Competitive Learning

Neural Networks

Bibliography

Rumelhart, D. E. and McClelland, J. L., *Parallel Distributed Processing*, MIT Press, 1986. - Chapter 5, pp. 151-193.

Kohonen, T., *Self-Organization and Associative Memory*, Springer-Verlag, 1984.

Carpenter, G. and S. Grossberg, A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition machine, *Computer Vision, Graphics, and Image Processing*, 37, 54-115, 1987.

Carpenter, G. and S. Grossberg, ART2; Self-organization of stable category recognition codes for analog input patterns, *Applied Optics*, vol. 26, no. 23, 1987.

Carpenter, G. and S. Grossberg, The ART of adaptive Pattern recognition by a self-organizing neural network, *Computer*, March, 1988.

Spontaneous Learning
Unsupervised Learning

No Teacher

The system must come up with a spontaneous but reasonable scheme of categorizing patterns

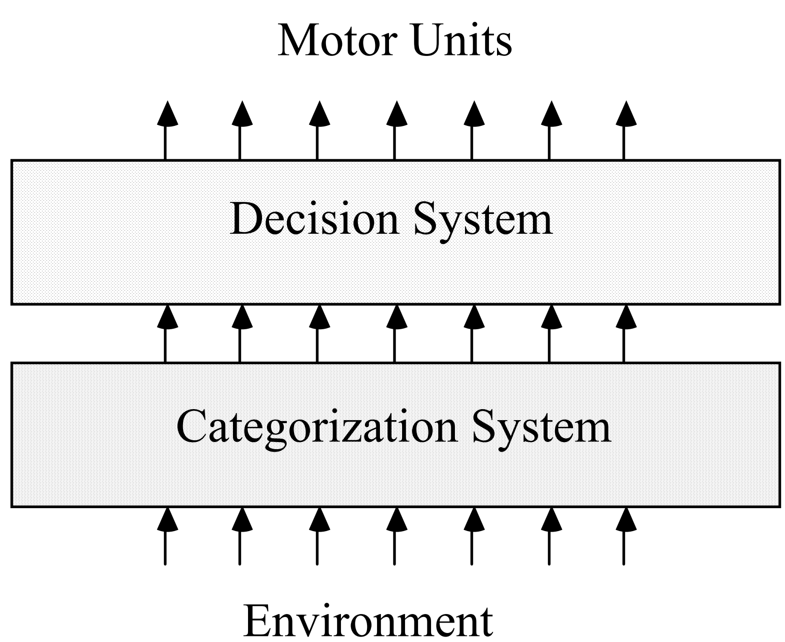
Like-to-Like

Example ART II Classifications

Supervised and Unsupervised have very different goals

Categorization vs Decision Systems

Different Target Applications

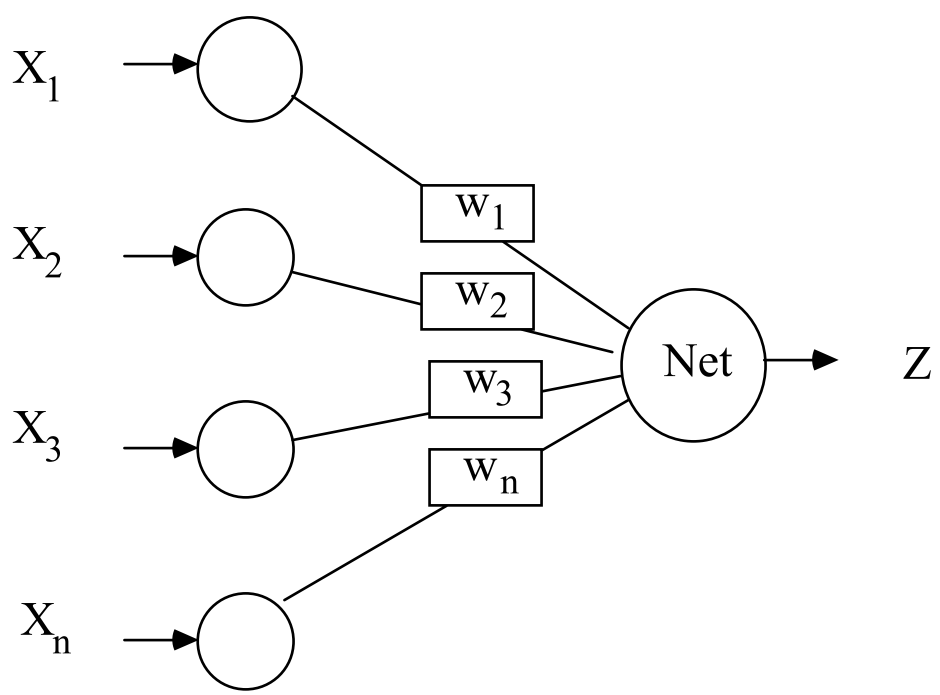


Competitive Learning

Most common scheme for spontaneous learning

Relatively simple and intuitive

Weight vectors a *prototypes*
assume real weights

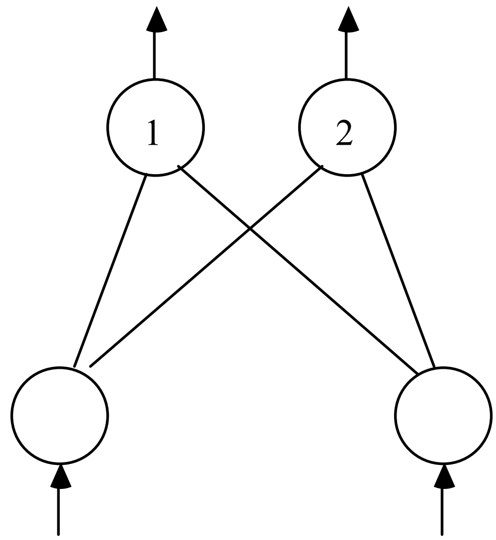


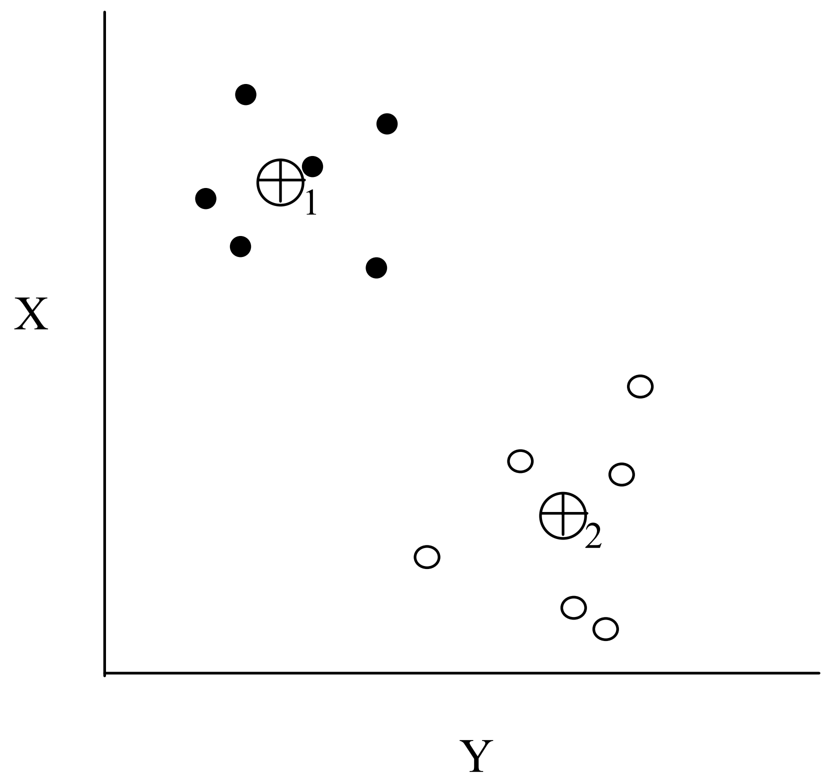
Net most active for pattern similar to weights

Standard Cluster Diagram

Localist Model

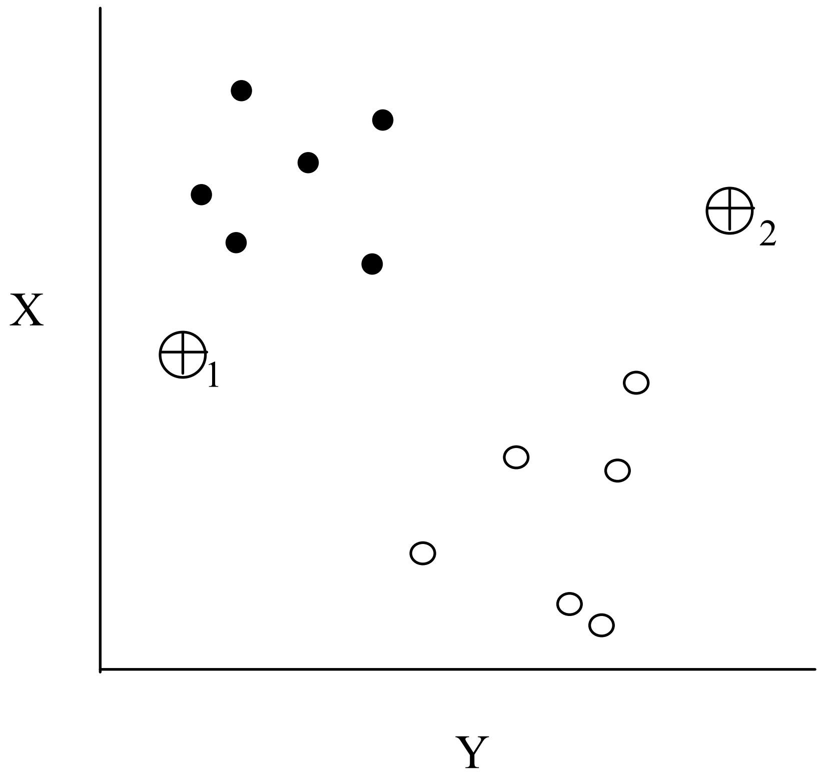
2 prototype example
(Lateral Inhibition, Winner take all)

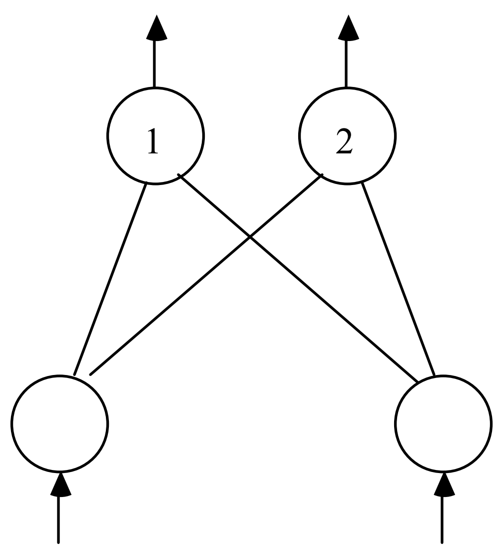




Desired Goal

How do we reach it from an initial state





Simple Competitive Learning Algorithm

Binary Inputs

Top nodes winner take all

Only winning unit has weights adjusted

Each unit as fixed weight ∑1, weight is shifted during learning

$$\Delta w_{ij} = \begin{cases} 0 \text{ if unit j loses else} \\ g(\frac{s_i}{n} - w_{ij}) \end{cases}$$

where n is the number of active s_i

Weight is shifted such that weight vector better matches the current winning input

Extended models

Arbitrary inputs and weights

can use a distance metric rather the *net*

Simple Unsupervised Learning Model With Distance Metric

Initialize *n* nodes in the attribute dimension space (could be very small *n* and be constructive)

Until Convergence (Δd very small)

   Input new **x**

   Choose node *i* closest to **x** (Argmin$_i$ (D(n_i,**x**))

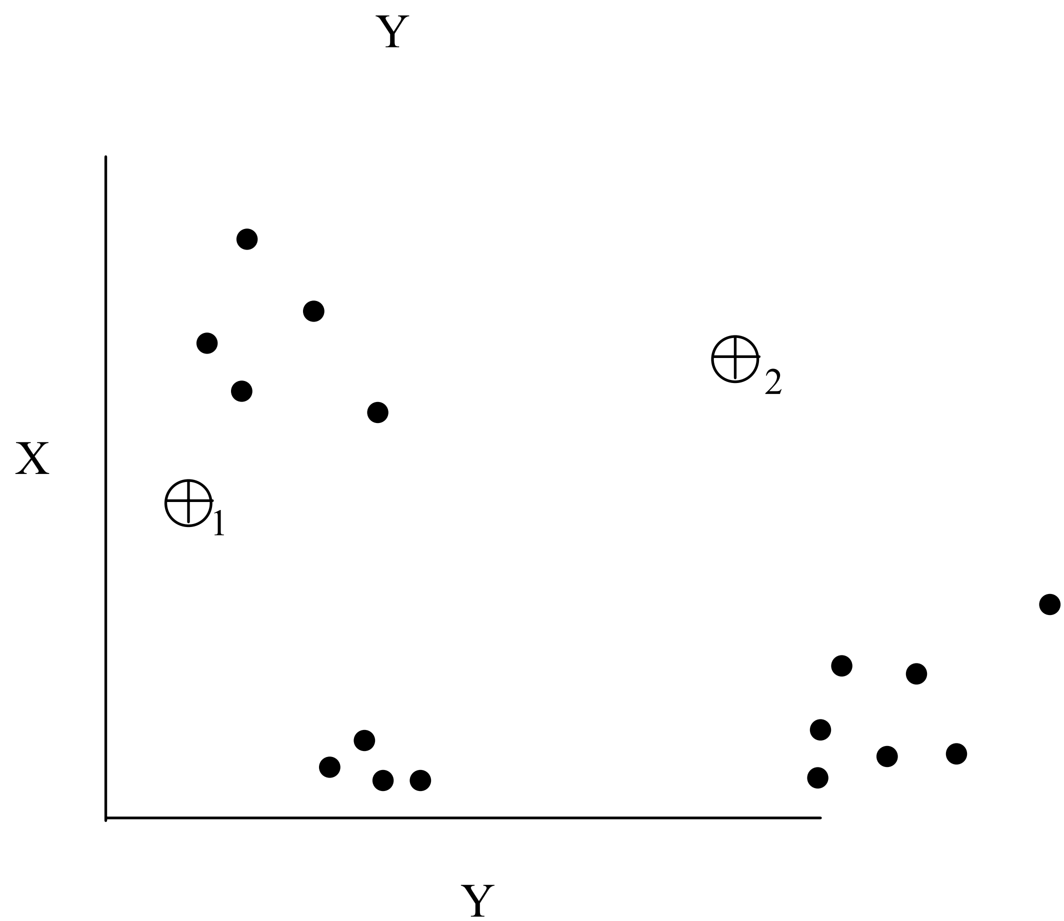
   Optional: Add new node at **x** (how to decide?)

   Move n_i slightly closer to **x** ($\Delta d_i = d_i + cx_i$)

      (d is a node dimension and c is a learning rate)

Optional: Prune nodes (how to decide?)

Dynamic Node Growth



What will happen here

vigilance metric for node growth

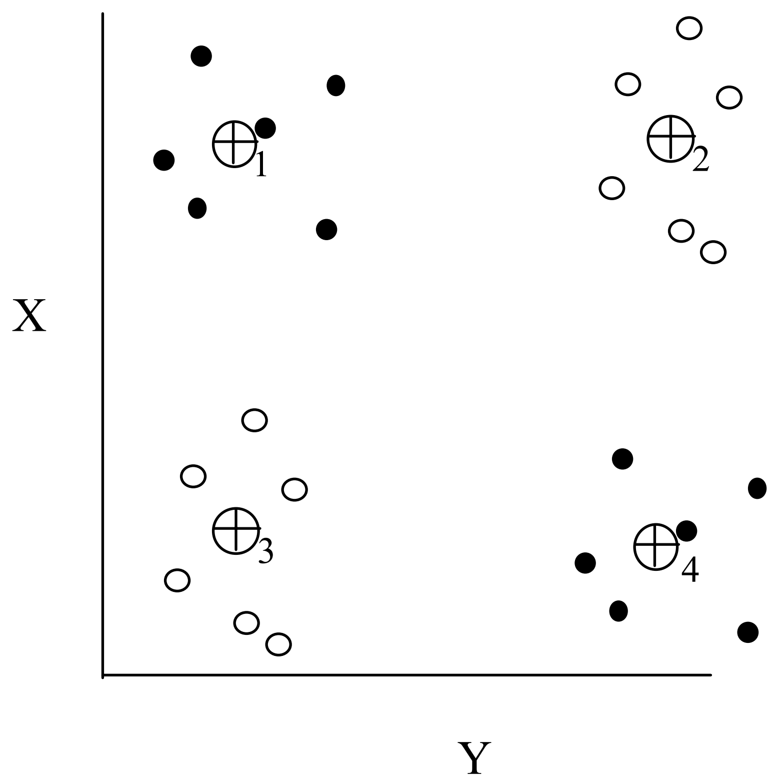
non-global vigilance

noisy patterns

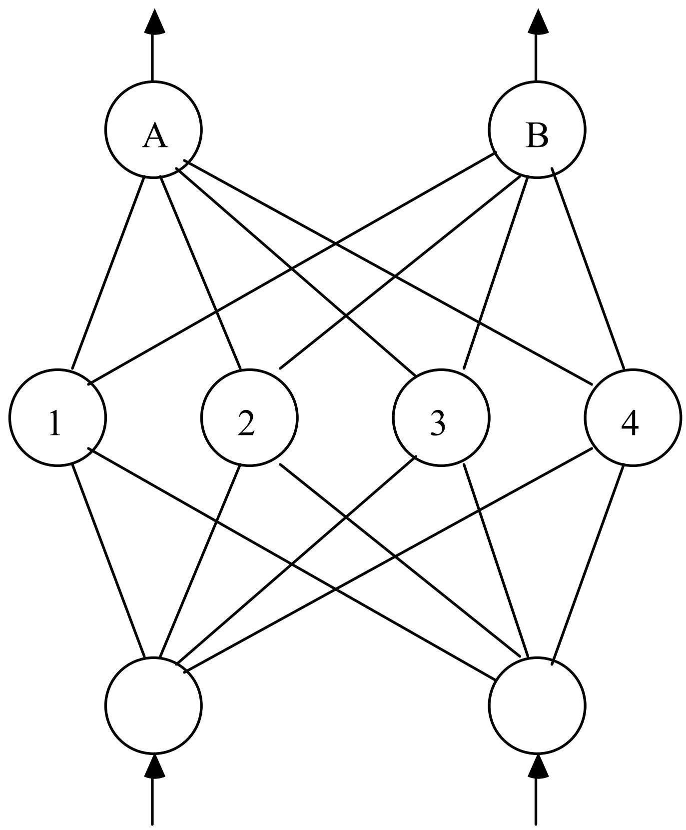
Supervised learning with competitive scheme

Simply assign output value to each prototype

Basically, multiple prototypes can have the same value



Multi-layer net using competitive learning



RCE Learning

(Restricted Coulomb Energy)

ART (Adaptive Resonance Theory)

Spontaneous Competitive Learner

Dynamic Node Growth

Global Vigilance

Competitive Learning

Powerful Intuitive Model

Focused applications (Categorizing)

Easily extended to supervised models

Potential Integration