Session #19

CMSC 409: Artificial Intelligence

http://www.people.vcu.edu/~mmanic/

Virginia Commonwealth University,
Fall 2023,
Dr. Milos Manic
(mmanic@vcu. edu)

1

CMSC 409: Artificial Intelligence

Session # 19

Topics for today

- Announcements
- Previous session review
- Clustering as competitive learning
  - *Kohonen Networks (WTA)*
  - *Derivation*
  - *Steps*
  - *Example*
  - *Problems & remedies*
- Clustering (cont.)
  - *Forming clusters as needed (FCAN)*

2

CMSC 409: Artificial Intelligence

Announcements

Session # 19

- IMPORTANT:
  - *Course materials (slides, assignments) are copyrighted by instructor & VCU. Sharing/posting/chatGPT/similar is copyright infringement and is strictly prohibited. Such must be immediately reported.*
- Canvas
  - *Prev. session slides updated*
- Additional TA
  - *Victor Cobilean <cobileanv@vcu.edu>, Harindra Sandun Mavikumbure mavikumbureh@vcu.edu*
  - *TA office hours: Thursdays, 3:30 - 4:30pm (Zoom)*
- Project #3
  - *Deadline Oct. 26; Review a week from the deadline*
- Project #4
  - *will be posted shortly, deadline Nov. 9*
- Paper (optional)
  - *The 3rd draft due Nov. 3 (noon)*
  - *In addition to previous draft, it should contain a technique (or selection thereof), you plan on using to solve the selected problem (check out the class paper instructions for the 3rd draft)*
- Subject line and signature
  - *Please use [CMSC 409] Last_Name Question*

3

Clustering

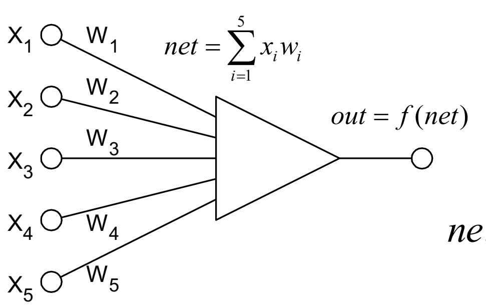
4

Clustering as competitive learning

- *Kohonen Networks (WTA)*
- *Derivation*
- *Steps*
- *Example*
- *Problems & remedies*

Kohonen, T. (1988) Self-Organization and Associative Memory, 2nd Ed. New York, Springer-Verlag.
Kohonen, T. (1982) Self-organized formation of topologically correct feature maps. Biological Cybernetics, 43:59-69.
Kohonen, T. (1990) The Self-Organizing Map. Proceedings of the IEEE, 78:1464-1480.
Kohonen, T. (1995) Self-Organizing Maps. Springer, Berlin.
Kohonen, T., Oja, E., Simula, O., Visa, A., and Kangas, J. (1996). Engineering applications of the self-organizing map. Proceedings of the IEEE, 84:1358-1384.

Kohonen Network



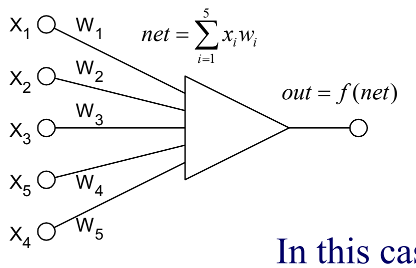
$$net = \sum_{i=1}^{5} x_i w_i = \mathbf{X}\mathbf{W}^T$$

Remember:

*If inputs are binaries, for example **X**=[1, -1, 1, -1, -1] then the maximum net value is when weights are identical to the input pattern:*

__W__=[1, -1, 1, -1, -1]

Kohonen Network



Also….

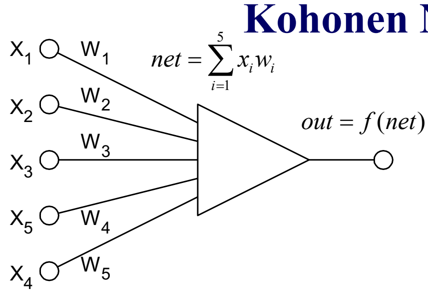
In this case $net = 5$.

For binary weights and patterns *net* value can be found using equation:

$$net = \sum_{i=1}^{n} x_i w_i = \mathbf{X}\mathbf{W}^T = n - 2HD$$

where n is the number of inputs and *HD* is the Hamming distance between input vector **X** and weight vector **W**.

Kohonen Network



This concept can be extended to weights and patterns with analog values as long as both lengths of the weight vector and input pattern vectors are the same.

The Euclidean distance between weight vector **W** and input vector **X** is:

$$\|\mathbf{W} - \mathbf{X}\| = \sqrt{(w_1 - x_1)^2 + (w_2 - x_2)^2 + \cdots + (w_n - x_n)^2}$$

Also can be written as:

$$\|\mathbf{W} - \mathbf{X}\| = \sqrt{\sum_{i=1}^{n} (w_i - x_i)^2} = \sqrt{\sum_{i=1}^{n} (w_i w_i - 2 w_i x_i + x_i x_i)}$$

$$\|\mathbf{W} - \mathbf{X}\| = \sqrt{\mathbf{W}\mathbf{W}^T - 2\mathbf{W}\mathbf{X}^T + \mathbf{X}\mathbf{X}^T} \quad \textit{(matrix form)}$$

Kohonen Network

$$\|\mathbf{W}-\mathbf{X}\| = \sqrt{\mathbf{W}\mathbf{W}^T - 2\mathbf{W}\mathbf{X}^T + \mathbf{X}\mathbf{X}^T}$$

Now, if the lengths of both the weight and input vectors are normalized to value of one:

$\|\mathbf{X}\| = 1$ and $\|\mathbf{W}\| = 1$

then the equation simplifies to:

$$\boxed{\|\mathbf{W}-\mathbf{X}\| = \sqrt{2 - 2\mathbf{W}\mathbf{X}^T}}$$

NOTE: the maximum value of net value (net=1), is when **W** and **X** are identical…

Kohonen Networks (WTA)

- *Derivation*
- *Steps*
- *Example*
- *Problems & remedies*

Kohonen Network

(unsupervised training process)

1. All patterns are normalized (the lengths of the pattern vectors are normalized to unity).

2. Weights are chosen randomly for all neurons

$$\begin{cases} z_1 = \dfrac{x_1}{\sqrt{\sum_{i=1}^{n} x_i^2}} \\ \quad \cdots\cdot \\ z_n = \dfrac{x_n}{\sqrt{\sum_{i=1}^{n} x_i^2}} \end{cases}$$

Kohonen Network

(unsupervised training process)

3. Lengths of the weight vectors are normalized to unity.

4. A pattern is applied to an input and *net* values are calculated for all neurons

$$net = \sum_{i=1}^{5} z_i v_i = \mathbf{ZV}^T$$

$$\begin{cases} v_1 = \dfrac{w_1}{\sqrt{\sum_{i=1}^{n} w_i^2}} \\ \quad \cdots\cdot \\ v_n = \dfrac{w_n}{\sqrt{\sum_{i=1}^{n} w_i^2}} \end{cases}$$

Kohonen Network
(unsupervised training process)

5. A winning neuron is chosen (neuron with largest *net* value).

6. Weights for the winner k are modified using a weighted average:

$$\mathbf{W}_k = \mathbf{V}_k + \alpha \mathbf{Z}$$

where:
α - is the learning constant,
k – index of winning neuron

weights of other neurons are not modified.

Kohonen Network
(unsupervised training process)

7. Weights for the winning neuron are normalized.

8. Another pattern is applied (go to step 4.).

$$\begin{cases} v_1 = \dfrac{w_1}{\sqrt{\sum_{i=1}^{n} w_i^2}} \\ \quad\dots\dots \\ v_n = \dfrac{w_n}{\sqrt{\sum_{i=1}^{n} w_i^2}} \end{cases}$$

Kohonen Network

(unsupervised training process)

During pattern applications **some neurons** are frequent winners and other never take part in the process. The latter ones are eliminated and the **number of recognized clusters** is equal to the **number of surviving neurons**.

NOTE: *number of clusters might not be known upfront. You can start with larger network and eliminate neurons as you go. However, there is a danger of misclassification in this case.*

Things to keep in mind:

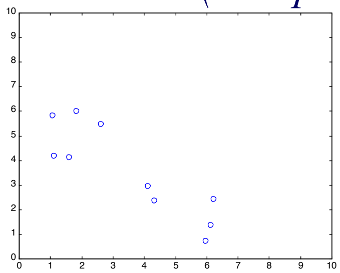
• Clusteing is strongly dependent on the initial set of randomly chosen weights, and order of updating.

• During the normalization process important information about the length of input patterns is lost.

Kohonen Networks (WTA)

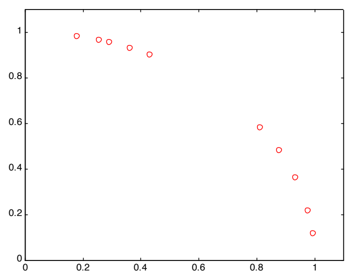
- *Derivation*
- *Steps*
- *Example*
- *Problems & remedies*

Kohonen Network
(unsupervised training process)



Original patterns

5.9630	0.7258
4.1168	2.9694
1.8184	6.0148
6.2139	2.4288
6.1290	1.3876
1.0562	5.8288
4.3185	2.3792
2.6108	5.4870
1.5999	4.1317
1.1046	4.1969

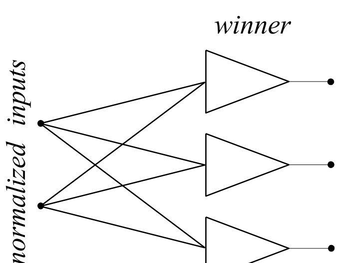


Normalized patterns

0.9927	0.1208
0.8110	0.5850
0.2894	0.9572
0.9314	0.3640
0.9753	0.2208
0.1783	0.9840
0.8759	0.4825
0.4296	0.9030
0.3611	0.9325
0.2545	0.9671

17

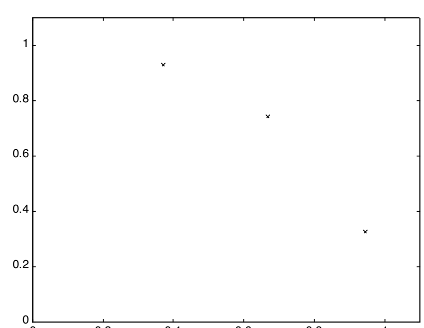
Kohonen Network
(unsupervised training process)



$$\begin{bmatrix} .9459 & .3243 \\ .669 & .7433 \\ .3714 & .9285 \end{bmatrix}$$

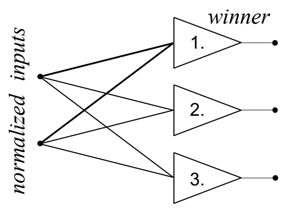
normalized initial weights

randomly chosen number of neurons…



18

Kohonen Network
(unsupervised training process)



applying the first pattern $Z_1 = [0.9927 \quad 0.1208]$

initial weights net

$$\begin{bmatrix} .9459 & .3243 \\ .6690 & .7433 \\ .3714 & .9285 \end{bmatrix} \Rightarrow \begin{bmatrix} 0.9782 \\ 0.7539 \\ 0.4809 \end{bmatrix}$$

From here, neuron #1 is the winner. Therefore, weights for neuron #1 are updated and normalized:

$\left(\frac{1.2437}{\sqrt{1.2437^2+0.3606^2}}\right)=0.96044$

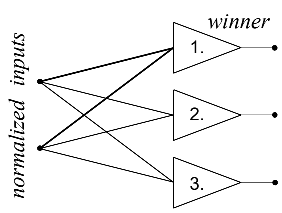
$\mathbf{W}_k = \mathbf{V}_k + \alpha\mathbf{Z} =$ (0.9459 0.3243) + alpha (0.9927 0.1208)=

= (0.9459 0.3243) + 0.3 (0.9927 0.1208) = (1.2437 0.3606) $\xRightarrow{normalization}$ (0.9605 0.2784)

$$\begin{bmatrix} .9459 & .3243 \\ .6690 & .7433 \\ .3714 & .9285 \end{bmatrix} \xRightarrow{\text{weight update}} \begin{bmatrix} .9605 & .2784 \\ .6690 & .7433 \\ .3714 & .9285 \end{bmatrix}$$

1st neuron updated

Kohonen Network
(unsupervised training process)



applying the second pattern $[0.8110 \quad 0.5850]$

initial weights net

$$\begin{bmatrix} .9605 & .2784 \\ .6690 & .7433 \\ .3714 & .9285 \end{bmatrix} \Rightarrow \begin{bmatrix} 0.9418 \\ 0.9774 \\ 0.8444 \end{bmatrix}$$

From here, neuron #2 is the winner. Therefore weights for neuron #2 are updated and normalized:

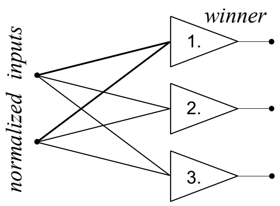
$\mathbf{W}_k = \mathbf{V}_k + \alpha\mathbf{Z} =$ (0.6690 0.7433) + alpha (0.8110 0.5850)

= (0.6690 0.7433) + 0.3 (0.8110 0.5850) = (0.9123 0.9188) $\xRightarrow{normalization}$ (0.7046 0.7096)

$$\begin{bmatrix} .9605 & .2784 \\ .6690 & .7433 \\ .3714 & .9285 \end{bmatrix} \xRightarrow{\text{weight update}} \begin{bmatrix} .9605 & .2784 \\ .7046 & .7096 \\ .3714 & .9285 \end{bmatrix}$$

2nd neuron updated

Kohonen Network
(unsupervised training process)



applying the third pattern $Z_3 = [0.2894 \quad 0.9572]$

initial weights → net

$$\begin{bmatrix} .9605 & .2784 \\ .7046 & .7096 \\ .3714 & .9285 \end{bmatrix} \Rightarrow \begin{bmatrix} 0.5445 \\ 0.8832 \\ 0.9962 \end{bmatrix}$$

From here, neuron #3 is the winner. Therefore weights for neuron #3 are updated and normalized:

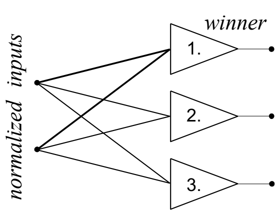
$$\mathbf{W}_k = \mathbf{V}_k + \alpha \mathbf{Z} = (0.3714 \quad 0.9285) + \text{alpha}\ (0.2894 \quad 0.9572)$$

$$= (0.3714 \quad 0.9285) + 0.3\ (0.2894 \quad 0.9572) = (0.4582 \quad 1.2156) \xRightarrow{\text{normalization}} (0.3527 \quad 0.9357)$$

$$\begin{bmatrix} .9605 & .2784 \\ .7046 & .7096 \\ .3714 & .9285 \end{bmatrix} \underset{\text{weight update}}{\Longrightarrow} \begin{bmatrix} .9605 & .2784 \\ .7046 & .7096 \\ .3527 & .9357 \end{bmatrix}$$

3rd neuron updated

Kohonen Network
(unsupervised training process)



applying the fourth pattern $Z_4 = [0.9314 \quad 0.3640]$

initial weights → net

$$\begin{bmatrix} .9605 & .2784 \\ .7046 & .7096 \\ .3527 & .9357 \end{bmatrix} \Rightarrow \begin{bmatrix} 0.996 \\ 0.915 \\ 0.669 \end{bmatrix}$$

From here, neuron #1 is the winner. Therefore weights for neuron #3 are updated and normalized:

$$\mathbf{W}_k = \mathbf{V}_k + \alpha \mathbf{Z} = (0.9605 \quad 0.2784) + \text{alpha}\ (0.9314 \quad 0.3640)$$

$$= (0.9605 \quad 0.2784) + 0.3\ (0.9314 \quad 0.3640) = (1.2399 \quad 0.3876) \xRightarrow{\text{normalization}} (0.9544 \quad 0.2983)$$

$$\begin{bmatrix} .9605 & .2784 \\ .7046 & .7096 \\ .3527 & .9357 \end{bmatrix} \underset{\text{weight update}}{\Longrightarrow} \begin{bmatrix} .9544 & .2983 \\ .7046 & .7096 \\ .3527 & .9357 \end{bmatrix}$$

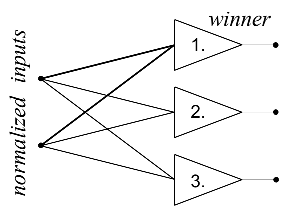
1st neuron updated

Kohonen Network
(unsupervised training process)

Process continues for all patterns

weights after the first iteration

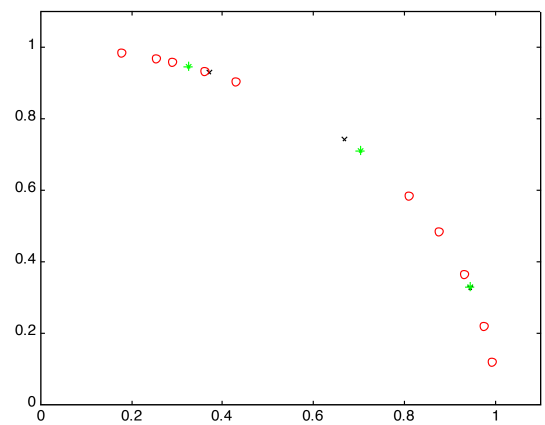
$$\begin{bmatrix} .9605 & .2784 \\ .6690 & .7433 \\ .3714 & .9285 \end{bmatrix}$$



o - normalized patterns

x - initial weights

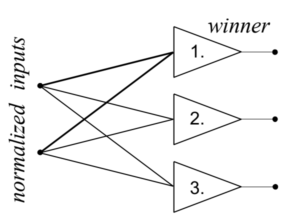
* - current weights



Kohonen Network
(unsupervised training process)

weights after the second iteration

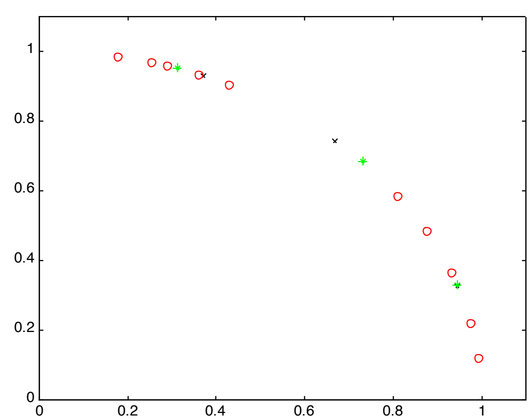
0.9439	0.3302
0.7309	0.6825
0.3119	0.9501



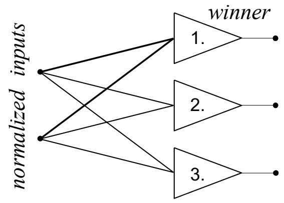
o - normalized patterns

x - initial weights

* - current weights



Kohonen Network
(unsupervised training process)



weights after the 30 iterations

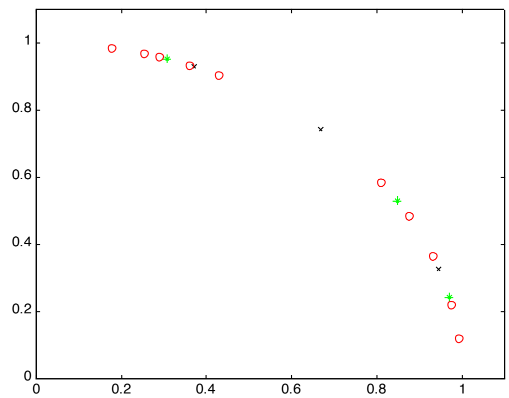
$$\begin{matrix} 0.9699 & 0.2435 \\ 0.8492 & 0.5281 \\ 0.3072 & 0.9516 \end{matrix}$$

o - normalized patterns

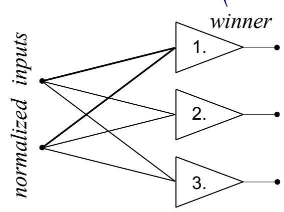
x - initial weights

* - current weights

weights are representing center of clusters

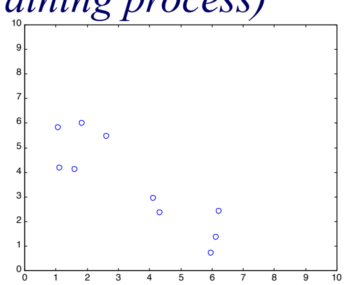


Kohonen Network
(unsupervised training process)



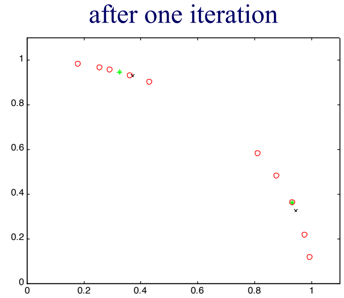
Initial weights

$$\begin{bmatrix} .9605 & .2784 \\ .7046 & .7096 \\ .3527 & .9357 \end{bmatrix}$$



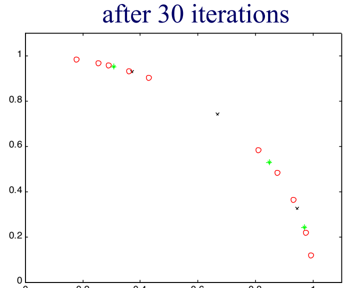
Patterns (initially)

after one iteration



Patterns (after one iterat.)

after 30 iterations



Patterns (after 30 iterat.)

Kohonen Networks (WTA)

- *Derivation*
- *Steps*
- *Example*
- *Problems & remedies*

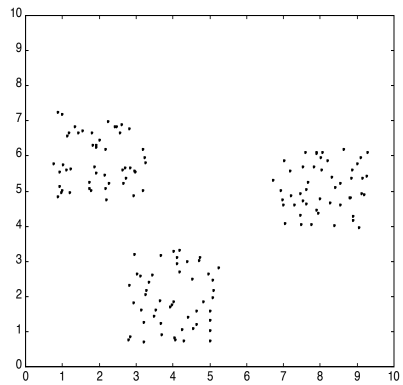
Kohonen Networks
Problems & Remedies

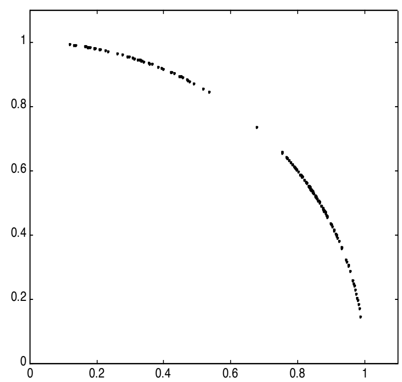
1. Important information about the length of the vector is lost during the normalization process

2. Clustering may depend on:
   a) Order patterns are applied
   b) Number of initial neurons
   c) Initial weights

Kohonen Networks
Problems & Remedies

Important information about length of the vector is lost during the normalization process





29

Kohonen Networks
Problems & Remedies

Problem:
Important information about length of the vector is lost during the normalization process.

Possible remedies:
• The problem can be solved by increasing a dimension by one and usage of vector angles as variables. Lengths are the same.
This approach (used by Kohonen) leads to complex trigonometric computations

• Other way to approach the problem is to project patterns into hypersphere of higher dimensionality. This way all patterns have the same length but important information is not lost.

30

Forming Clusters As Needed

Forming clusters as needed

using minimum distance concept

(much simpler and more efficient then ART)

1. First pattern is applied and the cluster is formed
2. Next pattern is applied and then:
   a) *If distance from all existing clusters is larger then threshold then a new cluster is formed,*
   b) *Else weights of the closest cluster are updated.*

$$\boxed{\mathbf{W}_k = \frac{m\mathbf{W}_k + \alpha\mathbf{X}}{m+1}}$$

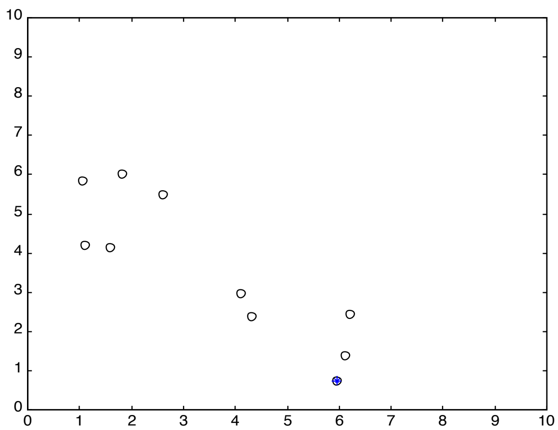
(averaged)

where:
- m - is the number of previous patterns of a given set which were used to update this particular neuron and,
- α - is the learning constant.

- *Learning is cautious!*
- *If many patterns form a cluster, then "cluster is heavy"…*

Forming clusters as needed using minimum distance concept

5.9630	0.7258
4.1168	2.9694
1.8184	6.0148
6.2139	2.4288
6.1290	1.3876
1.0562	5.8288
4.3185	2.3792
2.6108	5.4870
1.5999	4.1317
1.1046	4.1969



$$\mathbf{W}_k = \frac{m\mathbf{W}_k + \alpha\mathbf{X}}{m+1}$$

$\alpha = 1$

$RAD_{max} = 3$

members
- 1st cluster = 1

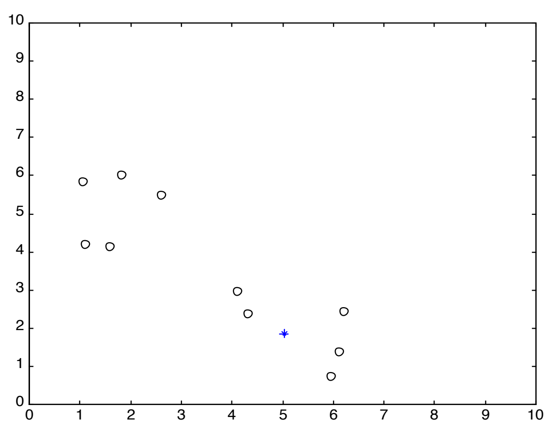
First pattern applied and the first cluster is formed:

5.9630 0.7258

33

Forming clusters as needed using minimum distance concept

5.9630	0.7258
4.1168	2.9694
1.8184	6.0148
6.2139	2.4288
6.1290	1.3876
1.0562	5.8288
4.3185	2.3792
2.6108	5.4870
1.5999	4.1317
1.1046	4.1969



$$\mathbf{W}_k = \frac{m\mathbf{W}_k + \alpha\mathbf{X}}{m+1}$$

$\alpha = 1$

$RAD_{max} = 3$

ED=2.9056

members
- 1st cluster = 2

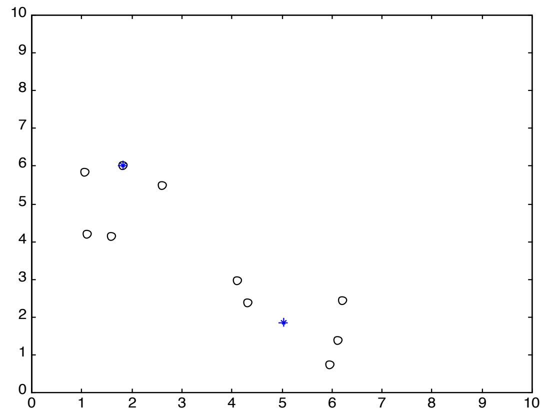
Second pattern applied and the 1st cluster COG is moved

5.9630 0.7258 $\xrightarrow{4.1168 \quad 2.9694}$ 5.0399 1.8476

34

Forming clusters as needed using minimum distance concept

5.9630	0.7258
4.1168	2.9694
1.8184	6.0148
6.2139	2.4288
6.1290	1.3876
1.0562	5.8288
4.3185	2.3792
2.6108	5.4870
1.5999	4.1317
1.1046	4.1969



$$\mathbf{W}_k = \frac{m\mathbf{W}_k + \alpha\mathbf{X}}{m+1}$$

$\alpha = 1$

$RAD_{max} = 3$

ED=5.2672

members
- 1st cluster = 2
- 2nd cluster = 1

Third pattern applied and a new cluster is formed

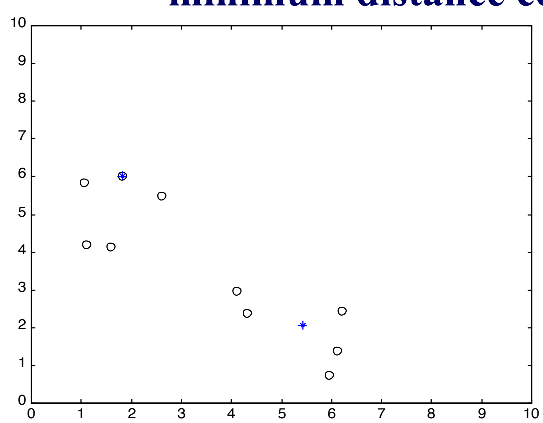
5.0399 1.8476

1.8184 6.0148 ➔

1. 5.0399 1.8476
2. 1.8184 6.0148

Forming clusters as needed using minimum distance concept

5.9630	0.7258
4.1168	2.9694
1.8184	6.0148
6.2139	2.4288
6.1290	1.3876
1.0562	5.8288
4.3185	2.3792
2.6108	5.4870
1.5999	4.1317
1.1046	4.1969



$$\mathbf{W}_k = \frac{m\mathbf{W}_k + \alpha\mathbf{X}}{m+1}$$

$\alpha = 1$

$RAD_{max} = 3$

ED=1.3100 5.6727

members
- 1st cluster = 3
- 2nd cluster = 1

Fourth pattern applied and the first cluster is moved

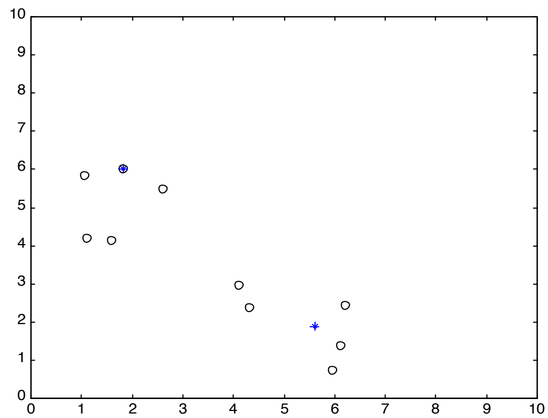
1. 5.0399 1.8476
2. 1.8184 6.0148

6.2139 2.4288 ➔

1. 5.4312 2.0414
2. 1.8184 6.0148

Forming clusters as needed using minimum distance concept

5.9630	0.7258
4.1168	2.9694
1.8184	6.0148
6.2139	2.4288
6.1290	1.3876
1.0562	5.8288
4.3185	2.3792
2.6108	5.4870
1.5999	4.1317
1.1046	4.1969



$$\mathbf{W}_k = \frac{m\mathbf{W}_k + \alpha\mathbf{X}}{m+1}$$

$\alpha = 1$

$RAD_{max} = 3$

ED=0.9562 6.3240

members
- 1st cluster = 4
- 2nd cluster = 1

Fifth pattern applied and the first cluster is moved again

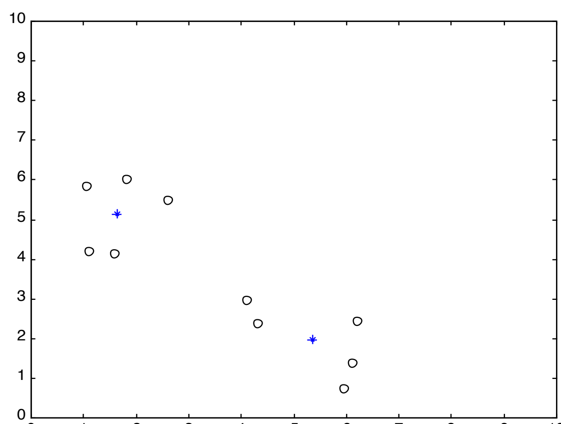
1. 5.4312 2.0414
2. 1.8184 6.0148

6.1290 1.3876 ➔

1. 5.6057 1.8779
2. 1.8184 6.0148

Forming clusters as needed using minimum distance concept

5.9630	0.7258
4.1168	2.9694
1.8184	6.0148
6.2139	2.4288
6.1290	1.3876
1.0562	5.8288
4.3185	2.3792
2.6108	5.4870
1.5999	4.1317
1.1046	4.1969



$$\mathbf{W}_k = \frac{m\mathbf{W}_k + \alpha\mathbf{X}}{m+1}$$

$\alpha = 1$

$RAD_{max} = 3$

ED=4.7887 1.3455

members
- 1st cluster = 5
- 2nd cluster = 5

After all 10 patterns

if α is =1, is exactly the COG!* ➔

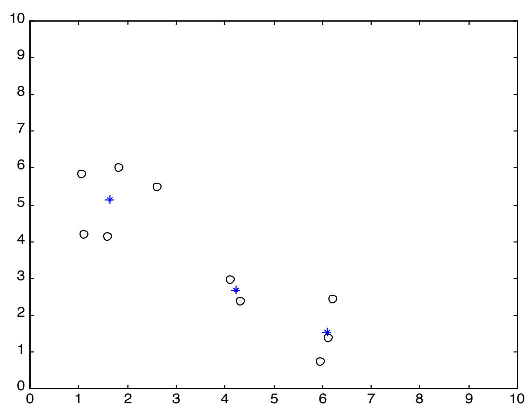
1. 5.3482 1.9782
2. 1.6380 5.1318

Now…

for the smaller radius (Rad_{max} =2)

Forming clusters as needed using minimum distance concept

5.9630	0.7258
4.1168	2.9694
1.8184	6.0148
6.2139	2.4288
6.1290	1.3876
1.0562	5.8288
4.3185	2.3792
2.6108	5.4870
1.5999	4.1317
1.1046	4.1969



$$\mathbf{W}_k = \frac{m\mathbf{W}_k + \alpha\mathbf{X}}{m+1}$$

$\alpha = 1$

$RAD_{max} = 2$

ED=5.6720 3.4655 1.3455

members
- •1st cluster = 3
- •2nd cluster = 2
- •3rd cluster = 5

After all 10 patterns

if α is =1, is exactly the COG!* ➔

1. 6.1020 1.5141
2. 4.2176 2.6743
3. 1.6380 5.1318

Forming clusters as needed using minimum distance concept

Kohonen method:

- Certain number of iterations is needed in order to achieve some results,
- Tends to produce number of clusters equal to number of neurons!

Forming clusters as needed using min. distance:

- Winning strategy used again,
- Each pattern applied only once,
- No normalization,
- Radius of attraction (Hamming Distance):
    - larger => may encompass fewer clusters
    - smaller => may produce larger number of clusters
- Simpler and faster method!

Things to remember…

- **Unsupervised training…**
    - *There is no label data, no desired output…*
    - *So, TE, testing error, accuracy…do not apply*

- **So, the stopping criterion?**
    - *No labels => no TE, so stop when CoGs "stabilize"*
    - *(keep in mind possibility of local minima)*

- **Unsupervised training "truly" unsupervised?**
    - *Not entirely…because we introduce assumptions*
        - *WTO (Kohonen) – algorithm designer makes assumption on initial network (number of neurons) and corresponding weights*
        - *FCAN - designer makes assumptions: 1. "threshold"; 2. order of patterns fed*
    - *Would be fully supervised if these assumptions were "automated" (no need from human "input")*

- **When put in production…**
    - *(when trained ANN used to cluster incoming data), soft activation function would enable "soft thresholds"*