



UNSUPERVISED LEARNING: COMPETITIVE LEARNING

INT301 Bio-computation, Week 12, 2021



Recall:

Simple Competitive Learning

- Update weights for winning neuron

$$\Delta w_{j*i} = \eta x_i$$

$$\Delta w_{j*i} = \eta' \left(\frac{x_i}{\sum_i x_i} - w_{j*i} \right)$$

$$\Delta w_{j*i} = \eta (x_i - w_{j*i})$$

1. Only the incoming weights of the winner node are modified.
2. Some units may never or rarely become a winner, and so weight vector may not be updated→**DEAD UNIT**



Leaky learning

- Modify weights of both winning and losing units but at different learning rates

$$w(t+1) = w(t) + \begin{cases} \eta_w (x - w(t)) \\ \eta_L (x - w(t)) \end{cases}$$

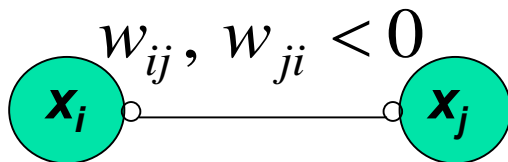
where $\eta_w \gg \eta_L$

which has the effect of slowly moving losing units towards denser regions pattern space.

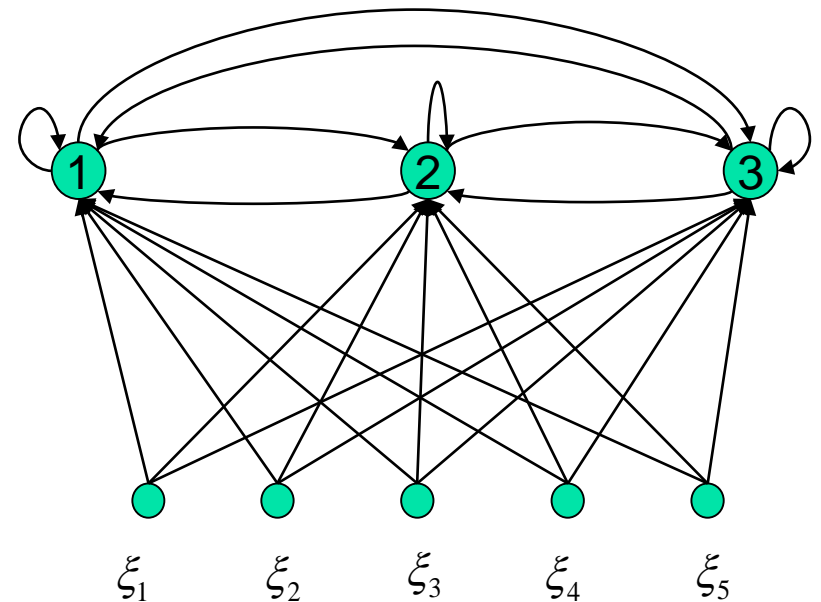
Maxnet

Lateral inhibition

output of each node feeds to others through inhibitory connections (with negative weights)



A specific competitive net that performs Winner Take All (WTA) competition is the **Maxnet**



Maxnet

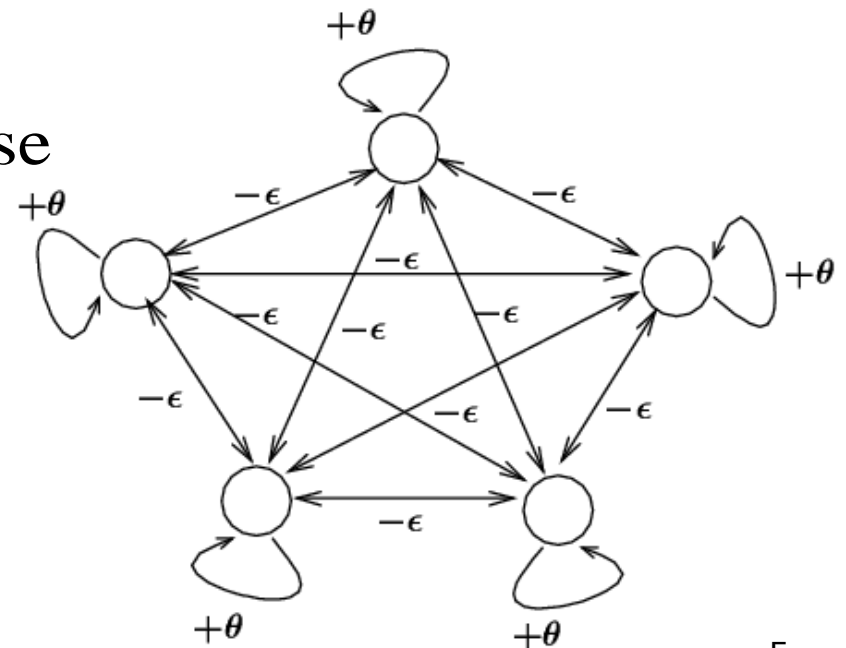
Maxnet

Lateral inhibition between competitors

$$\text{weights : } w_{ji} = \begin{cases} \theta & \text{if } i = j \\ -\epsilon & \text{otherwise} \end{cases}$$

node function :

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

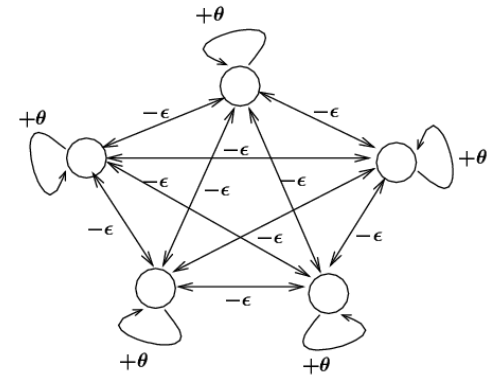


Maxnet

$$\text{weights : } w_{ji} = \begin{cases} \theta & \text{if } i = j \\ -\epsilon & \text{otherwise} \end{cases}$$

node function :

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$



Notes:

- Competition: iterative process until the net stabilizes (at most one node with positive activation)
 $0 < \epsilon < 1/m$, where m is the # of competitors
- ϵ too small: takes too long to converge
- ϵ too big: may suppress the entire network (no winner)

Mexican Hat Network

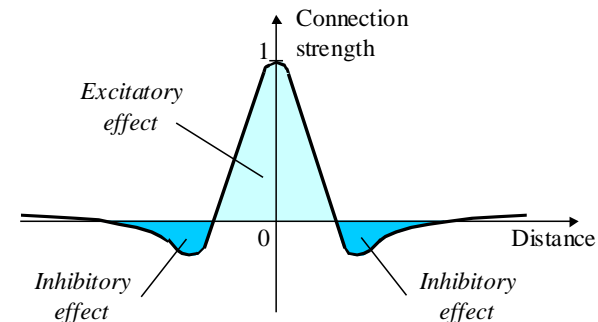
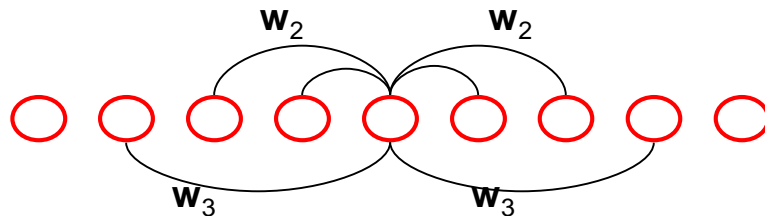


- Among all competing nodes, only one will win and all others will lose
 - The neuron with the largest activation level among all neurons in the output layer becomes the winner. This neuron is the only neuron that produces an output signal. The activity of all other neurons is suppressed in the competition.
- We mainly deal with single winner WTA, *but multiple winners WTA are possible*
 - The lateral connections produce excitatory or inhibitory effects, depending on the distance from the winning neuron
 - This is achieved by the use of a **Mexican Hat function**, which describes synaptic weights between neurons in the output layer.

Mexican Hat Network

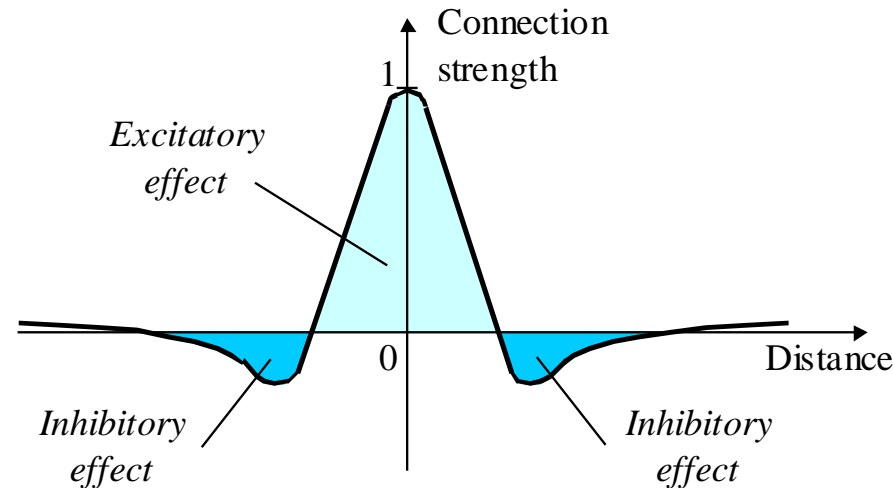


- **Architecture:** For a given node,
 - close neighbors: **cooperative** (mutually excitatory , $w > 0$)
 - distant neighbors: **competitive** (mutually inhibitory, $w < 0$)
 - too far away neighbors: irrelevant ($w = 0$)



- Need a definition of **distance (neighborhood)**:
 - one dimensional: ordering by index (1,2,...n)
 - two dimensional: lattice

Mexican Hat Network



■ Equilibrium:

- negative input = positive input for all nodes
- winner has the highest activation
- its cooperative neighbors all have positive activations
- its competitive neighbors all have negative (or zero) activations



Example

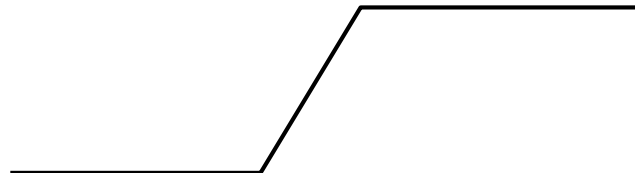
weights

$$w_{ij} = \begin{cases} c_1 & \text{if } \text{distance}(i, j) < k \ (c_1 > 0) \\ c_2 & \text{if } \text{distance}(i, j) = k \ (0 < c_2 < c_1) \\ c_3 & \text{if } \text{distance}(i, j) > k \ (c_3 \leq 0) \end{cases}$$

activation function

$$f(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } 0 \leq x \leq \max \\ \max & \text{if } x > \max \end{cases}$$

ramp function :





Vector Quantization

Application of competitive learning

■ Idea

- categorize a given set of input vectors into M classes using competitive learning algorithms
- represent any vector just by the class into which it falls

■ Important application of competitive learning (esp. in **data compression**)

- divides entire pattern space into a number of separate subspaces
- set of M units represent set of prototype vectors:
CODEBOOK
- new pattern x is assigned to a class based on its closeness to a prototype vector using Euclidean distances




Vector Quantization

- A codebook
 - a set of centroids/**codewords/codevector**):

$$\{m_1, m_2, \dots, m_K\}$$

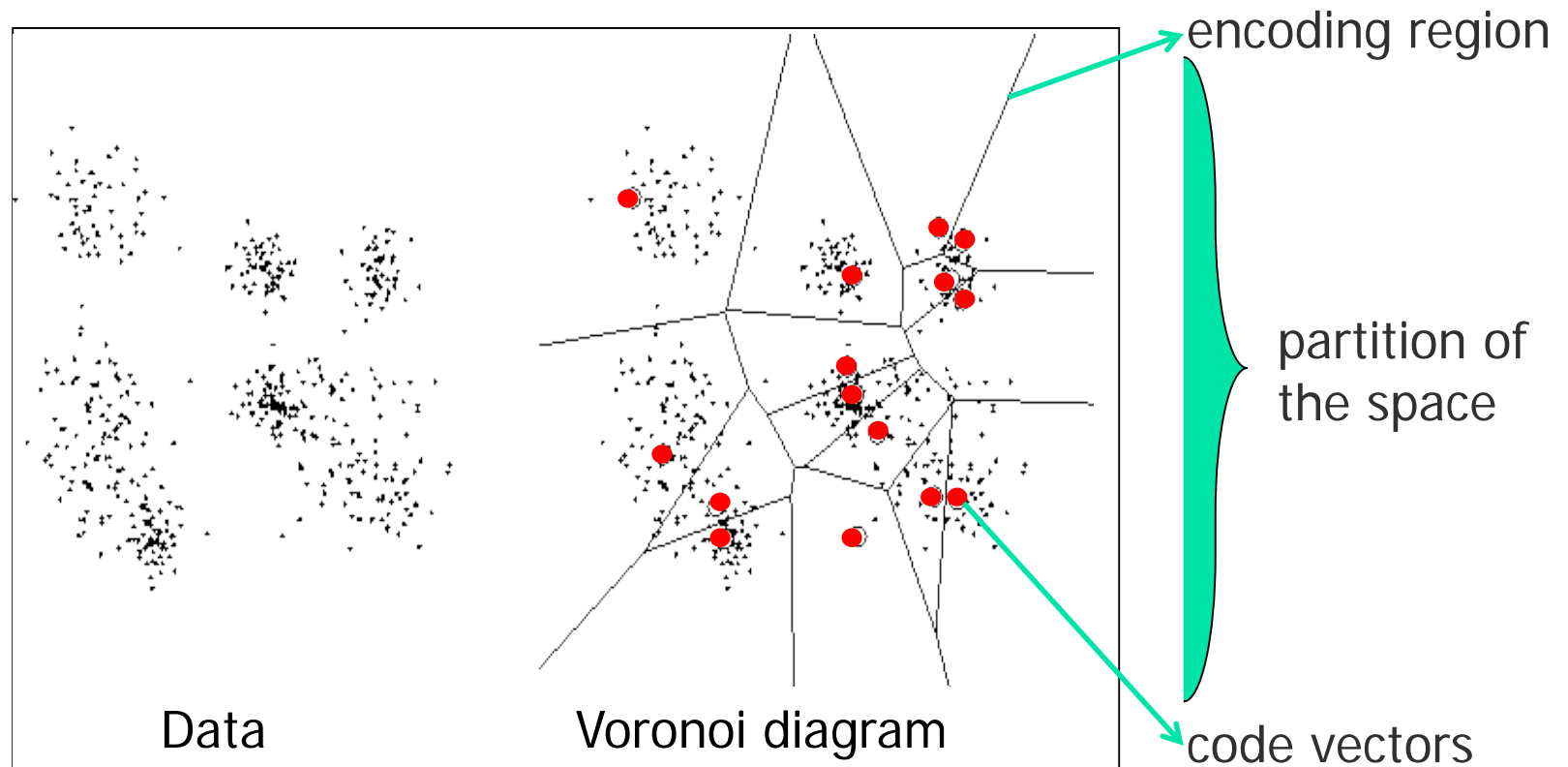
- A quantization function:


$$q(x_i) = m_k$$

Often, the nearest-neighbor function

- K-means can be used to construct the codebook

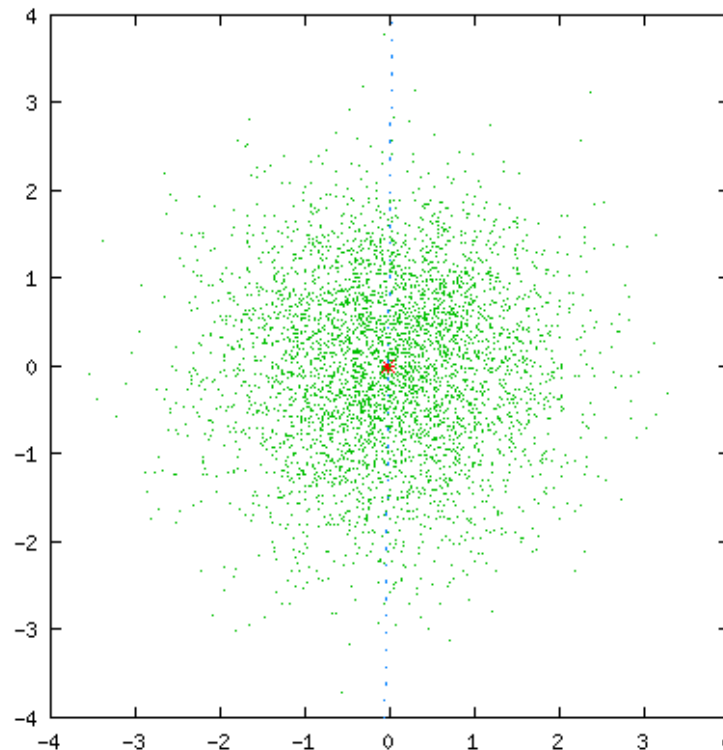
Vector Quantization



A Voronoi diagram is a partitioning of a plane into regions based on distance to points in a specific subset of the plane. For each seed there is a corresponding region consisting of all points closer to that seed than to any other. These regions are called Voronoi cells.

Vector Quantization

- LBG design algorithm for VQ
 - NOT within the scope of this module



Two-dimensional VQ animation



THANK YOU



VISIT US

WWW.XJTLU.EDU.CN



FOLLOW US

@XJTLU



Xi'an Jiaotong-Liverpool University

西交利物浦大學