Neural networks

Lukáš Obrdlík (lobrdlik@cs)
Marek Winkler (mwinkler@cs)

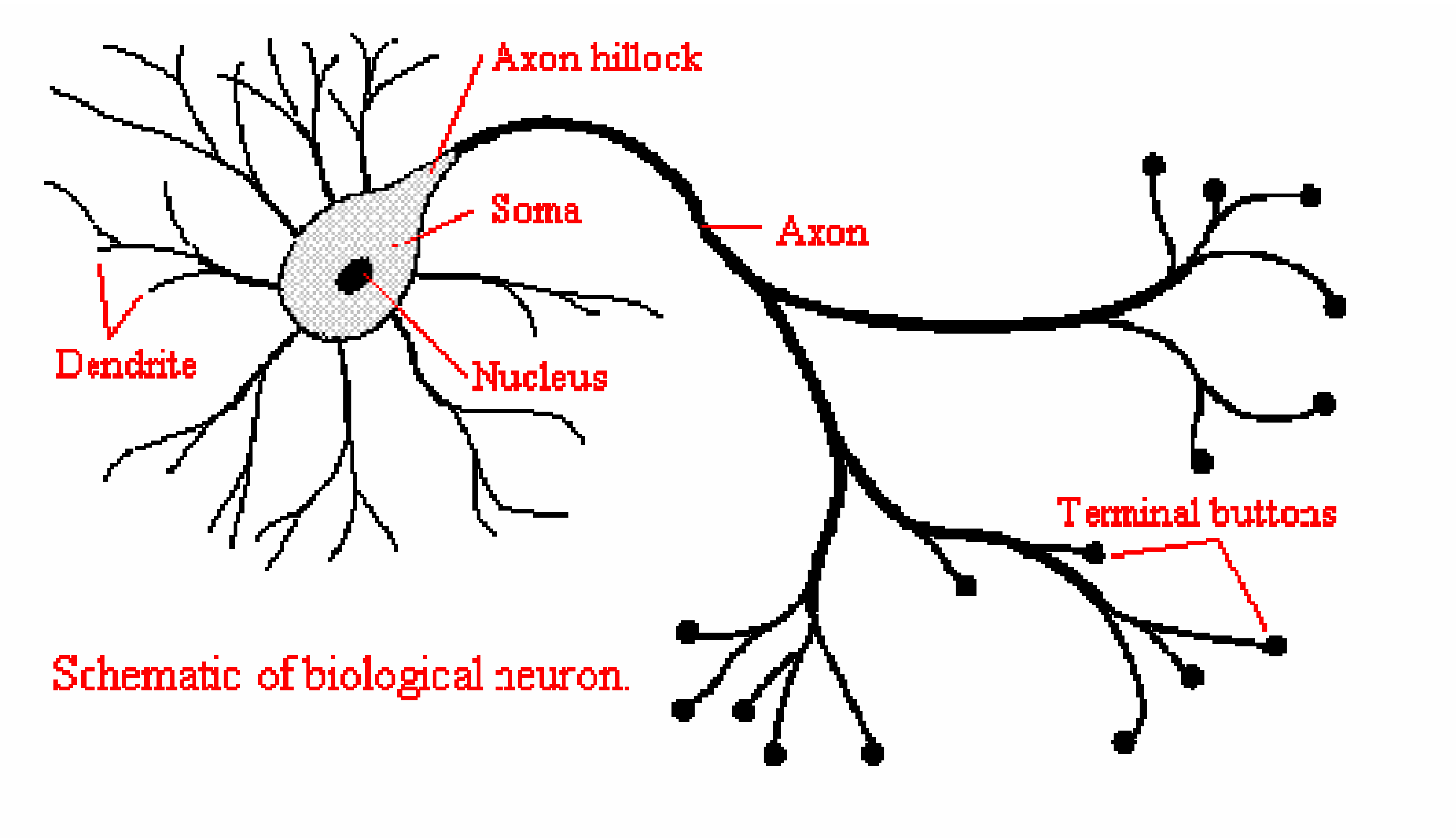
Biological neuron

- Primary part of neural system:

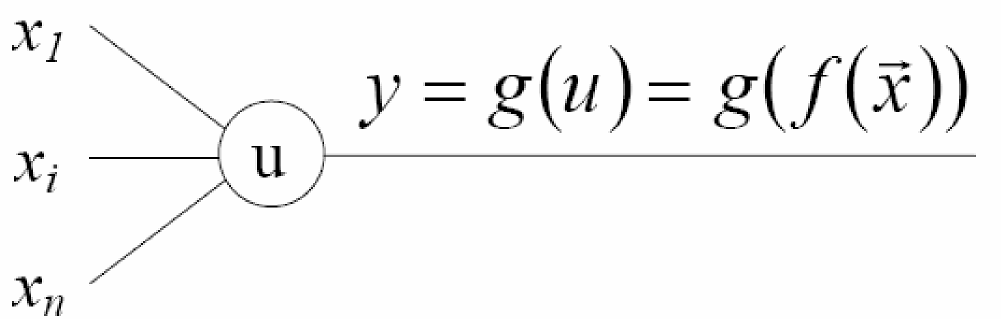
  1. Body – soma
  2. Inputs – dendrites (about 100 dendrites each neuron)
  3. Output – axon (only one per neuron)

- Synapse: interface between dendrite of one neuron and axon of other
- Synaptic bind (weight) can be excitant or inhibitive

Biological neuron



Actual general neuron model



f base function:

- linear base function (LBF) $u = \sum_{i=1}^{n} w_i x_i$

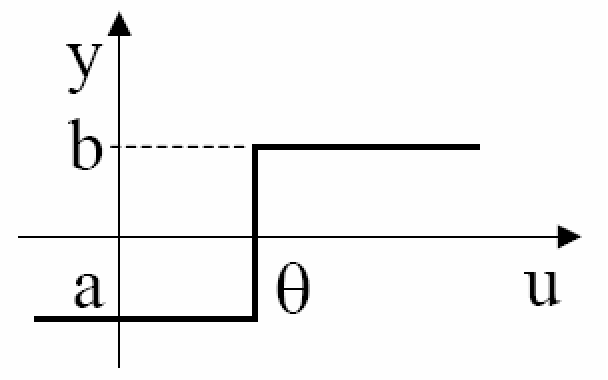
- radial base function (RBF) $u = \left\| \vec{x} - \vec{w} \right\| = \sqrt{\sum_{i=1}^{n} (x_i - w_i)^2}$

g trigger function :

- discrete
- piecewise continuous
- continuous

Linear base function (LBF)
Trigger function - discrete

$$y^{new} = \begin{cases} a & for \quad u < \theta \\ b & for \quad u > \theta \\ y^{old} & for \quad u = \theta \end{cases}$$



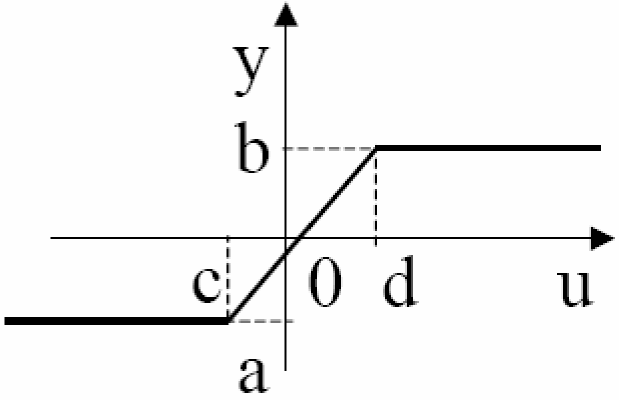
Usually

a = 0, b = 1 binary output

a = −1, b = 1 bipolar output

LBF - Trigger function piecewise continuous

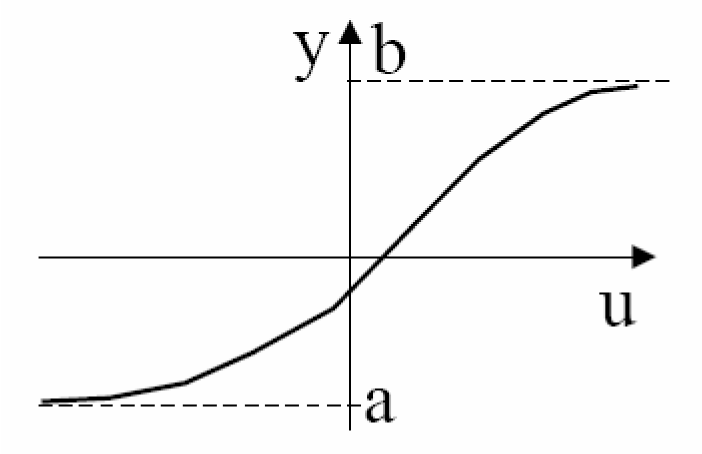
$$y=\begin{cases} a & \text{for } u<c \\ b & \text{for } u>d \\ a+\dfrac{(b-a)(u-c)}{d-c} & \text{for } c\le u\le d \end{cases}$$



LBF - Trigger function - continuous

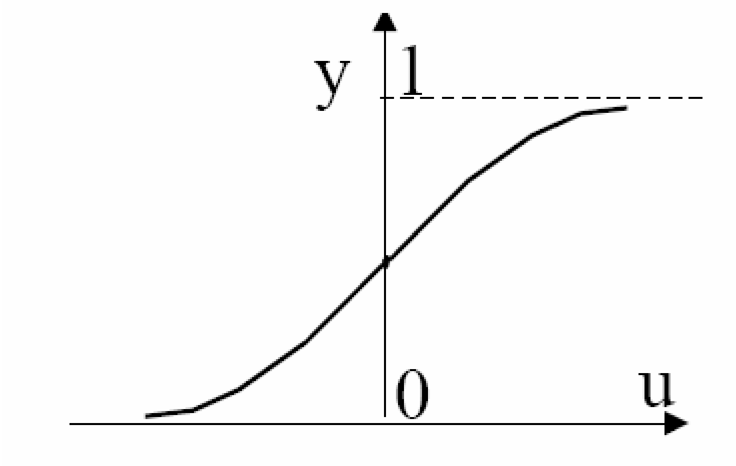
ex. sigmoid:

$$y = a + \frac{b-a}{1+e^{(-\lambda u + c)}}$$



For common values a = 0, b = 1, c = 0:

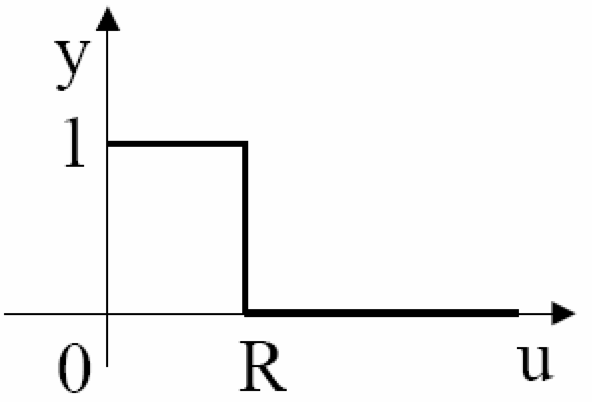
$$y = \frac{1}{1+e^{-\lambda u}}$$



Radial base function (RBF) Trigger function

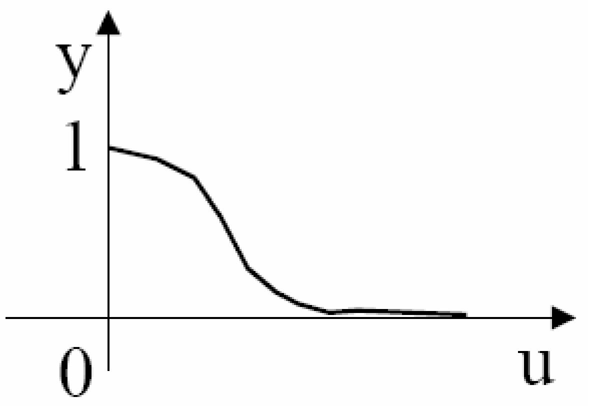
Discrete

$$y = \begin{cases} 1 & for \quad u \le R \\ 0 & for \quad u > R \end{cases}$$



Continuous

$$y = e^{-\left(\frac{u}{\sigma}\right)^2}$$

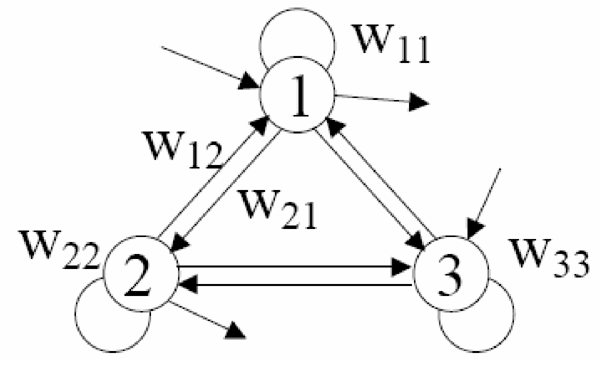


Neuron networks classification

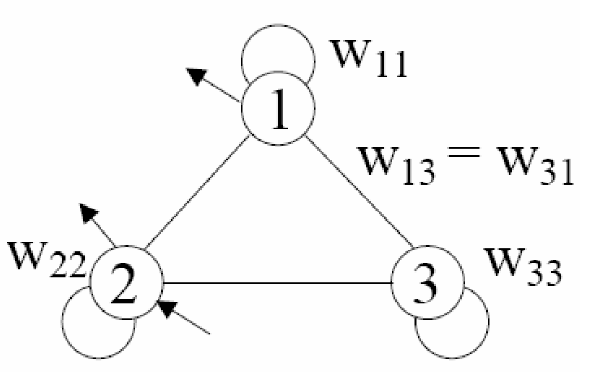
- Architecture
- Learning
- Application
- Implementation
- Computation
- Rating of neural networks

Classification – Architecture 1/4

1) Full connected network

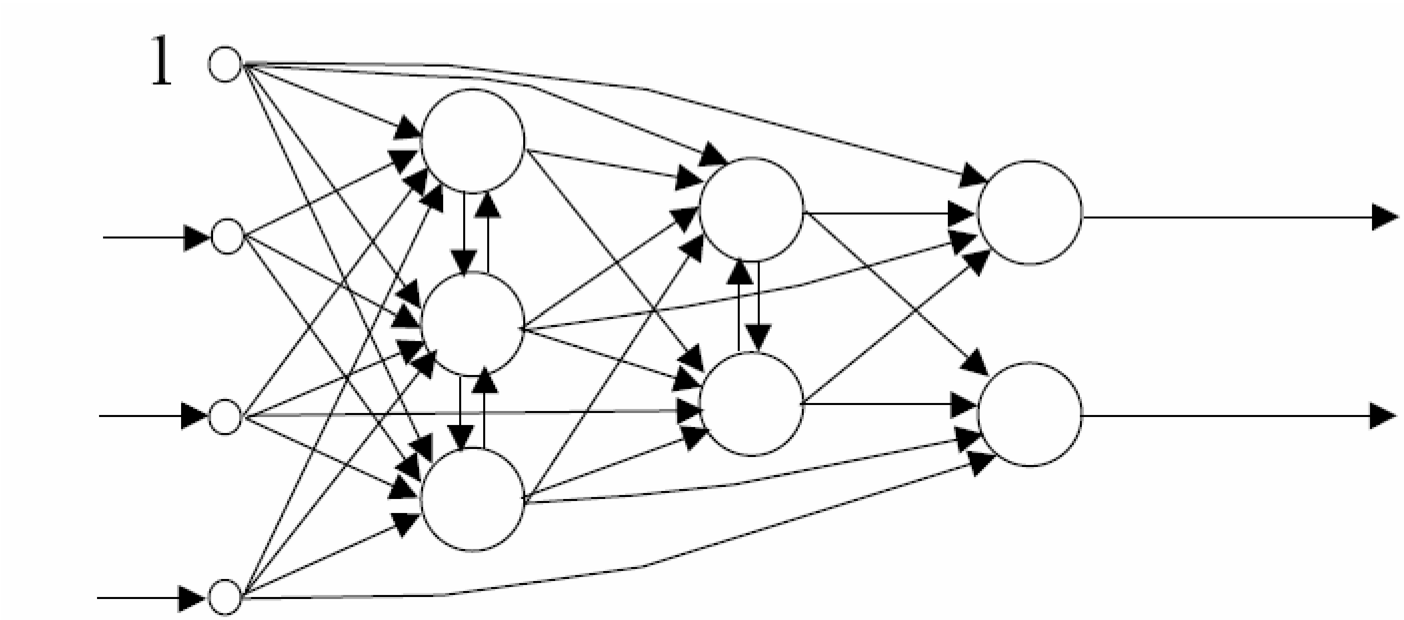


2) Full connected symmetrical network



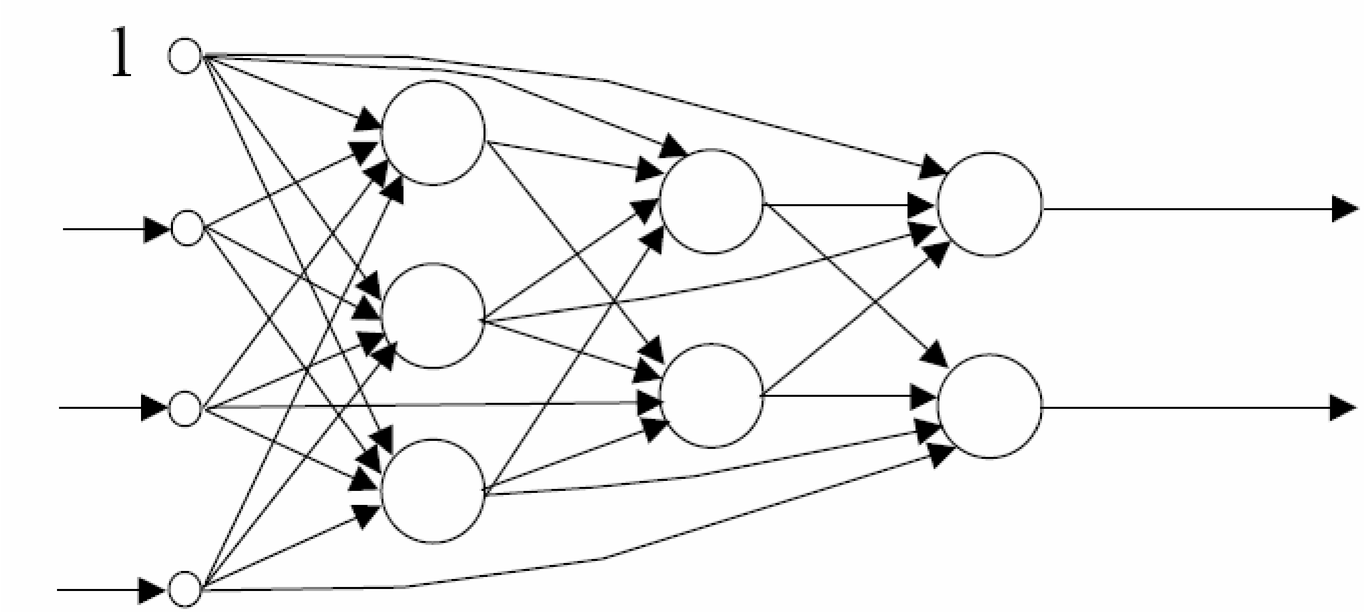
Classification – Architecture 2/4

3) Layer network



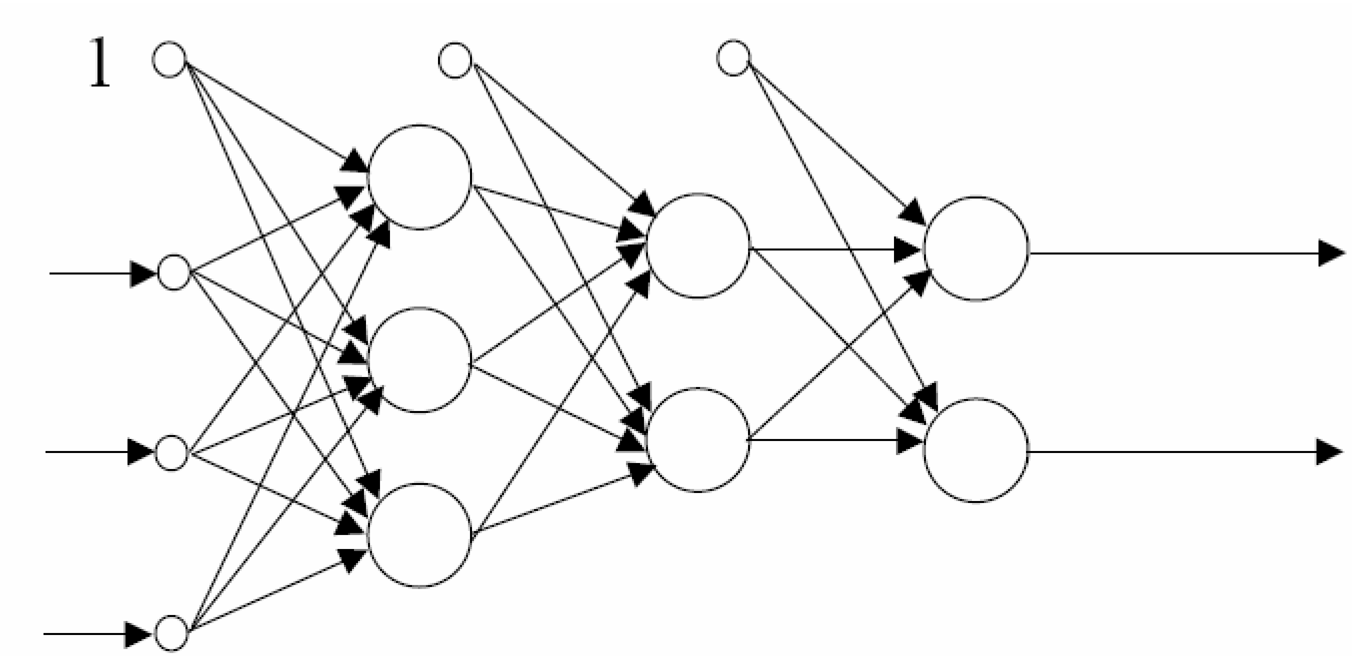
Classification – Architecture 3/4

4) Acyclic network



Classification – Architecture 4/4

5) Feed-forward network



Classification - learning

1. Correlation learning
2. Competitive learning (adapt only weights of the winning neuron eventually weights of its neighbours too
3. Adaptation learning (adapt weights of all neurons)

Classification - application

1. Classification (input vector categorization into one of the existing class)
2. Clustering (creating groups of similar input vectors)
3. Vector quantification (assigning input vector to nearest sample vector)
4. Association (auto-association, hetero-association)
5. Functional approximation
6. Presumption
7. Identification of systems (behaviour of systems approximation)
8. Optimalization

Classification

- Implementation
  - hardware
  - software
- Computation
  - synchronous
  - asynchronous

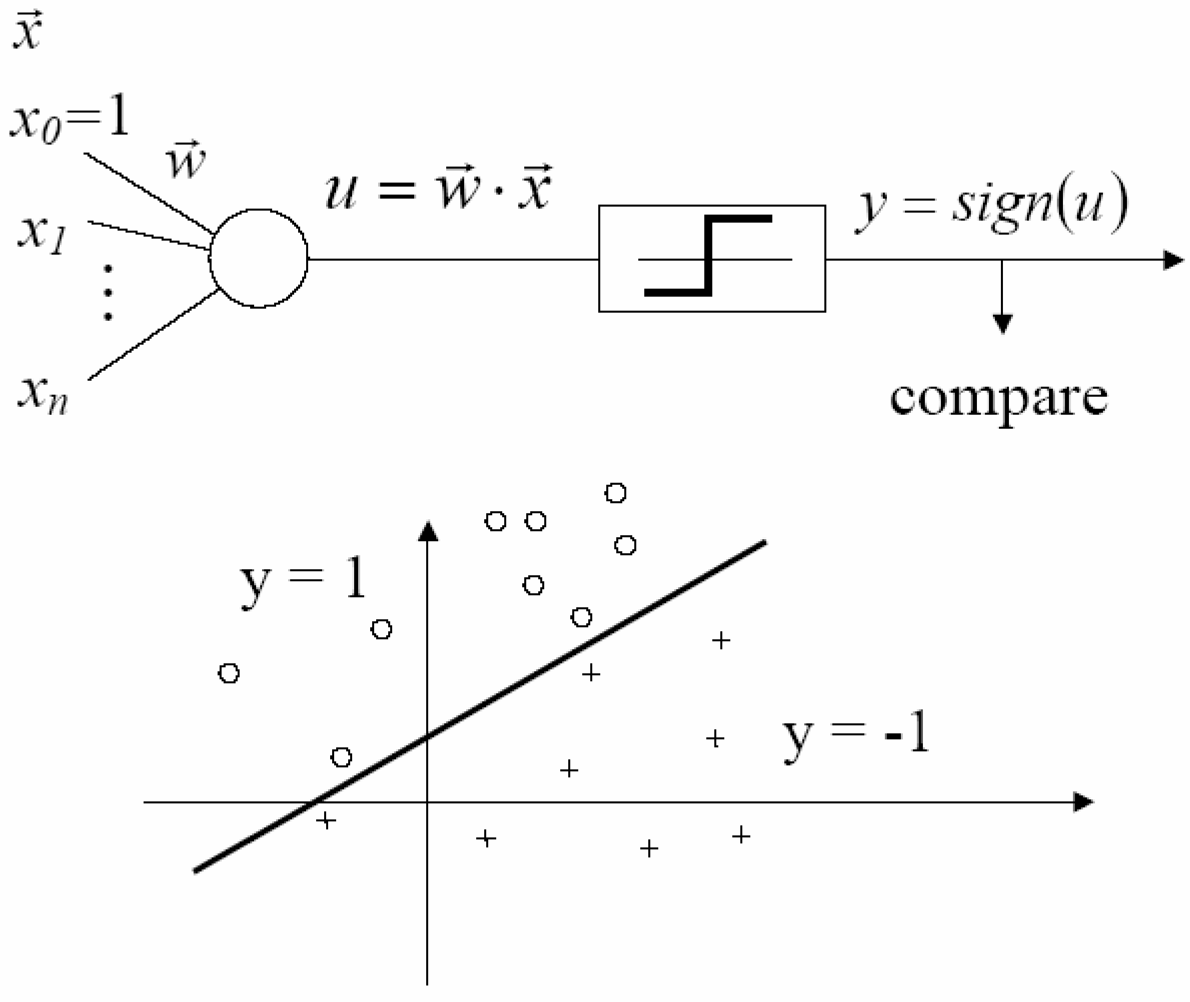
Classification - rating

1. How well can it learn patterns of the training set?
2. How is it successful with new (testing) patterns?
3. What are the requirements? (memory, time..)

Network quality response rating – may $\vec{d}$ is the demanded output vector and $\vec{o}$ is the existing output vector ⇒error is distance of these vectors.

- Euclid rating $$Err_{Euk} = \left\| \vec{d} - \vec{o} \right\| = \sqrt{\sum_{i=1}^{m} (d_i - o_i)^2}$$
- Hamming rating $$Err_{Ham} = \sum_{i=1}^{m} \left| d_i - o_i \right|$$

Perceptron



Perceptron

Basic rule:

$$\vec{w}(s) = random$$

$$\vec{w}(s+1) = \vec{w}(s) + 2p\vec{z}(s) \quad for \quad \vec{w}(s)\cdot\vec{z}(s) \le 0$$

$$\vec{w}(s+1) = \vec{w}(s) \quad else$$

Batch rule:

$$\vec{w}(s) = random$$

$$\vec{w}(s+1) = \vec{w}(s) + p\sum_{\vec{z}\in Z(\vec{w}(s))} \vec{Z}$$

References

- Zbořil, F., Lecture texts for Neural Networks
- Mehrotra,K., Mohan,C.,K., Ranka,S.: *Artificial Neural Networks*, The MIT Press, 1997
- Munakata, T.: *Fundamentals of the New Artificial Intelligence*, Springer 1998
- Hassoun, M.,H.: *Artificial Neural Networks*, The MIT Press, 1995
- Haykin,S.: *Neural Networks*, Macmillan College Publishing Company, 1994