COMPETITIVE LEARNING

DATE ___/___/___

Winner – Takes – All Network

During learning process, there exist a competition b/w the clusters to accommodate an object in its partition (cluster). finally only one cluster (the output neuron) wins the competition & object is placed in the corresponding cluster. there can be only one winning neuron that is active at a time. The ~~network~~ neural network of this type is also known as competive net.

prototypes of cluster : $v = \{v_1, v_2 - v_n\}$ where v set of prototypes for n clusters.

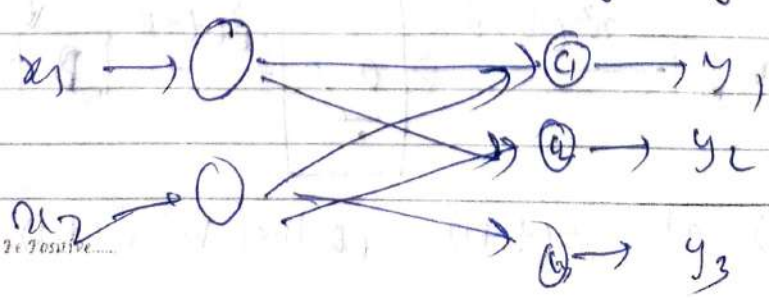
Winner - takes - all - network : Two layer (input & output.

Weights : Initially random weights. There is no bias values

Activation function: Step function

$$\phi_i = \begin{cases} 1 & , \text{ if } i = \arg\min_j a_i \\ 0 & , \text{ otherwise} \end{cases}$$

If the distance b/w the object x_j & the cluster prototype v_i is minimum, then the neuron wins & will be active by seting a_i to 1.



DATE: / /

By $x = \begin{bmatrix} .5 \\ .2 \end{bmatrix}$, $c_1 = \begin{bmatrix} .7 \\ .4 \end{bmatrix}$, $c_2 = \begin{bmatrix} .6 \\ .3 \end{bmatrix}$, $c_3 = \begin{bmatrix} .9 \\ .1 \end{bmatrix}$

$d_1(x, c_1) = \sqrt{(.5-.7)^2 + (.2-.4)^2} = \sqrt{.08} = .2828$

$d_2(x, c_2) = .1414$

$d_3(x, c_3) = .4123$

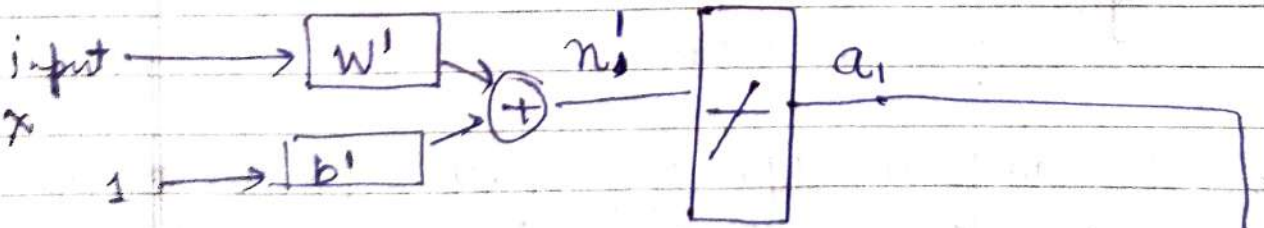
Here d_2 = is minimum distance b/w object x & cluster c_2, therefore neuron corresponding to cluster c_2 in the output layer is winner neuron.

Hamming Network →

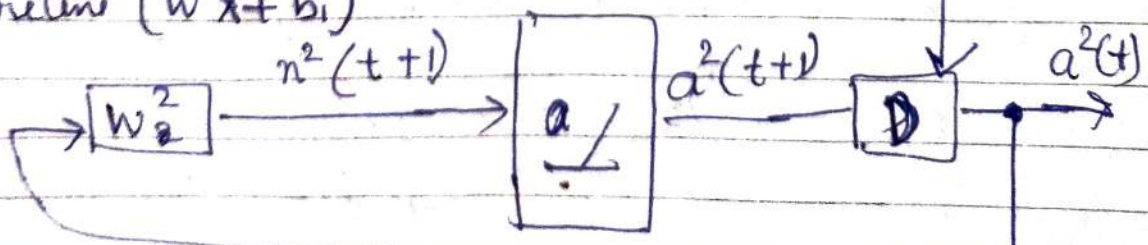
Hamming network consist of two layer.

Layer (1) performs a correlation b/w the input vector & the prototype vectors.

Layer (2) performs a competition to determine which of the prototype vectors is closest to the input vector



$a_1 = \text{purelin}(W^1 x + b_1)$



$a^2(0) = a^1 \quad a^2(t+1) = \text{poslin}(W^2 a^2(t))$

Competitive layer → Second layer is Competitive layer.

Transfer function for recurrent competitive layer

$$a = compet(n)$$

It works by finding the i^{th} index of the neuron with the largest net input, & setting its to 1 & all others output are set to 0.

DATE: __/__/____

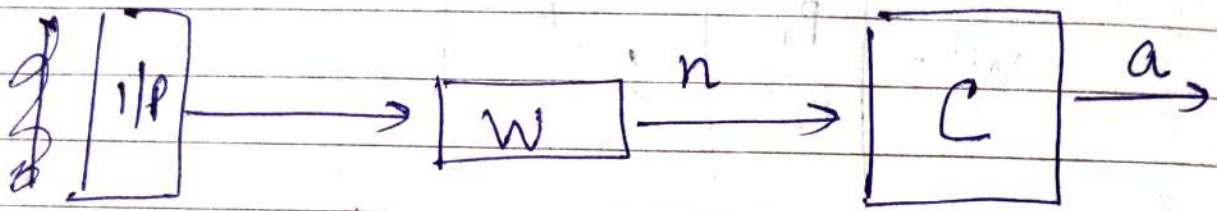
$$a_i = \begin{cases} 1, & \text{if } i = i^* \\ 0, & \text{if } i \neq i^* \end{cases}$$

where

$n_{i^*} \geq n_i \quad \forall i,$

and

$i^* \leq i \; \forall n_i = n_{i^*}$



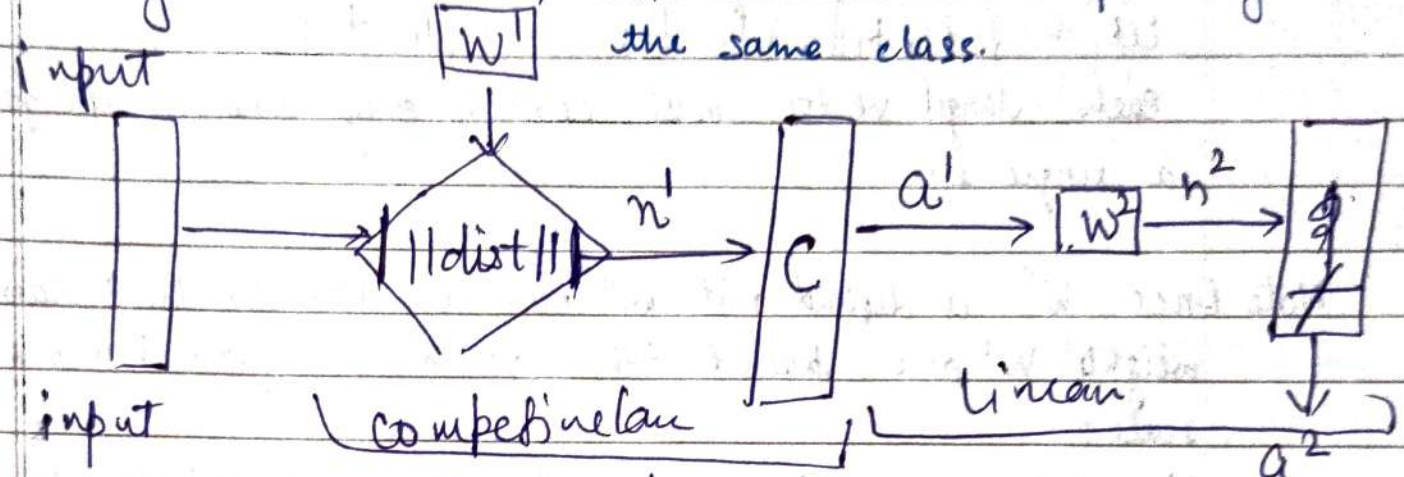
$$a = \text{compet}(W p^T)$$

DATE: __/__/__

Learning Vector Quantization :→

It is both supervised & unsupervised learning to form classifications.

In LVQ network, each neuron in the first layer is assigned to a class, with several neuron often assigned to the same class.



$$n_i^1 = \|_i w^1 - x_j\| \qquad a^2 = W^2 a^1$$

$$a^1 = compet(n^1)$$

The net input of the first layer of the LVQ

$$n_i^1 = -\|_i w^1 - x_j\|$$

the output of the first layer of LVQ.

$$a^1 = compet(n^1)$$

Therefore the neuron whose weight vector is closest to the input vector will output a 1, other 0.

DATE: / /

The net input of the first layer of the LVQ will be

$$n_i^1 = -\| w_{ij} - x_j \|$$

LVQ Learning –

Let $\{p_1, t_1\}, \{p_2, t_2\} \ldots \{p_Q, t_Q\}$

each target vector must contain only zero, except for a single 1.

Note Once W^2 is defined, it will never be altered. The Hidden weights W^1 are trained with a variation of the Kohonen rule.

The Kohonen rule is used to improve the hidden layer of LVQ network in two ways:

(1) p is classified correctly, then we move the weights w_{ij}^1 of the winning hidden neuron toward p.

$$w_{ij}^1(q) = w_{ij}^1(q-1) + \alpha\left(p(q) - w_{ij}^1(q-1)\right)$$
$$\text{if } a_k^2 = t_k = 1$$

(2) p is classified incorrectly, so we move the weights away from p

$$w_{ij}(q) = w_{ij}(q-1) - \alpha\left(p(q) - w_{ij}(q-1)\right)$$
$$\text{if } a_k^1 = 1 \neq t_k = 0$$

DATE: / /

Ex → class1 : $\left\{P_1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, P_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right\}$, class2 : $\left\{P_3 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, P_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}\right\}$

Solution - assignment target vectors to each input

$$\left\{P_1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, t_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}\right\}, \left\{P_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, t_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}\right\},$$

$$\left\{P_3 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, t_3 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}\right\}, \left\{P_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}, t_4 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}\right\}$$

The output layer weight matrix will be

$$W^2 = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

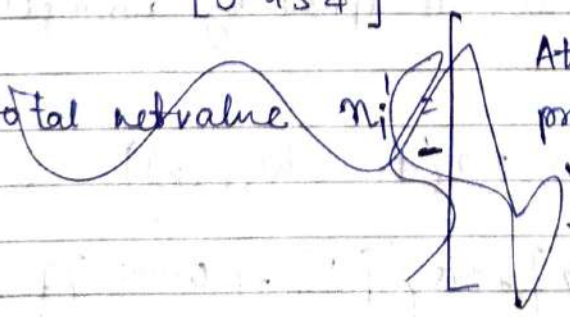
W^2 connects hidden neuron 1 & 2 to output neuron 1.

The column vector W^1 are initially set to random values.

$${}_1W^1_{11} = \begin{bmatrix} -0.543 \\ 0.840 \end{bmatrix}, {}_2W^1_{22} = \begin{bmatrix} -0.969 \\ -0.249 \end{bmatrix}, {}_3W^1_{33} = \begin{bmatrix} 0.997 \\ 0.094 \end{bmatrix}$$

$${}_4W^1_{44} = \begin{bmatrix} 0.456 \\ 0.954 \end{bmatrix}$$

total netvalue n_i^1



At each iteration of the training process, we present an input vector, find its response & then adjust the weight.

DATE:

In this case we will be given by presenting p_3

$$a^1 = \text{compet}(n^1) = \text{compet}\left(\begin{bmatrix} -\|w_{11} - p_3\| \\ -\|w_{22} - p_3\| \\ -\|w_{33} - p_3\| \\ -\|w_{44} - p_3\| \end{bmatrix}\right)$$

$$= \text{compet}\left(\begin{bmatrix} -2.40 \\ -2.11 \\ -1.09 \\ -2.03 \end{bmatrix}\right) = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Third hidden neuron has the closest weight vector to p_3

$$a_2 = W^{2T} \cdot a^1$$

$$= \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

This output indicate that p_3 is a member of class 2. This is correct so w_{33} is update by moving it towards p_3

$$w_{33}^1(1) = w_{33}(0) + \eta(p_3 - w_{33}(0))$$

$$= \begin{bmatrix} 0.997 \\ 0.094 \end{bmatrix} + 0.5\left(\begin{bmatrix} 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 0.997 \\ 0.094 \end{bmatrix}\right)$$

$$= \begin{bmatrix} 0.998 \\ -0.453 \end{bmatrix}$$

DATE: / /

Self Organising Map →

Kohonen rule.

$$_iw(q) = {}_iw(q-1) + \eta\,(P(q) - {}_iw(q-1))$$

$$= (1-\alpha)\,{}_iw(q-1) + \alpha\, p(q)$$

and

~~$w(q) = w(q-1)$ if i~~

$$_iw(q) = {}_iw(q-1) \quad i \neq i^*$$

~~the weight matrix that~~

$$P_1 = \begin{bmatrix} -0.1961 \\ 0.9806 \end{bmatrix}, P_2 = \begin{bmatrix} .1961 \\ .9806 \end{bmatrix}, P_3 = \begin{bmatrix} .9806 \\ .1961 \end{bmatrix}, P_4 = \begin{bmatrix} .9806 \\ -.1961 \end{bmatrix}$$

$$P_5 = \begin{bmatrix} -.5812 \\ -.8137 \end{bmatrix}, P_6 = \begin{bmatrix} -.8137 \\ -.5812 \end{bmatrix}$$

$$_1w = \begin{bmatrix} w_{11} \\ w_{12} \end{bmatrix} = \begin{bmatrix} .7071 \\ -.7071 \end{bmatrix}, {}_2w = \begin{bmatrix} .7071 \\ .7071 \end{bmatrix}, {}_3w = \begin{bmatrix} -1.0 \\ 0.0 \end{bmatrix}$$

$$w = \begin{bmatrix} {}_1w \\ {}_2w \\ {}_3w \end{bmatrix} = \begin{bmatrix} .7071 & .7071 & -1.0 \\ -.7071 & .7071 & 0.0 \end{bmatrix}$$

for P_2

$$a_4 = \text{compet}(w^T \cdot P_2) = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

Second neuron's weight weight vector is won.

Be Positive.....

DATE: __/__/

We apply Kohonen learning rule to winning neuron with $\eta = 0.5$

$$_2w(1) = {}_2w(0) + \eta\left(p_2 - {}_2w(0)\right) = \begin{bmatrix} .4516 \\ .8438 \end{bmatrix}$$

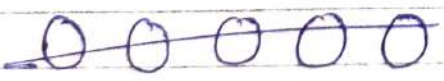
$_2w$ is closer to p_2

Self Organizing feature Map (SOFM) network first determines the winning neuron i^* using the same procedure as the competitive layer.
Next the weight vectors for all neuron within a certain neighborhood of the winning neuron are updated using the Kohonen rule.

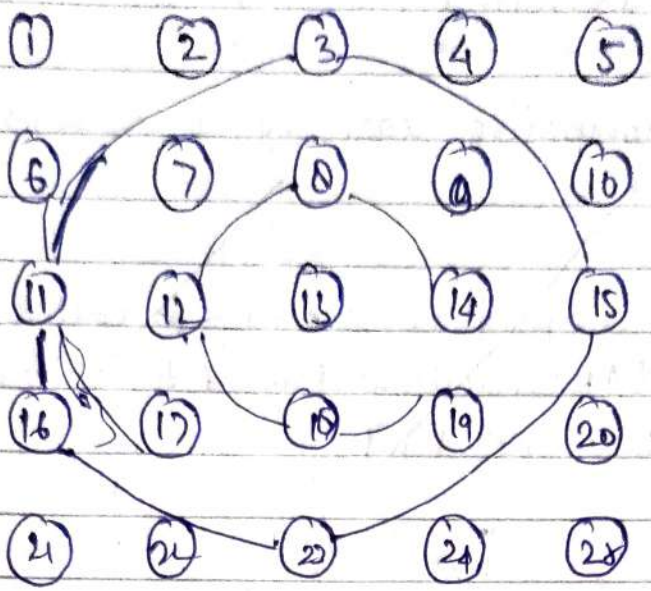
$$_iw(q) = {}_iw(q-1) + \eta\left(p(q) - {}_iw(q-1)\right)$$
$$= (1-\eta)\,{}_iw(q-1) + \eta p(q)$$
$$i \in N_{i^*}(d),$$

where the neighborhood $N_{i^*}(d)$ contains the indices for all of the neurons that lie within a radius d of winning neuron i^*:

$$N_i(d) = \{j, d_{ij} \le d\}$$



DATE: / /

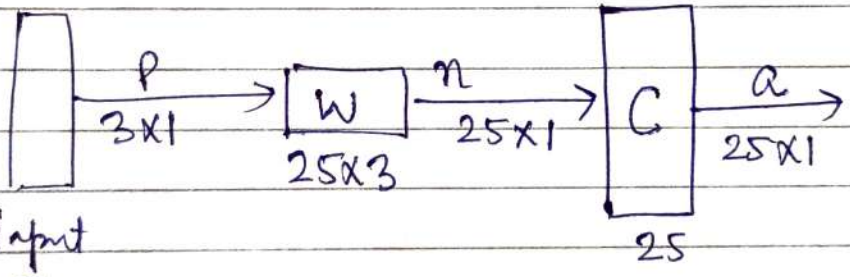


$N_{13}(1)$

$N_{13}(1) = \{8, 12, 13, 14, 18\}$

$N_{13}(2) = \{3, 7, 8, 9, 11, 12, 13, 14, 15, 17, 18, 19, 23\}$

Kohonen has suggested rectangular and hexagonal neighborhoods for different implementation.



$a = compet(W^T p)$

feature map

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25