Research on Humanities and Social Sciences
ISSN 2224-5766 (Paper) ISSN 2225-0484 (Online)
Vol.10, No.6, 2020

A Study of Self- Organizing Maps(SOM) Neural Network Using Matlab

Mahabad Abdula Sultan
Department of Information Technology. Erbil technology Institute, Erbil Polytechnic University
Mahabad street 64,Erbil,Iraq

Abstract
Kohonen's Self-Organizing Map (**SOM**) is one of the most popular artificial neural network algorithms.
Bing unsupervised neural network , Self Organizing Maps(**SOM**) have applications in different fields such as speech recognition, image processing and so on . This project includes a study of Self Organizing Maps neural network using **MATLAB** The structure, characteristics, implementation, applications and testing of this neural network for styles of one dimension and two dimensions have been considered It includes finding out the functions used for topology, and then finding out the distance function for this network throughout illustrative examples. Neural network implementation consists of three stages: Initialization (creation), Training, and Simulating. Explanation of the neighborhood concept is done. **MATLAB** software is used to perform how creating, training and simulating of a Self Organizing Map. Creation process consists of choosing a network parameters, plotting the results, then illustrating and identifying all functions used to create a Self Organizing Map. Training consists of weight initialization and weight vector creation. Simulating means testing the neural network using the initialization parameters and training vector created in the past two stages. The Study includes also four tests: The first test is used manual calculation procedure of the network mathematically. The other three tests are procedures for different applications using **MATLAB** language. It becomes evident from the graphs of the results that it's essential to have the weight vectors for the coordination field greater than the density of input vectors in the case of employing the network to seclude the training styles in output cells.
Keywords: Neural Network, Simulating, Self- Organizing Maps, Competitive layer, training
DOI: 10.7176/RHSS/10-6-01
Publication date: March 31st 2020

Introduction

1.1- Motivation for the work:
neural nets are a powerful tool for modeling problems for which the explicit from of the relationships among certain variables is not known
The need of knowledge for the specific feature and applications of the different types of neural networks.
Self organized map is one of the most important and widely used neural network.
Self organized map is a simple equivalent way for neural system of human to save information in a certain style

1.2- Literature survey:

1. Timo Honkela, and Ari M. Vepsäläinen (1991): (Interpreting imprecise expressions: experiments with Kohonen's Self-Organizing Maps and associative memory.)

Kohonen's Self-Organizing Map (SOM) is used to model the impreciseness of natural language interpretation. The author designed the study and conducted the practical experiments related to the SOM. Ari M. Vepsäläinen provided his expertise of associative memories

2. Timo Honkela (1993): (Neural nets that discuss: a general model of communication based on self-organizing maps)

The basic idea is to provide a model of adaptation that leads to intersubjectivity in interpretation of natural language expressions and to coherence in communication.

3. Timo Honkela, Ville Pulkki, and Teuvo Kohonen (1995): (Contextual relations of words in Grimm tales analyzed by self-organizing map.)

deals with the analysis of contextual relations of words. The work was based on the principle of self-organizing semantic maps presented by Ritter and Kohonen in 1989 in *Biological Cybernetics*. In their article, artificially generated, syntactically correct and meaningful short phrases were used as the input to the SOM. the method was refined and the input consisted of segments of text from a natural corpus.

4. Timo Honkela (1997): (Self-Organizing Maps of words for natural language processing applications. presents how the word category maps can be used in natural language processing (NLP) applications. The main emphasis is on the WEBSOM method, and the concrete results presented in the publication are based on the close collaboration of the WEBSOM team. also discusses the nature of the found clusters and the general principles of using the SOM, e.g., in graded classification, following the argumentations set by Ritter and Kohonen in their article in the journal *Biological Cybernetics* in 1989.

5. Timo Honkela (1997)(Learning to understand - general aspects of using Self-Organizing Maps in natural language processing.)

the introduction outlines some limits of the traditional symbolic approaches such as relying on fixed categorizations, problems related to handling language change, and, moreover, subjectivity and context-sensitivity of interpretation. One chapter summarizes the basic principles of using the SOM in natural language interpretation. Finally, the last chapter contains some epistemological considerations.

6. Samuel Kaski, Timo Honkela, Krista Lagus, and Teuvo Kohonen (1996): (Creating an order in digital libraries with self-organizing maps.)

Studies on the WEBSOM full-text analysis method are described

presents a fully unsupervised method with improvements in the document encoding. The original idea of using a two-stage SOM architecture was due to the author. Samuel Kaski was the main originator of the idea of using table lookups for the efficient encoding of the documents and the subsequent convolution. The original document maps were limited to some 10 000 documents at maximum. To facilitate considerably larger maps for up to one million documents, Professor Kohonen later suggested some new methods for shortcut computation. Many ideas and details, as well as implementations and experiments were developed jointly in a team consisting of Samuel Kaski, .

1.3 -Aim of the project
Study the features and applications of the Self organized map neural network
Using Matlab software

1.4 -Outline of the project
This Study contains five chapters:
After this Introduction ,chapter two involves the theoretical background of Neural Networks and Self Organized Map(som)
Based Implementation and application are found in chapter three and chapter four respectively .
Chapter five includes the conclusion and future works
Neural Network & Self Organizing Maps (SOM)

2.1 Artificial Neural Networks
An artificial neural network is an information processing system that has certain performance characteristics in common with biological neural networks. Artificial neural networks have been developed as generalizations of mathematical models of Human cognition or neural biology.
A neural network is characteristics by
It's pattern of connections between the neurons (called architecture).
Its method of determining the weights on the connections (call its training. Or learning algorithm).
Its activation function.

A neural network consists of a large number of simple processing elements called neurons, units, cells, or nodes. Each neuron is connected to other neurons by means of directed communication links, each with an associated weight. The weights represent information being used by the net to solve a problem.

Each neuron has an internal state. Called its activation or activity level. Which is a function of the inputs it has received. Typically, a neuron sends its activation as a signal to several other neurons. It is important to note that a neuron can send only one signal at a time. Although that signal is broadcast to several other neurons.

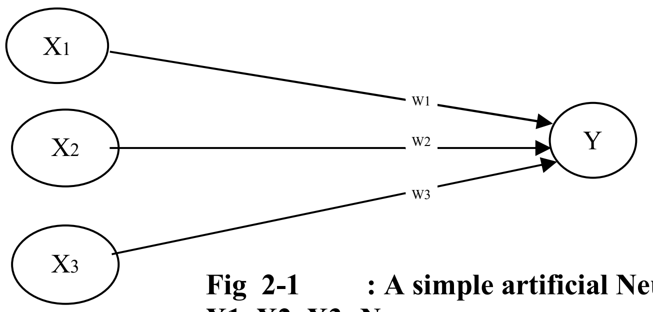


Fig 2-1 : A simple artificial Neuron:
X1, X2, X3: Neurons
Y: Receives inputs from Neurons X1, X2, and X3
W1, W2, W3 the weights on the connections from X1, X2 and X3 to Neuron Y

2 Biological Neuron

There is a close analogy between the structure of a biological neuron (i.e. a brain or nerve cell) and the processing element (or artificial neuron).

A biological Neuron has three types of components that are of particular interest in understanding an artificial neuron: its dendrites, soma and axon. The many dendrites receive signals from other neurons. The signals are electric impulses that are transmitted across synaptic gap by means of a chemical process.

The action of chemical transmitter modifies the incoming signal (typically by scaling the frequency of the signals that are received) in a manner similar to the action of the weights in an artificial neural network.

Some features of artificial neural networks that are suggested by biological neurons are:-

Information processing is local (Although other means of transmission, such as the action of hormones, may suggest means of overall process control).

Memory is distributed:

Long term memory resides in the neuron's synapses or weights.

Short-term memory corresponds to the signals sent by the neurons.

A Synapses' strength may be modified by experience.

Neurotransmitter's for synapses may be excitatory or inhibitory.

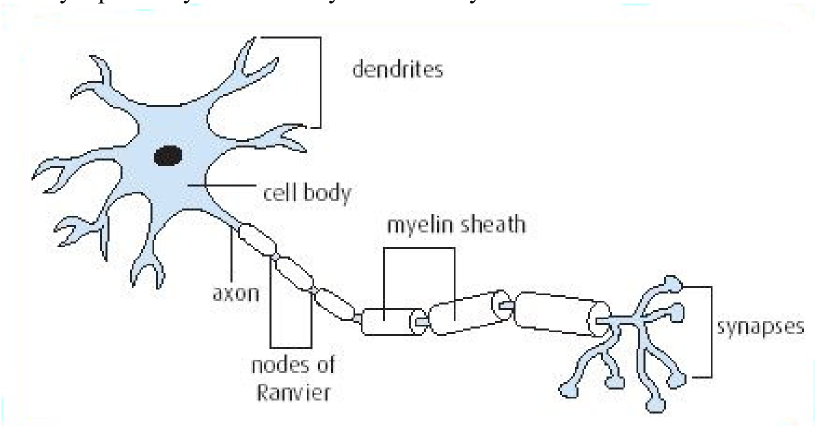


Fig 2-2 :Biological neuron

2.3 The Applications of using Neural nets:-

Adeline Networks Application example Signal Processing:.

Back propagation network example: Control

Neocognitron Network Pattern example: Recognition

NET talk example: Speech Production.

Self Organization Map network example :Speech recognition

2.4 Components of Neural Network Architecture

The Method of setting the weights (training). Activation Functions.

Architecture

It's convenient to visualize neurons as arranged in layers. Typically neurons in the same layer behave in the same manner. Key factors in determining the behavior of a neuron are its activation function and the pattern of weighted connections over which it sends and receives signals.

The arrangement of neurons into layers and the connection patterns within and between layers the net architecture.

Neural nets are often classified as single layer (with no hidden layer) or multi layer (with hidden layer) in determining the number of layers. The input units are not counted as a layer, because they perform no computation.

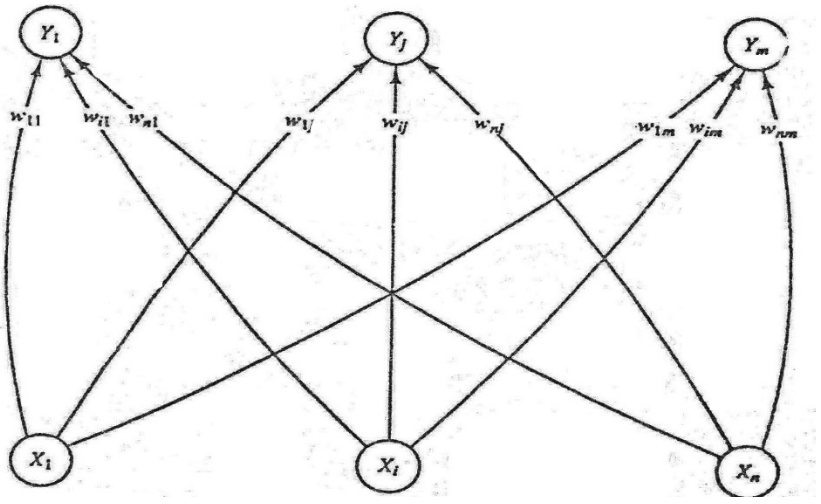


Fig 2- 3: Single Layer

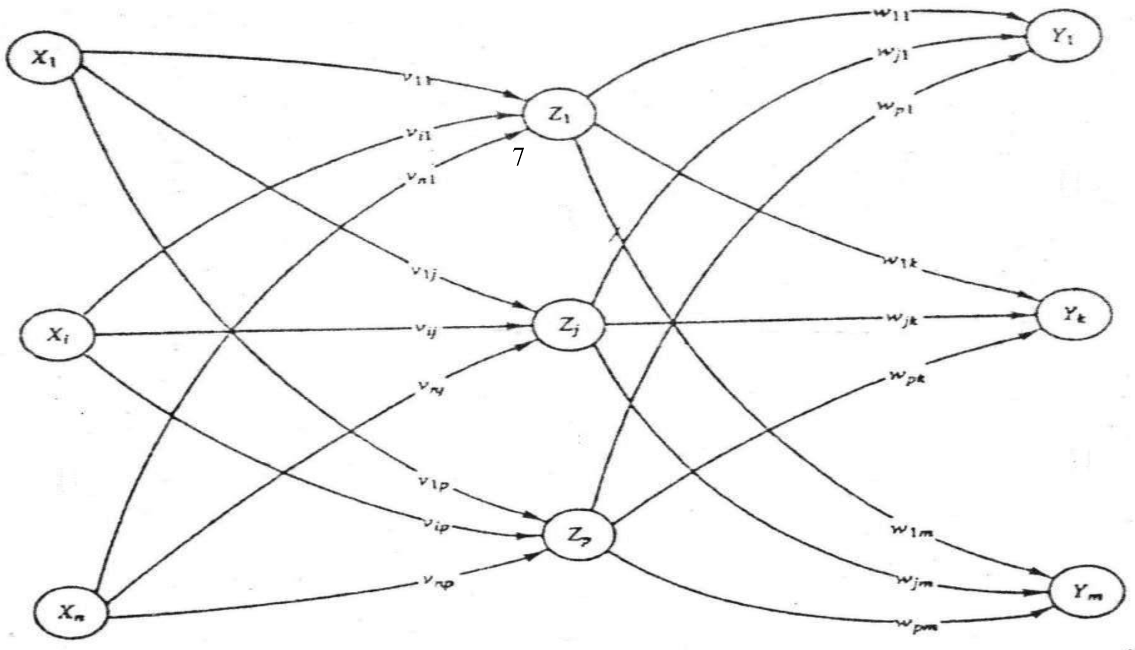


Fig 2-4: Multiple Layers

Competitive layer
The interconnections between neurons in the competitive layer are not shown in the architecture diagram for such nets.
A competitive layer forms a part of a large number of neural networks.
The competitive interconnections have weights –ε the operation of a winner - takes all competition.

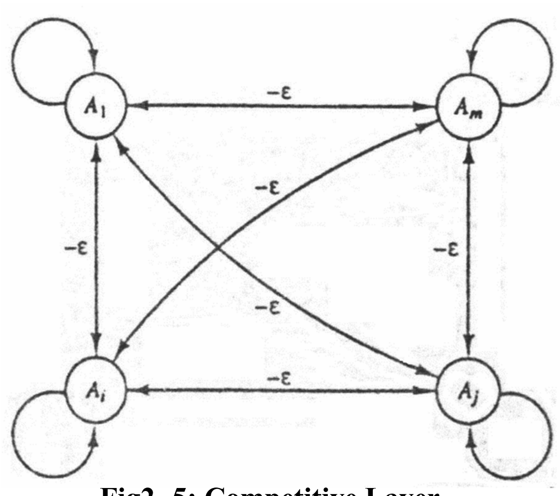


Fig2- 5: Competitive Layer

2.4.2 Setting the weights (the training).
The method of setting the values of the weights (training) as an important distinguishing characteristic of the different nets. For convenience we shall distinguish two types of training - Supervised and unsupervised-
Supervised Training

Training is accomplished by presenting a sequence of training vectors, or patterns. Each with an associated target output vector. The weights are then adjusted according to a learning algorithm. This process is known as supervised training.

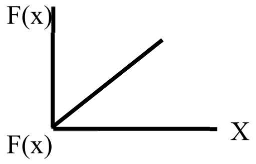
In this neural networks the output is bivalent element say, either 1 (if the input vector belongs to the category) or -1(if it doesn't belong)

Unsupervised Training

Self-organizing neural nets group similar input vectors together without the use training data to specify what a typical member of each group looks like or to which group each factor belongs. A sequence of input vectors is provided. But no target vectors are specified. The net modifies the weights so that the most similar input vectors are assigned to the same output (Or cluster unit) the neural net will produce an exemplar (representative) for each cluster formed. Self-organizing nets.

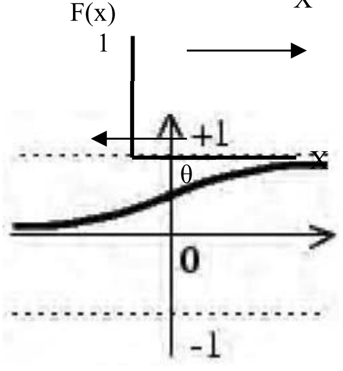
2.4.3 Activation functions
The basic operation of artificial neurons involves summing its weighted input signal and applying an output.
1--Identity function

$F(x) = x$ for all x ---------1



2--Binary step function (with threshold θ)

$$F(x) = \begin{cases} 1 & \text{If } x \geq \theta \\ 0 & \text{If } x < \theta \end{cases} \quad \text{---------2}$$



3--Binary sigmoid

$$f(x) = \frac{1}{1 - e^{-\sigma x}} \quad \text{-- --------3}$$

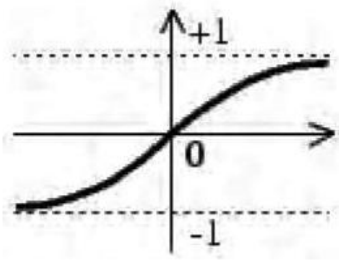
$$f'(x) = \sigma f(x)[1 - f(x)] \quad \text{---------4}$$

σ=steepness parameter

4--Bipolar Sigmoid

$$g(x) = 2f(x) - 1 = \frac{2}{1+e^{-\sigma x}} - 1 \qquad \text{-----------5}$$

$$g'(x) = \frac{\sigma}{2}[1-g(x)][1+g(x)] \qquad \text{---------6}$$



2.5 Topology preserving maps in the brain
The brain is capable of creating maps in its neural structures for the characteristics received as neural signals from perception systems represented by one or more dimensions. Every entity in our surroundings has many characteristics that brain can percept like, color, location, texture.

How can the brain process the multi dimensional signals? And how this signal can be represented by multi dimension in the biological neural structures?

2.6 Mapping of the visual field on the Cortex
The researchers have proved that Human being's brain cortex is distributed to functional groups which consist of billions of neurons connected together by billions of synapses.

These functional groups may be arranged in organized order, some of the most important of them are: Somatosensory Cortex, Auditory Cortex, Visual Cortex, Motor Cortex, for example the visual cortex is divided to several cell groups; each of them has its certain perceptional function for the visual signal, and it's located in the Visual Cortex at the rear part of the brain.

The visual data is diagramed in the Visual Cortex (Brain Cortex) as two dimensional small areas as in the figure (2-1).
From the figure (2-1) we note the followings:-
The adjacent area in the visual field (retina) is processed by adjacent areas in the brain cortex.
The surface of visual cortex in the brain is specialized to process the signals coming from center of visual field, and it being processed in higher resolution.

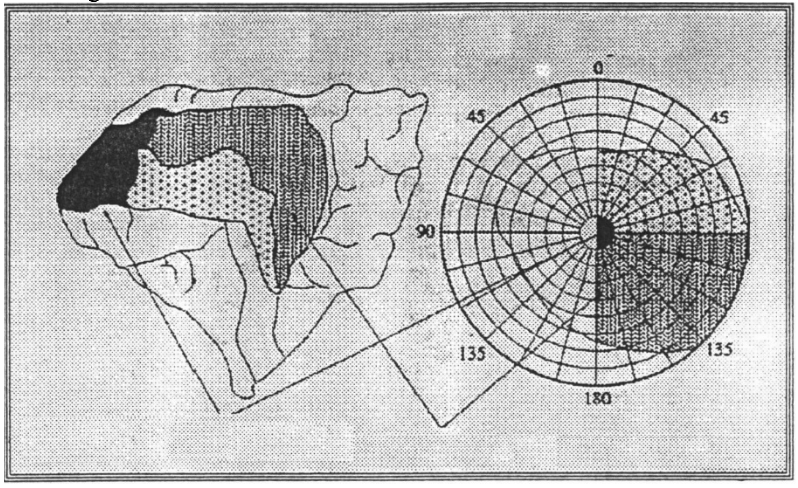


Fig 2-6: (Brain Cortex)

Then the cortex is discriminated by its harmony and unison in saving information
This way of representing the information carries two characteristics.
In every phase of representation, the upcoming information will be represented in a proper context or in its proper spot.

The cells related to information closed to each others are saved in vicinity to each others in away which is able to interact with each others through synopses.

2.7 Historical Nets of artificial neural networks.
The begging of neural nets (1940)
Warren McCulloch and Walter Pitts designed what are generally regarded as the first neural networks (McCulloch & Pitts. 1943) these researchers recognized that combining many simple neurons into neural system was the source of increased computational power.
Hebb Learning

Donlad Hebb. A psychological at McGill University designed the first learning law for artificial neural networks (Hebb 1949).

The first golden age of neural networks (1950 and 1960)

A.Perceptrons

Together with several other researchers {Block. 1962. Minsky and Papert. (Originally published 1969)}. Frank Rosenblatt (1958, 1959, 1962) introduced and developed a large class of artificial neural networks called perceptron's.

B.Adaline

Bernard Widrow and his student. Marcian (Ted) Hoff {Widrow & Hoff.1960} developed a learning rule (which usually either bears their names. Or is designated the least mean squares or delta rule) that is closely related to the perceptron learning rule.

The name Adeline interpreted as either adaptive linear Neuron or Adaptive Linear system is often given to these nets.

The quite years (1970). Kohonen, Andreson, Grossberg, Carpenter

Kohonen

The early work of Teuvo Kohonen (1972) of Helsinki Unversity of technology. Dealt with associative memory neural nets. His more recent work (Kohonen 1982) has been the development of self-organizing future maps that use a topological structure for the cluster units. These nets have been applied to speech recognition (For Finnish and Jappenes words) (Kohonen Torkkola. Shozakai. Kangas. & Venta 1987: Kohonen 1988) the solution of traveling Salesman problem (Angeniol. Vaubois. & Le Texier, 1988) and musical composition (Kohonen 1989).

Renewed Enthusiasm 1980

A. Back propagation

A method for propagating information about errors at the output a unit back to the hidden unites had been discovered in the previous decade (Werbos, 1974) but had not gained wide publicity. This method was also discovered independently by David Parker (1985) and by LeCun (1986) before it became widely known.

B.Hopfield nets

Hopfield has developed a number of neural networks based on fixed weights and adaptive activation (Hopfield 1982-1984. Hopfield and Tank, 1985 – 1986).

C. Neocognitron

Kunihiko Fukushima and his colleagues at NHK Laboratories in Tokyo have developed a series of specialized neural nets for character recognition. One example of such a net called a neogognitron .

D .Boltzmann Machine

A number of researchers have been involved in the development of nondeterministic neural nets, that is, nets in which weights or activations are changed on the bases of a probability density function.

2.8 Self Organizing Maps Neural Networks

The self organizing map developed by Kohonen. Which groups the input data into clusters. A common use for unsupervised learning adaptive resonance theory nets. In this form of learning the units that update their weights do so by forming a new weight vector that is linear combination of the old weight vector and the current input vector.

Typically, the unit whose weight vector was closest to the input vector is allowed to learn. The weight update for output (or cluster) unit (j) is given is

$W_j\,(new) = W_j\,(old) + \alpha\,[p - W_j\,(old)]$--------------7

Where (p) is the input vector, (Wj) is the weight vector for unit j (which also the jth column of the weight matrix) and (α) the learning rate, decreases as learning proceeds.

Self organizing network is a simple equivalent way for neural system of human to save information in a certain style.

The basic goal of (SOM) is transferring the input factors which has a random dimensional or multiple dimensional to one dimensional or two dimensional. Thus the neurons in competitive layers recognize like one dimensional or two dimensional

Self-organizing feature maps (SOFM) learn to classify input vectors according to how they are grouped in the input space. They differ from competitive layers in that neighboring neurons in the self-organizing map learn to recognize neighboring sections of the input space. Thus, self-organizing maps learn both the distribution (as do competitive layers) and topology of the input vectors they are trained on.

The neurons in the layer of an SOFM are arranged originally in physical positions according to a topology function. The functions gridtop, hextop or randtop can arrange the neurons in a grid, hexagonal, or random topology. Distances between neurons are calculated from their positions with a distance function. There are four distance functions, dist, boxdist, linkdist and mandist. Link distance is the most common. These topology and distance functions are described in detail later in this section.

2.9 Architectural of SOM

(SOM) Consists of two layers they are:

Input layer and output layer (also called competitive layer).

Every neuron in the input layer connected to each neuron in output layer such (Feed forward connection) it means that the input data spread out through network from input layers toward output layers.

The input layers don't perform any process, only receive input vectors from output world and send out to processor elements in the output layers for this network.

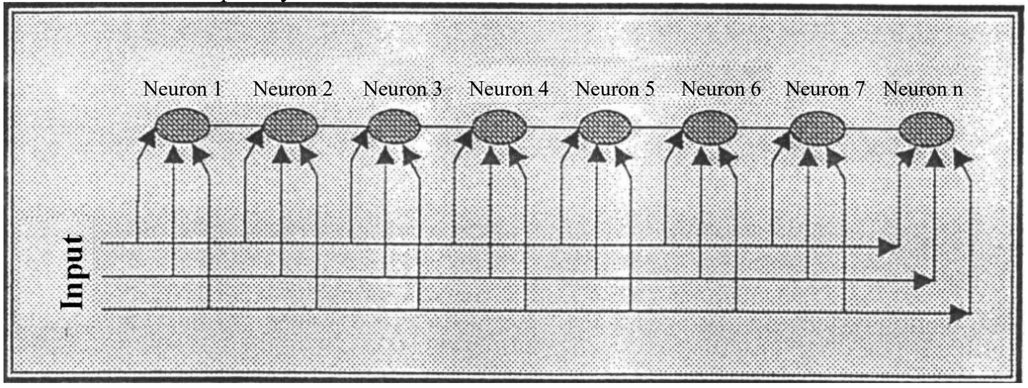


Fig 2-7: The structure of SOM (one dimension)

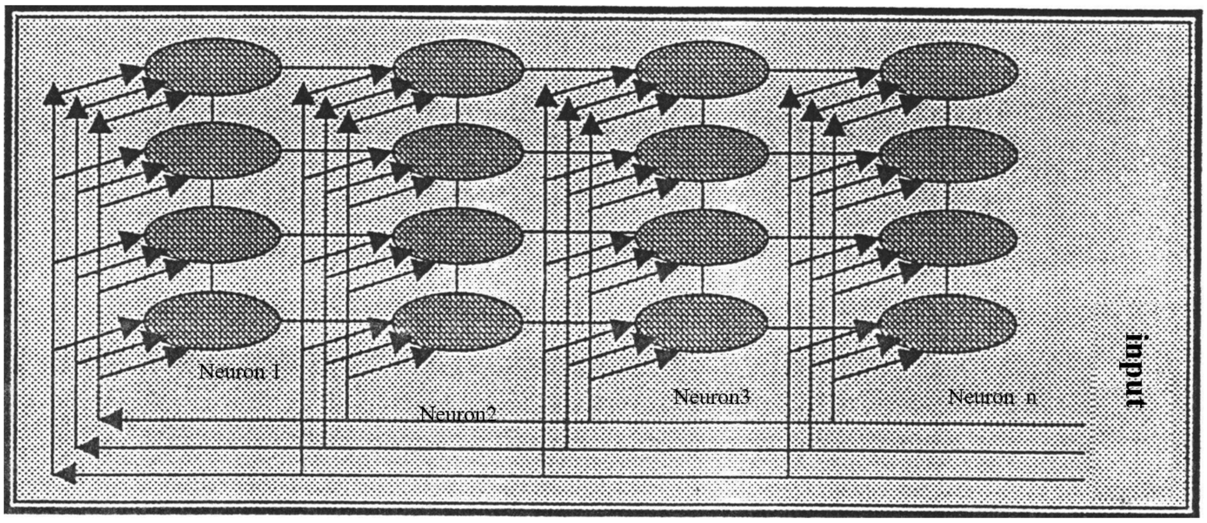


Fig 2-8 The structure of SOM (two dimensions)

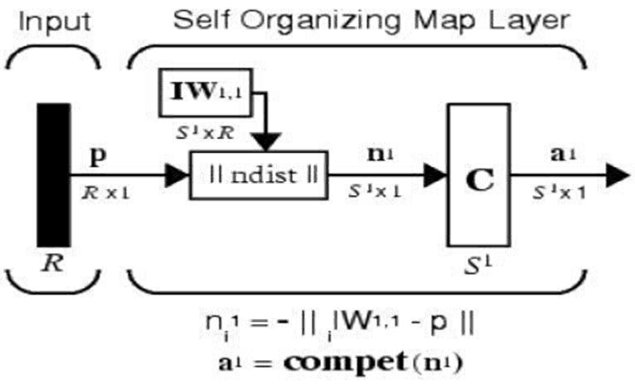

Input
Self Organizing Map Layer
$IW^{1,1}$
$S^1 x R$
p
R x 1
|| ndist ||
n^1
$S^1 x 1$
C
a^1
$S^1 x 1$
R
S^1
$n_i^1 = - || {}_iIW^{1,1} - p ||$
$a^1 = \textbf{compet}(n^1)$

Fig 2-9 Architecture of SOM

R-- input elements

S -- Number of rows (neurons)

W -- S x R weight matrix

P -- R x 1 matrix of 1 input (column) vectors and returns the S x 1 matrix of negative vector distances.
C compete transfer function 2.10 Neighborhood

The idea of neighborhood of the unit designated by (*) of radii R=2, 1, 0 in a one, dimensional topology (with 10 cluster units) are shown in figure 2-10

* * * {* (* [#] *) *} * *

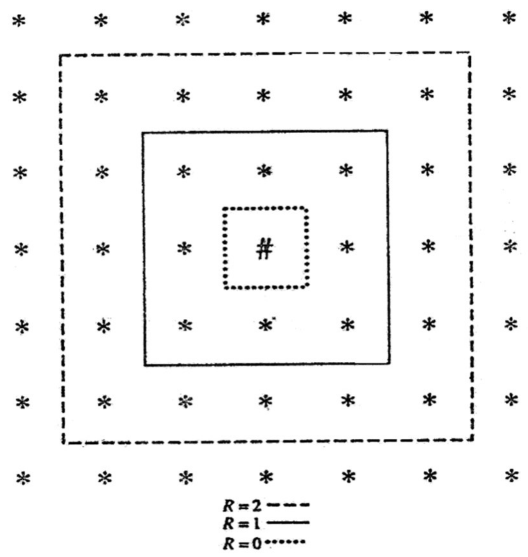
Fig 2-10: one dimensional topology { } R= 2( ) R=1 [ ] R=0

The neighborhood of radii R=2, 1, 0 are shown in figure (2-11) for a rectangular grid and in figure (2-12) for hexagonal grid (each with 49 units) in each illustration. By winning unit is indicated by the symbol (#) and the other units are denoted by

Note that each unit has eight nearest neighbors in the rectangular grid.

But only six in the hexagonal grid. Winning units that are close to the edge of the grid will have some neighborhoods that have fewer units that shown in the respective figure (Neighborhoods do not 'warp around 'from one side of the grid to the other 'missing' units are simply ignored).

fig 2-11; neighborhood for a rectangular grid



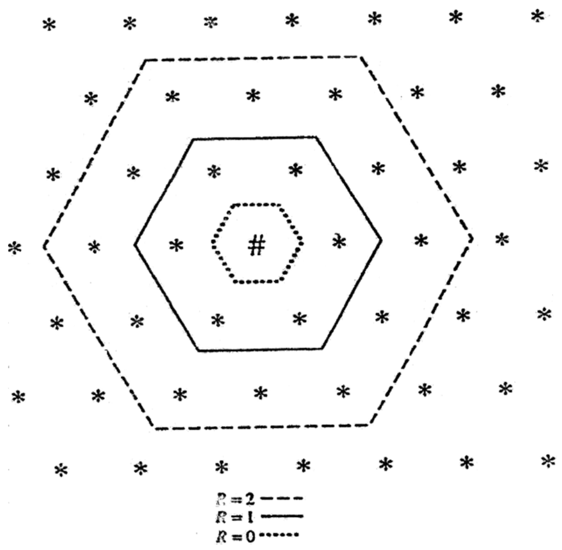


Fig 2-12; neighborhood for hexagonal grid

2.11 Algorithm of som

Step 0:

Initialize weights wij (i=No. of neurons in competitive layer, j=no. of values of input vectors).

Set topological neighborhood parameters(grid,hix,rand, radius=from n to 0)

Set learning rate parameters (α) [α=0.9]

```
Step1:
While stopping condition is false do steps 2-8
Step 2:
For each input vector P do step 3-5
Step 3:

For each j compute for input
$D(j) = \sum_i (wij - Pi)^2$ -------------8
Step 4:
Find index j such that D(j) is a minimum
Step 5:

For all units j within a specified neighborhood of j and for all i
Wij(new) = wij(old)+ α (Pi-wij(old))----------9
Step 6:
Update learning rate
α(t+1)= α (t)*0.5------------10
where t is the discrete-time index of the variables
Step 7:
Reduce radius of topological neighborhood at specified times
Step 8:
Test stopping condition which
```

Represents by the no of steps or nets stability and this done by Comparing the average distance(AD) of the net and with the value of (Tolerance distance(TD) during initializing if the average distance less or equal to tolerance distance

```
If   AD <=TD  then  nets stability
else     next step
```

It means the training completed but if the Average distance is more than Tolerance distance return to the next step and the average distance founded by:

$$D = \frac{\sqrt{\sum (di)^2}}{P} \quad \text{----------------11}$$

di = (P-w)^2 ---------------12

2.12 Determining the size of the network

To determine the size of the network we should know that the increment in the number of the neurons causes decrement in the error rate resulting in the net in addition it will get better classification for the inputs.

In the other hand a large number of neurons consuming more time in training process.

2.13 Disadvantage of SOM

The disadvantage of SOM is the net performs too much processing because at the first steps of training much of neurons adjust their weights in a correlated scheme.

For each step of training (Input one set of train) the net calculates the distances of this set and all neurons' weight in the competitive layers.

2.14 Advantage of SOM

One the main characteristics of SOM is the capability of connecting neurons to a group of weights in a way that achieves geometric order (Topology of net) and it can be assured for a net organization by calculating the Euclidian distances among the neurons.

Then the measured distances must comply with the original topology of this net.

The net called organized if the Euclidian distances among neurons are proportional to the geometric distances among neurons.

3.1 Matlab Based Implementation of SOM

The processing in som

includes four main elements

Initialization: it's the process of giving random small values to the weights in the Network.

Competition: After every input into the network the cells in the output layer calculate the activation level according to a certain function depending on competition basis, the winner cell is the one which has the higher activation level.

Cooperation: The winner cell determines the space which contains the neighbor cells in order to amend the weights of these cells.

Adaptive: increase the level of activation for winning cell resulting from competition function through modifying this cell weight to improve it's response for the input vector itself.

3.2 Neural Network Implementation includes three stages using MATLAB:-

1-Creating (Initialization)
2-training
3-Simlating

3.2.1 Creating a Self Organizing MAP Neural Network (newsom)

You can create a new SOFM network with the function newsom. This function defines variables used in two phases of learning:
Ordering-phase learning rate
Ordering-phase steps
Tuning-phase learning rate
Tuning-phase neighborhood distance
These values are used for training and adapting.
Consider the following test :

Suppose that we want to create a network having input vectors with two elements that fall in the range 0 to 2 and 0 to 1 respectively. Further suppose that we want to have six neurons in a hexagonal 2-by-3 network. The code to obtain this network is

```
 net = newsom([0 2; 0 1] , [2 3]);
```

Suppose also that the vectors to train on are

```
P = [.1 .3 1.2 1.1 1.8 1.7 .1 .3 1.2 1.1 1.8 1.7;...
 0.2 0.1 0.3 0.1 0.3 0.2 1.8 1.8 1.9 1.9 1.7 1.8]
```

We can plot all of this with

```
plot(P(1,:),P(2,:),'.g','markersize',20)
hold on
plotsom(net.iw{1,1},net.layers{1}.distances)
hold off
```

To give

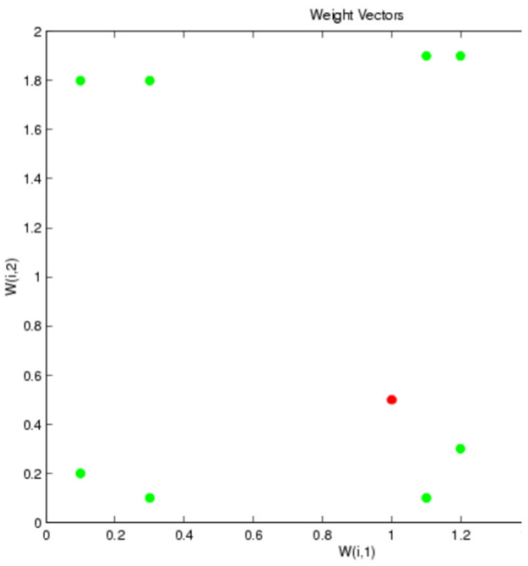


Fig 3-1: plotsom of fuzzy gray spots around the perimeter(P)

Training of som

The various training vectors are seen as fuzzy gray spots around the perimeter of this figure(3-1).
The initialization for newsom is midpoint. Thus, the initial network neurons are all concentrated at the black spot at (1, 0.5).

Learning occurs according to the learnsom learning parameter, shown here with its default value.

LP.order_lr	0.9	Ordering-phase learning rate
LP.order_ste	1000	Ordering-phasing steps
LP. tune_lr	0.02	Tuning-phase learning rate
LP. tune_nd	1	Tuning-phase neighborhood distance

Thus, feature maps, while learning to categorize their input, also learn both the topology and distribution of their input.

We can train the network for 1000 epochs with

```
net.trainParam.epochs = 1000;
net = train(net,P);
```

This training produces the following plot.

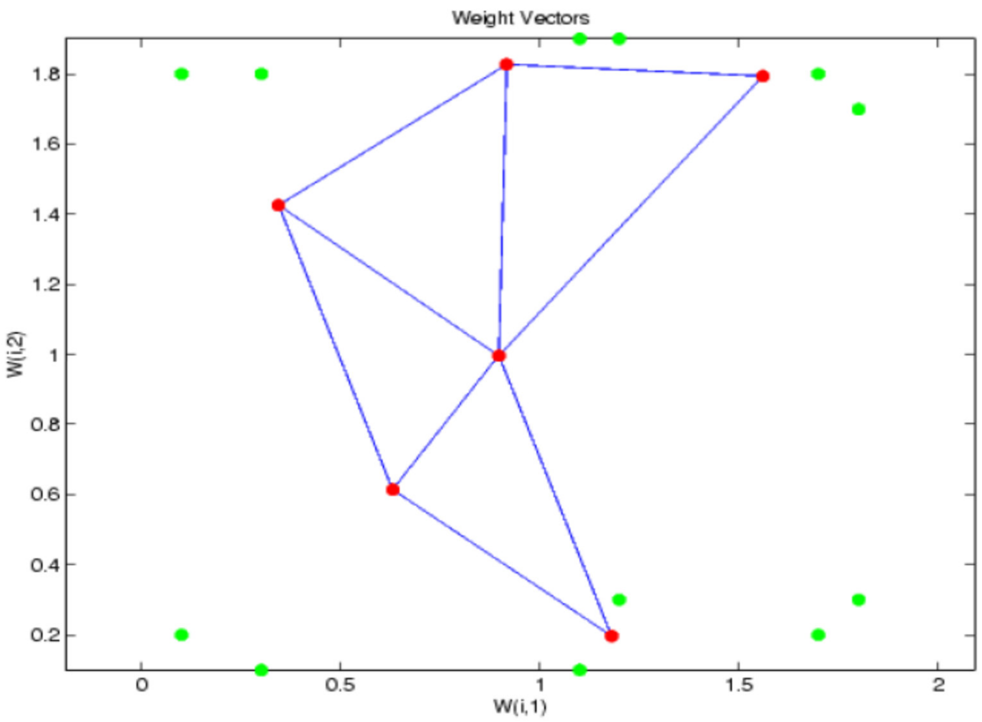


Fig 3-2:plotsom of training(net) after 1000 epoch

3.2.3 Simulating

When simulating a network, the negative distances between each neuron's weight vector and the input vector are calculated (negdist) to get the weighted inputs.

The weighted inputs are also the net inputs (netsum). The net inputs compete (compet) so that only the neuron with the most positive net input will output a 1

```
p = [1;0];
a = sim(net,p)
a =   (5,1)       1
```

We can see that the neurons have started to move toward the various training groups. Additional training is required to get the neurons closer to the various groups.

As noted previously, self-organizing maps differ from conventional competitive learning in terms of which neurons get their weights updated. Instead of updating only the winner, feature maps update the weights of the winner and its neighbors.

The result is that neighboring neurons tend to have similar weight vectors and to be responsive to similar input vectors

Applications and Results

4.1 Mathematical Test

Kohonen self-organizing map (SOM) to cluster four vectors

Let the vectors to be clustered be

(1, 1, 0, 0): (0, 0, 0, 1): (1, 0, 0, 0): (0, 0, 1, 1)

The maximum number of clusters to be formed is

m= 2

Supposed the learning rate (geometric decrease) is

α (0)= 0.6

α (t+1) = 0.5 α(t)

With only two clusters available, the neighborhood of node J (step 4) is set to that only one cluster updates its weights at each step (i.e. R=0)

Step 0 initial weight matrix:

$$\begin{bmatrix} 0.2 & 0.8 \\ 0.6 & 0.4 \\ 0.5 & 0.7 \\ 0.9 & 0.3 \end{bmatrix}$$

Initial radius

R = 0

Initial learning rate:

α (0)= 0.6

Step 1 Begin training.

Step 2 For First vector (1, 1, 0, 0) do steps 3-5.

Step 3 by equ.(8) D(1) = (0.2-1)2 + (0.6-1)2 +(0.5-0)2 + (0.9-0)2 = 1.86

D(2) = (0.8-1)2 + (0.4-1)2 +(0.7-0)2 + (0.3-0)2 = 0.98

Step 4 the input vector is closest to output node 2 so

J= 2

Step 5 the weights on the winning unit are update by equ.(9)

W12(new) = W12(old)+0.6(P- W12(old))

= 0.4 W12(old) + 0.6 Pi

This gives the weight matrix

$$\begin{bmatrix} 0.2 & 0.92 \\ 0.6 & 0.76 \\ 0.5 & 0.28 \\ 0.9 & 0.12 \end{bmatrix}$$

Step 2 for the second vector (0, 0, 0, 1) do steps 3-5

Step 3 D(1) = (0.2-0)2 + (0.6-0)2 +(0.5-0)2 + (0.9-1)2 = 0.66

D(2) = (0.92-0)2 + (0.76-0)2 +(0.28-0)2 + (0.12-1)2 = 2.2768

Step 4 the input vector is closest to output node 1 so

J= 1

Step 5 update the first column of the weight matrix

$$\begin{bmatrix} 0.08 & 0.92 \\ 0.24 & 0.76 \\ 0.20 & 0.28 \\ 0.96 & 0.12 \end{bmatrix}$$

Step 2 for the third vector (1, 0, 0, 0) do steps 3-5

Step 3 D(1) = (0.08-1)2 + (0.24-0)2 +(0.2-0)2 + (0.96-0)2 = 1.8656

D(2) = (0.92-1)2 + (0.76-0)2 +(0.28-0)2 + (0.12-0)2 = 0.6768

Step 4 the input vector is closest to output node 2 so

J= 2

Step 5 update the first column of the weight matrix

$$\begin{bmatrix} 0.08 & 0.968 \\ 0.24 & 0.304 \\ 0.20 & 0.112 \\ 0.96 & 0.048 \end{bmatrix}$$

Step 2 for the four vector (0, 0, 1, 1) do steps 3-5

Step 3 D(1) = (0.08-0)2 + (0.24-0)2 +(0.2-1)2 + (0.96-1)2 = 0.7056
D(2) = (0.968-0)2 + (0.304-0)2 +(0.112-1)2 + (0.048-1)2 = 2.724

Step 4 the input vector is closest to output node 1 so
J= 1

Step 5 update the first column of the weight matrix

$$\begin{bmatrix} 0.032 & 0.968 \\ 0.096 & 0.304 \\ 0.680 & 0.112 \\ 0.984 & 0.048 \end{bmatrix}$$

Step 6 Reduce the learning rate : by equ.(10)
α = 0.5 (0.6) = 0.3

The weight update equations are now
W12(new) = W12(old)+0.3(P- W12(old))
= 0.7 W12(old) + 0.3 Pi

The weight matrix after the second epoch of training is

$$\begin{bmatrix} 0.016 & 0.980 \\ 0.047 & 0.360 \\ 0.630 & 0.055 \\ 0.999 & 0.024 \end{bmatrix}$$

Modifying the adjustment procedures for the learning rate so that it decreases geometrically from 0.6 to 0.01 over 100 iterations (epochs) gives this converging matrix

$$\begin{bmatrix} 0.0 & 1.0 \\ 0.0 & 0.5 \\ 0.5 & 0.0 \\ 1.0 & 0.0 \end{bmatrix}$$

The first column of which is the average of the two vectors placed in cluster 1 and the second column of which is the average of the two vectors placed in cluster 2

4.2 Programming Test 1: A One-dimensional Self-organizing Map

Neurons in a 2-D layer learn to represent different regions of the input space where input vectors occur. In addition, neighboring neurons learn to respond to similar inputs, thus the layer learns the topology of the presented input space.
Here 100 data points are created on the unit circle.
A competitive network will be used to classify these points into natural classes.

```
angles = 0:0.5*pi/99:0.5*pi;
P = [sin(angles); cos(angles)];
plot(P(1,:),P(2,:),'+r')
```

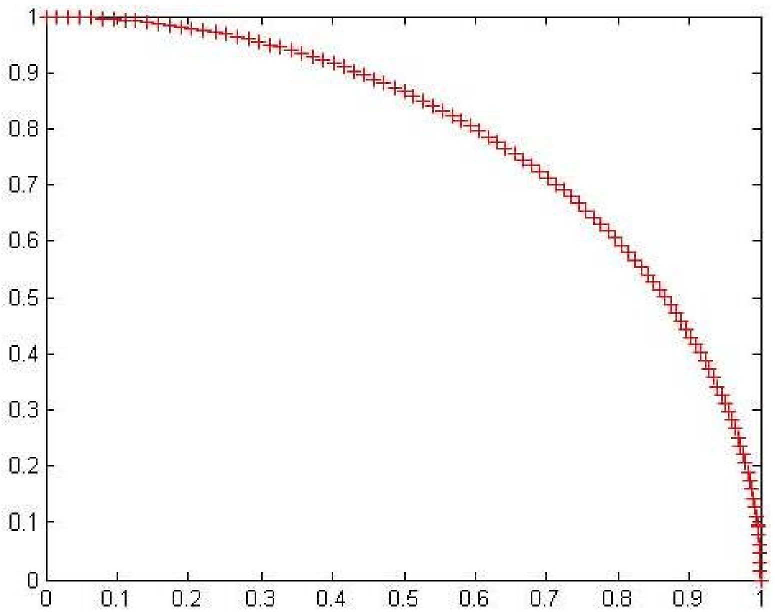


Fig 4-1: Plot of (100)data points are created on theunit circle.

The map will be a 1-dimensional layer of 10 neurons.

The first argument specifies two inputs, each with a range of 0 to 1. The second determines the network is one dimensional with 10 neurons.

```
net = newsom([0 1;0 1],[10]);
```

Specify the network is to be trained for 10 epochs and use TRAIN to train the network on the input data P:

```
net.trainParam.epochs = 10;
net = train(net,P);
TRAINR, Epoch 0/10
TRAINR, Epoch 10/10
TRAINR, Maximum epoch reached.
```

Now we plot the trained network with PLOTSOM.

The black dots are the neuron's weight vectors, and the lines connect each pair within a distance of 1.

```
plotsom(net.iw{1,1},net.layers{1}.distances)
```

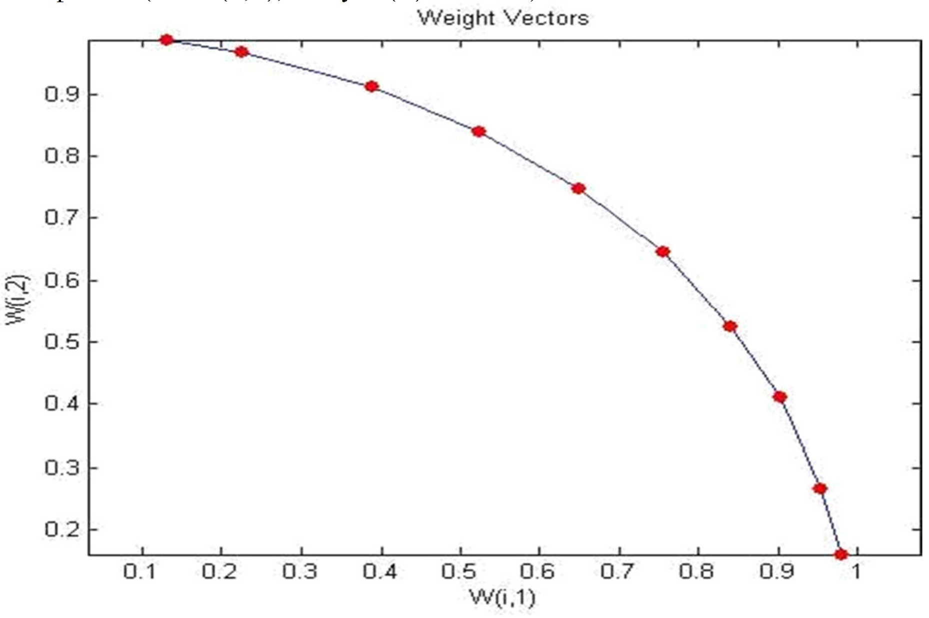


Fig 4-2: plot the trained network with PLOTSOM after (10) epochs

The map can now be used to classify inputs, like [1; 0]:

Either neuron 1 or 10 should have an output of 1, as the above input vector was at one end of the presented input space. The first pair of numbers indicates the neuron, and the single number indicates its output.

```
 p = [1; 0];
a = sim(net,p)
a =
   (10, 1)        1
```

4-3 programming Test 2:A Two-dimensional Self-organizing Map
This self-organizing map will learn to represent different regions of the input space where input vectors occur. In this demo, however, the neurons will arrange themselves in a two-dimensional grid, rather than a line. :
We would like to classify 1000 two-element vectors occurring in a rectangular shaped vector space.

```
P = rands(2,1000);
plot(P(1,:),P(2,:),'+r')
```

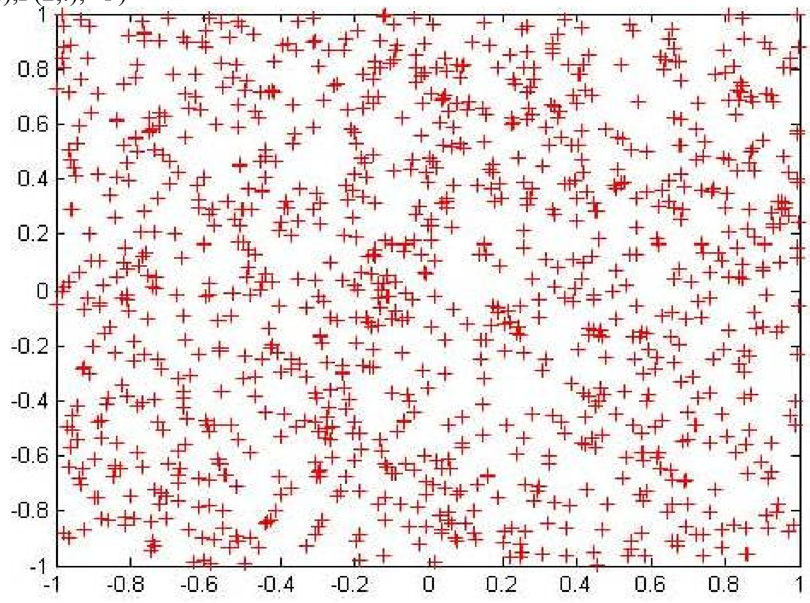


Fig 4-3: Plot of 1000Two-element vectors

We will use a 5 by 6 layer of neurons to classify the vectors above. We would like each neuron to respond to a different region of the rectangle, and neighboring neurons to respond to adjacent regions. We create a layer of 30 neurons spread out in a 5 by 6 grid:

```
 net = newsom([0 1; 0 1],[5 6]);
```

We can visualize the network we have just created with PLOTSOM.

Each neuron is represented by a black dot at the location of its two weights. Initially all the neurons have the same weights in the middle of the vectors, so only one dot appears.

```
plotsom(net.iw{1,1},net.layers{1}.distances)
```

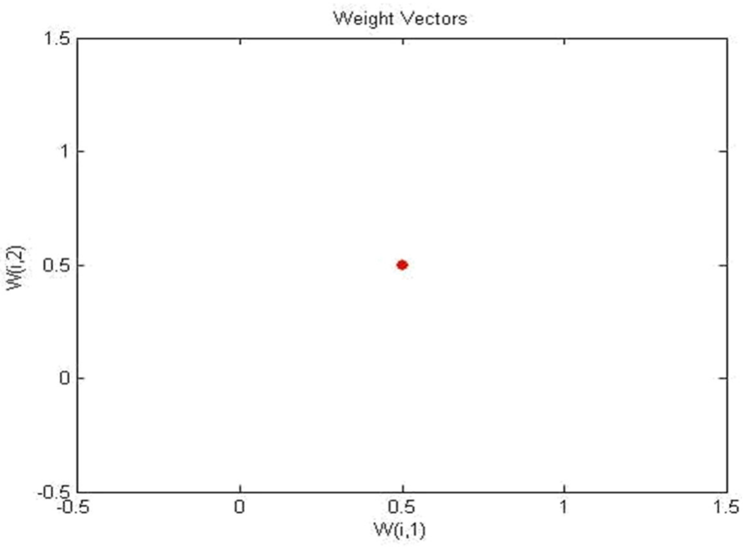


Fig 4-4: plotsom of initially all the neurons have the same weights in the middle of the vectors

Now we train the map on the 1000 vectors for 1 epoch and re-plot the network weights.

After training, note that the layer of neurons has begun to self-organize so that each neuron now classifies a different region of the input space, and adjacent (connected) neurons respond to adjacent regions.

```
net.trainParam.epochs = 1;
net = train(net,P);
plotsom(net.iw{1,1},net.layers{1}.distances)
TRAINR, Epoch 0/1
TRAINR, Epoch 1/1
TRAINR, Maximum epoch reached.
```

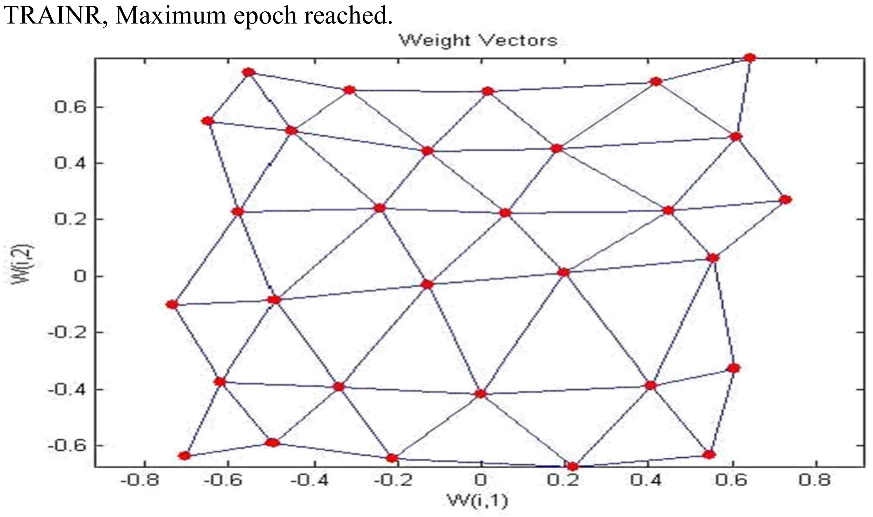


Fig 4-5: plotsom of (30) neurons after (1) epoch

We can now use SIM to classify vectors by giving them to the network and seeing which neuron responds.

The neuron indicated by "a" responded with a "1", so p belongs to that class.

```
p = [0.5; 0.3];
a = sim(net,p)
a =
  (24, 1)        1
```

4.4 A geometric test

The cluster units in a Khonen self-organizing map can viewed as having a position (given by their weight vector) for input patterns with two components, this position is easy to represent graphically. The topological relationships between cluster units in Kohonen Self-organizing maps are often indicated by drawing lines connecting the units.

In this test we assume a linear structure. The initial weights are chosen randomly. With each component having a value between -1 and 1. There are 50 cluster units. The 100 input vectors are chosen randomly from within a circle of radius 0.5 (centered at the origin). The initial learning rate is 0.5 it is reduced linearly to 0.01 over 100 epochs. Throughout training the winning unit and its nearest neighbor unit on either side (units J, J+1, and J-1) are allowed to learn.

Fig.4-6 shows the training pattern, Figs 4-7, 4-8, 4-9, 4-10, 4-11 show the cluster units initially and after 10, 20, 30 and 100 epochs respectively. Not only have the cluster units moved to represent the training inputs (i.e. all of the weight vectors for the cluster units now fall within the unit circle). But the curve connecting the cluster units, has smoothed out somewhat as training progresses. An even smoother curve can be obtained by starting with a larger radius and gradually reducing it to 0. This would involve using more training epochs.

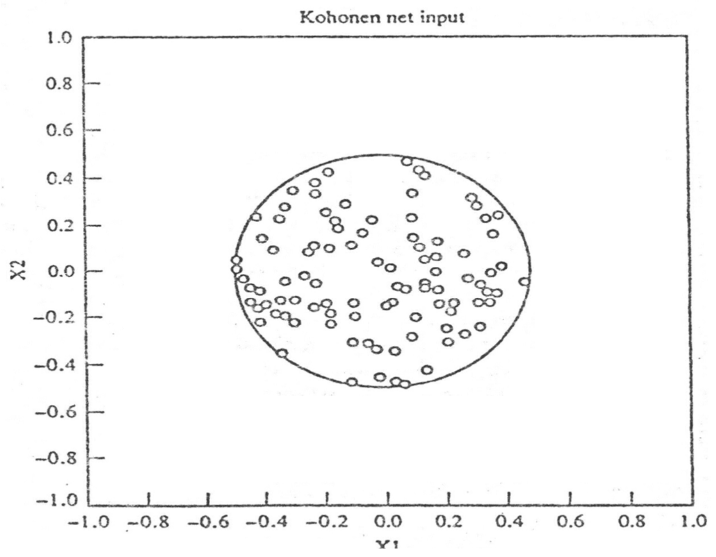


Fig 4-6: Input Patterns

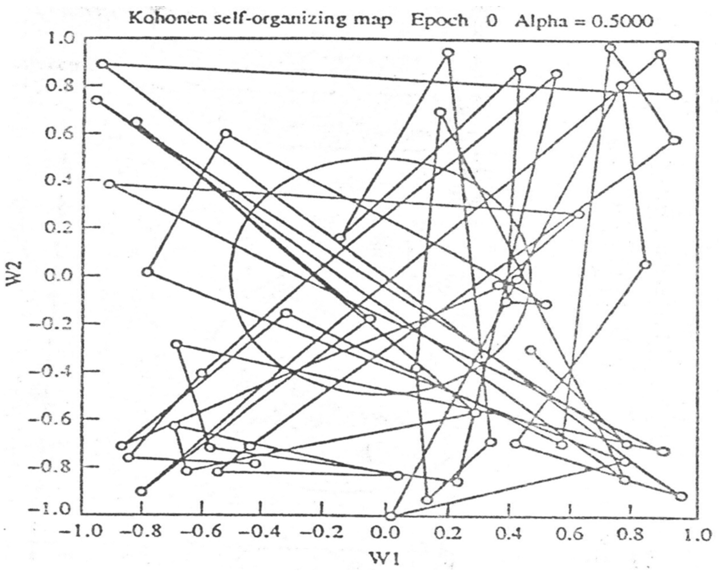


Fig 4-7:initial cluster units

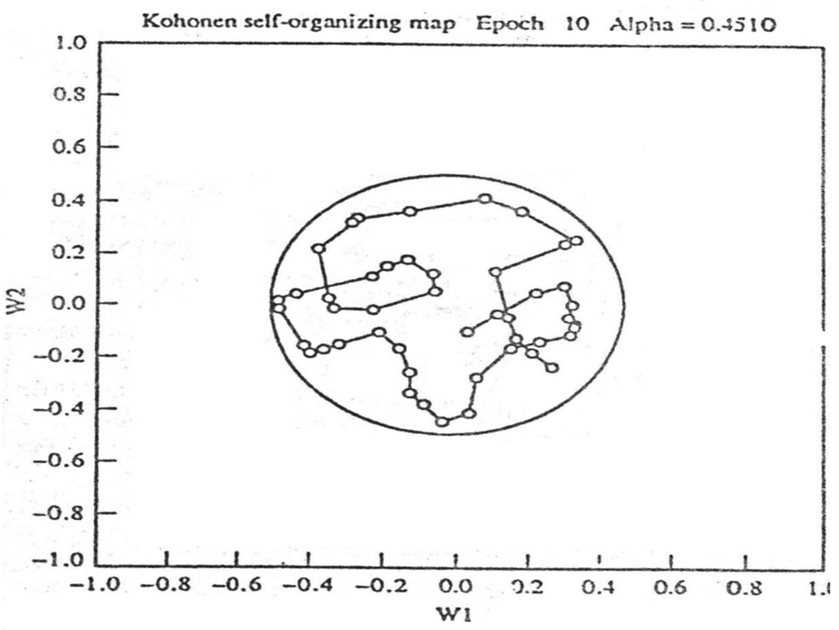


Fig 4-8 Cluster units after 10 epochs

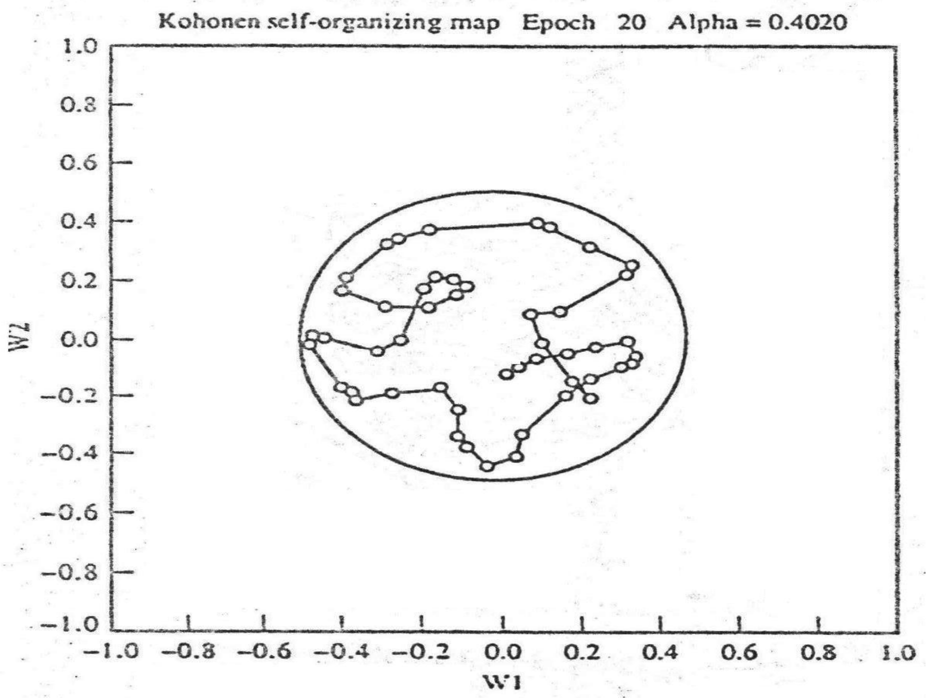


Fig4-9: Cluster units after 20 epochs

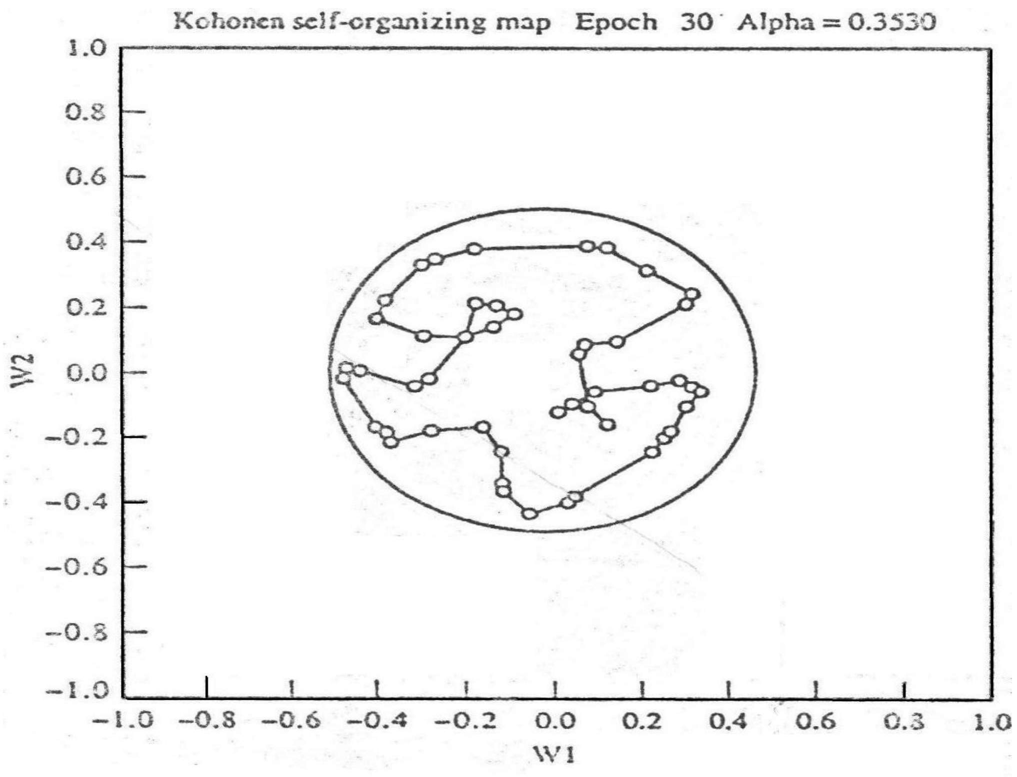


Fig 4-10: Cluster units after 30 epochs

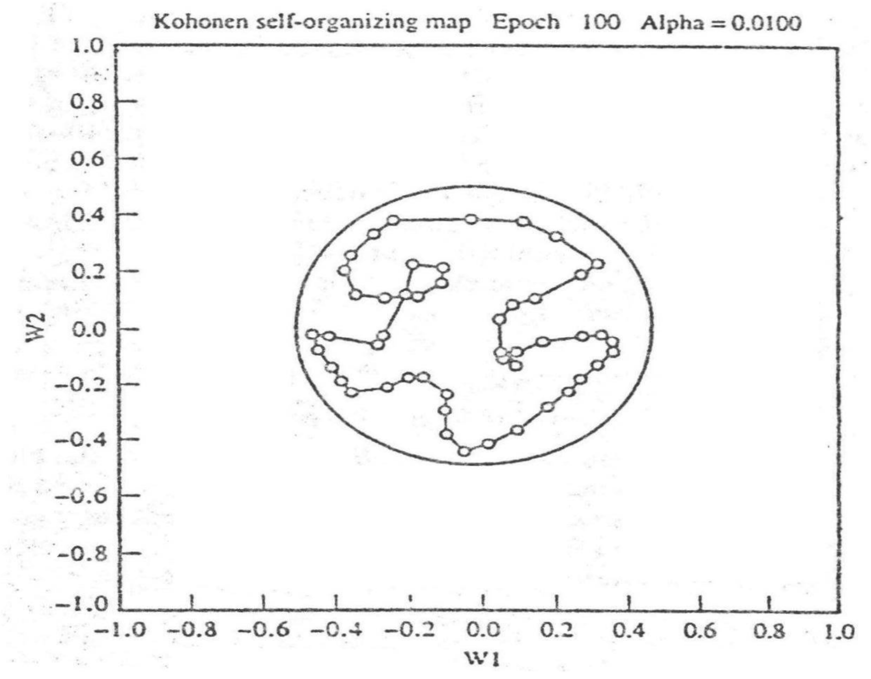


Fig 4-11Cluster units after 100 epochs

5.1 Conclusion

A self-organizing map learns to categorize input vectors. It also learns the distribution of input vectors

Feature maps allocate more neurons to recognize parts of the input space where many input vectors occur and allocate fewer neurons to parts of the input space where few input vectors occur.

Self-organizing maps also learn the topology of their input vectors. Neurons next to each other in the network learn to respond to similar vectors.

The layer of neurons can be imagined to be a rubber net that is stretched over the regions in the input space where input vectors occur.
Self-organizing maps allow neurons that are neighbors to the winning neuron to output values.
Thus the transition of output vectors is much smoother than that obtained with competitive layers, where only one neuron has an output at a time.

5.2 Suggestions for future works
1 .The segmentation of an image can be carried out by different techniques that are based mostly on the discontinuity and similarity of the grey levels of an image.
Neural networks try to simulate a structure similar to the one that is believed the human brain has Two-dimensional layers of cellular modules that are densely interconnected between them. Modeling most neural networks in the brain, especially in the cortex.
.The image has been transformed into a matrix, and segmented if needed by grey level selection; the algorithm for SOM will have as input region.
2. The WEBSOM method organizes a document collection on a map display that provides an overview facilitates interactive browsing. By virtue of the Self-Organizing Map algorithm, documents can be mapped onto a two-dimensional grid so that related documents appear close to each other. If the word forms are first organized into categories on a *word category map*, an encoding of the documents can be achieved that explicitly expresses the similarity of the word meanings. The encoded documents may then be organized by the SOM to produce a *document map*. The visualized document map provides a general view of the document collection.
In the WEBSOM project insightful methods have been developed for information retrieval based on the self-organizing map in the WEBSOM similar documents become mapped close to each other on the map, like the books on the shelves of a well-organized library the self-organized document map offers a general idea of the underlying document a space. The user may view any area of the map in detail by simple clicking the map image with the mouse
The WEBSOM browsing interface is implemented as a set of HTML documents that can be viewed using a graphical WWW browser.

References

1. : Experiments with Kohonen's Self-Organizing Maps and associative memory. Timo Honkela, and Ari M. Vepsäläinen :Proceedings of ICANN-91, International Conference on Artificial Neural Networks, T. Kohonen, K. Mäkisara, O. Simula and J. Kangas (eds.), North-Holland, vol. I, pp. 897-902.(1991)
2. Neural nets that discuss: a general model of communication based on self-organizing maps Timo Honkela (1993): Proceedings of ICANN'93, International Conference on Artificial Neural Networks, Amsterdam, Stan Gielen and Bert Kappen (eds.), Springer-Verlag, London, pp. 408-411.(1993)
3. Contextual relations of words in Grimm tales analyzed by self-organizing map. Timo Honkela, Ville Pulkki, and Teuvo Kohonen (1995): Proceedings of ICANN-95, International Conference on Artificial Neural Networks, F. Fogelman-Soulié and P. Gallinari (eds), vol. 2, EC2 et Cie, Paris, pp. 3-7.(1995)
4. Self-Organizing Maps of words for natural language processing applications. Timo Honkela (1997): Proceedings of International ICSC Symposium on Soft Computing, Nîmes, France, September 17-19, 1997, ICSC Academic Press, Millet, Alberta, Canada, pp. 401-407.
5. (Learning to understand - general aspects of using Self-Organizing Maps in natural language processing. Timo Honkela (1997) Proceedings of the CASYS'97, Computing Anticipatory Systems, Liège, Belgium, August, 1997, in press:
6. (Creating an order in digital libraries with self-organizing maps. Samuel Kaski, Timo Honkela, Krista Lagus, and Teuvo Kohonen (1996): Proceedings of WCNN'96, World Congress on Neural Networks, September 15-18, San Diego, California, Lawrence Erlbaum and INNS Press, Mahwah, NJ, pp. 814-817.
7. The language of technique computing Matlab version 6 releasing 12, September 22, 2000.
8. Paplisinki A.P, (April 17, 2002) "Self Organizing feature maps" the department of computer science, university of Calgary,Canada http:// pharos.cpsc.ucalgary.ca/include/footer.htm
9. Paul E.K, (2001), "what is an artificial neural network?" Email: Paul.keller@pnl.gov http://www.emsl.pn.gov:2080/project/neuron/neural/what.html
10 . Turhan T.m (November. 1997) "Kohonen's self organizing networks with 'conscience' " http://pages.cpsc.ucalgary.ca/~jacob/courses/winter2001/cpsc533/sliders/05.3-sofs-6up.pdf .
11. M. Sc Research (a study of self organizing neural network through image recognition application) By Raghad Al-Saigh September 2002.
University of Mosul
12. "Neural networks-algorithms and applications' By Fiona Neilsen 4i 12/12/2001 Supervisor: Greet Rasmussen. Niels Brock Business College.

13. Timo Honkela, Samuel Kaski, Krista lag us and Teuvo Kohonen Helsinki University of Technology Neural Networks, research center P.o.Box 2200, Fin-02015 HUT. Finland http://websom.hut.fi/websom.
14. "Image segmentation with Kohonen Neural Network self organizing maps." Constantino Carlos Reyes-Aldasoro Instituto Tecnologico Autonomo de Mexico creyes@lamport.rhon.itam.mx.
15. Fundamentals of Neural Networks (Architectures, Algorithms and applications)
    LAURENE FAUSETT
    Florida institute of Technology