

Neural Networks (Review)

Piero P. Bonissone

GE Corporate Research & Development

Bonissone@crd.ge.com

(Adapted from Roger Jang)

Outline

Introduction

Classifications

Amount of supervision

Architectures

Output Types

Node Types

Connection Weights

Reflections

Neural Nets: Classification

Supervised Learning

- Multilayer perceptrons
- Radial basis function networks
- Modular neural networks
- LVQ (learning vector quantization)

Unsupervised Learning

- Competitive learning networks
- Kohonen self-organizing networks
- ART (adaptive resonant theory)

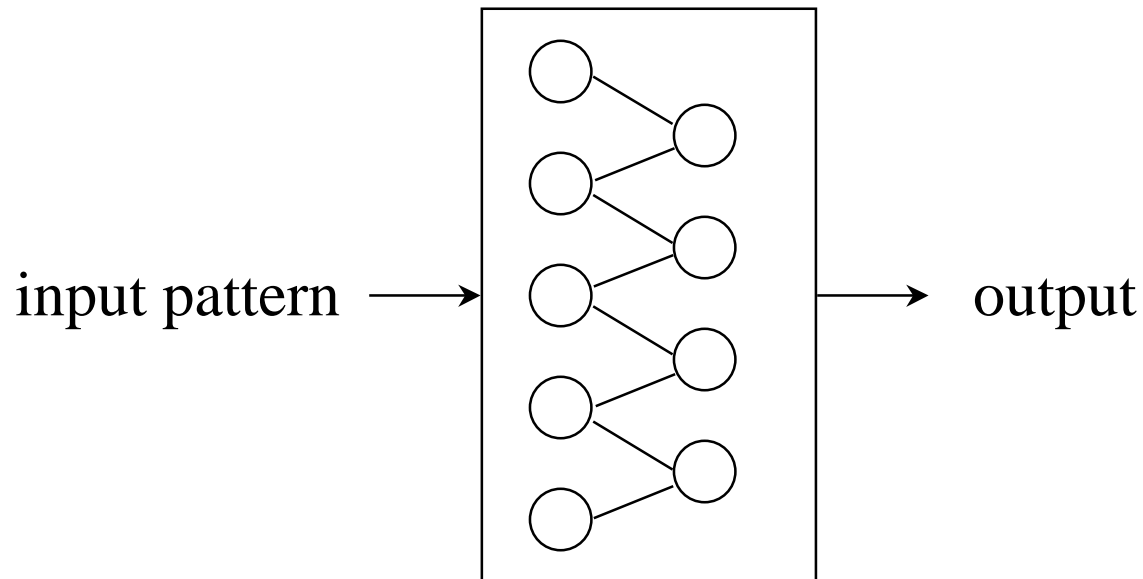
Others

- Hopfield networks

Supervised Neural Networks

Requirement:

known input-output relations



Perceptrons

- Rosenblatt: 1950s
- Input patterns represented is binary
- single layer network can be trained easily
- output o is computed by

$$o = f\left(\sum_{i=1}^n w_i x_i - \theta\right)$$

where

w_i is a (modifiable) weight

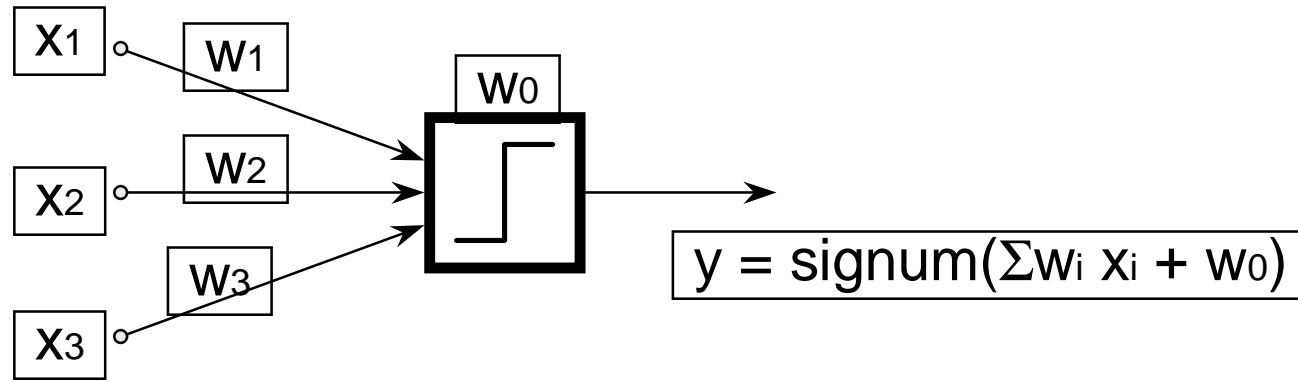
x_i is the input signal

θ is some threshold

$f(\cdot)$ is the activation function $f(x) = \text{sgn}(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$

Single-Layer Perceptrons

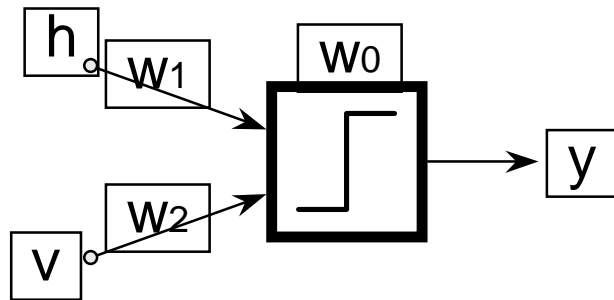
Network architecture



Single-Layer Perceptron

Example: Gender classification

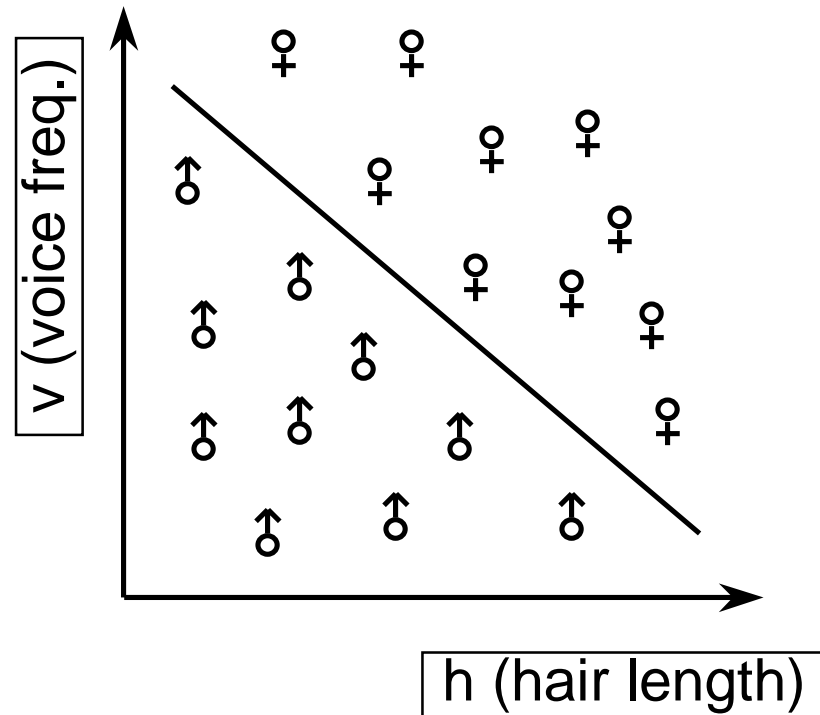
Network Arch.



$$y = \text{signum}(hw_1 + vw_2 + w_0)$$

= {
 -1 if female
 1 if male

Training data



Perceptron

Learning:

select an input vector

if the response is incorrect, modify all weights

$$\Delta w_i = \eta t_i x_i$$

where

t_i is a target output

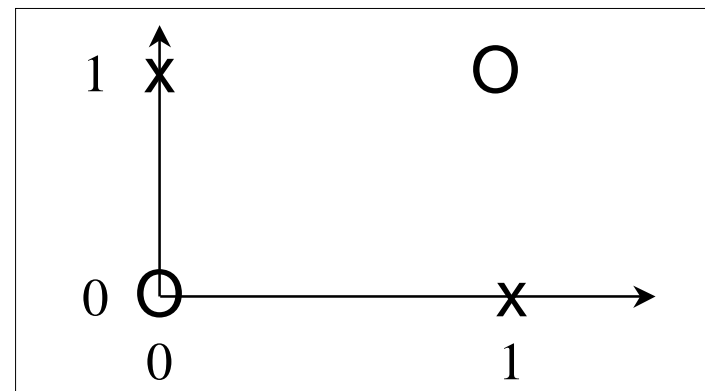
η is the learning rate

If there exists a set of weights, then the method is guaranteed to converge

XOR

Minsky and Papert reported a severe shortcoming of single layer perceptrons, the XOR problem...

x1	x2	output
0	0	0
0	1	1
1	0	1
1	1	0



not linearly separable

...which (together with a lack of proper training techniques for multi-layer perceptrons) all but killed interest in neural nets in the 70s and early 80s.

ADALINE

- Single layer network (proposed by Widrow and Hoff)
- Output is a weighted linear combination of weights

$$o = \sum_{i=1}^n w_i x_i - w_0$$

- The error is described as

$$E_p = \left(t_p - o_p \right)^2 \quad \text{(for pattern p)}$$

where

t_p is the target output

o_p is the actual output

ADALINE

To decrease the error, the derivative with respect to the weights is taken

$$\frac{\partial E_p}{\partial w_i} = -2(t_p - o_p)x_i$$

The delta rule is:

$$\Delta_p w_i = \eta(t_p - o_p)x_i$$

Multi-Layer Perceptrons

-Recall the output

$$o = f\left(\sum_{i=1}^n w_i x_i - \theta\right)$$

-and the squared error measure

$$E_p = (t_p - o_p)^2$$

which is approximated by

$${}^p E_k = \sum_{k=1}^n ({}^p t_k - {}^p o_k)^2$$

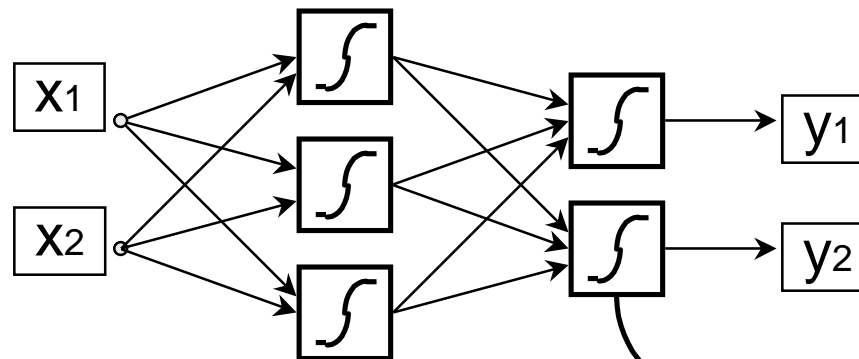
-and the activation function

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

then the learning rule for each node can be derived using the chain rule...

Multilayer Perceptrons (MLPs)

Network architecture



Learning rule:

- Steepest descent (Backprop)
- Conjugate gradient method
- All optim. methods using first derivative
- Derivative-free optim.

hyperbolic tangent
or logistic function

Backpropagation

...to propagate the error back through a multi-layer perceptron.

$$\Delta w_{ki} = -\eta \sum_p \frac{\partial E_p}{\partial w_{ki}}$$

Improvements

Momentum term

smoothes weight updating
can speed learning up

$$\Delta w = -\eta \nabla_w E + \alpha \Delta w_{prev}$$

MLP Decision Boundaries

