Basic Principles of Artificial Neural Networks

Artificial Neural Networks

These are information processing systems, whose structure and functionality are inspired from our present-day knowledge about biological neural systems.

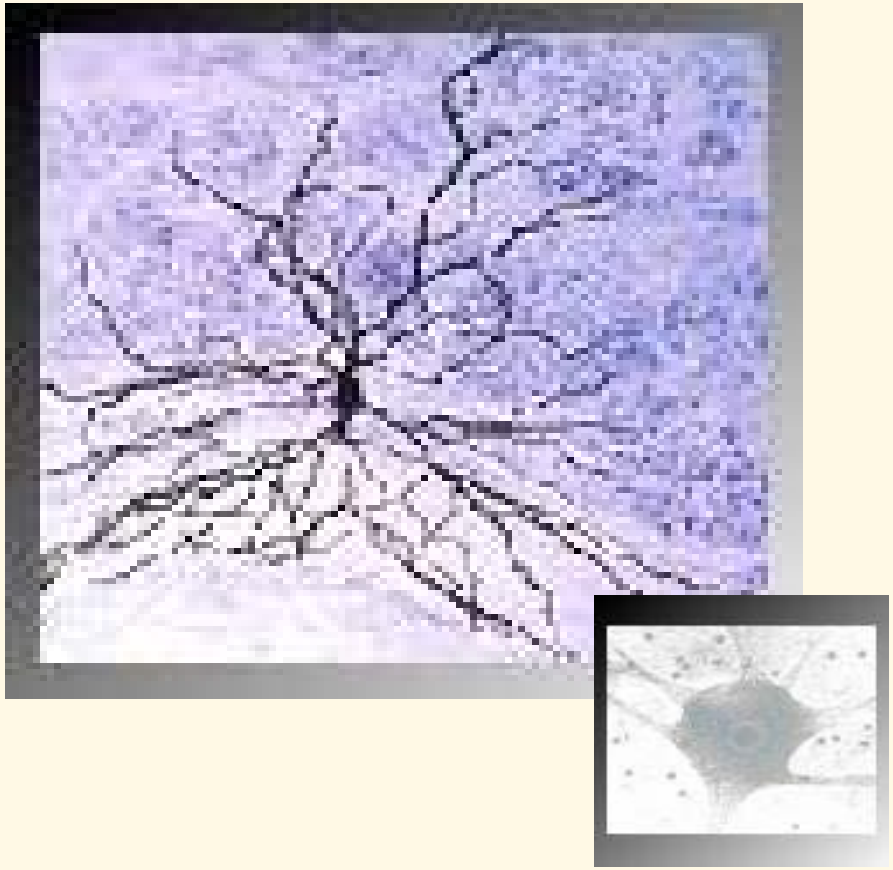
Biological Neural Networks

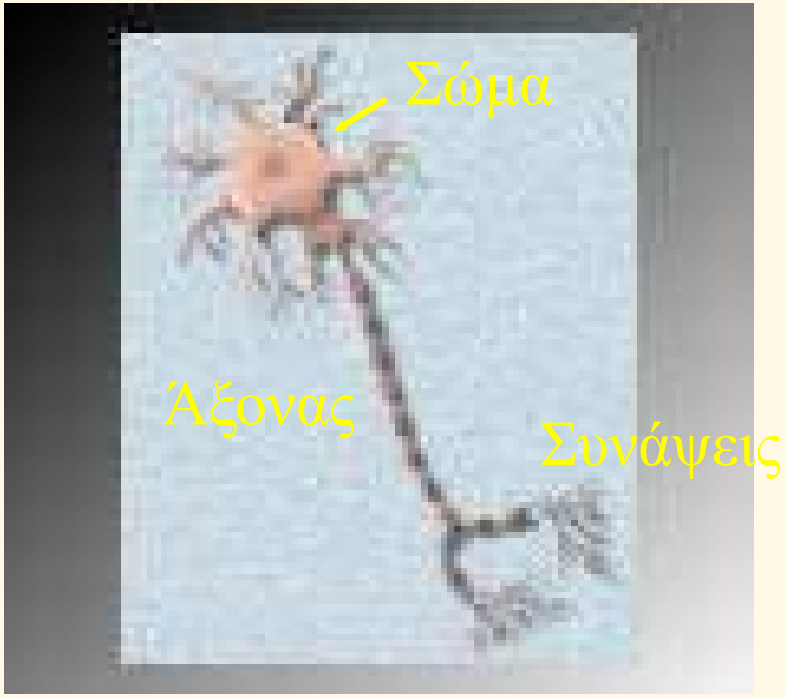
- Composed of cells called neurons
- 10^{11} neurons in the human
- Each neuron is connected with other neurons via a few thousand connections called synapses
- 10^{15} synapses in the human brain
- Exchange of signals between neurons (which is equivalent to the exchange of information) is of chemical nature

Biological Neural Networks

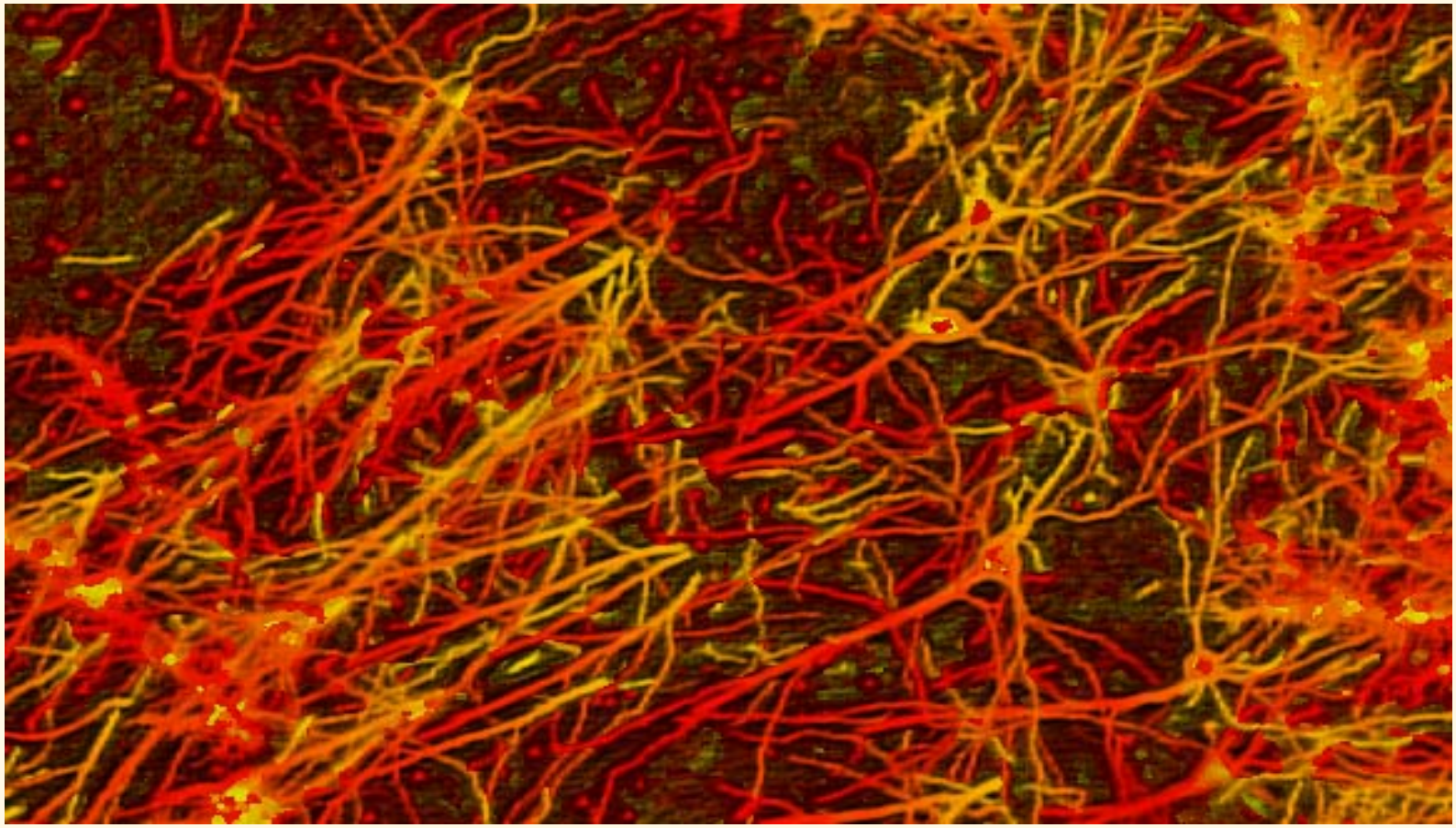
- Chemical substances (neurotransmitters) are released on the transmitting side of the synapse
- This contributes to the increase or decrease of the electrical potential in the neuron that receives the information
- If the potential is increased beyond a certain limit, the neuron is activated:
- A pulse of limited duration in produced, which, in its own turn, is transmitted through the synapses to other neurons

Neuron

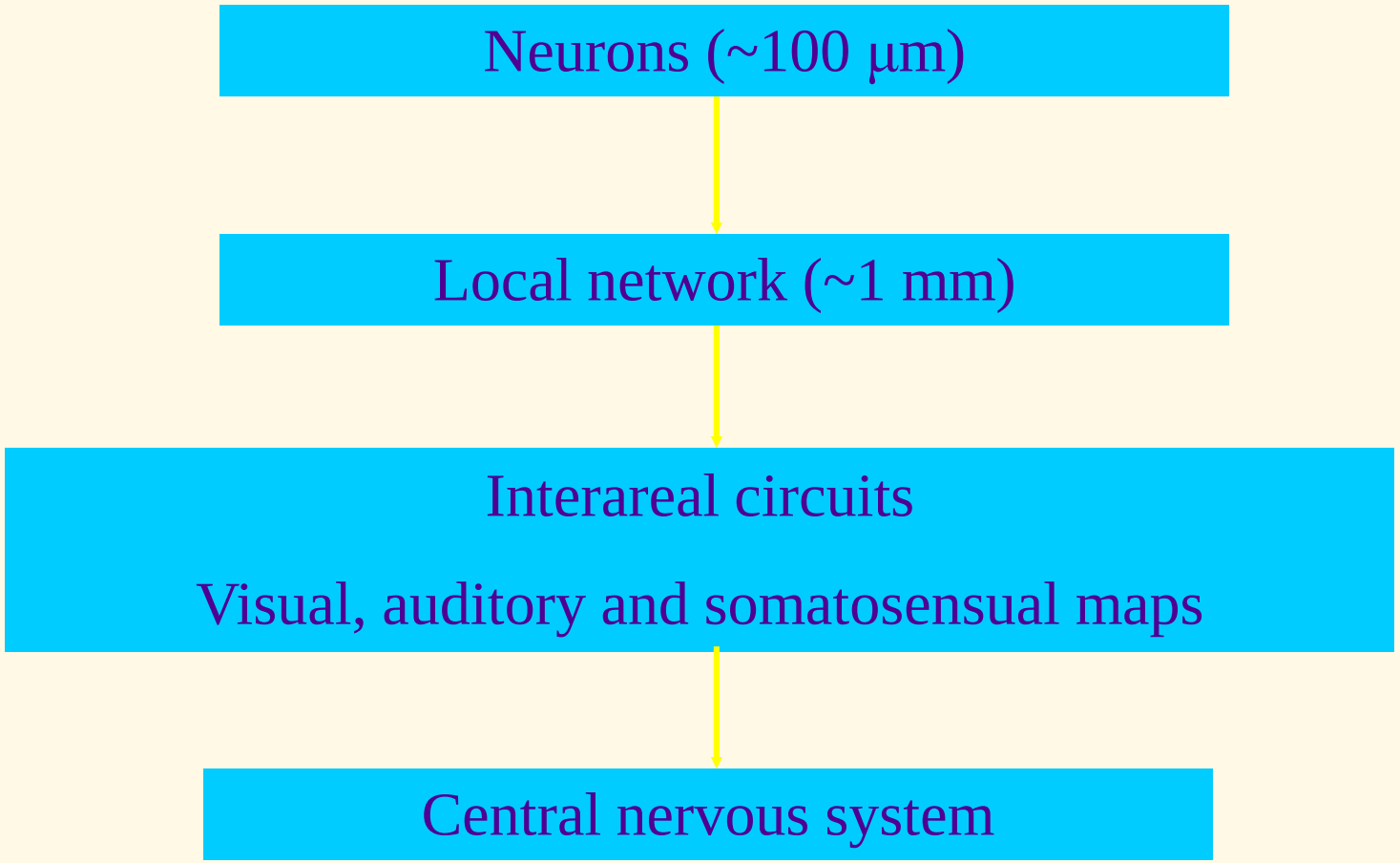




Biological Neural Network

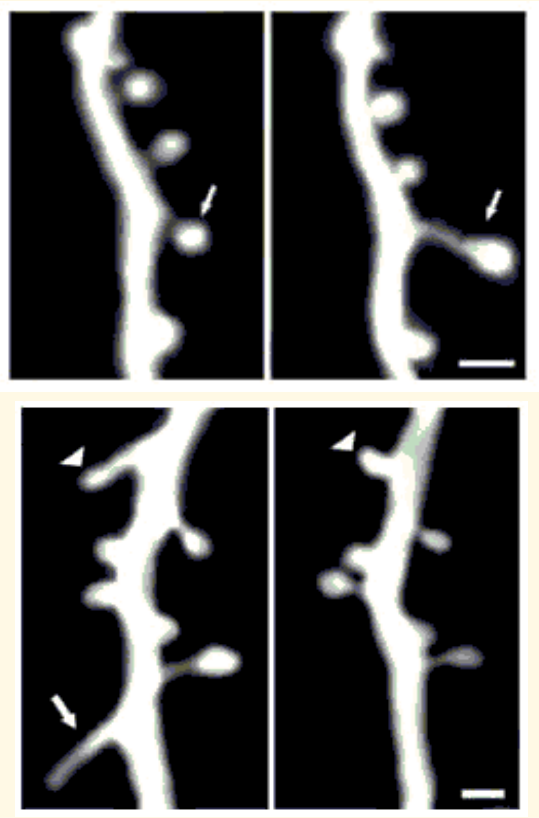


Complexity of structure



Important features of biological neural networks

- Plasticity: During learning periods, synapse strength (i.e. The capacity of synapses to transmit signals) is modified. Moreover, synapses can be created or destroyed.
- Large number of neurons + parallel processing = High processing speed

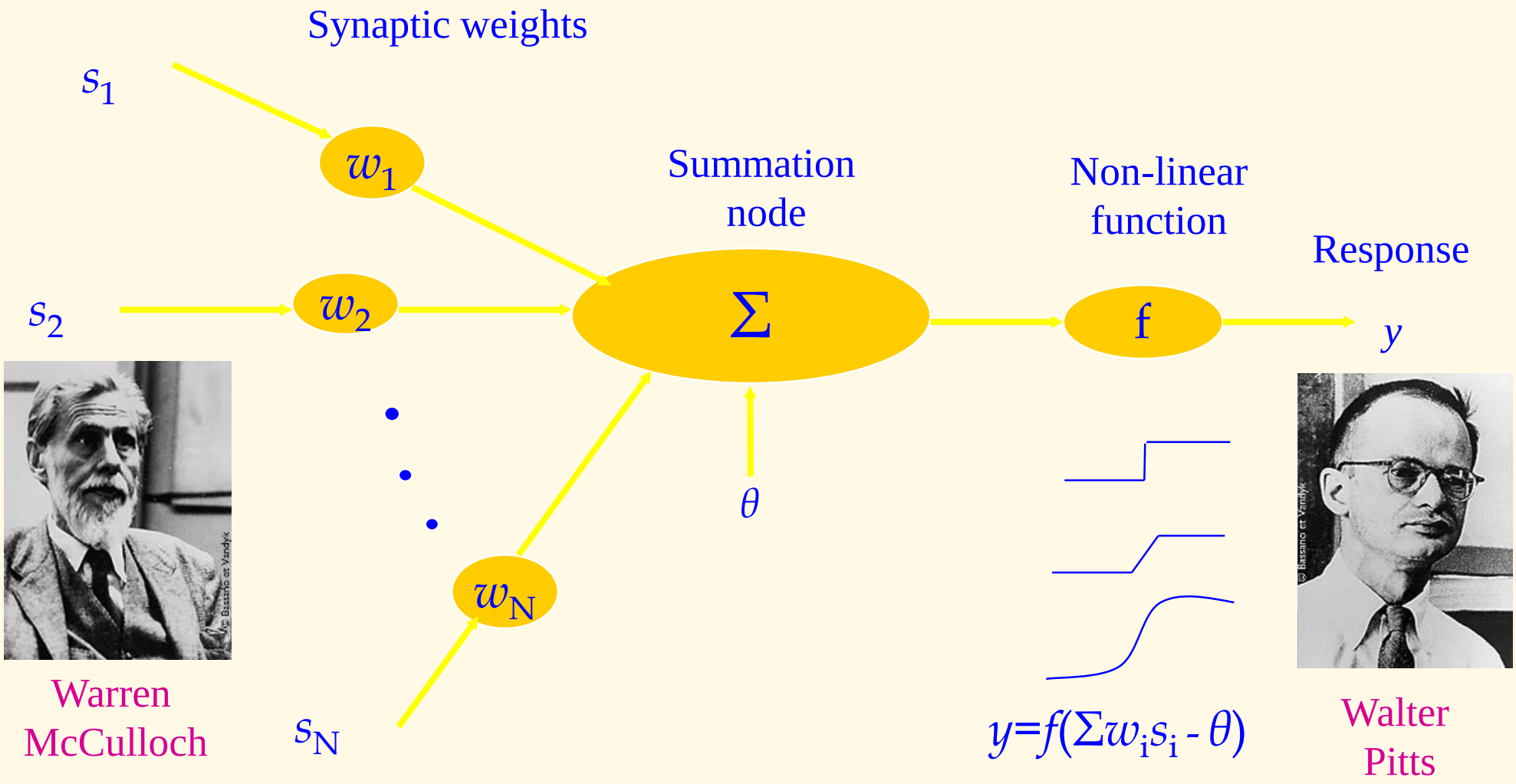


Growing synapse images:
Credit: Wen-Biao Gan,
University of New York

Brain-Computer Differences

Brain	**Conventional Computer**
Many simple structural elements (neurons)	Few complex processors
Few processing steps	Many processing steps
Learning through experience – generalization ability	Meticulous programming
High connectivity – Distributed information storage	Local information storage
Robustness concerning partial destruction	Failure in case of partial destruction

ANN: "McCulloch-Pitts" processing element (typical artificial „neuron")



Warren McCulloch

Walter Pitts

Important features of artificial neural networks

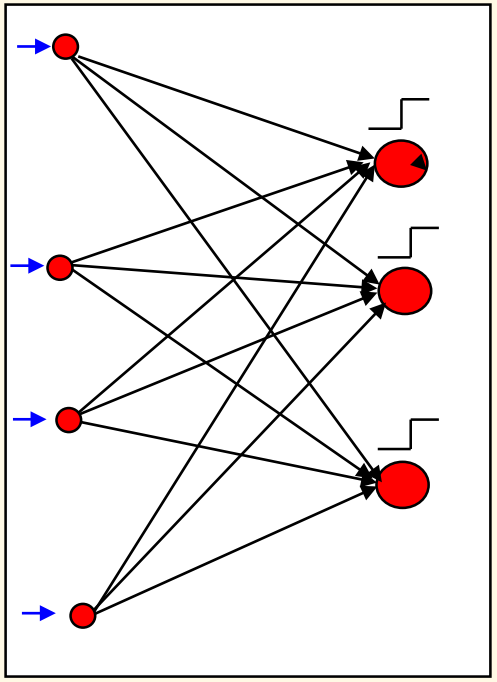
- **Distributed processing:** Information is stored collectively in a large area of the network, rather than locally in each processing unit.
- **Robust processing:** Removal or distortion of a small number of synaptic weights does not result in a substantial deterioration of the network's ability for reliable information processing
- **High degree of parallelism:** Performing operations in one neuron is independent of performing opetations in many other neurons. Operations in many neurons can be performed in parallel

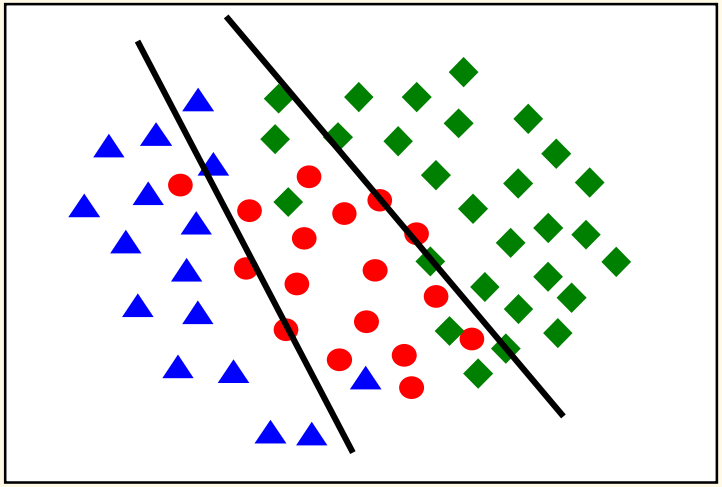
Important features of artificial neural networks

- **Non-linearity:** ANNs consist of a large number of non-linear processors. Some types of ANNs can approximate highly complex non-linear functions and solve difficult non-linear classification and regression problems
- **Learning through examples:** ANNs learn using examples. Learning is associated with the algorithmic updating of synaptic weights. No hard programming (use of specific problem-oriented programming rules) is involved
- **Generalization ability:** Correct response to examples that have not been shown to the network during training

Single-layered perceptron

- **Architecture:** Single layer of McCulloch-Pitts neurons. Neurons are connected with input nodes, where training patterns are presented
- ***Training mode:*** Supervised
- ***Use:*** Classification (linear boundaries between classes)





Multi-layered perceptron

- **Architecture:** Many layers of McCulloch-Pitts neurons.

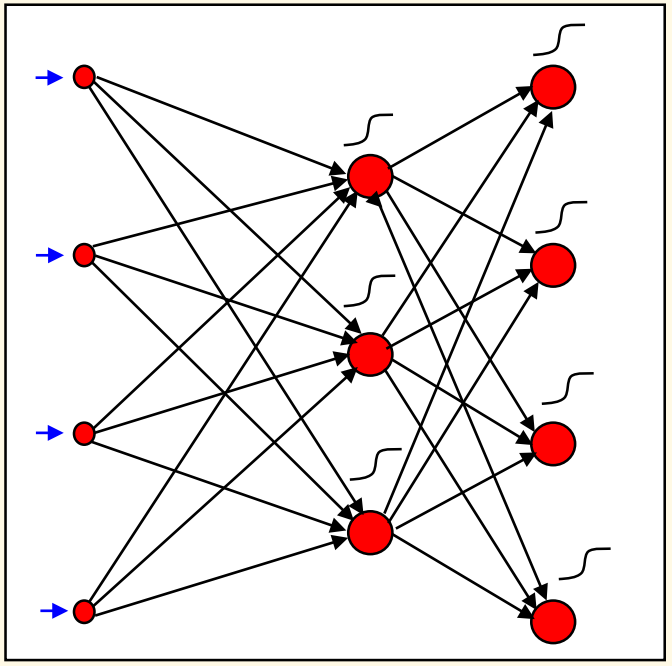
- ***Training mode:*** Supervised

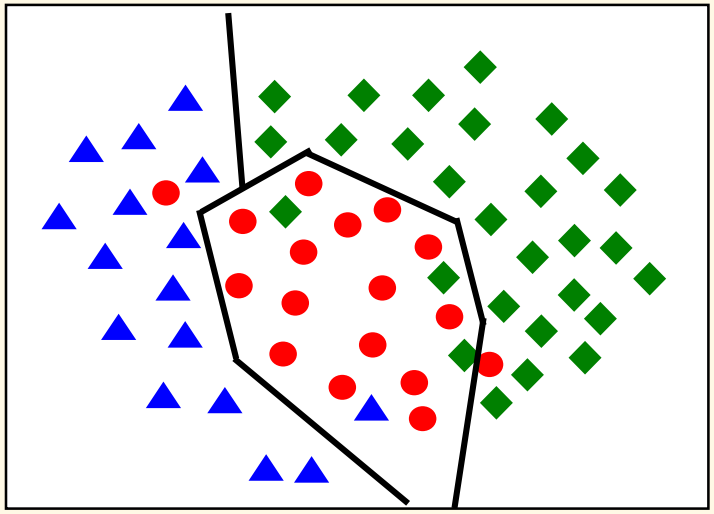
- ***Use:***

➢ Classification (non-linear boundaries between classes)

➢ Non-linear regression

➢ Non-linear system modeling and prediction





Radial basis function network

- **Architecture:** Two layers of neurons. The hidden layer uses activation functions with a local range (e.g. Gaussians)

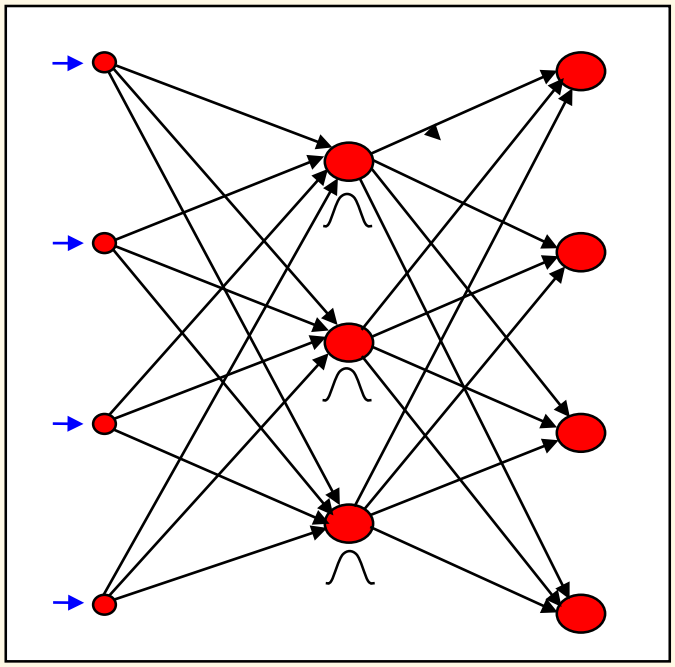
- ***Training mode:*** Supervised

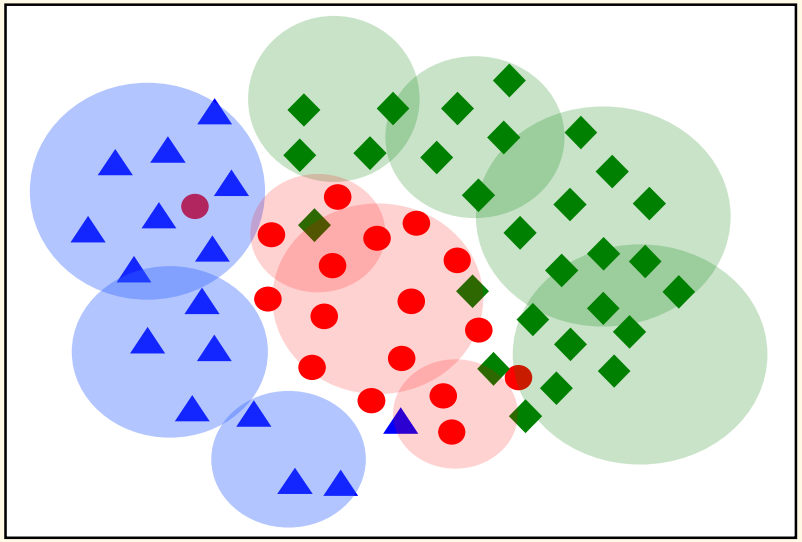
- ***Use:***

➢ Classification (non-linear boundaries between classes)

➢ Non-linear regression

➢ Non-linear system modeling and prediction



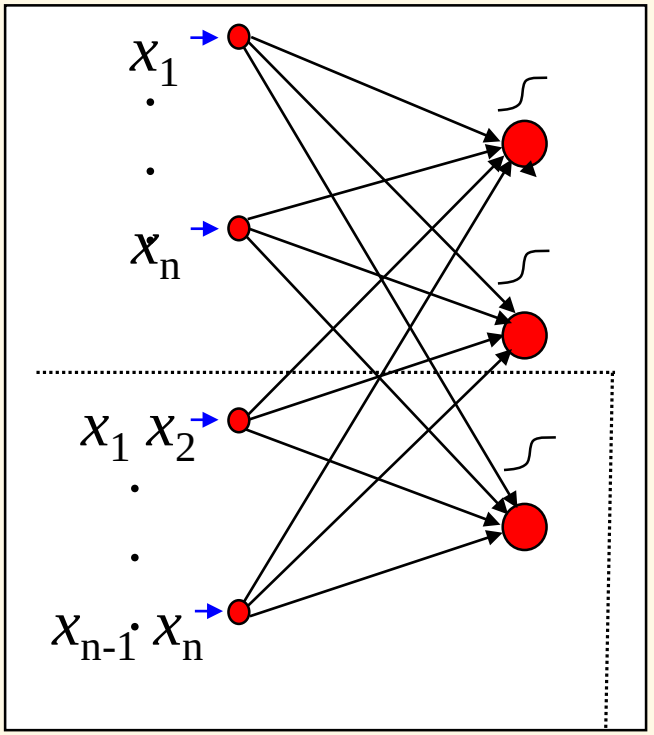


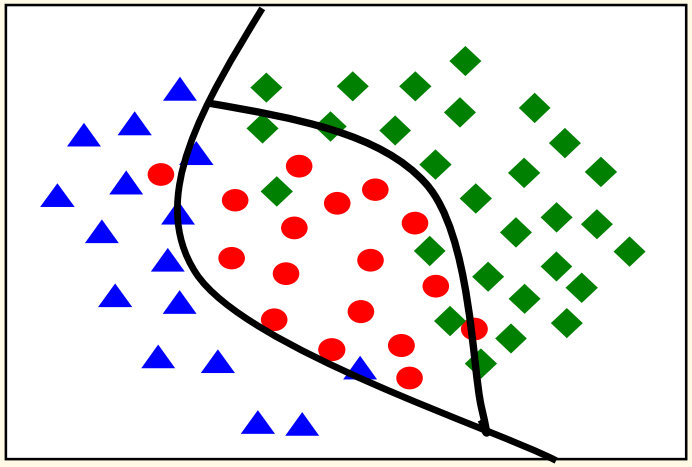
Functional link network

- **Architecture:**
  - Usually single-layered with McCulloch-Pitts neurons. A multilayered version is also possible
  - Extension of the input space using non-linear combinations (usually products) of the initial inputs.
  - More complex serarating functions than those of the single layered perceptron are achieved (e.g. conic sections for a second order network).

- ***Training mode:*** Supervised

- ***Use:***
  - Classification (non-linear boundaries between classes)



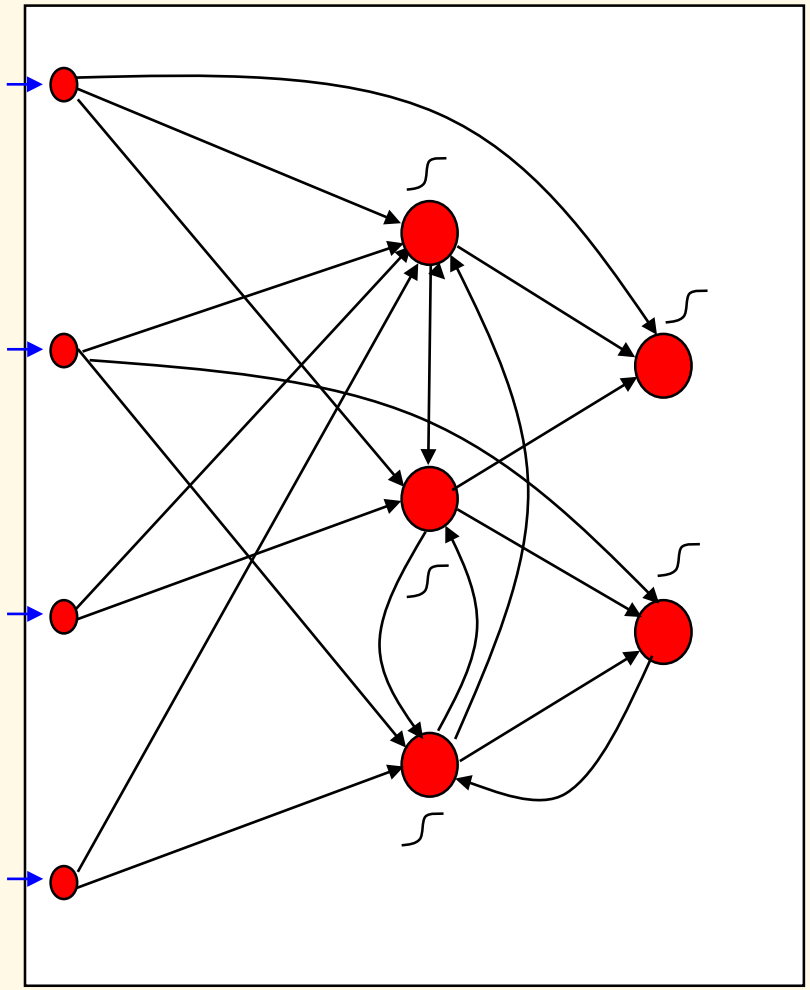


Recurrent network

- **Architecture:**
  - McCulloch-Pitts neurons loosely belonging to layers.
  - Arbitrary synapses between neurons, some directed to previous layers closer to the input. Synapses from output to input can also be present.

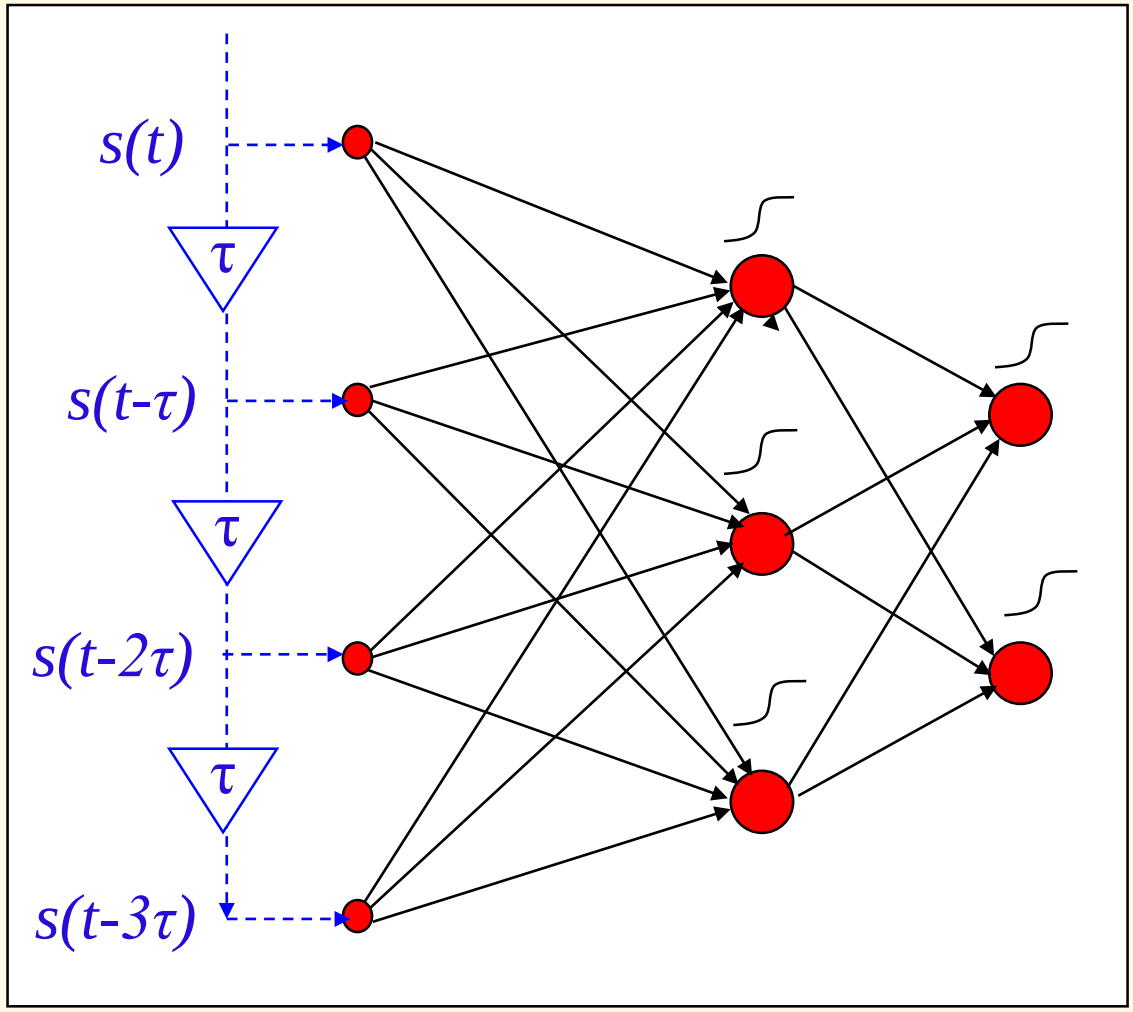
- ***Training mode:*** Supervised

- ***Use:***
  - Classification (non-linear boundaries between classes)
  - Non-linear regression
  - System modeling, prediction
  - Very effective in time-dependent applications



Time delay network

- **Architecture:**
  - Multiple layers. Incorporates multiple input instances for different points in time
- ***Training mode:*** Supervised
- ***Use:***
  - Time-series modeling and prediction

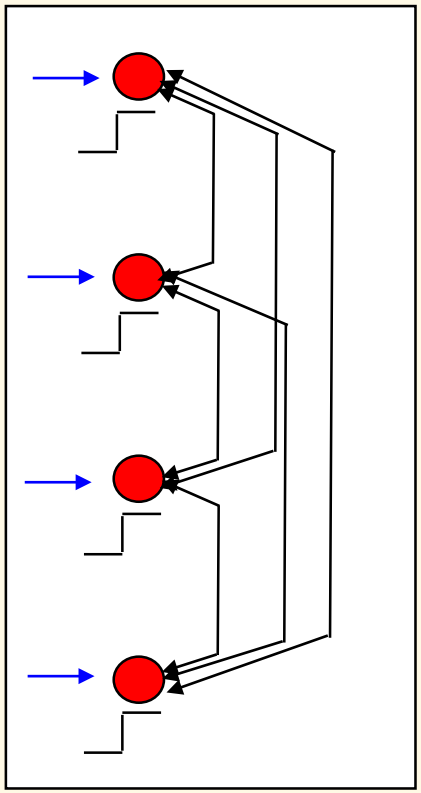


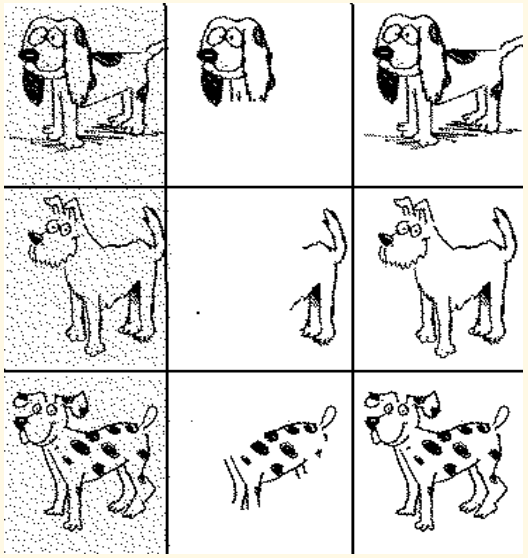
Hopfield network

- **Architecture:**
  - Single layered network with fully connected McCulloch-Pitts neurons
  - Output-input feedback during testing

- ***Training mode:*** Unsupervised

- ***Use:***
  - Distributed pattern storage and retrieval
  - Solution of combinatorial optimization problems (e.g. traveling salesman problem)





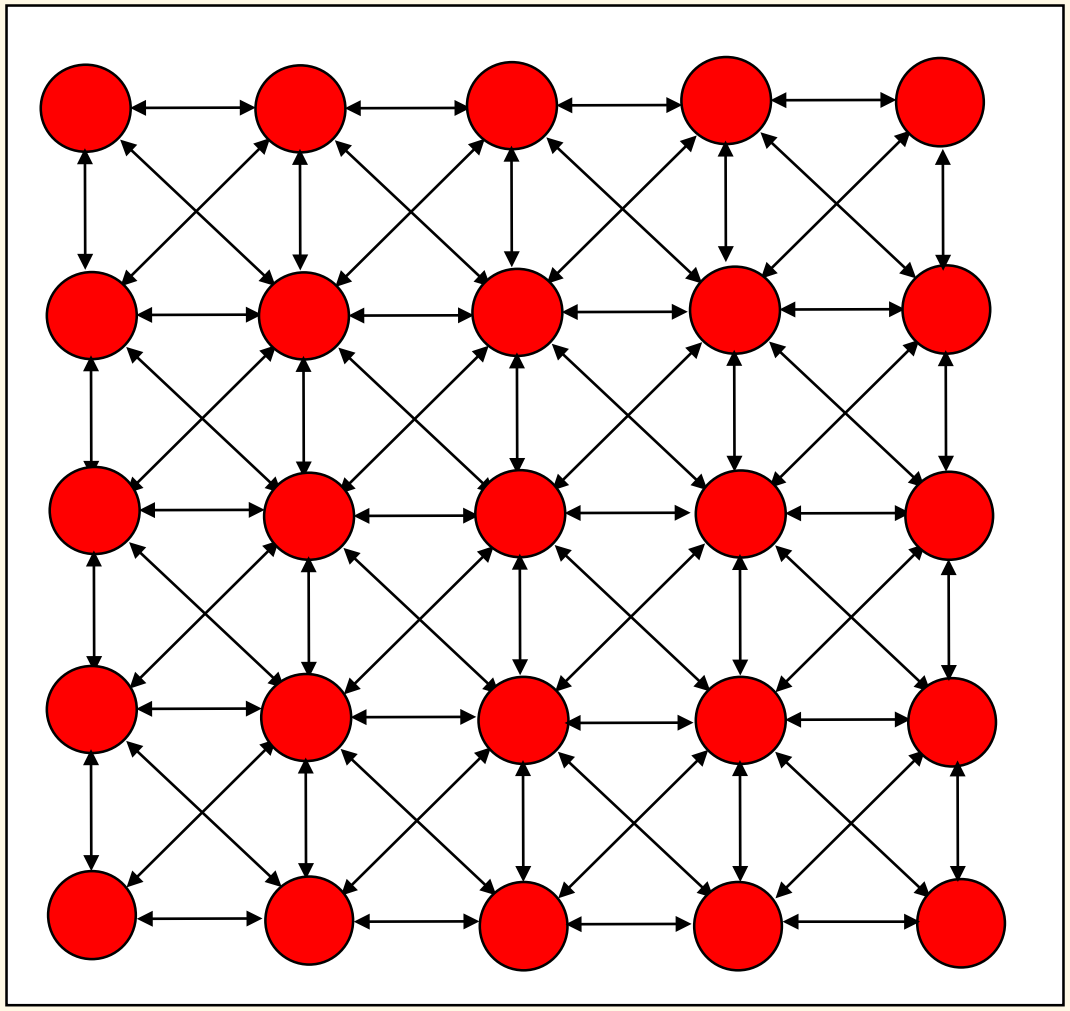
Cellular neural network

- **Architecture:**
  - Single layer of neurons
  - Topologocal organization of the neurons in a 2 dimensional grid with local synaptic connections

- ***Training mode:*** Supervised

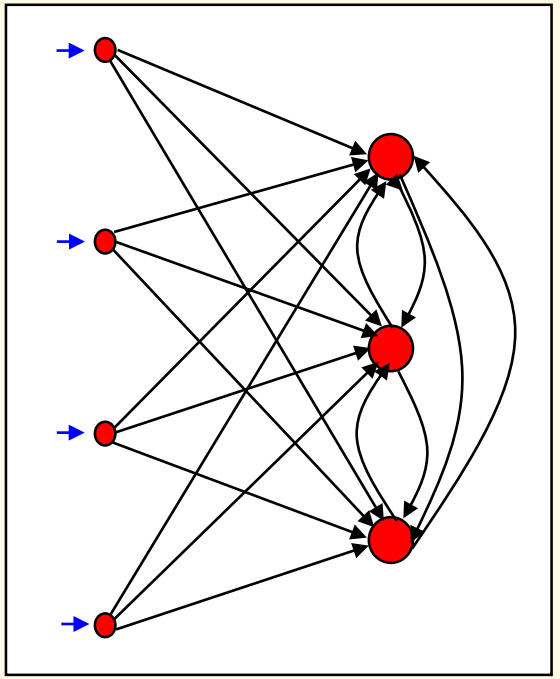
- ***Use:***

  On-line processing of digital images:
  - 2-dimensional filter design and implementation
  - Feature extraction



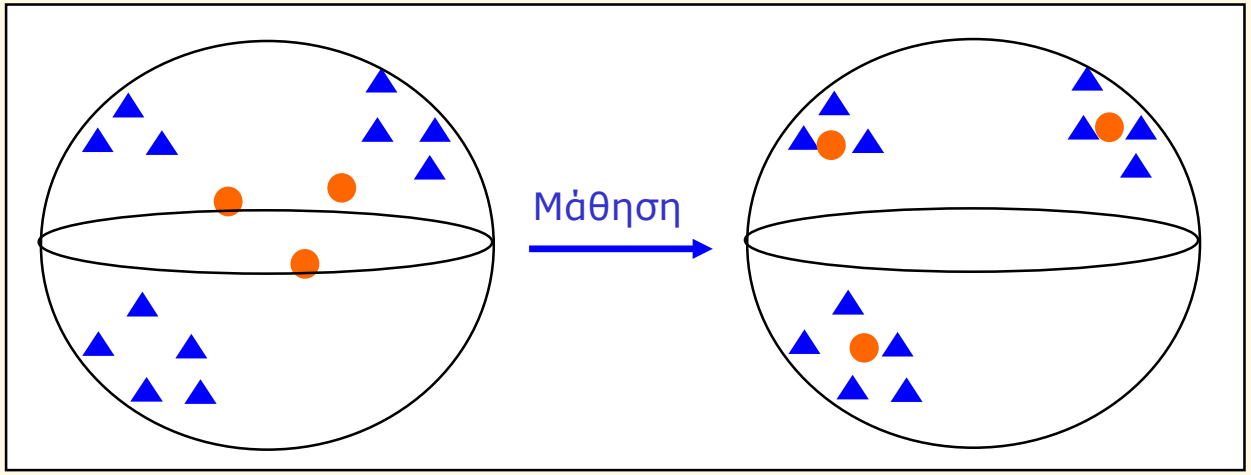
Competitive learning network

- **Architecture:**
  - Single layer of neurons
  - Additional lateral synaptic connections - lateral inhibition
  - Linear neurons



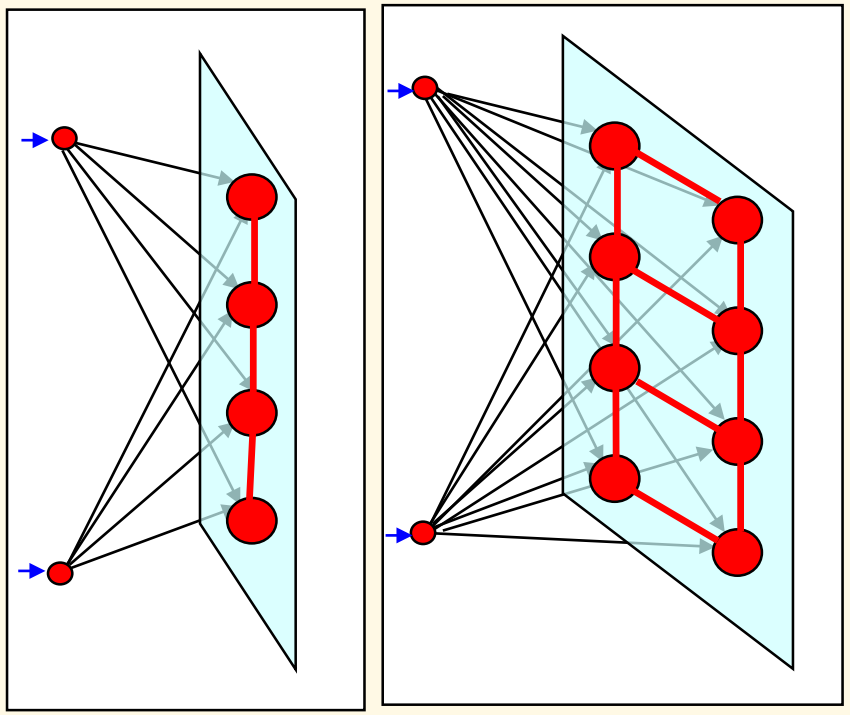
- ***Training mode:*** Unsupervised

- ***Use:***
  - Feature extraction (principal component analysis)
  - Clustering
  - Data compression



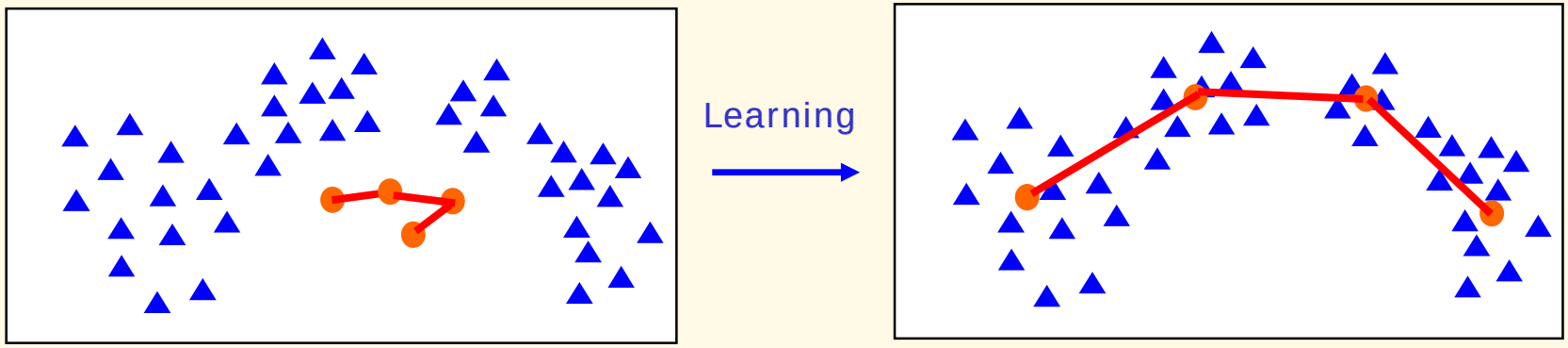
Self organizing maps

- **Architecture:**
  - Single layer of neurons connected to input vectors
  - Topological organization of neurons in a one or two dimensional grid
  - Linear activation



- ***Training mode:*** Unsupervised

- ***Use:***
  - Clustering
  - Data visualisation
  - Data compression



Relation (?) to rule-based systems, ontologies et.c.

- In most of the aforementioned architectures, the neural networks build internal representations of the input patterns using their hidden nodes
- A problem that has not been solved satisfactorily to date is to find a mechanism that leads to the interpretation of the internal representations. We would like to be able to extract rules or reasoning patterns from the structure of the internal representations.
- Existing models are not mature enough to indicate how this mapping is done in the brain
- Overcoming this hurdle may trigger the next leap forward in neural network research

Interdisciplinarity

- **Biological sciences - neurobiology:** Experimental search for suitable biological analogies that can lead to design and implementation of artificial neural networks

- **Physics:** similarities with complex magnetic materials (spin glasses)

- **Mathematics:** Study of non-linear dynamical systems

- **Psychology:** Study of the behavior of biological neural networks (e.g. effective learning methods). Can lead to design and implementation of artificial neural network algorithms (e.g. algorithms based on reinforcement learning)

- **Engineering**

➢ Combination of architectures and algorithms

➢ Combination of ANNs with other information processing systems

➢ Goal: Solution of problems such as: Classification, pattern recognition, regression, modeling and prediction, optimization, optimal control, data mining et.c.

BASIC NEURAL NETWORK EXAMPLE:
Pattern Retrieval Using Hopfield Network

Architecture: Fully connected single layer network with N neurons

- N^2 synaptic weights

$$w_{ij}, i = 1,..., N, j = 1,...N$$

Neurons: McCulloch-Pitts with hard limiter and zero activation threshold

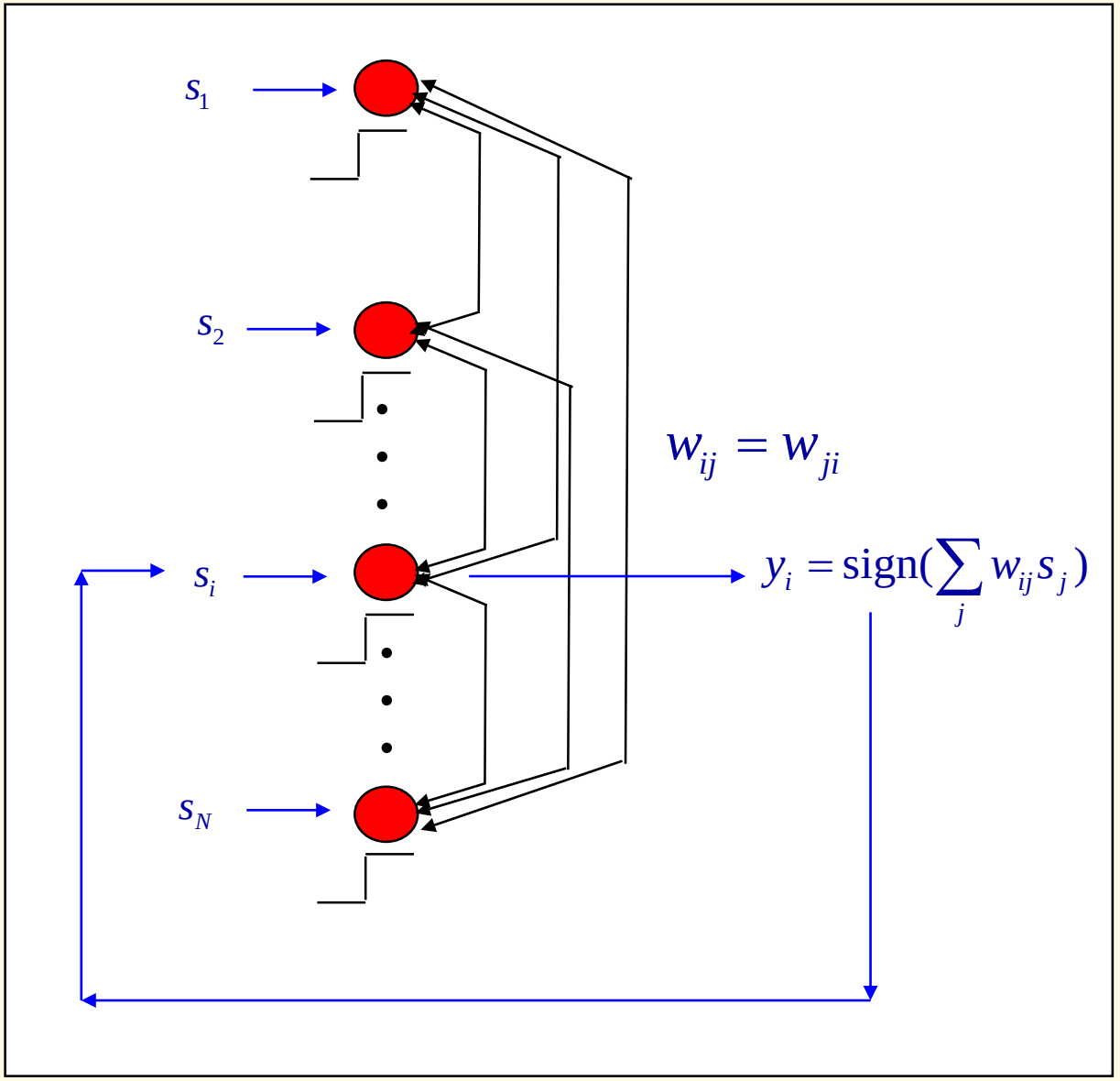
$$y_i = \text{sign}(\sum_j w_{ij} s_j)$$

Dynamics: Asynchronous update of the input nodes (iterative feedback of the neuron output nodes to the corresponding input nodes)

$$s_i^{'} = y_i = \text{sign}(\sum_j w_{ij} s_j)$$

Variations:

- Asynchronous update of inputs: Hopfield model
- Synchronous update of inputs: Little model


s_1
s_2
s_i
s_N
$w_{ij} = w_{ji}$
$y_i = \text{sign}(\sum_j w_{ij} s_j)$

Training: Hebb's rule

- If two neurons connected by a synapse are activated simultaneously, the strength of the synapse increases
- If two neurons connected by a synapse are not activated simultaneously, the strength of the synapse decreases
- This rule is actually observed for certain types of neurons in the brain. Therefore it is based on a solid neurobiological footing
- The Hopfield network is trained using Hebb's rule

Training: Given p patterns $z_i^\mu, i = 1, 2, ..., N$

$\mu = 1, 2, ..., p$

$z_i^\mu = \pm 1$ (randomly distributed)

Synaptic weights are determined using Hebb's rule as follows:

$$w_{ij} = \frac{1}{N} \sum_{\mu=1}^{p} z_i^\mu z_j^\mu$$

Comparison to biological neural networks

Many of the model's properties resemble those of biological neural networks:

- **Distributed processing:** Information is stored collectively in a large area of the network, rather than locally in each processing unit.
- **Robust processing:** Removal or distortion of a small number of synaptic weights does not result in a substantial deterioration of the network's ability for reliable information processing
- **High degree of parallelism:** Performing operations in one neuron is independent of performing opetations in many other neurons. Operations in many neurons can be performed in parallel
- **Learning through examples:** ANNs learn using examples. Learning is associated with the algorithmic updating of synaptic weights. No hard programming (use of specific problem-oriented programming rules) is involved
- **Generalization ability:** Correct response to examples that have not been shown to the network during training

Distributed processing

- Using the dynamic rule

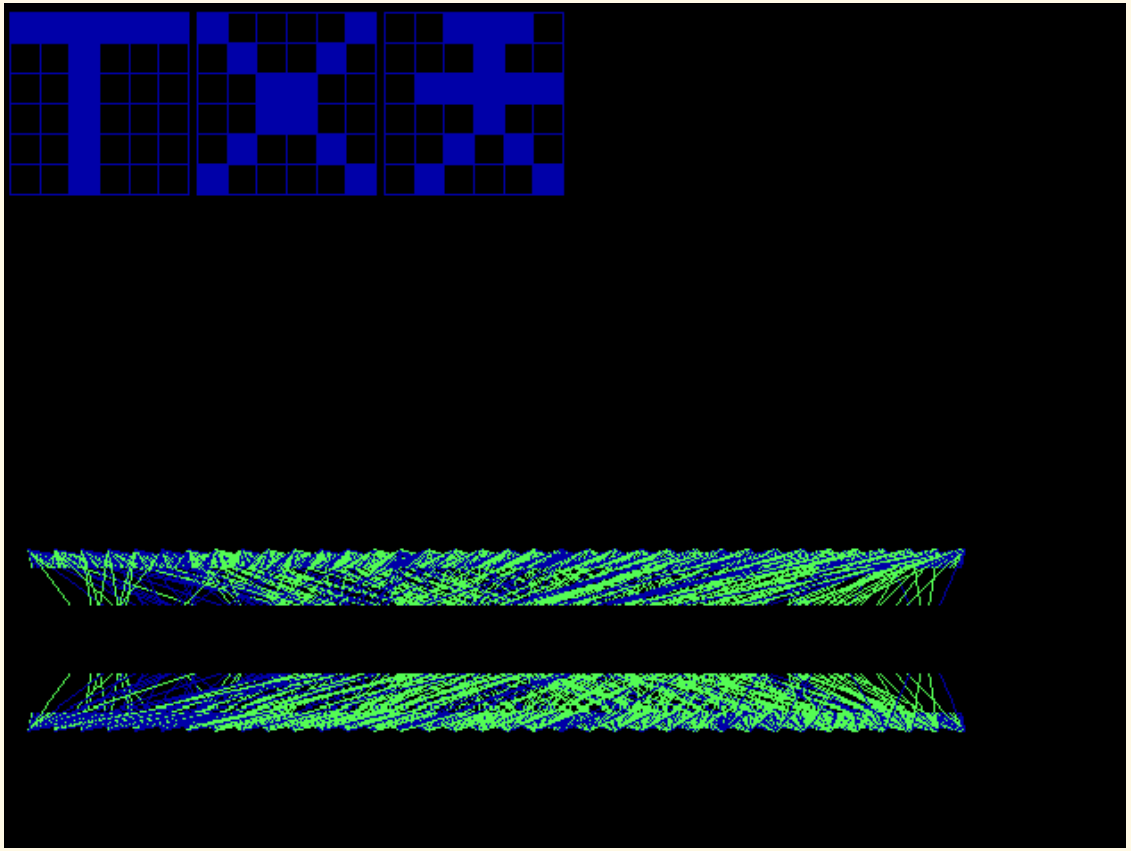
$$s_i' = y_i = \text{sign}(\sum_j w_{ij} s_j)$$

  it is obvious that each output depends on all synaptic weights.

- Moreover, taking into account Hebb's rule, we observe that each synaptic weight depends on all stored patterns. Patterns are stored globally and in a distributed manner, using many synapses.

$$w_{ij} = \frac{1}{N} \sum_{\mu=1}^{p} z_i^{\mu} z_j^{\mu}$$

- Therefore, to retrieve all components of a pattern, a synergy of all synaptic weights is required.



Distributed information processing: A network with 36 neurons which stores 3 patterns. The synaptic weights are depicted graphically (positive in blue, negative in green). The patterns are collectively stored in all synaptic weights.

High degree of parallelism

- Using the dynamic rule

$$s_i' = y_i = \text{sign}(\sum_j w_{ij} s_j)$$

the value of each output can be calculated independently of all other outputs. Therefore, each step in the retrieval process can be performed in parallel.

Learning through examples

- We show one pattern to the network as input. The goal is to have the pattern stored in the network.
- Let us examine whether this pattern has been indeed stored., i.e. whether it appears unchanged as output of the network. If this is true, the pattern will always be recycled unchanged, according to the dynamic rule.

$$y_i = \text{sign}(\sum_j w_{ij} s_j) = \frac{1}{N} \text{sign}(\sum_j \sum_\mu z_i^\mu z_j^\mu z_j^\nu)$$

Consider the sum:

$$\frac{1}{N} \sum_j \sum_\mu z_i^\mu z_j^\mu z_j^\nu = \frac{1}{N} z_i^\nu \sum_j z_j^\nu z_j^\nu + \frac{1}{N} \sum_j \sum_{\mu \neq \nu} z_i^\mu z_j^\mu z_j^\nu =$$

$$z_i^\nu + \frac{1}{N} \sum_j \sum_{\mu \neq \nu} z_i^\mu z_j^\mu z_j^\nu$$

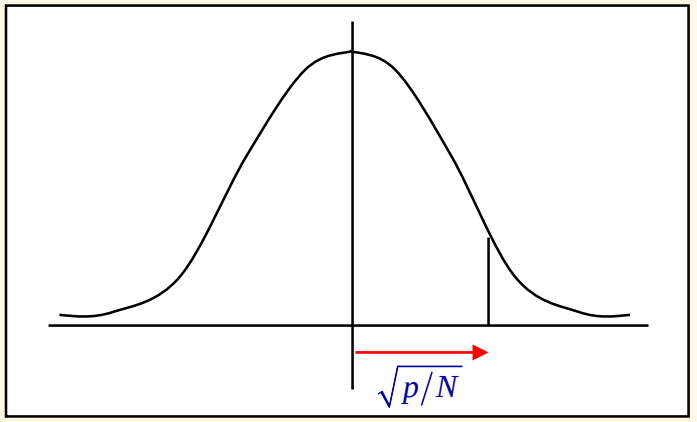
Obviously, the stability of the patterns would be guaranteed, if the second term were equal to zero.

$$k_i = -z_i^\nu \frac{1}{N} \sum_j \sum_{\mu \neq \nu} z_i^\mu z_j^\mu z_j^\nu$$

- The pattern components are randomly distributed and can take values equal to -1 or 1.
- Nk_i is a sum of i.i.d. Random variables, with each taking values -1 ή 1 with equal probability. There are *N(p-1)* such terms in the sums.
- We will consider the limit where *N* and *p* tend to infinity. Remember that processing in the brain is performed using a lot of neurons and for many patterns simultaneously.
- According to the Central Limit Theorem, Nk_i is normally distributed with mean 0 and standard deviation $\sqrt{N(p-1)}$.
- Therefore, k_i is normally distributed with mean 0 and standard deviation

$$\sigma = \frac{1}{N}\sqrt{N(p-1)} \simeq \sqrt{\frac{p}{N}}$$

as *N tends to infinity* **(diagram)**



What is the probability of error (unsuccessful retrieval)?

$$P_{\text{error}} = \frac{1}{\sigma\sqrt{2\pi}} \int_{1}^{\infty} \exp(-\frac{x^2}{2\sigma^2})\, dx = \frac{1}{2}\left[1 - \text{erf}\left(\frac{1}{\sqrt{2\sigma^2}}\right)\right]$$

$$= \frac{1}{2}\left[1 - \text{erf}\left(\sqrt{\frac{N}{2p}}\right)\right]$$

where

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_{0}^{x} \exp(-u^2)du$$

P_{error}	P/N
10^{-4}	0.072
10^{-3}	0.105
10^{-2}	0.185
10^{-1}	0.61

Capacity of the network: It is the maximum number of patterns that can be stored in the network, so that the probability of error does not exceed a certain threshold.

Example: If we are prepared to allow at most 1 error in 10000 trials, the capacity of our network is 0.072**N* patterns.

Generalization ability

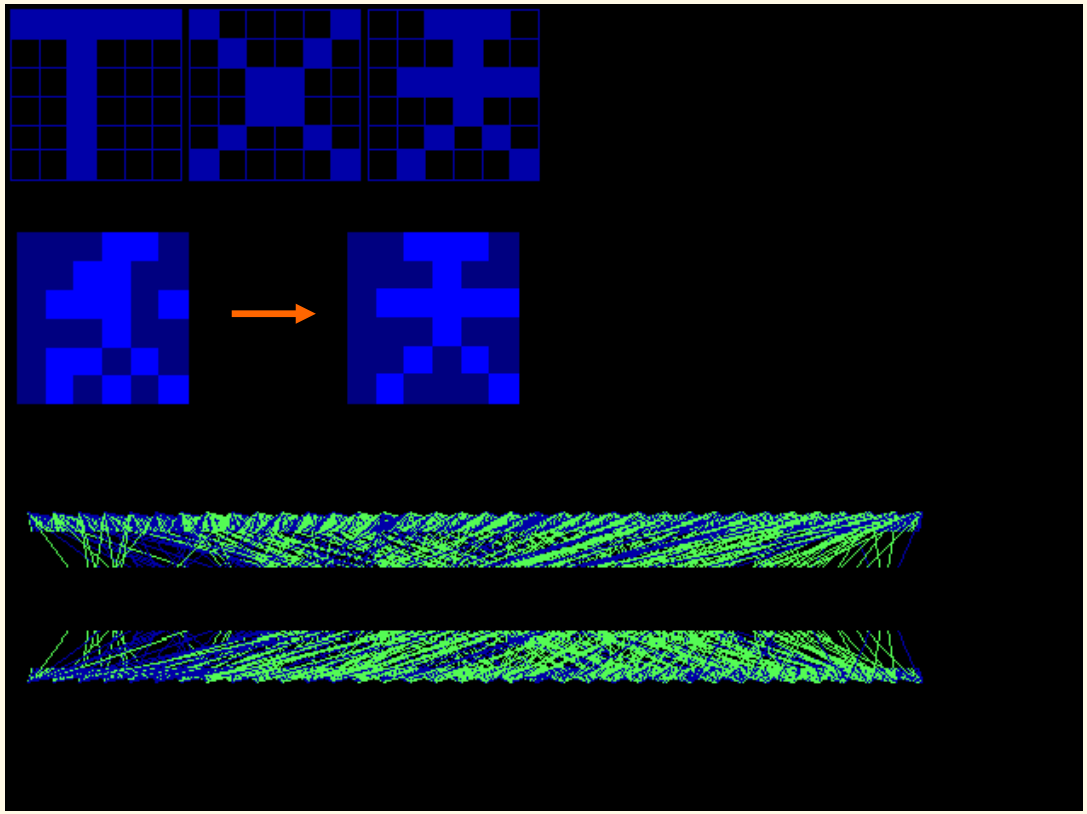
- Having fixed the values of the synaptic weights using Hebb's rule, we present to the network patterns which are similar to the initial ones, but with some of their components altered. We examine whether it is possible to retrieve the original pattern using the dynamic rule.
- Initial pattern: z_i^ν
- Altered pattern: $Z_i^\nu = \begin{cases} -z_i^\nu, i \in X, X \subset \{1,2,...N) \\ z_i^\nu, \quad i \notin X \end{cases}$
- Assume that C components have been altered:

$$\sum_j w_{ij} Z_j^\nu = \frac{1}{N} \sum_{j\mu} z_i^\mu z_j^\mu Z_j^\nu = \frac{1}{N} \sum_j \sum_\mu z_i^\mu z_j^\mu z_j^\nu - \frac{2}{N} \sum_{j\in X} \sum_\mu z_i^\mu z_j^\mu z_j^\nu$$

$$= z_i^\nu + \frac{1}{N} \sum_j \sum_{\mu\neq\nu} z_i^\mu z_j^\mu z_j^\nu - \frac{2}{N} z_i^\nu \sum_{j\in X} z_j^\nu z_j^\nu - \frac{2}{N} \sum_{j\in X} \sum_{\mu\neq\nu} z_i^\mu z_j^\mu z_j^\nu$$

$$= z_i^\nu - \frac{2C}{N} z_i^\nu + \frac{1}{N} \sum_j \sum_{\mu\neq\nu} z_i^\mu z_j^\mu z_j^\nu - \frac{2}{N} \sum_{j\in X} \sum_{\mu\neq\nu} z_i^\mu z_j^\mu z_j^\nu$$

- If $p \ll N$ and $C \ll N$, stability persists and pattern retrieval is achieved. The network is able to generalize.



Generalization: A network with 36 neurons which stores 3 patterns. The stored patterns are retrievable, even when distorted versions are presented to the network, depending of course on the capacity of the network.

Robust processing

- Suppose that a randomly selected fraction $\frac{\lambda}{N^2}$ of the synaptic weights, which were initially trained using Hebb's rule, is destroyed:

$$w_{ij} = \begin{cases} \frac{1}{N}\sum_{\mu=1}^{p} z_i^\mu z_j^\mu, (i,j) \notin Y \subset \{1,\ldots N\}\times\{1,\ldots N\} \\ 0, (i,j) \in Y \end{cases}$$

- If we present the pattern $\mathbf{z}_j$ to the network, we have:

$$\sum_j w_{ij} z_j^\nu = \frac{1}{N}\sum_{j\mu} z_i^\mu z_j^\mu z_j^\nu - \frac{1}{N}\sum_{(i,j)\in Y}\sum_{\mu} z_i^\mu z_j^\mu z_j^\nu$$

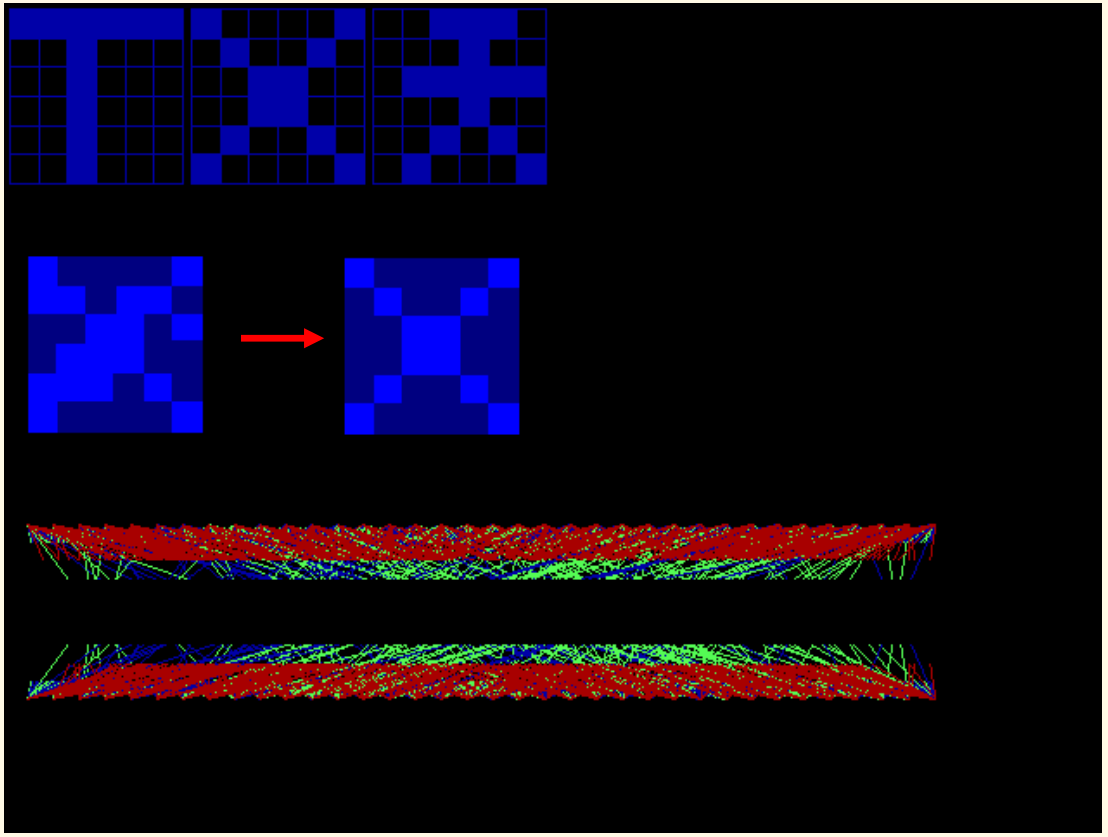
$$= z_i^\nu + \frac{1}{N}\sum_j\sum_{\mu\neq\nu} z_i^\mu z_j^\mu z_j^\nu - \frac{1}{N} z_i^\nu \sum_{(i,j)\in Y} z_j^\nu z_j^\nu - \frac{1}{N}\sum_{(i,j)\in Y}\sum_{\mu\neq\nu} z_i^\mu z_j^\mu z_j^\nu$$

- Let us now call λ_i the number of pairs whose first element is *i*.

$$\sum_j w_{ij} z_j^\nu = z_i^\nu - \frac{\lambda_i}{N} z_i^\nu + \frac{1}{N} \sum_j \sum_{\mu \neq \nu} z_i^\mu z_j^\mu z_j^\nu - \frac{1}{N} \sum_{(i,j) \in Y} \sum_{\mu \neq \nu} z_i^\mu z_j^\mu z_j^\nu$$

$$= z_i^\nu - \frac{\lambda_i}{N} z_i^\nu + \frac{1}{N} \sum_{(i,j) \notin Y} \sum_{\mu \neq \nu} z_i^\mu z_j^\mu z_j^\nu$$

- The noise which could prevent us from retrieving the pattern correctly, is represented by the last two terms.
- In every case, if $\lambda \ll N^2$ and $p \ll N$, all patterns are retrieved successfully.



Robust processing: A network with 36 neurons which stores 3 patterns. 30% of synaptic weights have been „destroyed” (rendered equal to zero). The stored patterns are retrievable, even when distorted versions are presented to the network, depending of course on the capacity of the network.

Energy function

- Energy function: $$E = -\frac{1}{2}\sum_{ij} w_{ij} s_i s_j$$

- *It is easy to show that energy does not increase as the network processes information according to the dynamic rule. During retrieval, the energy of the output pattern is never greater than the energy of the input pattern.*

- *Therefore the stored patterns correspond to local minima of the energy.*

- *This accounts for an important property of the Hopfield network. It is an associative memory.*

Energy function: Pattern retrieval

- **Assumption:** Symmetric synaptic weights: $w_{ij} = w_{ji}$ for all indices i και j
- (true for Hebb's rule, but here we are not bound by Hebb's rule and just assume that the weights are symmetric).
- Consider a pattern which is recycled through the network according to the dynamic rule:

$$s_k^{'} = y_k = \text{sign}(\sum_i w_{ki} s_i)$$

- **1st case:** $s_k^{'} = s_k \longrightarrow$ no change in energy

- **2nd case:** $s_k^{'} \neq s_k \longrightarrow$ Modified energy:

$$H^{'} = -\frac{1}{2} s_k^{'} \sum_i w_{ik} s_i - \frac{1}{2} s_k^{'} \sum_i w_{ki} s_i - \frac{1}{2} \sum_{i,j \neq k} w_{ij} s_i s_j$$

$$= \frac{1}{2} s_k \sum_i w_{ik} s_i + \frac{1}{2} s_k \sum_i w_{ki} s_i - \frac{1}{2} \sum_{i,j \neq k} w_{ij} s_i s_j$$

where we have separated terms containing the specific component from those that do not contain it.

- Change:

$$H' - H = s_k \sum_i (w_{ik} + w_{ki}) s_i = 2 s_k \sum_i w_{ki} s_i$$

- According to the dynamic rule, $\sum_i w_{ki} s_i$ has the same sign as s_k', i.e. the opposite of s_k. Therefore, feeding a pattern back from the output to the imput cannot lead to an increase of energy:

$$H' - H \leq 0$$

- *Provided that the synaptic weights are symmetric, the stored patterns (or, more generally, the attractors of the system), correspond to local minima of the energy function.*