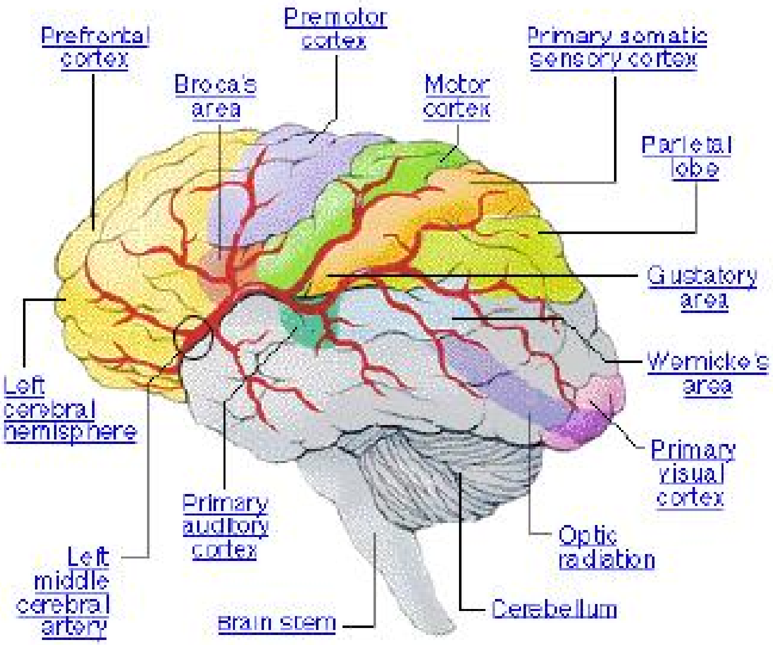
SELF-ORGANIZING FEATURE MAP

INT301 Bio-computation, Week 11, 2025

Self-Organizing Map – Biological Motivation

- Brain is a self-organizing system that can learn by itself by changing (adding, removing, strengthening) the interconnections between neurons.
- Neurons with similar functions are grouped together.



Self-Organizing Map – Biological Motivation

- Neurons with similar functions are grouped together.
- The brain processes multidimensional signals from the external world in a “2”-dimensional internal map.

THE JOURNAL OF COMPARATIVE NEUROLOGY 191:255-281 (1980)

Two-Dimensional Maps of the Cerebral Cortex

D. C. VAN ESSEN AND J. H. R. MAUNSELL
Division of Biology, California Institute of Technology, Pasadena, California 91125

Feature Maps

- Result of the brain’s self-organization
  - formation of feature maps in the brain that have a linear or planar topology (that is, they extend in one or two dimensions)

- Examples:
  - tonotopic map - sound frequencies are spatially mapped into regions of the cortex in an orderly progression from low to high frequencies.
  - retinotopic map - visual field is mapped in the visual cortex with higher resolution for the centre of the visual field
  - somatosensory map - mapping of touch

Feature Maps

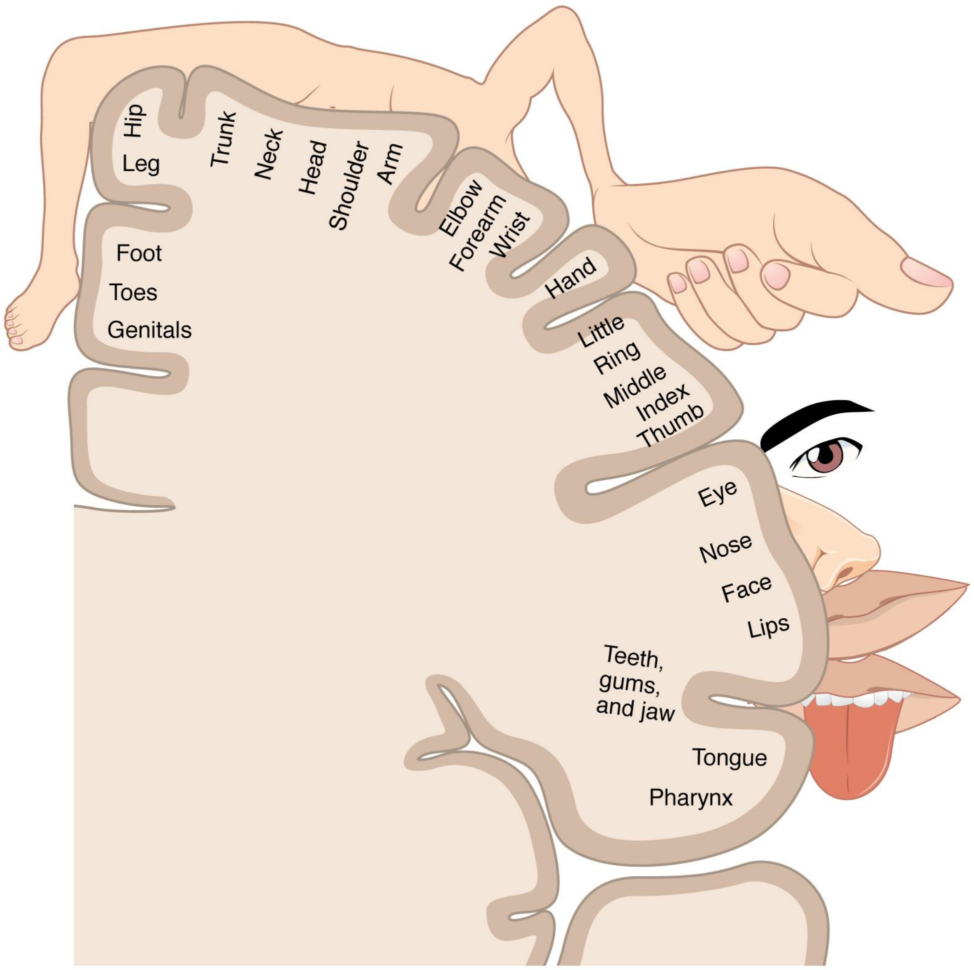
- Sensory experience is multidimensional
  - E.g. sound is characterised by pitch, intensity, noise…
- The brain maps the external multidimensional representation of the world into a similar 1 or 2 dimensional **internal representation**.
- That is, the brain processes the external signals in a **topology-preserving way.**
- So, if we are to have a hope of mimicking the way the brain learns, our system should be able to do the same thing.

Topographic Maps

- Extend the ideas of competitive learning to incorporate the neighborhood around inputs and neurons
- We want a nonlinear transformation of input pattern space onto output feature space which preserves neighbourhood relationship between the inputs
  - A feature map where nearby neurons respond to similar inputs
  - Neurons selectively tune to particular input patterns in such a way that the neurons become ordered with respect to each other so that a meaningful coordinate system for different input features is created

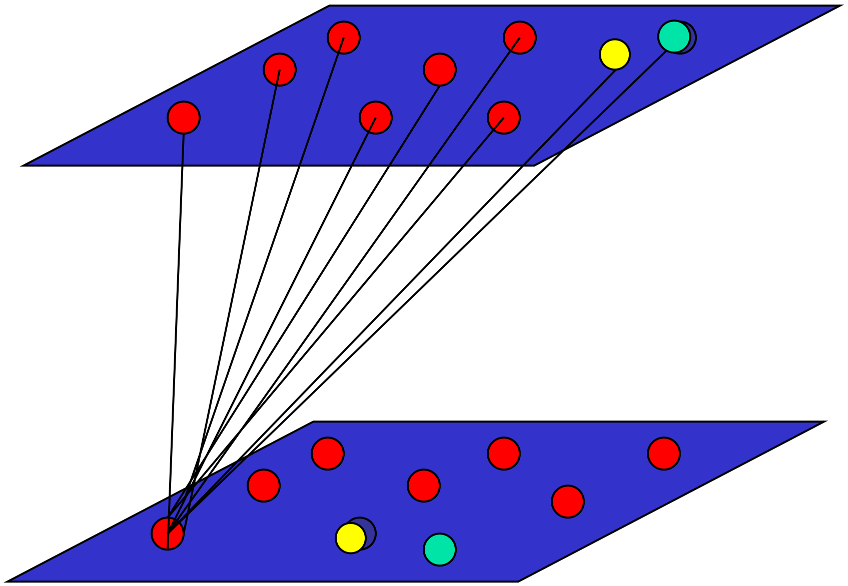
Topographic Maps

- E.g. the cortical homunculus, a map of somatosensory areas of the brain



Topographic Maps

- Spatial locations are indicative of the intrinsic statistical features of the input patterns: i.e., close in the input → close in the output



Topographic Maps

- Activity-based self-organization (von der Malsburg)
- Incorporation of competitive and cooperative mechanisms to generate feature maps using unsupervised learning networks

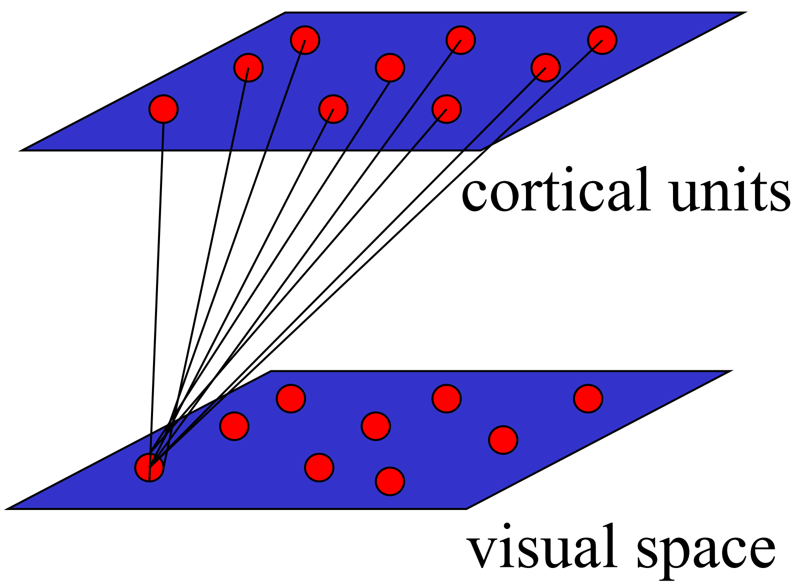
Proc. R. Soc. Lond. B. **194**, 431–445 (1976)
Printed in Great Britain

How patterned neural connections can be set up by self-organization

BY D. J. WILLSHAW AND C. VON DER MALSBURG
Max-Planck-Institut für Biophysikalische Chemie, Abteilung Neurobiologie, 3400 Göttingen, B.R.D.

Topographic Maps

- Biologically motivated: how can activity-based learning using highly interconnected circuits lead to orderly mapping of visual stimulus space onto cortical surface?
- 2 layer network each cortical unit fully connect to visual space via Hebbian units
- Interconnections of cortical units described by 'Mexican-hat' function: short-range excitation and long-range inhibition



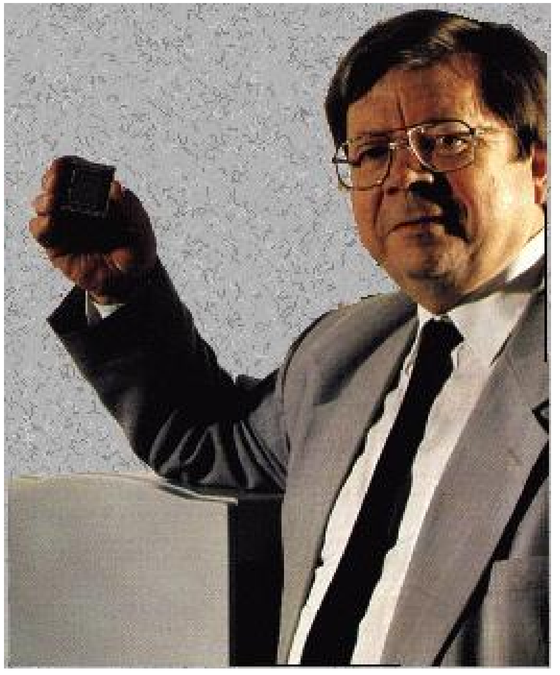
Topographic Maps

- Activity-based self-organization (von der Malsburg): after learning, a topographic map appears. However, input dimension is the same as output dimension

- Kohonen simplified this model and called it Kohonen's self-organizing map (SOM) algorithm
  - more general as it can perform **dimensionality reduction**
  - SOM can be viewed as a **vector quantization** type algorithm

Self-Organizing Map (SOM)

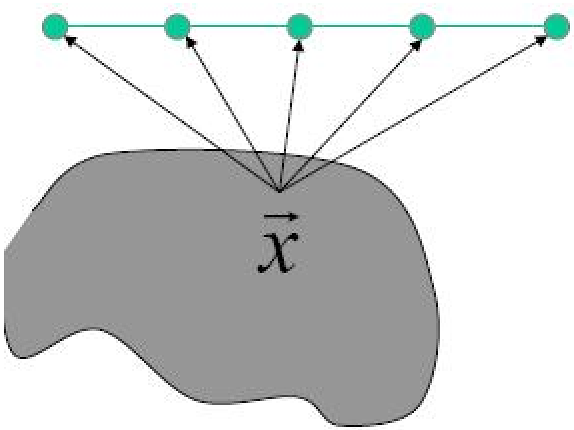
- a.k.a. as Kohonen networks
  - represents the embodiment of the ideas we have discussed so far
  - named after Dr. Eng. Teuvo Kohonen, Helsinki Uni of Technology

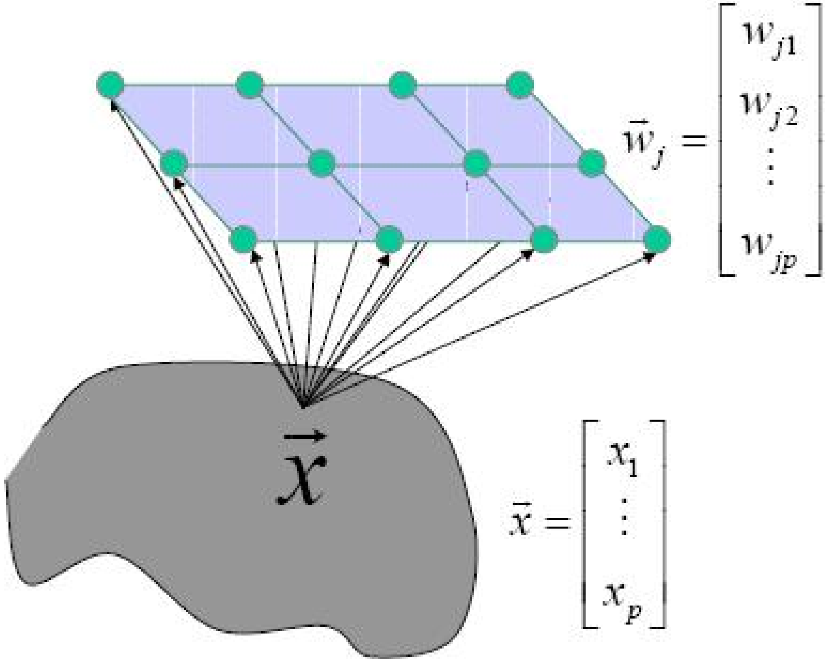


SOM

- The idea in an SOM is to transform an input of arbitrary dimension into a 1 or 2 dimensional discrete map

Two possible architectures





SOM

- The idea in an SOM is to transform an input of arbitrary dimension into a 1 or 2 dimensional discrete map

- Again, 2 layers of neurons with all inputs connecting to each output. Output neurons are held in a one or (usually) 2D lattice, where position in the lattice defines the distance between the neurons

SOM

- Once weights of net initialized, algorithm comprises 3 processes:
- 1. Competition
  - Given an input pattern, outputs compete to see who is winner based on a discriminant function (e.g. similarity of input vector and weight vector)
- 2. Cooperation
  - Winning neuron determines spatial location of a topological neighborhood within which output neurons excited
- 3. Synaptic Adaptation
  - Excite neurons adapt weights so that value of discriminant function increases (a similar input would result in enhanced response from winner)

SOM Training Algorithm

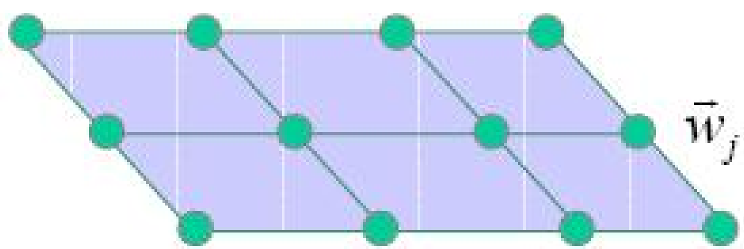
- Competition
- Cooperation
- Synaptic Adaptation

Learning Principle

Competitive learning where winning "spills over" to neighbors

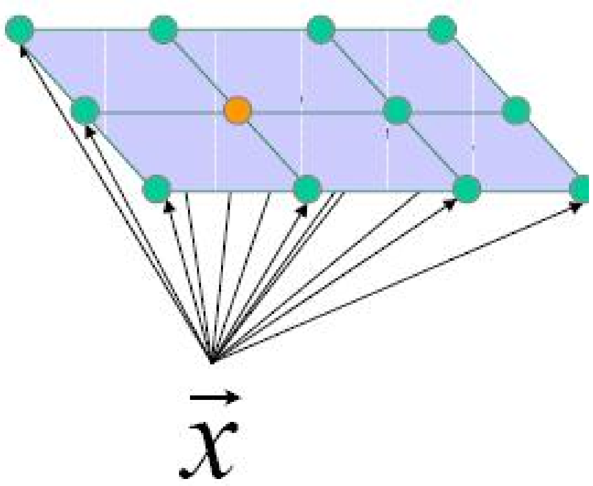
Initialization

Grid: size and structure fixed a priori (most of the times, 2-dimensional grid are used)



$$\vec{w}_j = \begin{bmatrix} w_{j1} \\ w_{j2} \\ \vdots \\ w_{jp} \end{bmatrix}$$

Competitive Process



Winner neuron

(1) $j^* = argmax_j(w_j^T x)$

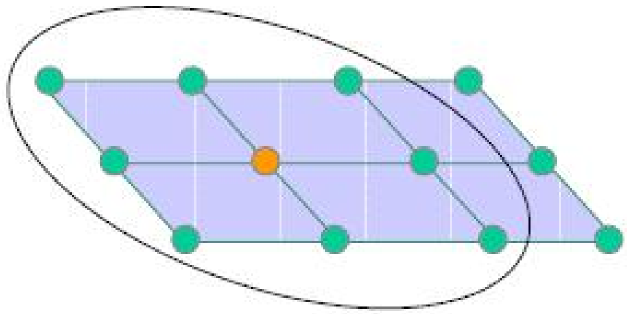
(2) $j^* = argmin_j(\| x - w_j \|)$

A continuous input space of activation patterns is mapped onto a discrete output space of neurons by a process of competition among the neurons in the network.

Cooperative Process

"The winning neuron locates the center of a topological neighborhood of cooperating neurons"

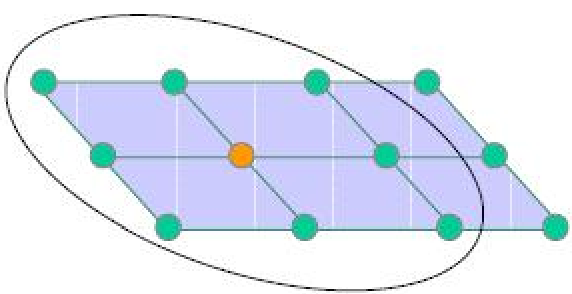
"… a neuron that is firing tends to excite the neurons in its immediate neighborhood more than those further away from it …"



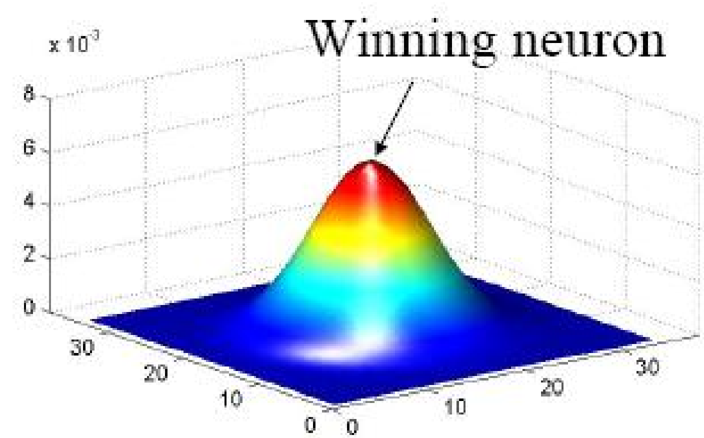
Cooperative Process

The topological neighborhood $h_{j,i}$

- symmetric around the winning neuron and achieve its maximum value at the winning neuron
- the amplitude decreases monotonically with the increasing lateral distance



$$h_{j,i} = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2} \right)$$



Cooperative Process

- The topological **neighborhood** $h_{j,i}$ *shrinks with time*

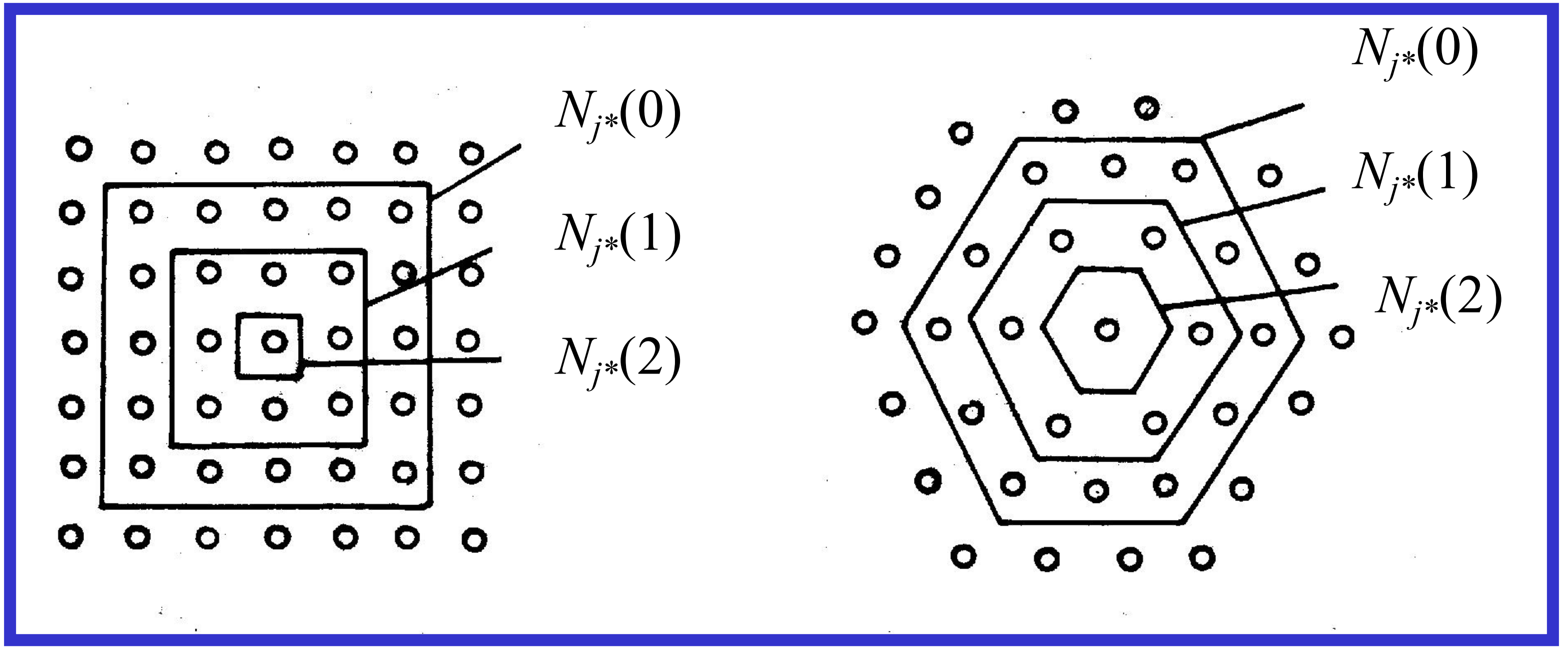
$$\sigma(t) = \sigma_0 \exp\left(-\frac{t}{\tau_1}\right) \qquad h_{j,i}(t) = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2(t)}\right)$$

- **Neighbors** of the winning node are also allowed to update, even if they are not close to winning!

Neighborhood

- Large neighborhood
  - Good global ordering
  - Bad local fit
- Small neighborhood
  - Bad global ordering
  - Good local fit
- By gradually shrinking the neighborhood we can get the best of both!
  - Ordering phase (large neighborhood)
  - Convergence phase (small neighborhood)

Neighborhood



Using a planar array of neurons with rectangular or hexagonal neighborhoods, an input vector **x** is applied simultaneously to all nodes.

Adaptive Process

Update the weights in relation to the inputs

$$\mathrm{w}_j(t+1) = \mathrm{w}_j(t) + \boxed{\eta(t)}\boxed{h_{j,i}}(t)(\mathrm{x} - \mathrm{w}_j(t))$$

Learning rate

$$\eta(t) = \eta_0 \exp\left(-\frac{t}{\tau_1}\right)$$

Neighborhood function

$$h_{j,i}(t) = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2(t)}\right)$$

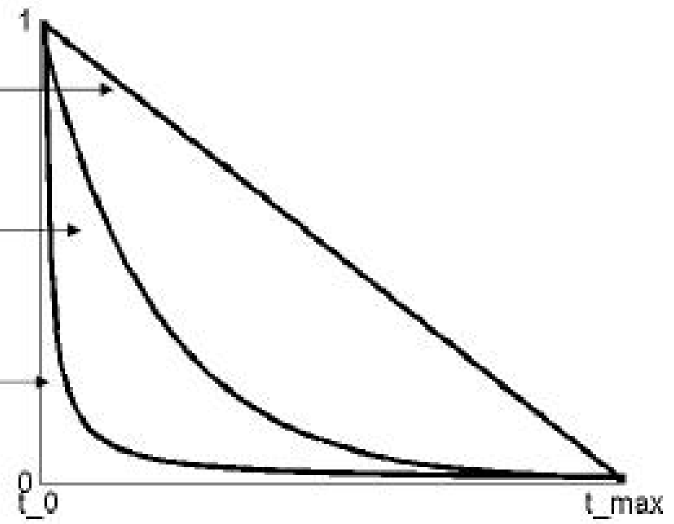
Learning Rate

Possible options:

$$\eta(t) = -at + b$$

$$\eta(t) = \exp(-at) + b$$

$$\eta(t) = \frac{1}{at} + b$$



Summary

Initialization

Draw an input sample x

Winner neuron j*

$$argmax_j(\, w_j^T x) \;\; or \;\; argmin_j(\, \| x - w_j \| \,)$$

Update

$$\mathrm{w}_j(t+1) = \mathrm{w}_j(t) + \eta(t) h_{j,i}(t)(\mathrm{x} - \mathrm{w}_j(t))$$

$$h_{j,i}(t) = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2(t)}\right) \qquad \eta(t) = \eta_0 \exp\left(-\frac{t}{\tau_1}\right)$$

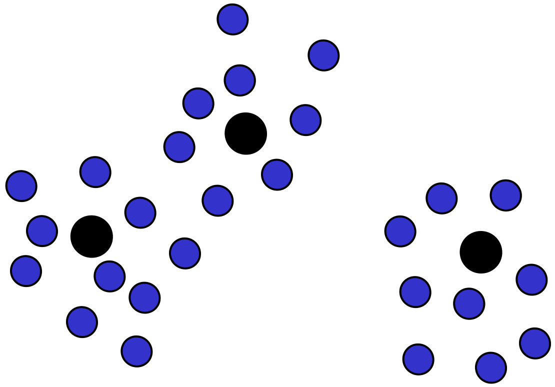
Property of SOM

- Approximate the input space
- Topological ordering
- Density matching
- Feature selection (features of the underlying distribution)

Recall:

Unsupervised Competitive Learning

- initialize K prototype vectors
- present a single example
- identify the closest prototype, i.e., the so-called ***winner***
- move the winner even closer towards the example

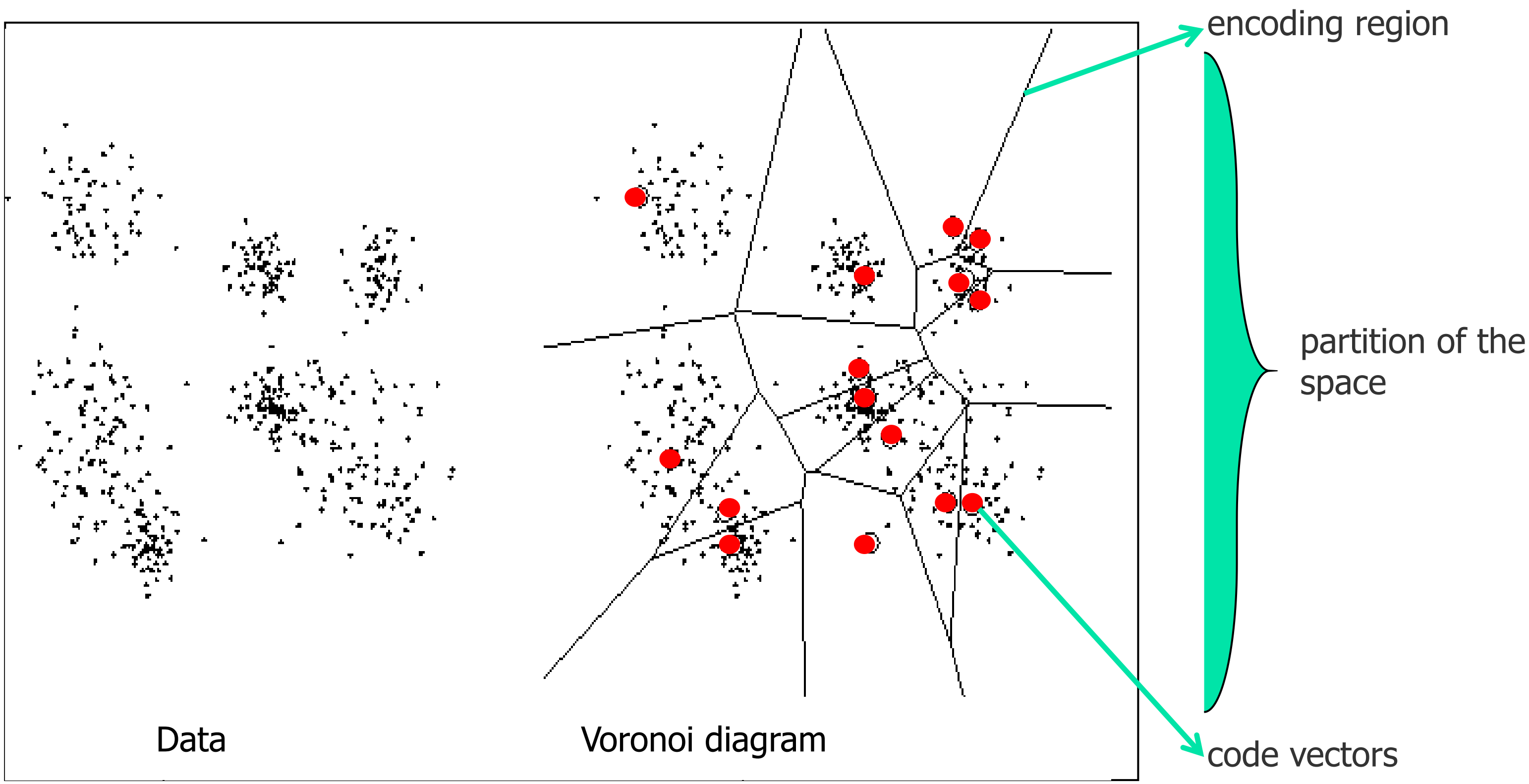


Intuitively clear, plausible procedure

- places prototypes in areas with high density of data
- identifies the most relevant combinations of features

Recall:

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.

More on Vector Quantization

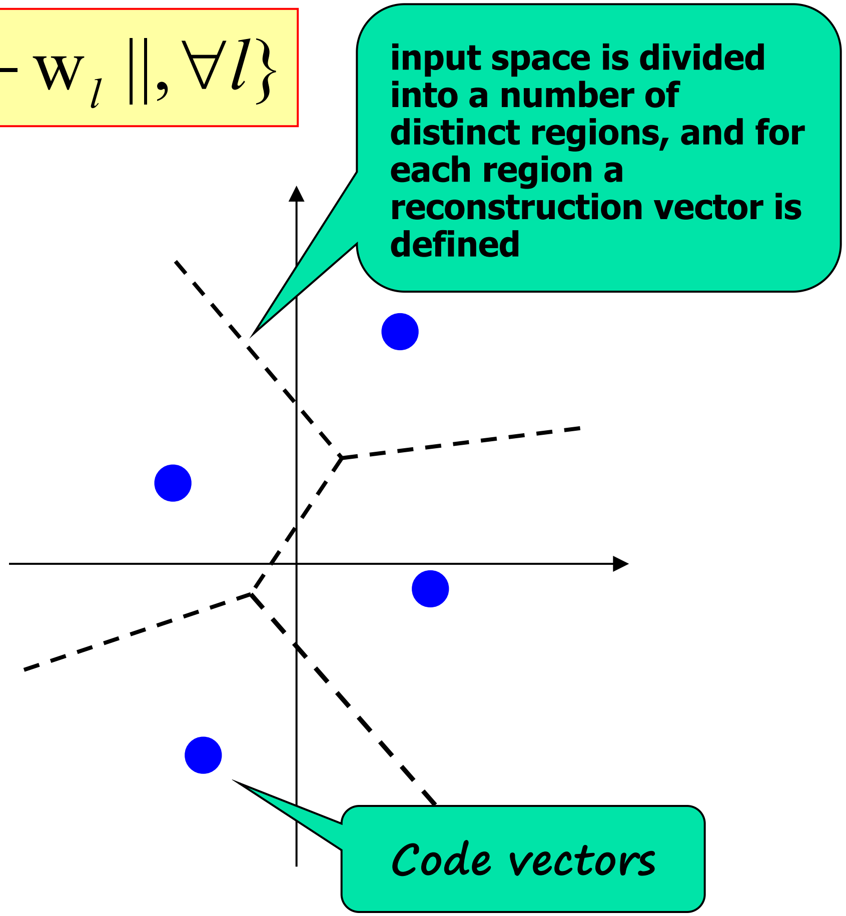
- In VQ techniques, a number of local Voronoi centers are formed to **represent input vectors**.
- For a set of *M* reference vectors, $\{\mathbf{w}_1,\ldots,\mathbf{w}_M\}$, an input vector $\mathbf{x}$ is considered being *best matched* by $\mathbf{w}_k$ in the sense that an appropriately defined distortion measure such as the squared Euclidean distance $||\mathbf{x}-\mathbf{w}_k||^2$ is minimal.
- The reference vectors partition the input space R^L into the Voronoi cells/polygons defined as

$$V_k = \{\mathrm{x} \in R^L \mid \| \mathrm{x} - \mathrm{w}_k \| \le \| \mathrm{x} - \mathrm{w}_l \|, \forall l\}$$

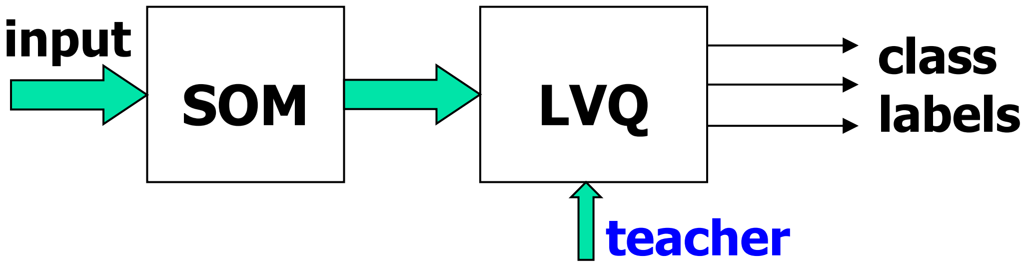
Vector Quantizer

$$V_k = \{\mathrm{x} \in R^L \mid \|\mathrm{x} - \mathrm{w}_k\| \leq \|\mathrm{x} - \mathrm{w}_l\|, \forall l\}$$

- The collection of possible reference vectors is called the ***codebook*** of the quantizer, and its members are called ***code vectors***.
- The SOM algorithm provides *an approximation method* for computing the Voronoi vectors in unsupervised manner.



Learning Vector Quantizer (LVQ)



- LVQ is a supervised learning technique that uses class information to move the Voronoi vectors slightly, so as to improve the quality of the classifier decision regions.

LVQ1

- An input vector **x** is picked at random from the input space.
- If the class labels of the input vector **x** and a Voronoi vector **w** agree, the Voronoi vector w is moved **in the direction** of the input vector x.
- If the class labels of the input vector **x** and the Voronoi vector **w** disagree, the Voronoi vector w is moved **away from** the input vector x.

if the winner belongs to the right class

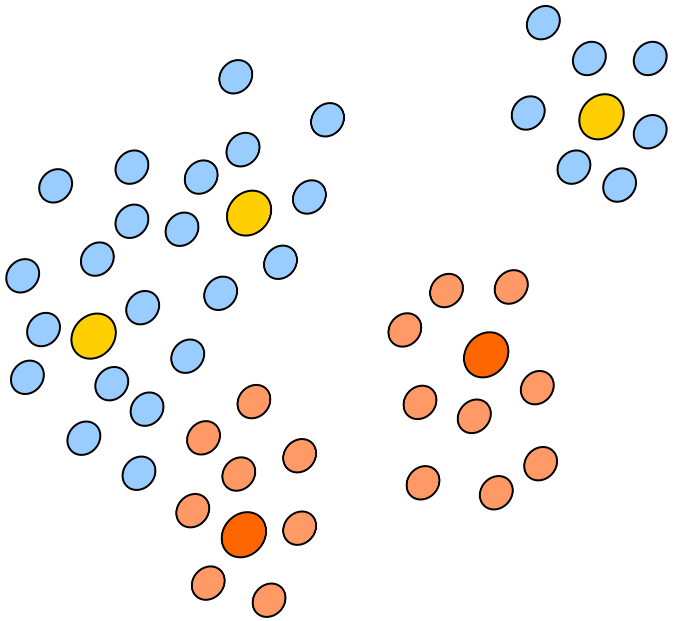
$$\mathrm{w}^{new} = \mathrm{w}^{old} + \eta(\mathrm{x} - \mathrm{w})$$

if the winner belongs to the wrong class

$$\mathrm{w}^{new} = \mathrm{w}^{old} - \eta(\mathrm{x} - \mathrm{w})$$

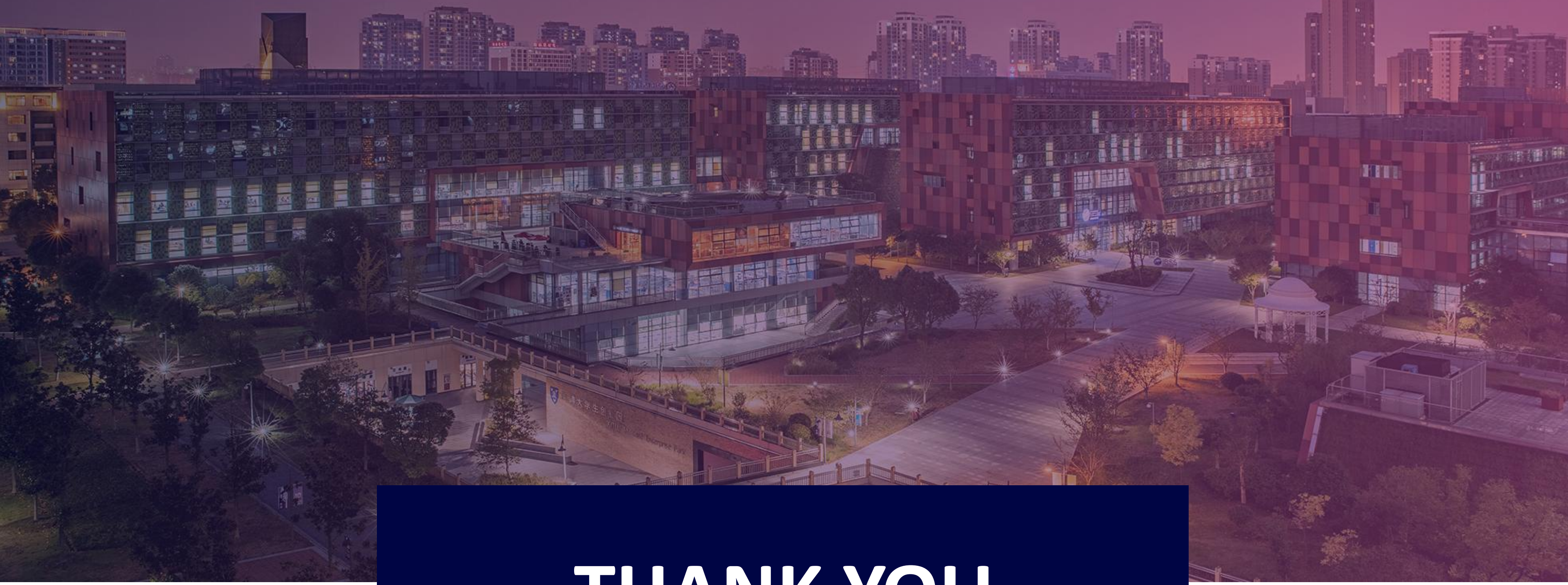
LVQ2

- Initialize prototype vectors (for different classes)
- Present a single example
- Identify closest correct and closest *wrong* prototypes
- Move the corresponding *winner* towards / away from the example



LVQ Discussion

- Stopping criteria
  - Codebook vectors have stabilized
  - Or maximum number of epochs has been reached
- Advantages
  - Appear plausible, intuitive, flexible
  - Fast and easy to implement
  - Frequently applied in a variety of problems involving the classification of structured data
- Disadvantages
  - Not stable for overlapping classes
  - Very sensitive to initialization



THANK YOU

VISIT US

WWW.XJTLU.EDU.CN

FOLLOW US

@XJTLU

Xi'an Jiaotong-Liverpool University
西交利物浦大学