

Artificial Intelligence & Machine Learning and Pattern Recognition — — Neural Network



Yanghui Rao

Assistant Prof., Ph.D

School of Mobile Information Engineering,

Sun Yat-sen University

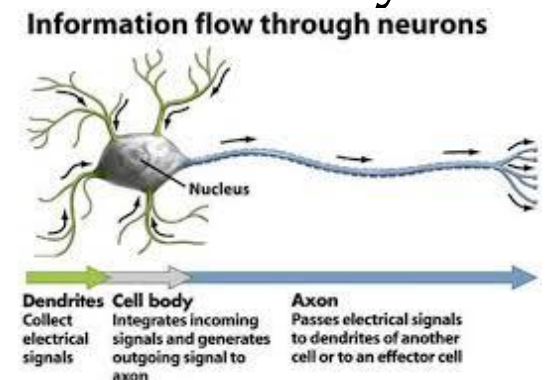
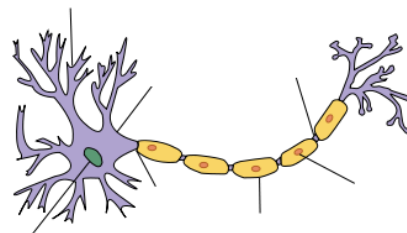
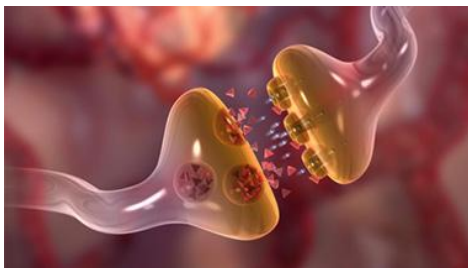
raoyangh@mail.sysu.edu.cn

Neural Network

- The study of artificial neural networks was inspired by attempts to simulate biological neural systems.
- The human brain consists of nerve cells called neurons (神经元) primarily .
- Neurons are linked together via strands of fiber called axons (轴突).
- Axons are used to transmit nerve impulses from one neuron to another whenever the neurons are stimulated.

Neural Network

- A neuron is connected to the axons of other neurons via dendrites (树突), which are extensions from the cell body of the neuron.
- The contact point between a dendrite and an axon is called a synapse (突触).
- The human brain learns by changing the strength of the synaptic connection between neurons upon repeated stimulation by the same impulse.



Neural Network

- A neural network consists of a large number of simple and interacting nodes (artificial neurons).
- Knowledge is represented by the strength of connections between these nodes.
- Knowledge is acquired by adjusting the connections through a process of learning.
- All the neurons process their inputs simultaneously and independently.

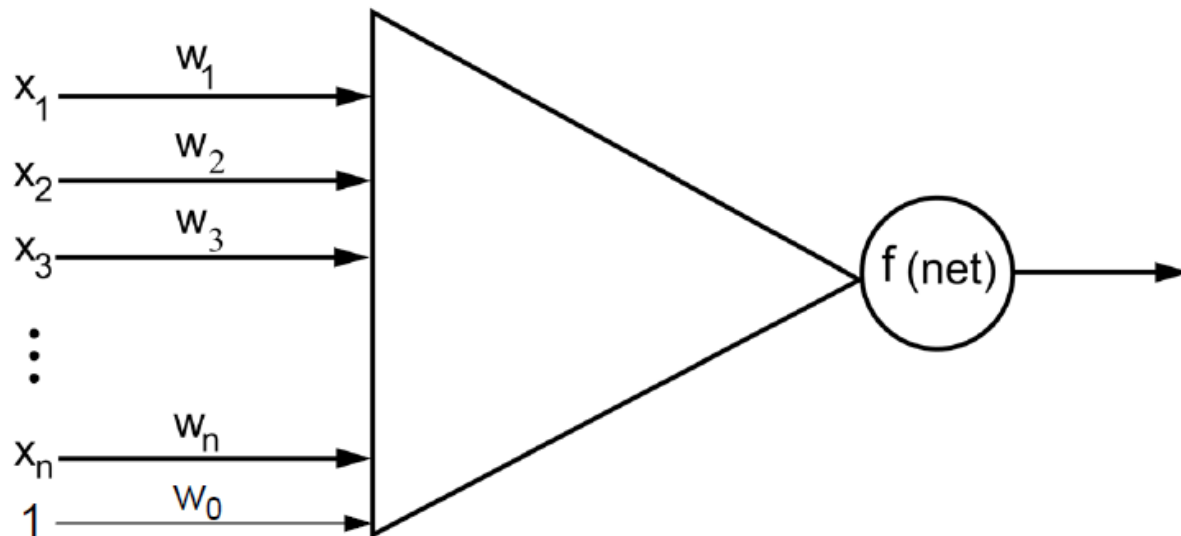
Artificial Neuron

- The unit of computation in neural networks is the artificial neuron.
- An artificial neuron consists of
 - Input signals x_i . These signals represent data from the environment or activation of other neurons.
 - A set of real-valued weights w_i . The values of these weights represent connection strengths.
 - An activation level $\sum_i w_i x_i$. The neuron's activation level is determined by the sum of the weighted inputs.
 - A threshold function f . This function computes the final output by determining if the activation is below or above a threshold.

Artificial Neuron

- Given the activation value $net = \sum_i w_i x_i$, the output of the neuron is given by

$$f(net) = \begin{cases} +1 & \text{if } \sum_i w_i x_i \geq 0 \\ -1 & \text{if } \sum_i w_i x_i < 0 \end{cases}$$



Example

- An artificial neuron can be used to compute the logic AND function.
 - The neuron has three inputs
 - x_1 and x_2 are the original inputs.
 - The third is the bias input which has a constant value of +1.
 - The input data and bias have weights of +1, +1, and -2 respectively.
- What about the logic OR function?

Artificial Neuron

- The perceptron learning algorithm (PLA) can be used to adjust the weights of an artificial neuron.
- The weights are adjusted until the outputs of the neuron become consistent with the true outputs of training examples.
- The following rule is used

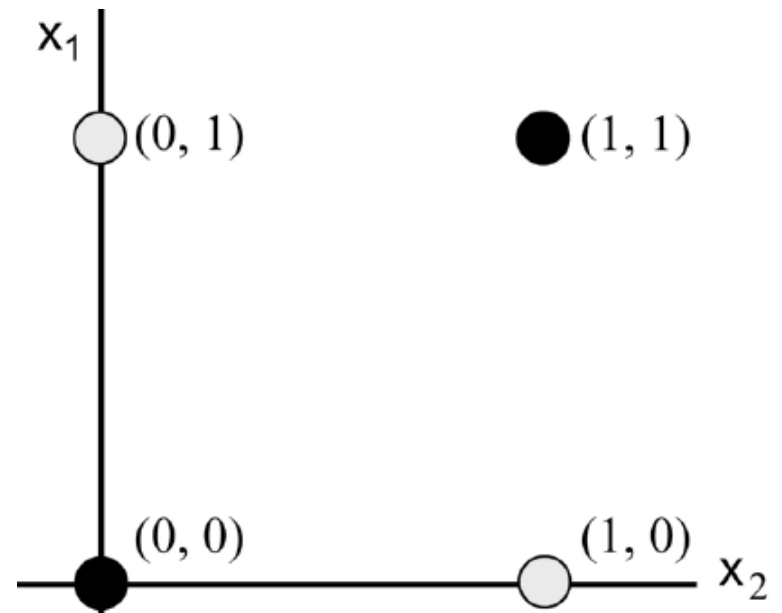
$$\mathbf{w}_{(t+1)} \leftarrow \mathbf{w}_{(t)} + y_{n(t)} \mathbf{x}_{n(t)}$$

Artificial Neuron

- Perceptron learning algorithm can not solve those problems where the patterns are not linearly separable.
- An example of this is the exclusive-OR problem.
- Multilayer networks are required for solving such kinds of problems.

Example

x_1	x_2	Output
1	1	-1
1	0	1
0	1	1
0	0	-1



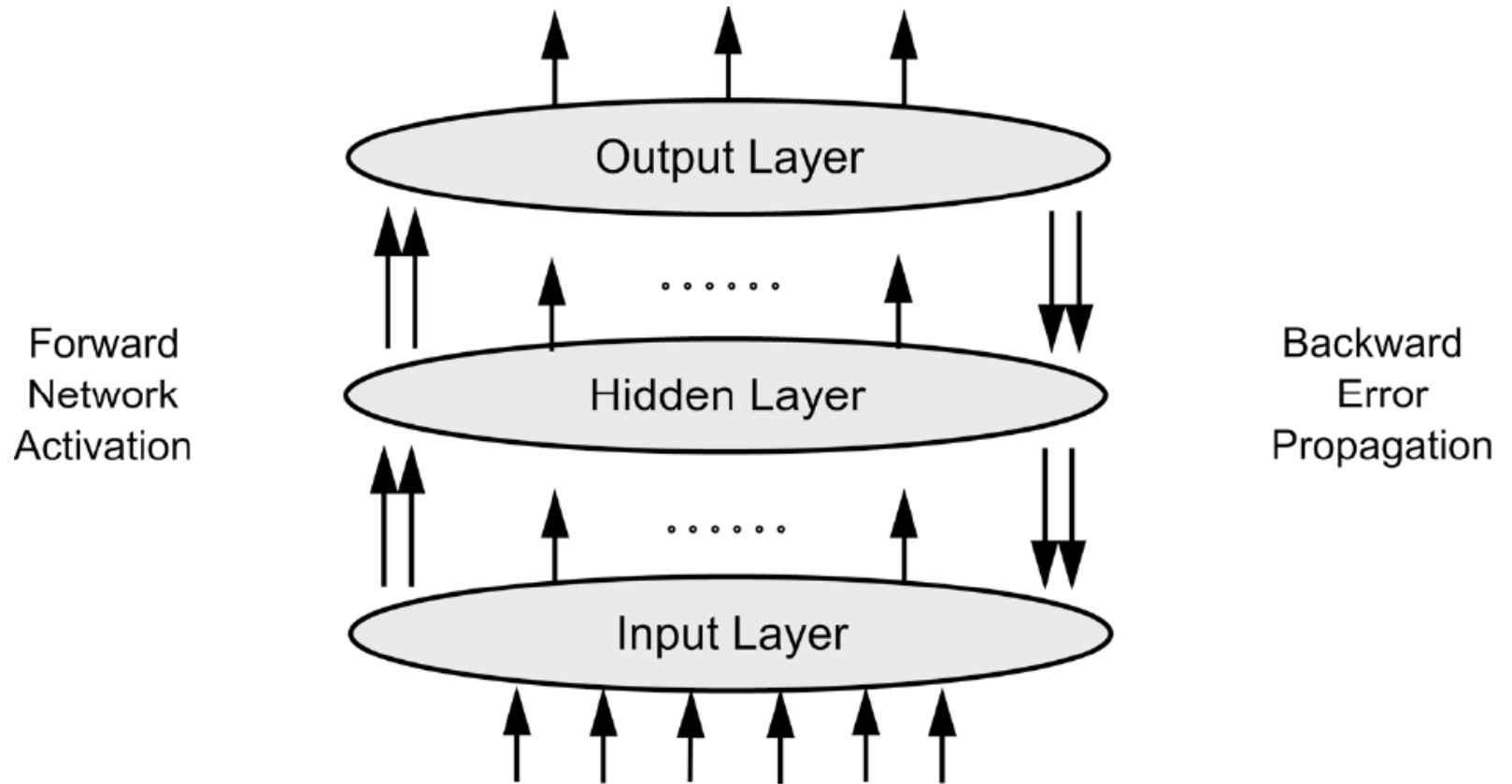
Neural Network

- The additional layers in between the input and output nodes are called hidden layers.
- The nodes embedded in these layers are called hidden nodes.
- We focus on feedforward neural networks, in which the nodes in one layer are connected only to the nodes in the next layer.
- The **backpropagation** learning algorithm is specifically designed for neural networks with multiple layers.

Neural Network

- There are two phases in each iteration of the training algorithm
 - The forward phase
 - The backward phase

Neural Network



Neural Network

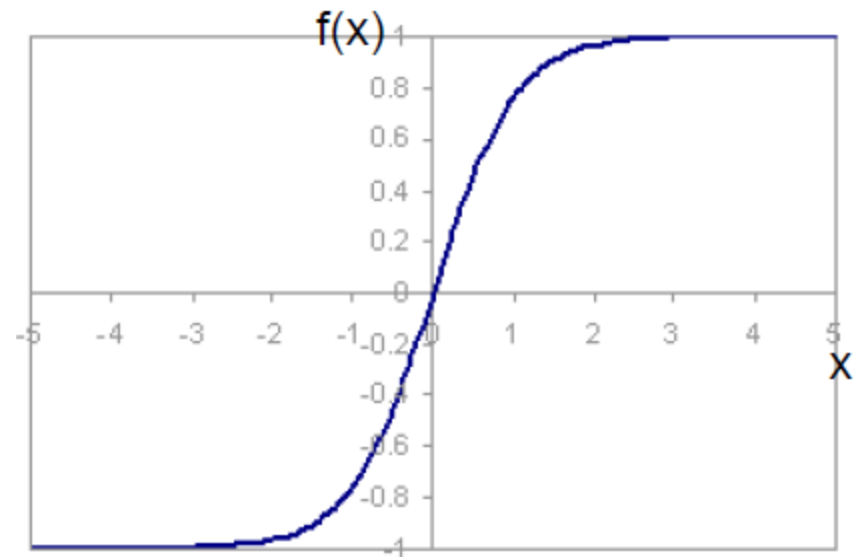
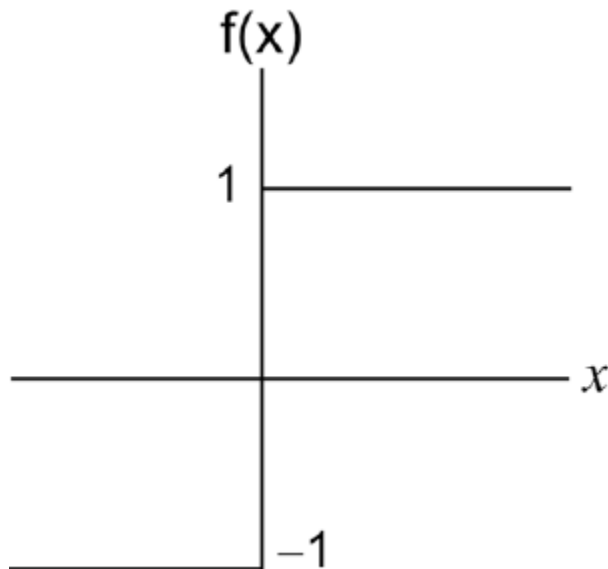
- During the forward phase, the weights obtained from the previous iteration are used to compute the output value of each neuron.
- Outputs of the neurons at level l are computed prior to computing the outputs at level $l+1$.

Neural Network

- During the backward phase, the weight update equation is applied in the reverse direction.
- In other words, the weights at level $l+1$ are updated before the weights at level l are updated.
- The learning algorithm allows us to use the errors for neurons at layer $l+1$ to estimate the errors for neurons at layer l .

Neural Network

- For this type of network, instead of the threshold function, another activation function is used.



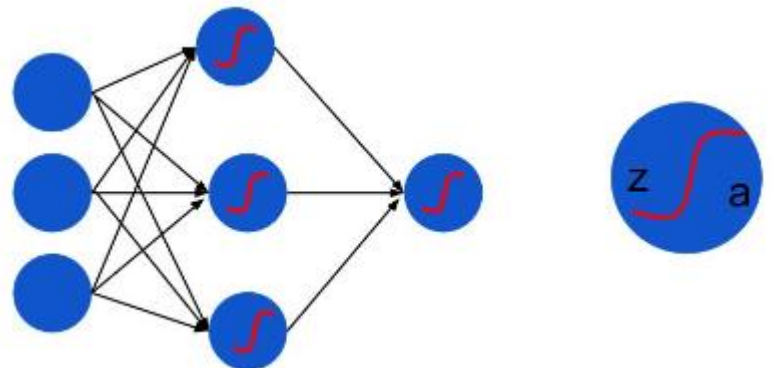
Neural Network

- A common activation function is the hyperbolic tangent function

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{e^{2x} - 1}{e^{2x} + 1}$$

- An important property of the function is that it is differentiable

$$f'(x) = 1 - f(x)^2$$



Neural Network

- Another common activation function is the **sigmoid** function

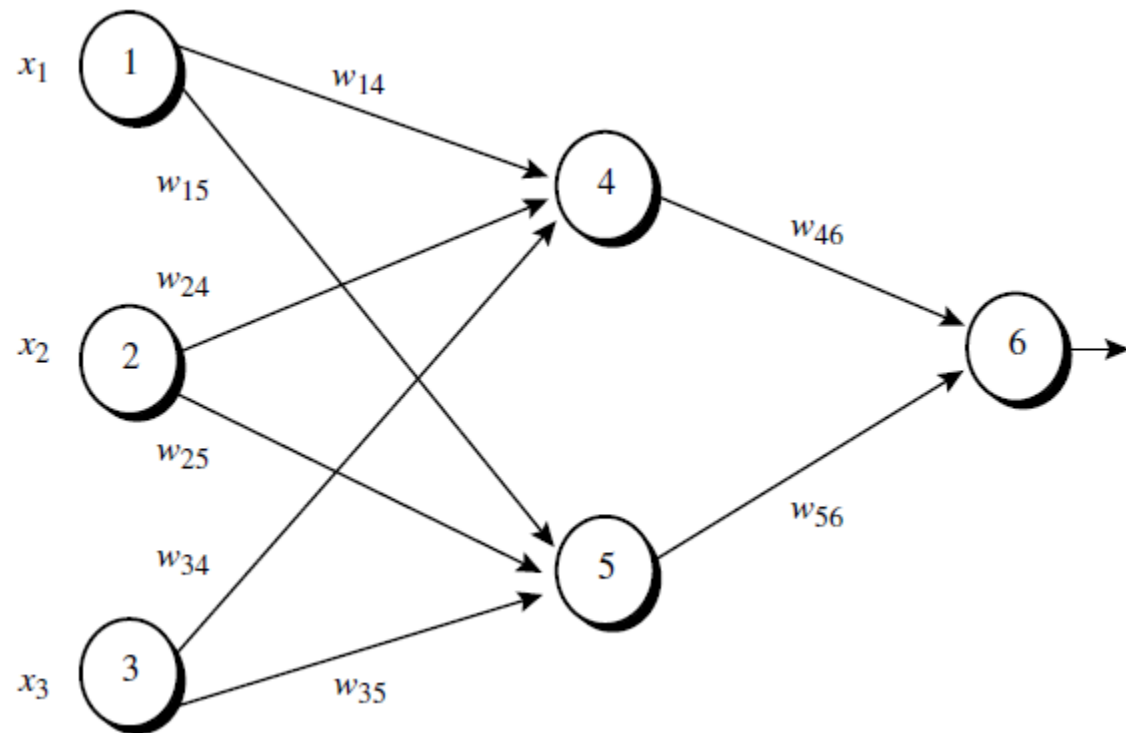
$$f(x) = \frac{1}{1 + e^{-x}}$$

- This function is also differentiable

$$f'(x) = f(x)(1 - f(x))$$

Example

- One training tuple $\mathbf{X}=(1,0,1)$, whose class label is 1.



Example

Initial input, weight, and bias values.

x_1	x_2	x_3	w_{14}	w_{15}	w_{24}	w_{25}	w_{34}	w_{35}	w_{46}	w_{56}	θ_4	θ_5	θ_6
1	0	1	0.2	-0.3	0.4	0.1	-0.5	0.2	-0.3	-0.2	-0.4	0.2	0.1

The net input and output calculations.

Unit j	Net input, I_j	Output, O_j
4	$0.2 + 0 - 0.5 - 0.4 = -0.7$	$1/(1 + e^{0.7}) = 0.332$
5	$-0.3 + 0 + 0.2 + 0.2 = 0.1$	$1/(1 + e^{-0.1}) = 0.525$
6	$(-0.3)(0.332) - (0.2)(0.525) + 0.1 = -0.105$	$1/(1 + e^{0.105}) = 0.474$

Example

Calculation of the error at each node.

<i>Unit j</i>	<i>Err_j</i>
6	$(0.474)(1 - 0.474)(1 - 0.474) = 0.1311$
5	$(0.525)(1 - 0.525)(0.1311)(-0.2) = -0.0065$
4	$(0.332)(1 - 0.332)(0.1311)(-0.3) = -0.0087$

Calculations for weight and bias updating.

<i>Weight or bias</i>	<i>New value</i>
w_{46}	$-0.3 + (0.9)(0.1311)(0.332) = -0.261$
w_{56}	$-0.2 + (0.9)(0.1311)(0.525) = -0.138$
w_{14}	$0.2 + (0.9)(-0.0087)(1) = 0.192$
w_{15}	$-0.3 + (0.9)(-0.0065)(1) = -0.306$
w_{24}	$0.4 + (0.9)(-0.0087)(0) = 0.4$
w_{25}	$0.1 + (0.9)(-0.0065)(0) = 0.1$
w_{34}	$-0.5 + (0.9)(-0.0087)(1) = -0.508$
w_{35}	$0.2 + (0.9)(-0.0065)(1) = 0.194$
θ_6	$0.1 + (0.9)(0.1311) = 0.218$
θ_5	$0.2 + (0.9)(-0.0065) = 0.194$
θ_4	$-0.4 + (0.9)(-0.0087) = -0.408$

Neural Network

- Given a unit j in a hidden or output layer, the net input, I_j , to unit j is $I_j = \sum_i w_{ij} O_i + \theta_j$

where w_{ij} is the weight of the connection from unit i in the previous layer to unit j ; O_i is the output of unit i from the previous layer; and θ_j is the bias of the unit.

- Given the net input I_j to unit j , then O_j , the output of unit j , is computed as $O_j = \frac{1}{1 + e^{-I_j}}$

Propagate the inputs forward

Backpropagate the error

- For a unit k in the output layer, the error Err_k is computed by

$$Err_k = O_k (1 - O_k) (T_k - O_k)$$

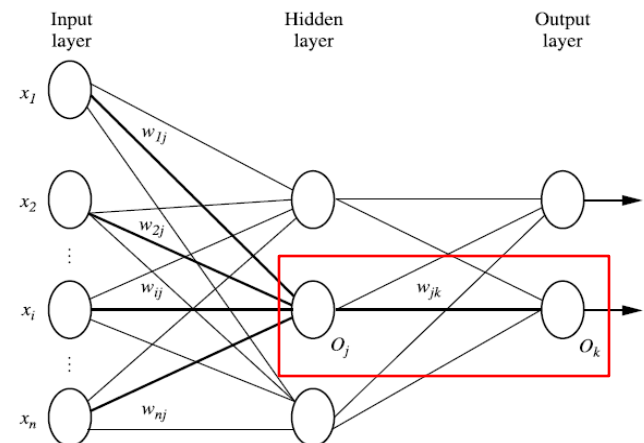
- The error of a hidden layer unit j is

$$Err_j = O_j (1 - O_j) \sum_k Err_k w_{jk}$$

- Weights are updated by

$$w_{jk} = w_{jk} + \eta Err_k O_j$$

$$\theta_k = \theta_k + \eta Err_k$$

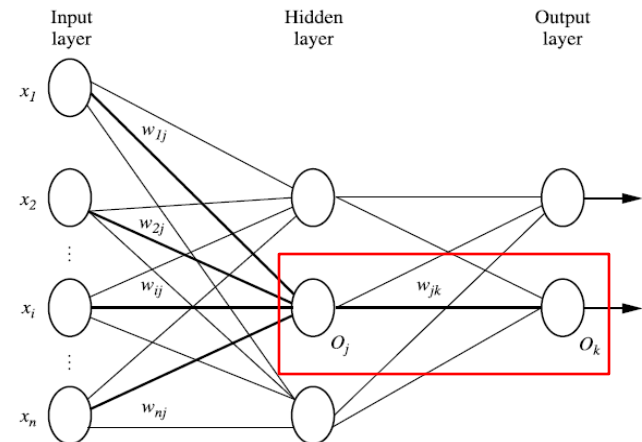


Neural Network

- Minimize the error of node O_k
- We define it as $E = \frac{1}{2}e^2 = \frac{1}{2}(T - O)^2$
- To adjust weight w_{jk} , we first calculate the partial derivation of E on w_{jk}

$$\begin{aligned}\frac{\partial E}{\partial w_{jk}} &= \frac{\partial E}{\partial e} \times \frac{\partial e}{\partial O_k} \times \frac{\partial O_k}{\partial w_{jk}} \\ &= -(e) \times (O_k(1 - O_k)) \times (O_j) \\ &= -(T_k - O_k)O_k(1 - O_k)O_j\end{aligned}$$

- and then use the “gradient decent”



Neural Network

- Backpropagation learning is based on the idea of an error surface.
- The surface represents cumulative error over a data set as a function of network weights.
- Each possible network weight configuration is represented by a point on the surface.

Neural Network

- The goal of the learning algorithm is to determine a set of weights that minimize the error.
- The learning algorithm should be designed to find the direction on the surface which most rapidly reduces the error.
- This can be achieved by moving in the opposite direction of the gradient vector at each surface point (i.e., by employing the gradient descent learning method).

Neural Network

- **Weakness**

- Long training time
- Require a number of parameters typically best determined empirically, e.g., the network topology or “structure”.
- Poor interpretability
 - Difficult to interpret the symbolic meaning behind the learned weights and of “hidden units” in the network

Neural Network

- **Strength**

- High tolerance to noisy data
- Well-suited for continuous-valued inputs and outputs
- Successful on a wide array of real-world data
- Techniques have recently been developed for the extraction of rules from trained neural networks

Neural Network

- **Rule extraction from networks:** network pruning
 - Simplify the network structure by removing weighted links that have the least effect on the trained network
 - The set of input and activation values are studied to derive rules describing the relationship between the input and hidden unit layers
- **Sensitivity analysis:** assess the impact that a given input variable has on a network output. The knowledge gained from this analysis can be represented in rules