Introduction
1. Learning without examples
2. Data are input to the system one by one
3. Mapping the data into one or more dimension space
4. Competitive learning

Hamming Distance
1. Define HD (Hamming Distance) between two binary codes $A$ and $B$ of the same length as the number of place in which $a_i$ and $b_i$ differ.
2. Ex:
   
   $A: [1 \ -1 \ 1 \ 1 \ 1 \ 1]$  
   $B: [1 \ 1 \ -1 \ -1 \ 1 \ 1]$  
   
   $=> HD(A, B) = 3$
**Hamming Distance (4/5)**

3. Or, HD can be defined as the lowest number of edges that must be traversed between the two relevant codes.

4. If $A$ and $B$ are bipolar binary components, then the scalar product of $A$, $B$:

**Hamming Distance (5/5)**

$$A \cdot B = \left[ n - HD(A, B) \right] - HD(A, B)$$

$$= n - 2HD(A, B)$$

---

**Hamming Net and MAXNET (1/30)**

1. For a two layer classifier of binary bipolar vectors, and $p$ classes, $p$ output neurons, the strongest response of a neuron indicates it has the minimum HD value and the category this neuron represents.

**Hamming Net and MAXNET (2/30)**

Figure 7.1: Block diagram of the minimum HD classifier

---

**Hamming Net and MAXNET (3/30)**

2. The MAXNET operates to suppress values at output instead of output values of Hamming network.

3. For the Hamming net in next figure, we have input vector $X$

   - $p$ classes => $p$ neurons for output

   - output vector $Y = [y_1, \ldots, y_p]$
Hamming Net and MAXNET(5/30)

4. for any output neuron, \( m, m=1, \ldots, p \), we have \( W_m = [w_{m1}, w_{m2}, \ldots, w_{mp}]^T \) and \( m=1,2,\ldots,p \)

to be the weights between input \( X \) and each output neuron.

5. Also, assuming that for each class \( m \), one has the prototype vector \( S^{(m)} \) as the standard to be matched.

Hamming Net and MAXNET(6/30)

6. For classifying \( p \) classes, one can say the \( m^{th} \) output is 1 if and only if

\[ X = S^{(m)} \Rightarrow \text{happens only if } W^{(m)} = S^{(m)} \]

output for the classifier are

\[ X^{S^{(1)}}, X^{S^{(2)}}, \ldots, X^{S^{(m)}}, \ldots, X^{S^{(p)}} \]

So when \( X = S^{(m)} \), the \( m^{th} \) output is \( n \) and other outputs are smaller than \( n \).

Hamming Net and MAXNET(7/30)

7. \[ X^{S^{(m)}} = (n \cdot HD(X, S^{(m)}) - HD(X, S^{(m)})) \]

\[ \frac{1}{2} X^{S^{(m)}} = n/2 - HD(X, S^{(m)}) \]

So the weight matrix:

\[ W_{m} = 1/2 S^{(m)} \]

\[ W_{H} = \frac{1}{2} \begin{bmatrix} S^{(1)}_1 & S^{(1)}_2 & \ldots & S^{(1)}_n \\ S^{(2)}_1 & S^{(2)}_2 & \ldots & S^{(2)}_n \\ \vdots & \vdots & \ddots & \vdots \\ S^{(p)}_1 & S^{(p)}_2 & \ldots & S^{(p)}_n \end{bmatrix} \]

Hamming Net and MAXNET(8/30)

8. By giving a fixed bias \( n/2 \) to the input then

\[ net_m = \frac{1}{2} X^{S^{(m)}} + n/2 \text{ for } m=1,2,\ldots,p \]

or

\[ net_m = n - HD(X, S^{(m)}) \]

Hamming Net and MAXNET(9/30)

9. To scale the input 0~n to 0~1 down, one can apply transfer function as

\[ f(net_m) = net_m \text{ for } m=1,2,\ldots,p \]

Hamming Net and MAXNET(10/30)

- Graph showing the activation function.
Hamming Net and MAXNET(11/30)

10. So for the node with the highest output means that the node has smallest HD between input and prototype vectors $S^{(1)}, \ldots, S^{(m)}$ i.e.

- $f(\text{net}_m) = 1$
- for other nodes $f(\text{net}_m) < 1$

Hamming Net and MAXNET(12/30)

11. MAXNET is employed as a second layer only for the cases where an enhancement of the initial dominant response of $m^{th}$ node is required. i.e., the purpose of MAXNET is to let $\max\{y_1, \ldots, y_p\}$ equal to 1 and let others equal to 0.

Hamming Net and MAXNET(14/30)

12. To achieve this, one can let

$$Y_i^{k+1} = f\left(Y_i^k - \varepsilon \sum_j (\varepsilon \cdot y_j^k)\right)$$

where $i = 1, \ldots, p$; $j = 1, \ldots, p$; $j \neq i$

$\therefore y_i$ is bounded by $0 \leq y_i \leq 1$ for $i = 1, \ldots, p$

output of Hamming Net.

And $Y_i^{k+1}$ can only be 1 or 0.

Hamming Net and MAXNET(15/30)

13. So $\varepsilon$ is bounded by $0 < \varepsilon < 1/p$

and

$$W_{\varepsilon} = \begin{bmatrix} 1 & -\varepsilon & -\varepsilon & \cdots & -\varepsilon \\ -\varepsilon & 1 & -\varepsilon & \cdots & -\varepsilon \\ -\varepsilon & -\varepsilon & 1 & \cdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & -\varepsilon \\ -\varepsilon & -\varepsilon & \cdots & -\varepsilon & 1 \end{bmatrix}_{(p \times p)}$$

$\varepsilon$: lateral interaction coefficient

Hamming Net and MAXNET(16/30)

And

$$\text{net}^k = W_M \cdot y^k$$
**Hamming Net and MAXNET(17/30)**

14. So the transfer function

\[ f(\text{net}) = \begin{cases} 
  0 & \text{net} < 0 \\
  \text{net} & \text{net} \geq 0 
\end{cases} \]

---

**Hamming Net and MAXNET(18/30)**

---

**Hamming Net and MAXNET(19/30)**

Ex: To have a Hamming Net for classifying C, I, T then

\[ S^{(1)} = [ 1 1 1 1 -1 -1 1 1 ] \]
\[ S^{(2)} = [ -1 1 -1 -1 1 -1 1 1 ] \]
\[ S^{(3)} = [ 1 1 1 -1 1 -1 -1 1 ] \]

---

**Hamming Net and MAXNET(20/30)**

---

**Hamming Net and MAXNET(21/30)**

So

\[ W_n = \begin{bmatrix}
  1 & 1 & 1 & 1 & -1 & -1 & 1 & 1 & 1 \\
  -1 & 1 & -1 & -1 & 1 & -1 & -1 & 1 & 1 \\
  1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 
\end{bmatrix} \]

\[ \text{net} = \frac{1}{2} W_n X + \frac{n}{2} \]

---

**Hamming Net and MAXNET(22/30)**

For

\[ X = [1 1 1 1 1 1 1 1] \]
\[ \text{net} = [7 3 5] \]

And

\[ f(\text{net}) = \begin{bmatrix}
  7/9 \\
  3/9 \\
  5/9 
\end{bmatrix} \]

\[ = Y \]
Hamming Net and MAXNET(23/30)

Input to MAXNET and select $\varepsilon=0.2 < 1/3(=1/p)$

So

$$W_M = \begin{bmatrix} 1 & 1/5 & 1/5 \\ 1/5 & 1 & 1/5 \\ 1/5 & 1/5 & 1 \end{bmatrix}$$

Hamming Net and MAXNET(24/30)

And

$$net^k = W_M Y^k$$

$$Y^{k+1} = f(net^k)$$

Hamming Net and MAXNET(25/30)

$k=0$

$$net^0 = \begin{bmatrix} 1 & -0.2 & -0.2 \\ -0.2 & 1 & -0.2 \\ -0.2 & -0.2 & 1 \end{bmatrix} \begin{bmatrix} 0.777 \\ 0.333 \\ 0.555 \end{bmatrix}$$

$$= \begin{bmatrix} 0.599 \\ 0.067 \\ 0.333 \end{bmatrix} > \alpha$$

$$\therefore Y^1 = f(net^0) = \begin{bmatrix} 0.599 \\ 0.067 \\ 0.333 \end{bmatrix}$$

Hamming Net and MAXNET(27/30)

$k=1$

$$net^1 = \begin{bmatrix} 0.520 & -0.120 & 0.120 \end{bmatrix}$$

$$Y^2 = [0.520 \ 0 \ 0.120]$$

Hamming Net and MAXNET(28/30)

$k=2$

$$net^2 = \begin{bmatrix} 0.480 & -0.14 & 0.096 \end{bmatrix}$$

$$Y^3 = [0.480 \ 0 \ 0.096]$$

Hamming Net and MAXNET(29/30)

$k=3$

$$net^3 = \begin{bmatrix} 0.461 & -0.115 & -10^{-7} \end{bmatrix}$$

$$Y^4 = [0.461 \ 0 \ 0]$$
Hamming Net and MAXNET(30/30)

Summary:
Hamming network only tells which class it will mostly like, not to restore distorted pattern.

Clustering—Unsupervised Learning(1/20)

1. Introduction
   a. to categorize or cluster data
   b. grouping similar objects and separating of dissimilar ones

2. Assuming pattern set \( \{X_1, X_2, \ldots, X_N\} \) is submitted to determine the decision function required to identify possible clusters,
   \( \Rightarrow \) by similarity rules

Clustering—Unsupervised Learning(2/20)

Euclidean Distance:
\[
|X - \bar{X}| = \sqrt{(X - \bar{X})^T (X - \bar{X})}
\]

\[
\cos \psi = \frac{X \cdot \bar{X}}{|X||\bar{X}|}
\]

Clustering—Unsupervised Learning(3/20)

3. Winner-Take-All learning
   Assuming input vectors are to be classified into one of specific number of \( p \) categories according to the clusters detected in the training set \( \{X_1, X_2, \ldots, X_N\} \).

Figure 7.5 Measures of similarity for clustering data: (a) distance and (b) a normalized scalar product.
Clustering—Unsupervised Learning (6/20)

Kohonen Network

\[ y = f(Wx) \]

for vector size = \( n \)

\[ W = \left[ \begin{array}{c} W'_1 \\ W'_2 \\ \vdots \\ W'_p \end{array} \right] \]

and

\[ \hat{W}_i = [w_{i1}, \ldots, w_{ip}] \quad \text{for} \ i = 1, \ldots, p \]

Clustering—Unsupervised Learning (7/20)

prior to the learning, normalization of all weight vectors is required

So

\[ \hat{W}_i = \frac{W_i}{\| W_i \|} \]

Clustering—Unsupervised Learning (8/20)

The meaning of training is to find \( \hat{W}_m \) from all such that

\[ \| x - \hat{W}_m \| = \min_{\| x - W_i \|} \| x - \hat{W}_i \| \]

i.e., the \( m^{th} \) neuron with vector is the closest approximation of the current \( X \).

Clustering—Unsupervised Learning (9/20)

From

\[ \| x - \hat{W}_m \| = \min_{\| x - W_i \|} \| x - \hat{W}_i \| \]

\[ \Rightarrow \| x - \hat{W}_m \| = \left( \sum_{i=1}^{p} (x'_i - 2\hat{W}''_m x'_i + 1) \right)^{1/2} \]

and

\[ \min_{\| x - W_i \|} = \max_{\| x - W_i \|} \left\{ \hat{W}''_m x'_i \right\} \]

Clustering—Unsupervised Learning (10/20)

So

\[ \hat{W}''_m x'_i = \max_{\| x - W_i \|} \left\{ \hat{W}''_m x'_i \right\} \]

The \( m^{th} \) neuron is the winning neuron and has the largest value of \( net_i, \ i = 1, \ldots, p \).

Clustering—Unsupervised Learning (11/20)

Thus \( W_m \) should be adjusted such that

\[ \| X - W_m \| \]

is reduced.

Then \( \| X - W_m \| \) can be reduced by the direction of gradient.

\[ \nabla W_m \| X - W_m \|^2 = -2(X - W_m) \]

Or,

increase in \( (X - W_m) \) direction.
For any \( X \) is just a single step, we want only fraction of \((X - W_m)\), thus for neuron \( m \):
\[
\nabla W'_m = \alpha (X - W_m) \quad 0.1 < \alpha < 0.7
\]
for other neurons:
\[
\nabla W' = 0
\]

In general:
\[
W^{i+1} = \hat{W} + \alpha^i (X - \hat{W}) \quad \text{where } m \text{: winning neuron}
\]
\[
\alpha : \text{learning constant}
\]
\[
W_i = \hat{W} \quad \text{for } i \neq m
\]

4. Geometrical interpretation
input vector \( \hat{x} \) (normalized) and weight \( \hat{w} \) is winner
\[
\Rightarrow \max \left\{ \hat{w}_i \cdot \hat{x} \right\} \quad \text{for } i = 1, \ldots, p
\]

So \( \hat{x} - \hat{w} \) is created.
And
\[
W'_n = \hat{W}_n + \Delta \hat{W}_n
\]
\[
= \hat{W}_n + \alpha \left( \hat{x} - \hat{w} \right)
\]

Thus the weight adjustment is mainly the rotation of the weight vector toward input vector without a significant length change.
And \( W'_m \) is not a normal, so in the new training stage, \( W'_m \) must be normalized again.
\( \hat{w} \) is a vector point to the gravity of each cluster on a unity sphere.
Clustering—Unsupervised Learning (19/20)

5. In the case of some patterns are known class, then
\[
\Delta W_w = \alpha \left( X - \bar{W}_w \right)
\]
\(\alpha > 0\) for correct node
\(\alpha < 0\) otherwise
This will accelerate the learning process significantly.

Feature Map (1/6)

1. Transform from high dimensional pattern space to low-dimension feature space.
2. Feature extraction:
   two category: natural structure
   no natural structure
   -- depend on pattern is similar to human perception or not

Feature Map (2/6)

Figure 7.10 Pattern structure: (a) natural similarity and (b) no natural similarity.

Feature Map (3/6)

3. another important aspect is to represent the feature as natural as possible.
4. Self organizing neural array that can map features from pattern space.

Feature Map (4/6)

5. Use one dimensional array or Two dimensional array
6. Example:
\[
X: \text{input vector} \\
i: \text{neuron } i \\
w_{ij}: \text{weight from input } x_j \text{ to neuron } i \\
y_i: \text{output of neuron } i
\]
1. One dimensional mapping
   set of pattern $X_i$, $i=1,2,\ldots$
   use a linear array $i_1 > i_2 > i_3 > \ldots$
   if $y_1 = \max_j \{y(X_j), i = 1,2,\ldots\}$
   $y_2 = \max_j \{y(X_j), i = 1,2,\ldots\}$
   $\vdots$

2. Thus, this learning is to find the best matching neuron cells which can activate their spatial neighbors to react to the same input.

3. Or to find $W_c$ such that
   $$\|X - W_c\| = \min_i \|X - W_i\|$$

4. In case of two winners, choose lower $i$.

5. If $c$ is the winning neuron then define $N_c$ as the neighborhood around $c$ and $N_c$ is always changing as learning going.

6. Then
   $$\Delta w_y = \begin{cases} \alpha (x_j - w_y) & \text{if } i \text{ is in the neighborhood } N_c \\ 0 & \text{otherwise} \end{cases}$$

7. So define
   $$y_1 = f(S(X, W_c))$$
   where $S(X, W_c)$ means the similarity between $X$ and $W_c$

8. Thus, (b) is the result of (a)

9. $\alpha$: $\alpha_t = \alpha_0 \left(1 - \frac{t}{T}\right)$
   where $t$: current training iteration
   $T$: total # of training steps to be done.

10. $\alpha$ starts from $\alpha_0$ and is decreased until it reaches value of 0.
Self-organizing Feature Map(5/6)

9. If rectangular neighborhood is used then
   \[ c-d < x < c+d \]
   \[ c-d < y < c+d \]
   and as learning continues
   \[ d = d_0 \left( 1 - \frac{t}{T} \right) \]
   \[ d \] is decreased from \( d_0 \) to 1

Self-organizing Feature Map--Example

10. Example:
    a. Input patterns are chosen randomly from \( U[0,1] \).
    b. The data employed in the experiment comprised 500 points distributed uniformly over the bipolar square \([-1,1] \times [-1, 1] \)
    c. The points thus describe a geometrically square topology.
    d. Thus, initial weights can be plotted as in next figure.
    d. “-” connection line connects two adjacent neuron in competitive layer.