Reconhecimento de Padrões
Self-Organizing Maps

Prof. George von Borries
Departamento de Estatística
Universidade de Brasília

1 - 2024

References

- Milijković, D. (2017) Brief Review of Self-Organizing Maps. MIPRO/CTS.
- Izenman, A.J. (2008) Modern Multivariate Statistical Techquiniques: Regression, Classification, and Manifold Learning. Springer.
- Kohonen, T. (2001) Self-Organizing Maps. Springer, third edition.

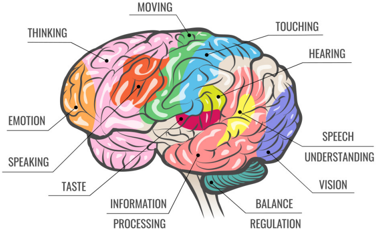
Following Milijković:

I. Introduction

Introduced by Finish Professor Teuvo Kohonen in the 1980s.
Self-organizing map (SOM), sometimes also called a Kohonen map use unsupervised, competitive learning to **produce low dimensional, discretized representation of presented high dimensional data**, while simultaneously preserving similarity relations between the presented data items. Such low dimensional representation is called a feature map.

III. B. Self-Organing Map

Neural networks of neurons with lateral communication of neurons topologically organized as self-organizing maps are common in neurobiology. Various neural functions are mapped onto identifiable regions of the brain.

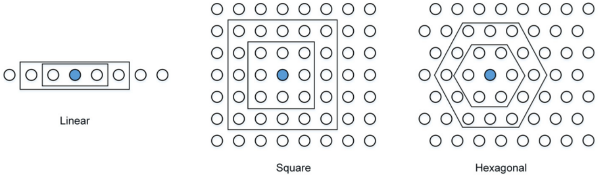


Source: vectorstock.com/35975559.

III. Self-Organization Maps

B. Self-Organing Map

A SOM is a single layer neural network with units set along an n-dimensional grid. Most applications use two- dimensional and rectangular grid, although many applications also use hexagonal grids, and some one, three, or more dimensional spaces. SOMs produce low- dimensional projection images of high-dimensional data distributions, in which the similarity relations between the data items are preserved.



Most common SOM grids and neuron neighborhoods.

C. Principles of Self-Organization in SOM

C. 1. Competition Process

For each input pattern vector presented to the map, all neurons calculate values of a discriminant function. The neuron that is most similar to the input pattern vector is the winner - best matching unit, **BMU**.

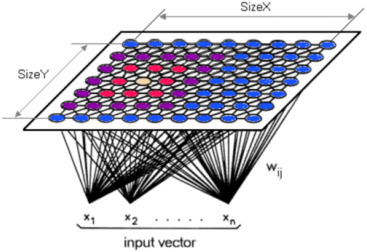
C. 2. Cooperative Process

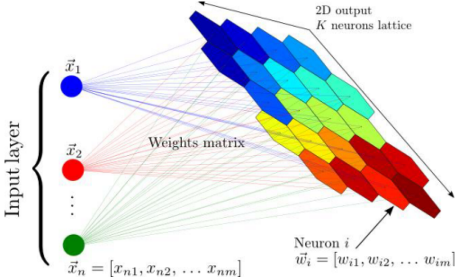
The winner neuron (BMU) finds the spatial location of a topological neighborhood of excited neurons. Neurons from this neighborhood may then cooperate.

C. 3. Synaptic Adaptation

Provides that excited neurons can modify their values of the discriminant function related to the presented input pattern vector by the **process of weight adjustments**.

D. Common Topologies





2-D SOM network.

IV. Learning Algorithm

A. Measures of Distance and Similarity

To determined similarity between the input vector and neurons measures of distance are used. Some popular distances among input pattern and SOM units are: Euclidian, Correlation, Direction cosine, Block distance.

B. Neighborhood Funcions

Neurons within a grid interact among themselves using a neighbor function. The most popular function is the multivariate Gaussian kernel function that rejects some neurons in the vicinity to the winning neuron,

$$h_k = \exp\left\{-\frac{\|\mathbf{m}_k - \mathbf{m}_k^*\|^2}{2\sigma^2}\right\} \mathbf{I}_{[k \in \mathcal{N}_c(k^*)]}$$

where $\sigma > 0$ is the neighborhood radius.

Other functions are possible, as cone and cylinder. Ordering algorithm is robust to the choice of function type if the neighbor radius and learning rate decrease to zero.

C. Initialization of SOMs

- Determine a retangular grid $r \times c$, with r rows and c columns;
- Map the collection of nodes into an ordered sequence: $1, 2, \ldots, r \times c$;
- Let $\mathbf{m}_k$ be a representative of the k-th node;
- Initialize $\mathbf{m}_k$ as

  - random values, completely independent of the training data set;
  - or random samples from the input training data;
  - or initialization that tries to reflect the distribution of the data (PCs).

D. Training

1. Sequential Training

- Randomly select $\mathbf{x}$ from the data set and standardize it;
- Compute the Euclidean distance between $\mathbf{x}$ and each representative of a node, i.e., $\|\mathbf{x} - \mathbf{m}_k\|$;
- Find $k^* = \arg\min_k \{\|\mathbf{x} - \mathbf{m}_k\|\}$, where k^* indicates the Best Matching Unit (BMU) or the winner node for the input vector $\mathbf{x}$.

D. Training

1. Sequential Training

- Look at those nodes that are neighbors of the winning node.

  - k' is a neighbor of k if the Euclidean distance between $\mathbf{m}_k$ and $\mathbf{m}_{k'}$ is smaller than a given threshold c.

  - $\mathcal{N}_c(k^*)$ is the set of nodes wihich are neighbors of the winning node k^*.

- Update each $\mathbf{m}_k$, $k \in \mathcal{N}_c(k^*)$

$$\mathbf{m}_k \leftarrow \mathbf{m}_k + \alpha\, h_k\, (\mathbf{x} - \mathbf{m}_k),\ k \in \mathcal{N}_c(k^*)$$

  with $0 < \alpha < 1$. For $k \notin \mathcal{N}_c(k^*), \alpha = 0$.

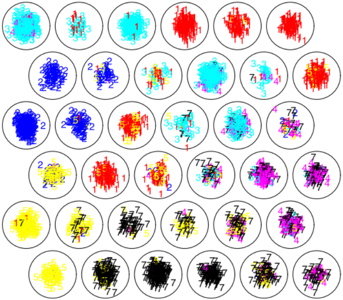
  h_k could be the Gaussian kernel function.

- Repeat the process a large number of times $(t = 1, \ldots, T)$.
- In the process, α decreases as a funciton of t.
- Run through the collection of input vectors one at a time.

D. Training

2. Batch Training

- Instead fo one node, do each step for a group of nodes and update $\mathbf{m}_k$ by averaging the result obtained for the group of nodes.
- Nodes are drawn in circles and data points that are mapped to a node are randomly plotted within the circles corresponding to a particular node. Inside the circles, we can also have a descriptive representation of the points from a node.



A 6×6 hexagonal batch-SOM plot.

V. Properties of SOM

- **Approximation of the Input Space** - The resulting mapping provides a good approximation to the input space. SOM also performs **dimensionality reduction** by mapping multidimensional data on SOM grid.
- **Topological Ordering** - Spatial locations of the neurons in the SOM lattice are topologically related to the features of the input space.
- **Density Matching** - The density of the output neurons in the map approximates the statistical distribution of the input space. Regions of the input space that contain more training vectors are represented with more output neurons.
- **Feature Selection** - Map extracts the principal features of the input space. It is capable of selecting the best features for approximation of the underlying statistical distribution of the input space.

Examples using `R`