Overview

- SOM‘s in context of NNs
- Biological motivation and maps
- Algorithm basics
- Details of learning
- Three essential processes
  - Competition
  - Cooperation
  - Adaptation
- Four examples

SOMs are basically different

- Neural networks for unsupervised learning attempt to discover spatial patterns from available data without using external help.

SOMs are basically different

- Neural networks for unsupervised learning attempt to discover spatial patterns from available data without using external help.
  - There is no information about the desired class (or output) d of an example x. So only x is given.

SOMs are basically different

- Neural networks for unsupervised learning attempt to discover spatial patterns from available data without using external help.
  - There is no information about the desired class (or output) d of an example x. So only x is given.
  - Self Organizing Maps (SOM) are a neural network model for unsupervised learning, which combine a competitive learning principle with a topological structuring of neurons such that adjacent neurons tend to have similar weight vectors.

Maps: Biological Motivation

- Hypotheses of neural development from neurobiology:
  - The structure self-organizes based on learning rules and system interaction.
  - Axons physically maintain neighborhood relationships as they grow.
  - For sensorical or motorical body parts these relationsships build maps

Maps Glossary

Guiding Principle:
adjacent receptors connected to adjacent neurons in the cortex

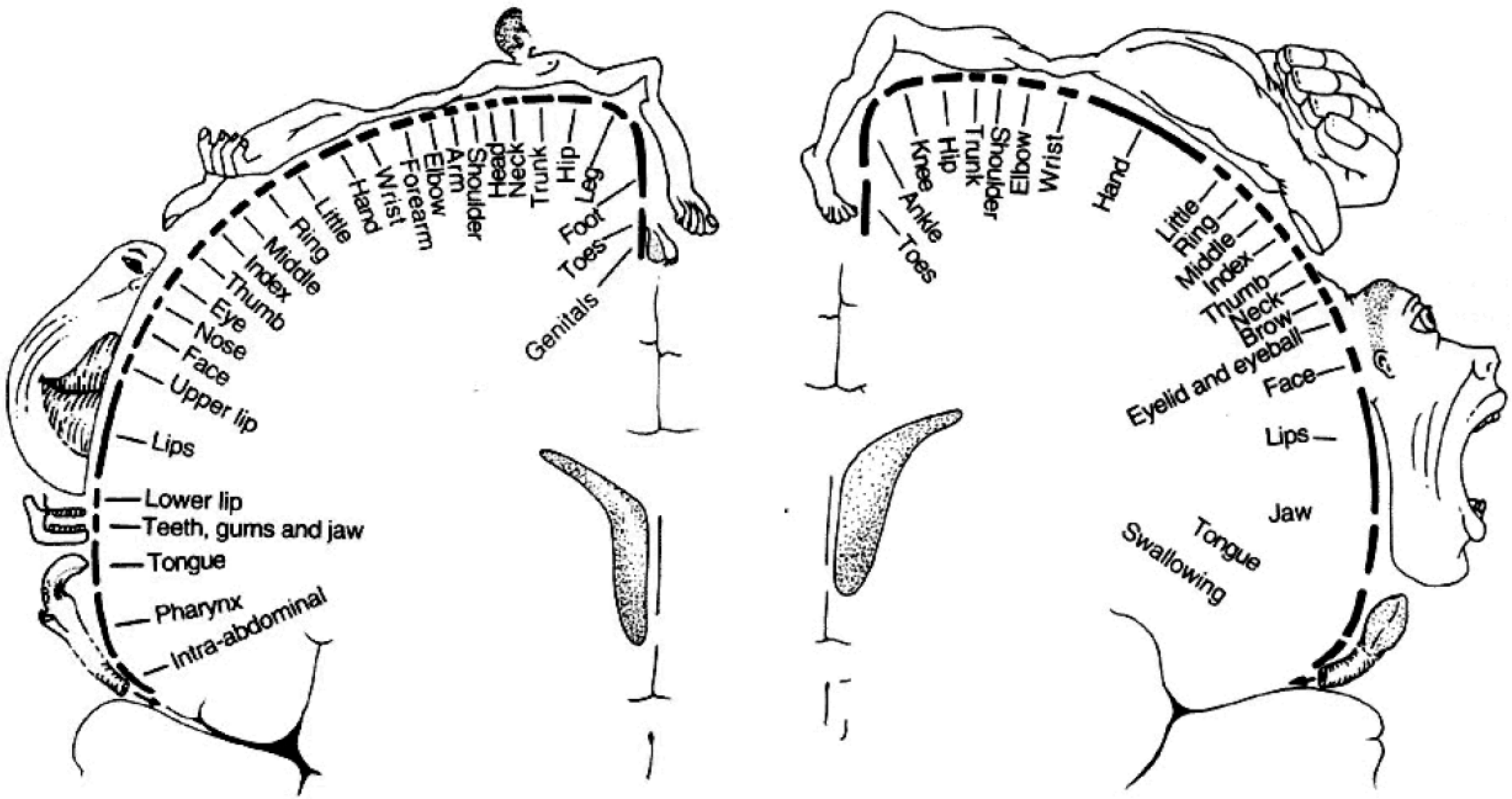
– Somatotopic map:

projection of body surface onto a brain area, called sematosensory cortex, responsible for sense of touch.

– Motor map:
similarily for movement commands

– closeness of limbs maps to closeness of "controlers"

Maps Illustration



Human sensory and motor maps

Maps Glossary

Principle: adjacent receptors connected to adjacent neurons in the cortex

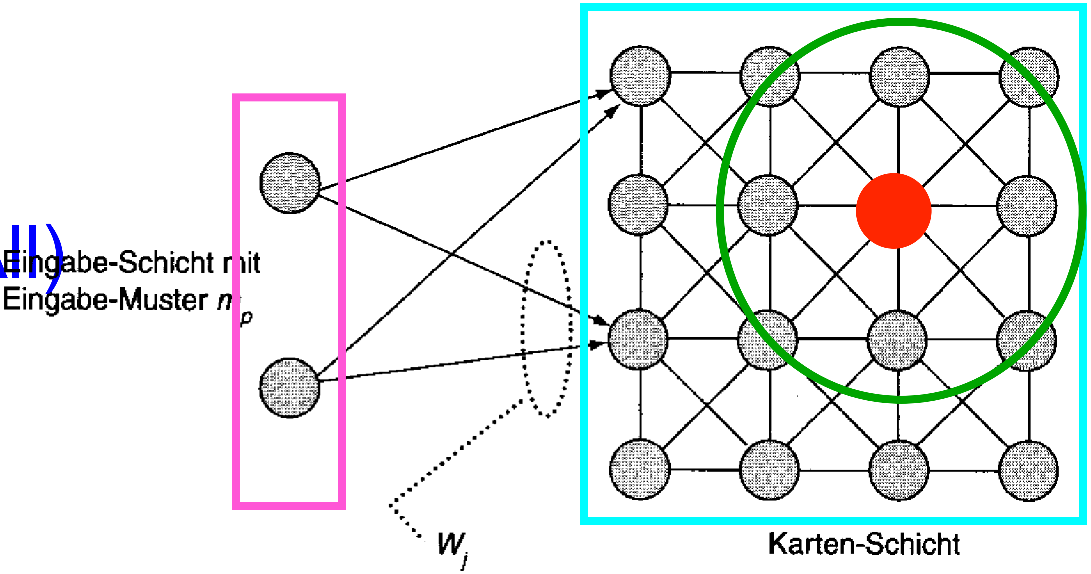
- Somatotopic map:

  projection of body surface onto a brain area, called sematosensory cortex, responsible for sense of touch.
- Motor map:

  Is similar for movement commands instead of touch.
- Retinotopic map:
  Is for vision. The area is called superior colliculus.
- Phonotopic map:
  Is for hearing: the auditory cortex.

Algorithm Basics

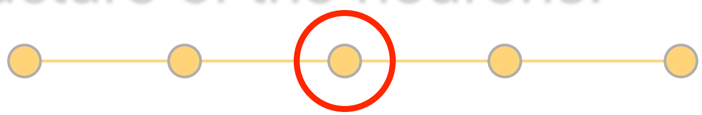
- WTA (**Winner** Takes All) algorithm
- Two layers:
  – Input layer: fully connected to each neuron in second layer
  – Map layer: 1/2/3 dimensional, neighborhoods relations organized as line, square (torus) or cube



W_i any to any

Architecture

- The input is connected with each neuron of a lattice.
- Topology of the lattice: It determines a neighborhood structure of the neurons.

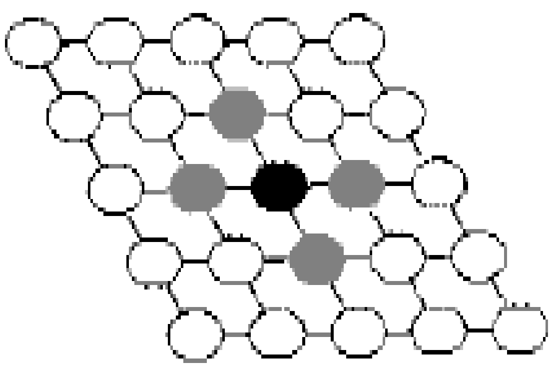


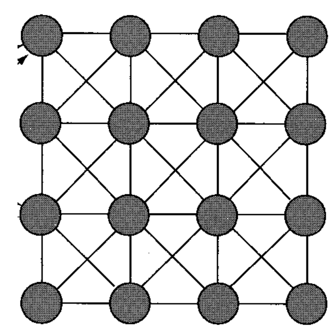
1-dimensional topology

A small neighborhood

2-dimensional topologies

many possible neighborhoods





Learning Algorithm Goal

- We have to find values for the weight vectors of the links from the input layer to the nodes of the lattice, in such a way that adjacent neurons will have similar weight vectors.
- For an input, the output of the neural network will be that neuron whose weight vector is most similar (with respect to Euclidean distance) to that input.
- In this way **each** (weight vector of a) **neuron is** the center of **a cluster** containing all the input examples which are mapped to that neuron.

Algorithm Essential Ingredients

- A continuous input space of activation patterns that are generated in accordance with a certain probability distribution.

- A topology of the network in the form of a lattice of neurons, which defines a discrete output space

- A time-varying neighborhood function $h_{ij(x)}(n)$ that is defined around a winning neuron $i(x)$.

- A learning-rate parameter h(n) that starts at an initial value h_0 and then decreases gradually with time, n, but never goes to zero.

SOM Training Algorithm Summary

- Initialization: choose $\boxed{\text{random small values}}$ for weight vectors such that $w_j(0)$ is different for all neurons j.
- Sampling: draw a sample x from input space.
- Similarity matching: find the best matching winning neuron i(x) at step n:

$$i(x) = \arg\min_j \| x(n) - w_j \| \qquad j \in [1, 2, \ldots, \ell]$$

- Updating: adjust synaptic weight vectors of all neurons using rule

$$w_j(n+1) = w_j(n) + \eta(n)\, h_{ij(x)}(n) \left(\mathrm{x} - w_j(n) \right)$$

- Continuation: go to Sampling step until no noticeable changes in the feature map are observed.

General Algorithm Formulas

- Learning formulas:

$$W_i = W_i + \eta(x - W_i)$$

$$W_i = W_i + \eta N(i,x)(x - W_i) \quad N(i,x) = \begin{cases} 1 \; for \; d(i,w) \le \lambda \\ 0 \quad else \end{cases}$$

$$net_j = \overbrace{\sum_i o_i w_{ij}}^{Input} + \overbrace{\theta_j}^{Bias}$$

$$W_i = W_i + \eta e^{-\frac{d_{ij}{}^2}{2\sigma^2}}(x - W_i)$$

$$act_j = \frac{1}{1 + \exp(-net_j)}$$

i	[1 … l]
W_i	synaptic weight of winning neuron
x	input pattern
η	learning rate
λ	neighborhood size

Learning

Informal description:

- Given: an input pattern x
- Find: the neuron **i** which has closest weight vector by competition ($w_i^T x$ will be the highest i.e. is winner).
- For each neuron j in the neighborhood N(i) of the winning neuron i:
  - update the weight vector of j.

Learning

- Neurons not in the neighborhood are left unchanged.
- The SOM algorithm:
  - Starts with large neighborhood size λ and gradually reduces it.
  - Gradually reduces the learning rate η.

Workings

– Upon repeated presentations of the training examples, the weight vectors tend to follow the distribution of the examples.

– This results in a topological ordering of the neurons, where neurons adjacent to each other tend to have similar weights.

Three essential processes

- competition
- cooperation
- weight adaptation

1 Competition

- Competition:
  - Competitive process: Find best match of input vector x with weight vectors:

$$W_{i(x)} = \arg\min_j \| x - W_j \| \qquad j = 1, 2, \ldots, \boxed{\ell}$$

winning neuron

total number of neurons

  - The input space of patterns is mapped onto a discrete output space of neurons by a process of competition among the neurons of the network.

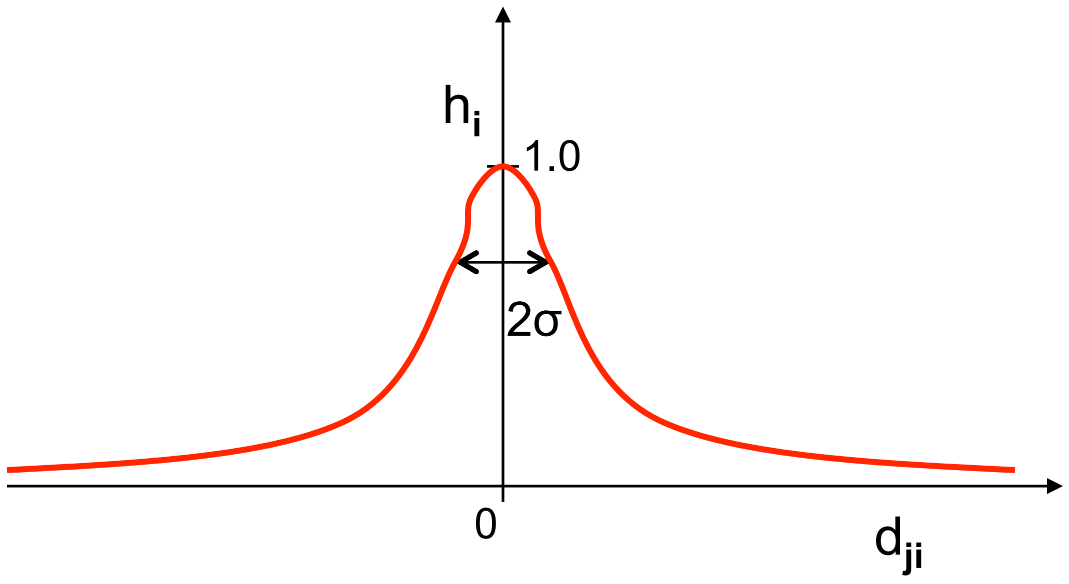
2 Cooperation

- Cooperation:
  - Cooperative process:
    The winning neuron locates the center of a topological neighborhood of cooperating neurons.
  - The topological neighborhood depends on lateral distance d_{ij} between the winner neuron i and neuron j.

Neighborhood Function

– Gaussian neighborhood function

$$h_i(d_{ji}) = \exp\left(-{d_{ji}^2}\Big/{2\sigma^2}\right)$$



Neighborhood Function

– σ (effective width) measures degree to which excited neurons in the vicinity of the winning neuron participate to the learning process.

exponential decay update

$$\sigma(n) = \boxed{\sigma_0} \exp\left(- \cancel{n / \boxed{T_1}} \right)$$

first time constant

– d_{ji}: lateral distance

- in one dimension lattice | j - i |
- in two dimension lattice $d_{ji} = \| r_j - r_i \|$
  r_j is the position of neuron j in the lattice.

3 Weight Adaptation

- Applied to all neurons inside the neighborhood of the winning neuron i.

$$\Delta w_j = \underbrace{\eta y_j x}_{\text{Hebbian term}} - \underbrace{g(y_j) w_j}_{\text{forgetting term}}$$

scalar function of response y_j

$$g(y_j) = \eta y_j$$

$$y_j = h_{i,j(x)}$$

$$w_j(n+1) = w_j(n) + \eta(n)\, h_{ij(x)}(n) \left(\mathrm{x} - w_j(n)\right)$$

exponential decay update:

$$\eta(n) = \boxed{\eta_0} \exp\left(- n / \boxed{T_2} \right)$$

second time constant

Two phases of weights adaptation

- Ordering phase:
  - Topological ordering of weight vectors.
  - May take 1000 or more $\boxed{\text{iterations}}$ of SOM algorithm.
- Important choice of parameter values:
  - $\eta(n)$: $\eta_0 = 0.1$ $T_2 = 1000$
    $\Rightarrow$ decrease gradually $\eta(n) \geq 0.01$
  - $h_{ji(x)}(n)$: σ_0 big enough $T_1 = \dfrac{1000}{\log(\sigma_0)}$
  - Initially the neighborhood of the winning neuron includes almost all neurons in the network, then it shrinks slowly with time.

Two phases of weights adaptation

- Convergence phase:
  - Fine tune feature map.
  - Must be at least 500 times the number of neurons in the network $\Rightarrow$ thousands or tens of thousands of iterations.
- Choice of parameter values:
  - $\eta(n)$ maintained on the order of 0.01.
  - $h_{ji(x)}(n)$ contains only the nearest neighbors of the winning neuron. It eventually reduces to one or zero neighboring neurons.

Summary of SOM

- Initialization: choose $\boxed{\text{random small values}}$ for weight vectors such that $w_j(0)$ is different for all neurons j.
- Sampling: drawn a sample example x from the input space.
- Similarity matching: find the best matching winning neuron i(x) at step n:

$$i(x) = \arg\min_j \| x(n) - w_j \| \qquad j \in [1, 2, \ldots, \ell]$$

- Updating: adjust synaptic weight vectors of all neurons using rule

$$w_j(n+1) = w_j(n) + \eta(n)\, h_{ij(x)}(n) \left(\mathrm{x} - w_j(n) \right)$$

- Continuation: go to Sampling step until no noticeable changes in the feature map are observed.

Summary SOM in Pseudo code

```
procedure train_SOM
begin
    randomize weights for all neurons
    for (i = 1 to iteration_number) do
    begin
        take one random input pattern
        find the winning neuron
        find neighbors of the winner
        modify synaptic weights of these neurons
        reduce the η  and  λ
    end
end
```

Example 1

A two dimensional lattice driven by a two dimensional distribution:

- 100 neurons arranged in a 2D lattice of 10 x 10 nodes.
- Input is bidimensional: $x = (x_1, x_2)$ from uniform distribution in region: $\{ (-1 < x_1 < +1); (-1 < x_2 < +1) \}$
- Weights are initialized with *random* values.

Visualization

- Neurons are visualized as changing positions in the *weight space* (which has the same dimension of the input space) as training takes place.

Example 1

Section 9.6 Computer Simulations

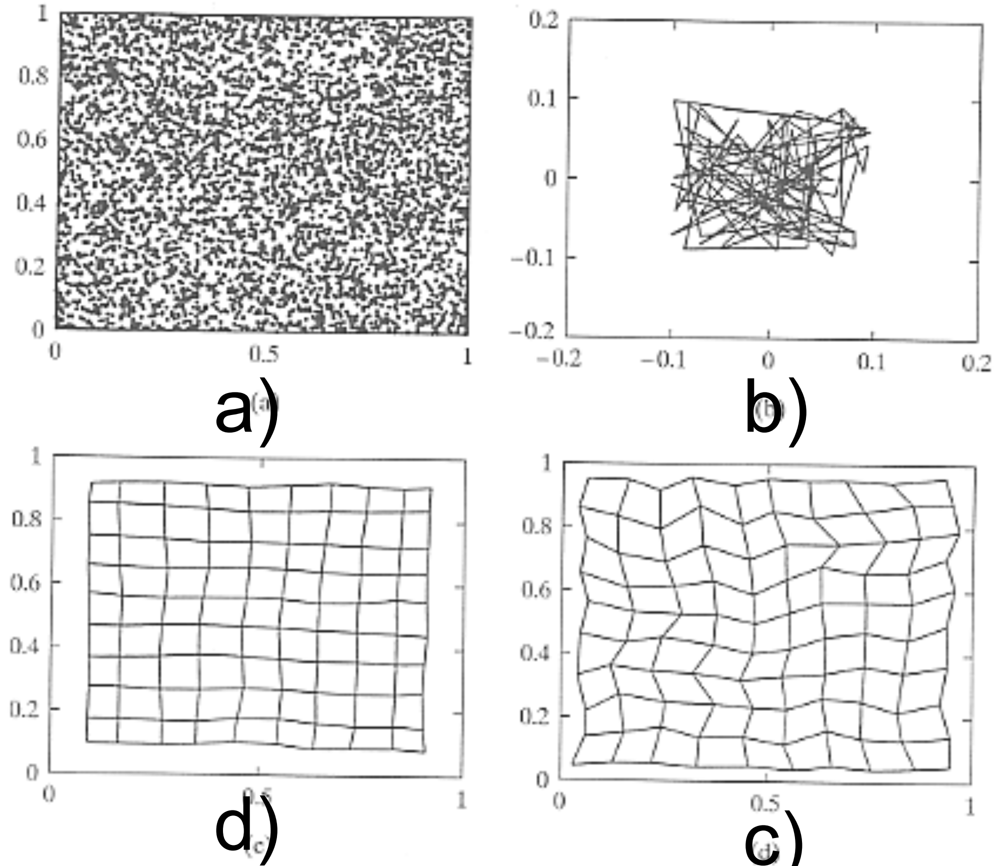


FIGURE 9.8 (a) Input data distribution. (b) Initial condition of the two-dimensional lattice. (c) Condition of the lattice at the end of the ordering phase. (d) Condition of the lattice at the end of the convergence phase.

Initial h function (Example 1)

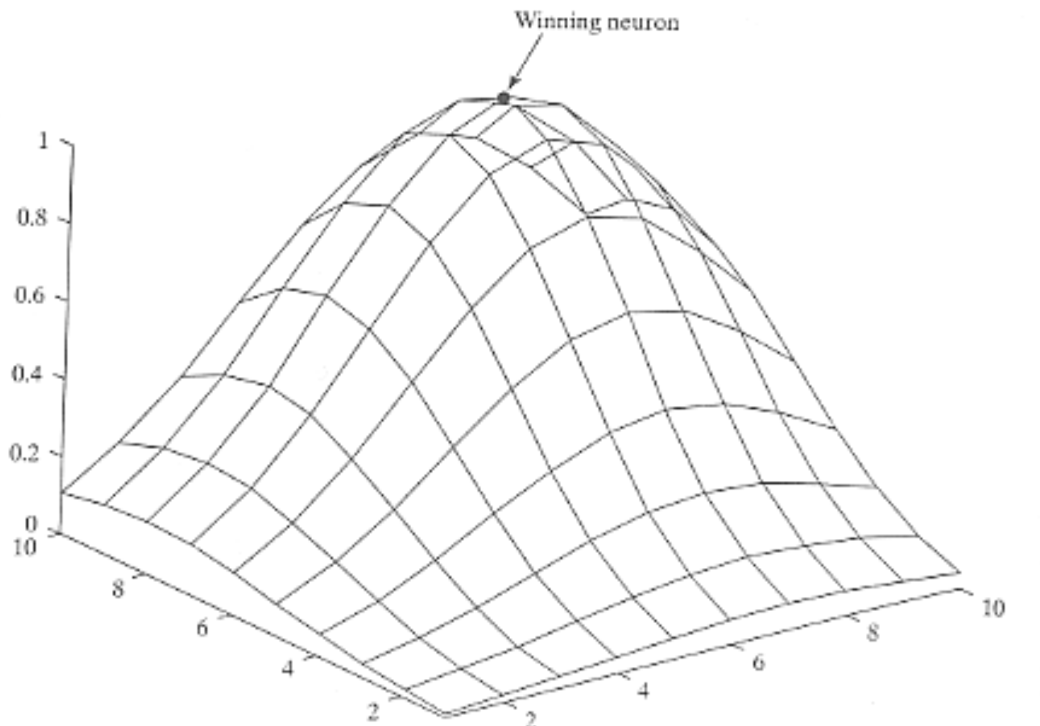


FIGURE 9.11 Initial condition of two-dimensional Gaussian neighborhood function centered on a winning neuron located at the point (7, 8) in a two-dimesional lattice of 10 × 10 neurons.

for the one-dimensional lattice, except for the fact that the neighborhood function is

Example 2

A one dimensional lattice driven by a two dimensional distribution:

- 100 neurons arranged in one dimensional lattice.

- Input space is the same as in Example 1.

- Weights are initialized with *random* values (again like in example 1).
- (Matlab programs for Examples 1, 2 available at
- http://www.mathworks.com/matlabcentral/fileexchange/6267-neural-networks-a-comprehensive-foundation-2e-book-companion-software/content/haykin

Example 2

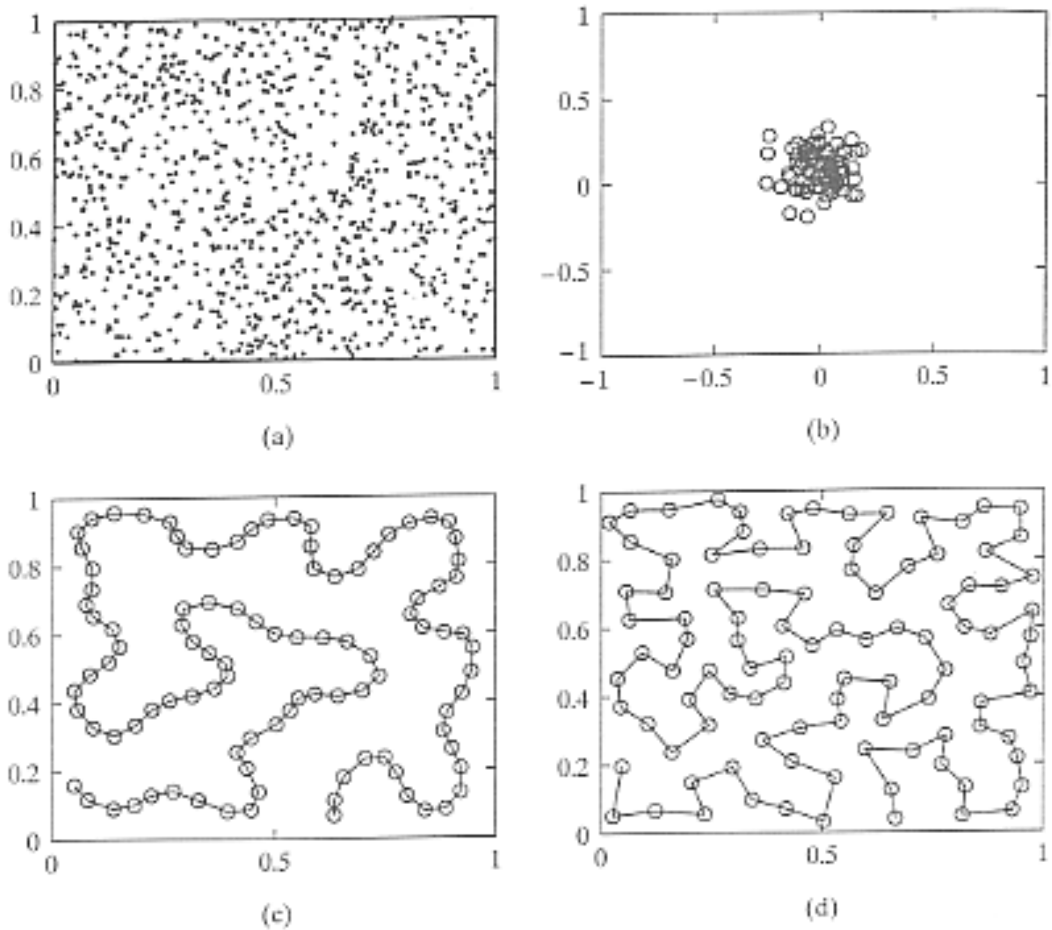
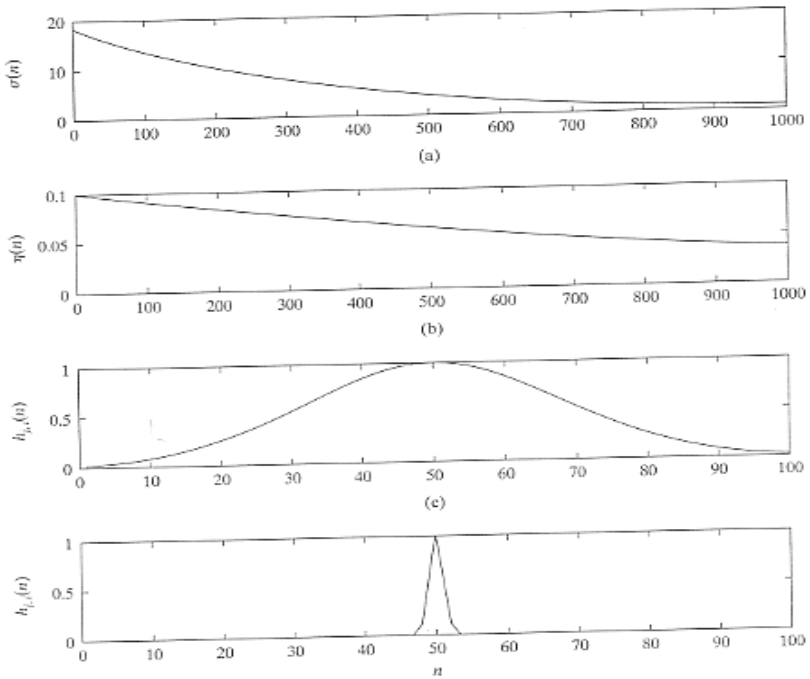


FIGURE 9.9 (a) Two-dimensional input data distribution. (b) Initial condition of the one-dimensional lattice. (c) Condition of the lattice at the end of the ordering phase. (d) Condition of the lattice at the end of the convergence phase.

resembels space filling Peano curves

Example 2: time developements



σ

η

h

FIGURE 9.10 (a) Exponential decay of neighborhood function parameter $\sigma(n)$. (b) Exponential decay of learning-rate parameter $\eta(n)$. (c) Initial shape of the Gaussian neighborhood function. (d) Shape of the neighborhood function at the end of the ordering phase (i.e., beginning of the convergence phase).

in Fig. 9.10a, starts with an initial value $\sigma_0 = 18$ and then shrinks to about 1 in 1000 iterations during the ordering phase. During that same phase, the learning-rate parameter $\eta(n)$ starts with an initial value $\eta_0 = 0.1$ and then decreases to 0.037. Figure 9.10c shows the initial Gaussian distribution of neurons around a winning neuron located at the midpoint of the one-dimensional lattice. Figure 9.10d shows the shape of the neighborhood function at the end of the ordering phase. During the convergence phase the learning-rate parameter decreases linearly from 0.037 to 0.001 in 5000 iterations. During the same phase the neighborhood function decreases essentially to zero.

The specifications of the ordering phase and convergence phase for the computer simulations in Fig. 9.8 involving the two-dimensional lattice are similar to those used

Ex3: Self Organizing Semantic Maps

- Class labels: assigned to neurons in a 2D lattice, depending on how each test pattern excites a particular neuron in the self organized networks.

$\Rightarrow$ Neurons in the 2D lattice are partitioned into a number of coherent regions.

Ex3: Self organizing semantic maps

		Dove	Hen	Duck	Goose	Owl	Hawk	Eagle	Fox	Dog	Wolf	Cat	Tiger	Lion	Horse	Zebra	Cow
is	Small	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0
	Medium	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
	Big	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
has	2 legs	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
	4 legs	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
	Hair	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
	Hooves	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
	Mane	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0
	Feathers	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
likes to	Hunt	0	0	0	0	1	1	1	1	0	1	1	1	1	0	0	0
	Run	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0
	Fly	1	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0
	Swim	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

SOM:
16 animals,
13 attributes
10 x 10 neurons
after test with
animal codes.

Dog	.	.	Fox	.	.	Cat	.	.	Eagle
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	Owl
.	.	.	.	.	.	Tiger	.	.	.
Wolf	.	.	.	.	.	.	.	.	Hawk
.	.	.	Lion	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	Dove
Horse	.	.	.	.	.	.	.	.	.
.	.	.	.	Cow	.	.	Hen	.	Goose
Zebra	.	.	.	.	.	.	Duck	.	.

Feature map containing labeled neurons with strongest responses to their respective outputs.

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_a \end{bmatrix} \quad \mathbf{x}_a \; as\; above, \mathbf{x}_s \; as\; 0.2 \cdot \delta_{i,animal}$$

Ex3: Self organizing semantic maps

hunters

Semantic map: Each neuron is marked by the particular animal for which it produces the best response.

Dog	Dog	Fox	Fox	Fox	Cat	Cat	Cat	Eagle	Eagle
Dog	Dog	Fox	Fox	Fox	Cat	Cat	Cat	Eagle	Eagle
Wolf	Wolf	Wolf	Fox	Cat	Tiger	Tiger	Tiger	Owl	Owl
Wolf	Wolf	Lion	Lion	Lion	Tiger	Tiger	Tiger	Hawk	Hawk
Wolf	Wolf	Lion	Lion	Lion	Tiger	Tiger	Tiger	Hawk	Hawk
Wolf	Wolf	Lion	Lion	Lion	Owl	Dove	Hawk	Dove	Dove
Horse	Horse	Lion	Lion	Lion	Dove	Hen	Hen	Dove	Dove
Horse	Horse	Zebra	Cow	Cow	Cow	Hen	Hen	Dove	Dove
Zebra	Zebra	Zebra	Cow	Cow	Cow	Hen	Hen	Duck	Goose
Zebra	Zebra	Zebra	Cow	Cow	Cow	Duck	Duck	Duck	Goose

birds

peaceful species

Ex 4: Real-World applications

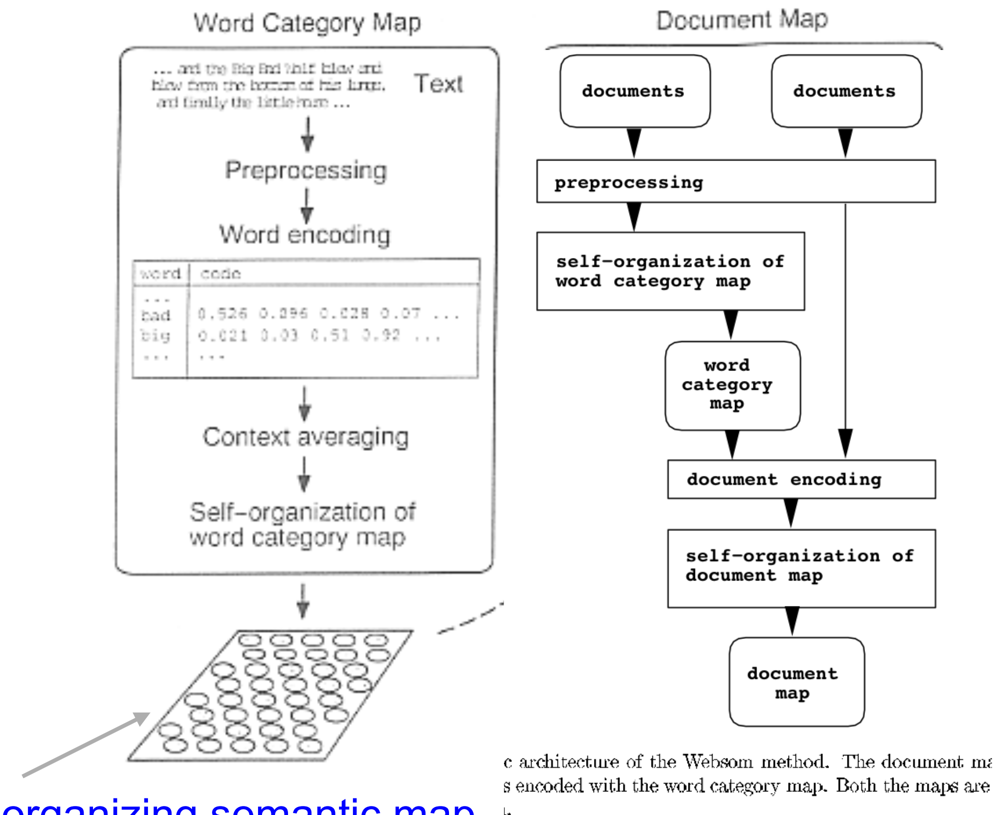
- See http://www.cis.hut.fi/research/som-research.

- WEBSOM: http://www.cis.hut.fi/websom

  Self-organizing maps of document collections.

  - Goal:
    Automatically order and organize arbitrary free-form textual document collections to enable their easier browsing and exploration.

Ex4: WEBSOM to classify docs



All words of document are mapped into the word category map

Histogram of “hits” on it is formed

Self-organizing semantic map.
15x21 neurons
Interrelated words that have similar contexts appear close to each other on the map

Self-organizing map.
Largest experiments have used:

- word-category map
  315 neurons with 270 inputs each
- Document-map
  104040 neurons with 315 inputs each

- Training done with 1124134 documents

Examples for some clear categories of words

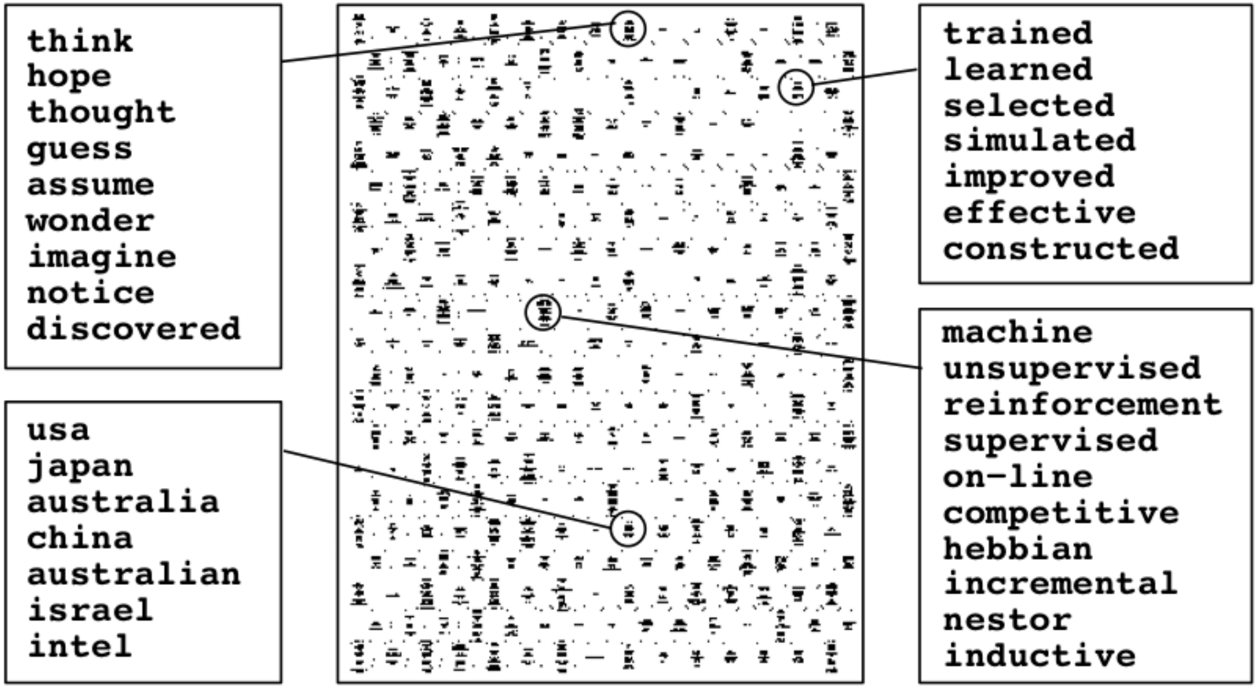
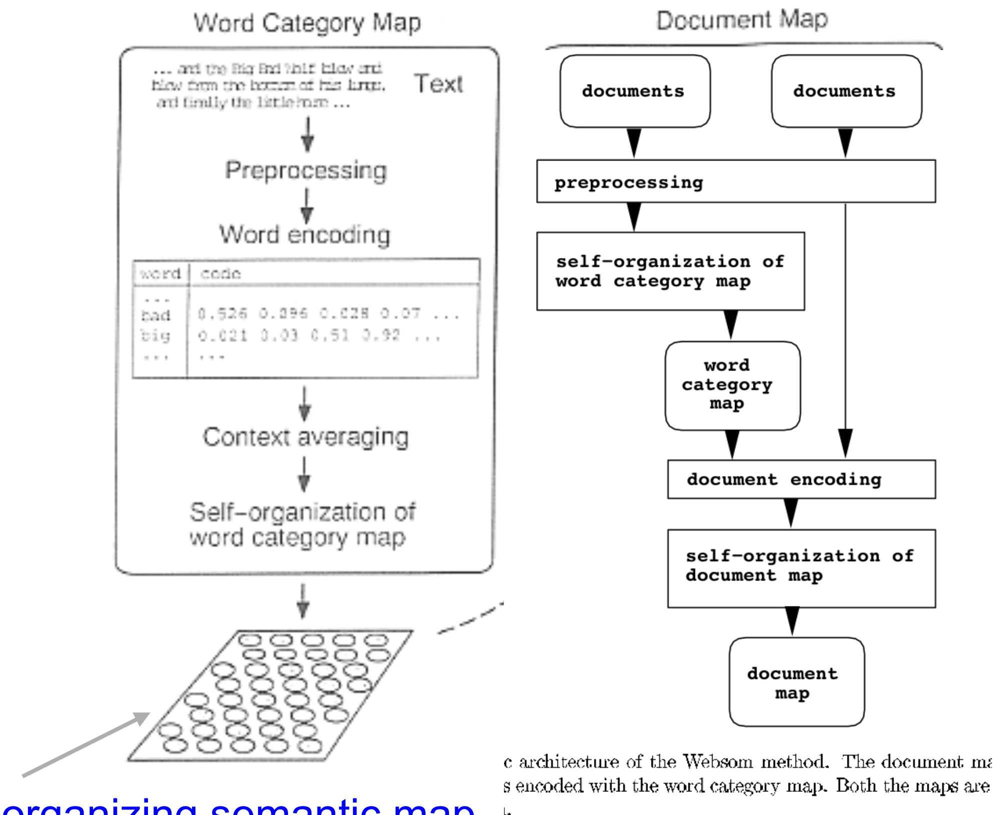


Figure 2: Examples of some clear "categories" of words on the word category map of the size of 15 by 21 nodes. The word labels of the map nodes have been shown with a tiny font on the map grid, and four nodes have been enlarged in the insets.

Ex4: WEBSOM to classify docs



All words of document are mapped into the word category map

Histogram of "hits" on it is formed

Self-organizing semantic map.
15x21 neurons
Interrelated words that have similar contexts appear close to each other on the map

Self-organizing map.
Largest experiments have used:

- word-category map
  315 neurons with 270 inputs each
- Document-map
  104040 neurons with 315 inputs each

- Training done with 1124134 documents