Introduction to artificial neural network technique

K. P. SUDHEER

Associate Professor

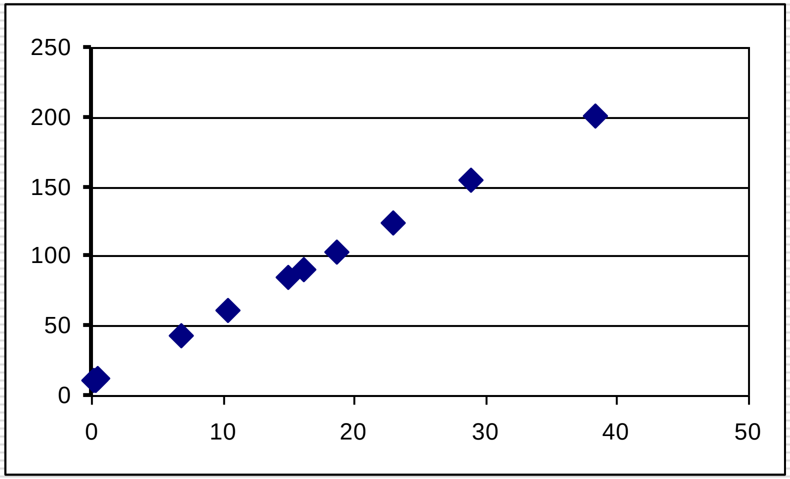
Department of Civil Engineering

Indian Institute of Technology Madras

Chennai, India

Relationship between variables

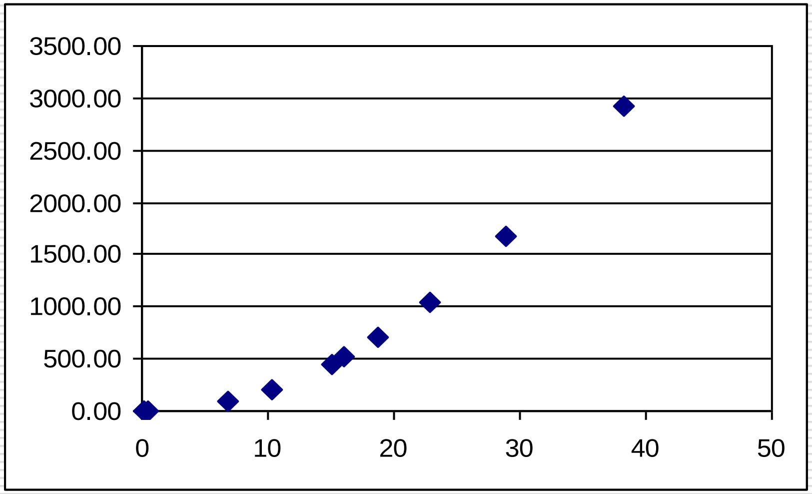
X	Y
0.23	11.15
18.69	103.45
10.31	61.55
0.4	12
0.15	10.75
15	85
38.24	201.2
28.91	154.55
16.06	90.3
22.86	124.3
6.77	43.85



LINEAR

Relationship between variables

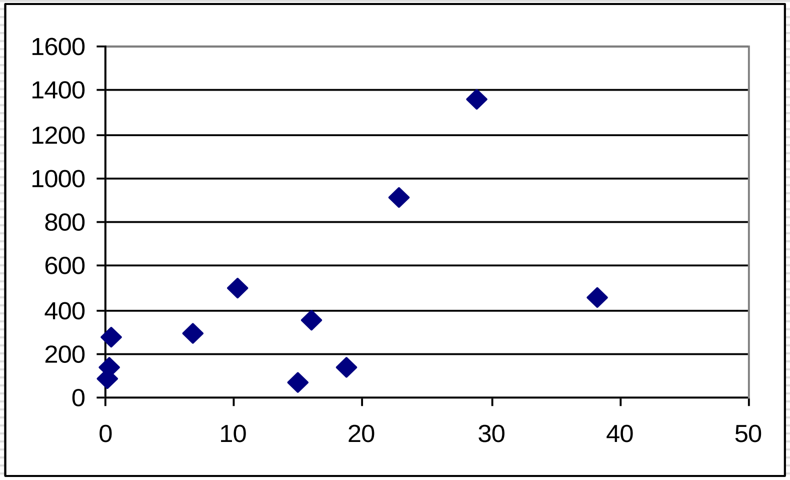
X	Y
0.23	4.13
18.69	698.68
10.31	212.64
0.4	3.72
0.15	4.40
15	450.05
38.24	2924.65
28.91	1671.63
16.06	515.90
22.86	1045.21
6.77	91.72



NON LINEAR: POWER

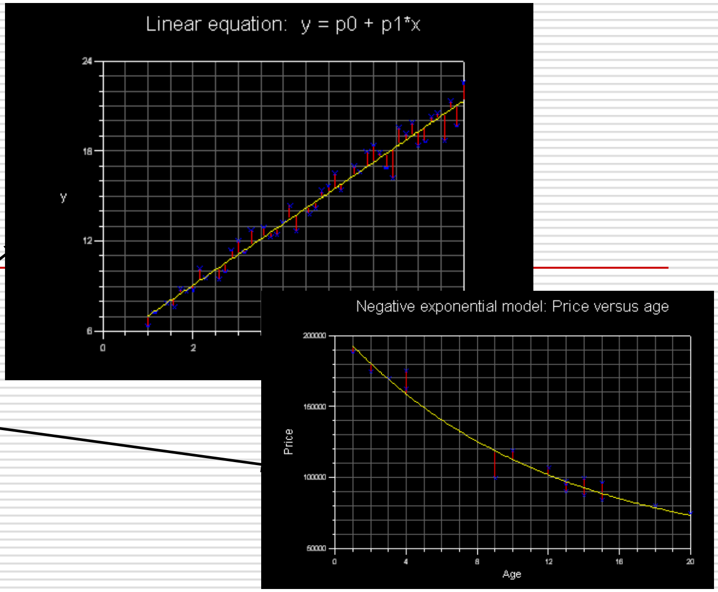
Relationship between variables

X	Y
0.23	139
18.69	135.5
10.31	500.4
0.4	276.8
0.15	88
15	68.5
38.24	453.2
28.91	1358.4
16.06	356
22.86	908.9
6.77	294.3

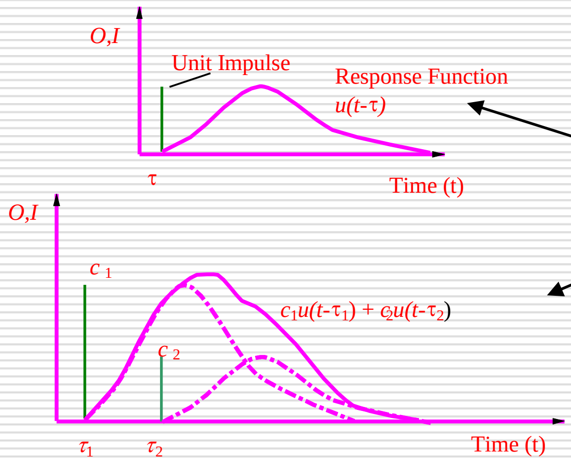


NON LINEAR: FORM ?

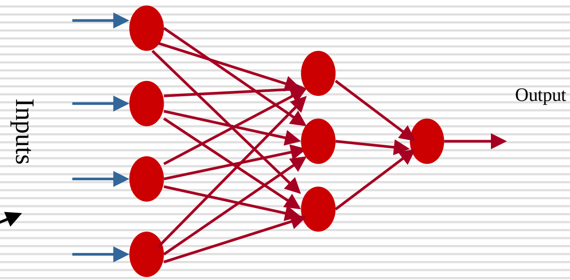
Data Driven Models



Regression analysis



Transfer function techniques



Soft computing techniques
(artificial neural network, fuzzy etc.)

What are Artificial Neural Networks?

An Artificial Intelligence method inspired by the biological neural networks of the human brain;

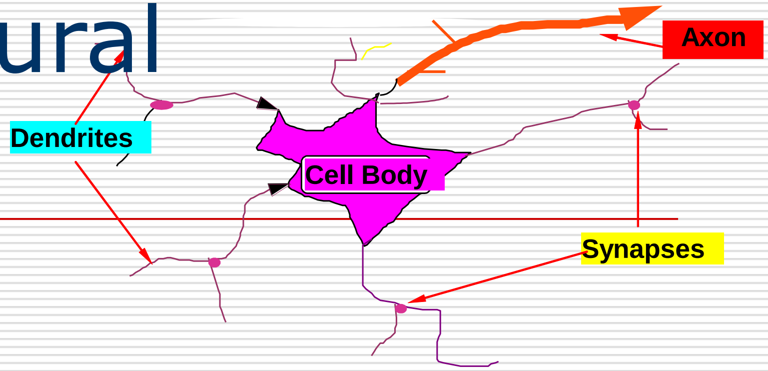
Animals are able to react adaptively to changes in their external and internal environment, and they use their nervous system to perform these behaviours.

Consist of many interconnected simple processors that perform summing functions with information stored in the weights on the connections.

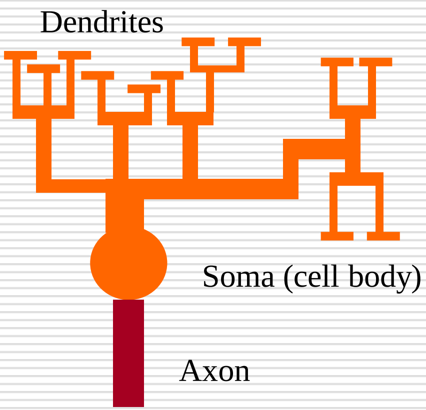
An appropriate model/simulation of the nervous system should be able to produce similar responses and behaviours in artificial systems.

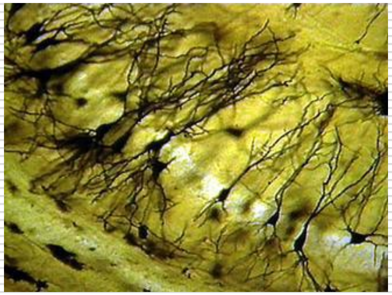
The nervous system is build by relatively simple units, the neurons, so copying their behaviour and functionality should be the solution.

What are Artificial Neural Networks?

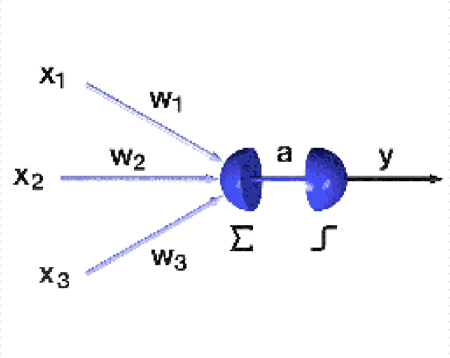


Biological Neural Networks





Artificial Neural Networks



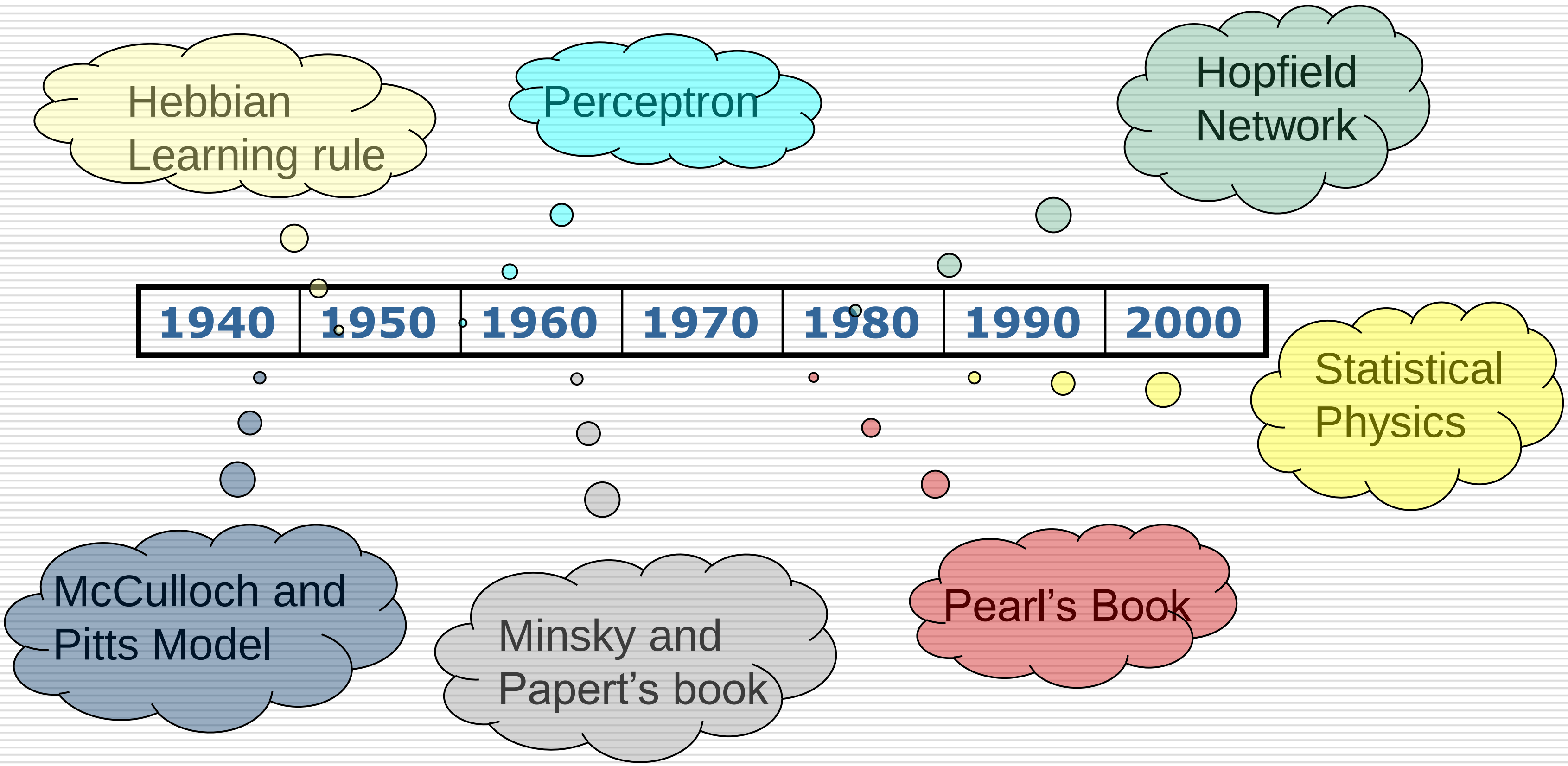
Stimulus

$$\boxed{u_i(t) = \sum_j w_{ij} \cdot x_j(t)}$$

Response

$$\boxed{y_i(t) = f\left(u_{rest} + u_i(t)\right)}$$

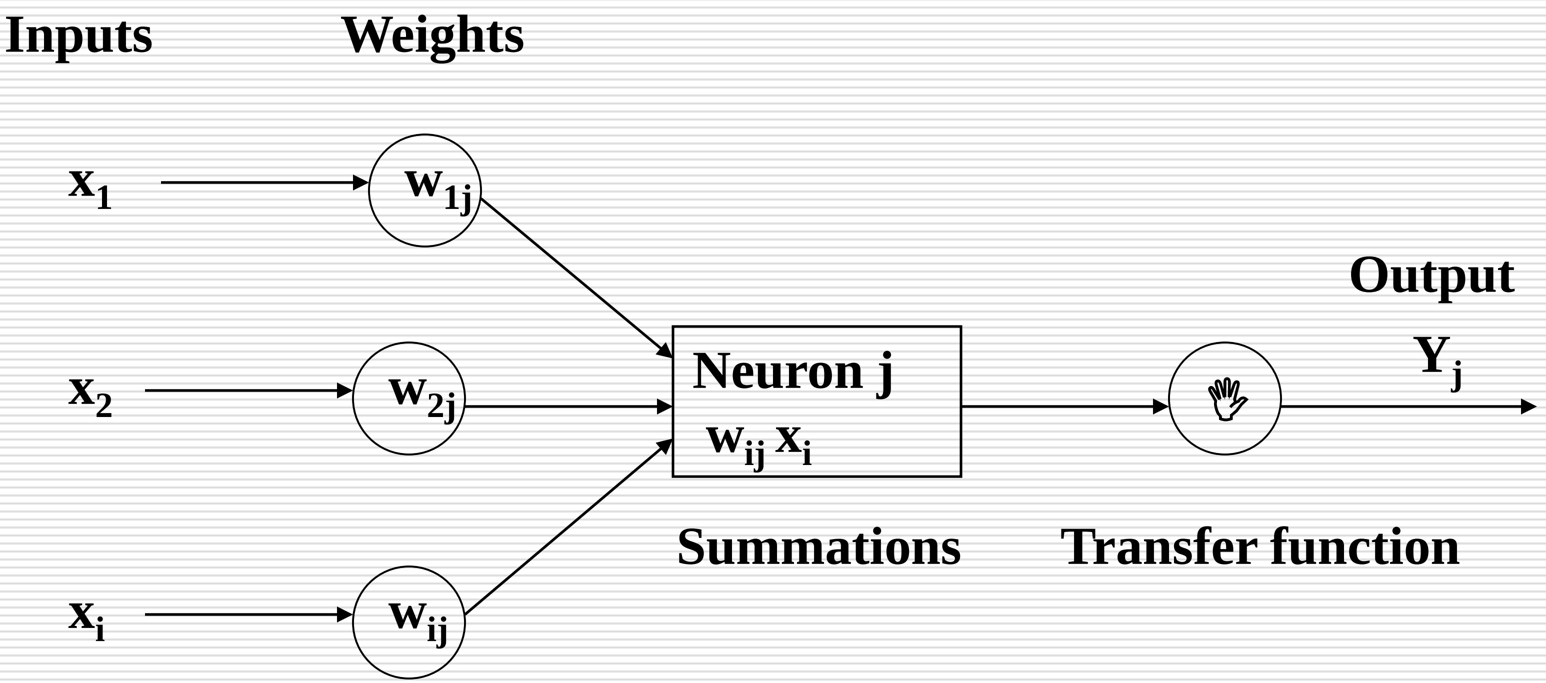
History of Artificial Neural Networks



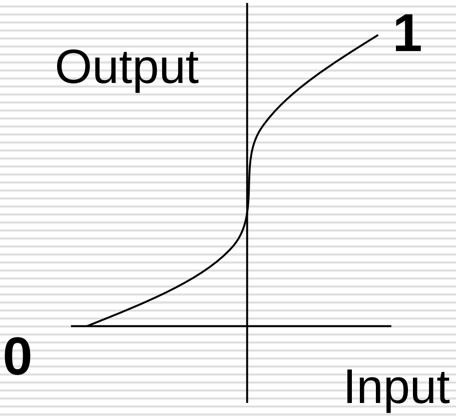
Different ANNs and Applications

Adaline	: Classification
Kohonen	: Speech recognition
Hopefield	: Database management
Elman	: Pattern recognition
MLP	: Mapping & time series
Radial Basis Function	: Mapping & time series

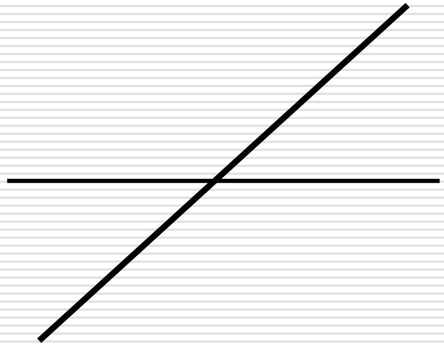
Processing Information in an Artificial Neuron



Transfer Functions

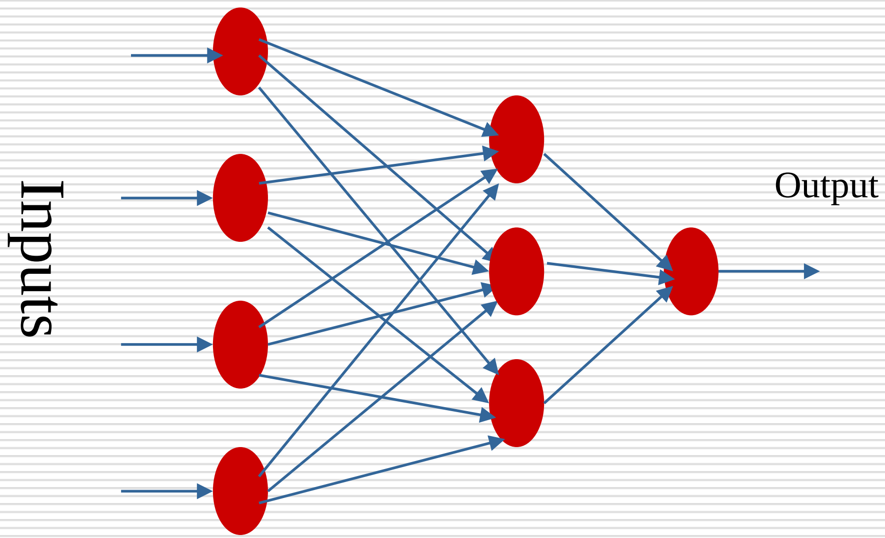


$$SIGMOID: f(n) = \frac{1}{1+e^{-n}}$$



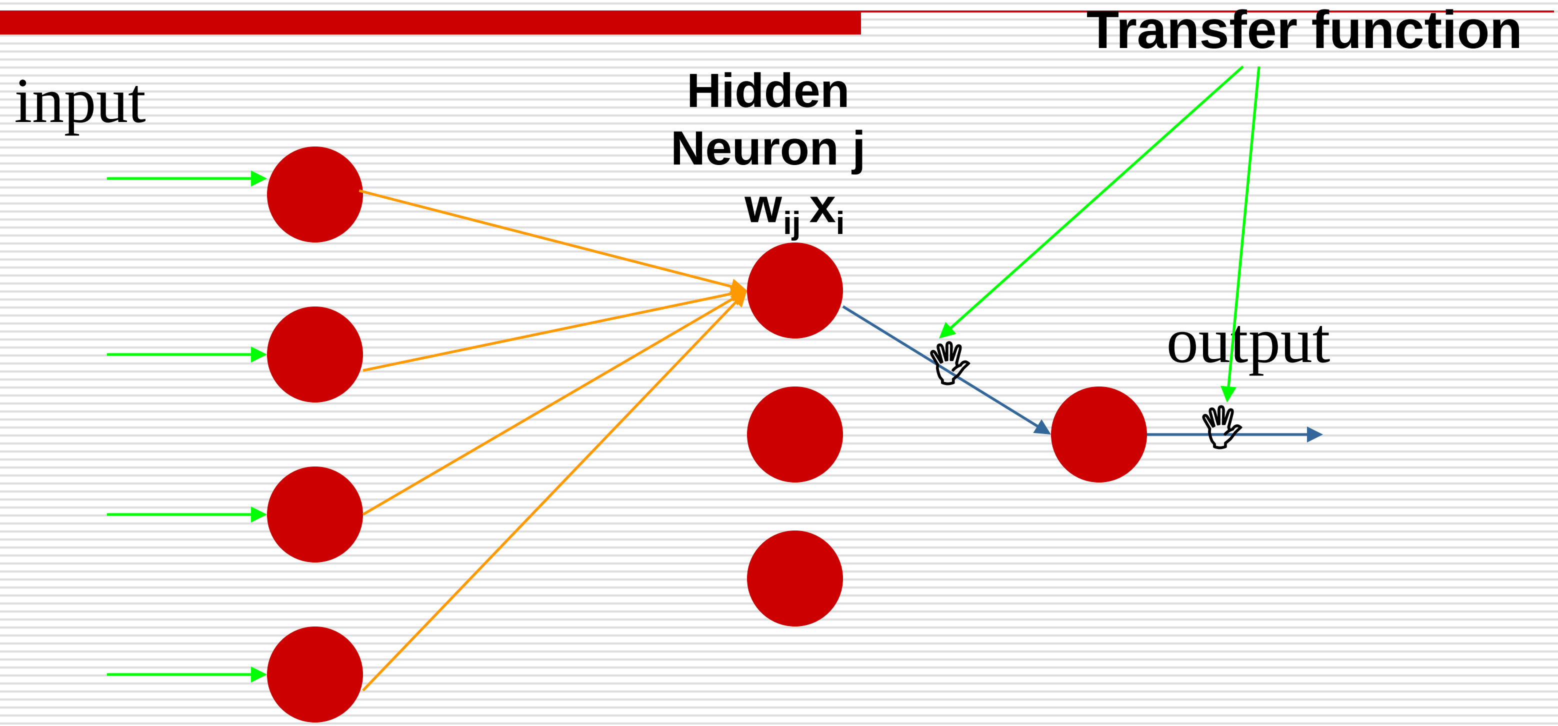
$$LINEAR: f(n) = n$$

Multi Layer Perceptron (MLP) is the most commonly used ANN



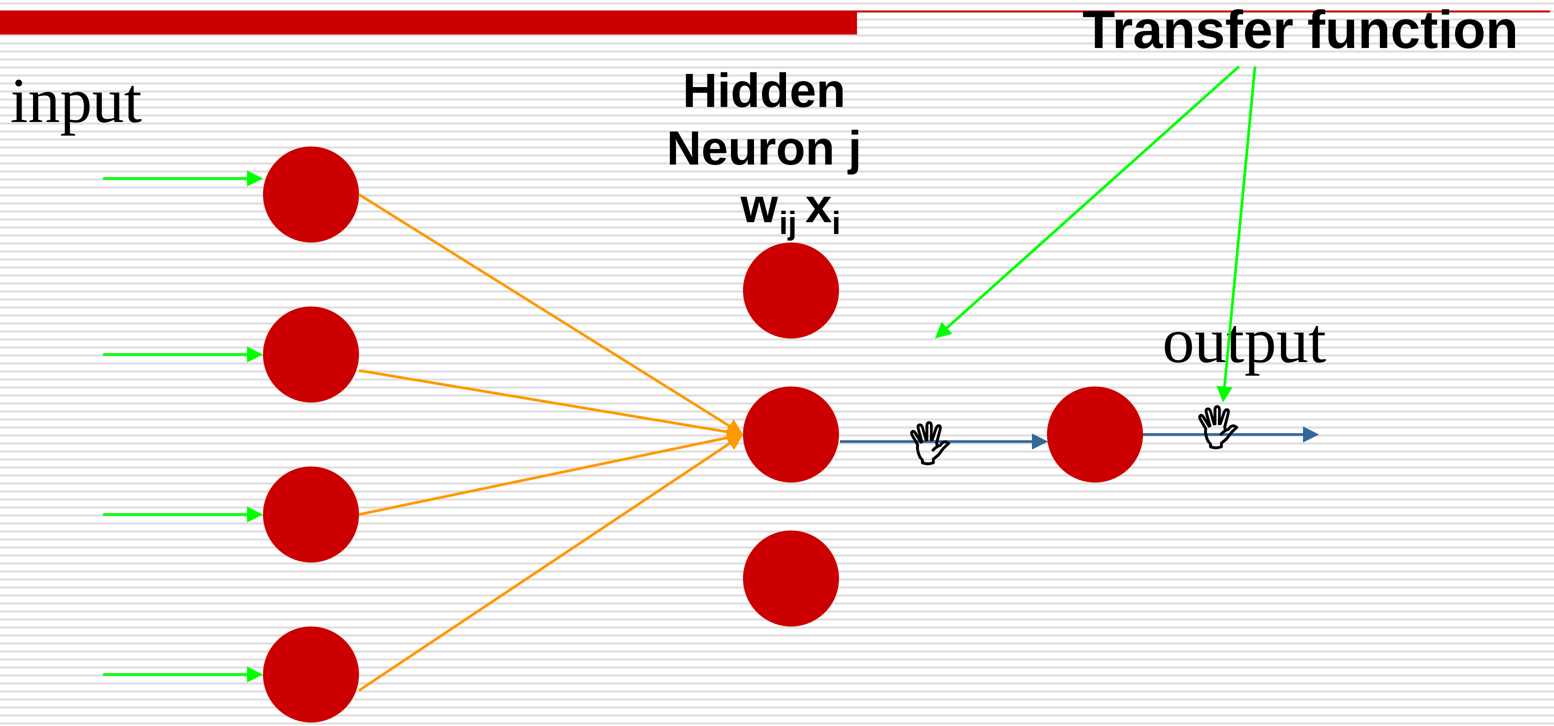
Parameter Estimation ?

Multi Layer Perceptron (MLP)



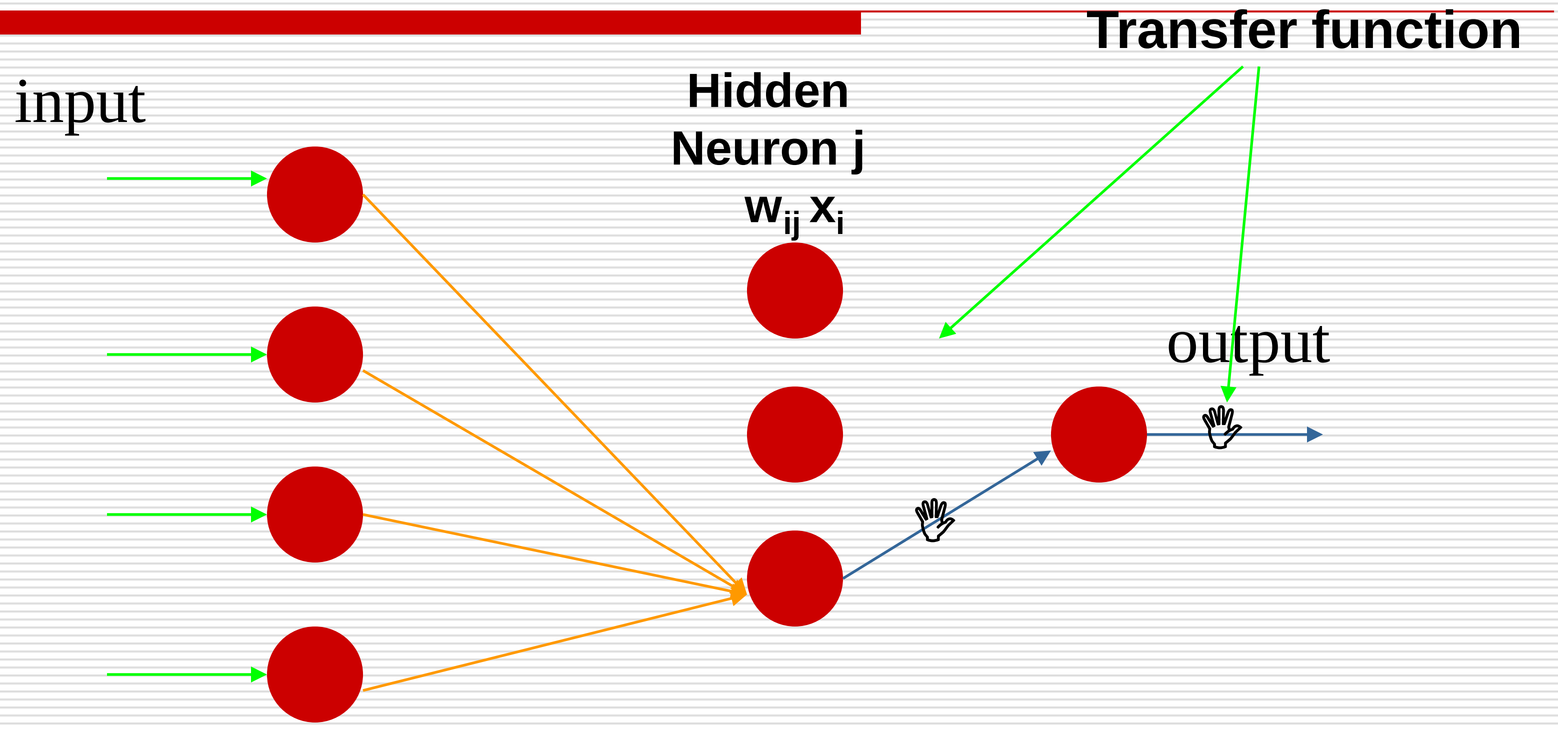
Parameter Estimation ?

Multi Layer Perceptron (MLP)



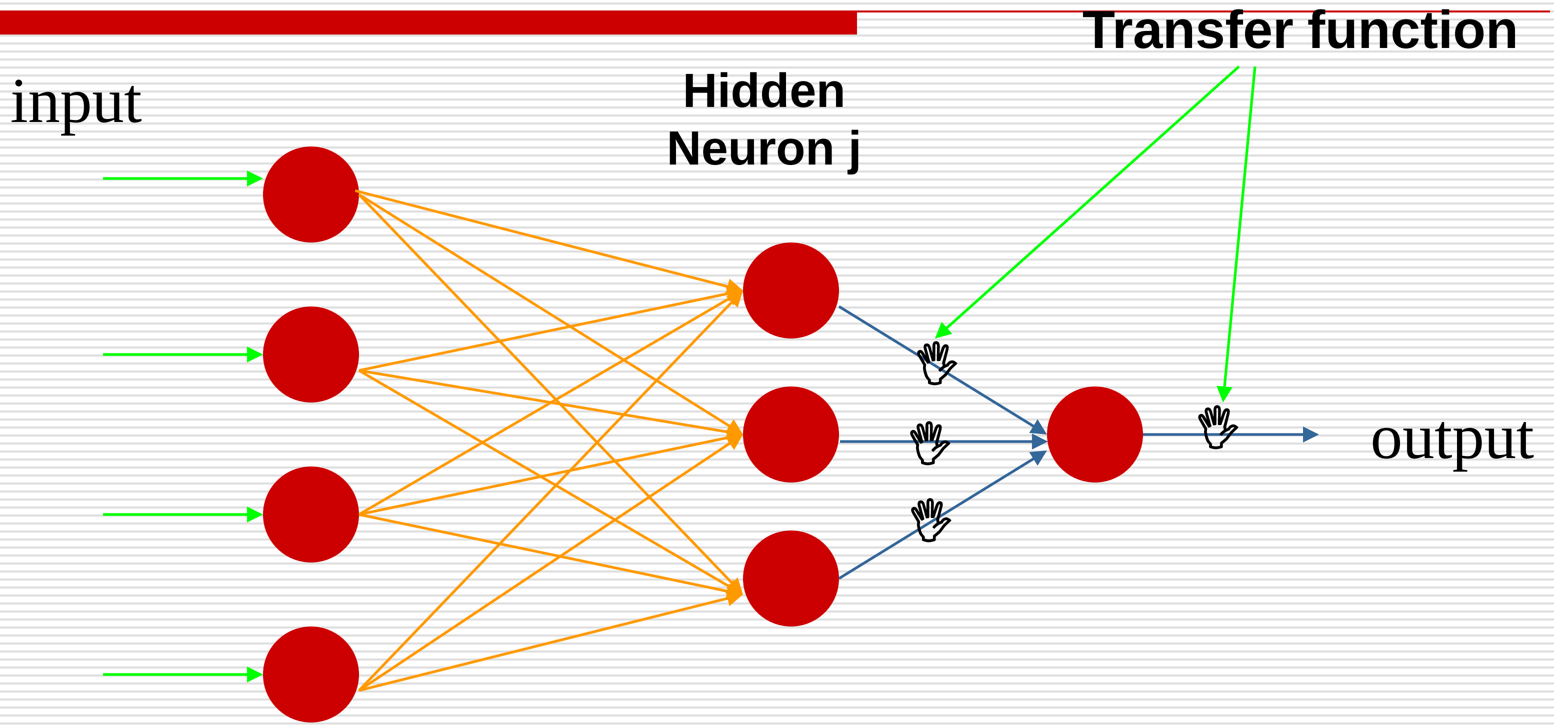
Parameter Estimation ?

Multi Layer Perceptron (MLP)



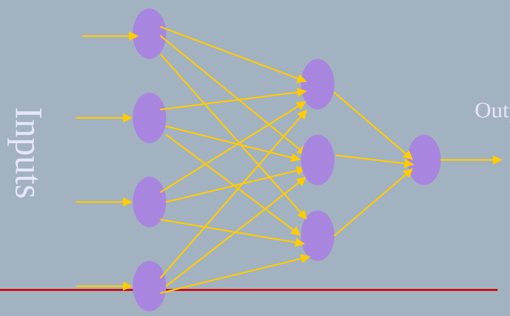
Parameter Estimation ?

Multi Layer Perceptron (MLP)



Parameter Estimation ?

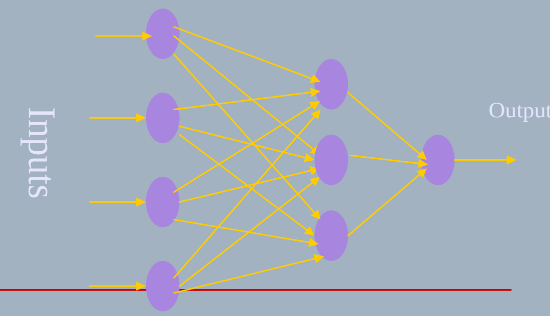
Forward computation



At the hidden layer

$$x_j^{(p)} = \sum_k w_{jk} y_k^{(p)}$$

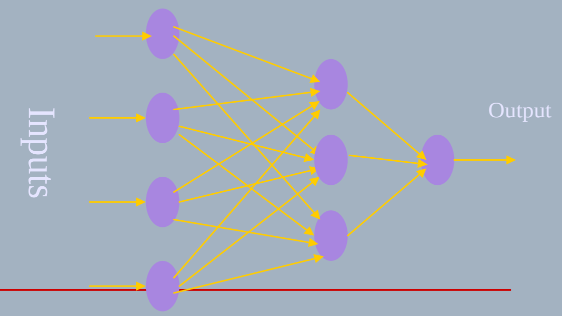
Forward computation



At the hidden layer

$$x_j^{(p)} = \sum_k w_{jk} y_k^{(p)} \qquad y_j^{(p)} = f(x_j^{(p)}) = f\left(\sum_k w_{jk} y_k^{(p)}\right)\frac{1}{2}$$

Forward computation



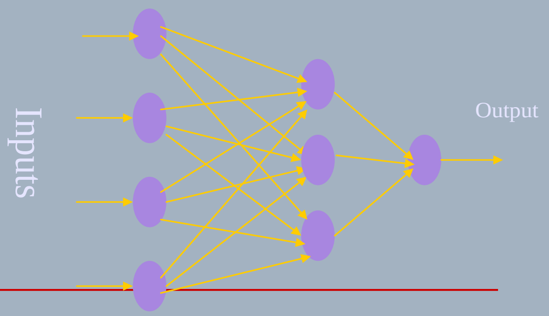
At the hidden layer

$$x_j^{(p)} = \sum_k w_{jk} y_k^{(p)} \qquad y_j^{(p)} = f(x_j^{(p)}) = f\left(\sum_k w_{jk} y_k^{(p)}\right)$$

At the output layer

$$x_i^{(p)} = \sum_j w_{ij} y_j^{(p)} = \sum_j w_{ij}\, f\left(\sum_k w_{jk} y_k^{(p)}\right)$$

Forward computation



At the hidden layer

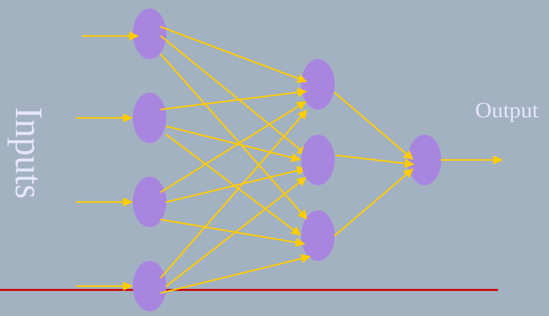
$$x_j^{(p)} = \sum_k w_{jk} y_k^{(p)} \qquad y_j^{(p)} = f(x_j^{(p)}) = f\left(\sum_k w_{jk} y_k^{(p)}\right)$$

At the output layer

$$x_i^{(p)} = \sum_j w_{ij} y_j^{(p)} = \sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)$$

$$y_i^{(p)} = f(x_i^{(p)}) = f\left(\sum_j w_{ij} y_j^{(p)}\right) = f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)\right)$$

Error Back Propagation algorithm



Error computation

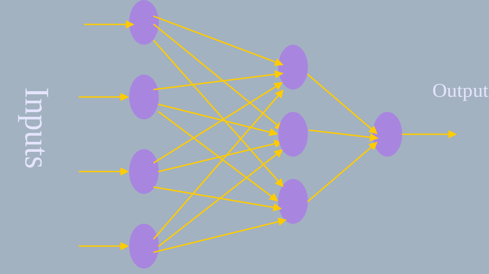
$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)} \right)^2$$

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - f\left(\sum_j w_{ij}\, f\left(\sum_k w_{jk} y_k^{(p)} \right) \right) \right)^2$$

Weight updating (gradient descent)

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}}$$

Error Back Propagation algorithm



Hidden-output connection

$$E=\frac{1}{2}\sum_p\sum_i\left(d_i^{(p)}-y_i^{(p)}\right)^2$$

$$E=\frac{1}{2}\sum_p\sum_i\left(d_i^{(p)}-f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)\right)\right)^2$$

$$x_i^{(p)}=\sum_j w_{ij} y_j^{(p)}=\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)$$

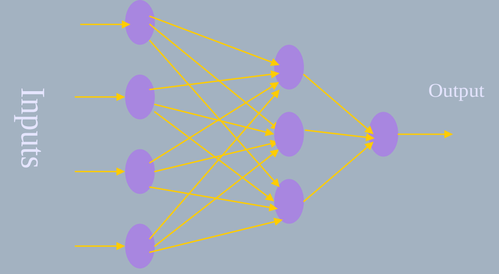
$$y_i^{(p)}=f(x_i^{(p)})=f\left(\sum_j w_{ij} y_j^{(p)}\right)=f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)\right)$$

$$\frac{\partial E}{\partial w_{ij}}=\frac{\partial E}{\partial y_i^{(p)}}\frac{\partial y_i^{(p)}}{\partial w_{ij}}$$

$$\frac{\partial y_i^{(p)}}{\partial w_{ij}}=\frac{\partial y_i^{(p)}}{\partial x_i^{(p)}}\frac{\partial x_i^{(p)}}{\partial w_{ij}}$$

$$\frac{\partial E}{\partial w_{ij}}=\frac{\partial E}{\partial y_i^{(p)}}\frac{\partial y_i^{(p)}}{\partial x_i^{(p)}}\frac{\partial x_i^{(p)}}{\partial w_{ij}}=-\sum_p\left(d_i^{(p)}-y_i^{(p)}\right)f'\left(x_i^{(p)}\right)y_j^{(p)}$$

Error Back Propagation algorithm



Hidden-output connection

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)} \right)^2$$

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_i^{(p)}} \frac{\partial y_i^{(p)}}{\partial w_{ij}}$$

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)} \right) \right) \right)^2$$

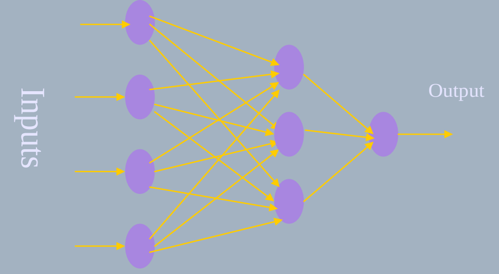
$$x_i^{(p)} = \sum_j w_{ij} y_j^{(p)} = \sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)} \right)$$

$$y_i^{(p)} = f(x_i^{(p)}) = f\left(\sum_j w_{ij} y_j^{(p)} \right) = f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)} \right) \right)$$

$$\frac{\partial y_i^{(p)}}{\partial w_{ij}} = \frac{\partial y_i^{(p)}}{\partial x_i^{(p)}} \frac{\partial x_i^{(p)}}{\partial w_{ij}}$$

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_i^{(p)}} \frac{\partial y_i^{(p)}}{\partial x_i^{(p)}} \frac{\partial x_i^{(p)}}{\partial w_{ij}} = -\sum_p \left(d_i^{(p)} - y_i^{(p)} \right) f'\left(x_i^{(p)} \right) y_j^{(p)}$$

Error Back Propagation algorithm



Hidden-output connection

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right)^2$$

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_i^{(p)}} \frac{\partial y_i^{(p)}}{\partial w_{ij}}$$

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)} \right)\right)\right)^2$$

$$x_i^{(p)} = \sum_j w_{ij} y_j^{(p)} = \sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)} \right)$$

$$y_i^{(p)} = f(x_i^{(p)}) = f\left(\sum_j w_{ij} y_j^{(p)} \right) = f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)} \right)\right)$$

$$\frac{\partial y_i^{(p)}}{\partial w_{ij}} = \frac{\partial y_i^{(p)}}{\partial x_i^{(p)}} \frac{\partial x_i^{(p)}}{\partial w_{ij}}$$

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_i^{(p)}} \frac{\partial y_i^{(p)}}{\partial x_i^{(p)}} \frac{\partial x_i^{(p)}}{\partial w_{ij}} = -\sum_p \left(d_i^{(p)} - y_i^{(p)} \right) f'\left(x_i^{(p)} \right) y_j^{(p)}$$

Error Back Propagation algorithm

$$E=\frac{1}{2}\sum_p\sum_i\left(d_i^{(p)}-y_i^{(p)}\right)^2$$

$$E=\frac{1}{2}\sum_p\sum_i\left(d_i^{(p)}-f\left(\sum_j w_{ij}f\left(\sum_k w_{jk}y_k^{(p)}\right)\right)\right)^2$$

Hidden-output connection

$$x_i^{(p)}=\sum_j w_{ij}y_j^{(p)}=\sum_j w_{ij}f\left(\sum_k w_{jk}y_k^{(p)}\right)$$

$$y_i^{(p)}=f(x_i^{(p)})=f\left(\sum_j w_{ij}y_j^{(p)}\right)=f\left(\sum_j w_{ij}f\left(\sum_k w_{jk}y_k^{(p)}\right)\right)$$

$$\frac{\partial E}{\partial w_{ij}}=\frac{\partial E}{\partial y_i^{(p)}}\frac{\partial y_i^{(p)}}{\partial w_{ij}}$$

$$\frac{\partial y_i^{(p)}}{\partial w_{ij}}=\frac{\partial y_i^{(p)}}{\partial x_i^{(p)}}\frac{\partial x_i^{(p)}}{\partial w_{ij}}$$

Gradient of the output neuron = slope of the transfer function × error

Delta W = Gradient of the neuron x previous output

$$\frac{\partial E}{\partial w_{ij}}=\frac{\partial E}{\partial y_i^{(p)}}\frac{\partial y_i^{(p)}}{\partial x_i^{(p)}}\frac{\partial x_i^{(p)}}{\partial w_{ij}}=-\sum_p\left(d_i^{(p)}-y_i^{(p)}\right)f'\left(x_i^{(p)}\right)y_j^{(p)}$$

$$\Delta w_{ij}=\eta\sum_p\left(d_i^{(p)}-y_i^{(p)}\right)f'\left(x_i^{(p)}\right)y_j^{(p)}=\eta\sum_p\delta_i^{(p)}y_j^{(p)}$$

Error Back Propagation algorithm

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right)^2$$

Input – Hidden connection

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)\right)\right)^2$$

$$x_j^{(p)} = \sum_k w_{jk} y_k^{(p)}$$

$$y_j^{(p)} = f(x_j^{(p)}) = f\left(\sum_k w_{jk} y_k^{(p)}\right)$$

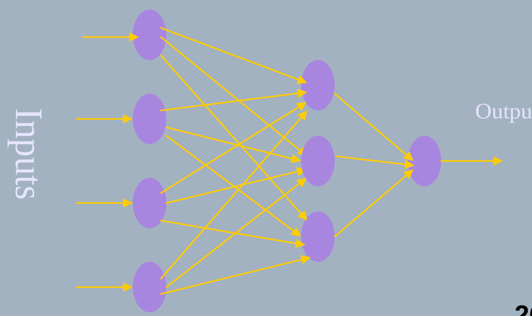
$$x_i^{(p)} = \sum_j w_{ij} y_j^{(p)} = \sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)$$

$$y_i^{(p)} = f(x_i^{(p)}) = f\left(\sum_j w_{ij} y_j^{(p)}\right) = f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)\right)$$

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} \frac{\partial y_j^{(p)}}{\partial w_{jk}}$$

$$\frac{\partial y_j^{(p)}}{\partial w_{jk}} = \frac{\partial y_j^{(p)}}{\partial x_j^{(p)}} \frac{\partial x_j^{(p)}}{\partial w_{jk}}$$

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} \frac{\partial y_j^{(p)}}{\partial x_j^{(p)}} \frac{\partial x_j^{(p)}}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} f'\left(x_j^{(p)}\right) y_k^{(p)}$$



Error Back Propagation algorithm

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right)^2$$

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)\right)\right)^2$$

Input – Hidden connection

$$x_j^{(p)} = \sum_k w_{jk} y_k^{(p)}$$

$$y_j^{(p)} = f(x_j^{(p)}) = f\left(\sum_k w_{jk} y_k^{(p)}\right)$$

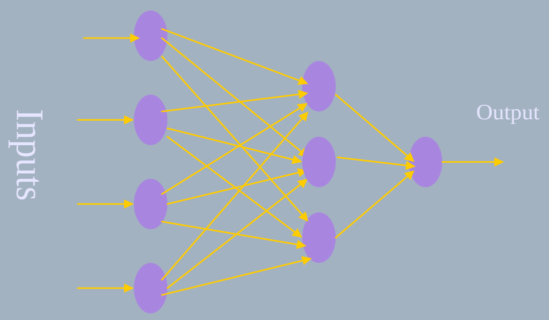
$$x_i^{(p)} = \sum_j w_{ij} y_j^{(p)} = \sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)$$

$$y_i^{(p)} = f(x_i^{(p)}) = f\left(\sum_j w_{ij} y_j^{(p)}\right) = f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)\right)$$

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} \frac{\partial y_j^{(p)}}{\partial w_{jk}}$$

$$\frac{\partial y_j^{(p)}}{\partial w_{jk}} = \frac{\partial y_j^{(p)}}{\partial x_j^{(p)}} \frac{\partial x_j^{(p)}}{\partial w_{jk}}$$

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} \frac{\partial y_j^{(p)}}{\partial x_j^{(p)}} \frac{\partial x_j^{(p)}}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} f'\left(x_j^{(p)}\right) y_k^{(p)}$$



Error Back Propagation algorithm

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right)^2$$

Input – Hidden connection

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)} \right) \right) \right)^2$$

$$x_j^{(p)} = \sum_k w_{jk} y_k^{(p)}$$

$$y_j^{(p)} = f(x_j^{(p)}) = f\left(\sum_k w_{jk} y_k^{(p)} \right)$$

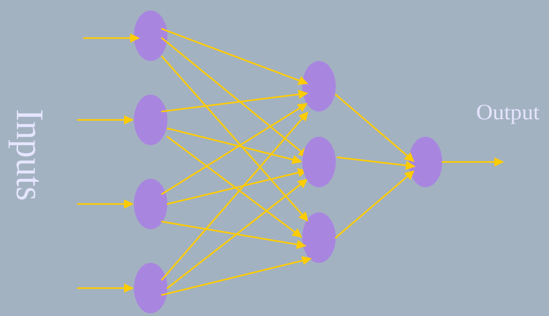
$$x_i^{(p)} = \sum_j w_{ij} y_j^{(p)} = \sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)} \right)$$

$$y_i^{(p)} = f(x_i^{(p)}) = f\left(\sum_j w_{ij} y_j^{(p)} \right) = f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)} \right) \right)$$

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} \frac{\partial y_j^{(p)}}{\partial w_{jk}}$$

$$\frac{\partial y_j^{(p)}}{\partial w_{jk}} = \frac{\partial y_j^{(p)}}{\partial x_j^{(p)}} \frac{\partial x_j^{(p)}}{\partial w_{jk}}$$

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} \frac{\partial y_j^{(p)}}{\partial x_j^{(p)}} \frac{\partial x_j^{(p)}}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} f'\left(x_j^{(p)}\right) y_k^{(p)}$$



Error Back Propagation algorithm

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right)^2$$

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)\right)\right)^2$$

Input – Hidden connection

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} \frac{\partial y_j^{(p)}}{\partial w_{jk}}$$

$$\frac{\partial y_j^{(p)}}{\partial w_{jk}} = \frac{\partial y_j^{(p)}}{\partial x_j^{(p)}} \frac{\partial x_j^{(p)}}{\partial w_{jk}}$$

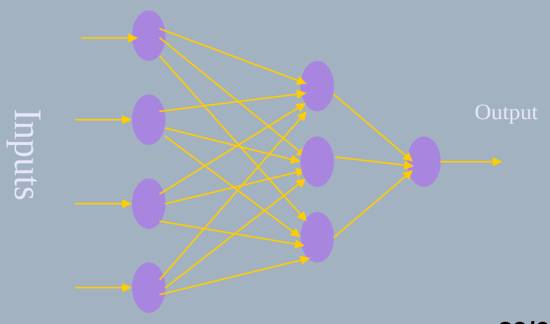
$$x_j^{(p)} = \sum_k w_{jk} y_k^{(p)}$$

$$y_j^{(p)} = f(x_j^{(p)}) = f\left(\sum_k w_{jk} y_k^{(p)}\right)$$

$$x_i^{(p)} = \sum_j w_{ij} y_j^{(p)} = \sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)$$

$$y_i^{(p)} = f(x_i^{(p)}) = f\left(\sum_j w_{ij} y_j^{(p)}\right) = f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)\right)$$

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} \frac{\partial y_j^{(p)}}{\partial x_j^{(p)}} \frac{\partial x_j^{(p)}}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} f'\left(x_j^{(p)}\right) y_k^{(p)}$$



Error Back Propagation algorithm

Input – Hidden connection

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right)^2$$

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)\right)\right)^2$$

$$x_j^{(p)} = \sum_k w_{jk} y_k^{(p)}$$

$$y_j^{(p)} = f(x_j^{(p)}) = f\left(\sum_k w_{jk} y_k^{(p)}\right)$$

$$x_i^{(p)} = \sum_j w_{ij} y_j^{(p)} = \sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)$$

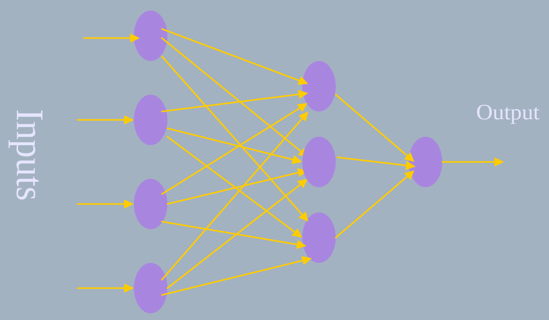
$$y_i^{(p)} = f(x_i^{(p)}) = f\left(\sum_j w_{ij} y_j^{(p)}\right) = f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)\right)$$

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} \frac{\partial y_j^{(p)}}{\partial x_j^{(p)}} \frac{\partial x_j^{(p)}}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} f'\left(x_j^{(p)}\right) y_k^{(p)}$$

$$\frac{\partial E}{\partial y_j^{(p)}} = -\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right) \frac{\partial\left(f\left(x_i^{(p)}\right)\right)}{\partial y_j^{(p)}}$$

$$= -\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right) \frac{\partial\left(f\left(x_i^{(p)}\right)\right)}{\partial x_i^{(p)}} \frac{\partial x_i^{(p)}}{\partial y_j^{(p)}}$$

$$= -\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right) f'\left(x_i^{(p)}\right) w_{ij}$$



Error Back Propagation algorithm

Input – Hidden connection

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right)^2$$

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)} \right)\right)\right)^2$$

$$x_j^{(p)} = \sum_k w_{jk} y_k^{(p)}$$

$$y_j^{(p)} = f(x_j^{(p)}) = f\left(\sum_k w_{jk} y_k^{(p)} \right)$$

$$x_i^{(p)} = \sum_j w_{ij} y_j^{(p)} = \sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)} \right)$$

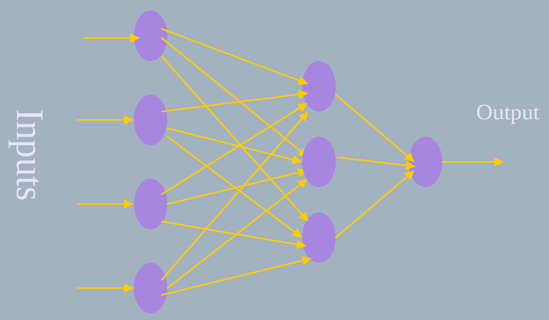
$$y_i^{(p)} = f(x_i^{(p)}) = f\left(\sum_j w_{ij} y_j^{(p)} \right) = f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)} \right)\right)$$

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} \frac{\partial y_j^{(p)}}{\partial x_j^{(p)}} \frac{\partial x_j^{(p)}}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} f'\left(x_j^{(p)}\right) y_k^{(p)}$$

$$\frac{\partial E}{\partial y_j^{(p)}} = -\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right) \frac{\partial\left(f\left(x_i^{(p)}\right)\right)}{\partial y_j^{(p)}}$$

$$= -\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right) \frac{\partial\left(f\left(x_i^{(p)}\right)\right)}{\partial x_i^{(p)}} \frac{\partial x_i^{(p)}}{\partial y_j^{(p)}}$$

$$= -\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right) f'\left(x_i^{(p)}\right) w_{ij}$$



Error Back Propagation algorithm

Input – Hidden connection

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right)^2$$

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)} \right)\right)\right)^2$$

$$x_j^{(p)} = \sum_k w_{jk} y_k^{(p)}$$

$$y_j^{(p)} = f(x_j^{(p)}) = f\left(\sum_k w_{jk} y_k^{(p)} \right)$$

$$x_i^{(p)} = \sum_j w_{ij} y_j^{(p)} = \sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)} \right)$$

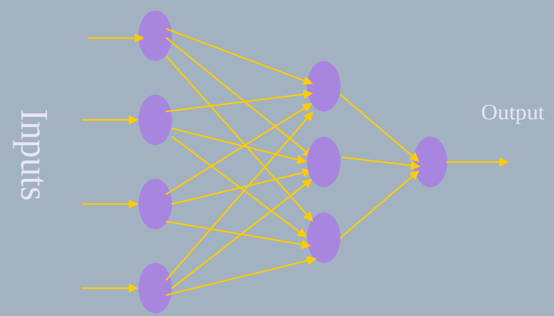
$$y_i^{(p)} = f(x_i^{(p)}) = f\left(\sum_j w_{ij} y_j^{(p)} \right) = f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)} \right)\right)$$

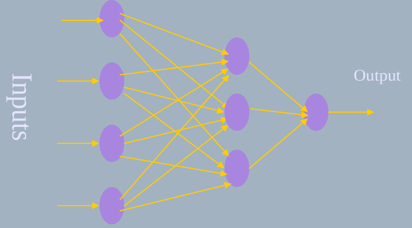
$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} \frac{\partial y_j^{(p)}}{\partial x_j^{(p)}} \frac{\partial x_j^{(p)}}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} f'\left(x_j^{(p)}\right) y_k^{(p)}$$

$$\frac{\partial E}{\partial y_j^{(p)}} = -\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right) \frac{\partial\left(f\left(x_i^{(p)}\right)\right)}{\partial y_j^{(p)}}$$

$$= -\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right) \frac{\partial\left(f\left(x_i^{(p)}\right)\right)}{\partial x_i^{(p)}} \frac{\partial x_i^{(p)}}{\partial y_j^{(p)}}$$

$$= -\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right) f'\left(x_i^{(p)}\right) w_{ij}$$





Error Back Propagation algorithm

Input – Hidden connection

$$E=\frac{1}{2}\sum_p\sum_i\left(d_i^{(p)}-y_i^{(p)}\right)^2$$

$$E=\frac{1}{2}\sum_p\sum_i\left(d_i^{(p)}-f\left(\sum_j w_{ij}\,f\left(\sum_k w_{jk}y_k^{(p)}\right)\right)\right)^2$$

$$\frac{\partial E}{\partial w_{jk}}=\frac{\partial E}{\partial y_j^{(p)}}\frac{\partial y_j^{(p)}}{\partial x_j^{(p)}}\frac{\partial x_j^{(p)}}{\partial w_{jk}}=\frac{\partial E}{\partial y_j^{(p)}}f'\left(x_j^{(p)}\right)y_k^{(p)}$$

$$x_j^{(p)}=\sum_k w_{jk}y_k^{(p)}$$

$$y_j^{(p)}=f(x_j^{(p)})=f\left(\sum_k w_{jk}y_k^{(p)}\right)$$

$$x_i^{(p)}=\sum_j w_{ij}y_j^{(p)}=\sum_j w_{ij}\,f\left(\sum_k w_{jk}y_k^{(p)}\right)$$

$$y_i^{(p)}=f(x_i^{(p)})=f\left(\sum_j w_{ij}y_j^{(p)}\right)=f\left(\sum_j w_{ij}\,f\left(\sum_k w_{jk}y_k^{(p)}\right)\right)$$

$$\frac{\partial E}{\partial y_j^{(p)}}=-\sum_p\sum_i\left(d_i^{(p)}-y_i^{(p)}\right)\frac{\partial\left(f\left(x_i^{(p)}\right)\right)}{\partial y_j^{(p)}}$$

$$=-\sum_p\sum_i\left(d_i^{(p)}-y_i^{(p)}\right)\frac{\partial\left(f\left(x_i^{(p)}\right)\right)}{\partial x_i^{(p)}}\frac{\partial x_i^{(p)}}{\partial y_j^{(p)}}$$

$$=-\sum_p\sum_i\left(d_i^{(p)}-y_i^{(p)}\right)f'\left(x_i^{(p)}\right)w_{ij}$$

$$\frac{\partial E}{\partial w_{jk}}=-\sum_p\sum_i\left(d_i^{(p)}-y_i^{(p)}\right)f'\left(x_i^{(p)}\right)w_{ij}\,f'\left(x_j^{(p)}\right)y_k^{(p)}$$

Error Back Propagation algorithm

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right)^2$$

$$E = \frac{1}{2}\sum_p \sum_i \left(d_i^{(p)} - f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)\right)\right)^2$$

Input – Hidden connection

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} \frac{\partial y_j^{(p)}}{\partial x_j^{(p)}} \frac{\partial x_j^{(p)}}{\partial w_{jk}} = \frac{\partial E}{\partial y_j^{(p)}} f'\left(x_j^{(p)}\right) y_k^{(p)}$$

$$x_j^{(p)} = \sum_k w_{jk} y_k^{(p)}$$

$$y_j^{(p)} = f(x_j^{(p)}) = f\left(\sum_k w_{jk} y_k^{(p)}\right)$$

$$x_i^{(p)} = \sum_j w_{ij} y_j^{(p)} = \sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)$$

$$y_i^{(p)} = f(x_i^{(p)}) = f\left(\sum_j w_{ij} y_j^{(p)}\right) = f\left(\sum_j w_{ij} f\left(\sum_k w_{jk} y_k^{(p)}\right)\right)$$

$$\frac{\partial E}{\partial y_j^{(p)}} = -\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right) \frac{\partial\left(f\left(x_i^{(p)}\right)\right)}{\partial y_j^{(p)}}$$

$$= -\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right) \frac{\partial\left(f\left(x_i^{(p)}\right)\right)}{\partial x_i^{(p)}} \frac{\partial x_i^{(p)}}{\partial y_j^{(p)}}$$

$$= -\sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right) f'\left(x_i^{(p)}\right) w_{ij}$$

$$\Delta w_{jk} = \eta \sum_p \sum_i \left(d_i^{(p)} - y_i^{(p)}\right) f'\left(x_i^{(p)}\right) w_{ij} f'\left(x_j^{(p)}\right) y_k^{(p)}$$

$$= \eta \sum_p \sum_i \delta_i^{(p)} w_{ij} f'\left(x_j^{(p)}\right) y_k^{(p)}$$

$$= \eta \sum_p \delta_j^{(p)} y_k^{(p)}$$

Gradient of the neuron= slope of the transfer function×[Σ{(weight of the neuron to the next neuron) × (gradient of the next neuron)}]

Delta W = Gradient of the neuron x previous output

Backpropagation Algorithm (Rumelhart et al., 1986)

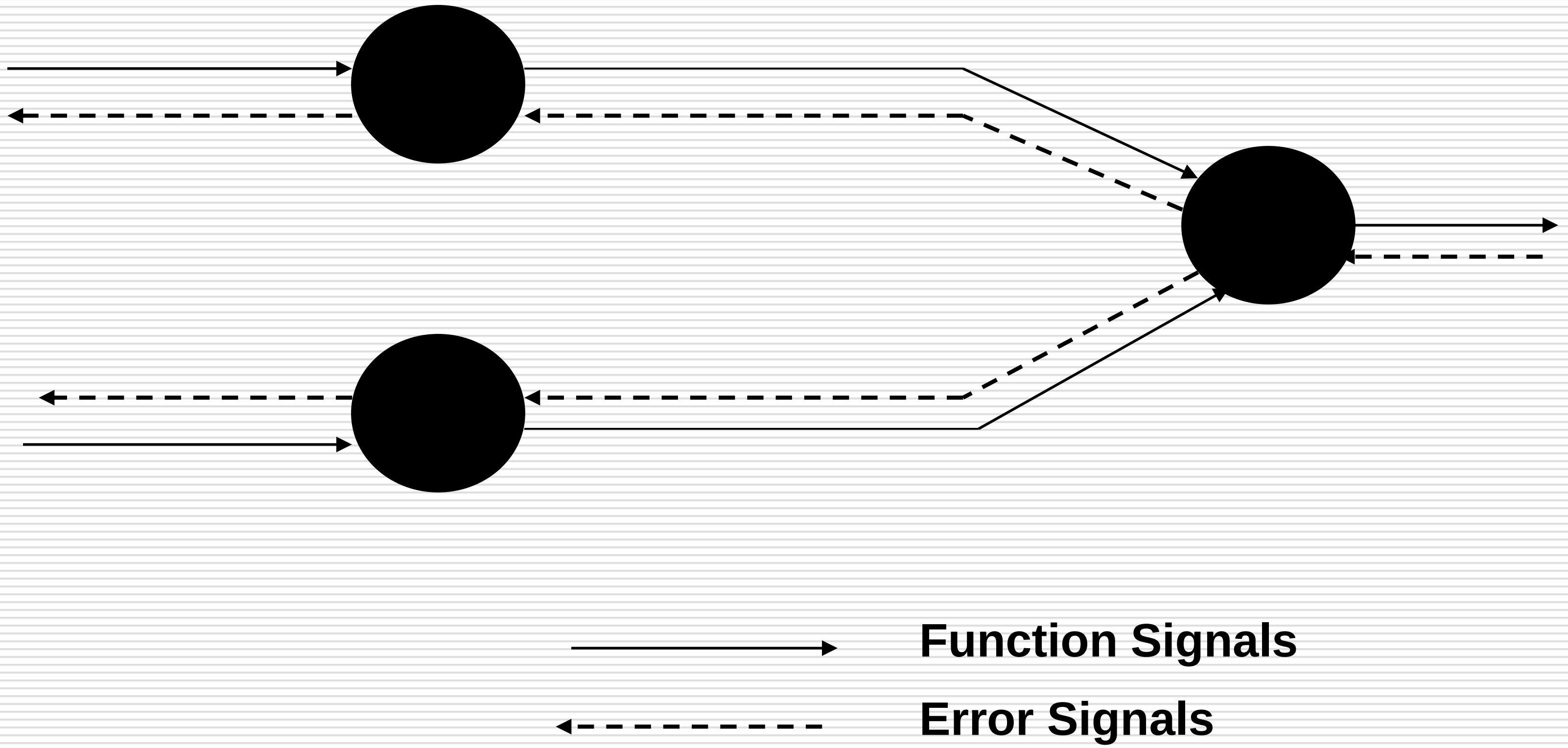


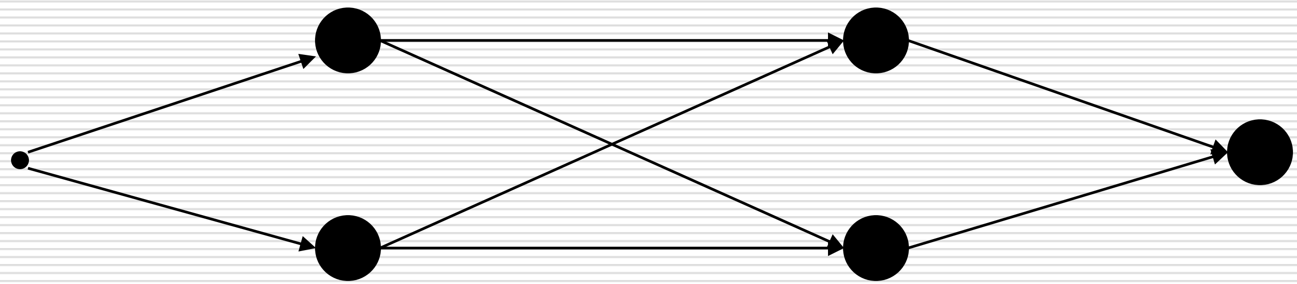
Illustration of Backpropagation Algorithm (y = x)

- Hidden layer transfer function: Sigmoid function = F(n)= 1/(1+exp(-n)), where n is the net input to the neuron.

Derivative= F'(n) = (output of the neuron)(1-output of the neuron) : Slope of the transfer function.

- Output layer transfer function: Linear function= F(n)=n; Output=Input to the neuron

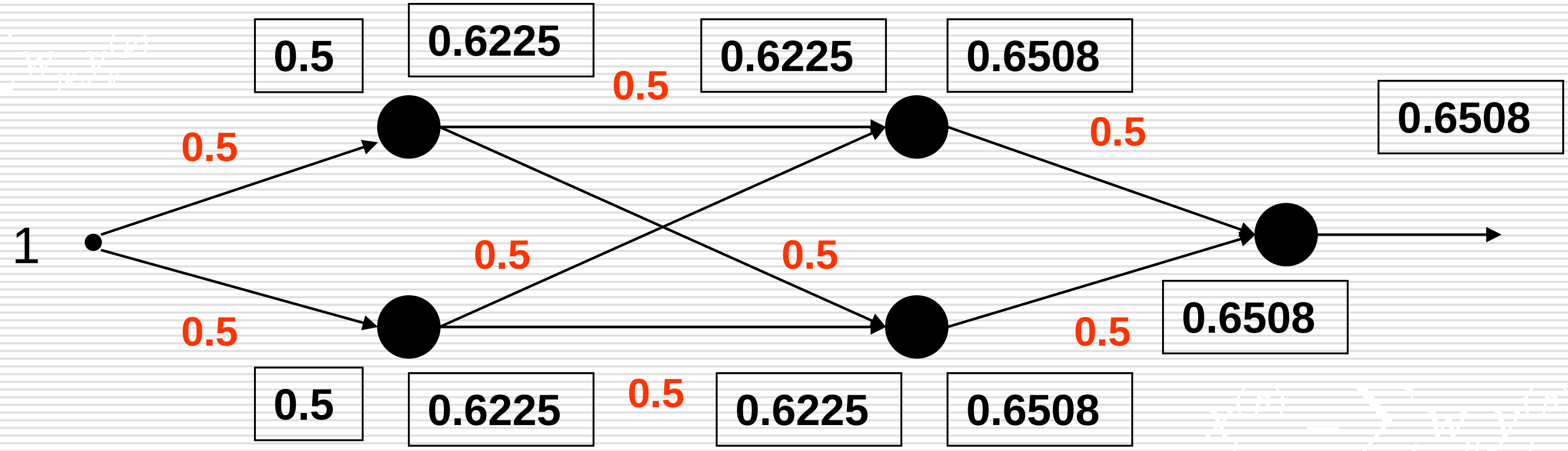
Derivative= F'(n)= 1



First Pass

G1= (0.6225)(1-0.6225)(0.0397)(0.5)(2)=0.0093

G2= (0.6508)(1-0.6508)(0.3492)(0.5)=0.0397



Gradient of the neuron= **G** =slope of the transfer function×[Σ{(weight of the neuron to the next neuron) × (gradient of the next neuron)}]

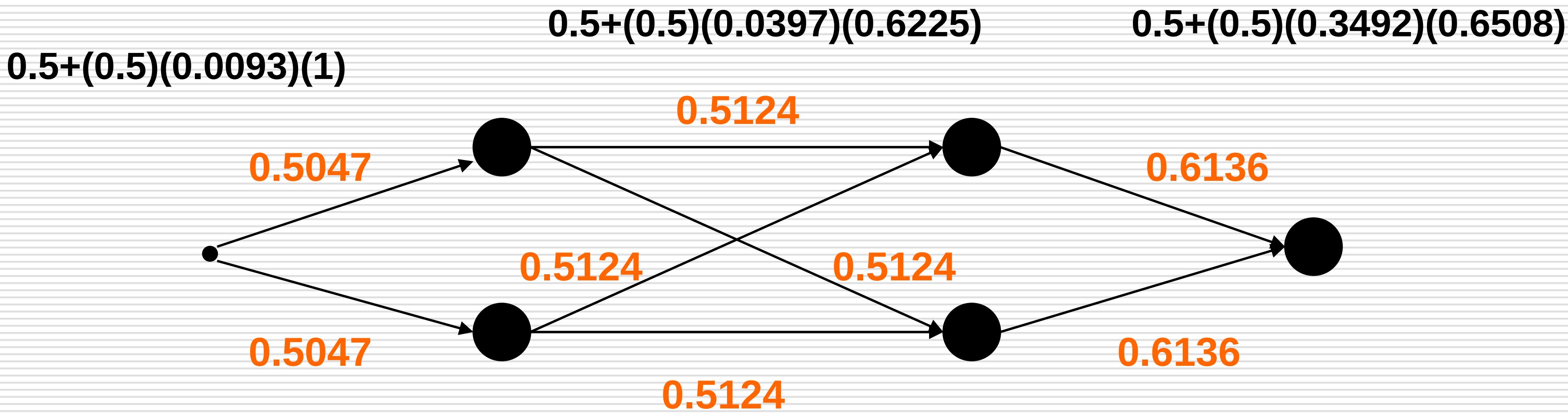
Gradient of the output neuron = slope of the transfer function × error

G3=(1)(0.3492)=0.3492

Error=1-0.6508=0.3492

Weight Update 1

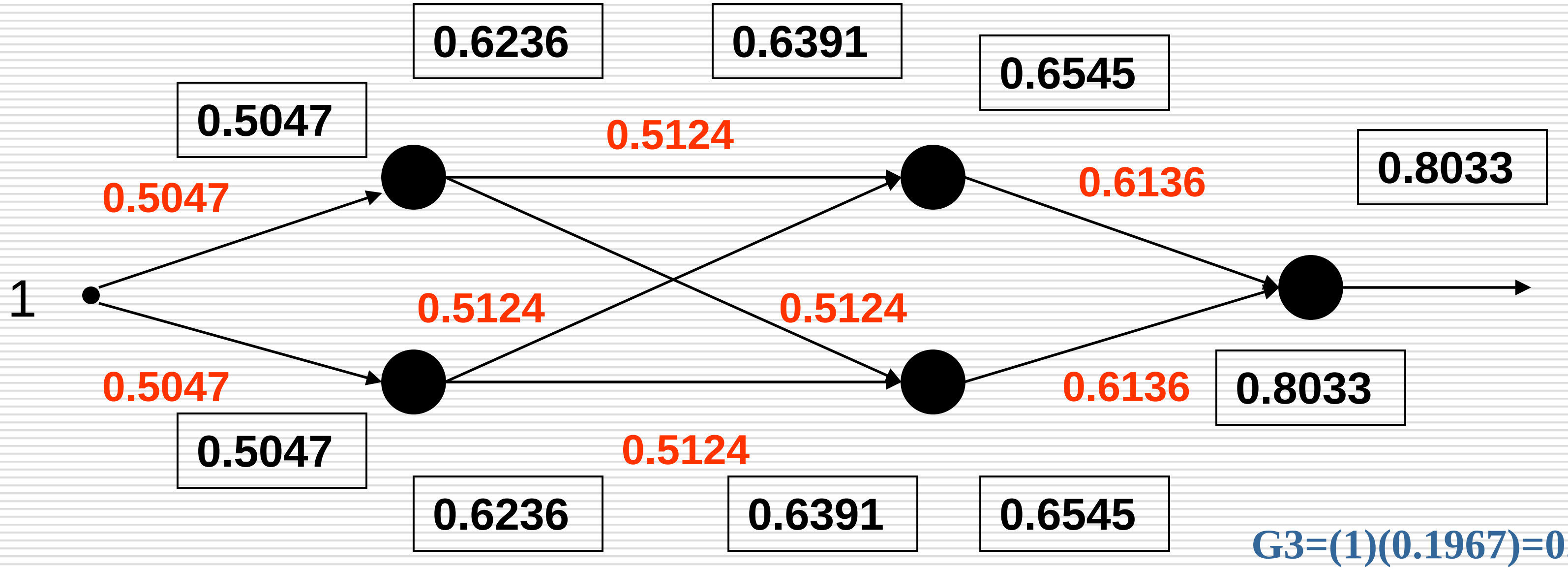
New Weight=Old Weight + {(learning rate)(gradient)(prior input)}



Second Pass

G1= (0.6236)(1-0.6236)(0.5124)(0.0273)(2)=0.0066

G2= (0.6545)(1-0.6545)(0.1967)(0.6136)=0.0273



G3=(1)(0.1967)=0.1967

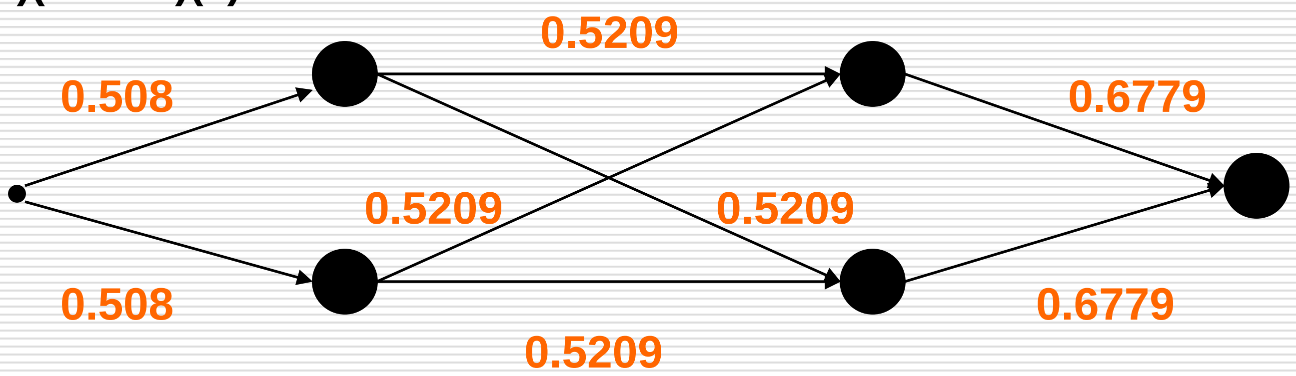
Error=1-0.8033=0.1967

Weight Update 2

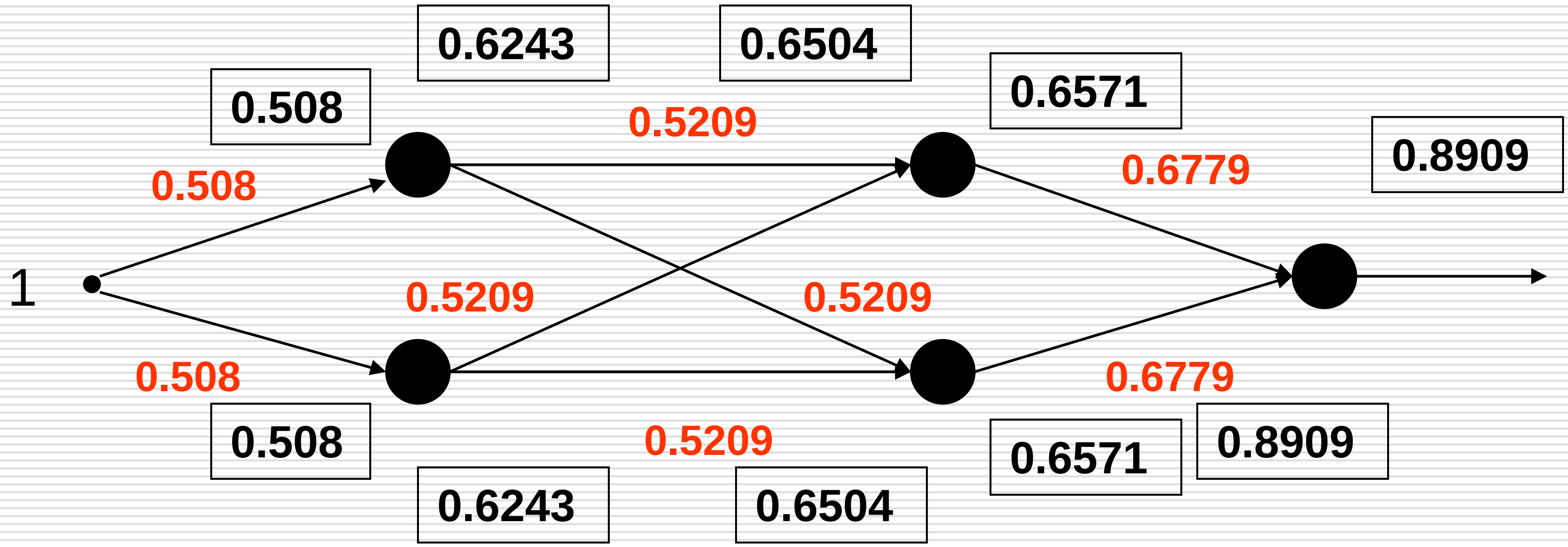
New Weight=Old Weight + {(learning rate)(gradient)(prior output)}

0.5124+(0.5)(0.0273)(0.6236) **0.6136+(0.5)(0.1967)(0.6545)**

0.5047+(0.5)(0.0066)(1)



Third Pass


0.6243
0.6504
0.508
0.6571
0.5209
0.8909
0.508
0.6779
1
0.5209
0.5209
0.508
0.6779
0.508
0.8909
0.6571
0.5209
0.6243
0.6504

Weight Update Summary

	Weights			Output	Expected Output	Error
	w1	w2	w3			
Initial conditions	0.5	0.5	0.5	0.6508	1	0.3492
Pass 1 Update	0.5047	0.5124	0.6136	0.8033	1	0.1967
Pass 2 Update	0.508	0.5209	0.6779	0.8909	1	0.1091

W1: Weights from the input to the input layer
W2: Weights from the input layer to the hidden layer
W3: Weights from the hidden layer to the output layer

Training Algorithm

- The process of feedforward and backpropagation continues until the required mean squared error has been reached.
- Typical mse: 1e-5
- Other complicated backpropagation training algorithms also available.

Gradient in Detail

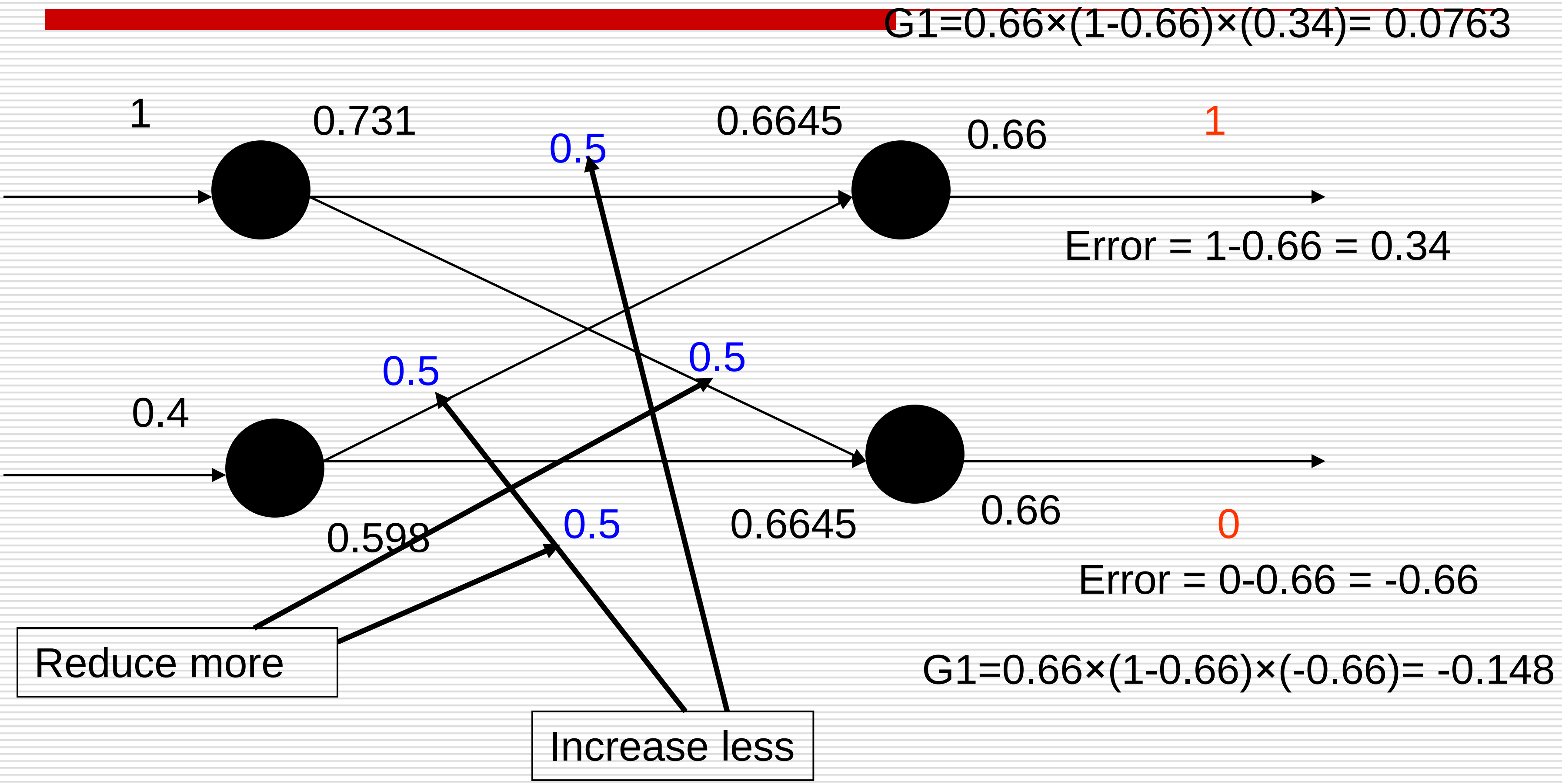
- Gradient : Rate of change of error w.r.t rate of change in net input to neuron
  - o For output neurons
    - ▪ Slope of the transfer function × error

  - o For hidden neurons : A bit complicated ! : error fed back in terms of gradient of successive neurons

    - ▪ Slope of the transfer function × [Σ (gradient of next neuron × weight connecting the neuron to the next neuron)]
    - ▪ Why summation? Share the responsibility!!

  - o ***Therefore: Credit Assignment Problem***

An Example


G1=0.66×(1-0.66)×(0.34)= 0.0763
1
0.731
0.5
0.6645
0.66
1
Error = 1-0.66 = 0.34
0.5
0.5
0.4
0.598
0.5
0.6645
0.66
0
Error = 0-0.66 = -0.66
G1=0.66×(1-0.66)×(-0.66)= -0.148
Reduce more
Increase less

Improving performance

- Changing the number of layers and number of neurons in each layer.
- Variation in Transfer functions.
- Changing the learning rate.
- Training for longer times.
- Type of pre-processing and post-processing.

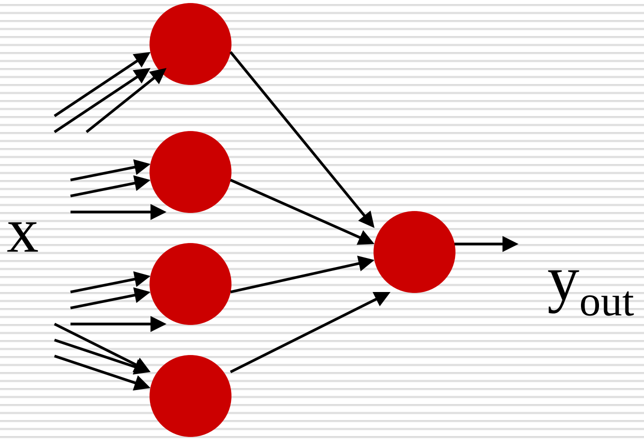
RBF neural networks

RBF = radial basis function

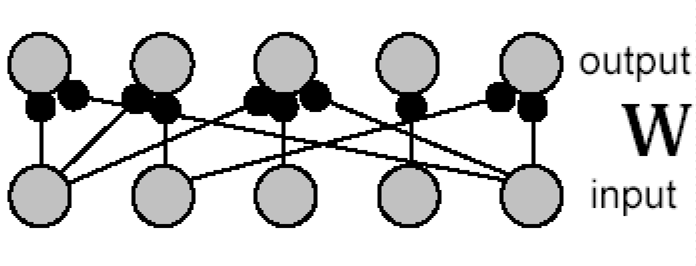
$$r(x) = \bar{r}(\| x - c \|)$$

Example: $$f(x) = e^{-\frac{\|x-w\|^2}{2a^2}}$$ Gaussian RBF

$$y_{out} = \sum_{k=1}^{4} w_k^2 \cdot e^{-\frac{\|x-w^{1,k}\|^2}{2(a_k)^2}}$$

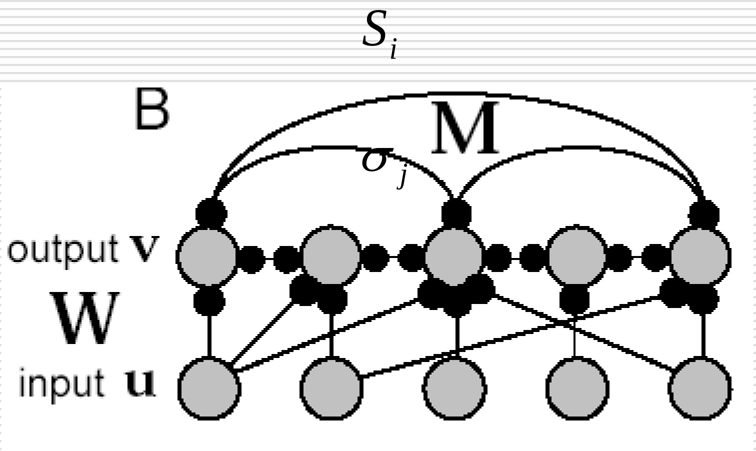


Recurrent Neural Network



Feedforward:

Recurrent:



Learning: Three Tasks

1. Compute Outputs
2. Compare Outputs with Desired Targets
3. Adjust Weights and Repeat the Process

ANNApplication Development

- Preliminary steps of system development are done
- ANN Application Development Process

1. Collect Data
2. Separate into Training and Test Sets
3. Define a Network Structure
4. Select a Learning Algorithm
5. Set Parameters, Values, Initialize Weights
6. Transform Data to Network Inputs
7. Start Training, and Determine and Revise Weights
8. Stop and Test
9. Implementation: Use the Network with New Cases

ANN Model Building

Decisions the builder must make

- Input and Output vector
- Size of training and test data
- Learning algorithms
- Topology: number of processing elements and their configurations
- Transformation (transfer) function
- Learning rate for each layer
- Diagnostic and validation tools

Input/Output Selection

$$Y^m = f(X^n)$$

X^n **is an** n**-dimensional input vector consisting of variables** $x_1, \ldots\ldots, x_i, \ldots, x_n$

Y^m ***is an*** m**-dimensional output vector consisting of the resulting variables of interest** $y_1, \ldots\ldots, y_i, \ldots\ldots, y_m$

Data Division into Training/Validation

Tokar and Johnson (1999) report that the way the data are divided affect the results.

Shahin et al. (2000) suggest that the statistical properties of the data subsets should indicate that each subset represent the same population.

Cross validation or bootstrapping is employed to remove the uncertainty in parameter estimation (Sudheer et al., 2002).

Training Algorithm

The training is a non-linear optimization of an error function.

Several training algorithms are available and the most commonly employed is the Backpropagation (Rumelhart et al., 1986)

Evolutionary algorithms such as Genetic Algorithm, Simulated Annealing etc. are being employed recently

Standard Backpropagation
Cascade Correlation
GA based algorithms
Simulated annealing based algorithms
Orthogonal Least Square Algorithms

Network Topology

Input Vector
Hidden Layers/ Hidden Neurons
Output Vector

Data Pre-Processing

Scaling/Standardization

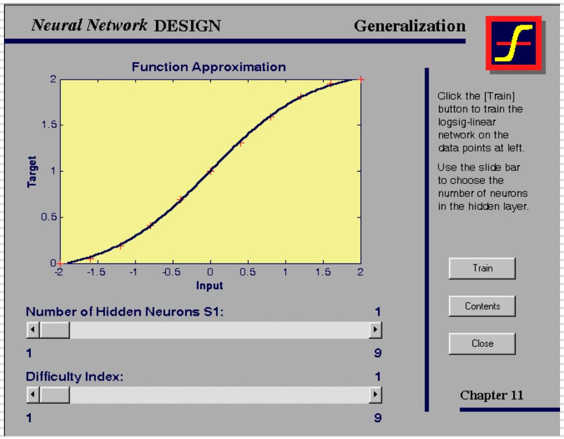
Range 0 to1 or 0.1 to 0.9 or –1 to +1 etc.

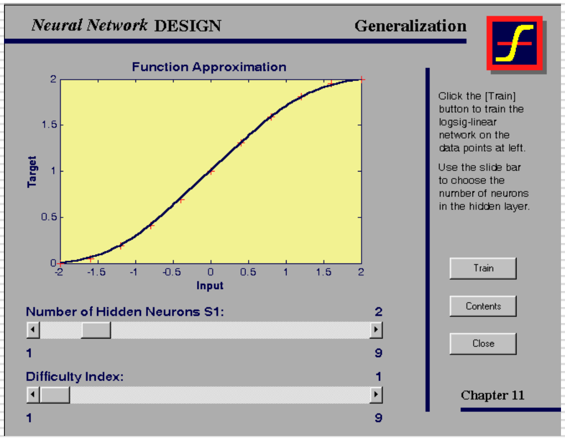
Minns and Hall (1996) as well as Maier and Dandy (2000) have discussed the importance of scaling.

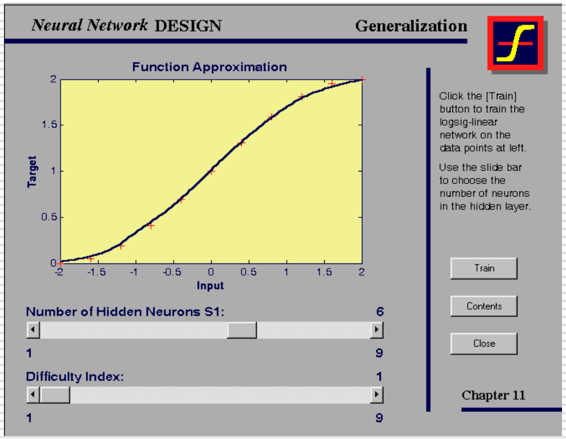
Sudheer et al. (2003)

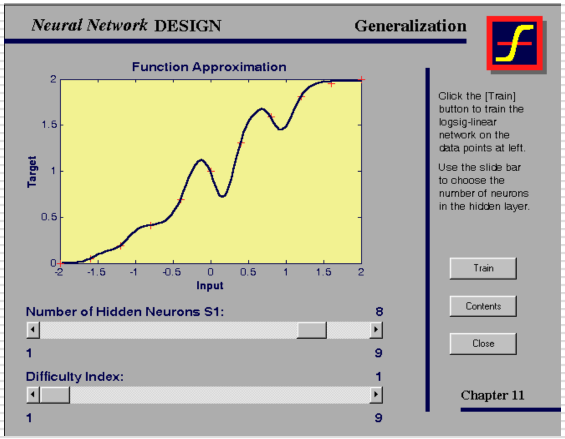
Reports that a transformation of the data into normal domain improves the model performance

Effect of Hidden Neuron









Transfer Function ?

Learning rate ?

Momentum rate ?

Performance Evaluation

No single evaluation measure is available (Sudheer and Jain, 2003) and generally a multi criteria assessment is performed.

Two types: relative and absolute

- Relative indices are non-dimensional and give a relative comparison with other models

- Absolute statistics are in measures unit and gives the individual models' capability.

McCabe (1999) recommend any evaluation should include a combination of the two.

Summary

Learning tasks of artificial neural networks can be reformulated as function approximation tasks.

Neural networks can be considered as nonlinear function approximating tools (i.e., linear combinations of nonlinear basis functions), where the parameters of the networks should be found by applying optimisation methods.

The optimisation is done with respect to the approximation error measure.

In general it is enough to have a single hidden layer neural network (MLP, RBF or other) to learn the approximation of a nonlinear function. In such cases general optimisation can be applied to find the change rules for the synaptic weights.

Challenges Ahead

There are no theories that describes the best method for constructing a network for a given application

The uncertainty in model outputs have to quantified to improve the confidence in models predictions