Introduction to Computational Intelligence

Unsupervised feedforward neural networks

Igor Farkaš
Centre for Cognitive Science
Comenius University Bratislava

Feature extraction

- starts from an initial set of measured data
- the number of input variables for a dataset = dimensionality
- builds derived values (features) intended to be informative and non-redundant
- linear or nonlinear FE methods
- facilitating the subsequent learning and generalization steps
- in some cases leading to better human interpretations
- feature extraction entails dimensionality reduction
- nonlinear methods offer more variability of solutions
- FE methods may be based on neural networks
- We show two examples: PCA (FE) and SOM (data visualisation)

Hebbian learning – a single neuron

- Canadian psychologist Donald Hebb (1949) postulated:

“When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.”

- Let us assume one linear neuron with n inputs:

$$y = \sum_{j=1}^{n} w_j x_j = \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}$$

- According to Hebb's postulate:

$$w_j(t+1) = w_j(t) + \alpha\, y\, x_j \quad \text{for} \quad j = 1,2,\ldots,n$$

where α is the learning rate.

- Oja's rule: $$w_j(t+1) = w_j(t) + \alpha\, y\, x_i - \underbrace{\alpha\, y^2 w_j(t)}_{\text{stabilizing term}}$$

Single neuron behavior

Properties of linear neuron trained by Oja's rule on zero-meaned data vectors (i.e. $\langle \mathbf{x} \rangle = \mathbf{0}$):

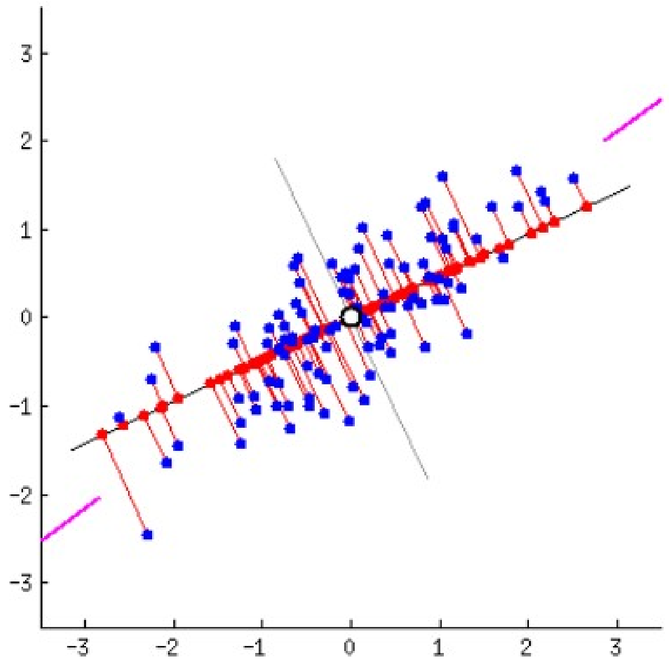
- weight vector $\boldsymbol{w}$ converges to match the 1st eigenvector of data covariance matrix $\mathbf{R} = \langle \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}} \rangle$
- after convergence, $\boldsymbol{w}$ maximizes $\langle y^2 \rangle$
- after convergence, the vector norm $\|\boldsymbol{w}\| = 1$.

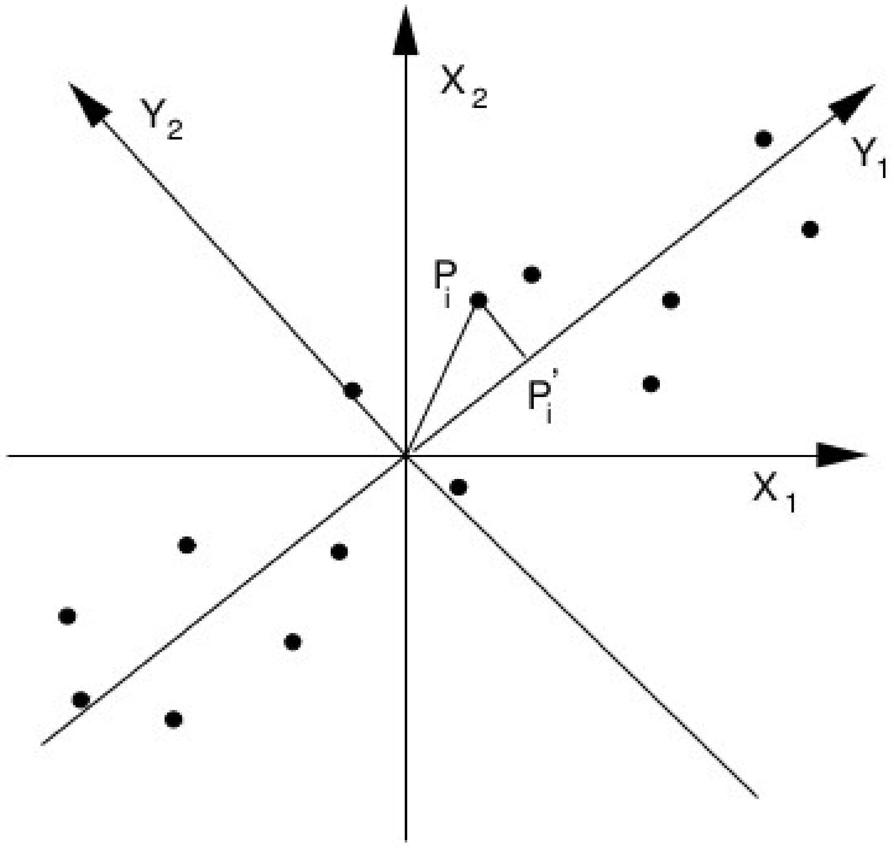
- Hence, the linear Hebbian neuron projects input vectors to the direction that maximizes the data discrimination capability (maximum information preservation).
- If more units are used, they can be trained to perform PCA (e.g. using GHA algorithm)

Linear feature extraction

X = original space (2D)
Y = projection space (2D)

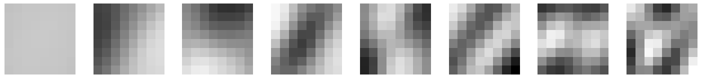
Direction Y1 captures maximum variance in original 2D data.



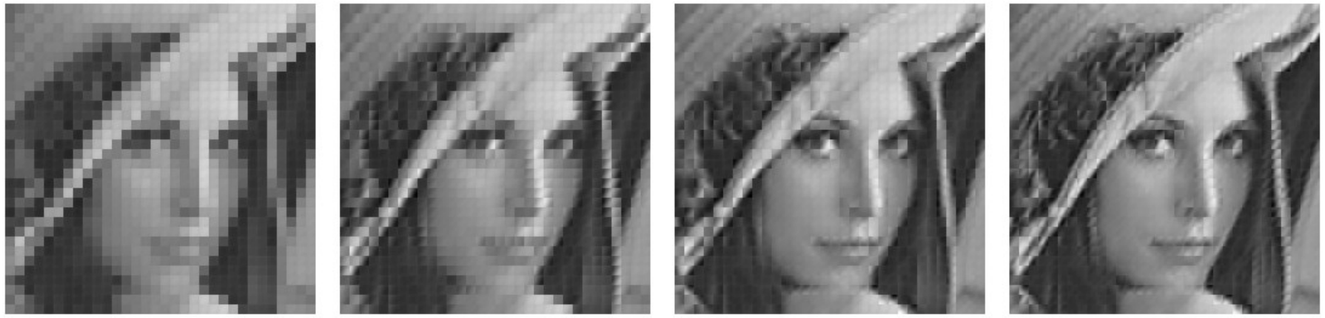


https://stats.stackexchange.com/questions/2691/making-sense-of-principal-component-analysis-eigenvectors-eigenvalues

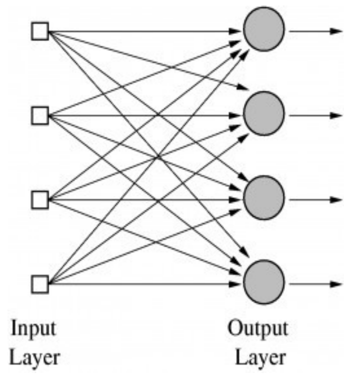
Example of PCA application

- Image of 256x256 pixels, converted to gray image (0...255)
- NN trained on sub-images 8x8 (input size)
- 8 largest eigenvectors (weight vectors): 

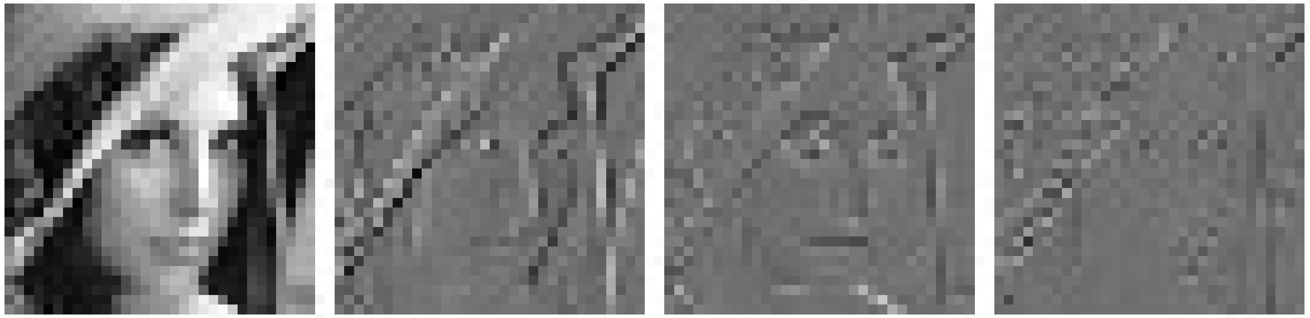
Reconstructions:



64 inputs



Neuron outputs:

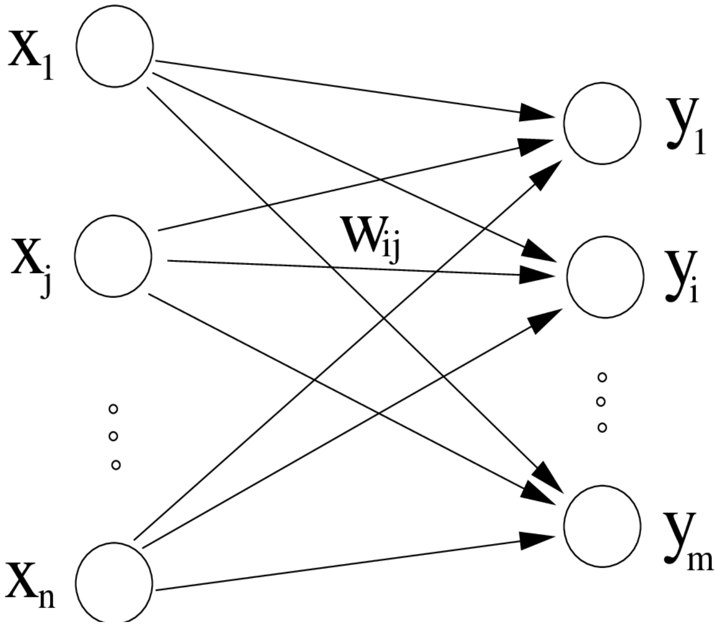


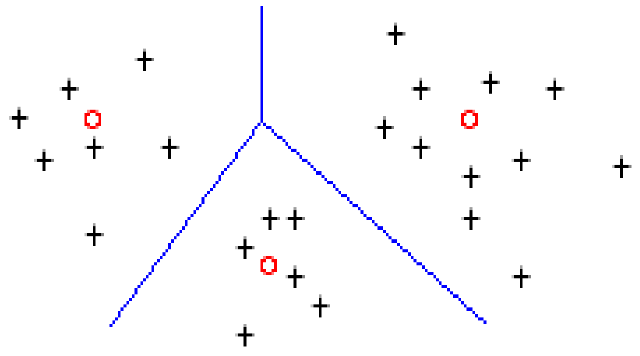
Competitive learning

- a kind of unsupervised learning
- Features:
  - linear neurons
  - winner: $y_c = \max_i \{\boldsymbol{w}_i^T . \boldsymbol{x}\}$
    - i.e. best matching unit *c*
  - winner-take-all adaptation:

    $$\Delta \boldsymbol{w}_c = \alpha(\boldsymbol{x} - \boldsymbol{w}_c) \qquad \alpha \in (0,1)$$

    $$\|\boldsymbol{w}_c\| \leftarrow 1$$
  - risk of "dead" neurons
- algorithm: in each iteration
  - find winner, adapt its weights
- useful for clustering

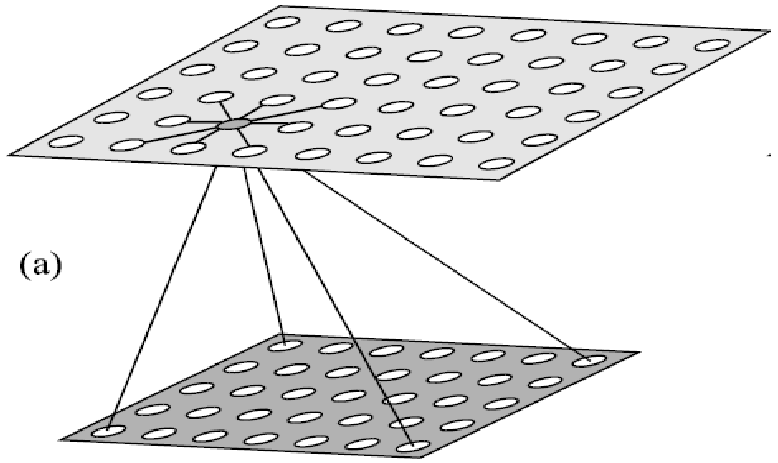


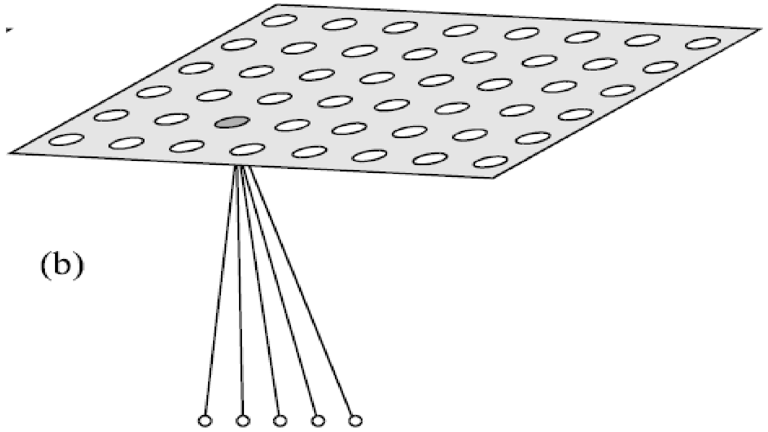


Feature mapping

biologically motivated models

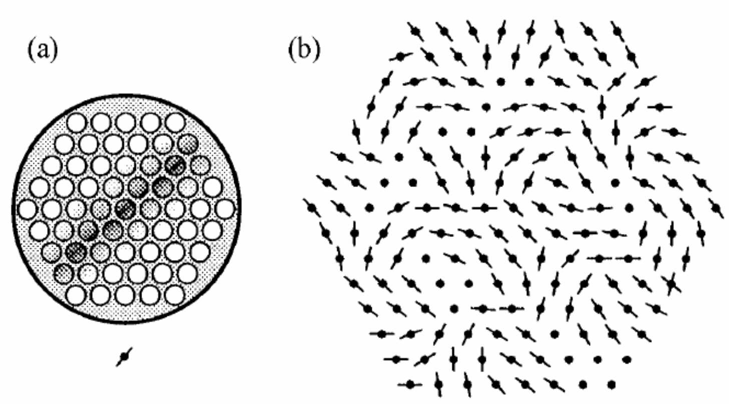
model with extracted features

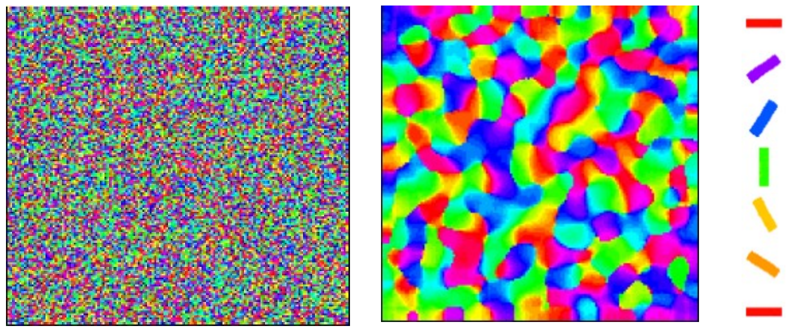




e.g. mapping from retina to cortex -> orientation map

- introduced topology of neurons in the map
- winner-take-most due to neuron cooperation

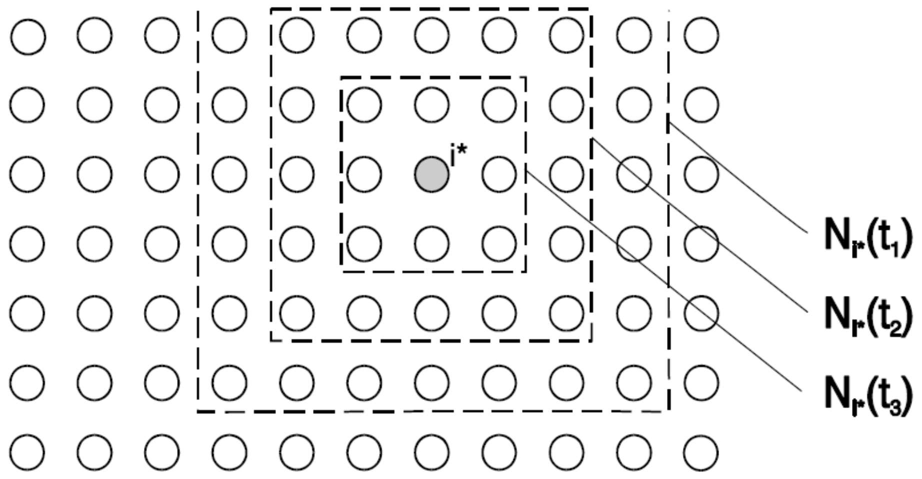


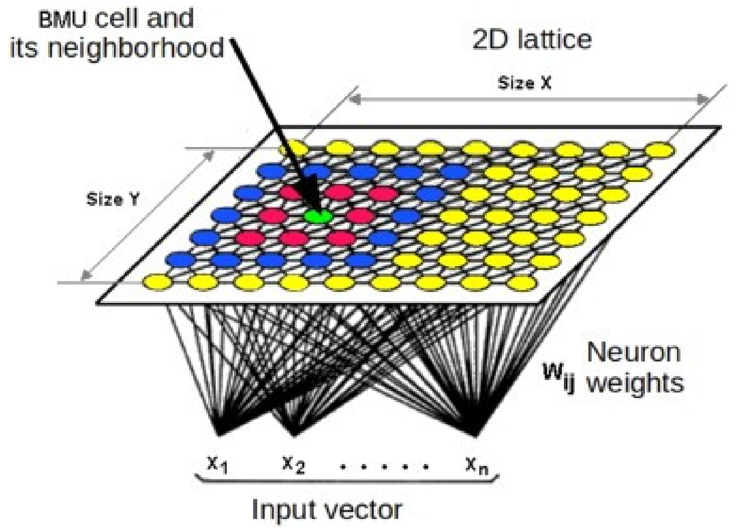


SOM: Neighborhood function

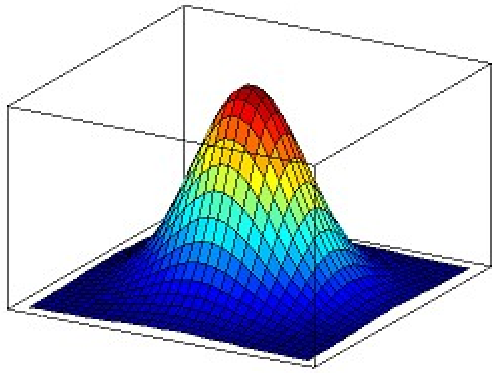
- computationally efficient substitute for lateral interactions
- neurons adapt only within the winner neighborhood
- neighborhood radius decreases in time

- rectangular neighborhood:





- alternative: Gaussian neighborhood



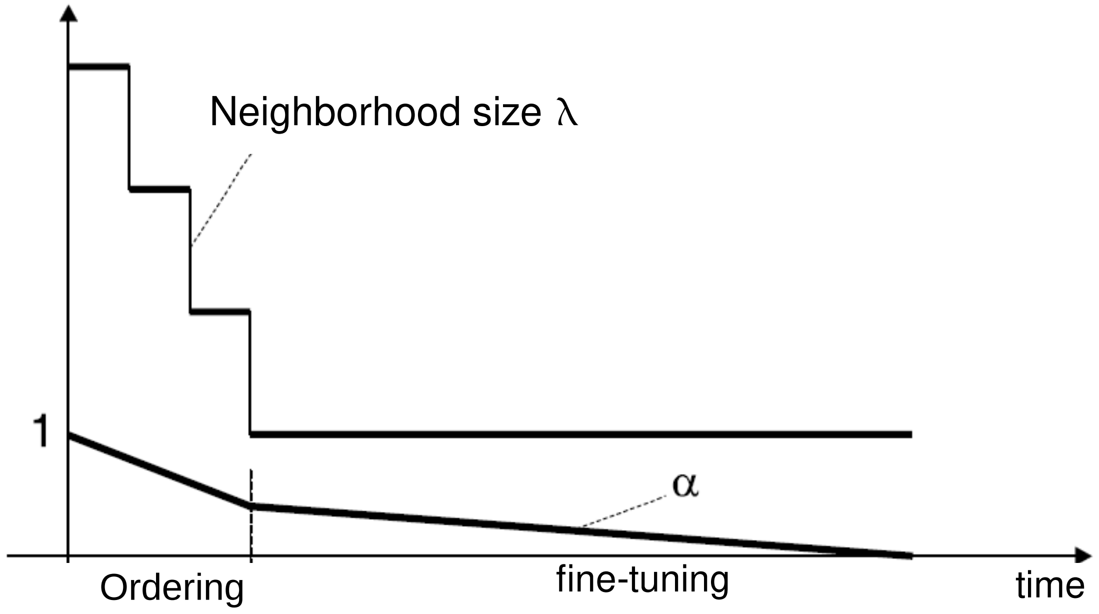
SOM algorithm

(Kohonen, 1982)

- randomly choose an input $\boldsymbol{x}$
- find winner $i*$ for $\boldsymbol{x}$ $\qquad i^* = \arg\min_i \|\boldsymbol{x} - \boldsymbol{w}_i\|$
- adapt weights within the neighborhood

$$\boldsymbol{w}_i(t+1) = \boldsymbol{w}_i(t) + \alpha(t)\, h(i^*, i)\, [\boldsymbol{x}(t) - \boldsymbol{w}_i(t)]$$

- update SOM parameters (neighborhood, learning rate)
- repeat until stopping criterion is met

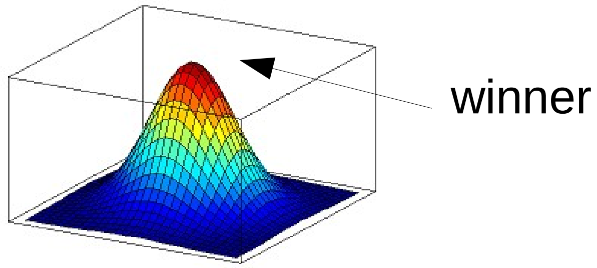


Input-output mapping:

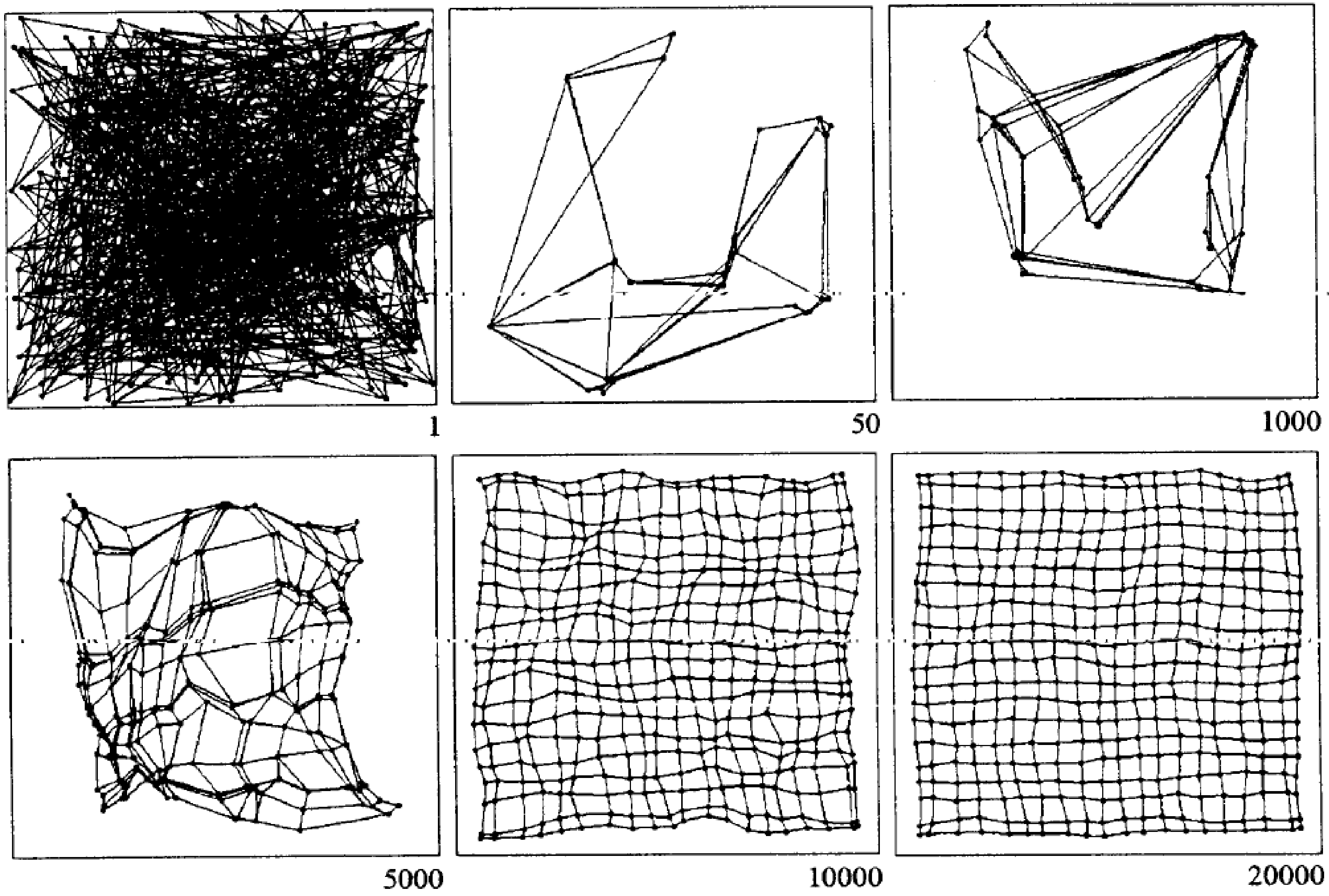
$\mathbf{X} \rightarrow \{1,2,\ldots,m\}$ or

$\mathbf{X} \rightarrow \mathbf{Y} \qquad \boldsymbol{y} = [y_1, y_2, \ldots, y_m]$

where e.g. $y_i = \exp(-\|\boldsymbol{x}\text{-}\boldsymbol{w}_i\|)$



Example: 2D inputs, 20x20 neurons

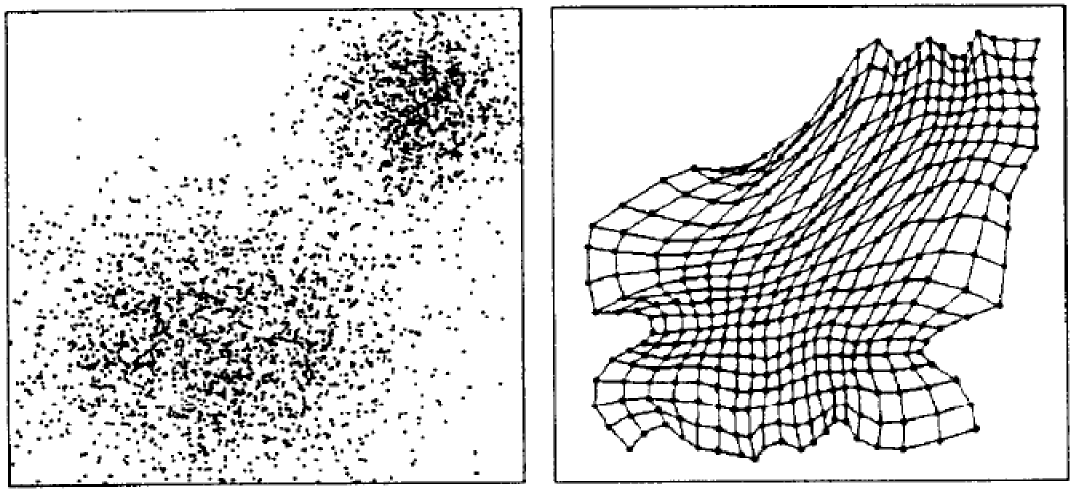


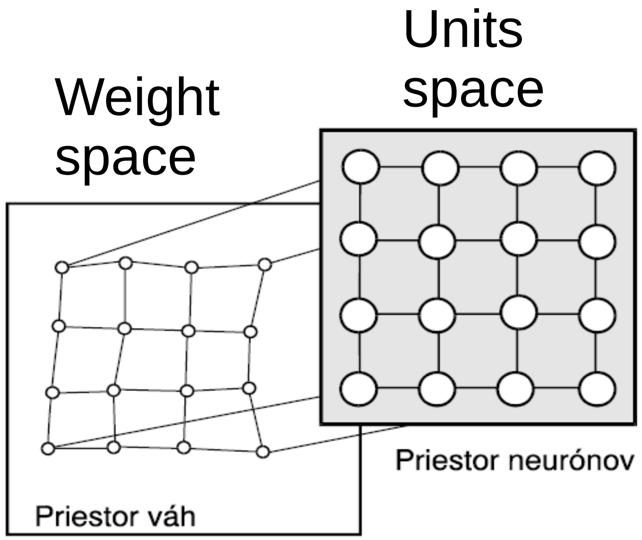
Weight vectors (knots) become topographically ordered at the end of training

SOM approximates input data distribution

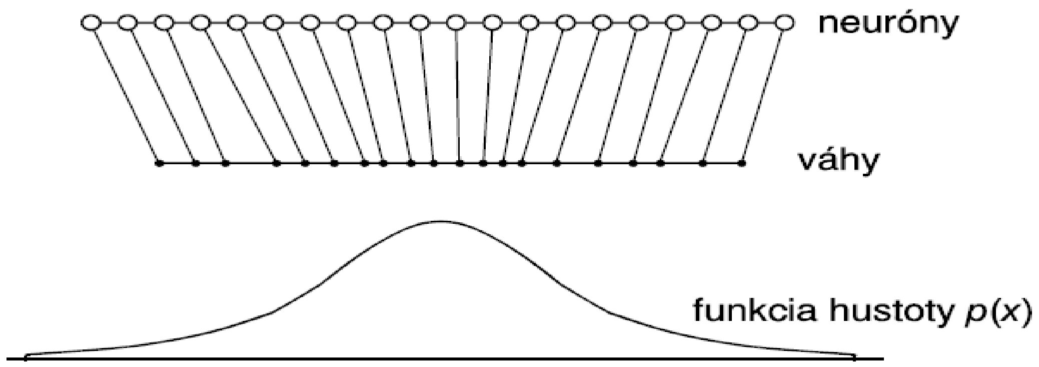
- by allocating more neurons to denser areas

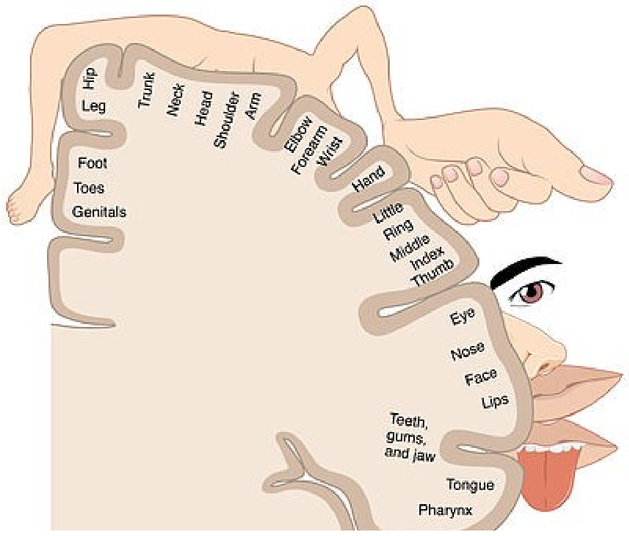
2D:





1D:



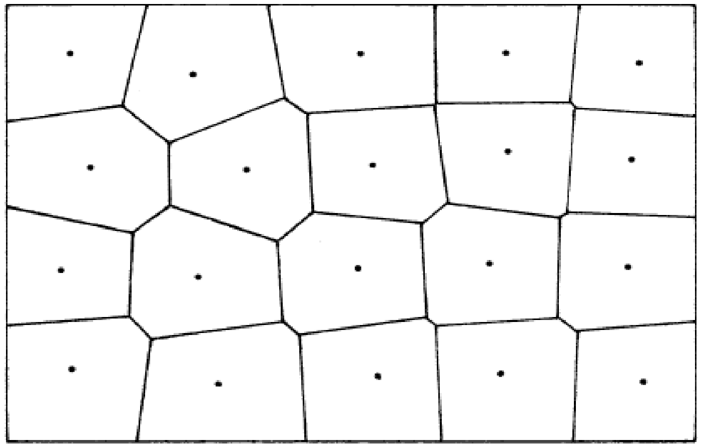


SOM performs two tasks simultaneously

Input clustering

Voronoi compartments:

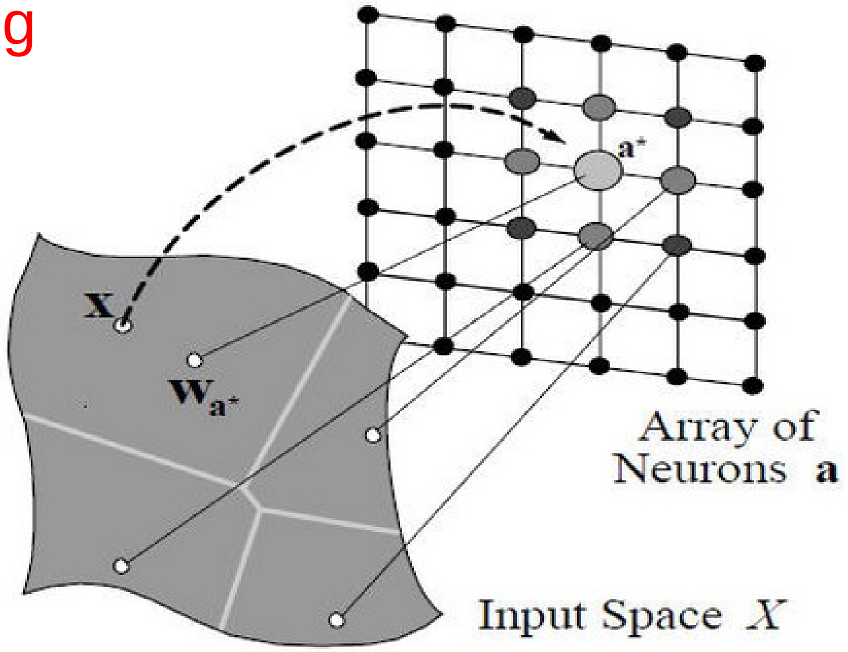
$$V_i = \{x \mid \|x - w_i\| < \|x - w_j\|, \ \forall j \neq i\}$$



Voronoi tessellation of input space

Topographic mapping

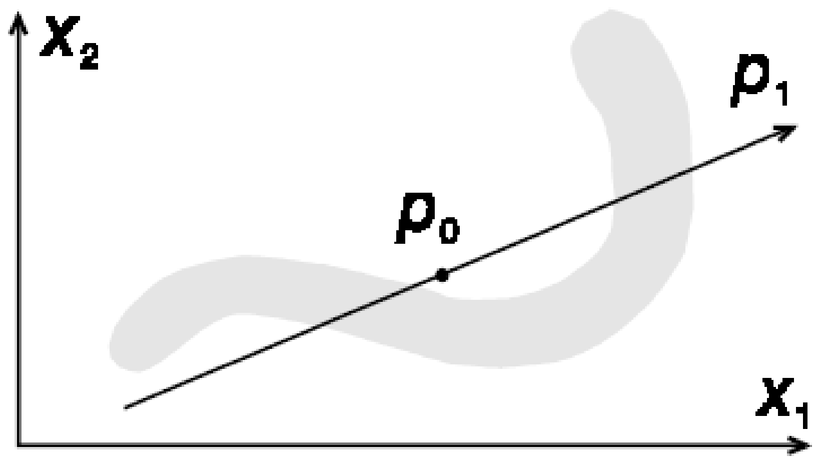
i.e. similar stimuli in input space are mapped close to each other in the SOM



Comparison of SOM to PCA

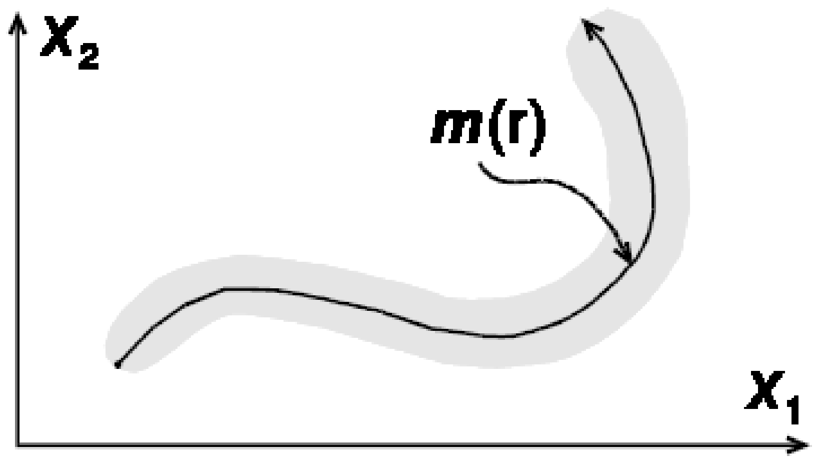
- feature extraction and mapping, difference in feature representation

PCA



(linear) principal components
One unit represents 1 dimension

SOM

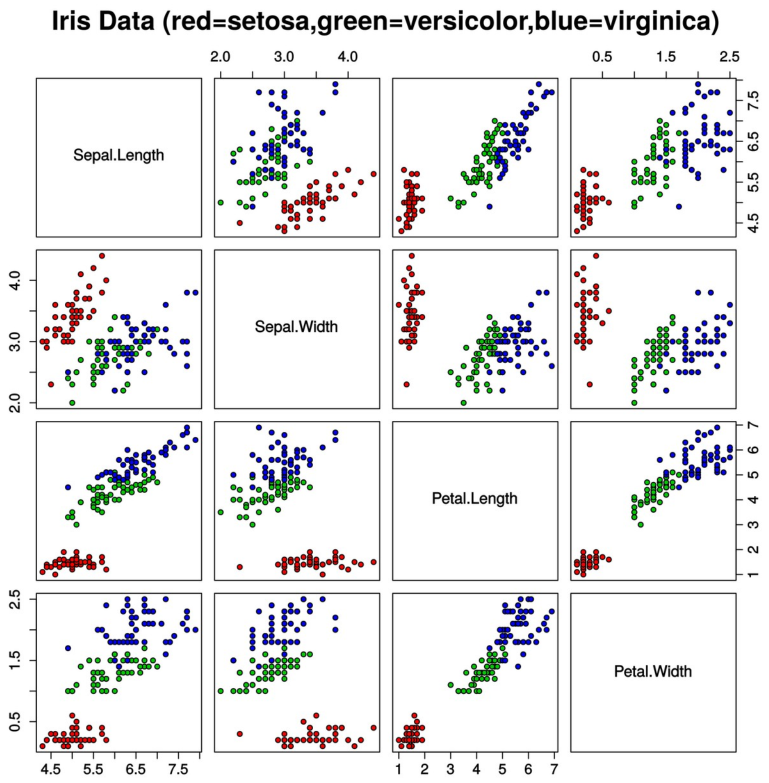


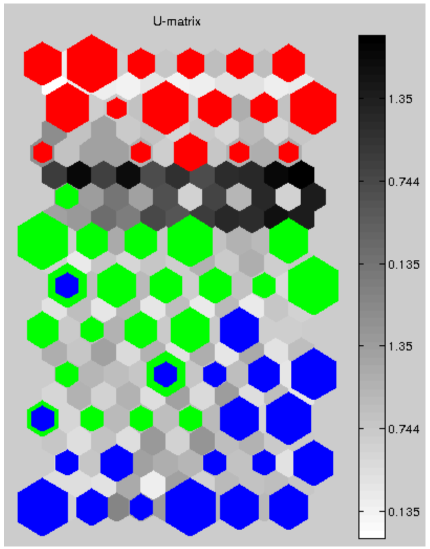
(nonlinear) principal manifold
More units represent 1 dimension

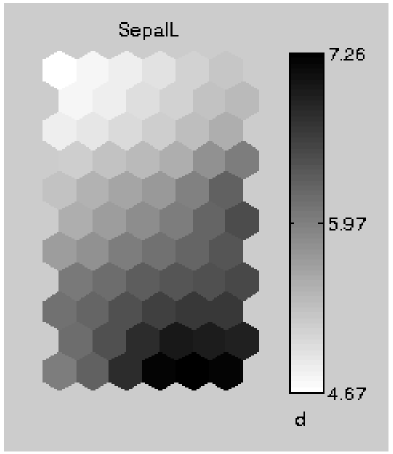
SOM as a nonlinear generalization of PCA

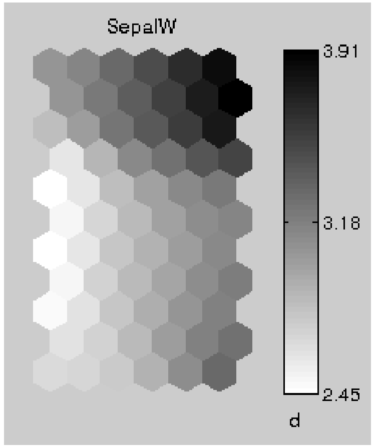
Application: Data visualization

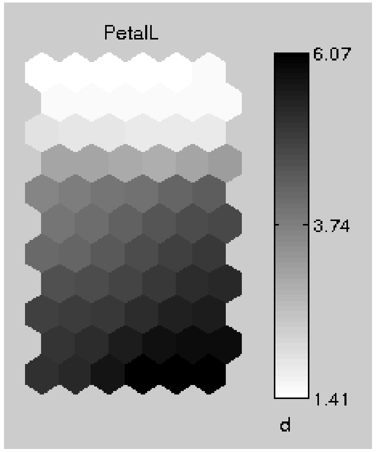
- 7x10 SOM with hexa grid
- trained on 4D Iris data
- 3 classes
- Neuron labels generated
- according to votes
- Plots of component weights reveal an order

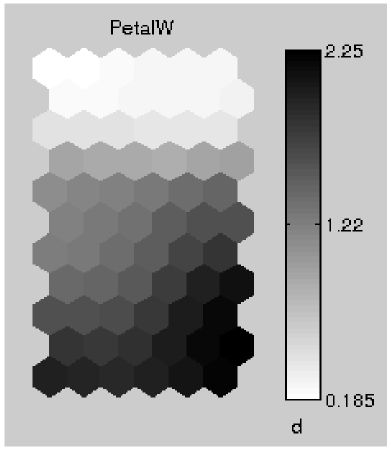












Summary

- Unsupervised methods extract features from data (in various ways)
- PCA – standard method for linear reduction of data dimensionality (preserves maximal variance)
- A single-layer neural net trained by Hebbian-like learning rule can implement PCA
- No need to calculate covariance matrix of input data
- SOM – a very popular algorithm for input clustering and topographic mapping
- useful for data visualization