

Data Mining

COIS 4400H / AMOD 5440H

Artificial Neural Networks



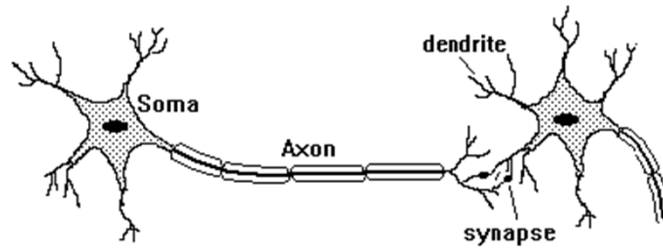
1

Outline

- Background
- Architectures
- Learning
- Uses
- Characteristics
- Neural Network Examples

2

Biological Neuron

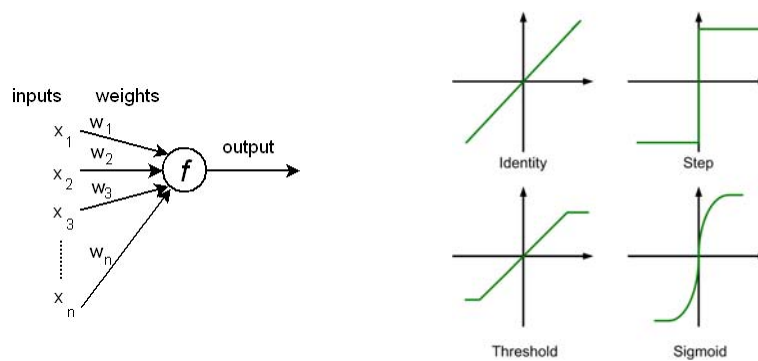


http://www.ii.metu.edu.tr/~ion526/demo/chapter1/section1.1/figure1_1e.html

3

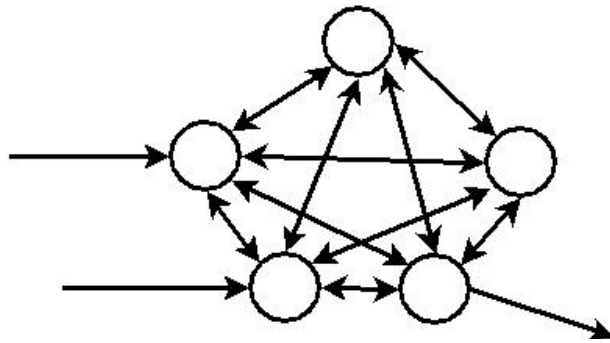
Single Perceptron

invented in 1957 at the Cornell Aeronautical Laboratory by Rosenblatt
can implement AND, OR, NOT functions



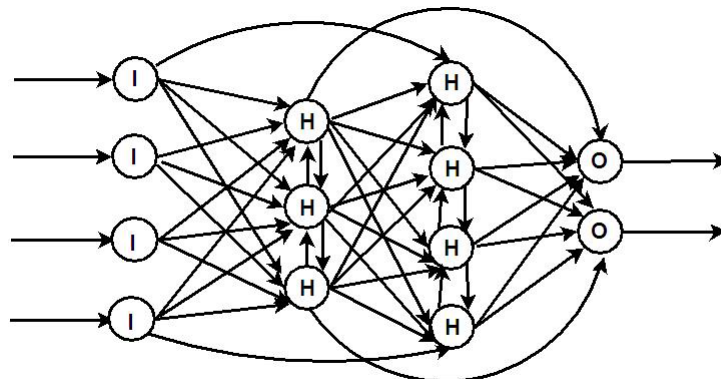
4

Fully Connected Network



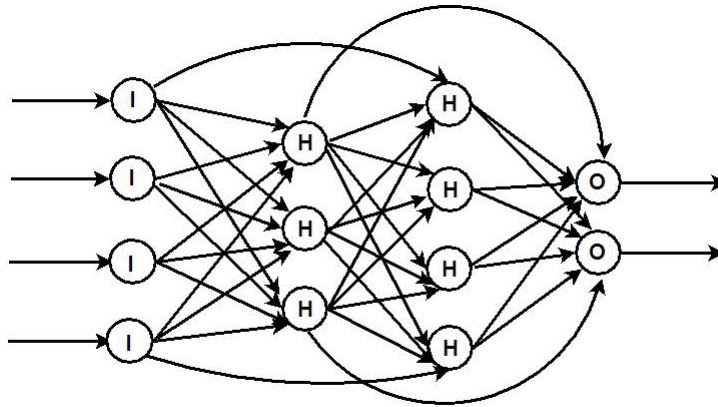
5

Layered Network



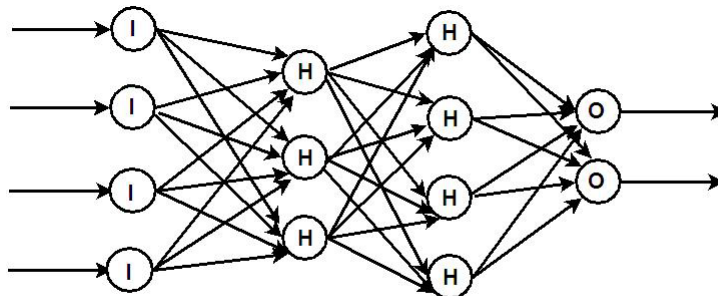
6

Acyclic Network



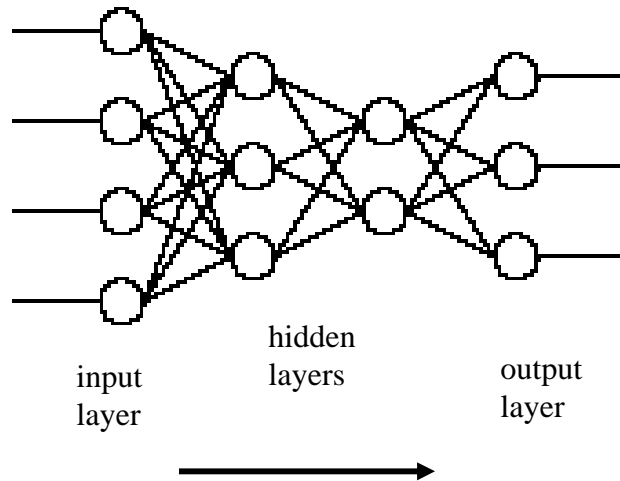
7

A feedforward 4-3-4-2 network



8

Connecting Multiple Perceptrons



9

Learning Methods

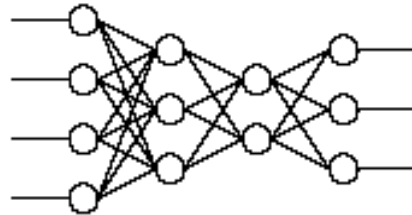
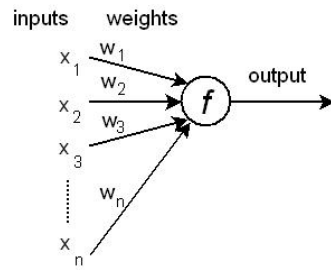
3 types:

- **Correlation**
When different neurons fire at the same time, the weight representing the connection between them should be increased (associative learning)
- **Competitive Learning**
For each sample (set of input values), one node will be the best match. The weights between this winner and the input nodes should be increased.
- **Feedback-based**
Correct behaviour should be rewarded by increasing weights.

Learning means adjusting the weights.

10

Neural Networks: Backpropagation



*randomly initialize weights
for each sample do*

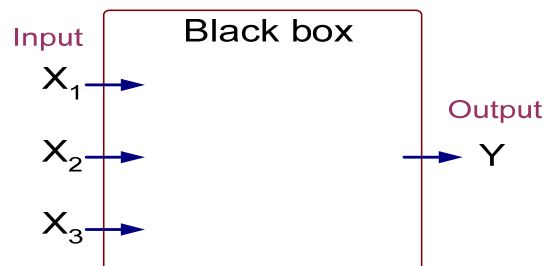
1. present sample to input nodes
2. propagate data through layers, using weights and activation functions
3. calculate results at output nodes
4. determine error at output nodes
5. propagate error backwards to adjust the weights

repeat until stopping criterion satisfied

11

Artificial Neural Networks (ANN)

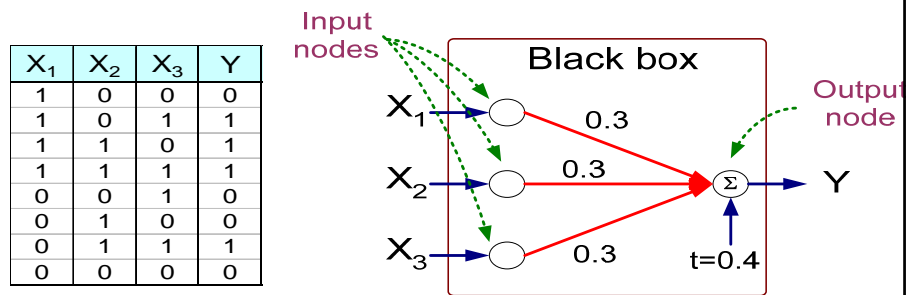
X_1	X_2	X_3	Y
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	0
0	1	0	0
0	1	1	1
0	0	0	0



Output Y is 1 if at least two of the three inputs are equal to 1.

12

Artificial Neural Networks (ANN)

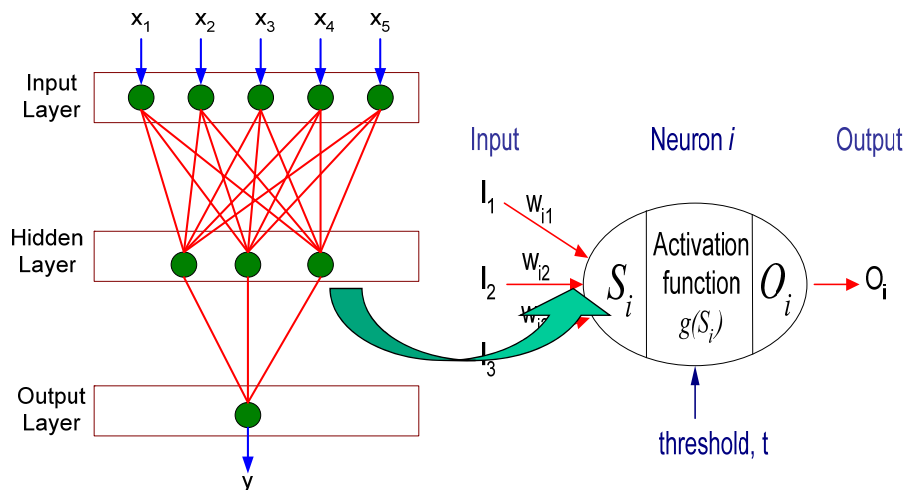


$$Y = I(0.3X_1 + 0.3X_2 + 0.3X_3 - 0.4 > 0)$$

$$\text{where } I(z) = \begin{cases} 1 & \text{if } z \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$

13

General Structure of ANN



Training ANN means learning the weights of the neurons

14

What Neural Networks can be used for

- pattern association
- character recognition
- image compression
- principal component analysis
- classification
- clustering
- vector quantization
- function approximation
- forecasting
- optimization
- control applications
- searching

15

Applications of Neural Networks in Astronomy:

- star/galaxy separation
- spectral and morphological classification of galaxies
- spectral classification of stars
- determine number of binary stars in a cluster
- reduce input dimensionality
- classification of planetary nebulae
- predictions of solar flux and sunspots
- classification of asteroid spectra
- adaptive optics
- spacecraft control
- interpolation of HI distribution in Perseus
- classification of white dwarfs
- detection and classification of CCD defects
- search for antimatter
- ...

16

Characteristics of Artificial Neural Networks

- slow
- poor interpretability of results.
- able to approximate any target function.
- can learn to ignore irrelevant or redundant attributes
- easy to parallelize.
- may converge to local minimum because of greedy optimization, but convergence to global maximum can be achieved through simulated annealing
- choice of network structure non-trivial and time-consuming
- sensitive to noise (a validation set may help here)

17

Some Considerations

- How can we best initialize the weights?
- When do the weights change?
- When should training stop?
- How many input samples do we need?
- How do we determine whether the network has learned something?
- How many hidden layers do we need?
- How many nodes do we need in each hidden layer?
- By how much do the weights change?

18

Network pruning algorithm

Train a neural network

While (performance doesn't degrade by more than a threshold value)

 delete a node or connection

 (retrain the neural network)

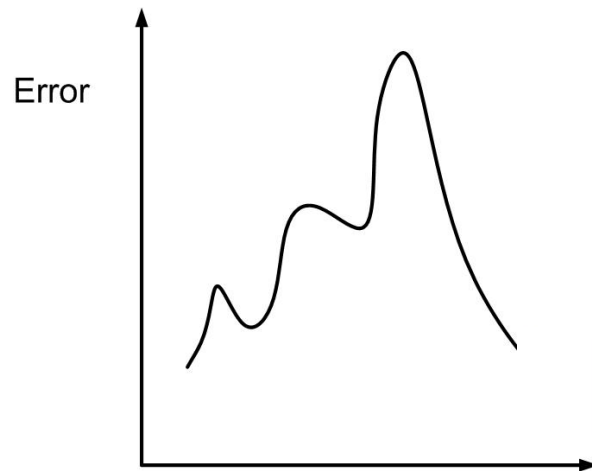
End while

Prune:

- connections with small weights
- Input nodes

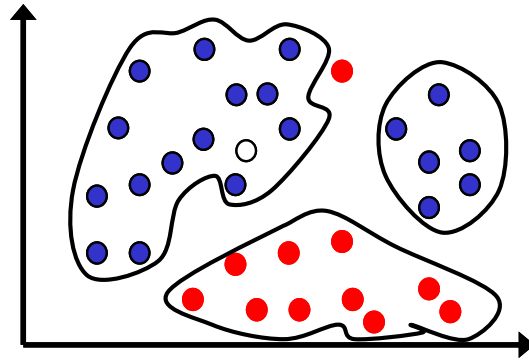
How can we evaluate the performance of a pruned network?

19



20

Why are they useful?



21

Preprocessing requirements

?

23

Topological Organized Networks: Self-Organizing Maps

Select Network Topology

Initialize weights

Initialize current neighborhood size

While (not finished)

 select input sample

 compute Euclidean distance to the weights of

 output nodes

 select the closest output node

 update weights of all nodes within neighborhood of winner

End while

