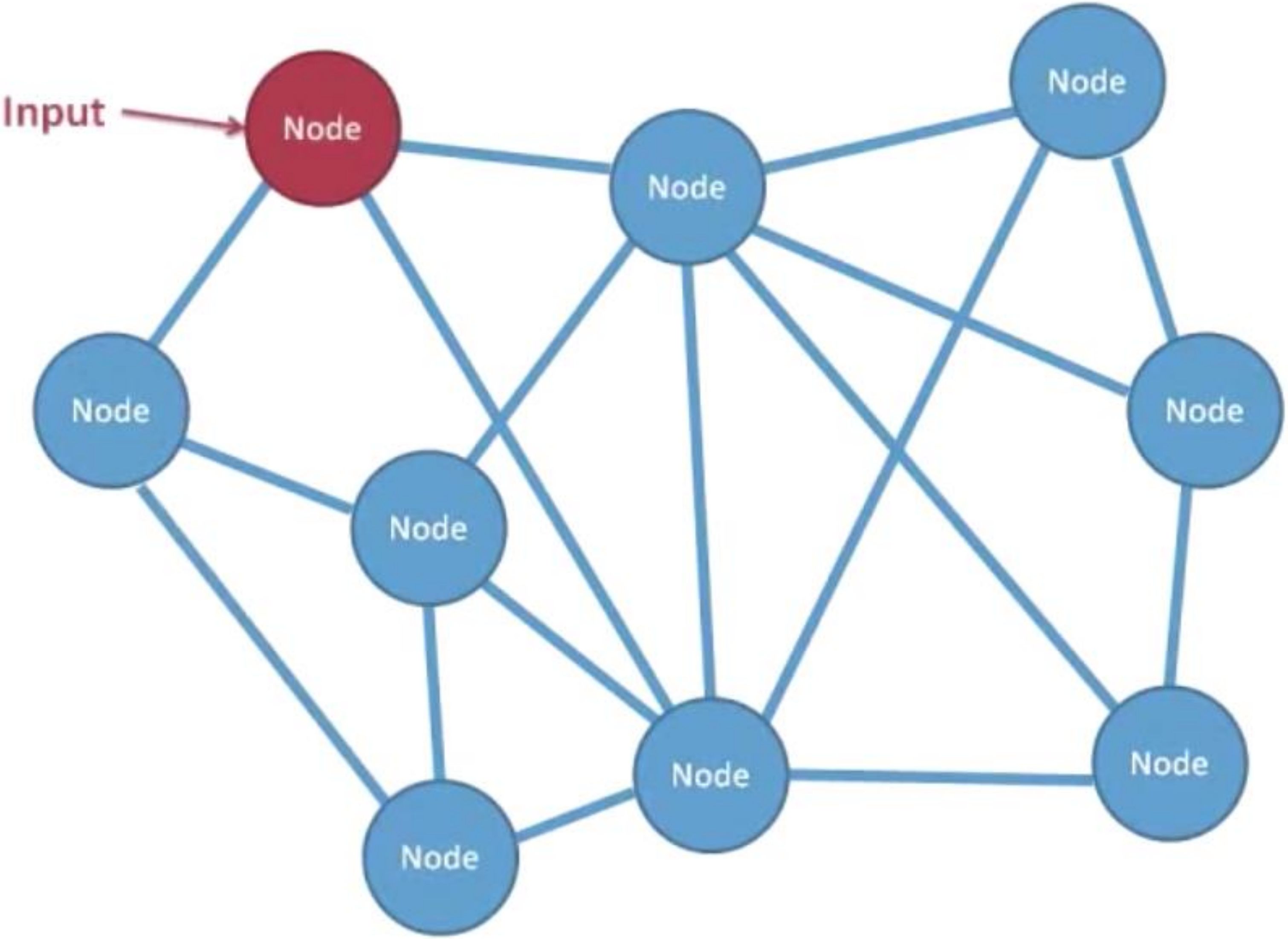
Artificial Neural Network



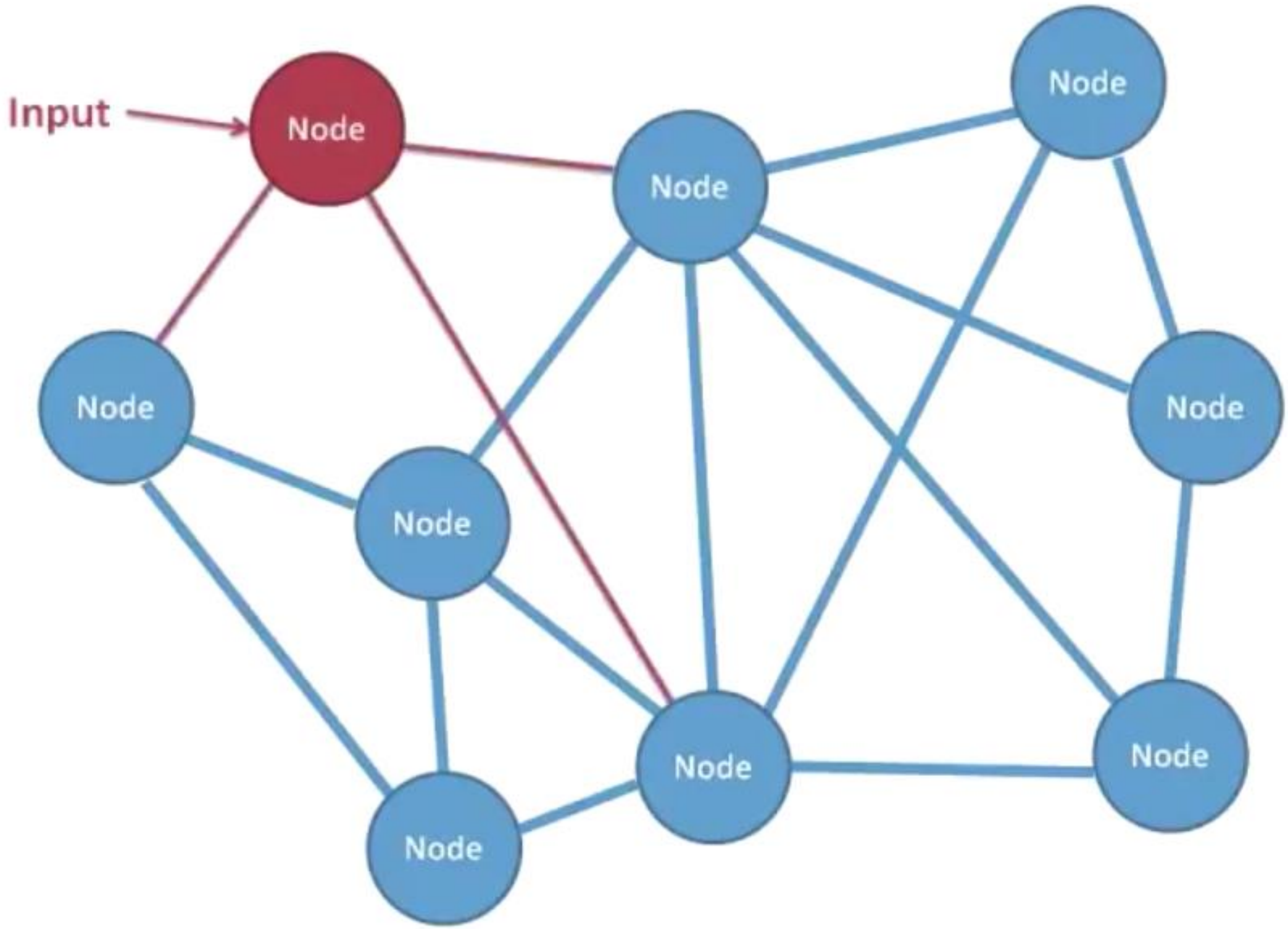
Artificial Neural Network



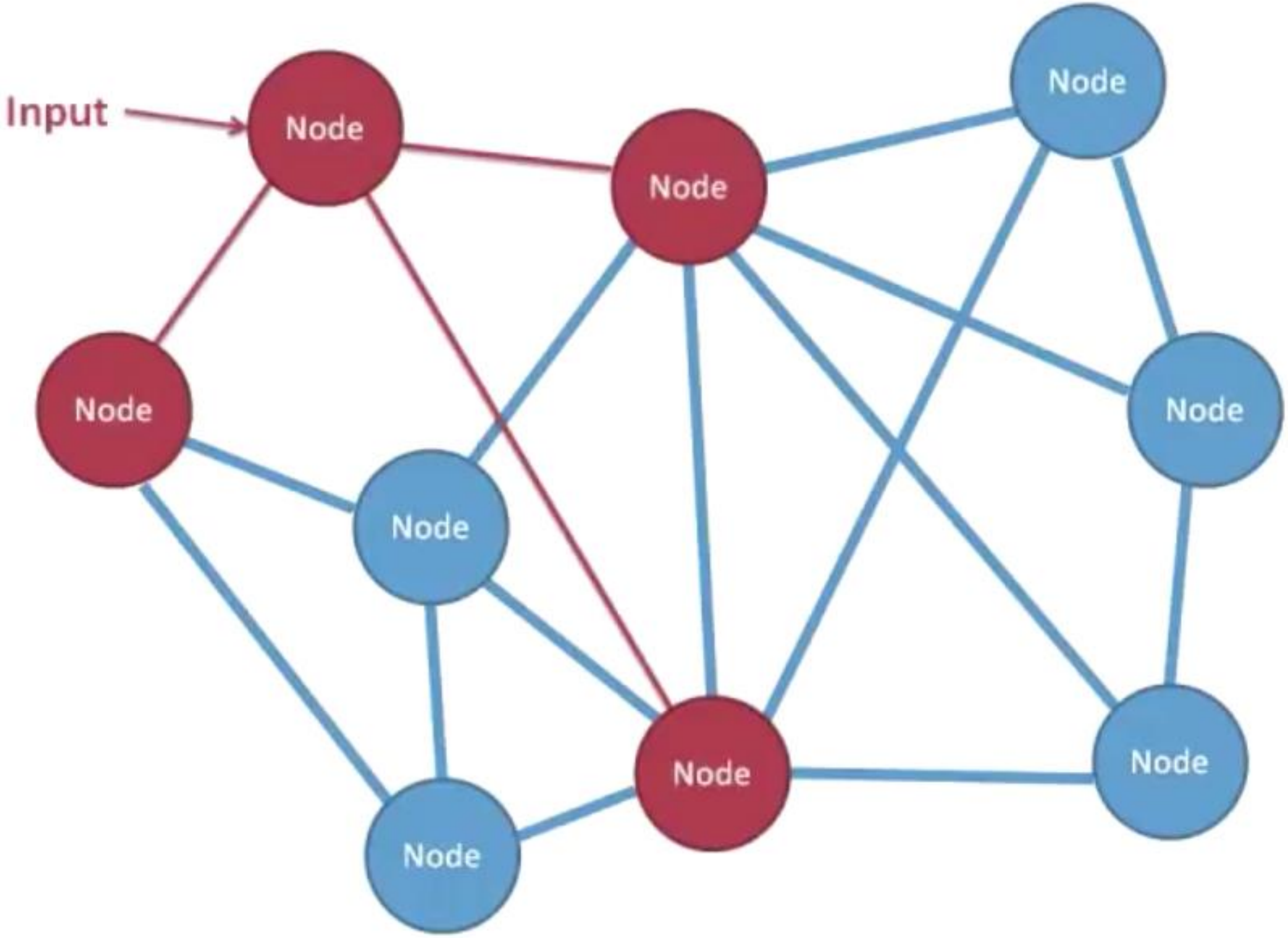
Artificial Neural Network



Artificial Neural Network

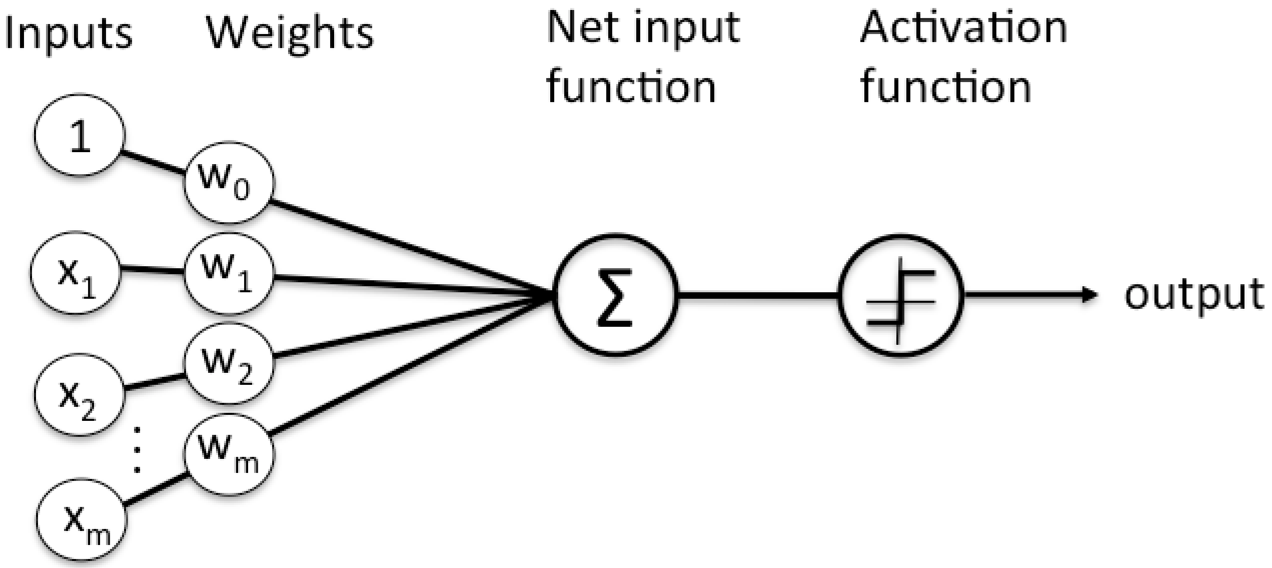


Artificial Neural Network



Artificial Neural Network

Perceptron: mimic the operation of neuron

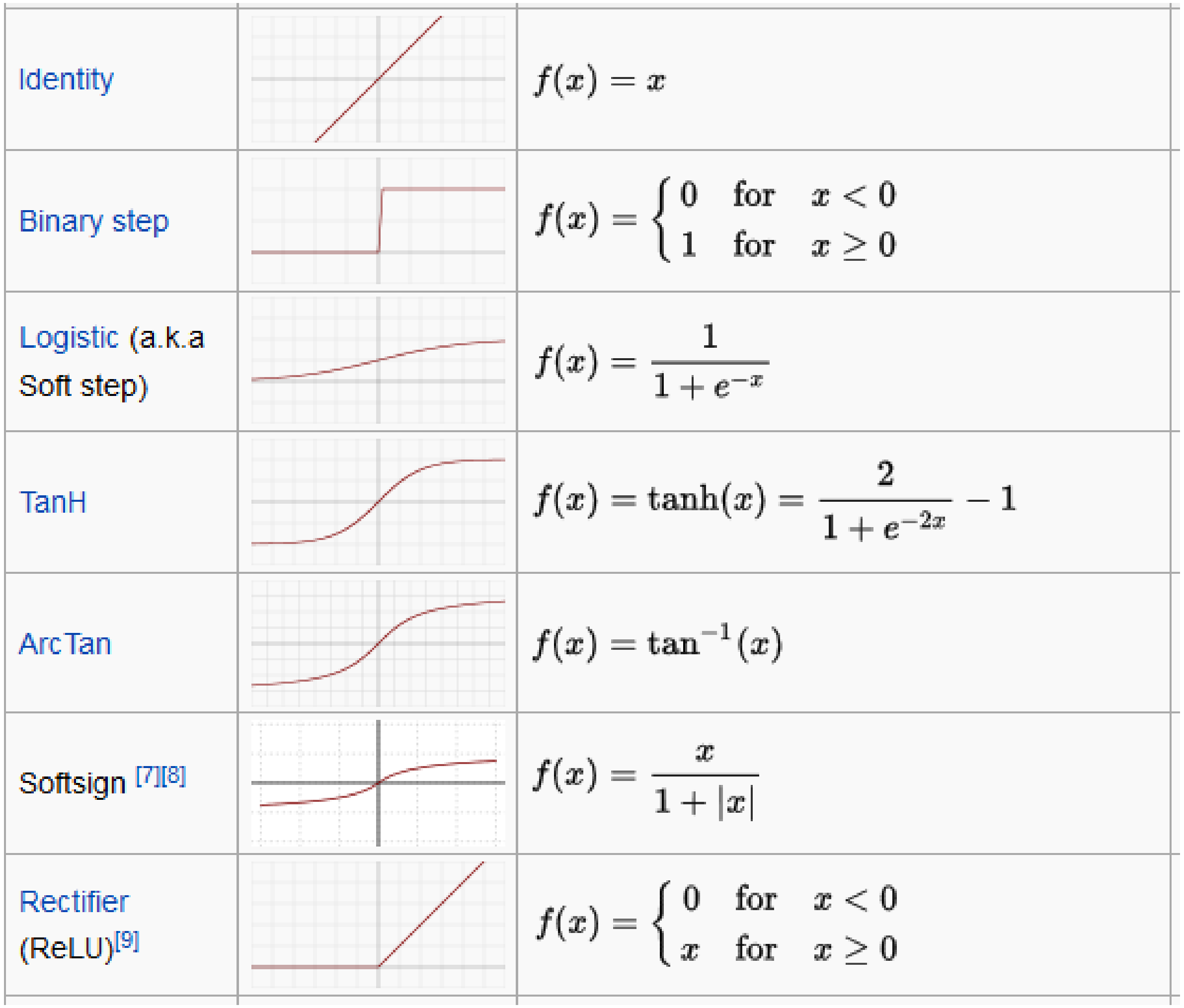


$$net = 1 * w_0 + x_1 * w_1 + x_2 * w_2 + x_3 * w_3 = w^T x$$

$$y = f(z) : activation\ \ function$$

Artificial Neural Network

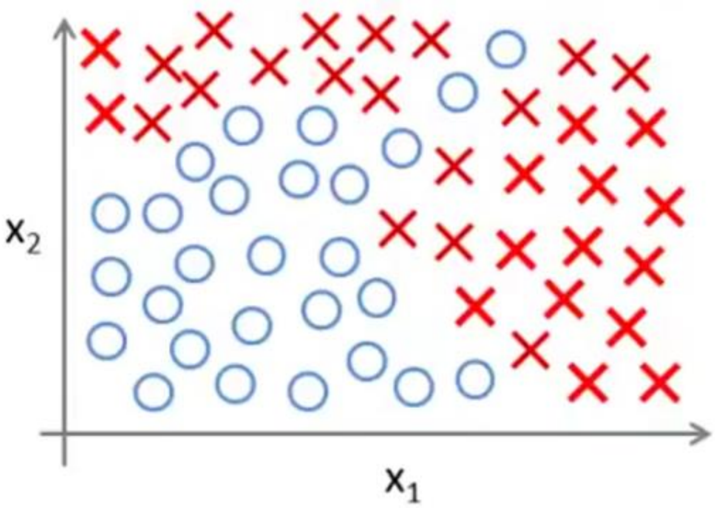
Popular activation function



Identity		$f(x) = x$
Binary step		$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$
ArcTan		$f(x) = \tan^{-1}(x)$
Softsign [7][8]		$f(x) = \frac{x}{1 + \lvert x \rvert}$
Rectifier (ReLU)[9]		$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$

Artificial Neural Network

Neural Network and logistic regression



$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^3 x_2 + \theta_6 x_1 x_2^2 + \dots)$$

Many features, and their relation is complex

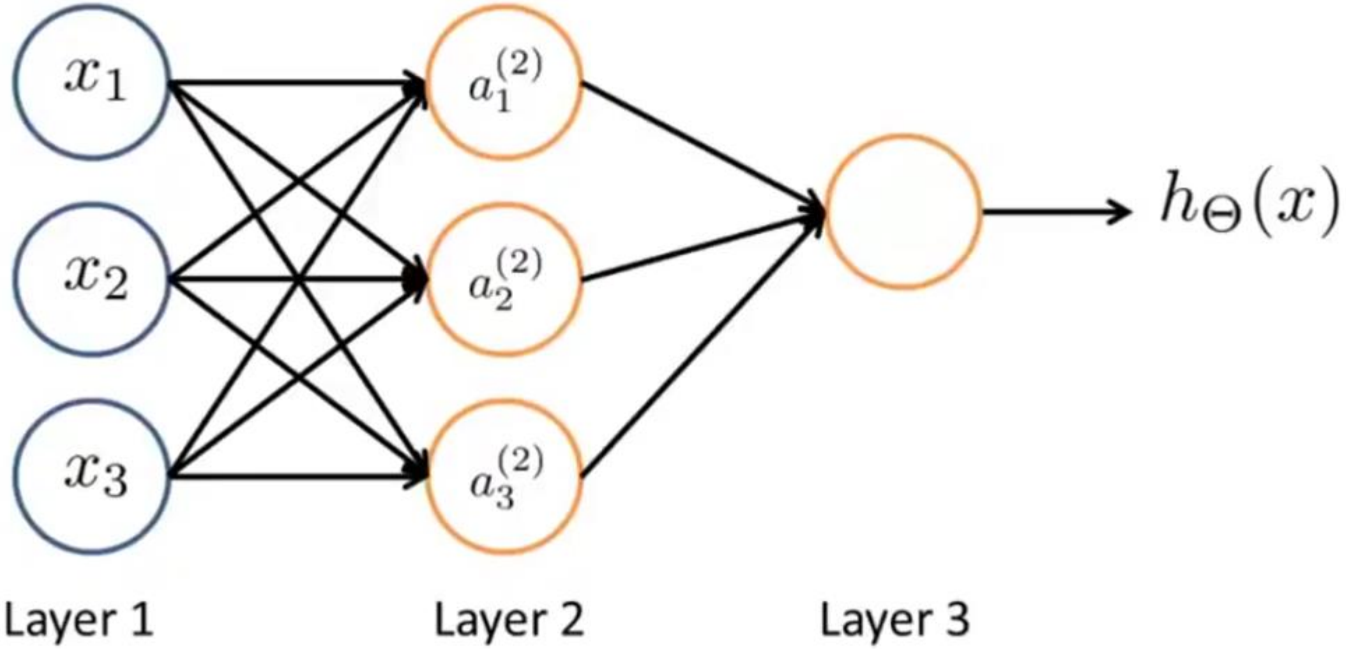
100 original features => 5000 second order features

=> 170.000 third order features

=> Not really a good way to introduce too many features to build non-linear classification

Artificial Neural Network

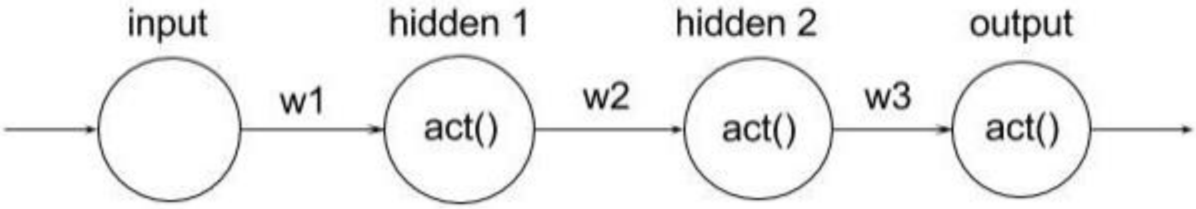
Artificial Neural Network: neurons connect to each others



Input Layer Hidden Layer Output layer

Artificial Neural Network

Neural Network: feed forward structure



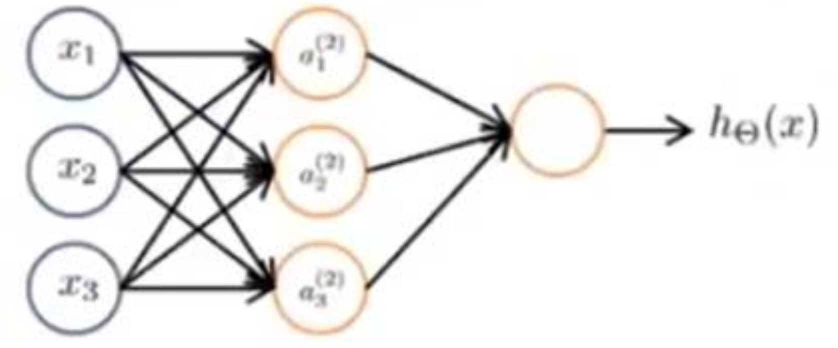
$$a^1 = g(w^1 x)$$

$$a^2 = g(w^2 a^1)$$

$$output = a^3 = g(w^3 a^2)$$

Artificial Neural Network

Neural Network: feed forward structure



$a_i^{(j)}$ = "activation" of unit i in layer j

w^j = matrix of weights controlling function mapping from layer j to layer $j+1$

$$a_1^2 = g(w_{10}^1 x_0 + w_{11}^1 x_1 + w_{12}^1 x_2 + w_{13}^1 x_3)$$

$$a_2^2 = g(w_{20}^1 x_0 + w_{21}^1 x_1 + w_{22}^1 x_2 + w_{23}^1 x_3)$$

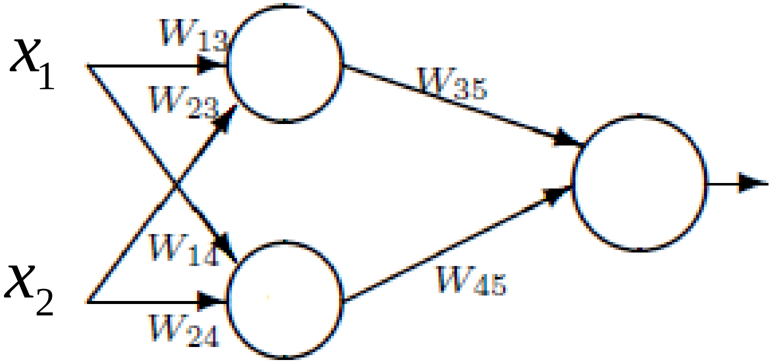
$$a_3^2 = g(w_{30}^1 x_0 + w_{31}^1 x_1 + w_{32}^1 x_2 + w_{33}^1 x_3)$$

$$h(x) = a_1^3 = g(w_{10}^2 a_0^2 + w_{11}^2 a_1^2 + w_{12}^2 a_2^2 + w_{13}^2 a_3^2)$$

Artificial Neural Network

Exercise: Calculate the output of the following network with x_1=1, x_2=0;

$w_{13} = 2$ $w_{23} = -3$	$w_{35} = 2$ $w_{45} = -1$
$w_{14} = 1$ $w_{24} = 4$	



Given the following activation function

$$f(v) = \begin{cases} 1 & \text{if } v \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Outline

- Artificial Neural Network
- Back Propagation Gradient Descent

Artificial Neural Network

How to train the network: Gradient descents algorithm

Cost function

$$E(w) = \frac{1}{2m} \sum \; (y - h(x))^2$$

Where y is the expected value of the output

h(x) is the predicted value of the output

$$w := w - \alpha \frac{\partial E}{\partial_w}$$

Artificial Neural Network

Back propagation: Update the weighting w layer by layer, starting from the last layer of the network.

Apply the chain rule derivation

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

$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial a} \cdot \frac{\partial a}{\partial net} \cdot \frac{\partial net}{\partial w}$$

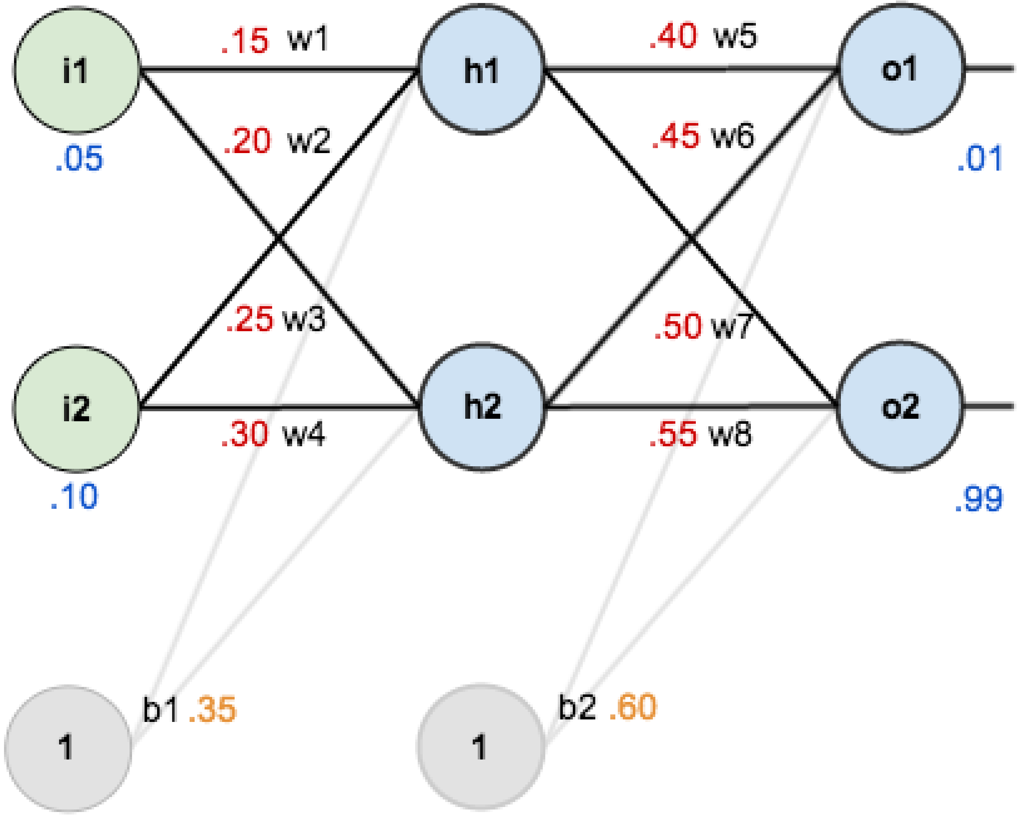
Artificial Neural Network

Example: Given the following network with follow with initial value and training data. Apply the back propagation to update the weighting w

The output of

the network

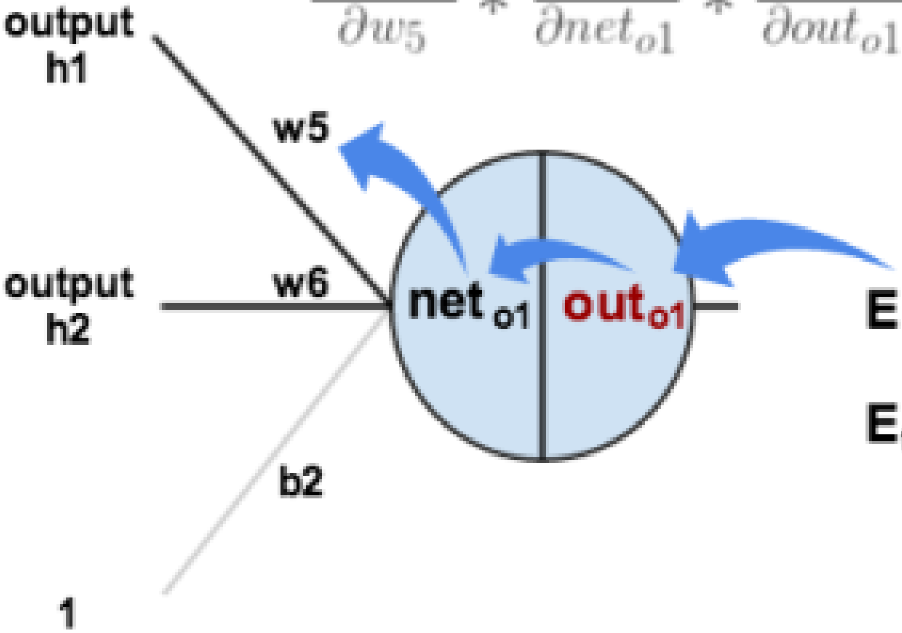
The total error



Artificial Neural Network

Using back propagation gradient descent to update w_5

$$\frac{\partial net_{o1}}{\partial w_5} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial E_{total}}{\partial out_{o1}} = \frac{\partial E_{total}}{\partial w_5}$$



$$E_{o1} = \tfrac{1}{2}(target_{o1} - out_{o1})^2$$

$$E_{total} = E_{o1} + E_{o2}$$

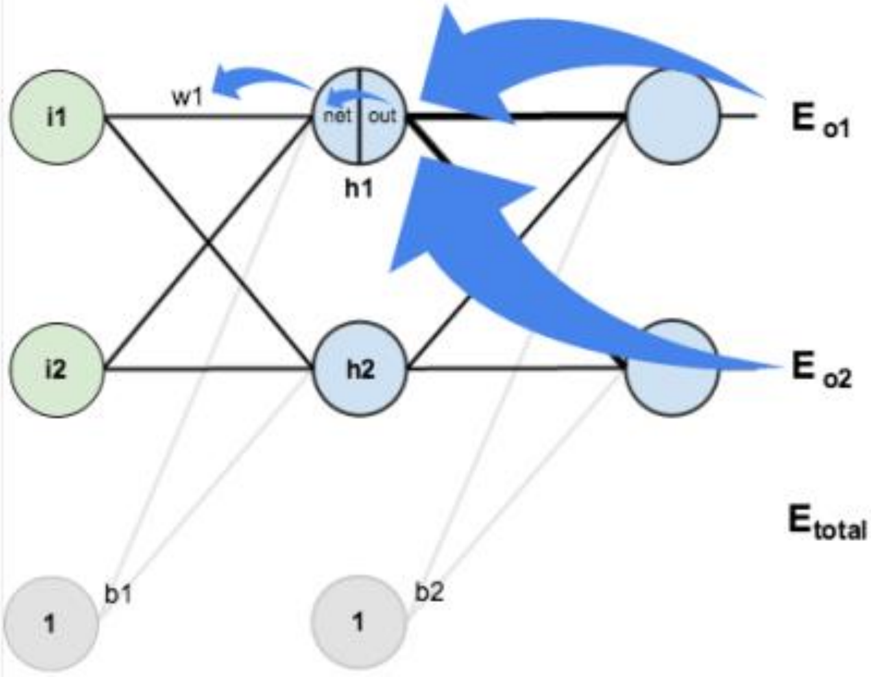
Artificial Neural Network

Using back propagation gradient descent to update w_1

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial out_{h1}} * \frac{\partial out_{h1}}{\partial net_{h1}} * \frac{\partial net_{h1}}{\partial w_1}$$

$$\downarrow$$

$$\frac{\partial E_{total}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial out_{h1}} + \frac{\partial E_{o2}}{\partial out_{h1}}$$



$$E_{total} = E_{o1} + E_{o2}$$