NEURAL NETWORKS +

MOHAMMAD GHODDOSI

NEURAL NETWORKS VARIATIONS

- MLP
- Autoencoders
- RBF
- SOM
- LVQ
- ART
- Hopfield
- Boltzmann Machine

RBF

MOHAMMAD GHODDOSI

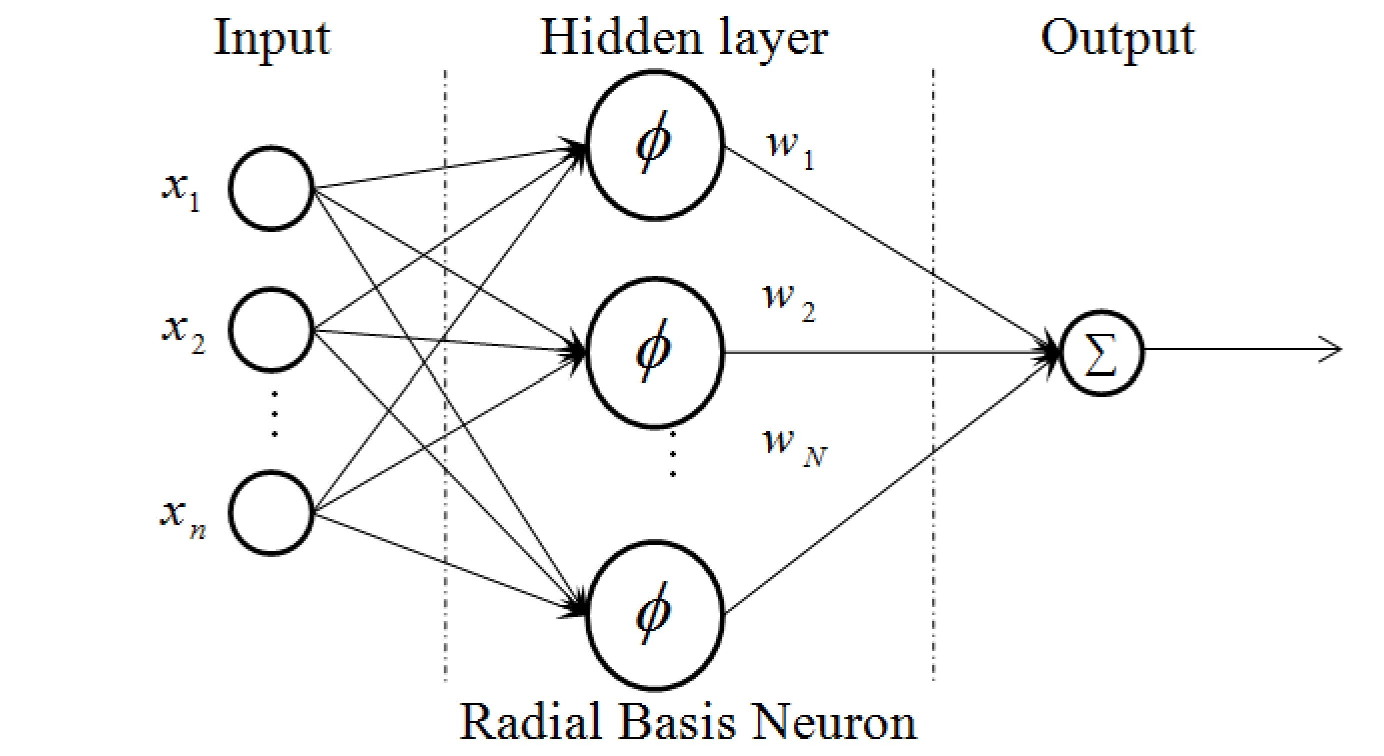
RADIAL BASIS FUNCTION (RBF)

- Like SVM kernels
- Use a kernel layer as hidden layer (without learning)
- Use a single layer for classification (with supervised learning)

COVER’S THEOREM (1965)

- A complex pattern classification problem cast in a high dimensional space nonlinearly, is more likely to be linearly separable than in a low dimensional space.

RBF ARCHITECTURE



RBF EQUATIONS

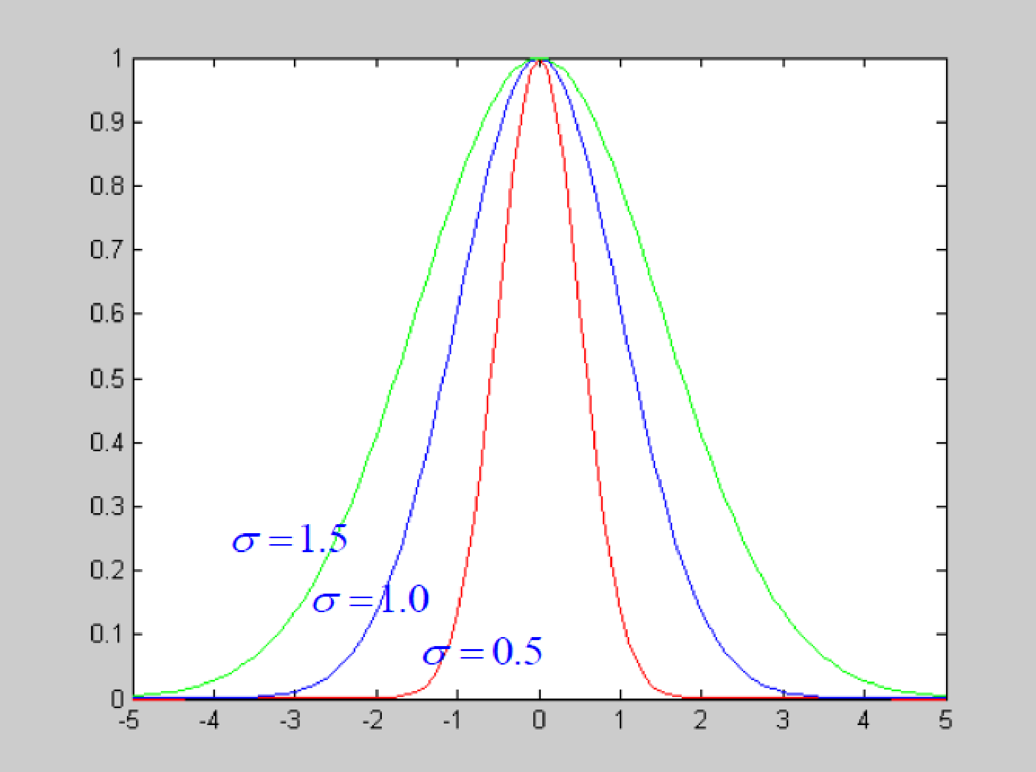
$$Z = W^T . \varphi(X)$$

$$H = \begin{cases} 0 & if\ Z < 0 \\ 1 & if\ Z > 0 \end{cases}$$

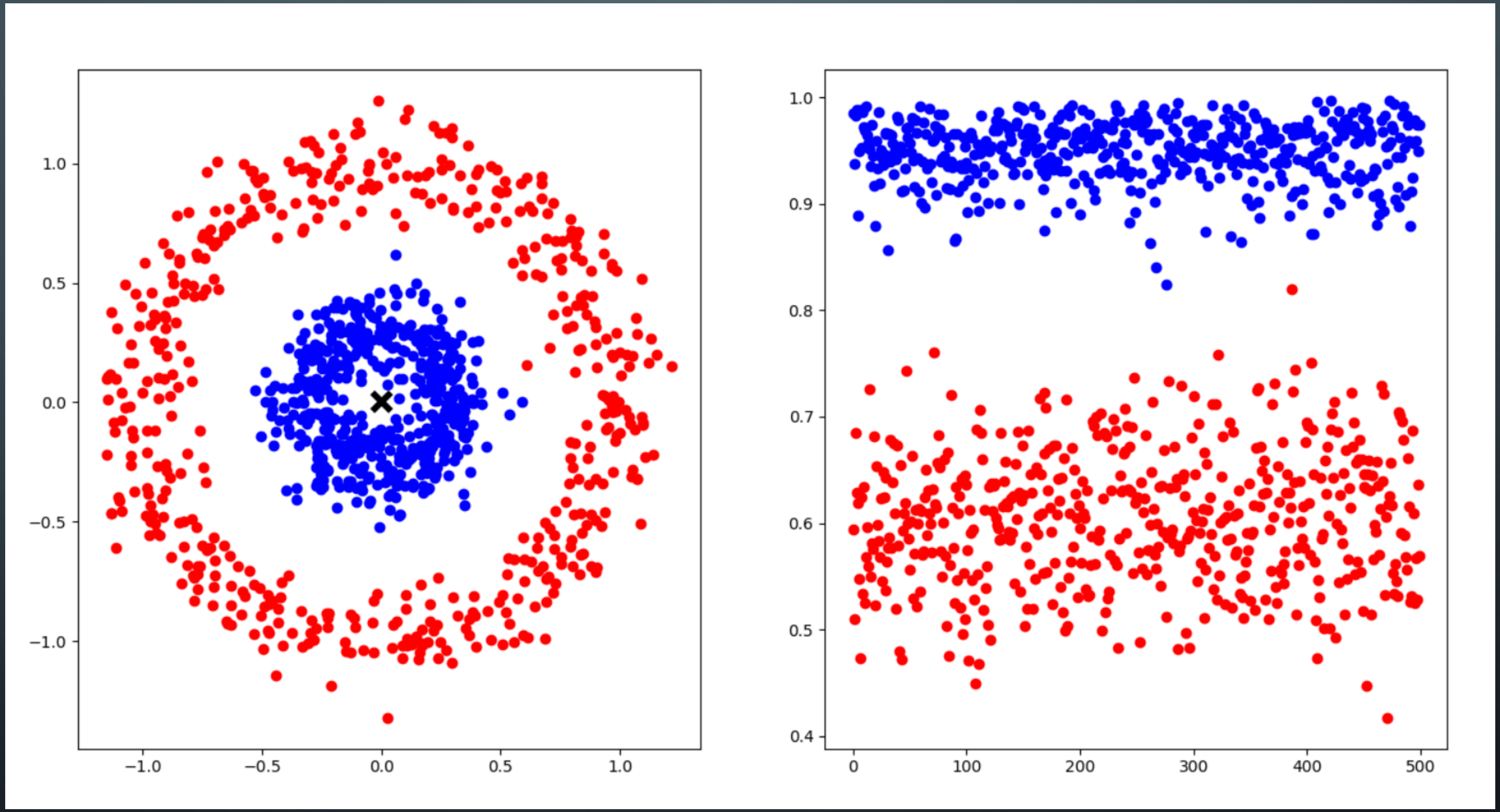
RADIAL BASIS FUNCTIONS

- Center: x_i
- Distance measure: $\mathrm{r} = ||x - x_i||$
- Shape: ϕ

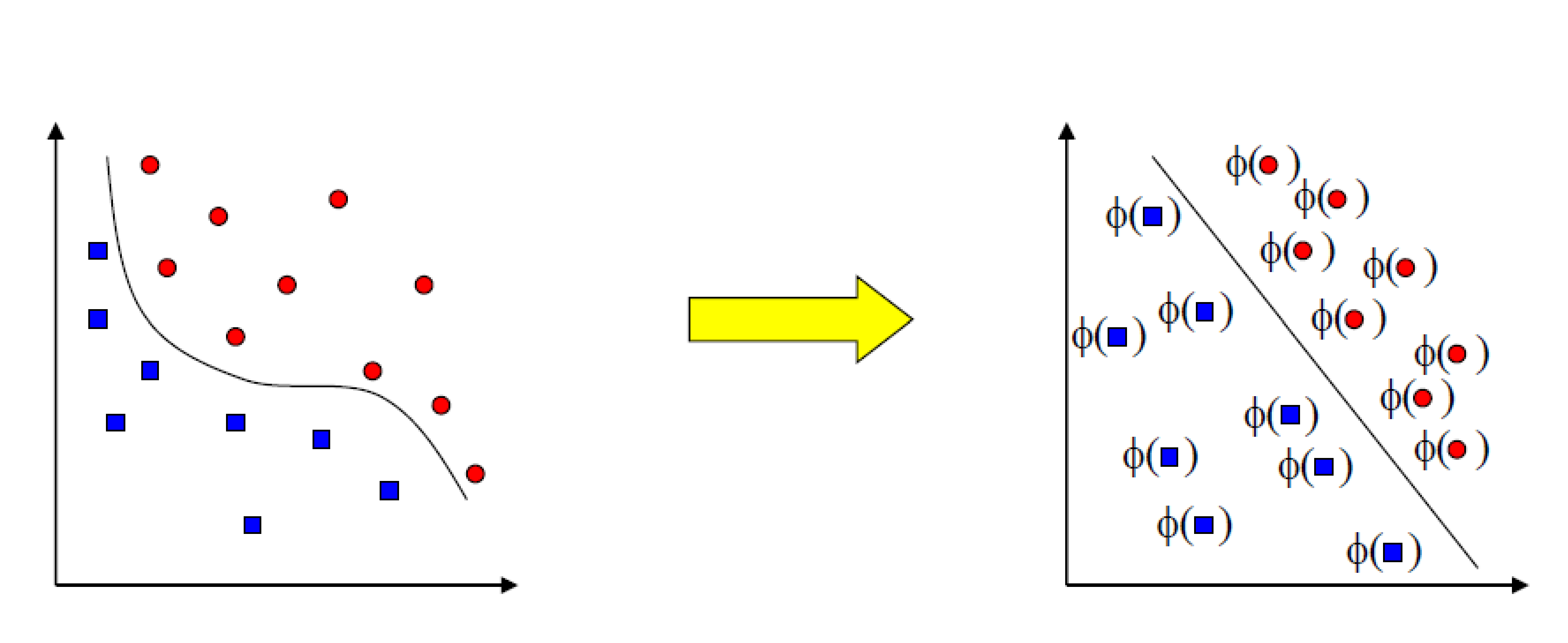
$$\phi(\mathrm{r}) = \mathrm{e}^{-\frac{r^2}{2\sigma^2}}$$



EXAMPLE (CIRCLE)



RBF KERNELS



RBF TRAINING

- Non-iterative

$$Y = W^T.\varphi(X)$$

$$Z.\ \phi(X)^{-1} = W^T$$

SOM

MOHAMMAD GHODDOSI

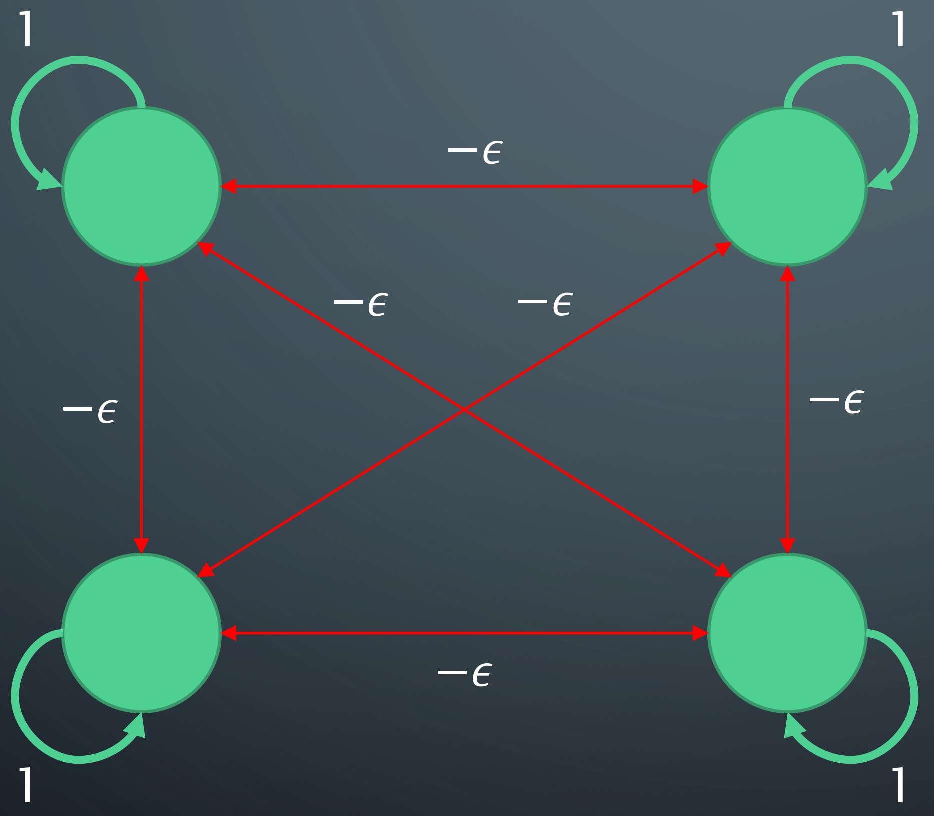
COMPETITIVE NETS

- Competition in biological neurons
  - Problem solving
  - Decision making
- Winner-take-all competition

COMPETITIVE NETS

- Fixed weights
  - MAXNET
  - Hamming
- Kohonen Self Organizing Map (SOM)

MAXNET



$$w_{ij} = \begin{cases} 1 & if\ i = j \\ -\epsilon & if\ i \neq j \end{cases}$$

EXAMPLE

$a_1(0) = 0.6,\ a_2(0) = 0.2$

$a_3(0) = 0.8,\ a_4(0) = 0.4$

$\epsilon = 0.2 \quad 1/M = 0.25$

step k	$a_1(k)$	$a_2(k)$	$a_3(k)$	$a_4(k)$
0	0.6	0.2	0.8	0.4
1	0.32	0	0.56	0.08
2	0.192	0	0.48	0
3	0.096	0	0.442	0
4	0.008	0	0.422	0
5	0	0	0.421	0

KOHONEN SOM

- Topological-preserving map or self-organizing feature maps
- Winner-takes-all learning
- Winner only take place for winner
- Data mining and data exploration

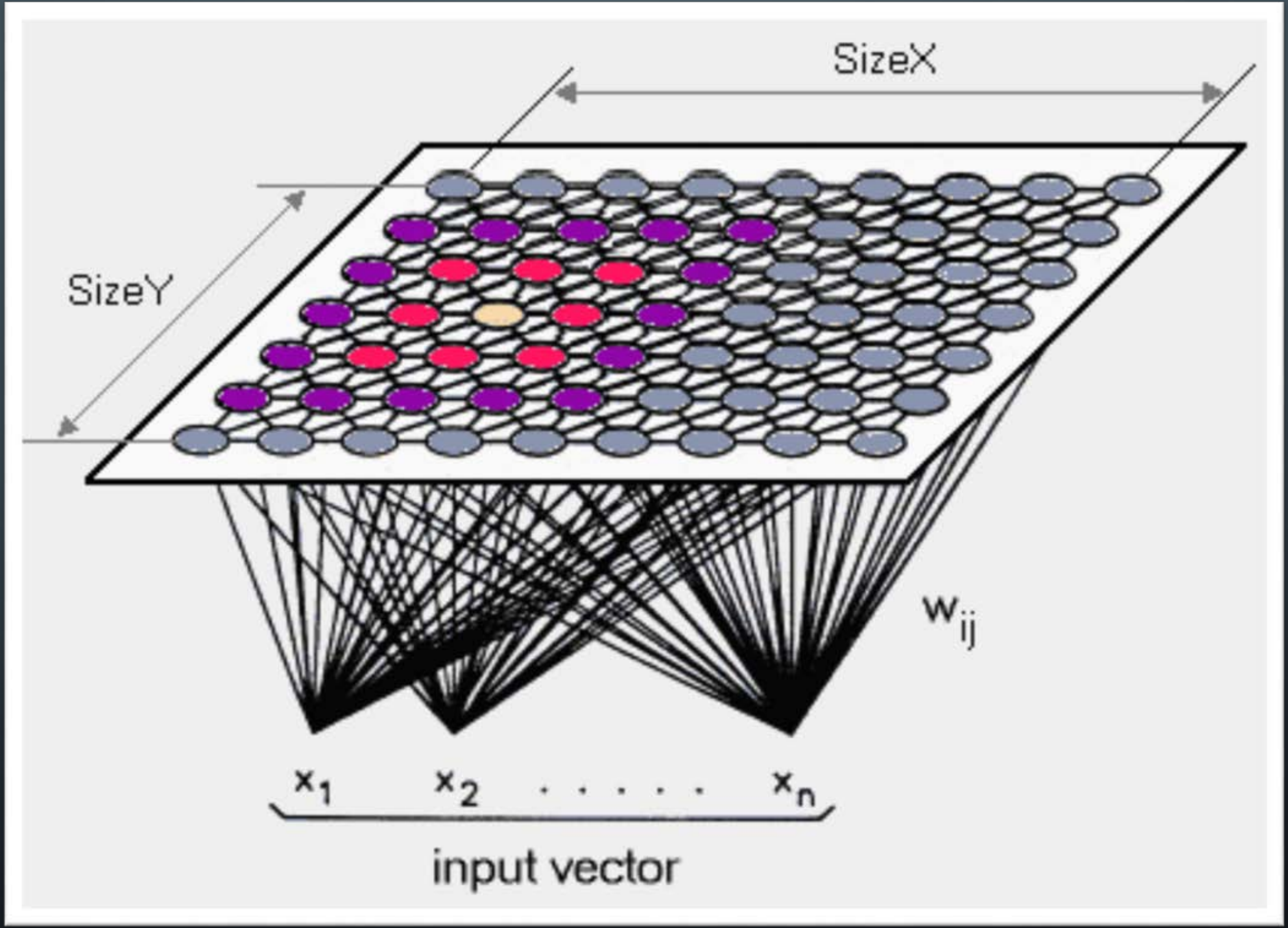
SOM ARCHITECTURE

- One input layer
- One output layer (representing classes)
- Fully connected

SOM TRAINING

- all weights are random at first
- Two phase unsupervised learning
  - Competing phase
    - Winner-take-all
  - Cooperation
    - Winner and neurons close to the winner
  - Rewarding phase
    - Update winners weights

SOM OUTPUT LAYER

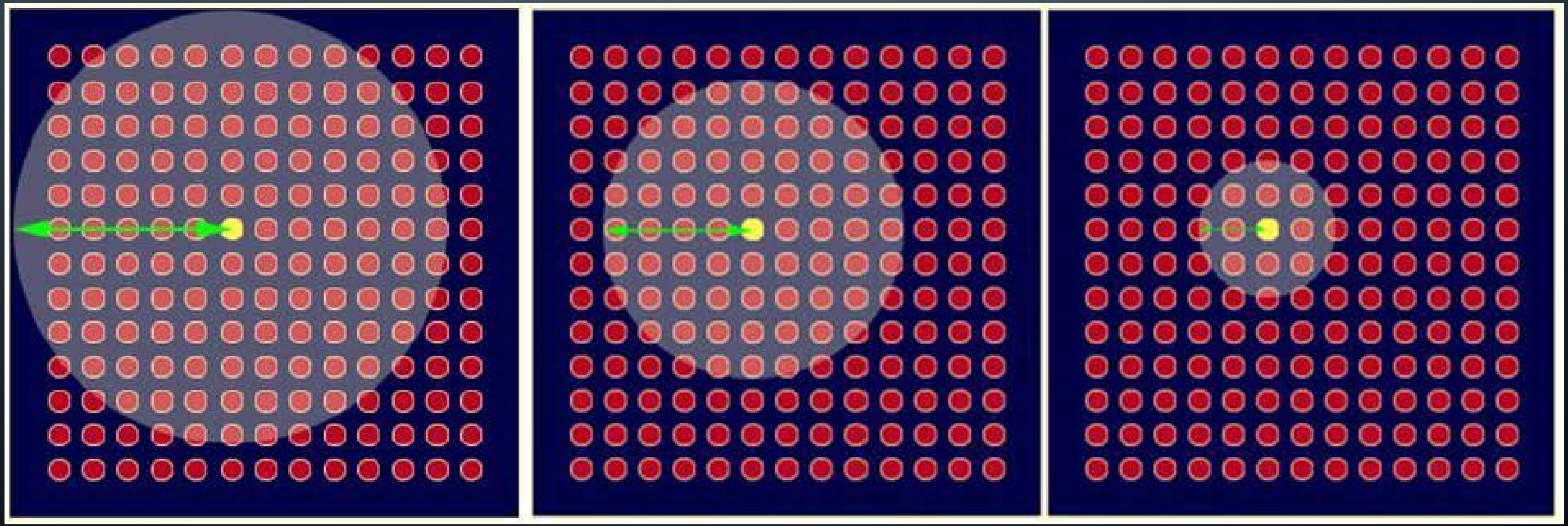


COMPETITION

- Input X
- Weight vector W_j for neuron j
- Select neuron with largest $W_j^T.X$
- Select neuron with min $||X \; - W_j||$

COOPERATION

- Using topological neighborhood centered on winner
- The neighborhood gets smaller and smaller over time



WEIGHT UPDATE

- Lerning rate * neighborhood degree * difference between weight and input

$$w_j(n+1) = w_J(n) + \eta(n)h_{j,i(x)}(n)[X - w_j(n)]$$

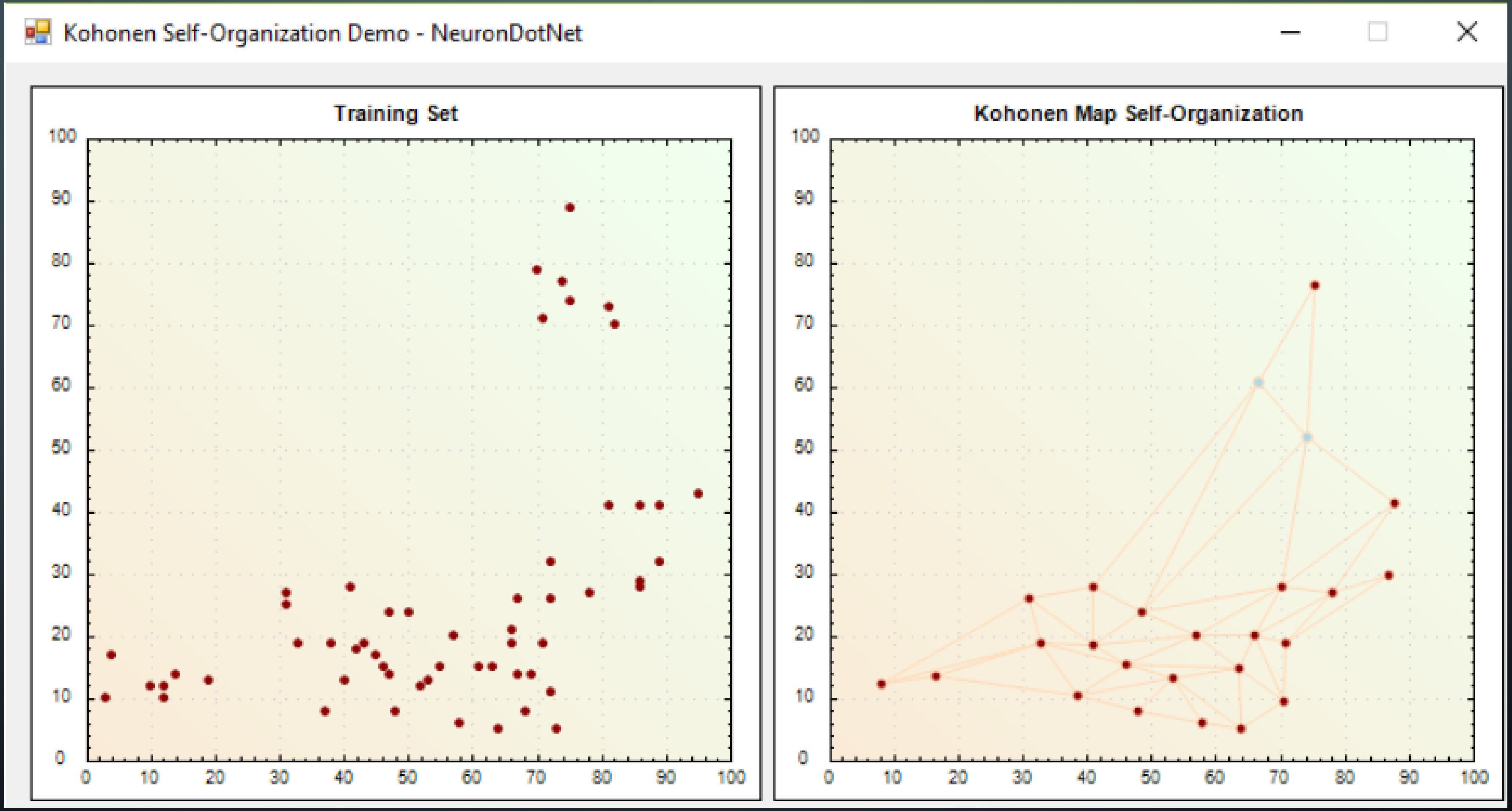
$$h_{j,i(x)}(n) = \exp(-\frac{d_{ij}^2}{2\sigma(n)^2})$$

$$\sigma(n) = \sigma_0 \exp(-\frac{n}{\tau_1}) \qquad \eta(n) = \eta_0 \exp(-\frac{n}{\tau_2})$$

SOM LEARNING PHASES

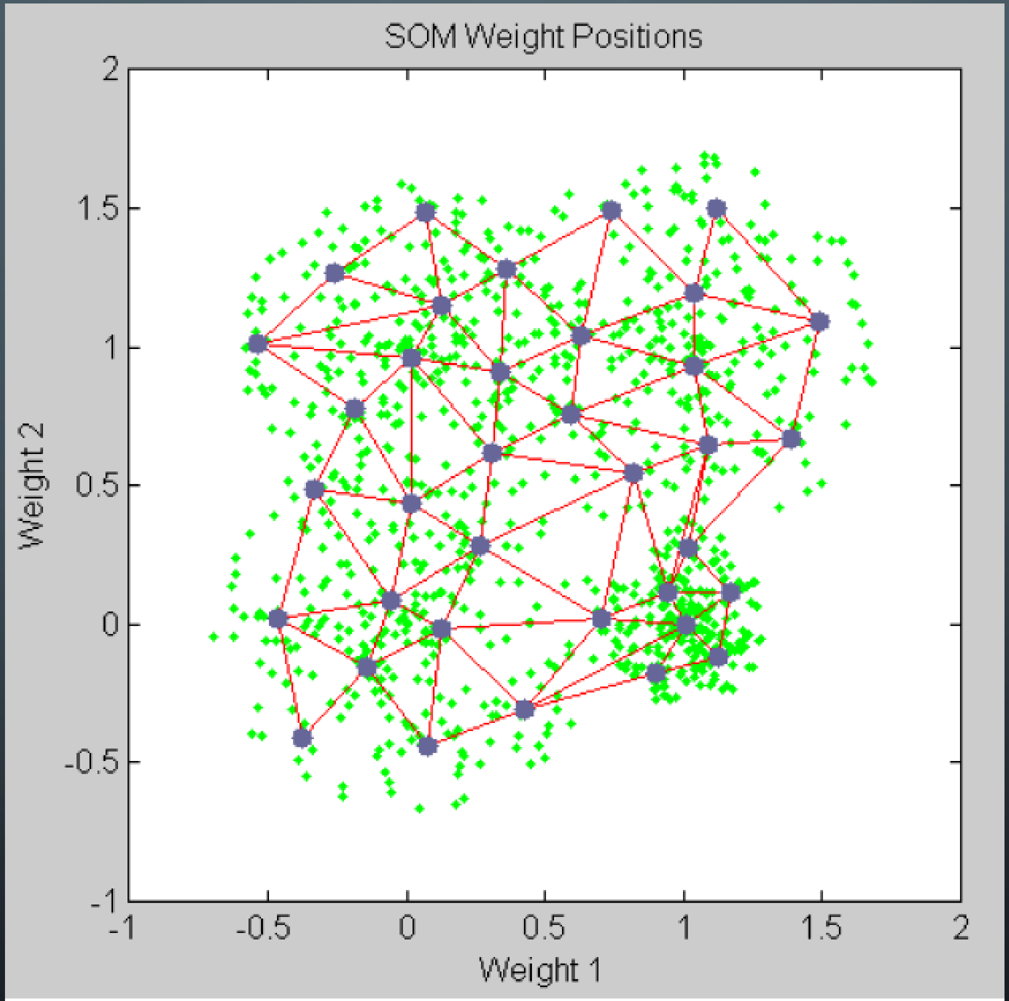
- Ordering phase
- Convergence phase

SOM EXAMPLE

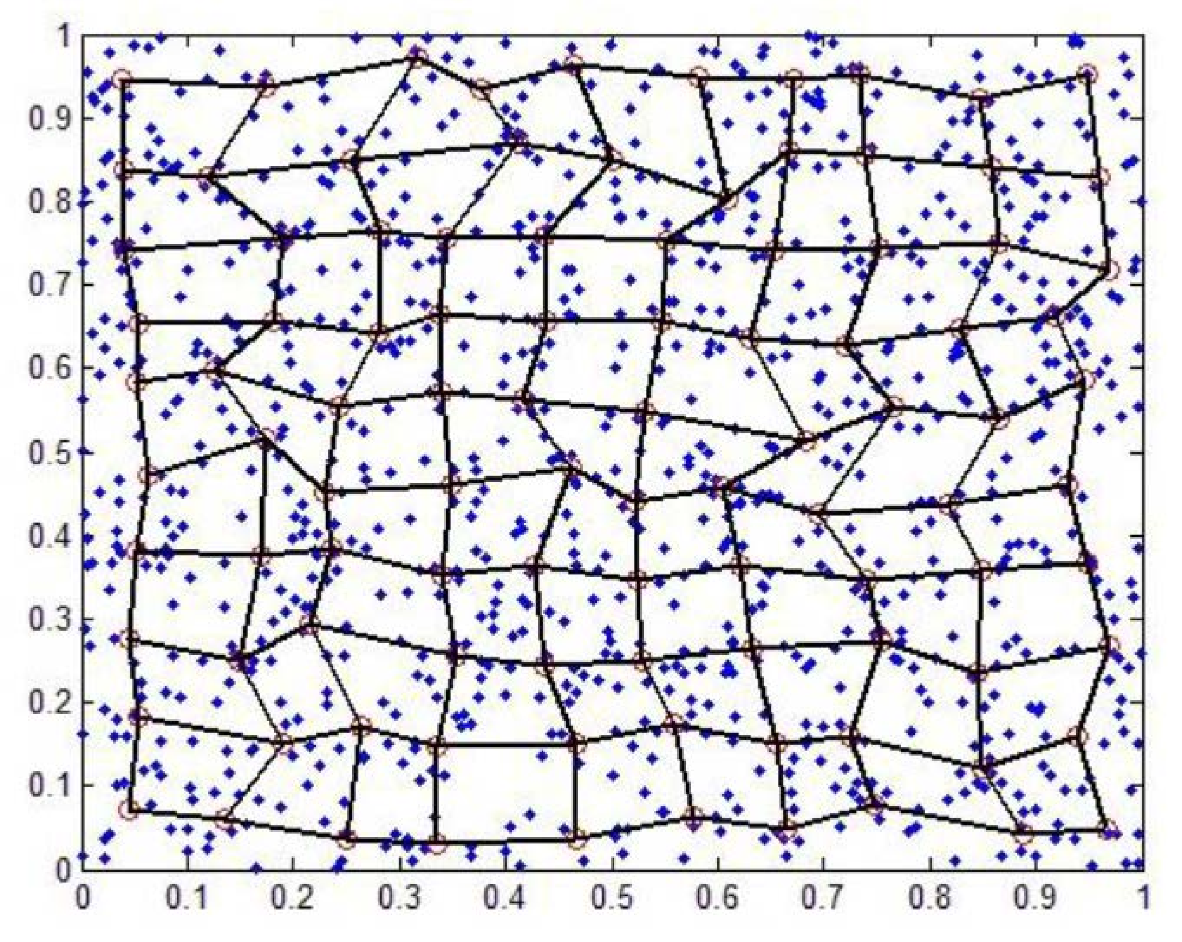


Kohonen Self-Organization Demo - NeuronDotNet
Training Set
Kohonen Map Self-Organization
0
10
20
30
40
50
60
70
80
90
100

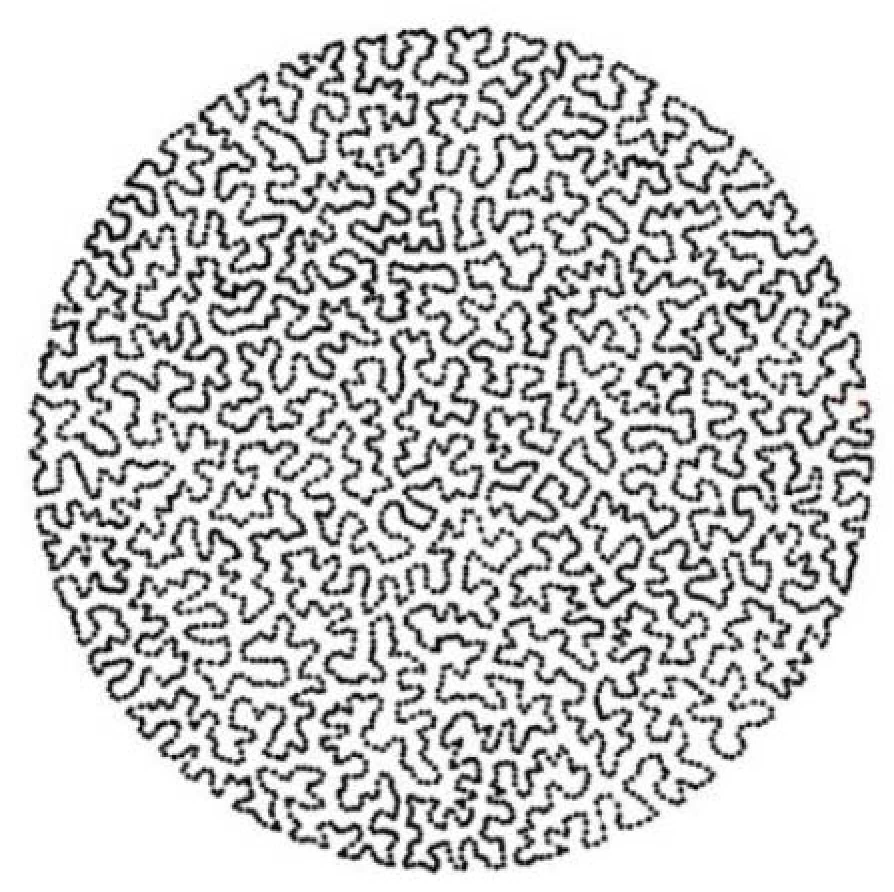
SOM EXAMPLE


SOM Weight Positions
Weight 2
Weight 1
2
1.5
1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1
1.5
2

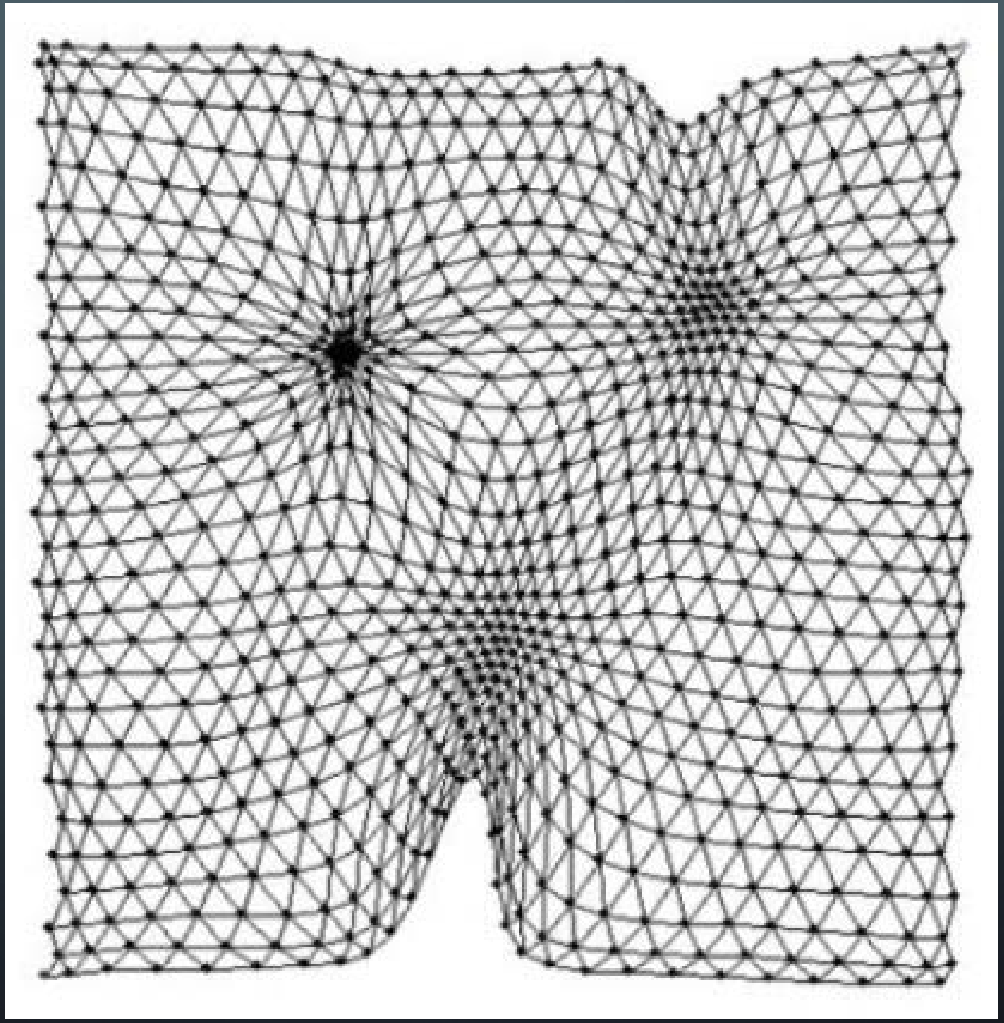
SOM EXAMPLE



SOM EXAMPLE



SOM EXAMPLE



SOM PROPERTIES

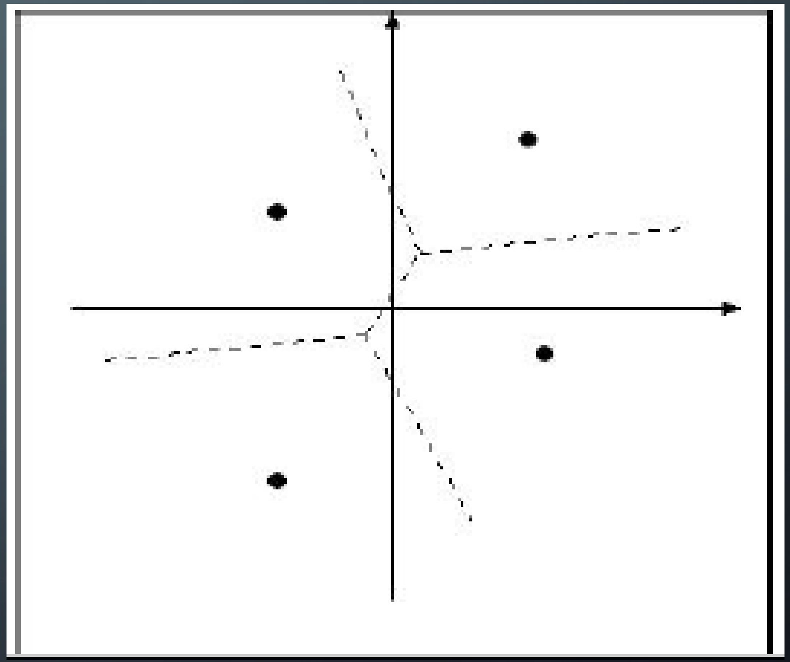
- Approximation of the Input Space
- Topological ordering
- Density matching
- Feature selection

LVQ

MOHAMMAD GHODDOSI

LEARNING VECTOR QUANTIZATION

- Competitive network
- Supervised classification
- Fixed output weights
- Input space is divided into regions



LVQ ARCHITECTURE

- One hidden layer
- Each hidden unit is assigned to a class

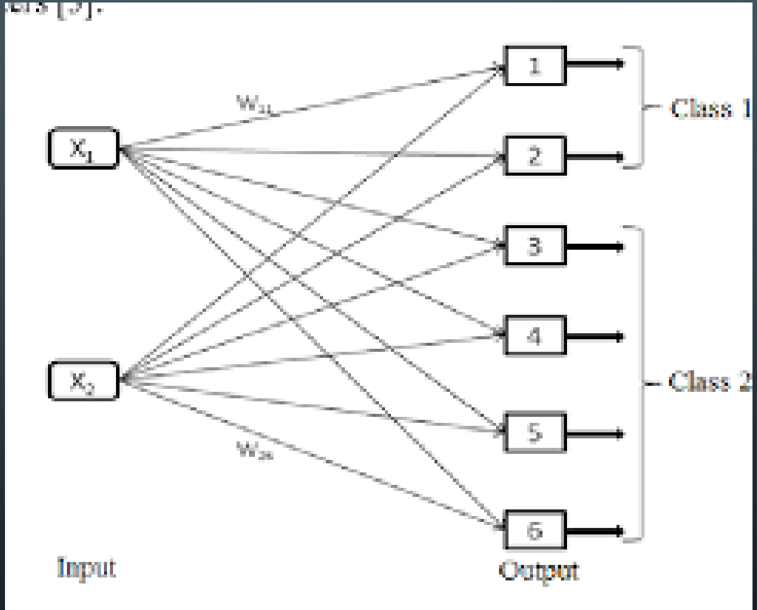


Fig. 2. Example of LVQ network.

LVQ – FIRST LAYER

- Competitive layer
- Euclidean distance
- Closest neuron to input vector is the winner
- Just winner drives learning rule

LVQ TRAINING

- If vector x is classified correctly, then the weight vector of the winner is moved towards x

$$\Delta \mathrm{w}_{\mathrm{j}^*} = \alpha(x - \mathrm{w}_{\mathrm{j}^*})$$

- If vector x is classified incorrectly, then the weight vector of the winner is moved away from x

$$\Delta \mathrm{w}_{\mathrm{j}^*} = -\alpha(x - \mathrm{w}_{\mathrm{j}^*})$$

SOM–LVQ

HOPFIELD

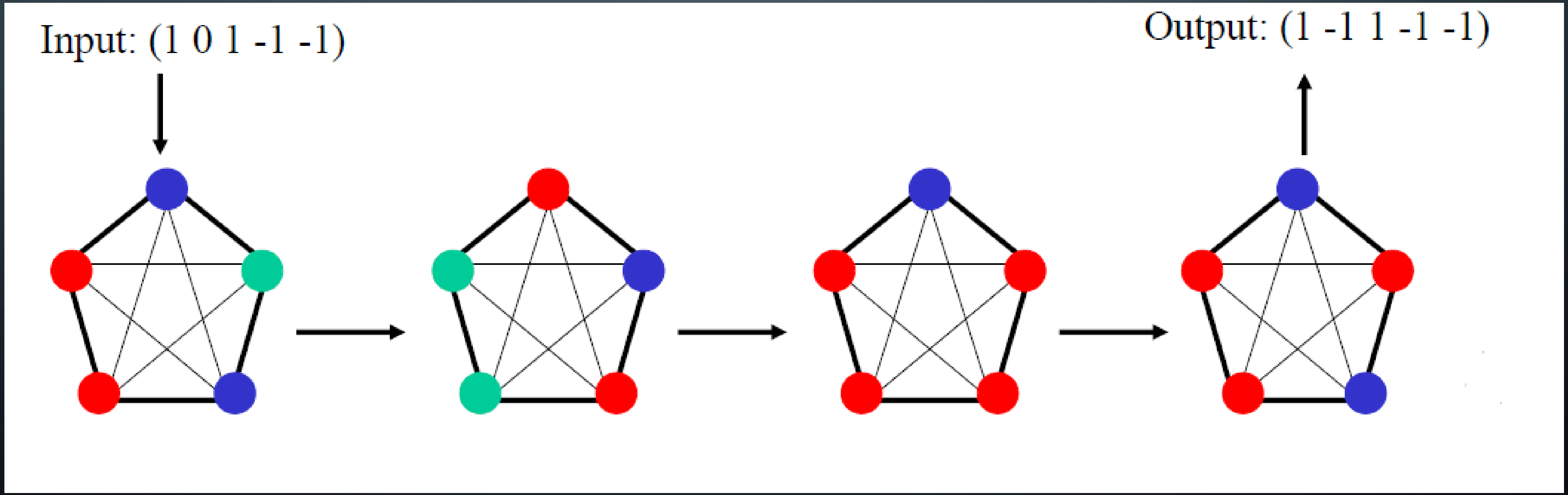
MOHAMMAD GHODDOSI

HOPFIELD

- Associative memory
- Robustness
- Input pattern A reminds the network to pattern B

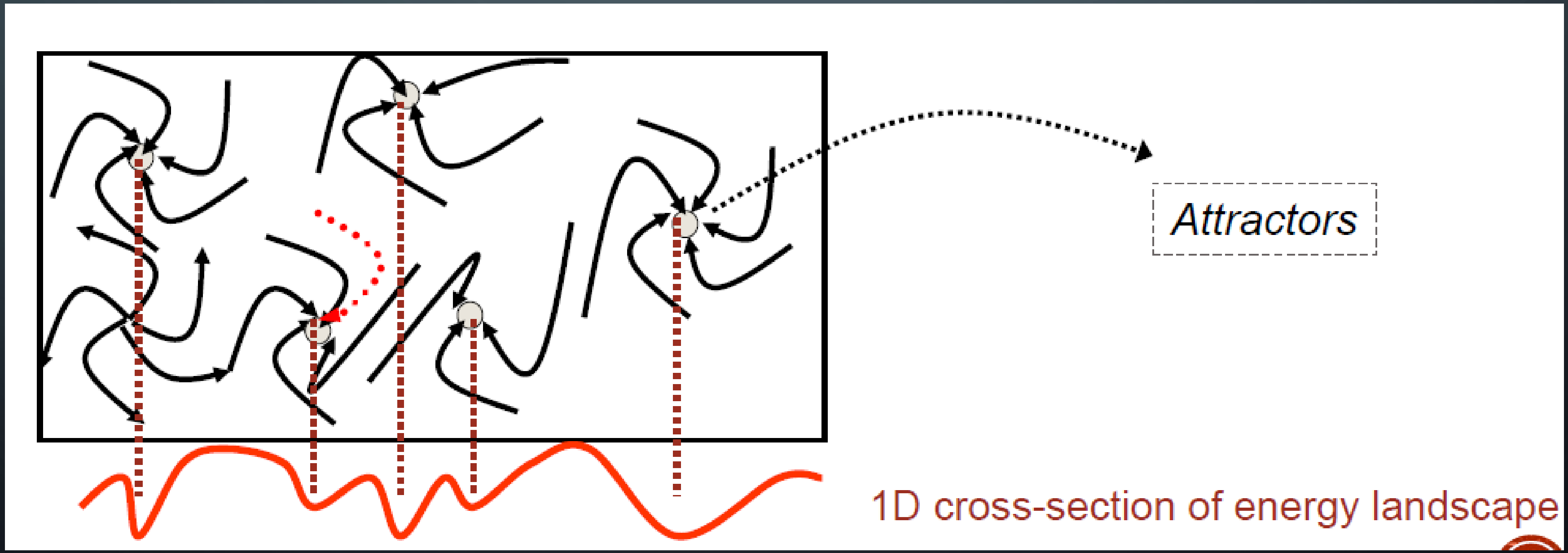
ASSOCIATIVE MEMORY NETWORK

- Input: noisy or corrupted pattern
- Output: corresponding pattern

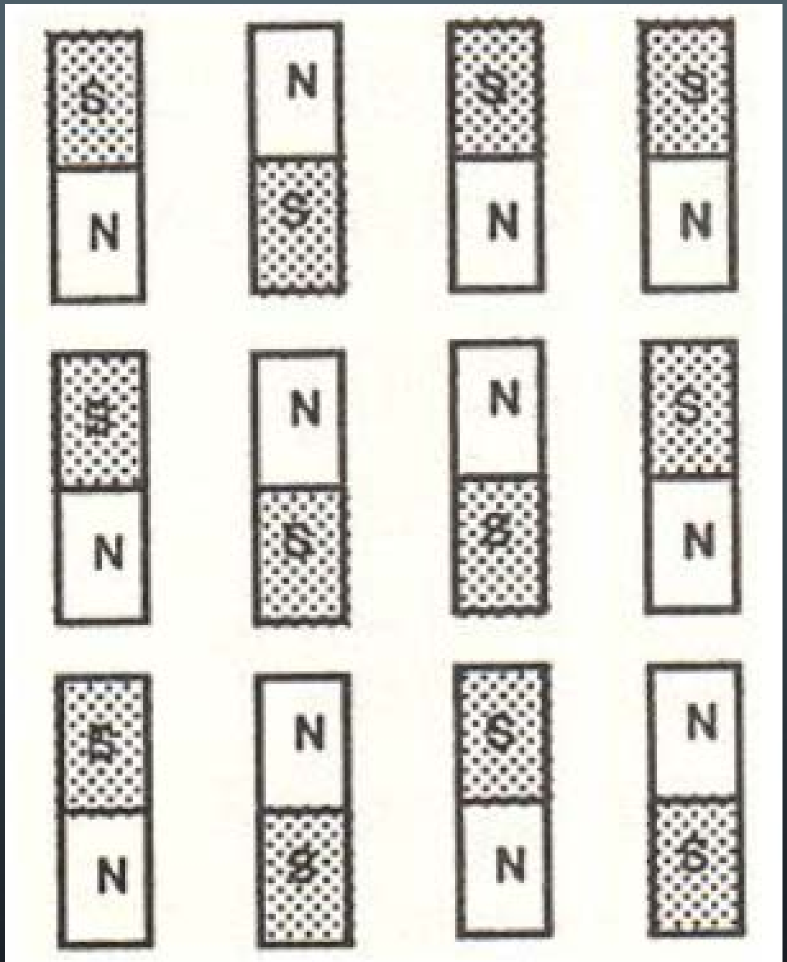


HOPFIELD MODEL

- Binary neurons
- Attractor model



EXAMPLE



HOPFIELD NETWORK

- Single layer recurrent network
- Fixed weights
- Fully connected
- All neurons act as input and output neurons
- Output of each neuron is 1 or −1

HOPFIELD WEIGHTS

- We use input-output pairs $(x_1, y_1), (x_2, y_2), (x_3, y_3), \dots (x_P, y_P)$

$$W_{ij} = \frac{1}{P} \sum_{k=1}^{P} y_k^{(i)} x_k^{(j)}$$

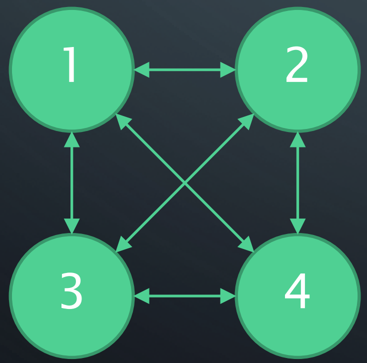
- X and y could be same

EXAMPLE (X=Y)

- We have 2 patterns

1	−1
1	−1

−1	1
−1	−1



$$w_{12} = w_{21} = \frac{-1-1}{2} = -1$$

$$w_{13} = w_{31} = \frac{1+1}{2} = 1$$

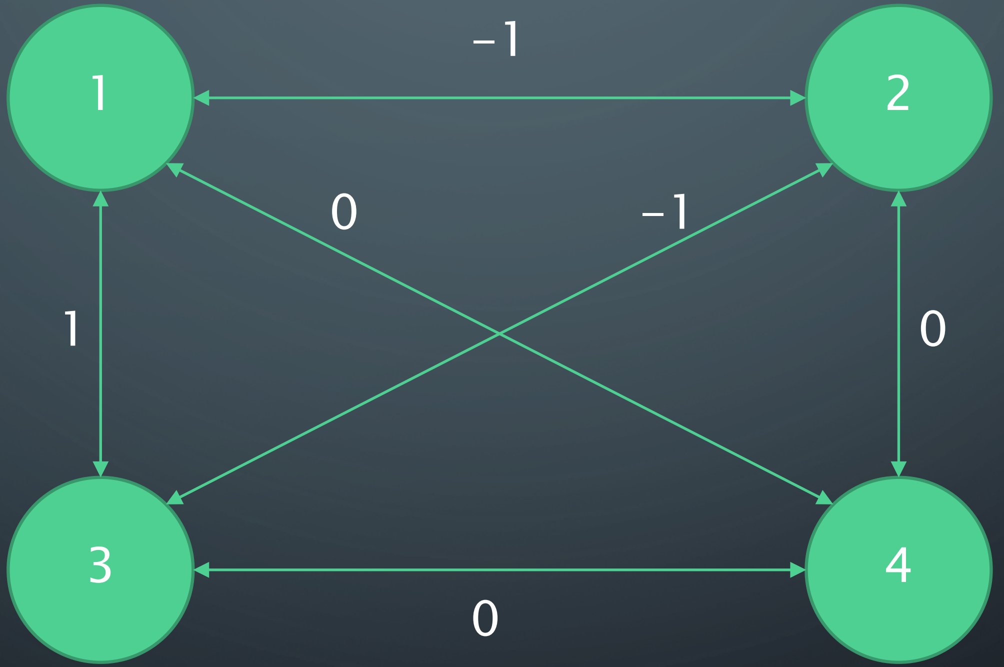
$$w_{14} = w_{41} = \frac{-1+1}{2} = 0$$

$$w_{23} = w_{32} = \frac{-1-1}{2} = -1$$

$$w_{24} = w_{42} = \frac{1-1}{2} = 0$$

$$w_{34} = w_{43} = \frac{-1+1}{2} = 0$$

EXAMPLE (X=Y)

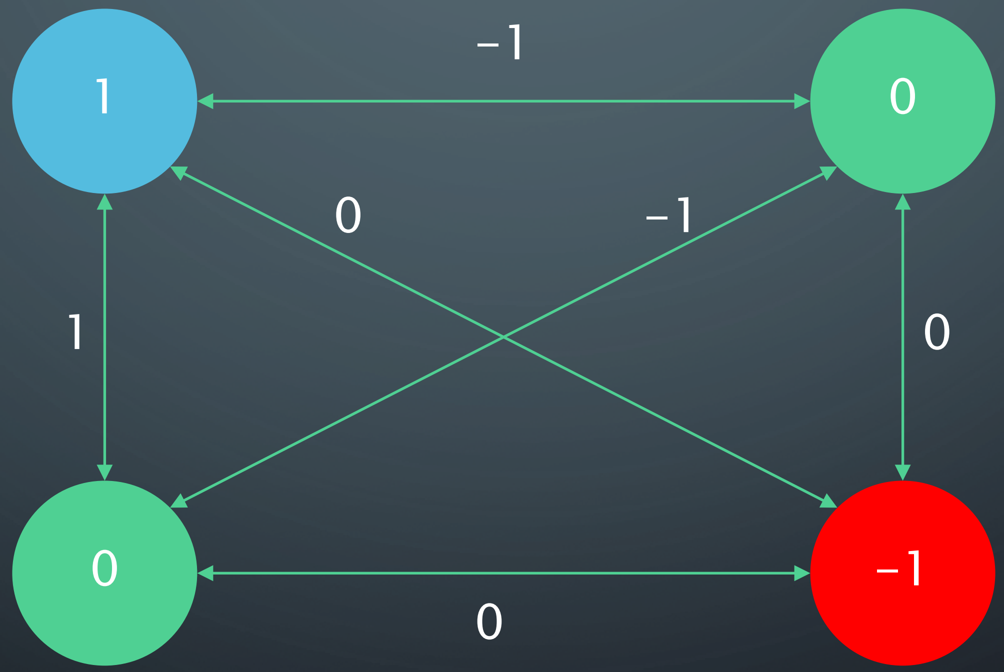


EXAMPLE (X=Y)

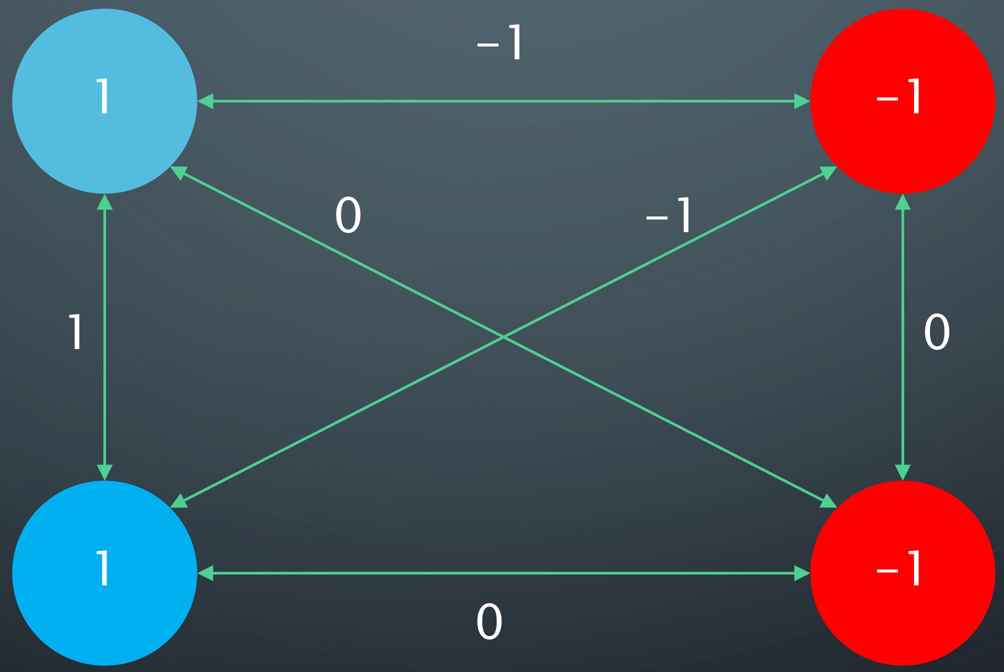
- We have an noisy input pattern

1	0
0	−1

EXAMPLE (X=Y)

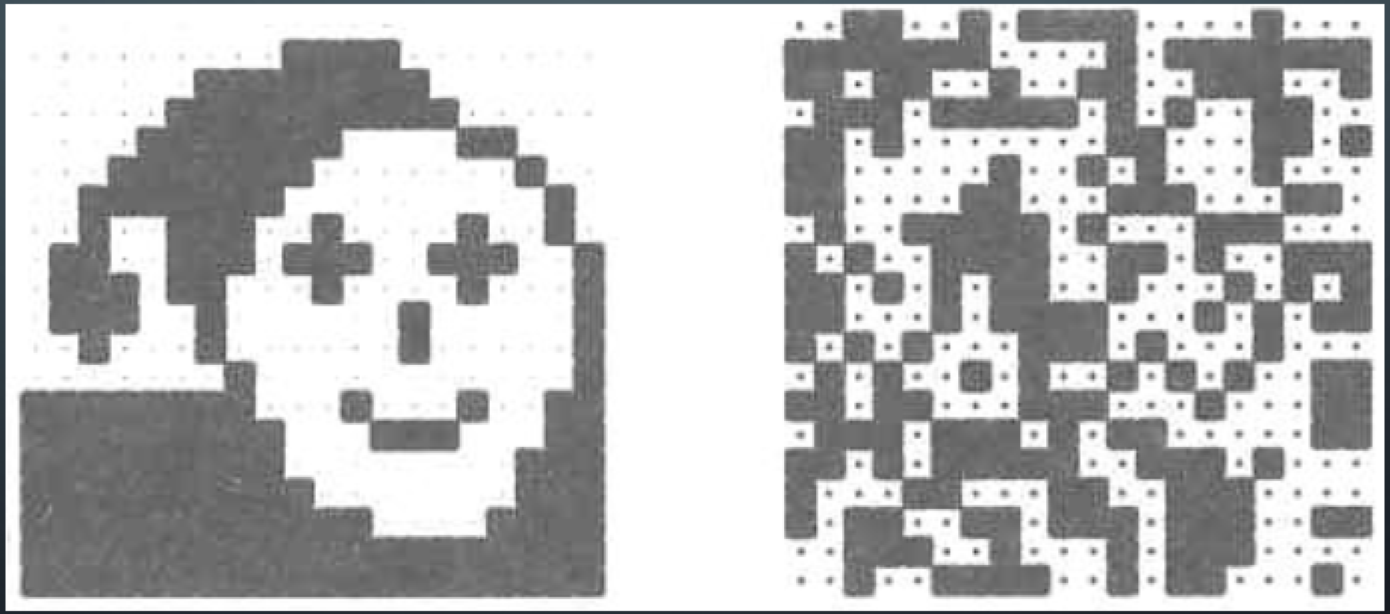


EXAMPLE (X=Y)



EXAMPLE

- One Hopfield model train with 1 face and 19 random patterns



EXAMPLE