Computational Intelligence
Lecture 17: Unsupervised Training

Farzaneh Abdollahi

Department of Electrical Engineering

Amirkabir University of Technology

Fall 2010

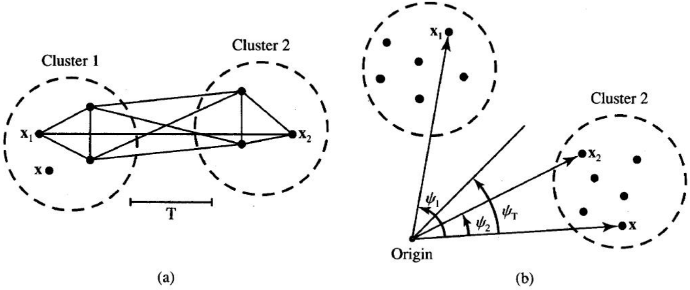
Unsupervised Learning of Clusters

Kohonen network (Kohonen 1988)

Winner Take-All Learning

Kohonen network (Kohonen 1988)

- Unsupervised classification is called clustering
- Clustering is considered as the grouping of similar objects and separating of dissimilar ones.
- Sometimes even number of clusters are not known a priori.
  - The clustering technique should
    1. identify # of classes according to a certain criterion
    2. assign the membership of the patterns in these classes.
- The clustering technique presented here, knows # of clusters in a priori
- The input is required to be identifies as member of one of the possible clusters.

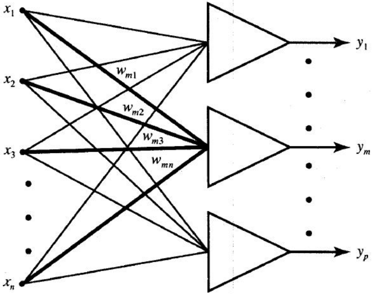


Measures of similarity for clustering data: (a) distance and (b) a normalized scalar product.

- Since it is unsupervised, criterion for similarity should be defined:
  1. Euclidean distance between two patterns x, for x_i: $\|x - x_i\| = \sqrt{(x - x_i)^T (x - x_i)}$
  2. scalar product (cos of the angle between x and x_i): $\cos\psi = \frac{x^T x_i}{\|x\|\|x_i\|}$. For $cos\psi_2 < \cos\psi_1$: x is more similar to x_2 than x_1.
- scalar product is simpler for implementation
- For normalaized vectors both method yield similar results

Winner Take-All Learning

- Objective is learn the weights s.t. classifies input vectors into one of the specified number of p categories
- Training set $\{x_l, x_z, ..., x_N\}$.
- $y = \Gamma[Wx]$
- Γ is diagonal continuous activation function
- $W = [w_1^T \;\; w_2^T \;\; ... \;\; w_p^T]^T$, $w_i = [w_{i1} \; ... \; w_{in}] \;\; i = 1, ...p$
- Each neuron represent a cluster
- Winner take-all learning: Only the weight corresponding to the winner neuron is updated



(Adapted weights highlighted)

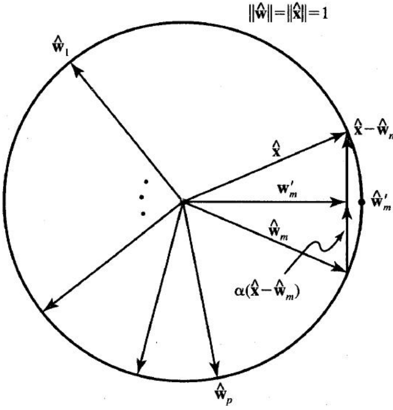
- Before start learning all weights should be normalized $\hat{w}_i = \frac{w_i}{\|w_i\|} \; i = 1, ..., p$
- For $x = s^m$, by selecting $w = s^m$ mth neuron will be max.
- But s^m. center of cluster is not known a priori
- Proper learning rule is required to find the centers
- The weight vectors should be modified accordingly so that they become more similar to the current input vector
- The winning neuron (the closest approximation of the current input x), to be updated is selected s.t. $\|x - \hat{w}_m\| = (x^T x - 2\hat{w}_m^T x + 1)^{1/2} = \min_{i=1,...p}\{\|x - \hat{w}_i\|\}$
- Searching for the min of p distances corresponds to finding the max among the p scalar products $\hat{w}_m^T x = \max_{i=1,..,p}(\hat{w}_i^T x)$
- If the weights are not normalized the above statement is not always true

- After identifying the winning neuron, its weights must be adjusted s. t. the distance $\|x - w_m\|$ is reduced.
- The wining weight is updated along the gradient direction in the weight space $\nabla_{w_m}\|x - w_m\|^2 = -2(x - w_m)$
- $\therefore$ Learning rule in kth step is

$$
\begin{aligned}
\hat{w}_m^{k+1} &= \hat{w}_m^k + \alpha^k(x - \hat{w}_m^k) \\
\hat{w}_i^{k+1} &= \hat{w}_i^k \quad i \neq m
\end{aligned}
$$

- α^k: a suitable learning constant (between 0.1 to 0.7),
- m is the winning neuron selected based on the scalar product comparison (largest net_i)
- During the learning the clusters are developed,
- the network weights acquire similarity to input data within clusters.
- To avoid unconstrained growth of weights, α is usually reduced monotonically and the learning slows down.

- Geometrical interpretation of the rule is shown in Fig.
- Assume that in this step $\hat{x}$ is the normalized input vector of x and $\hat{w}_m$ yield the maximum scalar product $\hat{w}_m^T$, for $i = 1, 2, ..., p$.
- To implement the rule for $x = \hat{x}$, an increment of the weight vector is computed as a fraction of $\hat{x} - \hat{w}_m^T$
- $\therefore$ weight adjustment is the rotation of the weight vector $\hat{w}_m$ toward the input vector without a significant length change.
- The adjusted weight vector w'_m's length is below unity $\rightsquigarrow$ for next step it should be renormalized



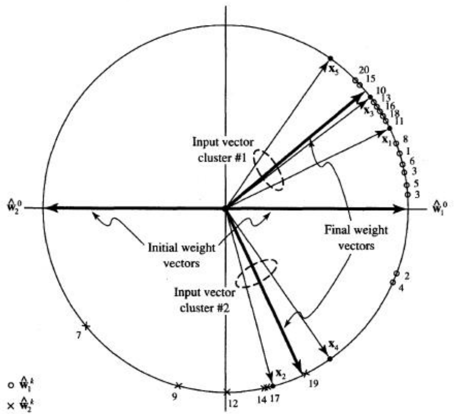
- (Simpson 1990) proposed a supervised Kohonen network:
  - $\alpha > 0$ for proper node responses
  - $\alpha < 0$ otherwise
- Another modification of the winner-take-all learning rule for the cases which clusters may be hard to distinguish named Leaky competitive learning
  - Both the winners' and losers' weights are adjusted in proportion to their level of responses.
- **Recall Mode** The network trained in the winner-take-all mode responds instantaneously during feedforward recall
- The response is $y = \Gamma[Wx]$
- The layer now performs as a filter of the input vectors such that the largest output neuron is found as $y_m = max(y_1, ..., y_p)$

- **Weight Initializing**
- Random initial weight vectors should be selected s.t. uniformly distributed on the unity hypersphere in n-dimensional pattern space.
- Self-organization of the network suffers from some limitations:
  - Because of its single-layer architecture, linearly nonseparable patterns cannot be efficiently handled by this network.
  - The network training may not always be successful even for linearly separable patterns.
- The weights may get stuck in isolated regions without forming adequate clusters.
- In such cases the training must be reinitialized with new initial weights,
- After the weights have been trained to provide coarse clustering, the learning constant α should be reduced

Example

- Consider the normalized patterns for training: $\{x_1, x_2, x_3, x_4, x_5\} = \{\begin{bmatrix} 0.8 \\ 0.6 \end{bmatrix}, \begin{bmatrix} 0.1736 \\ -0.9848 \end{bmatrix}, \begin{bmatrix} 0.707 \\ 0.707 \end{bmatrix}, \begin{bmatrix} 0.342 \\ -0.9397 \end{bmatrix}, \begin{bmatrix} 0.6 \\ 0.8 \end{bmatrix}\}$
- $\#$ of clusters: 2, and $\alpha = 1/2$
- The normalized initial weights: $w_1^0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, w_2^0 = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$
- The inputs are submitted in ascending sequence and recycled
- The first neuron won all of the first six competitions in a row $\rightsquigarrow$ the first weight is only updated.
- In the second training cycle, x_2 and x_4 has been detected by the second neuron.
- After 20 steps the weights are adjusted properly and index the center of clusters.

Example Cont'd



(a)

Step k	$\hat{w}_1^k$ ∡ deg	$\hat{w}_2^k$ ∡ deg
1	18.46	−180.00°
2	−30.77	—
3	7.11	—
4	−31.45	—
5	10.85	—
6	23.86	—
7	—	−130.22
8	34.43	—
9	—	−100.01
10	43.78	—
11	40.33	—
12	—	−90.00
13	42.67	—
14	—	−80.02
15	47.90	—
16	42.39	—
17	—	−80.01
18	43.69	—
19	—	−75.01
20	48.42	—

(weight vectors of unity length)
(— means no change)

(b)

(a) training patterns and weight assignments and (b) weight learning, Steps 1 through 20.