COM 424 E: NEURAL NETWORKS

Lecture 02:Learning Techniques

E.M

Introduction

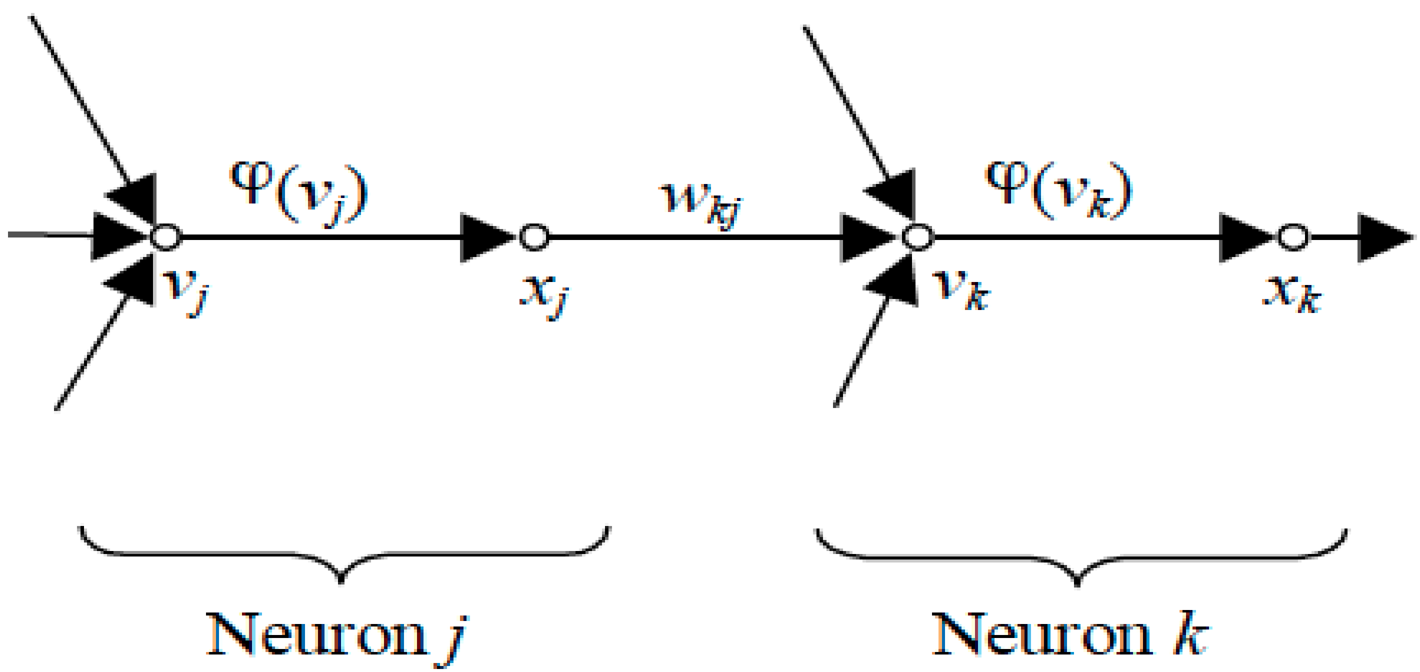
- One of the most important ANN features is ability to learn from the environment
- ANN learns through an iterative process of synaptic weights and threshold adaptation
- After each iteration ANN should have more knowledge about the environment

Definition of learning

- Definition of learning in the ANN context:
- Learning is a process where unknown ANN parameters are adapted through continuous process of stimulation from the environment
- Learning is determined by the way how the change of parameters takes place
- This definition implies the following events:
- The environment stimulates the ANN
- ANN changes due to environment
- ANN responds differently to the environment due to the change

Notation

- v_j and v_k are activations of neurons j and k
- x_j and x_k are outputs of neurons j and k
- Let $w_{ki}(n)$ be synaptic weights at time n



- If in step n synaptic weight $w_{kj}(n)$ is changed by $\Delta w_{kj}(n)$ we get the new weight:

$$w_{kj}(n+1) = w_{kj}(n) + \Delta w_{kj}(n)$$

  where $w_{kj}(n)$ and $w_{kj}(n+1)$ are old and new weights between neurons k and j

- A set of rules that are solution to the learning problem is called a **learning algorithm**
- There is no unique learning algorithm, but many different learning algorithms, each with its advantages and drawbacks

Algorithms and learning paradigms

- Learning algorithms determine how weight correction $\Delta w_{kj}(n)$ is computed
- Learning paradigms determine the relation of the ANN to the environment
- Three basic learning paradigms are:
  - Supervised learning
  - Reinforcement learning
  - Unsupervised learning

Basic learning approaches

- According to learning algorithm:
  - Error-correction learning
  - Hebb learning
  - Competitive learning
- According to learning paradigm:
  - Supervised learning
  - Reinforcement learning
  - Unsupervised learning

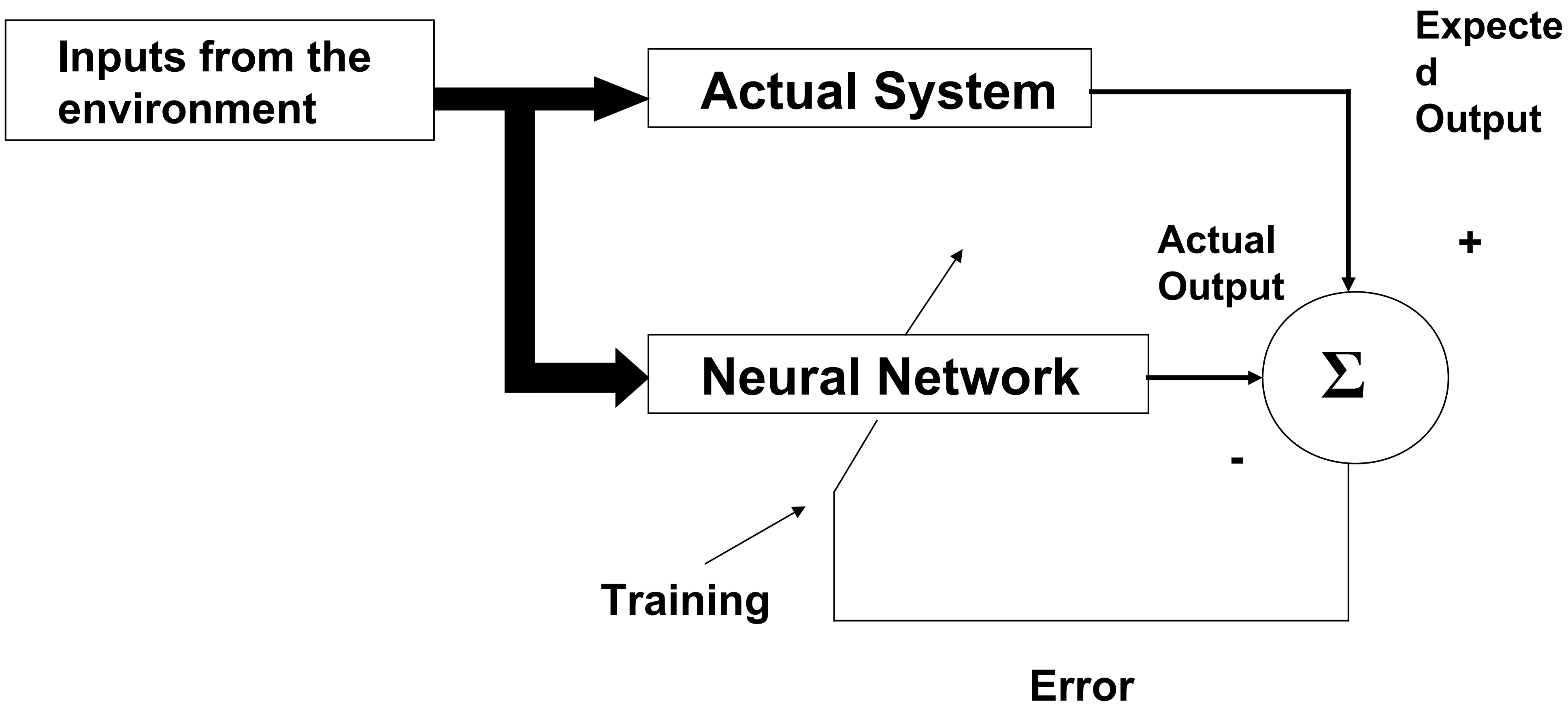
Supervised Learning

- The network is provided with a correct answer (output) for every input pattern
- During learning, produced output is compared with the desired output
- The difference between both output is used to modify learning weights according to the learning algorithm
- Recognizing hand-written digits, pattern recognition and etc.
- Neural Network models: perceptron, feed-forward, back-propagation a
-

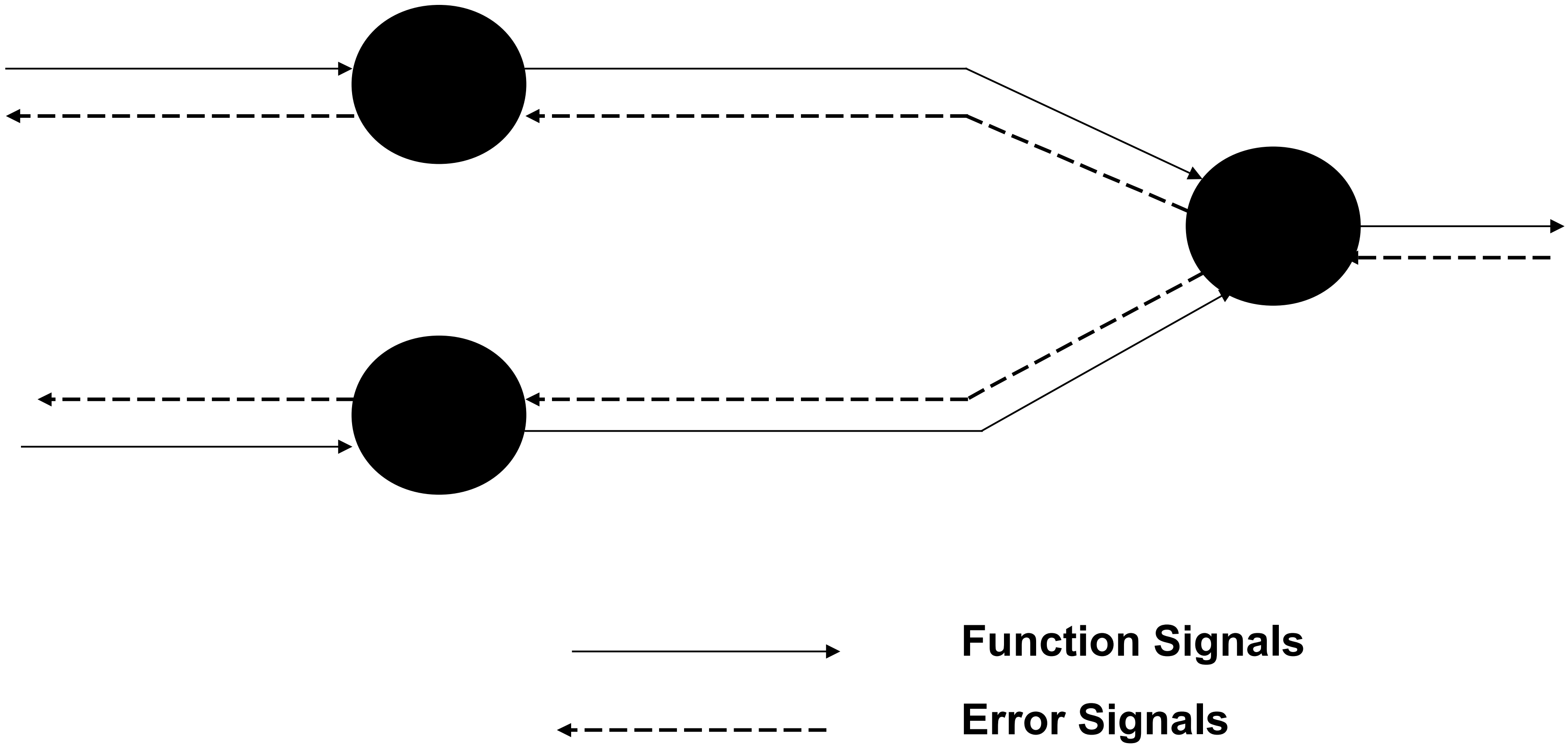
8

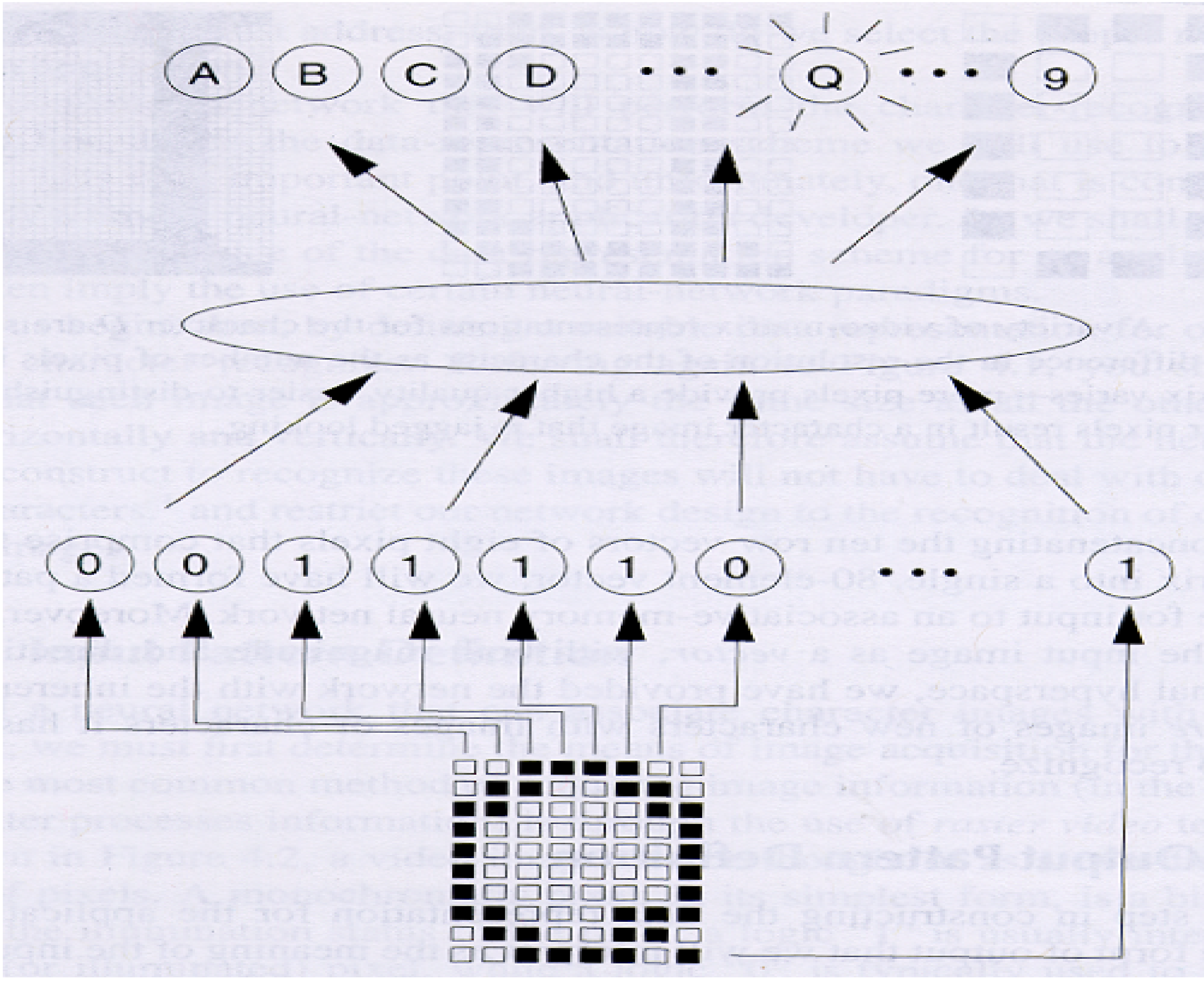
Learning Techniques

- Supervised Learning:



Signal Flow
Backpropagation of Errors




A
B
C
D
Q
9
0 0 1 1 1 1 0 1

Unsupervised Learning

- Targets are not provided
- Does not require a correct answer associated with each input pattern in the training set
- Explores the underlying structure in the data, or correlations between patterns in the data, and organizes patterns into categories from these correlations
- Appropriate for clustering task
- Find similar groups of documents in the web, content addressable memory, clustering.
- Neural Network models: Kohonen, self organizing maps, Hopfield networks.

Reinforcement Learning

- Target is provided, but the desired output is absent.
- The net is only provided with guidance to determine the produced output is correct or vise versa.
- Weights are modified in the units that have errors

13

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.

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.

Applications

- Used in complex function approximations, feature extraction & classification, and optimization & control problems
- Applicability in all areas of science and technology.

Hebbian Learning

- In 1949, Donald Hebb proposed one of the key ideas in biological learning, commonly known as Hebb's Law.
- Hebb's Law states that if neuron i is near enough to excite neuron j and repeatedly participates in its activation, the synaptic connection between these two neurons is strengthened and neuron j becomes more sensitive to stimuli from neuron i.
- Hebb's Law can be represented in the form of two rules:
  - If two neurons on either side of a connection are activated synchronously, then the weight of that connection is increased.
  - If two neurons on either side of a connection are activated asynchronously, then the weight of that connection is decreased.

- Hebbian learning implies that weights can only increase. To resolve this problem, we might impose a limit on the growth of synaptic weights.
- It can be implemented by introducing a non-linear forgetting factor into Hebb's Law
- Forgetting factor usually falls in the interval between 0 and 1, typically between 0.01 and 0.1, to allow only a little "forgetting" while limiting the weight growth.

Hebbian Learning Algorithm

- Step 1: Initialisation
  - Set initial synaptic weights and threshold to small random values, say in an interval [0,1].
- Step 2: Activation
  - Compute the neuron output at iteration p
  - $$y_j(p) = \sum_{i=1}^{n} x_i(p)\, w_{ij}(p) - \theta_j$$
  - where n is the number of neuron inputs, and θj is the threshold value of neuron j.
- Step 3: Learning
  - Update the weights in the network:

$$w_{ij}(p+1) = w_{ij}(p) + \Delta w_{ij}(p)$$

  - where Äwij(p) is the weight correction at iteration p.

- The weight correction is determine by the generalised activity product rule:

$$\Delta w_{ij}(p) = \varphi \, y_j(p)\left[\lambda \; x_i(p) - w_{ij}(p)\right]$$

- Step 4: Iteration
  - Increase iteration p by one, go back to Step 2.

Competitive Learning

- Neurons compete among themselves to be activated
- While in Hebbian Learning, several output neurons can be activated simultaneously, in competitive learning, only a single output neuron is active at any time.
- The output neuron that wins the “competition” is called the winner-takes-all neuron.

Error-correction learning

- Belongs to the supervised learning paradigm
- Let $d_k(n)$ be desired output of neuron k at time n
- Let $y_k(n)$ be obtained output of neuron k at time n
- Output $y_k(n)$ is obtained using input vector $x(n)$
- Input vector $x(n)$ and desired output $d_k(n)$ represent an example that is presented to ANN at moment n
- Error is the difference between desired and obtained output of neuron k at moment n:
    - $e_k(n) = d_k(n) - y_k(n)$