★ Introduction to Neuro-Fuzzy and Soft Computing →

Soft Computing is an emerging approach to computing which parallels the remarkable ability of the human mind to reason and learn in an environment of uncertainity & imprecision.

In confronting real-world computing problems, it is frequently advantageous to use several computing techniques synergistically rather than exclusively, resulting in construction of complementary hybrid intelligent systems. The quintessence of designing Is of this kind is neuro-fuzzy computing: neural nws that recognize patterns & adapt themselves to cope with changing environments; fuzzy inference systems that incorporate human knowledge & perform inferencing & decision making. The integration of these two complementary approaches, together with certain derivative-free optimization techniques, results in a novel discipline called neuro-fuzzy & soft computing.

Neuro-Fuzzy & Soft computing characteristics →

① Human expertise → Utilizes human expertise in the form of fuzzy if-then rules as well as in conventional knowledge rep? to solve practical problems.

② Biologically inspired computing models are used

③ New optimization techniques (GA, SA, RS, ASM) are used.

④ Soft computing relies mainly on numerical computation.

⑤ Soft computing has found a lot of new application domains such as, adaptive signal processing, adaptive control, non-linear system, pattern recognition.

② Cross-

out assuming too much background knowledge of problem solved, They rely heavily on high-speed number-crunching ulations to find rules or regularity in data sets.

tolerance exist.

mputing is an integrated approach that can usually specific techniques within subtasks to construct generally tory solutions to real-world problems

OBC

In classical set, element belong or doesn't belong is given. In fuzzy, degree to which it belongs to set is given

* Fuzzy Sets →

If X is a collection of objects denoted by x, then fuzzy set $\tilde{A}$ in X is a set of ordered pairs which is represented as.

$$\tilde{A} = \{ x, \mu_{\tilde{A}}(x) \mid x \in X \}$$

where X → Universe of discourse

$0 \leq \mu_{\tilde{A}}(x) \leq 1$

↑ membership function

$\mu_{\tilde{A}} = 0 \rightarrow$ No membership

$= 1 \rightarrow$ full membership

* Different notations for representing fuzzy set :-

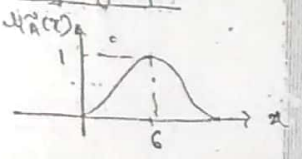
① Ordered Pair :- $\tilde{A} = \{ (x_1, \mu_{\tilde{A}}(x_1)), \ldots \}$

eg $\tilde{6} = \{ (5, 0.8), (7, 0.1), (8, 0.2) \ldots \}$

② As a sum of membership function :-

$$\tilde{A} = \left\{ \frac{\mu_{\tilde{A}}(x_1)}{x_1} + \ldots \right\} \quad \text{eg} \quad \tilde{6} = \left\{ \frac{0.1}{3} + \frac{0.3}{4} + \ldots \right\}$$

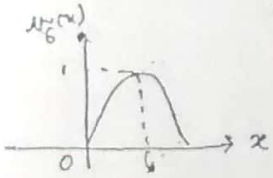
③ Using graphs :-



$$\mu_{\tilde{6}}(x) = \frac{1}{1+(x-6)^2}$$

* Q1) Model a fuzzy set to represent "Numbers close to 6".

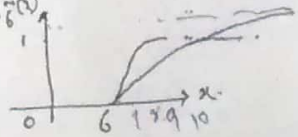
$\tilde{6} = \{ (6,1), (5,0.8), (4,0.7), (3,0.6), (2,0.8), (7,0.8), (8,0.7), (9,0.6), (10,0.5) \}$



$$\mu_{\tilde{6}}(x) = \frac{1}{1+(x-6)^2} \rightarrow (6,1), (5,0.5), (4,0.2), (3,0.1), (7,0.5), (8,0.2), (9,0.1)$$

* Q2) Model a fuzzy set to represent "Numbers larger than 6".

$\tilde{6} = \{ (6,0.0), (7,0.1), (8,0.2), (9,0.3), (10,0.4) \}$



$$\mu_{\tilde{B}}(x) = \begin{cases} 0 & x \leq 6 \\ \dfrac{1}{1+(x-6)^2} & x > 6 \end{cases}$$

* Sol) $\tilde{A} = \left\{ \frac{0.1}{2} + \frac{0.5}{3} + \frac{0.3}{4} + \frac{0.2}{5} \right\}$ & $\tilde{B} = \left\{ \frac{0.5}{2} + \frac{0.7}{3} + \frac{0.2}{4} + \frac{0.4}{5} \right\}$

Find complement, union, intersection, difference & prove Demorgan's law.

→ **Complement**

$$\overline{A} = 1 - \mu_A(x) = \left\{ \frac{0.9}{2} + \frac{0.5}{3} + \frac{0.7}{4} + \frac{0.8}{5} \right\}$$

$$\overline{B} = 1 - \mu_B(x) = \left\{ \frac{0.5}{2} + \frac{0.3}{3} + \frac{0.8}{4} + \frac{0.6}{5} \right\}$$

relative complement is not possible.

Union

$$\mu_{A \cup B}(x) = \left\{ \frac{0.5}{2} + \frac{0.7}{3} + \frac{0.3}{4} + \frac{0.4}{5} \right\}$$

Intersection

$$\mu_{A \cap B}(x) = \left\{ \frac{0.1}{2} + \frac{0.5}{3} + \frac{0.2}{4} + \frac{0.2}{5} \right\}$$

Difference

$B/A = B \cap \overline{A}$

$$B/A = B \cap \overline{A} = \left\{ \frac{0.5}{2} + \frac{0.5}{3} + \frac{0.2}{4} + \frac{0.4}{5} \right\}$$

$A/B = A \cap \overline{B}$

$$A/B = A \cap \overline{B} = \left\{ \frac{0.5}{2} + \frac{0.3}{3} + \frac{0.3}{4} + \frac{0.2}{5} \right\}$$

Demorgan's law

$$\overline{A \cup B} = \overline{A} \cap \overline{B}$$

$\overline{A \cup B} =$

$$\overline{A \cup B} = \left\{ \frac{0.5}{2} + \frac{0.3}{3} + \frac{0.7}{4} + \frac{0.6}{5} \right\} \quad \ldots$$

$$\overline{A} \cap \overline{B} = \left\{ \frac{0.5}{2} + \frac{0.3}{3} + \frac{0.7}{4} + \frac{0.6}{5} \right\} \quad \therefore \text{LHS} = \text{RHS}$$

$$\overline{A \cap B} = \overline{A} \cup \overline{B}$$

$$\overline{A} \cup \overline{B} = \left\{ \frac{0.9}{2} + \frac{0.5}{3} + \frac{0.8}{4} + \frac{0.8}{5} \right\} \quad \therefore \text{LHS} = \text{RHS}.$$

$$\overline{A \cap B} = \left\{ \frac{0.9}{2} + \frac{0.5}{3} + \frac{0.8}{4} + \frac{0.8}{5} \right\}$$

* **Basic Definition and Terminology →**

① **Support of a fuzzy set:** —

$$S(A) = \{ x \in X \mid \mu_A(x) > 0 \}$$

eg. $X = \{2, 4, 5, 7, 8\}$ & $\tilde{A} = \{(2,0)\ (4,0.2)\ (5,0.5)\ (7,0.8)\ (8,1)\}$

$\therefore S(A) = \{4, 5, 7, 8\}$.

A fuzzy set whose support is single element with membership 1 is called as fuzzy singleton. $\tilde{A} \cap \tilde{A} = \mu_A$

② **Cross-over point:** —

$$C(A) = \{ x \in X \mid \mu_A(x) = 0.5 \}$$

eg. $C(A) = \{5\}$

Mele Operations on fuzzy sets

Algebric sum: $(\tilde{A} + \tilde{B})$ of fuzzy sets $\mu_{A+B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x).\mu_B(x)$

Product: $(\tilde{A} \cdot \tilde{B}) = \mu_{AB}(x) = \mu_A(x) \cdot \mu_B(x)$

Bounded sum $(A \oplus B)$ of A and B $= \mu_{A \oplus B}(x) = \min\{1, \mu_A(x) + \mu_B(x)\}$

Bounded diff $(A \ominus B)$ of A and B $=$

$\mu_{A \ominus B}(x) = \max\{0, \mu_A(x) - \mu_B(x)\}$.

★ Properties of fuzzy set :-

① Commutative

$\tilde{A} \cup \tilde{B} = \tilde{B} \cup \tilde{A}$

$\tilde{A} \cap \tilde{B} = \tilde{B} \cap \tilde{A}$

② Associative

$\tilde{A} \cup (\tilde{B} \cup \tilde{C}) = (\tilde{A} \cup \tilde{B}) \cup \tilde{C}$

$\tilde{A} \cap (\tilde{B} \cap \tilde{C}) = (\tilde{A} \cap \tilde{B}) \cap \tilde{C}$

③ Idempotency

$\tilde{A} \cup \tilde{A} = \tilde{A}$

$\tilde{A} \cap \tilde{A} = \tilde{A}$

④ Distributive

$\tilde{A} \cup (\tilde{B} \cap \tilde{C}) = (\tilde{A} \cup \tilde{B}) \cap (\tilde{A} \cup \tilde{C})$

$\tilde{A} \cap (\tilde{B} \cup \tilde{C}) = (\tilde{A} \cap \tilde{B}) \cup (\tilde{A} \cap \tilde{C})$

⑤ Identity

$\tilde{A} \cup \phi = \tilde{A}$, $\tilde{A} \cap X = \tilde{A}$

$\tilde{A} \cap \phi = \phi$, $\tilde{A} \cup X = X$

⑥ Transitivity

If $A \subseteq B \subseteq C$ then $A \subseteq C$

⑦ Demorgan's law

$\overline{A \cup B} = \overline{A} \cap \overline{B}$ & $\overline{A \cap B} = \overline{A} \cup \overline{B}$

★ Set-theoretic Operations :-

① Empty set → $\mu_\phi(x) = 0$, membership of all $x = 0$, $\tilde{A} = \{\}$

② Universal set → $\mu_X(x) = 1$, membership of all $x = 1$

③ Comparision → is $\tilde{A} = \tilde{B}$

$\tilde{A} = \tilde{B} \Rightarrow \mu_A(x) = \mu_B(x)$; $\forall x \in X$

④ Containment → A is in B

$\tilde{A} \subset \tilde{B}$ if $\mu_{\tilde{A}}(x) < \mu_{\tilde{B}}(x)$, $\forall x \in X$

⑤ Union → $\tilde{C} = \tilde{A} \cup \tilde{B}$

where, $\mu_{\tilde{C}}(x) = \max\{\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)\}$

⑥ Intersection → $\tilde{C} = \tilde{A} \cap \tilde{B}$

$\mu_{\tilde{A} \cap \tilde{B}}(x) = \min\{\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)\}$

⑦ complement →

i) Absolute Complement → $\overline{\mu_{\tilde{A}}(x)} = 1 - \mu_{\tilde{A}}(x)$

ii) Relative Complement → Complement of A w.r.t B is,

$B - A = \mu_{\tilde{B}}(x) - \mu_{\tilde{A}}(x)$ provided $\mu_{\tilde{B}}(x) > \mu_{\tilde{A}}(x)$

⑧ Difference →

$\tilde{A} / \tilde{B} = \tilde{A} \cap \overline{\tilde{B}}$

$\tilde{B} / \tilde{A} = \tilde{B} \cap \overline{\tilde{A}}$

⑨ Involution → $\overline{\overline{A}} = A$

⑩

⑤ α-level set or cut :-

$$A_\alpha = \{ x \in X \mid \mu_A(x) \geq \alpha \}$$

④ Strong α-level set cut :-

$$A'_\alpha = \{ x \in X \mid \mu_A(x) > \alpha \}$$

eg.

$A = \{ (2, 0.2), (3, 0.4), (4, 0.5), (5, 0.7), (6, 1) \}$ find α cut & Strong α cut for all possible values of α.

→

$A_{0.2} = \{2, 3, 4, 5, 6\}$	$A'_{0.2} = \{3, 4, 5, 6\}$
$A_{0.4} = \{3, 4, 5, 6\}$	$A'_{0.4} = \{4, 5, 6\}$
$A_{0.5} = \{4, 5, 6\}$	$A'_{0.5} = \{5, 6\}$
$A_{0.7} = \{5, 6\}$	$A'_{0.7} = \{6\}$
$A_1 = \{6\}$	$A'_1 = \{\phi\}$

⑤ Cardinality / Power :-

$$|A| = \Sigma \mu_A(x)$$

eg: $X = \{1 \ldots 10\}$ $A = \{(3, 0.1), (4, 0.2), (5, 0.3)\}$

$$\therefore |A| = 0.1 + 0.2 + 0.3 = 0.6$$

⑥ Relative Cardinality :-

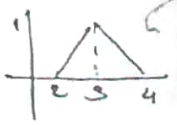
$$\|A\| = \frac{|A|}{|X|} = \frac{0.6}{10}$$

⑦ Width of fuzzy set :- number of elements with membership > 0

⑧ Nucleus of fuzzy set :-

$$\text{Nucleus}(A) = \{ x \in X \mid \mu_A(x) = 1 \}$$

$$N(A) = \{3\}$$



⑨ Height of fuzzy set :-

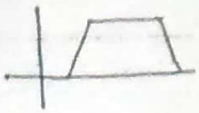
$$H(A) = \max(\mu_A(x))$$

If $H(A) = 1$ → normal fuzzy set, If $H(A) > 1$ → subnormal fuzzy set.

★ Types of Fuzzy Set →

① Convex :- Whose membership function has only one value at peak.

② Non Convex :- Membership function has more than one value at peak.

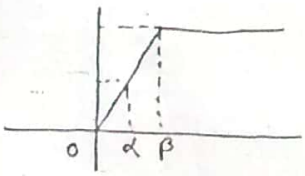


③ Increasing :- Always has increasing membership values are constant.

④ Decreasing :- Always has constant decreasing membership values.

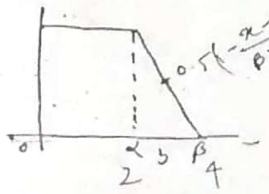
★ Member Function Formulation and Parameterization →

① Γ Function → (Two Parameter)



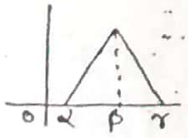
$$\Gamma(x,\alpha,\beta) = \begin{cases} 0 & x<\alpha \\ \alpha/\beta- & \alpha<x<\beta \\ 1 & x>\beta \end{cases}$$

② L Function → (Two Parameter)



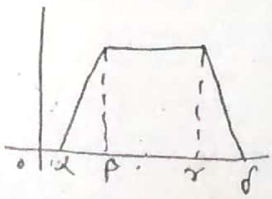
$$L(x,\alpha,\beta) = \begin{cases} 1 & x<\alpha \\ \beta-x/\beta-\alpha & \alpha<x<\beta \\ 0 & x>\beta \end{cases}$$

③ Λ Function → (Three Parameter)



$$\Lambda(x,\alpha,\beta,\gamma) = \begin{cases} 0 & x \\ -\alpha/\beta & \alpha \le x \le \beta \\ -x/ & \beta \le x \le \gamma \\ 0 & \gamma \end{cases}$$

④ Π function → (Four Parameter)



$$\Pi(x,\alpha,\beta,\gamma,\delta) = \begin{cases} 0 & x<\alpha \\ /\beta-\alpha & \alpha<x<\beta \\ 1 & \beta<x<\gamma \\ /\delta-\gamma & \gamma<x<\delta \\ 0 & x>\delta \end{cases}$$

★ Extension Principle →

The extension principle is a basic concept of fuzzy set theory that provides a general procedure for extending crisp domains of mathematical expressions to fuzzy domains.

Suppose f is a function from X to Y and A is a fuzzy set on X defined as, $A = \mu_A(x_1)/x_1 + \mu_A(x_2)/x_2 + \dots + \mu_A(x_n)/x_n$

- Then the extension principle states that the image of fuzzy set A
- under the mapping $f(\cdot)$ can be expressed as a fuzzy set B,

$$B = f(A) = \mu_A(x_1)/y_1 + \mu_A(x_2)/y_2 + \dots + \mu_A(x_n)/y_n$$

where $y_i = f(x_i)$, $i = 1 \dots n$.

eg.

$$A = \frac{0.1}{-2} + \frac{0.4}{-1} + \frac{0.8}{0} + \frac{0.9}{1} + \frac{0.3}{2} \quad \& \quad f(x) = x^2 - 3$$

Upon applying Extension principle, we have

$$B = \frac{0.1}{1} + \frac{0.4}{-2} + \frac{0.8}{-3} + \frac{0.9}{-2} + \frac{0.3}{1}$$

$$= \frac{0.8}{-3} + \frac{0.3}{1} + \frac{0.9}{-2}$$

Suppose that function f is a mapping from an n-dimensional Cartesian product space $X_1 \times X_2 \times \dots \times X_n$ to a one-dimensional universe Y such that $y = f(x_1, \dots, x_n)$ & suppose $A_1, \dots, A_n$ are n fuzzy sets in $X_1, \dots, X_n$ resp. Then the extension principle asserts that the fuzzy set B induced by the mapping f is defined by.

$$\mu_B(y) = \begin{cases} \max_{(x_1,\dots,x_n),\, f(x_1,\dots,x_n)=y} [\min_i \mu_{A_i}(x_i)] & \text{if } f^{-1}(y) \neq \phi \\ 0 & \text{if } f^{-1}(y) = \phi \end{cases}$$

② Multiple disjunctive antecedents →

IF x is A_1 OR x is A_2, --- OR x is A_n, THEN y is

This can be written as.

IF x is A_n THEN y is B_m.

where $A_m = A_1 \cup A_2 \cdots \cup A_n$.

based on fuzzy union op^n $\mu_{A_m}(x) = \max [\mu_{A_1}(x) \cdots \mu_{A_n}(x)]$

⑤ Conditional Statements (with ELSE and UNLESS) →

Stmts of the kind ~~*~~ IF A_1 THEN (B_1 ELSE B_2) can be decomposed into two simple canonical rule forms, connected by "OR"

IF A_1 THEN B_1

OR

IF NOT A_1 THEN B_2

~~If~~

* IF A_1 (THEN B_1) UNLESS A_2 can be decomposed as

IF A_1 THEN B_1

OR

IF A_2 THEN NOT B_1

* IF A_1 THEN (B_1) ELSE IF A_2 THEN (B_2) can be decomposed as

IF A_1 THEN B_1

OR

IF NOT A_1 AND IF A_2 THEN B_2

④ Nested IF-THEN rules →

The rule " IF A_1 THEN [IF A_2 THEN (B_1)]" can be of the form

IF A_1 AND A_2 THEN B_1

Aggregation of Fuzzy Rules →

The rule-based system involves more than one rule. Aggregation of rules is the process of obtaining the overall consequents from the individual consequents provided by each rule. The foll. two methods exist that aid in determining the aggregation of fuzzy rules:

★ Fuzzy Relations →

It is a set of tuples where each tuple has membership degree between 0 and 1.

★ $x = \{1, 2, 3\}$ model a relation between x and y $(x = y)$

$$R = \left\{ \frac{1}{(1,1)} + \frac{1}{(2,2)} + \frac{1}{(3,3)} + \frac{0.8}{(1,2)} + \frac{0.5}{(1,3)} + \frac{0.8}{(2,1)} + \frac{0.8}{(2,3)} + \frac{0.5}{(3,1)} + \frac{0.8}{(3,2)} \right\}$$

x \ y	1	2	3
1	1	0.8	0.5
2	0.8	1	0.8
3	0.5	0.8	1

$$\mu_R = \begin{cases} 0.5 & |x-y| = 2 \\ 0.8 & |x-y| = 1 \\ 1 & x = y \end{cases}$$

★ Operations on fuzzy relations :-

① Intersection :-

$$R = \begin{bmatrix} 0.8 & 1 & 0.1 & 0.7 \\ 0 & 0.8 & 0 & 0 \\ 0.9 & 1 & 0.7 & 0.8 \end{bmatrix} \qquad S = \begin{bmatrix} 0.4 & 0 & 0.9 & 0.6 \\ 0.9 & 0.4 & 0.5 & 0.7 \\ 0.3 & 0 & 0.8 & 0.5 \end{bmatrix}$$

$$R \cap S = \begin{bmatrix} 0.4 & 0 & 0.1 & 0.6 \\ 0 & 0.4 & 0 & 0 \\ 0.3 & 0 & 0.7 & 0.5 \end{bmatrix}$$

② Union :-

$$R \cup S = \begin{bmatrix} 0.8 & 1 & 0.9 & 0.7 \\ 0.9 & 0.8 & 0.5 & 0.7 \\ 0.9 & 1 & 0.7 & 0.8 \end{bmatrix}$$

③ Projection :-

fuzzy set (rel) —projection→ fuzzy set

take a relation R. projection on $X = \frac{1}{x_1} + \frac{0.8}{x_2} + \frac{1}{x_3}$ (max value of row)

projection on $Y = \frac{0.9}{y_1} + \frac{1}{y_2} + \frac{0.7}{y_3} + \frac{0.8}{y_4}$

④ Cylindrical Extension :-

Fuzzy set $\xrightarrow{C.E}$ Fuzzy relation.

$A = \frac{1}{x_1} + \frac{0.8}{x_2} + \frac{1}{x_3} + \frac{0.6}{x_4}$ & $Y = \frac{1}{y_1} + \frac{1}{y_2} + \frac{1}{y_3} + \frac{1}{y_4}$

$ce(A) =$

1	1	1	1
0.8	0.8	0.8	0.8
1	1	1	1
0.6	0.6	0.6	0.6

⑤ Composition :-

1) $B = A \circ R$

A: set, R: relation.

B on Y = proj ($ce(A) \cap R$) on Y.

2) Max - min method :-

$T = R \circ S$

$\chi_T(x,z) = \underset{y \in Y}{V} [\chi_R(x,y) \wedge \chi_S(y,z)]$

3) Max - Product method :-

$T = R \circ S$

$\chi_T(x,z) = \underset{y \in Y}{V} [\chi_R(x,y) \cdot \chi_S(y,z)]$

★ (Q1) Given x = Ht in cm = {170, 172.5, 175, 177.5, 180, 182.5, 18

A relation bet^n x & y is given by, R = " x is somewhat taller th

find Y ?

→ $x = \left\{ \frac{0}{170} + \frac{0.1}{172.5} + \frac{0.4}{175} + \frac{0.7}{177.5} + \frac{0.9}{180} + \frac{1}{182.5} + \frac{1}{185} \right\}$

$ce(x) =$

170	0	0	0	0	0	0
172.5	0.1	0.1	0.1	0.1	0.1	0
175	0.4	0.4	0.4	0.4	0.4	0
177.5	0.7	0.7	0.7	0.7	0.7	0
180	0.9	0.9	0.9	0.9	0.9	0
182.5	1	1	1	1	1	1
185						

$R =$

	170	172.5	175	177.5	180	182.5	185
170	0.8	0	0	0	0	0	0
172.5	1	0.8	0	0	0	0	0
175	1	1	0.8	0	0	0	0
177.5	1	1	1	0.8	0	0	0
180	1	1	1	1	0.8	0	0
182.5	1	1	1	1	1	0.8	0
185	1	1	1	1	1	1	0.8

$Ce(2) \cap R =$

	0	0	0	0	0	0	0
	0.1	0.1	0	0	0	0	0
	0.4	0.4	0.4	0	0	0	0
	0.7	0.7	0.7	0.7	0	0	0
	0.9	0.9	0.9	0.9	0.9	0	0
	1	1	1	1	1	0.8	0
	1	1	1	1	1	1	0.8

$$\text{Proj on } y = \tilde{y} = \left\{ \frac{1}{y_1} + \frac{1}{y_2} + \frac{1}{y_3} + \frac{1}{y_4} + \frac{1}{y_5} + \frac{1}{y_6} + \frac{0.8}{y_7} \right\}$$

*Q2) Given, $R =$

	y_1	y_2
x_1	0.6	0.3
x_2	0.2	0.9

$S =$

	z_1	z_2	z_3
y_1	1	0.5	0.3
y_2	0.8	0.4	0.7

find $T = R \circ S$ ← $T = R \cdot S$

→ ① Max-min. $T = R \circ S$

$T =$

	z_1	z_2	z_3
x_1	0.6	0.5	0.3
x_2	0.8	0.4	0.7

$0.6 = \max[\min(0.6, 1), \min(0.3, 0.7)]$

$= \max(0.6, 0.3)$

$= 0.6$

② Max-product. $T = R \cdot S$

$T =$

	z_1	z_2	z_3
x_1	0.6	0.3	0.21
x_2	0.72	0.36	0.63

$\max(0.6 \times 1, 0.3 \times 0.7)$

$\max(0.6, 0.24) = 0.6$

0.6

0.24

* Types of fuzzy relation →

1) Classical relation.
2) Equivalance relation :-

Let R → relation on universe X

then R is Equivalance rel? if foll. properties are satisfied

① Reflexivity :- $X_R(x_i, x_i) = 1$ or $(x_i, x_i) \in R$

② Symmetry :- $X_R(x_i, x_j) = X_R(x_j, x_i)$

③ Transitivity :- $X_R(x_i, x_j)$ & $X_R(x_j, x_k) = 1$, so $X_R(x_i, x_k) = 1$

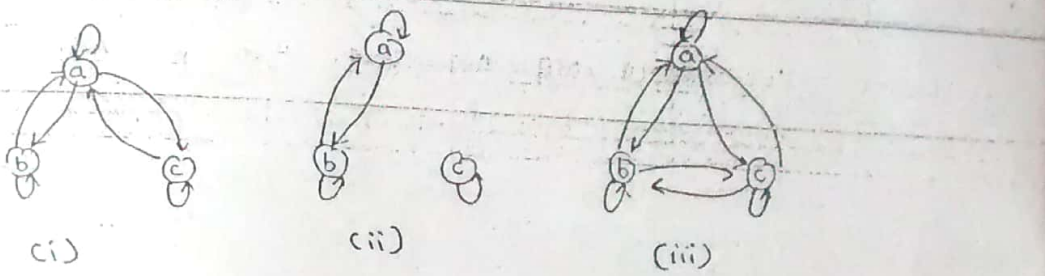
3) Tolerance / Proximity relation :-

A tolerance relation R_1 on universe X is one where the only the properties of reflexivity & symmetry are satisfied.

$R_1^{n-1} = R_1 \circ R_1 - \cdots \circ R_1 = R$

tolerance relation — Equivalance relation.

★ (Q1) $X = \{a, b, c\}$



(i) (ii) (iii)

(i) Equivalance → No, Transitivity not satisfied
Tolerance → yes.

(ii) Equivalance → No,
Tolerance → yes

(iii) Equivalance → yes.

★ "Fuzzification" →

Transforms a set to an approximating set that is more fuzzy.

$\underline{u_i}$ → constant & x_i is fuzzified → Support fuzzification

x_i → constant & u_i is fuzzified → Grade fuzzification

Support Fuzzification :- SF of A is denoted by $(A; K)$ & defined as,

$A = SF(A; K) = u_1 K(x_1) + \cdots + u_n K(x_n)$

where $u_i K(x_i)$ → fuzzy set that is product of scalar constant u_i & fuzzy set $K(x_i)$

eg $A = \frac{0.8}{1} + \frac{0.5}{2}$

assume, $K(1) = \frac{1}{1} + \frac{0.3}{2}$ & $K(2) = \frac{1}{2} + \frac{0.3}{1} + \frac{0.2}{3}$

Then, $SF(A;K) = 0.8\,K(1) + 0.5\,K(2)$

$$= 0.8\left(\frac{1}{1} + \frac{0.3}{2}\right) + 0.5\left(\frac{1}{2} + \frac{0.2}{1} + \frac{0.2}{3}\right)$$

$$= \frac{0.8}{1} + \frac{0.24}{2} + \frac{0.5}{2} + \frac{0.15}{1} + \frac{0.1}{3}$$

$$SF(A;K) = \frac{0.8}{1} + \frac{0.5}{2} + \frac{0.1}{3}$$

? § Grade Fuzzification:-

$$GF(A;K) = \frac{K(\mu_1)}{x_1} + \dots + \frac{K(\mu_n)}{x_n}$$

eg: $K(0.8) = \frac{1}{0.8} + \frac{0.7}{0.6} + \frac{0.3}{0.5}$ & $K(0.5) = \frac{1}{0.5} + \frac{0.6}{0.4} + \frac{0.5}{0.3}$

$$GF(A;K) = \frac{\left(\frac{1}{0.8} + \frac{0.7}{0.6} + \frac{0.3}{0.5}\right)}{1} + \frac{\left(\frac{1}{0.5} + \frac{0.6}{0.4} + \frac{0.5}{0.3}\right)}{2}$$

✱ Defuzzification Methods →

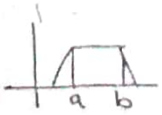
① Centroid Method → Centroid of Δ = $\frac{1}{3}$ x sum of co-ordinates

$$x^* = \frac{\sum A x}{\in A}$$

② Weighted Average Method → $x^* = \frac{\sum \text{centre} \times \text{Height}}{\sum \text{Height}}$

③ Centre of Sums → $x^* = \frac{\sum \text{centre} \times \text{Area}}{\sum \text{Area}}$

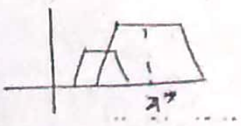
④ Mean of maxima → $x^* = \frac{a+b}{2}$



⑤ Height / Max membership principle → (only for convex)

$$\mu(x^*) \geq \mu(x) \quad ; \forall x \in X$$

⑥ Centre of largest area →

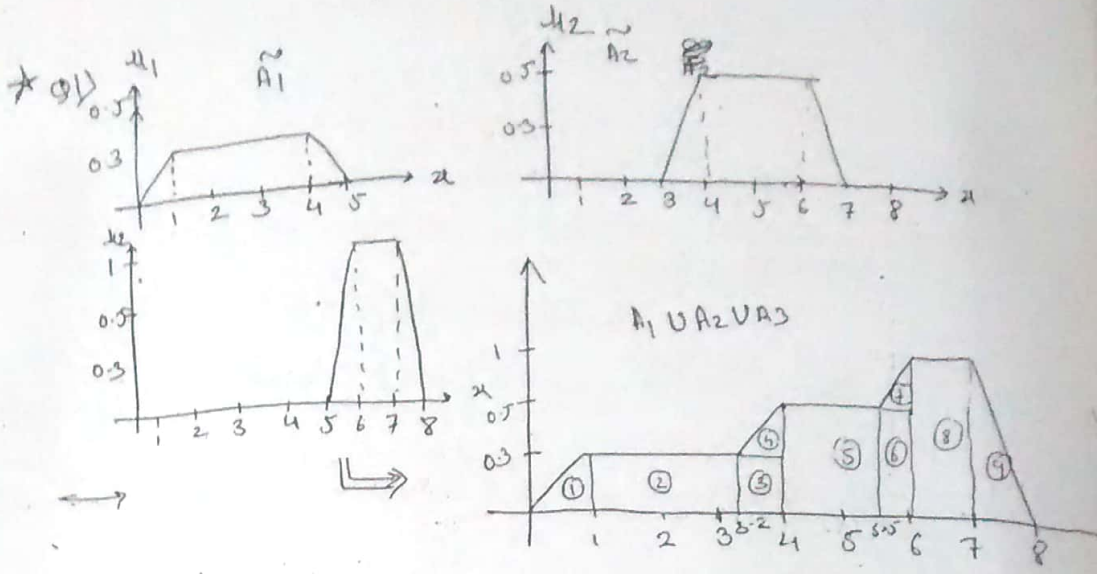


⑦ 1st or last of maxima →

1st = a

last = b.

★ Q1)



① Centroid →

Area no	Area (A)	$\bar{x}$ (centroid)	$A\bar{x}$
1	$\frac{1}{2}\times1\times0.3=0.15$	$0+1+1/3=0.66$	0.10
2	$2.2\times0.3=0.66$	$1+1+3.2+3.2/4=2.1$	1.39
3	$0.8\times0.3=0.24$	$3.2+3.2+4+4/4=3.6$	0.86
4	$\frac{1}{2}\times0.8\times0.2=0.08$	$3.2+4+4/3=3.74$	0.3
5	$1.5\times0.5=0.75$	$4+4+5.5+5.5/4=4.75$	3.56
6	$0.5\times0.5=0.25$	$5.5+5.5+6+6/4=5.75$	1.44
7	$\frac{1}{2}\times0.5\times0.5=0.125$	5.84	0.73
8	$1\times1=1$	$6+6+7+7/4=6.5$	6.5
9	$\frac{1}{2}\times1\times1=0.5$	7.34	3.67
	$\Sigma A = 3.76$		$\Sigma A\bar{x}=18.55$

$$x^* = \frac{\Sigma A\bar{x}}{\Sigma A} = 4.93$$

② Weighted Average → $\Sigma CxH / \Sigma H$

$$x^* = \frac{2.5\times0.3+5\times0.5+6.5\times1}{0.3+0.5+1} = 5.41$$

③ Centre of Sums → $\Sigma CxA / A$

$A_1 = \frac{1}{2}\times1\times0.3+0.3\times3+0.3\times1\times\frac{1}{2} = 1.2$

$A_2 = \frac{1}{2}\times1\times0.5+\frac{1}{2}\times1\times0.5+2\times0.5 = 1.5$

$A_3 = \frac{1}{2}\times1\times1+\frac{1}{2}\times1\times1+1\times1 = 2$

$$x^* = \frac{2.5\times1.2+5\times1.5+6.5\times2}{1.2+1.5+2}$$

$$x^* = 5$$

④ Mean of maxima → $x^* = \frac{6+7}{2} = 6.5$

⑤ Height → not applicable

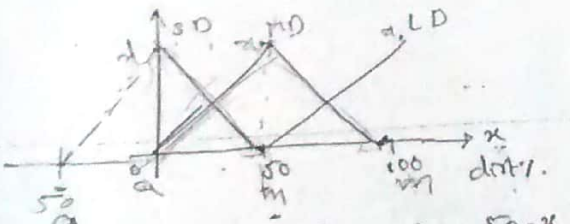
⑥ Centre of largest Area → $x^* = 6.5$

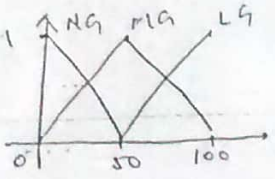
⑦ 1st or last of maxima → first = 6 , last = 7

★ Fuzzy Modeling →

Q) Design a fuzzy controller to determine wash-time of a domestic washing mlc. assume that i/p is dirt & grease on clothes use 3 descriptors for each i/p variable & 5 descriptors for o/p variable. Devise a set of rules for control action & defuzification. Clearly indicate that if clothes are soiled to a larger degree the wash time required will be more,

→ Dirt Descripter = < SD, MD, LD > } i/p
Grease Descripter = < NG, MG, LG >
Wash time = < VS, S, M, L, VL > — o/p





$\mu_{dirt}(x) = \mu_{SD}(x) = \frac{50-x}{50}$, $0 \le x < 50$

$\mu_{MD}(x) = \frac{x}{50}$, $0 \le x \le 50$

$= \frac{100-x}{50}$, $50 < x \le 100$

$\mu_{LD}(x) = \frac{x-50}{50}$, $50 \le x \le 100$

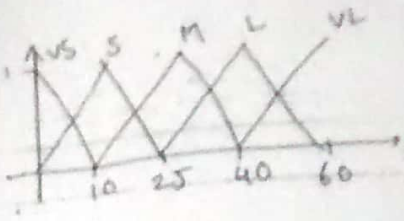
$\mu_{grease}(y) =$

$\mu_{NG}(y) = \frac{50-y}{50}$, $0 \le y < 50$

$\mu_{MG}(y) = \frac{y}{50}$, $0 \le y < 50$

$= \frac{100-y}{50}$, $50 < y \le 100$

$\mu_{LG}(y) = \frac{y-50}{50}$, $50 \le y \le 100$



$\mu_{VS} = \frac{10-z}{10}$, $0 \le z \le 10$

$\mu_S = \frac{z}{10}$ $0 \le z \le 10$

$= \frac{25-z}{15}$, $0 \le z \le 25$

$\mu_m = \frac{z-10}{15}$, $10 \le z \le 25$

$= \frac{40-z}{15}$, $25 \le z \le 40$

$\mu_L = \frac{z-25}{15}$, $25 \le z \le 40$

$= \frac{60-z}{20}$, $40 \le z \le 60$

$\mu_{VL} = \frac{z-40}{20}$, $40 \le z \le 60$

Rule base →		HG	MG	LG
	SD	VS	M	L
	MD	S	M	L
	LD	M	L	VL

① $x = 60\%$ $y = 70\%$.

$\mu_{MD} = \frac{100-x}{50} = \frac{100-60}{50} = \frac{4}{5}$ $\mu_{MG} = \frac{3}{5}$

$\mu_{LD} = \frac{x-50}{50} = \frac{60-50}{50} = \frac{1}{5}$ $\mu_{LG} = \frac{2}{5}$

	$\mu_{MG} = \frac{3}{5}$	$\mu_{LG} = \frac{2}{5}$
$\mu_{MD} = 4/5$	$\mu_{MZ} = 3/5$	$\mu_{LZ} = 2/5$
$\mu_{LD} = 1/5$	$\mu_{LZ} = 1/5$	$\mu_{MZ} = 1/5$

max = 3/5 falls in medium range.

$\therefore \mu_m(z) = \frac{z-10}{15} = \frac{3}{5}$ & $\frac{40-z}{15} = \frac{3}{5}$

$5z - 50 = 45$ & $z = 25$

$z = 19$

$z = \frac{19+25}{2} = \underline{\underline{22 \text{ min}}}$.

② $x = 80\%$, $y = 90\%$.

$\mu_{MD} = \frac{100-x}{50} = \frac{2}{5}$ $\mu_{MG} = \frac{100-y}{50} = \frac{1}{5}$

$\mu_{LD} = \frac{x-50}{50} = \frac{3}{5}$ $\mu_{LG} = \frac{4}{5}$

	1/5	4/5
4/5	1/5	4/5
3/5	1/5	3/5

max 3/5 falls in very large

$$\frac{z-40}{20} = \frac{3}{5}$$

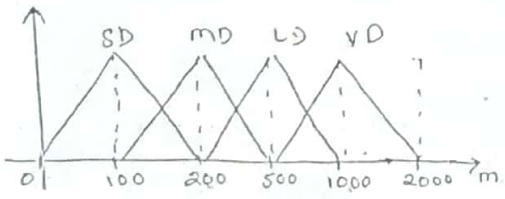
$$z = 52 \text{ mins}$$

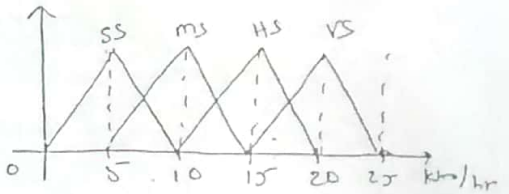
★ Q2) Design a fuzzy controller for a train approaching station. The i/p's are distance from station & speed of train & o/p is break power. Use traingular function & 4 descriptors for each variable.

→ Distance → {SD, MD, LD, VD}

Speed → {SS, MS, HS, VS}

Break power → {LP, MP, HP, VHP}





$$\mu_{SD} = \frac{x}{100} \;\&\; \frac{200-x}{100}$$

$$\mu_{MD} = \frac{x-100}{100} \;\&\; \frac{500-x}{300}$$

$$\mu_{LD} = \frac{x-200}{300} \;\&\; \frac{1000-x}{500}$$

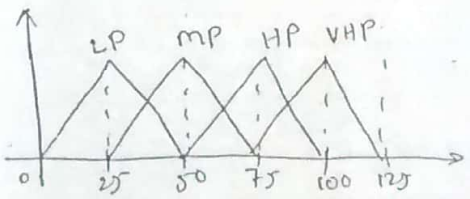
$$\mu_{VD} = \frac{x-500}{500} \;\&\; \frac{2000-x}{1000}$$

$$\mu_{SS} = \frac{4}{5} \;\&\; \frac{10-4}{5}$$

$$\mu_{MS} = \frac{4-5}{5} \;\&\; \frac{15-4}{5}$$

$$\mu_{HS} = \frac{4-10}{5} \;\&\; \frac{20-4}{5}$$

$$\mu_{VS} = \frac{4-15}{5} \;\&\; \frac{25-4}{5}$$



$$\mu_{LP} = \frac{z}{25} \;\&\; \frac{50-z}{25} \qquad \mu_{HP} = \frac{z-50}{25} \;\&\; \frac{100-z}{25}$$

$$\mu_{MP} = \frac{z-25}{25} \;\&\; \frac{100-z}{25} \qquad \mu_{VHP} = \frac{z-75}{25} \;\&\; \frac{125-z}{25}$$

	SS	MS	HS	VS
SD	mp	HP	VHP	VHP
MD	SP	mp	HP	HP
LD	SP	SP	mp	mp
VD	SP	SP	SP	mp

Let, $x = 100\,m$, $y = 24.6\ km/hr$

$$= \frac{24.6 \times 1000}{60 \times 60}$$

$$= 6.83\ m/s.$$

$$\mu_{SD} = \frac{200 - x}{100} = 1 \qquad \mu_{SS} = \frac{10 - y}{5} = 0.64$$

$$\mu_{mp} = \frac{x - 100}{100} = 0 \qquad \mu_{MS} = \frac{y - 5}{5} = 0.36.$$

	SS 0.64	MS 0.36
≤ SP	0.64	0.36
≥ mp	0	0

max is 0.64 belong to mp.

$$\frac{z - 25}{25} = 0.64 \quad \& \quad \frac{75 - z}{25} = 0.64$$

$$z_1 = 41 \quad \& \quad z_2 = 59$$

$$z = \frac{41 + 59}{2}$$

$$z = 50$$

1) Design a fuzzy controller to regulate the temperature of a domestic shower Assume that

a) Temperature is adjusted by single tap

b) Flow of water is constant

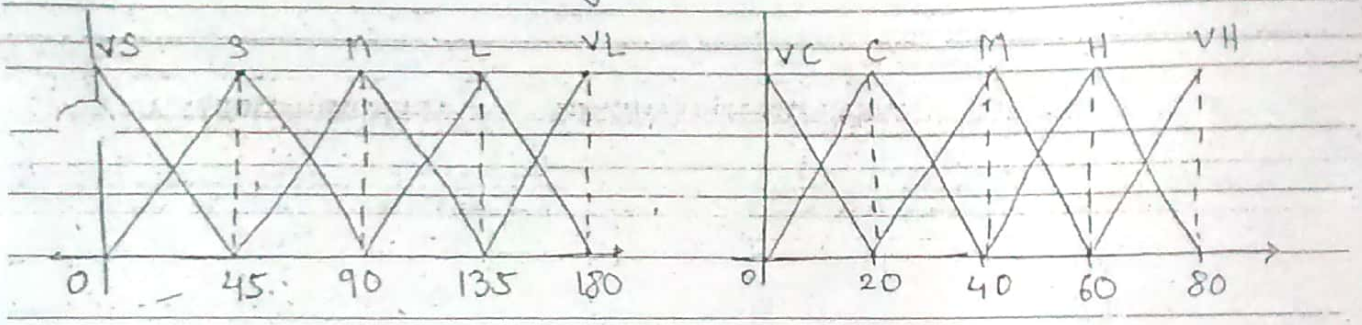
⟶

Descriptor → shower → { VS, S, M, L, VL }

(angle in degrees)

Temperature → { VC, C, M, H, VH }

(degree)



$\mu_{VS} = \frac{45-x}{45}$

$\mu_S = \frac{x}{45}$

$= \frac{90-x}{45}$

$\mu_m = \frac{x-45}{45}$

$= \frac{135-x}{45}$

$\mu_L = \frac{x-90}{45}$

$= \frac{180-x}{45}$

$\mu_{VL} = \frac{x-135}{45}$

$\mu_{VC} = \frac{20-y}{20}$

$\mu_C = \frac{y}{20}$

$= \frac{40-y}{20}$

$\mu_m = \frac{y-20}{20}$

$= \frac{60-y}{20}$

$\mu_H = \frac{y-40}{20}$

$= \frac{80-y}{20}$

$\mu_{VH} = \frac{y-60}{20}$

Rulebase →	VS	VC
	S	C
	M	M
	L	H
	VL	VH

Defuzzification → Shower tap position = 80°

$$\frac{90-x}{45} \quad \& \quad \frac{x-45}{45}$$

$$\therefore \frac{90-80}{45} = \frac{2}{9} \quad \therefore \frac{80-45}{45} = \frac{7}{9}$$

$\frac{7}{9}$ is maximum. (medium Range)

$$\therefore \frac{y-20}{20} = \frac{7}{9} \quad \& \quad \frac{60-y}{20} = \frac{7}{9}$$

$$y = 35.35 \quad \& \quad y = 44.4$$

$$y = 40$$

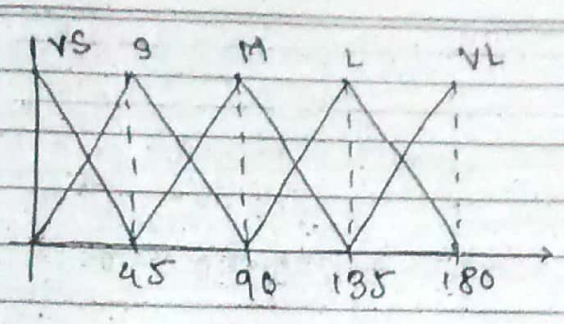
2) Design a fuzzy controller for maintaining the temperature of water in a tank at a fixed level of temperature. Input variables are ① cold water flow into tank. ② Steam flow into tank.

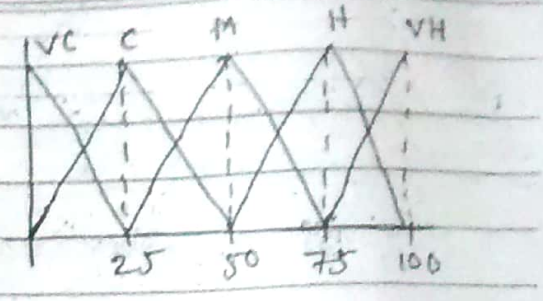
→

Descriptor → Heat value = { VS, S, M, L, VL }

Cold value = { VS, S, M, L, VL }

Water temp. = { VC, C, M, H, VH }





$\mu_{VS} = \frac{45-x}{45}$

$\mu_S = \frac{x}{45}$

$= \frac{90-x}{45}$

$\mu_m = \frac{x-45}{45}$

$= \frac{135-x}{45}$

$\mu_L = \frac{x-90}{45}$

$= \frac{180-x}{45}$

$\mu_{VL} = \frac{x-135}{45}$

$\mu_{VC} = \frac{25-z}{25}$

$\mu_C = \frac{z}{25}$

$= \frac{50-z}{25}$

$\mu_m = \frac{z-25}{25}$

$= \frac{75-z}{25}$

$\mu_L = \frac{z-50}{25}$

$= \frac{100-z}{25}$

$\mu_{VH} = \frac{z-75}{25}$

Rulebase →

cold → VS	S	M	L	VL
VS	M	C	C	VC	VC
S	H	M	C	VC	VC
M	H	H	M	C	VC
L	VH	VH	H	M	M
VL	(VH)	VH	H	H	M

↑ Heat

Defuzzification → Cold value = 45°

Hot value = 120°

Cold value

$$\mu_S = \frac{90-45}{45} = 1$$

$$\mu_m = \frac{45-45}{45} = 0$$

Hot value.

$$\mu_m = \frac{135-120}{45} = \frac{1}{3}$$

$$\mu_L = \frac{120-90}{45} = \frac{2}{3}$$

		(1) S	(0) m
(1/3)	m	1/3	0
(2/3)	L	2/3	0

2/3 is maximum (Very Hot value)

$$\mu_{VH} = \frac{z-75}{25} = \frac{2}{3}$$

$$z = 92°$$

Fuzzy Reasoning →

In fuzzy logic both the antecedents and consequents are allowed to be fuzzy propositions. There exist four modes of fuzzy approximate reasoning, which include:

1) Categorical reasoning:-

In this type of reasoning, the antecedents contain no fuzzy quantifiers and fuzzy probabilities. The antecedents are assumed to be in canonical form.

L, M, N, ... = fuzzy variables taking in the universes U, V, W

A, B, C = fuzzy predicates.

① The projection rule of inference is defined by $\frac{L, M; \text{ is } R}{L \text{ is } [R \downarrow L]}$

where $[R \downarrow L]$ denotes the projection of fuzzy relation R on L.

② The conjunction rule of inference, is given by

$$L \text{ is } A,\ L \text{ is } B \Rightarrow L \text{ is } A \cap B$$

$$(L, M) \text{ is } A,\ L \text{ is } B \Rightarrow (L, M) \text{ is } A \cap (B \times V)$$

$$(L, M) \text{ is } A,\ (M, N) \text{ is } B \Rightarrow (L, M, N) = (A \times W) \cap (U \times B)$$

③ The disjunction rule of inference is given by

$$L \text{ is } A \text{ OR } L \text{ is } B \Rightarrow L \text{ is } A \times B$$

$$L \text{ is } A,\ M \text{ is } B \Rightarrow (L, M) \text{ is } A \times B$$

④ The negative rule of inference is given by

$$\text{NOT } (L \text{ is } A) \Rightarrow L \text{ is } \overline{A}$$

⑤ The compositional rule of inference is given by

$$L \text{ is } A,\ (L, M) \text{ is } R \Rightarrow M \text{ is } A \cdot R$$

where A.R denotes the max-min composition of a fuzzy set A and a fuzzy relation R given by,

$$\mu_{A \cdot R}(v) = \max_u \min [\mu_A(u), \mu_R(u, v)]$$

⑥ The extension principle is defined as

$$L \text{ is } A \Rightarrow f(L) \text{ is } f(A)$$

where "f" is a mapping from u to v so that L is mapped into f(L); & based on the extension principle, MF of f(A) is defined as

$$\mu_{f(A)}(v) = \sup_{u} \mu_A(u), \quad u \in U,\ v \in V$$

6) Qualitative Reasoning :-

In qualitative reasoning the i/p-o/p relationship of a system is expressed as a collection of fuzzy IF THEN rules where the antecedents & consequents involve fuzzy linguistic variables. Let A and B be the fuzzy i/p variables and C be the fuzzy o/p variable. The relation among A, B & C may be expressed as

used in control system

If A is x_1 AND B is y_1, THEN C is z_1

⋮

If A is x_n AND B is y_n, THEN C is z_n

where x_i, y_i & z_i, $i = 1$ to n, are fuzzy subsets of their respective universe of discourse.

8) Syllogistic Reasoning :-

In this, antecedents with fuzzy quantifiers are related to inference rules. A fuzzy syllogism can be expressed as follows:

$x = k_1$ A's are B's :

$y = k_2$ C's are D's

$z = k_3$ E's are F's

In the above A, B, C, D, E & F are fuzzy predicates; k_1 and k_2 are the given fuzzy quantifiers and k_3 is the fuzzy quantifier which has to be decided. All the fuzzy predicates provide a collection of fuzzy syllogisms. These syllogisms create a set of inference rules, which combines evidence through conjunction and disjunction.

Given below are some important fuzzy syllogisms.

1. Produce syllogism :- C. $A \wedge B$, $F = C \wedge D$
2. Chaining syllogism :- $C = B$, $F = D$, $E = A$
3. Consequent conjunction syllogism :- $F = B \wedge D$, $A = C = E$
4. Consequent disjunction syllogism : $F = B \vee B$, $A = C = E$
5. Precondition conjunction syllogism : $E = A \wedge C$, $B = D = F$
6. Precondition disjunction syllogism : $E = A \vee C$, $B = D = F$

Dispositional Reasoning :-

In this kind of reasoning, the antecedents are dispositions that may contain, implicitly or explicitly, the fuzzy quantifier "usually". Usuality plays a major role in dispositional reasoning & it links together the dispositional & syllogistic modes of reasoning.

① Dispositional projection rule of inference :-

usually (L(L,M) is R) ⇒ usually (L is [R↓L]).

where [R↓L] is the projection of fuzzy relation R on L.

② Dispositional chaining hypersyllogism :-

R1 A's are B's, R2 B's are C's, usually (B C A)

usually (→ (Q1 (·) Q2) A's are C's)

③ Dispositional consequence conjunction syllogism :-

usually (A's are B's), usually (A's are C's) ⇒

(2 usually (-) 1 (A's are (B and C)'s)

④ Dispositional entailment rule of inference :-

usually (x is A), A ⊂ B ⇒ usually (x is B)

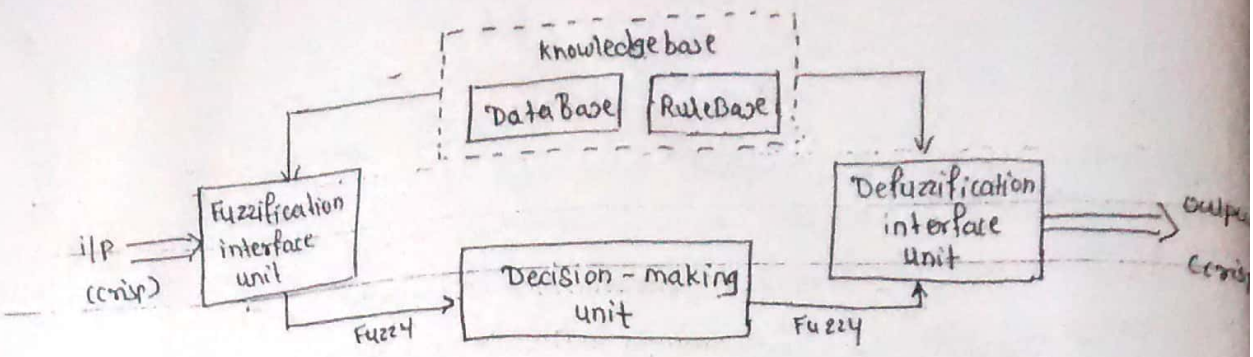
x is A, usually (A ⊂ B) ⇒ usually (x is B)

usually (x is A), usually (A ⊂ B) ⇒ usually^2 (x is B)

↳ less specific than usually.

Fuzzy Inference System (FIS) →

The key unit of a fuzzy logic system is FIS. The primary work of this system is decision making. FIS uses "IF ... THEN" rules along with connectors "OR" or "AND" for making necessary decision rules. The i/p to FIS may be fuzzy or crisp, but the o/p from FIS is always a fuzzy set. When FIS is used as a controller, it is necessary to have crisp o/p. Hence, there should be a defuzzification unit for converting fuzzy variables into crisp variables along FIS.



A FLS is constructed of five functional blocks. They are.

RuleBase → Contains numerous fuzzy IF-THEN rules.

DataBase → Defines the membership functions of fuzzy sets used in fuzzy rules.

Decision-making unit → performs operation on the rules.

Fuzzification interface unit → converts the crisp quantities into fuzzy quantities.

Defuzzification interface unit → converts the fuzzy quantities into crisp quantities.

Initially, in the fuzzification unit, the crisp i/p is converted into a fuzzy i/p. Various fuzzification methods are employed for this. After this process, rule base is formed. Database and rule base are collectively called the knowledge base. Finally, defuzzification process is carried out to produce crisp o/p. Mainly, the fuzzy rules are formed in the rule base & suitable decisions are made in the decision-making unit.

Methods of FIS :- There are two important types of FIS. Mamdani & Sugeno. The difference between the two methods lies in the consequent of fuzzy rules. Fuzzy sets are used as rule consequents in Mamdani FIS & linear functions of i/p variables are used as rule consequents in Sugeno's method. Mamdani's rule finds a greater acceptance in all universal approximators than Sugeno's model.

✱ Mamdani FIS →

Ebsahim Mamdani proposed this system in the year 1975 to control a steam engine and boiler combination by synthesizing a set of fuzzy rules obtained from people working on the system. In this case, the o/p MF's are expected to be fuzzy sets. After aggregation process, each o/p variable contains a fuzzy set, hence defuzzification is important at the o/p stage. The foll steps have to be followed to compute the o/p from this FIS

Step1:- Determine a set of fuzzy rules.

Step2:- Make the i/ps fuzzy using i/p MFs.

Step3:- Combine the fuzzified i/ps according to the fuzzy rules for establishing a rule strength.

Step4:- Determine the consequent of the rule by combining the rule strength & the o/p membership function.

Step5:- Combine all the consequents to get an o/p distribution.

Step6:- Finally, a defuzzified o/p distribution is obtained.

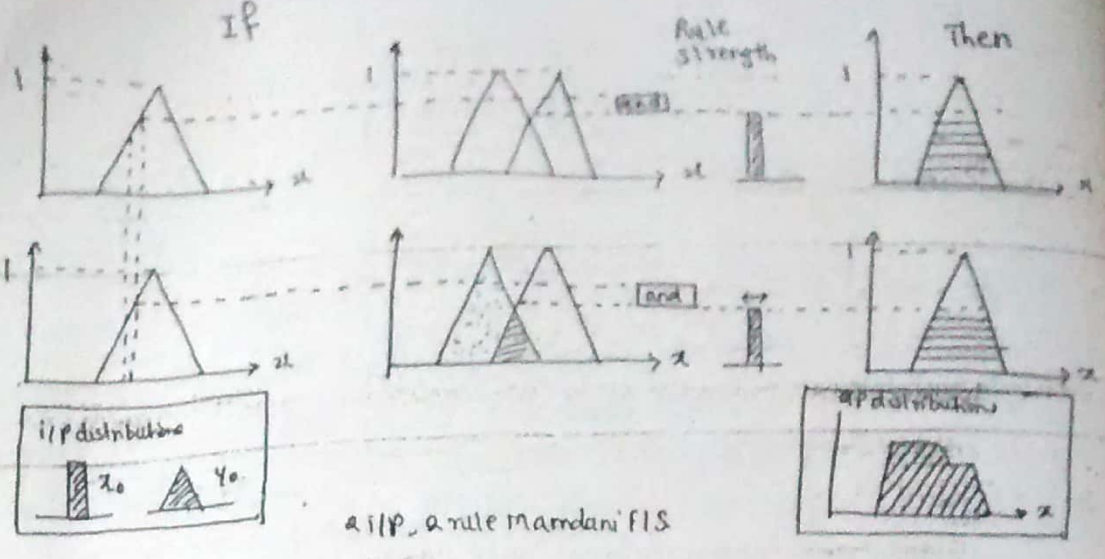
The fuzzy rules are formed using "IF-THEN" stmts and "AND/OR" connectives. The consequence of the rule can be obtained in two steps:

1) by computing the rule strength completely using the fuzzified i/ps from the fuzzy combination.

2) by clipping the o/p MF at the rule strength.

The o/p of all the fuzzy rules are combined to obtain one fuzzy o/p distribution. From FIS, it is desired to get only one crisp o/p that may be obtained from DF process. (center of mass, mean of maxima).

Consider a 2 i/p Mamdani FIS with 2 rules. The model fuzzifies the 2 i/ps by finding the intersection of 2 crisp i/p values with the i/p membership function. The minimum operation is used to compute the fuzzy i/p "and" for combining the 2 fuzzified i/ps to obtain a rule strength. The o/p MF is clipped at the rule strength. Finally the maximum operator is used to compute the fuzzy o/p "or" for combining the o/p of 2 rules.



a i/p, a rule mamdani FIS

★ Takagi - Sugeno Fuzzy Model →

Sugeno Fuzzy model was proposed by Takagi, Sugeno & Kang in 1985. The format of fuzzy rule of a sugeno model is given by.

IF x is A and y is B THEN $z = f(x,y)$

where AB are fuzzy sets in the antecedents and $z = f(x,y)$ is a crisp function in the consequent. Generally, $f(x,y)$ is a polynomial in the i/p variables x & y. If $f(x,y)$ is a first-order polynomial, we get first-order Sugeno fuzzy model. If f is a constant, we get zero-order model which is functionally equivalent to RBF nlw under certain minor constraints.

The main steps of the fuzzy inference process namely,

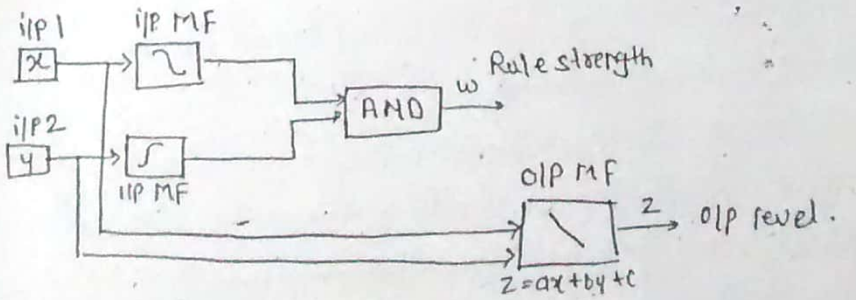
1) fuzzifying the i/ps.
2) applying the fuzzy operator.

The rule format is given by.

"If $3 = x$ and $5 = y$ then output is $z = ax + by + c$".

For a Sugeno model of zero order, o/p level z is constant.

The Sugeno system uses adaptive techniques for constructing fuzzy models. which are used to customize the MFs.



☆ Comparison between Mamdani & Sugeno Method :-

The main difference lies in the o/p MFs. The Sugeno o/p MFs are either linear or constant. The difference also lies in the consequents of their fuzzy rules & as a result their aggregation & defuzzification procedures differ suitably. A large no of fuzzy rules must be employed in Sugeno for approximating periodic or highly oscillatory functions. The configuration of Sugeno can be reduced & it becomes smaller that that of mamdani system. If non-traingular or non-trapezoidal fuzzy i/p sets are used. Sugeno controllers have more adjustable parameters in the rule consequent and the number of parameters grows exponentially with increase of the no of i/p variables. Formation of Mamdani FIS is more easier than Sugeno FIS.

Advantages of Mamdani → 1) It has widespread acceptance.
2) It is well-suitable for human i/p.
3) It is intuitive

Advantages of Sugeno → 1) Computationally efficient.
2) Compact & works well with linear techniques, opt & adaptive tech.
3) best suited for mathematical analysis.
4) guaranteed continuity of o/p surface.

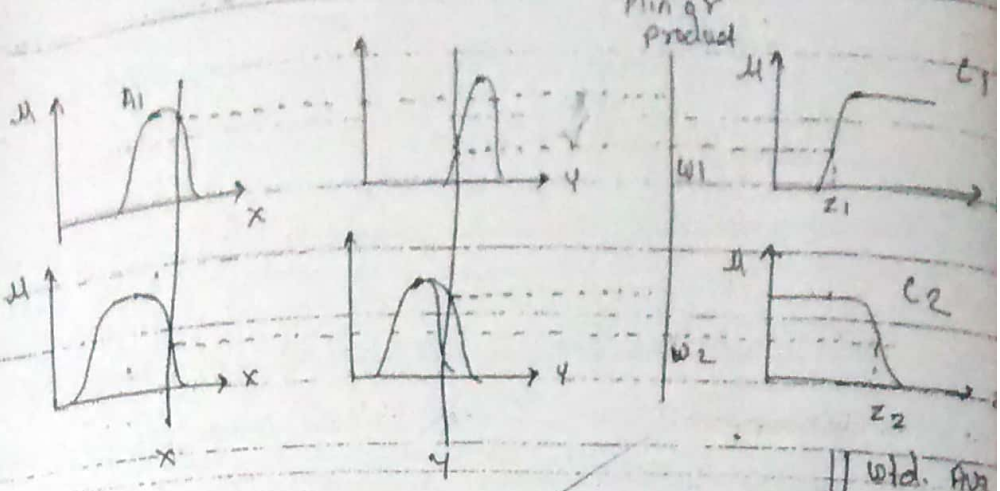
☆ Tsukamoto Fuzzy Models →

In this, the consequent of each fuzzy if-then rule is represented by a fuzzy set with a monotonical MF as a result, the inferred o/p of each rule is defined as a crisp value induced by the rule's firing strength. The overall o/p is taken as the weighted average of each rule's output.

Since each rule infers a crisp output, model aggregates each rule's o/p by the method of weighted average & thus avoids the time-consuming process of defuzzification.

An example of a single-input Tsukamoto fuzzy model is,

If x is small then y is C1
If x is medium then y is C2
If x is large then y is C3



$$z = \frac{w_1 z_1 + w_2 z_2}{w_1 + w_2}$$

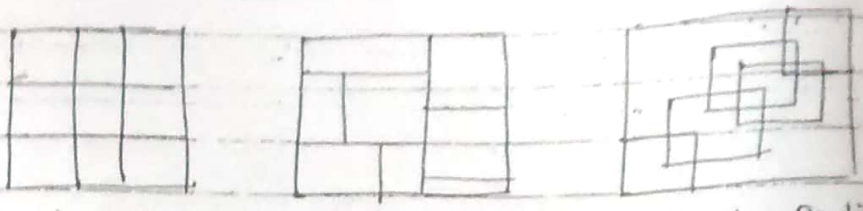
Since the reasoning mechanism of Tsukamoto fuzzy model does not follow strictly the compositional rule of inference, the o/p is always crisp even when the i/p are fuzzy.

* Input space Partitioning →

In FIS, the antecedent of a fuzzy rule defines a local fuzzy region, while the consequent describes the behavior within the region via various constituents. The consequent constituent can be a consequent MF (mandani & Tsukamoto), a constant value (zero-order sugeno), or a linear equation (first order sugeno). Different consequent constituents result in different FIS, but their antecedents are always the same. Therefore, the following methods of partitioning i/p spaces to form the antecedents of fuzzy rules is applicable to all three types of FIS.

1. Grid Partition :→ This method is often chosen in designing a fuzzy controller, which usually involves only several state variables as the i/ps to the controller. This partitioning strategy needs only a small number of MFs for each i/p. However it encounters problems when we have a moderately large no. of i/ps. i.e. fuzzy model with 10 i/p & 2 MFs on each i/p would result in $2^{10} = 1024$ fuzzy if-then-rules which is prohibitively large. This problem can be removed by other methods

curse of dimensionality



grid partition. tree partition scatter partition

2) Tree Partition: → In this each region can be uniquely specified along a corresponding decision tree. The tree partition relieves the problem of an exponential increase in the no of rules. However, more MFs for each i/p are needed to define these fuzzy regions, and these MFs do not usually bear clear linguistic meanings such as "small" "big" as seen in other words, orthogonality holds roughly in X x Y but not in either X or Y alone.

3) Scatter partition: → By covering a subset of the whole i/p space that characterizes a region of possible occurrence of the i/p vectors. the scatter partition can also limit the no. of rules to a reasonable amount. However, the scatter partition is usually dictated by desired i/p - o/p data pairs & thus, in general, orthogonality does not hold in x, y or X x Y. This makes it hard to estimate the overall mapping directly from the consequent of each rule's o/p.

Chapter 3: NEURAL NETWORKS

✱ Introduction to NN:—

Biological Nervous system → Brain → Information processing system similarly ANN → Infor^n proce. system

ANN resembles brain in two respects.

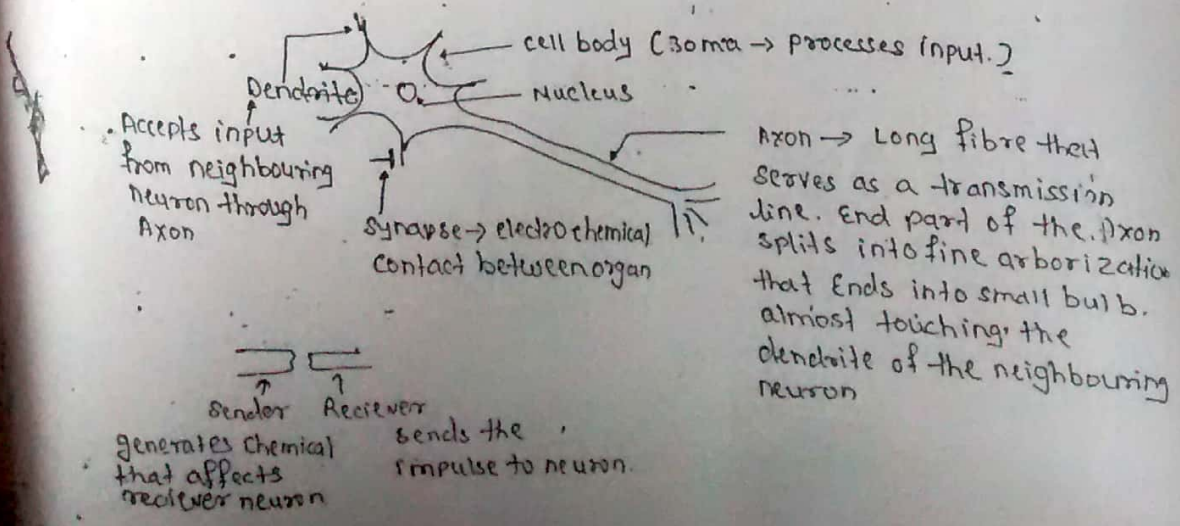
↳ 1) ANN acquires the knowledge from the Environment through a learning process.

2) Interneuron connection called as Synaptic wt's are used to store the acquired knowledge.

	Biological NN	Artificial NN
① Speed →	Slow	Fast
② Processing →	Parallel execution	One by one
③ Size & complexity of operation →	Less	more, difficult to implement complex operation
④ Fault tolerance →	Exist	Doesn't exist
⑤ Storage →	If new data added old is not erased	erased
⑥ Control mechanism →	Every neuron acts independently	CPU

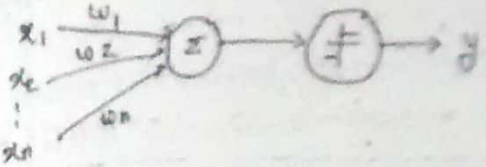
✱ Biological Neuron →

A Basic element of a BNN is a neuron. neuron is a elementary nerve cell with information processing ability. Each neuron has 3 major regions: cell body, Axon and Dendrite.



★ Basic ANN model / Mc'Culloch & Pitt's model of neuron

Mc'culloch & Pitt's proposed a computational model.



Incoming impulses can be excitatory or inhibitory

Excitatory (if they cause firing of neuron)

Inhibitory (if they hinder the firing of neuron)

$x_1 w_1 + x_2 w_2 + \cdots + x_n w_n$

$$y = \begin{cases} 1 & \text{if } \sum_{i=0}^{n} x_i w_i \geq T \\ 0 & \text{if } \sum_{i=0}^{n} x_i w_i < T \end{cases}$$

★ Simulation of NOT gate using M & P model.

I/P	O/P.
x	y
1	0
0	1

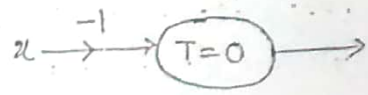
$w * x = w * 1 = w < T$

$w * x = w * 0 = 0 \geq T$

If $T = 0.5$ a weight is < 0.5 suppose we get 0.4 Then $0.4 < 0.5$ $y = 0$ (T)

$T = 0$, $w = -1$

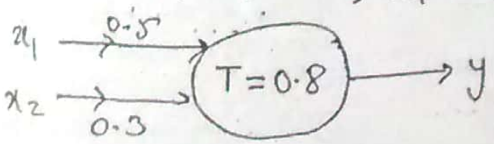
If T is < 0 $-0.1, -0.2, -1$



★ Simulation of AND gate :-

x_1	x_2	y	
0	0	0	$0 < T$
0	1	0	$w_2 < T$.
1	0	0	$w_1 < T$
0	0	0	$w_1 + w_2$
1	1	1	$w_1 + w_2 \geq T$

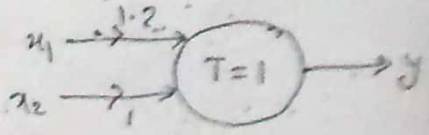
$T = 0.8$, $w_2 = 0.3$, $w_1 = 0.5$

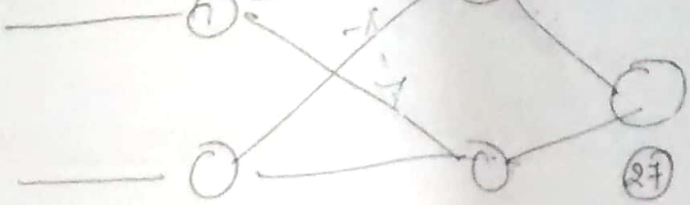


★ Simulation of OR gate :-

x_1	x_2	y	
0	0	0	$0 < T$
0	1	1	$w_2 \geq T$
1	0	1	$w_1 \geq T$
1	1	1	$w_1 + w_2 \geq T$

$T = 1$, $w_2 = 1$, $w_1 = 1.2$





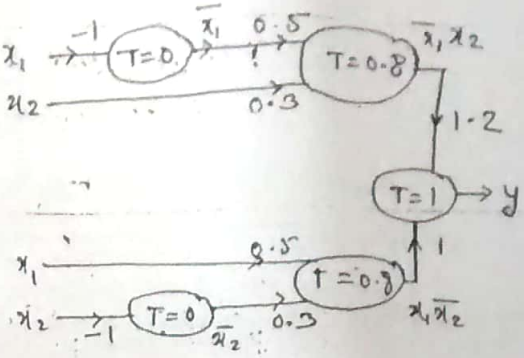
x

* Simulation of Ex-OR Gate :-

x_1	x_2	y	
0	0	0	$0 < T$
0	1	1	$w_2 \geq T$
1	0	1	$w_1 \geq T$
1	1	0	$w_1 + w_2 < T$

we are not able to select values for T, w_1, w_2

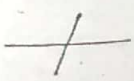
$\therefore$ Ex-OR = $\bar{x_1} x_2 + x_1 \bar{x_2}$



A Types of Activation Function / Transfer Functions →

Activation function $f(x)$ is used to give output of a neuron in terms of a local field x.

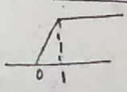
1) Linear function / Identity



$f(x) = x$

Matlab → purelin

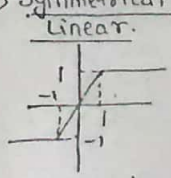
2) Saturating linear



$$f(x) = \begin{cases} 0 & x < 0 \\ x & 0 \leq x \leq 1 \\ 1 & x > 1 \end{cases}$$

Matlab → Satlin

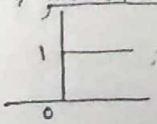
3) Symmetrical Saturach linear.



$$f(x) = \begin{cases} -1 & x < 0 \\ x & -1 \leq x \leq 1 \\ 1 & x > 1 \end{cases}$$

Matlab → satlins

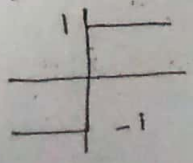
4) Hardlimit / Threshold / Unipolar binary



$$f(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases}$$

Matlab → Hardlim.

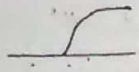
5) Symmetrical hardlimit / Bipolar binary.



$$f(x) = \begin{cases} -1 & x < 0 \\ 1 & x \geq 0 \end{cases}$$

Matlab → hardlims

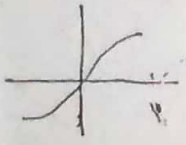
6) Unipolar continuous / Sigmoid.



$$f(x) = \frac{1}{1 + \exp(-\lambda x)}$$

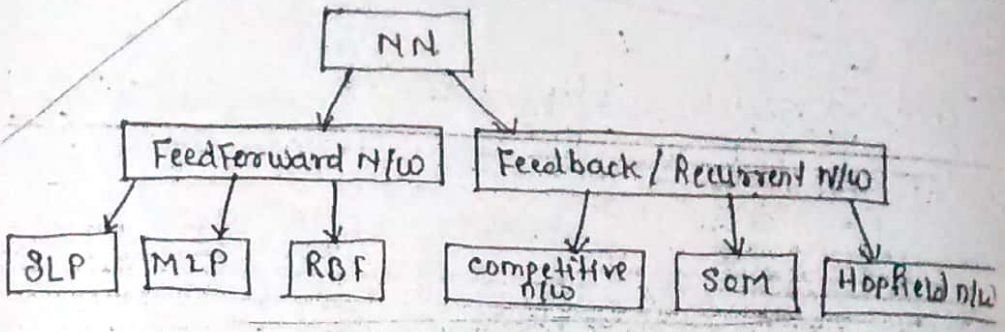
7) Bipolar continuous

$$f(x) = \left[\frac{2}{1 + \exp(-\lambda x)}\right] - 1$$



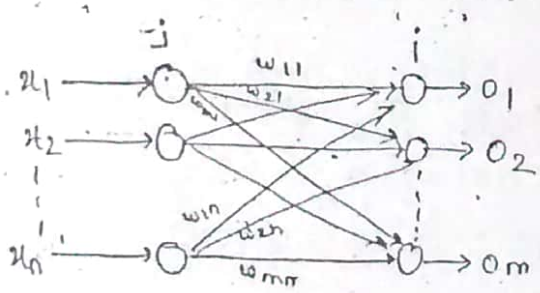
★ Network Architectures / Topologies / ANN models →

Neurons are interconnected to each other. Th arrangement of neurons is called as network archit



(1) Feedforward Network :-

In this type of network the neurons are co in a forward direction. connection is allowed fro layer j to layer i if and only if $j < i$



i/p vector = $[x_1\ x_2 \cdots]$

o/p vector = $[o_1\ o_2 \cdots]$

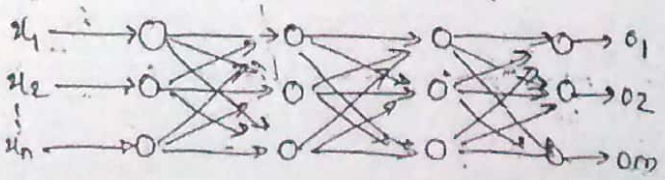
wt. matrix = $\begin{bmatrix} w_{11} & w_{12} \cdots \\ \vdots & \\ w_{m1} & w_{m2} \cdots \end{bmatrix}$

$$net_i = \sum_{j=1}^{n} w_{ij} x_j \quad \text{for } i = 1 \text{ to } m$$

$$o_i = f(net_i)$$

(1.1) SLP (Single layer Perceptron n/w) :- In this type of n/w ore only input & output layers are present (diagram is above)

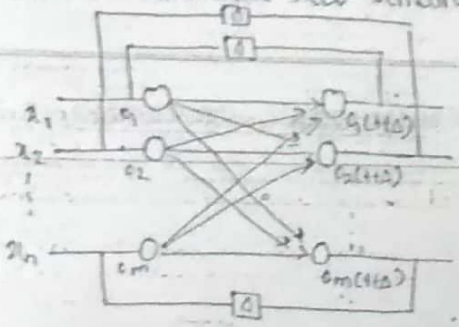
(1.2) MLP (Multi layer Perceptron n/w) :- In this type of n/w there may be one or more hidden layers are pre in between input & output layers.



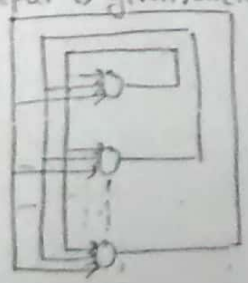
(1.3) RBF (Radial Basis Function) :- In this type of n/w, a single hidden layer is present.

(2) Feedback / Recurrent n/w →

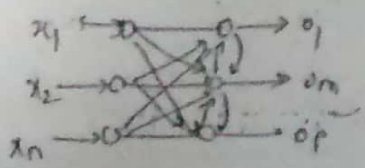
Feedback n/w :- Feedback n/w can be obtained from the feedforward n/w by connecting it's o/p to i/p. The inputs are applied at the initial instant, then the input is removed and the n/w remains autonomous thereafter ($t > 0$)



Recurrent n/w :- A single layer of neurons, where each neuron's output is given back to all neurons.

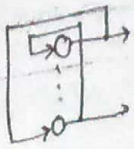


(2.1) Competitive n/w :- In this type of n/w the neurons of the output layer's compete between themselves to find out maximum o/p



(2.2) Som (self organizing map) :- The input neurons activate the closest output neuron.

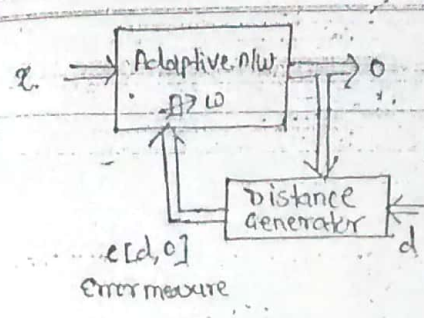
(2.3) Hopfield n/w :- Each neuron is connected to every other neuron but not back to itself.



★ Types of Learning →

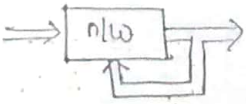
Learning of neural network means setting / updating the weights.

① Supervised learning :-



In this type of learning when input is applied supervisor provides a desired response. The difference between the actual response (o) and the desired response (d) is calculated called as Error measure which is used to correct the n/w parameters.

② Unsupervised learning :-



In this type of learning supervisor is not present due to this there is no ideal / guess of output. n/w modifies it's weights based on patterns of input and / or output.

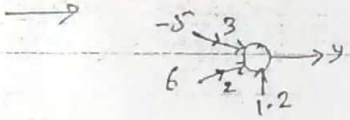
③ Hybrid learning :- In this type a combination of supervised and unsupervised learnings is used.

④ Competitive learning :- In this the output neurons compete between themselves. The neuron having maximum response is declared as winner neuron and the wt's of winner neurons are modified else remains unchanged.

★ A single layer NN is to be designed with 6 i/p & 2 o/p's. The o/ps are to be limited to 4 continuous over the range 0 to 1. Answer
1) How many neurons are required.?
2) What are dimensions of wt. matrix?
3) What kind of transfer function should be used?

→ 1) 8 neurons
2) $[\,]_{2\times 6}$ $\begin{bmatrix} w_{11} & w_{12} & \cdots & w_{16} \\ w_{21} & w_{21} & \cdots & w_{26} \end{bmatrix}$
3) unipolar continuous $\frac{1}{1+\exp(-\lambda net)}$

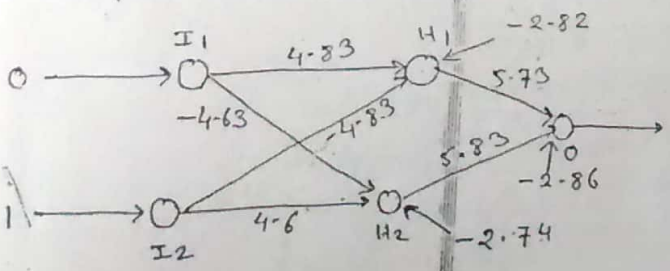
★ Given a 2 i/p neuron with following parameters, $b=1.2$, $w=[3,2]$ & $x=[-5,6]$ calculate neuron's o/p for foll. transfer function.

→ 

$net = -5\times 3 + 6\times 2 + 1.2 = -1.8$

① Hardlimit., o/p = 0 ② Sym. hard limit., o/p = -1
③ linear, o/p = -1.8 ④ Saturating linear, o/p = -1
⑤ unipolar continuous, o/p = 0.1418 ⑥ Bipolar cont, o/p = -0.71

★ Compute the output of the foll. n/w for unipolar continuous.



$H_1(net) = (4.83\times 0) + (-4.83\times 1) - 2.82 = -7.65$

$H_1(out) = 4.758\times 10^{-4}$

$H_2(net) = (-4.63\times 0) + (4.6\times 1) - 2.74 = 1.86$

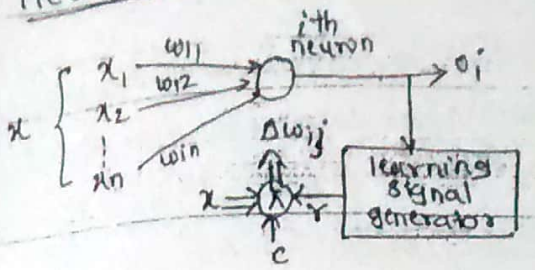
$H_2(out) = 0.865$

$O(net) = (4.758\times 10^{-4}\times 5.73) + (0.865\times 5.83) - 2.86 = 2.117$

$\boxed{O(out) = 0.899}$

★ Learning Rules :—

★ ① Hebbian Learning :- applicable for ~~binary~~ / continuous / discrete



learning signal, $r = o_i$

$\Delta w_{ij} = c \cdot r \cdot x_j$

$\Delta w_{ij} = c o_i x_j$

$w_{new} = w_{old} + \Delta w_{ij}$

Q) Initial wt. vector $w_1 = \begin{bmatrix} -1 \\ 0 \\ 0.5 \end{bmatrix}$ needs to be trained using three i/p vectors, $x_1 = \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix}$, $x_2 = \begin{bmatrix} 1 \\ 0.5 \\ -2 \\ -1.5 \end{bmatrix}$, $x_3 = \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix}$ & $c = 1$; $\lambda = 1$. Find final vector for bipolar binary & bipolar continuous neuron.

⟶ Bipolar Binary → $o = 1$ if $net \geq 0$ else -1

Step1 :- when x_1 is applied to n/w

$net_1 = w_1^t x_1 = 3$

$o_1 = 1$

$w_2 = w_1 + c o_1 x_1$

$w_2 = w_1 + x_1$

$w_2 = \begin{bmatrix} 2 \\ -3 \\ 1.5 \\ 0.5 \end{bmatrix}$

Step2 :- when x_2 is applied.

$net_2 = w_2^t x_2 = -0.25$

$o_2 = -1$

$w_3 = w_2 + c o_2 x_2$

$w_3 = w_2 - x_2$

$w_3 = \begin{bmatrix} 1 \\ -2.5 \\ 3.5 \\ 2 \end{bmatrix}$

Step3 :- when x_3 is applied.

$net_3 = w_3^t x_3 = -3$, $o_3 = -1$

$w_4 = w_3 + c o_3 x_3 = w_3 - x_3 = \begin{bmatrix} 1 \\ -3.5 \\ 4.5 \\ 0.5 \end{bmatrix}$ //

Bipolar Continuous → $o = \left[\frac{2}{1+\exp(-\lambda net)}\right] - 1$

Step1 :-

$net_1 = w_1^t x_1 = 3$

$o_1 = 0.905$

$w_2 = w_1 + c o_1 x_1$

$w_2 = \begin{bmatrix} 1.905 \\ -2.81 \\ 1.357 \\ 0.5 \end{bmatrix}$

Step2 :-

$net_2 = w_2^t x_2 = -0.154$

$o_2 = -0.077$

$w_3 = w_2 + c o_2 x_2$

$w_3 = \begin{bmatrix} 1.828 \\ -2.772 \\ 1.512 \\ 0.616 \end{bmatrix}$

Step3 :-

$net_3 = w_3^t x_3 = -3.36$

$o_3 = -0.932$

$w_4 = w_3 + c o_3 x_3$

$w_4 = \begin{bmatrix} 1.828 \\ -3.70 \\ 2.44 \\ -0.783 \end{bmatrix}$ //

$net_1 = 1 + 1 \cdot 1 + (-1) \times -1 + (+1 \cdot \times 0) + 0 \cdot 1$

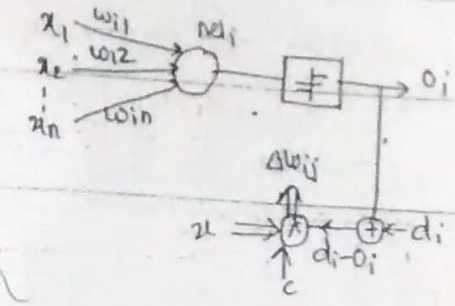
$= 1 + (-1) + 0$

$= 1 \cdot 0 + 0 \cdot 1$

$= 1$

(30)

(2) Perceptron learning Rule:- applicable only for binary neuron.



$r = d_i - o_i$

$\Delta w_{ij} = c[d_i - o_i] x_j$

w_i's are updated when $d_i \neq o_i$

$d_i = 1, o_i = -1$ ie. $\Delta w_{ij} = +2cx$

$d_i = -1, o_i = 1$ ie. $\Delta w_{ij} = -2cx$

$\therefore \boxed{\Delta w_{ij} = \pm 2cx}$

Q1) The three i/p vectors needs to be trained. Find final vector for Perceptron learning.

$x_1 = \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix}$ $x_2 = \begin{bmatrix} 0 \\ 1.5 \\ -0.5 \\ -1 \end{bmatrix}$ $x_3 = \begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1 \end{bmatrix}$ $w_1 = \begin{bmatrix} 0.1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}$, $d_1 = -1, d_2 = -1$, $d_3 = 1$, $c = 0.1$

→ **Step1:-** $net_1 = w_1^t x_1 = 2.5$

$o_1 = 1$

$d_1 \neq o_1$

$\therefore w_2 = w_1 + c(d_1 - o_1)x_1$

$= \begin{bmatrix} 0.8 \\ -0.6 \\ 0 \\ 0.7 \end{bmatrix}$

Step2:- $net_2 = w_2^t x_2 = -1.6$

$o_2 = -1$

$d_2 = o_2$

$\therefore$ updation is not required

$w_3 = w_2$

Step3:- $net_3 = w_3^t x_3 = -2.1$

$o_3 = -1$

$d_3 \neq o_3$

$\therefore w_4 = w_3 + c(d_3 - o_3)x_3$

$= \begin{bmatrix} 0.6 \\ -0.4 \\ 0.1 \\ 0.5 \end{bmatrix}$

Q2) Implement perceptron LR, $c = 1$, $w_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$, $x_1 = \begin{bmatrix} 2 \\ -1 \end{bmatrix}$ $d_1 = -1$, $x_2 = \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix}$ $d_2 = 1$. Repeat training sequence until 2 correct responses in a row are achieved.

→ **Step1:-** apply x_1

$net_1 = w_1^t x_1 = 1$

$o_1 = 1$

$w_2 = w_1 + c(d_1 - o_1)x_1$

$= \begin{bmatrix} -4 \\ -1 \\ 2 \end{bmatrix}$

Step2:- apply x_2

$net_2 = w_2^t x_2 = -1$

$o_2 = -1$

$w_3 = w_2 + c(d_2 - o_2)x_2$

$= \begin{bmatrix} -4 \\ -3 \\ 0 \end{bmatrix}$

Step3:- apply x_1

$net_3 = w_3^t x_1 = -11$

$o_3 = -1$

$o_3 = d_1$

$\therefore w_4 = w_3 = \begin{bmatrix} -4 \\ -3 \\ 0 \end{bmatrix}$

Step4:- apply x_2

$net_4 = w_4^t x_2 = 3$, $o_4 = 1$, $o_4 = d_2$ $\therefore w_5 = w_4 = \begin{bmatrix} -4 \\ -3 \\ 0 \end{bmatrix}$

Q3) A single neuron n/w using $f(net) = sgn(net)$ has been trained using the pairs of x_i, d_i as follows. Find wt. vector w_1.

$x_1 = \begin{bmatrix} 1 \\ -2 \\ 3 \\ -1 \end{bmatrix} d_1 = -1$, $x_2 = \begin{bmatrix} 0 \\ -1 \\ 2 \\ -1 \end{bmatrix} d_2 = 1$, $x_3 = \begin{bmatrix} -2 \\ 0 \\ -3 \\ -1 \end{bmatrix} d_3 = -1$, $w_4 = \begin{bmatrix} 3 \\ 2 \\ 6 \\ 1 \end{bmatrix}$

$c = 1$

$\longrightarrow w_4 = w_3 + \Delta w_3$

$\therefore w_3 = w_4 - \Delta w_3$

As we know; $\Delta w_3 = \pm 2 c x_3$. As $d_3 = -1$ we consider -ve sign

$\Delta w_3 = -2 c x_3 = \begin{bmatrix} 4 \\ 0 \\ 6 \\ 2 \end{bmatrix}$ & $w_3 = \begin{bmatrix} -1 \\ 2 \\ 0 \\ -1 \end{bmatrix}$

$w_3 = w_2 + \Delta w_2$

$\therefore w_2 = w_3 - \Delta w_2$

$\Delta w_2 = 2 c x_2 = \begin{bmatrix} 0 \\ -2 \\ 4 \\ -2 \end{bmatrix} \quad \therefore w_2 = \begin{bmatrix} -1 \\ 4 \\ -4 \\ 1 \end{bmatrix}$

$w_2 = w_1 + \Delta w_1$

$\therefore w_1 = w_2 - \Delta w_1$

$\Delta w_1 = -2 c x_1 = \begin{bmatrix} -2 \\ 4 \\ -6 \\ 2 \end{bmatrix} \quad \therefore w_1 = \begin{bmatrix} 1 \\ 8 \\ 2 \\ -1 \end{bmatrix}$ //

Q4) wts & biases are initialized to small random values as.

$w_1 = \begin{bmatrix} 0.5 \\ -1 \\ -0.5 \end{bmatrix} b_1 = 0.5$, $x_1 = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} d_1 = 0$, $x_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} d_2 = 1$

find final wt vector

$\longrightarrow$ Step1 :- $net_1 = w_1^t x_1 + b_1 = 2.5$

$o_1 = 1$

$d_1 \neq o_1$

$w_2 = w_1 + c(d_1 - o_1) x_1$

$= \begin{bmatrix} -0.5 \\ 0 \\ 0.5 \end{bmatrix}$

$b_2 = b_1 + (d_1 - o_1)$

$= -0.5$

Step2 :- $net_2 = w_2^t x_2 = -2.5$

$o_2 = 0$

$d_2 \neq o_2$

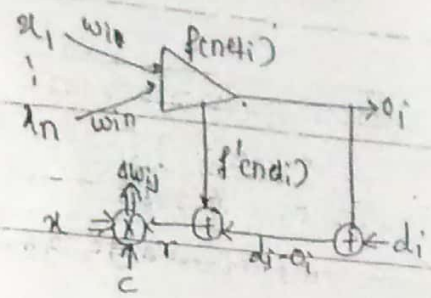
$w_3 = w_2 + c(d_2 - o_2) x_2$

$= \begin{bmatrix} 0.5 \\ 1 \\ -0.5 \end{bmatrix}$ //

$b_3 = b_2 + (d_2 - o_2)$

$= 0.5$ //

③ Delta learning Rule :→ applicable only for continuous neurons.



$r = [d_i - o_i] f'(net_i)$

$\Delta w_{ij} = c[d_i - o_i] f'(net_i) x_j$

$f'(net_i) = \frac{1}{2}(1-o^2)$ ← Bipolar

$f'(net_i) = o(1-o)$ ← unipolar)

* Proove. 1) $f'(net) = o(1-o)$ 2) $f'(net) = \frac{1}{2}(1-o^2)$

1) $f'(net) = o(1-o)$ ← unipolar.

$f(net) = \frac{1}{1+e^{-x}}$

$f'(net) = \frac{(1+e^{-x})\frac{d}{dx}(1) - 1\cdot\frac{d}{dx}(1+e^{-x})}{(1+e^{-x})^2} = \frac{e^{-x}}{(1+e^{-x})^2} = \frac{1+e^{-x}-1}{(1+e^{-x})^2}$

$= \frac{1+e^{-x}}{(1+e^{-x})^2} - \frac{1}{(1+e^{-x})^2} = \frac{1}{(1+e^{-x})} - \frac{1}{(1+e^{-x})^2} = f(net) - f(net)^2$

$= o - o^2$

$= o(1-o)$ //

RHS.

2) $f'(net) = \frac{1}{2}(1-o^2)$ ← Bipolar.

LHS → $f(net) = \frac{2}{1+e^{-x}} - 1$

$f'(net) = \frac{2e^{-x}}{(1+e^{-x})^2}$ — (I)

RHS → $\frac{1}{2}(1-o^2) = \frac{1}{2}\left[1 - \left[\frac{2}{1+e^{-x}} - 1\right]^2\right]$

$= \frac{1}{2}\left[1 - \left[\frac{4}{(1+e^{-x})^2} - \frac{4}{1+e^{-x}} + 1\right]\right]$

$= \frac{1}{2}\left[\frac{-4}{(1+e^{-x})^2} + \frac{4}{1+e^{-x}}\right]$

$= \frac{-2}{(1+e^{-x})^2} + \frac{2}{(1+e^{-x})} = \frac{-2+2(1+e^{-x})}{(1+e^{-x})^2}$

$= \frac{2e^{-x}}{(1+e^{-x})^2}$ — (II)

From I & II

LHS = RHS.

Q) $x_1 = \begin{bmatrix} -\frac{1}{2} \\ 0 \\ -1 \end{bmatrix}$ $x_2 = \begin{bmatrix} 0 \\ 1.5 \\ -0.5 \\ -1 \end{bmatrix}$ $\begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1 \end{bmatrix}$ $\begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}$

$d_1 = -1$, $d_2 = -1$, $d_3 = 1$ $\lambda = 1$, $c =$ use delta LR.

→

Step1:- $net_1 = w_1^t x_1 = 2.5$

$o_1 = 0.848$

$o_1' = \frac{1}{2}(1 - o_1^2) = 0.140$

$w_2 = w_1 + c[d_1 - o_1] o_1' x_1$

$= \begin{bmatrix} 0.974 \\ -0.948 \\ 0 \\ 0.526 \end{bmatrix}$

:- $net_2 = w$ 1.948 Step3:-

$o_2 = -0$ $net_3 = w_3^t x_3$

$o_2' = 0$ $= -2.4$

$= w_2 + c[d$ $]o_2' x_2$ $o_3 = -0.84$

$o_3' = 0.14$

$= \begin{bmatrix} 0.97 \\ -0.9 \\ 0.0 \\ 0.5 \end{bmatrix}$ $w_4 = \begin{bmatrix} 0.947 \\ -0.929 \\ 0.016 \\ 0.505 \end{bmatrix}$

④ Widrow-Hoff learning Rule Independent of activation fun

$r = d_i -$ x

$\Delta w_{ij} = c[$ $- w_i^t x]$;

Q) for above values of x_1, 3, d_1, d_2, c use WH LR.

→ Step1:- Step2: Step3:-

$net_1 = 2.5$ ne -1.3 $net_3 = -1.7325$

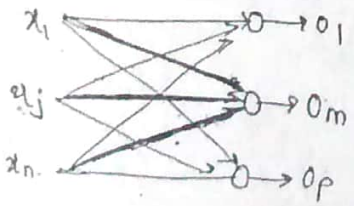
$w_2 = w_1 + c[d_1 - net_1] x_1$ $w_3 =$ $c[d_2 - net_2$ $w_4 = w_3 + c[d_3 - net_3] x_3$

$= \begin{bmatrix} 0.65 \\ -0.3 \\ 0 \\ 0.85 \end{bmatrix}$ $= \begin{bmatrix} 6.5 \\ -2.55 \\ -0.15 \\ 2.82 \end{bmatrix}$ $= \begin{bmatrix} 0.38 \\ 0.018 \\ 0.12 \\ 0.55 \end{bmatrix}$

⑤ Correlation learning Rule: $r = d_i$

$\Delta w_{ij} =$ x_j

⑥ Winner-take-all learning e:-



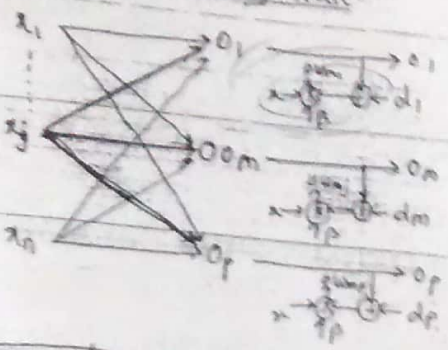
elative lea w_j used in unsupervised
g.
euron has mum response due to
; then m eclared as winner
$w_m = [w$ $w_{m2} \cdots w_{mn}]^t$
fan-in ve of mth neuron are updated
t of wt's rains unchanged
$\Delta w_{mj} =$ $(x_j - w_{mj})$

$w_m^t x = \max_{i=1 \text{ to } p} (w_i^t x)$

⑦ Outstar Learning Rule:-

Supervised, fan-out vectors are updated

$\Delta w_{mj} = \beta(d_m - w_{mj})$



Learning Rule	Wt adjustment	Initial wt.	Learning	neuron characteristic	neuron/layer
1) Hebbian	$\Delta w_{ij} = c o_i x_j \quad \forall i$	0	unsupervised	Any	neuron
2) Perceptron	$\Delta w_{ij} = c[d_i - o_i] x_j$	Any	Supervised	Binary	neuron
3) Delta	$\Delta w_{ij} = c[d_i - o_i] f'(net_i) x_j$	Any	Supervised	Continuous	neuron
4) Widrow-Hoff	$\Delta w_{ij} = c[d_i - w_i x] x_j$	Any	Supervised	Any	neuron
5) correlation	$\Delta w_{ij} = c d_i x_j$	0	Supervised	Any	neuron
6) Winner-take-all	$\Delta w_{mj} = \alpha(x_j - w_{mj})$	Random normalized	unsupervised	continuous	layer
7) outstar	$\Delta w_{mj} = \beta(d_m - w_{mj})$	0	Supervised	continuous	layer

★ Design a perceptron nlo for AND function for bipolar i/ps & targets.

→

x_1	x_2	y
-1	-1	-1
-1	1	-1
1	-1	-1
1	1	1

$w_1 = 0.5, \quad w_2 = 0.8 \quad b = 0$

$net_1 = (-1 \times 0.5) + (-1 \times 0.8) + 0 = -1.3$

$o_1 = -1 \quad \therefore o_1 = y_1$ ∴ no updation

$net_2 = (-1 \times 0.5) + (1 \times 0.8) + 0 = 0.3$

$o_2 = 1 \quad \therefore o_2 \neq y_2$

$w_1 = 0.5 + \underset{(t=0)(x)}{(-1-1)(-1)} = 2.5$

$w_2 = 0.8 + (-1-1)(1) = -1.2$

$b = 0 + (-1-1) = -2$

$(t - o)(x)$
$(b - o)(1)$

$net_3 = (1 \times 2.5) + (-1 \times -1.2) - 2 = 1.7$

$o_3 = 1 \quad \therefore o_3 \neq y_3$

$w_1 = 2.5 + (-1-1) \times 1 = 0.5$

$w_2 = -1.2 + (-1-1) \times (-1) = 0.8$

$b = -2 + (-1-1) = -4$

$net_4 = 0.5 + 0.8 - 4 = -2.7$

$o_4 = -1 \quad \therefore o_4 \neq y_4$

w_1	w_2	b
0.5	0.8	0
2.5	-1.2	2
0.5	0.8	-4
+2.5	2.8	-2

$w_1 = 0.5 + (1+1)(1) = 2.5$

$w_2 = 0.8 + (1+1) \times 1 = 2.8$

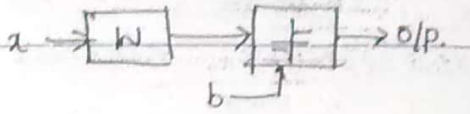
$b = -4 + (1+1) = -2$

$\boxed{w_1 = 2.5, \; w_2 = 2.8, \; b = -2}$

✱ SUPERVISED LEARNING NEURAL NETWORKS →

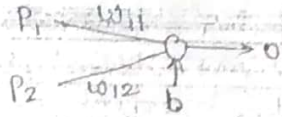
✱ 1) Singler Layer Perceptron :→

Perceptron Architecture :-



o/p = 1 or 0. Thus each neuron in the nlw divides the i/p space into two regions. This is useful to determine the boundary between these regions.

✱ Examples →

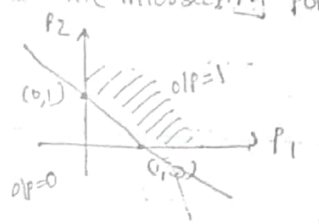


$o = hardlim(w_{11}P_1 + w_{12}P_2 + b)$ i/p vector for which the net i/p is zero determines the decision boundary.

$\therefore w_{11}P_1 + w_{12}P_2 + b = 0$

let's take $w_{11} = w_{12} = 1$ & $b = -1$

$\therefore P_1 + P_2 - 1 = 0$ To draw the line we need to find the intersecting points of P_1 & P_2 axes.



$P_1 + P_2 - 1 = 0$

$P_1 = 0, P_2 = 1 \quad (0,1)$.

$P_2 = 0; P_1 = 1 \quad (1,0)$

Now to find which decision region belongs to o/p = 1

Pick one pt. (2, 0)

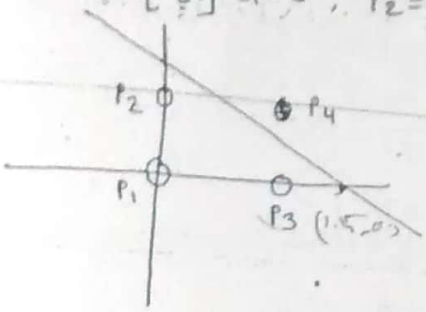
hardlim $(wp + b)$ = 0

$\begin{bmatrix} 2 \\ 0 \end{bmatrix} [1\ 1] - 1 = 2 - 1 = 1$

Decision boundary is always orthogonal to the weight vector and it always points towards the region wher neuron o/p is 1

* Implement perceptron n/w for AND function -

$P_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} t_1 = 0$, $P_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} t_2 = 0$, $P_3 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} t_3 = 0$, $P_4 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} t_4 = 1$



let's take,
$w = [2 \ 2]$
To find the value of bias pick one pt. on decision boundary [1.5 0]

$w^t p + b = 0$

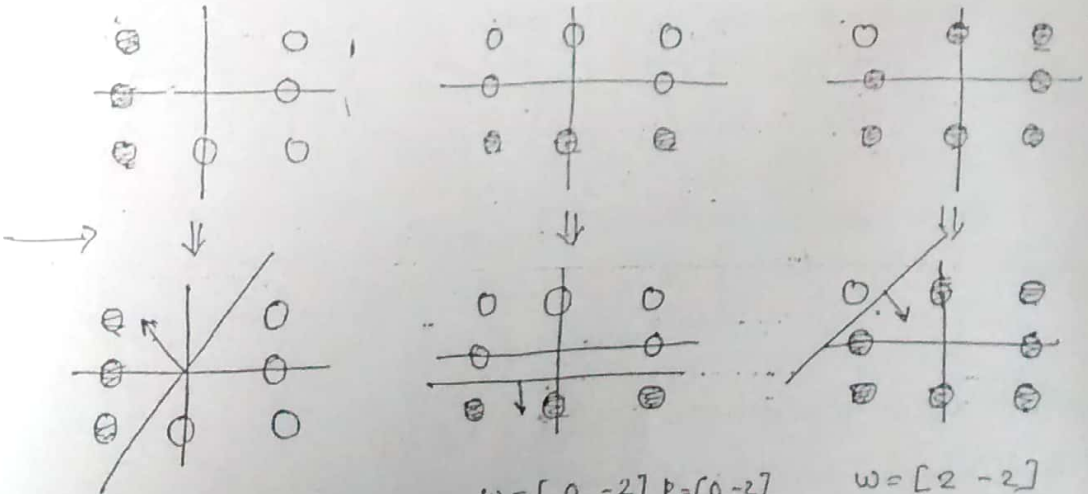
$[1.5 \ 0] \begin{bmatrix} 2 \\ 2 \end{bmatrix} + b = 0$

$b = -3$

Test $(w^t p + b)$ hardlim. take P_1 to test.

hardlim $\left([2 \ 2] [0 \ 0] - 3 \right)$ = hardlim$(-3) = 0$ // correct.

* Solve the three classification problems by drawing decision boundary. find wt & bias values that result in single neuron perceptron with chosen decision boundaries.



$w = [-2 \ 1]$

$w^t p + b = 0$

$[-2 \ 1] [0 \ 0] + b = 0$

$b = 0$.

Test hardlim $([-2 \ 1] \begin{bmatrix} 2 \\ -2 \end{bmatrix} + 0)$

hardlim(-4)

$= 0$

correct

$w = [0 \ -2]$ $P = [0 \ -2]$

$b = -2$

$w = [2 \ -2]$

$P = [-2 \ 1]$

$b = 6$

$\star \left\{ p_1 = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} t_1 = 0 \right\} \quad \left\{ p_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} t_2 = 1 \right\} \quad w_1 = \begin{bmatrix} 0.5 \\ -1 \\ -0.5 \end{bmatrix} \quad b_1 = 0.5$

Repeat iteration unless & until both i/p vectors are correctly classified.

$\longrightarrow$

$o_1 = \text{hardlim}(w_1^T p_1 + b_1) = hl\left(\begin{bmatrix} 0.5 & -1 & -0.5 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} + 0.5 \right)$

$= hl(2.5)$

$o_1 = 1$

apply P1 $\rightarrow$

$e = t_1 - o_1 = 0 - 1 = -1.$

$w_2, w_{new} = w_{old} + e \cdot p_1 = \begin{bmatrix} -0.5 \\ 0 \\ 0.5 \end{bmatrix}$ $\quad e = t_1 - o_1 =$

apply P2 $\rightarrow$ $b_2, b_{new} = b_{old} + e = -0.5$

$o_2 = \text{hardlim}(w_2^T p_2 + b) = hl\left(\begin{bmatrix} -0.5 & 0 & 0.5 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} - 0.5 \right)$

$= 0$

$e = t_2 - o_2 = 1 - 0 = 1$

$w_3, w_{new} = w_{old} + e \cdot p_2 = \begin{bmatrix} 0.5 \\ 1 \\ -0.5 \end{bmatrix}$

$b_3, b_{new} = b_{old} + e = -0.5 + 1 = 0.5$

$o_3 = \text{hardlim}(w_3^T p_1 + b) = 1, \quad e = -1$

apply P1 $\rightarrow$ $w_4 = \begin{bmatrix} -0.5 \\ 2 \\ 0.5 \end{bmatrix}$

$b_4 = -0.5$

$o_4 = \text{hardlim}(w_4^T p_2 + b) = 1, \quad e = 0$

apply P2 $\rightarrow$ $w_5 = \begin{bmatrix} -0.5 \\ 2 \\ 0.5 \end{bmatrix} \quad b_5 = -0.5$

apply P1 $\rightarrow$ $o_5 = \text{hardlim}(w_5^T p_1 + b) = 0, \quad e = 0$

$w_6 = \begin{bmatrix} -0.5 \\ 2 \\ 0.5 \end{bmatrix} \quad b_6 = -0.5$

Both i/p vectors are correctly classified.

★ Solve the following classification problem with perceptron LR. apply each i/p vector in order for as many repetitions as it takes to ensure that the problem is solved. Draw a graph of the problem only after you have found a solution.

$\left\{P_1=\begin{bmatrix}2\\2\end{bmatrix} t_1=0\right\}\left\{P_2=\begin{bmatrix}1\\-2\end{bmatrix} t_2=1\right\}\left\{P_3=\begin{bmatrix}-2\\2\end{bmatrix} t_3=0\right\}\left\{P_4=\begin{bmatrix}-1\\1\end{bmatrix} t_4=1\right\}$

use initial wt's & bias as, $w(0)=[0\ 0]$, $b(0)=0$

→ P_1

$o_1 = hl(w(0)P_1 + b(0)) = hl([0\ 0]\begin{bmatrix}2\\2\end{bmatrix}+0) = hl(0) = 1$

$o_1 \neq t_1 \quad \therefore e = t_1 - o_1 = 0-1 = -1$

$w(1) = w(0) + e P_1^t = [0\ 0] + (-1)[2\ 2] = [-2\ 2]$

$b(1) = b(0) + e = 0 + (-1) = -1$

P_2

$o_2 = hl(w(1)P_2 + b(1)) = hl([-2\ 2]\begin{bmatrix}1\\-2\end{bmatrix}+(-1)) = 1$

$o_2 = t_2$

$\therefore w(2) = w(1)$ & $b(2) = b(1)$

P_3 $o_3 = 0$, $o_3 = t_3$

$\therefore w(3) = w(2)$ & $b(3) = b(2)$

P_4 $o_4 = 0$, $o_4 \neq t_4 \quad \therefore e = t_4 - o_4 = 1-0 = 1$

$w(4) = w(3) + e P_4^t = [-2\ -2] + (1)[-1\ 1] = [-3\ -1]$

$b(4) = b(3) + e = -1 + 1 = 0$

P_1 $o_5 = t_1 \quad \therefore w(5) = w(4)$, $b(5) = b(4)$

P_2 $o_6 = 0$, $t_2 = 1$, $o_6 \neq t_2$, $e = 1$.

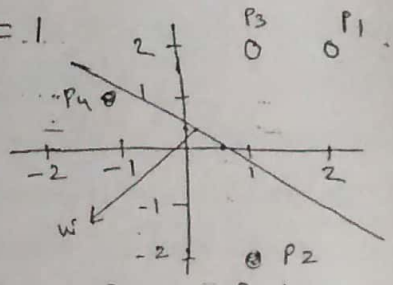
$w(6) = [-2\ -3]$ $b(6) = 1$

P_3 $o_7 = 0$, $t_3 = 0$, $e = 0$

P_4 $o_8 = 0$, $t_4 = 1$, $e = 0$

P_1 $o_9 = 0$, $t_1 = 0$, $e = 0$

P_2 $o_{10} = 0$, $t_2 = 1$, $e = 0$



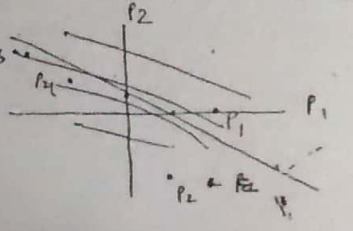
algo. has converged. final soln is $w[-2\ -3]$ $b = 1$.

$w_{11}P_1 + w_{12}P_2 + b = 0$

$-2P_1 - 3P_2 + 1 = 0$

$P_1 = 0$, $P_2 = 1/3 = 0.3$ $(0, 0.3)$

$P_2 = 0$, $P_1 = 0.5$ $(0.5, 0)$



★ Train perceptron nlw to solve s proble sing Perceptron Le

$\{P_1=\begin{bmatrix}1\\1\end{bmatrix} t_1=\begin{bmatrix}0\\0\end{bmatrix}\}$ $\{P_2=\begin{bmatrix}1\\2\end{bmatrix} t$ $0\end{bmatrix}\}$ $\{P$ $\begin{bmatrix}2\\-1\end{bmatrix} t_3=\begin{bmatrix}0\\1\end{bmatrix}\}$

$\{P_4=\begin{bmatrix}2\\0\end{bmatrix} t_4=\begin{bmatrix}0\\1\end{bmatrix}\}$ $\{P_5=\begin{bmatrix}-1\\2\end{bmatrix} t$ $1\end{bmatrix}\}$ $\{P$ $\begin{bmatrix}-2\\1\end{bmatrix} t_1=\begin{bmatrix}1\\0\end{bmatrix}\}$

~~$\{P_7=\begin{bmatrix}-1\\-1\end{bmatrix} t_7=\begin{bmatrix}1\\1\end{bmatrix}\}$ $\{P_8=\begin{bmatrix}-2\\-2\end{bmatrix}$ $3=\begin{bmatrix}1\\1\end{bmatrix}\}$~~

Initial w's & biases. $w(0)=\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$ $:)=\begin{bmatrix}1\\1\end{bmatrix}$

$\longrightarrow$

~~$o_1 = hl(w(0)P_1+b(0)) = hl($ $\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}$ $\end{bmatrix}) = \begin{bmatrix}1\end{bmatrix}$~~

$e=t_1-o_1=\begin{bmatrix}0\\0\end{bmatrix}-\begin{bmatrix}1\\1\end{bmatrix}=$ $\begin{bmatrix}1\\1\end{bmatrix}$

$w(1)=w(0)+eP_1^t=\begin{bmatrix}1 & 1\\0 & \end{bmatrix}+\begin{bmatrix}-1\\-1\end{bmatrix}[$ $1]=\begin{bmatrix}0 & -1\\-1 & 0\end{bmatrix}$

$b(1)=b(0)+e=\begin{bmatrix}1\\1\end{bmatrix}+$ $]=\begin{bmatrix}0\\0\end{bmatrix}$

$o_2=hl[w(1)P_2+b(1)=1$ $\begin{bmatrix}0 & -1\\-1 & 0\end{bmatrix}$ $\begin{bmatrix}1\\1\end{bmatrix}+\begin{bmatrix}0\\0\end{bmatrix})=\begin{bmatrix}0\\0\end{bmatrix}$

$e=t_2-o_2=\begin{bmatrix}0\\0\end{bmatrix}-$ $]=\begin{bmatrix}0\\0\end{bmatrix}$

$w(2)=w(1)+eP_2^t=\begin{bmatrix}$ $-1\\0\end{bmatrix}$

$b(2)=b(1)+e=\begin{bmatrix}0\\0\end{bmatrix}$

$o_3=hl(w(2)P_3+b(2))=$ $0\end{bmatrix}$

$e=\begin{bmatrix}-1\\1\end{bmatrix}$

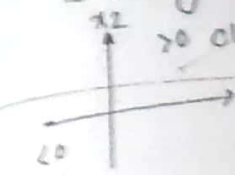
$w(3)=\begin{bmatrix}-2 & 0\\1 & -1\end{bmatrix}$ $b($ $=\begin{bmatrix}-1\\1\end{bmatrix}$

$w(3)=w(4)=w(5)=w($ $w(7)=$ $)$

$b(3)=b(4)=b(5)=b($ $b(7)=b$

* Perceptron Convergence Theorem:-

A classifier for two linearly seperable classes of patterns is always trainable in a finite number of training steps



$w^* = w^{k_0} = w^{k_0+1} = w^{k_0+2} \ldots$

k_0 → Solution w. vector, k_0 → training step number

Starting at which no more misclassification occurs.

$w(n+1) = w(n)$ if $w^t(n)\,x(n) > 0$ & $x(n) \in$ class1

$w(n+1) = w(n)$ if $w^t(n)\,x(n) < 0$ & $x(n) \in$ class2

otherwise w's are updated as,

$w(n+1) = w(n) - \eta\, x(n)$ if $w^t(n)\,x(n) > 0$ & $x(n) \in$ class2

$w(n+1) = w(n) + \eta\, x(n)$ if $w^t(n)\,x(n) < 0$ & $x(n) \in$ class1

Summary:→

Variables & Parameters:- $x(n) = [+1, x_1(n), x_2(n), \ldots x_m(n)]$

$w(n) = [b(n), w_1(n), w_2(n), \ldots w_m(n)]$

$d(n), y(n), \eta$.

① Initialisation → Set $w(0) = 0$ for $n = 1$ to m perform foll. steps.

② Activation → Apply $x(n)$ & $d(n)$

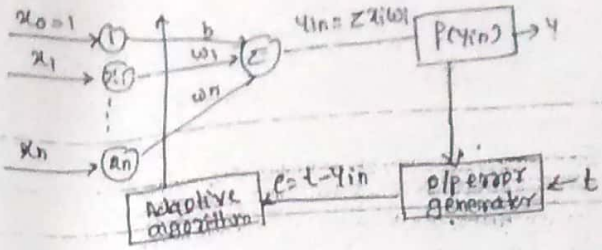
③ Calculation → $y(n) = \text{Sgn}[w^t(n)\,x(n)]$

④ Adjustment → $w(n+1) = w(n) + \eta\,[d(n) - y(n)]\,x(n)$

⑤ Continuation → Increment n by 1 & go back to step 2.

② Adaline → (Adaptive Linear Neuron)

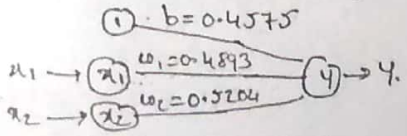
The units with linear activation function are called linear units. A n/w with a single linear unit is called an Adaline. In this i/p-o/p relationship is linear. Adaline uses bipolar activation for it's i/p signals & it's target o/p. Adaline is a n/w which has only one o/p unit. Adaline is a single unit neuron, which recieves i/p from several units & also from one unit called bias. i/p's are either of two values (+1/-1) & w's have signs (+ve/-ve) initially random w's are assigned.



Training:-

1) w's & bias are set to some random values but not zero & set learning rate
2) perform steps 3-7 when stopping condition is false.
3) perform steps 4-6 for each bipolar training pair s:t
4) set activations for i/p units $i=1$ to n , $x_i = s_i$
5) calculate the net i/p to the o/p unit. $y_{in} = b + \sum_{i=1}^{n} x_i w_i$
6) update the w's & bias for $i=1$ to n , $w_i(new) = w_i(old) + \alpha(t-y_{in})x_i$
$b(new) = b(old) + \alpha(t-y_{in})$
7) If the highest wt. change that occurred during training is smaller than a specified tolerance then stop the training process, else continue.

* Q1) use Adaline nlw to train AND NOT function with bipolar i/p & targets perform 2 epochs of training.



x_1	x_2	1	t
1	1	1	-1
1	-1	1	1
-1	1	1	+1
-1	-1	1	-1

let's take
$w_1 = w_2 = b = 0$
$\alpha = 0.2$

for 1st i/p. $y_{in} = b + x_1 w_1 + x_2 w_2 = 0.6$
$(t - y_{in}) = (-1 - 0.6) = -1.6$
$w_i(new) = w_i(old) + \alpha(t-y_{in})x_i$

$w_1(new) = 0 + 0.2(-1.6)x_1 = -0.12$
$w_2(new) = 0 + 0.2(-1.6)x_1 = -0.12$
$b(new) = b(old) + \alpha(t-y_{in})$
$= 0 + 0.2(-1.6) = -0.12$

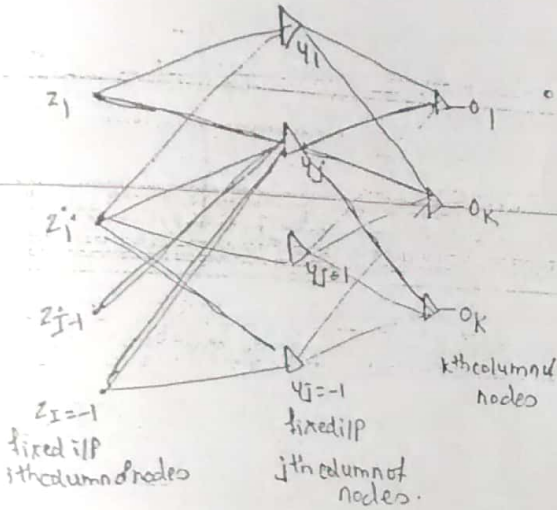
EPOCH-1	x_1	x_2	1	t	y_{in}	w_1	w_2	w_3	$E=(t-y_{in})^2$
	1	1	1	-1	0.6	-0.12	-0.12	-0.12	2.56
	1	-1	1	1	-0.12	0.10	-0.34	0.10	1.25
	-1	1	1	+1	-0.34	0.24	-0.48	-0.03	0.43
	-1	-1	1	-1	0.21	0.48	-0.25	-0.27	1.47

EPOCH-2	x_1	x_2	1	t	y_{in}	w_1	w_2	w_3	$E=(t-y_{in})^2$
	1	1	1	-1	-0.02	0.28	-0.43	-0.46	0.95
	1	-1	1	+1	0.25	0.43	-0.58	-0.31	0.57
	-1	1	1	+1	-1.33	0.37	-0.81	-0.25	0.106
	-1	-1	1	-1	-0.11	0.55	-0.38	0.43	0.8

ms. total error
1) 5.71
2) 2.43

(3) Multilayer Perceptron nlw:-

★ **Generalised Delta LR:-** We will derive a general expression for the wt. increment Δv_{ji} for any layer of neurons that is not an output layer.



-ve gradient-descent formula for hidden layer

$$\Delta v_{ji} = -\eta \frac{\partial \varepsilon}{\partial v_{ji}}$$

$$\frac{\partial \varepsilon}{\partial v_{ji}} = \frac{\partial \varepsilon}{\partial net_j} \cdot \frac{\partial net_j}{\partial v_{ji}}$$

$\frac{\partial \varepsilon}{\partial net_j}$ → δ_{y_j} error signal for hidden layer

$\frac{\partial net_j}{\partial v_{ji}}$ → z_i i/p.

$$\therefore \Delta v_{ji} = \eta \delta_{y_j} z_i$$

$$\delta_{y_j} = -\frac{\partial \varepsilon}{\partial net_j} \longrightarrow \delta_{y_j} = -\frac{\partial \varepsilon}{\partial y_j} \cdot \frac{\partial y_j}{\partial net_j}$$

$\frac{\partial y_j}{\partial net_j} = f_j'(net_j)$

$$\frac{\partial \varepsilon}{\partial y_j} = \frac{\partial}{\partial y_j}\left(\frac{1}{2}\sum_{K=1}^{K}\{d_K - f(net_K(y))\}^2\right)$$

$$= -\sum_{K=1}^{K}(d_K - o_K)\frac{\partial}{\partial y_j} f(net_K(y))$$

$$= -\sum_{K=1}^{K}(d_K - o_K) f'(net_K)\cdot\frac{\partial (net_K)}{\partial y_j}$$

($d_K - o_K$ → δ_{o_K}; $\frac{\partial (net_K)}{\partial y_j}$ → w_{Kj})

$$= -\sum_{K=1}^{K}\delta_{o_K} w_{Kj}$$

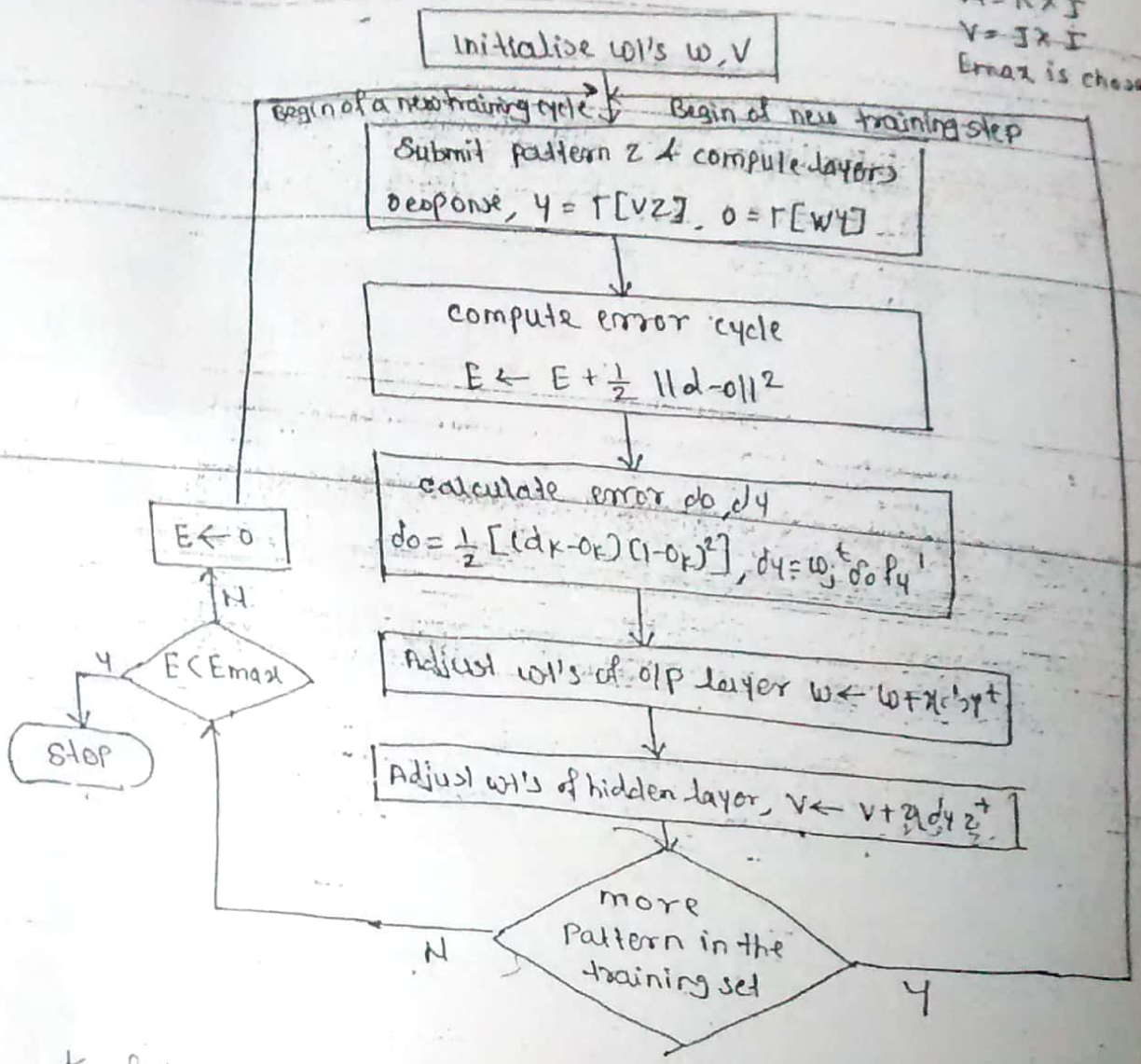
$$\delta_{y_j} = \left(\sum_{K=1}^{K}\delta_{o_K} w_{Kj}\right) f_j'(net_j)$$

$$\boxed{\Delta v_{ji} = \eta f_j'(net_j) z_i \sum_{K=1}^{K}\delta_{o_K} w_{Kj}}$$

→ $\frac{1}{2}(1-o^2)$ or $o(1-o)$

★ Error back Propagation Training —

$W = k \times J$
$V = J \times I$
Emax is chosen.



★ If the n/w do not get trained what are the possible reasons for the same? Explain what you will do to get it trained. / Learning factor

→

① Initial wt → Selected in such a way that n/w requires more no of training cycle to train the n/w. Adjusting wt. will train n/w properly.

② Fixing up the desired o/p → Based on i/p pattern if we are not deciding desired o/p properly n/w will not get trained.

③ Nonseperable patterns → If features of 2 classes of pattern are similar, n/w will not be trained properly. Features should be differentiable. Euclidian distance between 2 cluster must be properly specified. so that identifying the patterns will

be easy. Increase no. of hidden layers, performance may get improved.

④ learning Constant → η should be varied.

η ↑ faster training, 10^{-3} to 10 successful

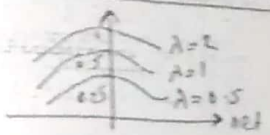
η has to be increased for some training cycle & then it has to be reduced.

⑤ Momentum → $\Delta w(t) = -\eta \nabla E(t) + \alpha \Delta w(t-1)$

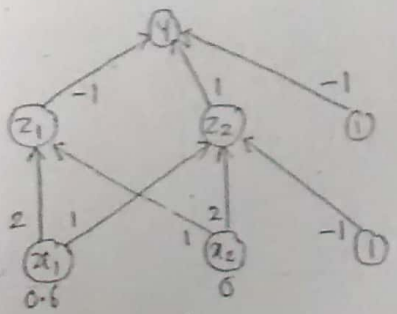
momentum gives push to training — momentum — previous wt. adjustment

α → 0.1 to 0.8

⑥ Steepness of activation function → It is advisable to keep λ at a standard value of 1, change & observe of activation function strongly affect speed of nw training.



5) Compute the o/p of foll. nw using unipolar cont. fn & EBPTA algo. $w = [-1\ \ 1]$ $w_0 = -1$, $V = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$ $v_0 = [0\ \ -1]$, $t = 0.9$, $\eta = 0.3$

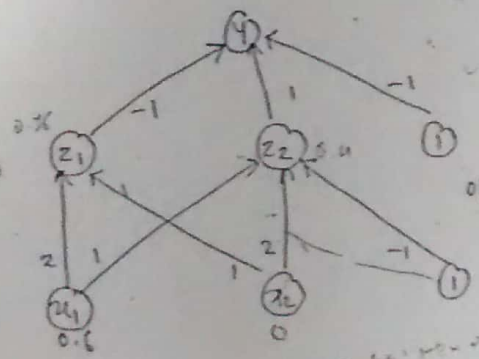


→ Feedforward stage:-

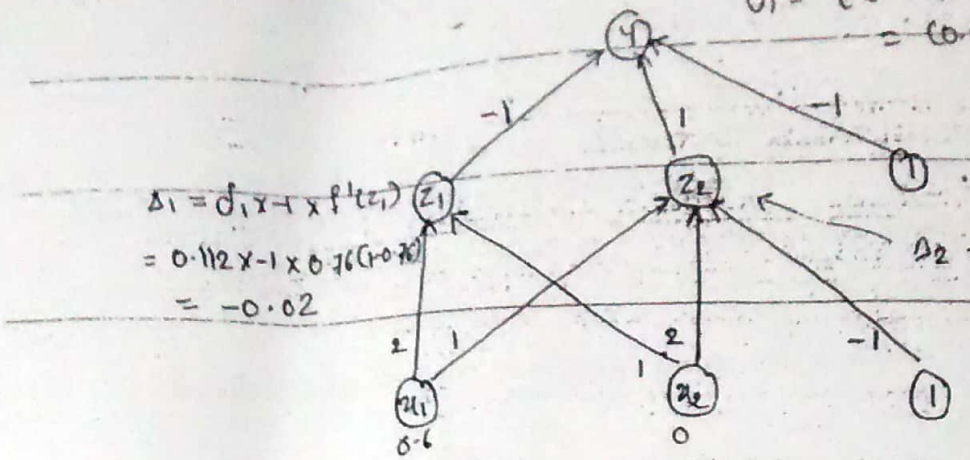
$o/p(y) = f(0.76(-1) + 0.4(1) - 1)$
$= 0.2$

$o/p(z_1) = f(0.6(2) + 1(0)) = 0.76$

$o/p(z_2) = f(-1 + 0.6(1) + 2(0)) = 0.4$



Back propagation of error -



$\delta_1 = (t-y) f'(y)$

$= (0.9-0.2)\, 0.2(1-0.2)$

$= 0.112$

$\Delta_1 = \delta_1 \times 1 \times f'(z_1)$

$= 0.112 \times -1 \times 0.76(1-0.76)$

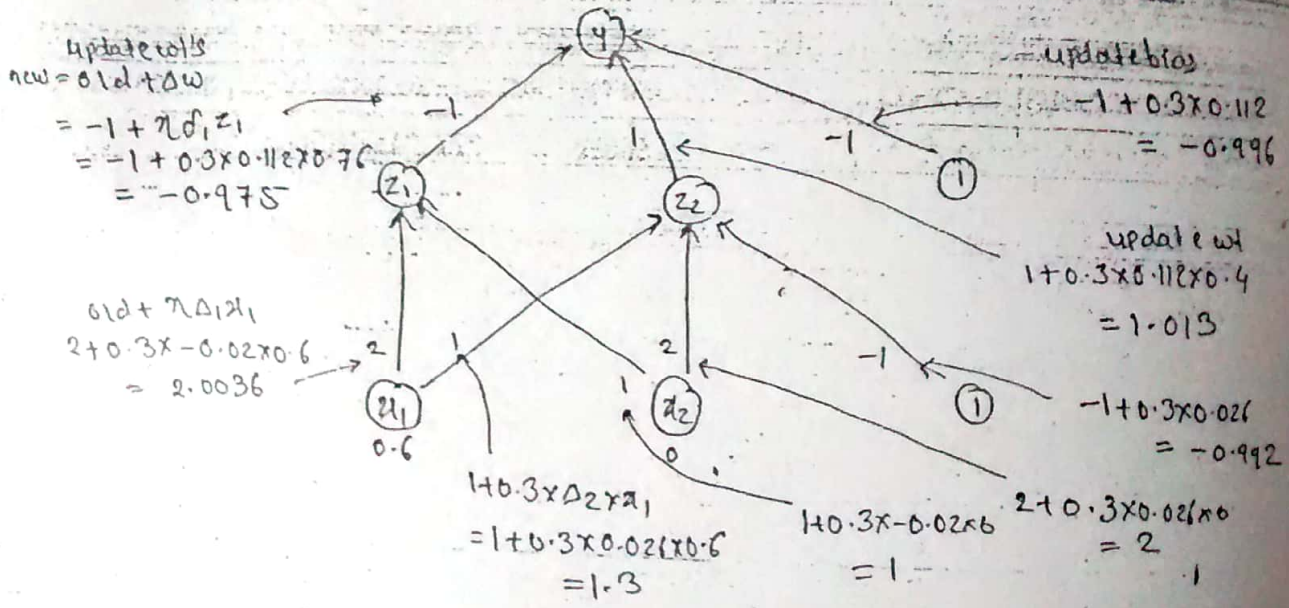
$= -0.02$

$\Delta_2 = \delta_1 \times 1 \times f'(z_2)$

$= 0.112 \times 1 \times 0.4(1-0.4)$

$= 0.026$

Updation of weights -



Update wt's

$new = old + \Delta w$

$= -1 + \eta \delta_1 z_1$

$= -1 + 0.3 \times 0.112 \times 0.76$

$= -0.975$

Update bias

$-1 + 0.3 \times 0.112$

$= -0.996$

$old + \eta \Delta_1 x_1$

$2 + 0.3 \times -0.02 \times 0.6$

$= 2.0036$

update wt

$1 + 0.3 \times 0.112 \times 0.4$

$= 1.013$

$-1 + 0.3 \times 0.026$

$= -0.992$

$1 + 0.3 \times \Delta_2 \times x_1$

$= 1 + 0.3 \times 0.026 \times 0.6$

$= 1.3$

$1 + 0.3 \times -0.02 \times 0$

$= 1$

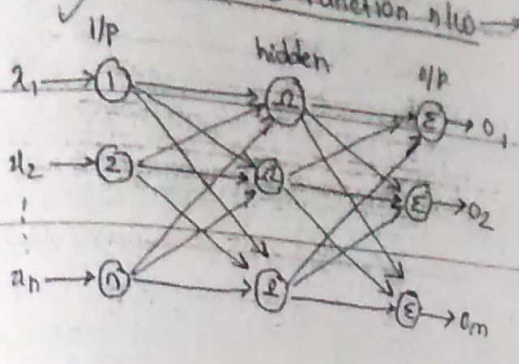
$2 + 0.3 \times 0.026 \times 0$

$= 2$

Updated wt's. $\rightarrow V = \begin{bmatrix} 2.0036 & 1.3 \\ 1 & 2 \end{bmatrix}$ $W = [-0.975 \quad 1.013]$

Updated bias $\rightarrow$ $V_0 = [0 \quad -0.992]$ $W_0 = [-0.996]$

(4) Radial Basis Function n/w →



1) 3 layers: i/p, hidden, o/p
2) i/p → n neurons, o/p → m neurons
3) i/p — hidden → hypothetical conn
4) hidden - o/p → weighted conn
5) Gaussian activation function is used to compute o/p. $f(x) = e^{-x^2}$
6) hidden layer applies nonlinear & o/p layer applies linear transformation

*

RBF n/w	MLP n/w
1) Single hidden layer	1) May have one or more hidden layer
2) Hidden layer serves different purpose from those in the o/p layer	2) computation nodes in a hidden or o/p layer share a common neuron model.
3) hidden layer - nonlinear & o/p layer - linear.	3) hidden & o/p - nonlinear
4) Activation fun of each hidden unit computes Euclidian distance between i/p vector & center of unit.	4) computes inner product of i/p vector & synaptic wt. vector of that unit.
5) Construct local approximation to nonlinear i/p-o/p mapping.	5) construct global approximation to nonlinear i/p-o/p mapping.

* Theorem on Seperability of Patterns (Cover's theorem) :→

When a RBF n/w is used to perform a complex pattern classification task, problem is basically solved by transforming it into a high dimensional space in a nonlinear manner.

$\mathcal{H} \rightarrow x_1, x_2, \ldots x_n \rightarrow$ each of which is assigned to one of two classes $\mathcal{H}_1$ and $\mathcal{H}_2$.

These points are said to be seperable w.r.t. family of surfaces for each pattern $x \in \mathcal{H}$

$\varphi(x) = [\varphi_1(x), \varphi_2(x), \ldots \varphi_{m_1}(x)]^T$

$\varphi(x)$ → vector in an m_0 dimensional i/p space
→ vector $\varphi(x)$ maps points in m_0 dimensional space into corresponding points in a new space of dimension m_1.

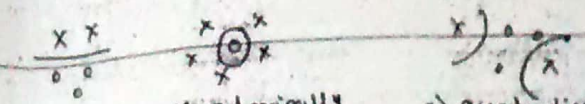
$\phi_i(x)$ → hidden function.

A dichotomy $\{x_1, x_2\}$ is said to be ϕ seperable if there exists an m_1 dimensional vector w such that

$w^T \phi(x) > 0$, for $x \in x_1$

$w^T \phi(x) < 0$, for $x \in x_2$

hyperplane → $w^T \phi(x) = 0$



a) linearly seperable b) Spherically seperable c) quadratically seperable.

★ RBF Learning Strategies :- Learning strategies are followed in the design of RBF n/w.

① Fixed centers selected at Random :→

Assume fixed radial basis function defining the activation function of hidden units. The locations of centers may be chosen randomly from training data set.

RBF centered at t_i is, defined as,

$$G(\|x - t_i\|^2) = \exp\left(-\frac{m_1}{d^2_{max}} \|x - t_i\|^2\right)$$

$G = \{\phi_{i=1}, \phi_{i=2}, b\}$ m_1 → no. of centers, d_{max} → max. distance between chosen centers.

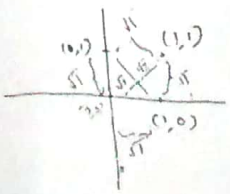
$$G^+ = (G^T G)^{-1} G^T$$

$$w = G^+ d$$

★ Q) Design a XOR problem with given data set as (1,1), (0,1), (0,0), (1,0) and also find G^+ & w.

→

i/p pattern	Desired o/p
(1,1)	0
(0,1)	1
(0,0)	0
(1,0)	1



we select. $t_1 = (1,1)$

$t_2 = (0,0)$

$t_1 = (1,1)$ $\quad t_2 = (0,0)$

$e^{-\|x-t_1\|^2}$ $\quad e^{-\|x-t_2\|^2}$ $\quad b$

$$G = \begin{bmatrix} 1 & 0.1353 & 1 \\ 0.3678 & 0.3678 & 1 \\ 0.1353 & 1 & 1 \\ 0.3678 & 0.3678 & 1 \end{bmatrix}$$

$$G^T = \begin{bmatrix} 1 & 0.3678 & 0.1353 & 0.3678 \\ 0.1353 & 0.3678 & 1 & 0.3678 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$$G^TG = \begin{bmatrix} 1.2889 & 0.5412 & 1.8709 \\ 0.5412 & 1.2889 & 1.8709 \\ 1.8709 & 1.8709 & 4 \end{bmatrix}$$

$|G^TG| = 0.8392$

Adjacency.

$$\frac{1}{|G^TG|} \begin{bmatrix} 1.6553 & 1.3355 & -1.3985 \\ 1.3355 & 1.6553 & -1.3989 \\ -1.3989 & 1.3989 & 1.3684 \end{bmatrix} \begin{bmatrix} 1 & 0.3678 & 0.1353 & 0.3678 \\ 0.1353 & 0.3678 & 1 & 0.3678 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

G^T

$$= \frac{1}{0.8392} \begin{bmatrix} 0.4371 & -0.2989 & -1.0414 & -0.2989 \\ 0.1606 & -0.2959 & 0.4371 & -0.2989 \\ 0.1588 & 1.3684 & 2.5780 & 1.3684 \end{bmatrix}$$

$$G^\dagger = \begin{bmatrix} 1.5273 & -1.2496 & 0.6727 & -1.2499 \\ 0.6727 & -1.2496 & 1.8292 & -1.2496 \\ -0.9202 & 1.4202 & -0.9202 & 1.42 \end{bmatrix}$$

$W = G^\dagger d.$

$$= G^\dagger \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

$$W = \begin{bmatrix} -2.5018 \\ -2.5018 \\ 2.8404 \end{bmatrix}$$

② Strict Interpolation with Regularisation →

$$\phi = \begin{bmatrix} t_1=0.1 & t_2=0.1 & t_3=0.0 & t_4=1.0 \\ 1.0 & 0.3678 & 0.1353 & 0.3678 \\ 0.3678 & 1.0 & 0.3678 & 0.1353 \\ 0.1353 & 0.3678 & 1.0 & 0.3678 \\ 0.3678 & 0.1353 & 0.3678 & 1.0 \end{bmatrix}$$

$$\phi^{-1} = \begin{bmatrix} 1.3373 & -0.4918 & 0.1809 & -0.4918 \\ -0.4918 & 0.13373 & -0.4918 & 0.1809 \\ 0.1809 & -0.4918 & 1.3373 & -0.4918 \\ 0.4918 & 0.1809 & -0.4918 & 1.3373 \end{bmatrix}$$

$$W = \phi^{-1} d$$

$$W = \begin{bmatrix} -0.9837 \\ 1.5182 \\ -0.9837 \\ 1.5182 \end{bmatrix}$$

☆ UNSUPERVISED LEARNING NEURAL NETWORKS →

① Hopfield Network →

☆ Associative Memories :- It, refers to a memory organisation in which memory is accessed by it's contents as opposed to an explicit address like traditional computer memory. This type of memory allows recall of information based on partial knowledge of it's contents. ANN can be used as associative memory eg. Hopfield Net.

AC → [ABC / DEF] → ABC.

Content addr. memory

There are two classes of associative memories.

① Autoassociative :- i/p & o/p patterns are similar

eg. HN

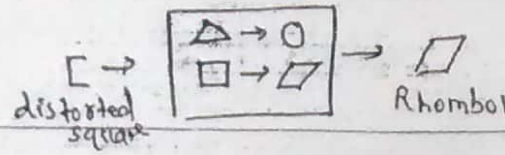
i/p → [Stored Patterns] → o/p

- corrupted noisy
- associate i/p to closest stored pattern

[→ distorted square → ADD → □

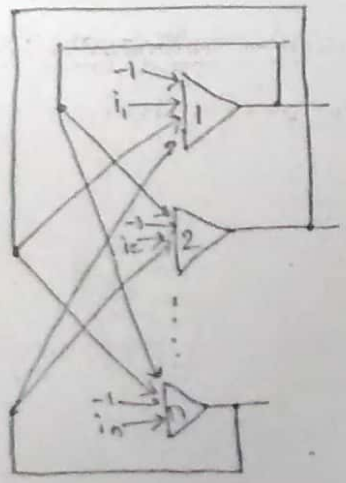
③ Heteroassociative:- i/p & o/p patterns are not same association betn pairs of patterns are stored.

eg. BAM.



☆ Hopfield Network: ⟶

A single n/w of interconnected, binary valued neurons can store multiple stable states here each neuron is connected to others but not back to itself. wt's are symmetric. i/p pattern → n/w → converges to stable state nearest to that pattern.



Algorithm:- wt assignment:-

Suppose n/w stores m patterns

$$w_{ji} = \begin{cases} \sum_{p=1}^{m} p_{j,p}\, p_{i,p}, & i \neq j \\ 0, & i = j \end{cases}$$

Calculation of Activation:-

① at time, $t = 0$

$$o_j(0) = p_j$$

② at time, $t > 0$

$$o_j(t+1) = f_h\left(\sum w_{ji}\, o_i(t)\right)$$

$$f_h(a) = \begin{cases} 1 & a > 0 \\ -1/0 & a < 0 \\ o_j(t) & a = 0 \end{cases}$$

③ Repeat step 2 until equilibrium ie. activation levels of nodes remains unchanged with further iteration the pattern upon equilibrium represents stored pattern that best matches i/p.

★ Stability of HN →

nlw changes state from $t \to t+1$, change in the energy level is given as, $\Delta E = E(t+1) - E(t)$

$= -\left[\left(\frac{1}{2}\sum w_{ki} o_i\right) + I_k - \theta_k\right] \Delta o_k$

($\frac{1}{2}\sum w_{ki} o_i$ + I_k : i/p; θ_k : Threshold; Δo_k : $o_k(t+1) - o_k(t)$)

Three possible values of $\Delta o_k = -1, 1, 0$

for $\Delta o_k = -1$ ~~1st two terms~~ → i/p to nlw & $\Delta o_k = -1$

$o_k(t+1) = 0$ & $o_k(t) = 1$

total i/p < Threshold

$\therefore \Delta E < 0$ Energy is reduced.

for $\Delta o_k = 1$ $o_k(t+1) = 1$, $o_k(t) = 0$

total i/p > Threshold

$\Delta E < 0$ ∴ Energy is reduced

for $\Delta o_k = 0$ $o_k(t+1) = o_k(t)$

$\Delta E = 0$ Energy is retained.

∴ Energy level always decreasing or constant whenever neuron changes it's state. nlw will retain a minimum energy state & stop this proves stability of nlw.

★ Energy minimisation →

i) Consider HN, $w_{ij} = w_{ji}$ & $w_{ii} = 0$

ii) S_i → State of neuron i, $i = 1 \dots N$

iii) $E = -\frac{1}{2} \sum_{i=1}^{N} \sum_{\substack{j=1 \\ i \neq j}}^{N} w_{ji} S_i S_j$

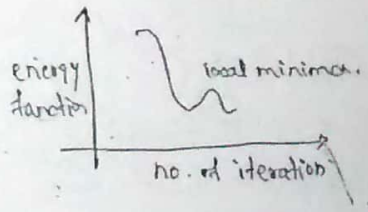
iv) $\Delta E = -S_j \sum_{\substack{i=1 \\ i \neq j}}^{N} w_{ji} S_i$

$S_i \to S_j$ change in energy

v) E decreased for state $\{S_i \mid i = 1 \dots N\}$

vi) State changes will continue until a local minima of energy landscape is reached.

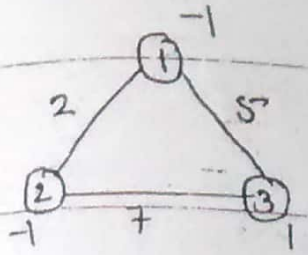
vii) Energy fun has minimum value at stable state.



★ Storage capacity of HN →

S.C = 0.15 n. , linearly varies with size 'N' of nlw.

* Q1) Comment on stability for foll. nlw.



at, $t=0$ $\quad \langle -1 \quad -1 \quad 1 \rangle$

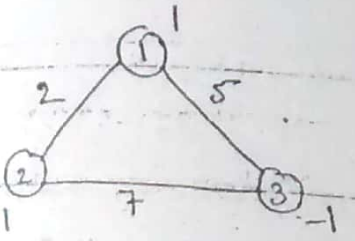
at, $t=1$

$\sigma_1(1) = \text{sgn}|2\times -1 + 5\times 1| = \text{sgn}|3| = 1$

$\sigma_2(1) = \text{sgn}|2\times -1 + 7\times 1| = \text{sgn}|5| = 1$

$\sigma_3(1) = \text{sgn}|7\times 1 + 5\times -1| = \text{sgn}|-12| = -1$

$t=1$, $\langle 1 \quad 1 \quad -1 \rangle$



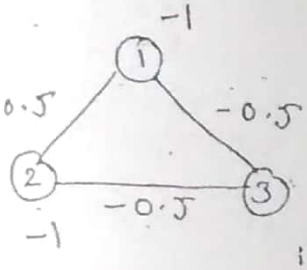
at, $t=2$ $\quad \sigma_1(2) = -1$, $\sigma_2(2) = -1$, $\sigma_3(2) = 1$

$t=2$, $\langle -1 \quad -1 \quad 1 \rangle$ same as that of $t=0$

Stated toggled between $\langle -1 \quad -1 \quad 1 \rangle$ & $\langle 1 \quad 1 \quad -1 \rangle$

unstable nlw, wt's are not properly assigned.

For selecting proper wt's. in HN $= w_{ij} = \frac{1}{N-1} \sigma_i(1)\, \sigma_j(1)$



$\langle -1 \quad -1 \quad 1 \rangle$

$w_{12} = \frac{1}{3-1}(-1)(-1) = 0.5$

$w_{13} = \frac{1}{3-1}(-1)(1) = -0.5$

$w_{23} = \frac{1}{3-1}(-1)(1) = -0.5$

Test. at $t=1$. $\langle -1 \quad -1 \quad 1 \rangle$

at $t=2$ $\langle -1 \quad -1 \quad 1 \rangle$

Same

∴ stable n/w

* Q2) Train autoassociative nlw for i/p vector $[-1 \; 1 \; 1 \; 1]$ & also test the nlw for the same i/p vector. Test the nlw with one missing, one mistake, two missing & two mistake entries in test vector.

→ $x = [-1 \; 1 \; 1 \; 1]$

$$w = \varepsilon x^T x = \begin{bmatrix} -1 \\ 1 \\ 1 \\ 1 \end{bmatrix} [-1 \; 1 \; 1 \; 1] = \begin{bmatrix} 1 & -1 & -1 & -1 \\ -1 & 1 & 1 & 1 \\ -1 & 1 & 1 & 1 \\ -1 & 1 & 1 & 1 \end{bmatrix}$$

testing vector

$y_{inj} = x \cdot w$

$$= [-1 \; 1 \; 1 \; 1] \begin{bmatrix} +1 & -1 & -1 & -1 \\ -1 & 1 & 1 & 1 \\ -1 & 1 & 1 & 1 \\ -1 & 1 & 1 & 1 \end{bmatrix} = [-4 \; 4 \; 4 \; 4] \quad 1\times 4$$

4×4

$y_j = [-1 \; 1 \; 1 \; 1]$ correct response.

one missing entry. $x = [0 \; 1 \; 1 \; 1]$

$y_{inj} = x \cdot w = [-3 \; 3 \; 3 \; 3] = [-1 \; 1 \; 1 \; 1]$ correct.

one mistake $x = [-1 \ -1 \ 1 \ 1]$

$y_{inj} = xW = [-2 \ 2 \ 2 \ 2]$

$y_j = [-1 \ 1 \ 1 \ 1]$ correct

two missing $x = [0 \ 0 \ 1 \ 1]$

$y_{inj} = [-2 \ 2 \ 2 \ 2]$

$y_j = [-1 \ 1 \ 1 \ 1]$ correct

two mistake $x = [-1 \ 1 \ -1 \ 1]$

$y = [0 \ 0 \ 0 \ 0]$ incorrect response.

★ Q3) Construct an autoassociative NN with I/P vector $[1 \ 1 \ 1 \ -1]$. Test the NN with missing entries in first & second components of stored vector. Apply asynchronous updation.

→ $x = [1 \ 1 \ 1 \ -1]$ $W = \begin{bmatrix} 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ -1 & -1 & -1 & 1 \end{bmatrix}$ $W = \begin{bmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ -1 & -1 & -1 & 0 \end{bmatrix}$

We carry out asynchronous updation as y_1, y_4, y_3, y_2

$x = [0 \ 0 \ 1 \ -1]$ $y = [0 \ 0 \ 1 \ -1]$ at $t = 0$

update y_1 $y_{in1} = x_1 + \sum_{j=1}^{4} y_j w_{j1} = 0 + [0 \ 0 \ 1 \ -1] \begin{bmatrix} 0 \\ 1 \\ 1 \\ -1 \end{bmatrix} = 2$

$y_1 = 1$

$y = [1 \ 0 \ 1 \ -1]$ → no convergence

update y_4 $y_{in4} = x_4 + \sum_{j=1}^{4} y_j w_{j4} = -1 + [1 \ 0 \ 1 \ -1] \begin{bmatrix} -1 \\ -1 \\ -1 \\ 0 \end{bmatrix} = -3$

$y_4 = -1$

$y = [1 \ 0 \ 1 \ -1]$ → no convergence.

update y_3 $y_{in3} = x_3 + \sum_{j=1}^{4} y_j w_{j3} = 1 + [1 \ 0 \ 1 \ -1] \begin{bmatrix} 1 \\ 0 \\ 1 \\ -1 \end{bmatrix} = 3$

$y_3 = 1$

$y = [1 \ 0 \ 1 \ -1]$ → no convergence

update y_2 $y_{in2} = x_2 + \sum_{j=1}^{4} y_j w_{j2} = 0 + [1 \ 0 \ 1 \ -1] \begin{bmatrix} 1 \\ 0 \\ 1 \\ -1 \end{bmatrix} = 3$

$y_2 = 1$

$y = [1 \ 1 \ 1 \ -1]$ → converges with vector x.

★ Q4) Construct an autoassociative nlw to store the vectors $x_1 = [1\ 1\ 1\ 1\ 1]$, $x_2 = [1\ -1\ -1\ 1\ -1]$; $x_3 = [-1\ 1\ -1\ -1\ -1]$ using HN test patterns, $x_1' = [1\ 1\ 1\ -1\ 1]$, $x_2' = [1\ -1\ -1\ -1\ -1]$, $x_3' = [1\ 1\ -1\ -1\ -1]$ Compare test pattern energy with stored pattern energy.

→ $W = \sum x_i^t x_i$

$$= \begin{bmatrix} 3 & -1 & 1 & -1 & 1 \\ -1 & 3 & 1 & -1 & 1 \\ 1 & 1 & 3 & 1 & 3 \\ -1 & -1 & 1 & 3 & 1 \\ 1 & 1 & 3 & 1 & 3 \end{bmatrix} \Rightarrow \begin{bmatrix} 0 & -1 & 1 & -1 & 1 \\ -1 & 0 & 1 & -1 & 1 \\ 1 & 1 & 0 & 1 & 3 \\ -1 & -1 & 1 & 0 & 1 \\ 1 & 1 & 3 & 1 & 0 \end{bmatrix}$$

$E = -0.5 [x_i \omega^t x_i^t]$

$E_1 = -0.5 [x_1 \omega^t x_1^t] = -10$ $\quad E_1' = -0.5 [x_1' \omega^t x_1'^t] =$

$E_2 = -0.5 [x_2 \omega^t x_2^t] = -6$ $\quad E_2' = -0.5 [x_2' \omega^t x_2'^t] =$

$E_3 = -0.5 [x_3 \omega^t x_3^t] = -10$ $\quad E_3' = -0.5 [x_3' \omega^t x_3'^t] =$

Applying test patterns.

For 1st pattern, $x_1' = [1\ 1\ 1\ -1\ 1]$ $\quad y = [1\ 1\ 1\ -1\ 1]$

update y_4 $\quad y_{in4} = x_{4t} + \sum_{j=1}^{5} y_j w_{j4} = -1 + [1\ 1\ 1\ -1\ 1] \begin{bmatrix} 3 \\ -1 \\ 1 \\ 0 \\ 1 \end{bmatrix}$

$= 3$

$y_4 = 1$

$\therefore y = [1\ 1\ 1\ 1\ 1]$ → converges to x_1.

For 2nd pattern.

$x_2' = [1\ -1\ -1\ -1\ -1]$, $y = [1\ -1\ -1\ -1\ -1]$

update y_4 $\quad y_{in4} = x_{4t} + \sum_{j=1}^{5} y_j w_{j4} = -1 + [1\ -1\ -1\ -1\ -1] \begin{bmatrix} 3 \\ -1 \\ 1 \\ 0 \\ 1 \end{bmatrix}$

$= 1$

$y_4 = 1$

$y = [1\ -1\ -1\ 1\ -1]$ → converges to x_2

For 3rd pattern

$x_3' = [1\ 1\ -1\ -1\ -1]$ $\quad y = [1\ 1\ -1\ -1\ -1]$

update y_1

$y_{in1} = x_1 + \sum_{j=1}^{5} y_j w_{j1}$

$= 1 + [1\ 1\ -1\ -1\ -1] \begin{bmatrix} 0 \\ -1 \\ 1 \\ 3 \\ 1 \end{bmatrix}$

$= -5$

$y_1 = -1$

$y = [-1\ 1\ -1\ -1\ -1]$ → converges to x_3

★ Q5) Consider a simple HN made up of two neurons. The four possible states of the nw are $x_1 = [1\ 1]^t$ $x_2 = [-1\ 1]^t$, $x_3 = [-1\ -1]^t$, $x_4 = [1\ -1]^t$ & $W = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}$

1) Determine that states x_2 & x_4 are stable, whereas states x_1 & x_3 exhibit a limit cycle. Do this demonstration using following tool. ① The alignment (stability) condition.
② The Energy function.

2) What is the length of limit cycle characterizing states x_1 & x_3.

→ ① alignment condⁿ →

$$W \cdot x_2 = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}, \quad \text{sgn}\begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \end{bmatrix} = x_2.$$

$\therefore x_2$ is stable.

$$W \cdot x_4 = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad \text{sgn}\begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} = x_4$$

$\therefore x_4$ is stable.

$$W \cdot x_1 = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix} \cdot \text{sgn}\begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix} = x_3$$

$\therefore x_1$ is unstable.

$$W \cdot x_3 = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} = x_1.$$

$\therefore x_3$ is unstable.

answer 2) → x_1 & x_3 exhibit a limit cycle with length = 2.

② energy fun →

$$E = -\frac{1}{2} \sum_{\substack{i,j \\ i \neq j}} w_{ji} s_i s_j$$

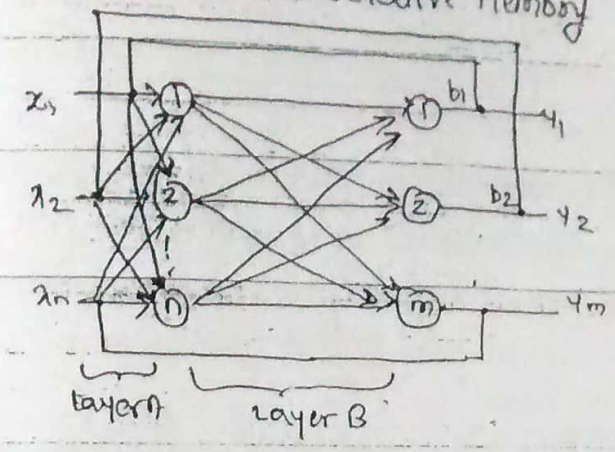
$$= -\frac{1}{2} w_{12} s_1 s_2 - \frac{1}{2} w_{21} s_2 s_1$$

$$= -w_{12} s_1 s_2 \quad \text{since } w_{12} = w_{21}$$

$$= s_1 s_2 \quad \text{since } w_{12} = -1$$

State	Energy $s_1 s_2$		
x_1 [1 1]	1	unstable	minimum value represent stability ∴ x_2 & x_4
x_2 [-1 1]	-1	<0 stable	
x_3 [-1 -1]	1	unstable	
x_4 [1 -1]	-1	<0 stable.	

Bidirectional Associative Memory (BAM) →



BAM is a heteroassociative memory. This is also called as Resonance model because information is passed back & forth until convergence.

x is applied from layer A to B with wt. matrix w & again y is applied from layer B to A with w^T this is repeated until the network gets converged.

algorithm →

① wt. assignment: → $w_{ij} = \sum_{p=1}^{m} P_{i,p} Q_{j,q}$

② Calculation of activation: → 1) at $t = 0$

$O_j(0) = P_i$

2) at $t > 0$

$O_j(t+1) = f(\sum w_{ij} O_j(t))$

3) Repeat step 2 until equilibrium

Upon equilibrium output represent the pattern which is associated with the input pattern.

Q1) Three pairs of vectors are stored using BAM. Show that memory associated perfectly.

$A_1 = [1\ -1\ -1]$ & $B_1 = [1\ -1]$, $A_2 = [-1\ 1\ -1]$ & $B_2 = [-1\ 1]$

$A_3 = [-1\ -1\ 1]$ & $B_3 = [1\ -1]$

→

$W = \sum A_i^T B_i$

$= \begin{bmatrix} 1 & -1 \\ -3 & 3 \\ 1 & -1 \end{bmatrix}$

$\begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} [1\ -1]$

① Apply A_1 → $f(A_1 W) = f(3\ -3) = [1\ -1] = B_1$ correct

$[1\ -1\ -1] \begin{bmatrix} 1 & -1 \\ -3 & 3 \\ 1 & -1 \end{bmatrix}$ $f(3\ -3) = (1\ -1) \cdot B_1$

② Apply B_1

→ $f(W^T B_1) \neq A_1$

$\begin{bmatrix} 1 & -3 & 1 \\ -1 & 3 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix}$

Q2. Train heteroassociative memory n/w to store S=(s1 s2 s3 s4) & o/p vectors are t=(t1 t2)

s1	s2	s3	s4	t1	t2
1	0	0	0	0	1
1	1	0	0	0	1
0	0	0	1	1	0
0	0	1	1	1	0

$$\rightarrow W = \sum S^T(\cdot) = \begin{bmatrix}1\\0\\0\\0\end{bmatrix}[0\ 1] + \begin{bmatrix}1\\1\\0\\0\end{bmatrix}[0\ 1] + \begin{bmatrix}0\\0\\0\\1\end{bmatrix}[1\ 0] + \begin{bmatrix}0\\0\\1\\1\end{bmatrix}[1\ 0]$$

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

Test with similar vector.

$x = [1\ 0\ 0\ 0]$

$f(x \cdot w) = f[0\ 2] = [0\ 1]$ = correct response

with unsimilar (one missing entry in 1st component of 2nd vector)

$x = [0\ 1\ 0\ 0]$

$f(x \cdot w) = f(0\ 2) = (0\ 1)$ = correct response

Q3. find the wt. matrix in bipolar form for BAM for foll. i/p-o/p vector pair.

S =		t =		first we convert to bipolar →		
1 0 0 0		1 0		1 -1 -1 -1		1 -1
1 0 0 1		1 0		1 -1 -1 1		1 -1
0 1 0 0		0 1		-1 1 -1 -1		-1 1
0 0 1 0		0 1		-1 -1 1 -1		-1 1

$$\rightarrow W = \begin{bmatrix}4&-4\\-4&4\\-2&2\\2&-2\end{bmatrix}$$

Test vector [1 -1 -1 -1]

$f(x \cdot w) = f(8\ -8) = [1\ -1]$ correct

Test vector [1 0 -1 -1] missing entry.

$f(x \cdot w) = f(4\ -4) = [1\ -1]$ correct

★ (2) Principal Component Analysis →

PCA is one of the most successful techniques that has been used in image recognition & compression. The purpose of PCA is to reduce the large dimensionality of data space to smaller intrinsic dimensionality of feature space which are needed to describe the data economically.

Q1) Apply PCA, find eigen vector (Principal component)

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2.0	1.6
1.0	1.1
1.5	1.6
1.10	0.9
total → 18.1/10	19.1/10
$\bar{x}$ → 1.81	$\bar{y}$ → 1.91

Covariance matrix, $C = \begin{bmatrix} c_{x,x} & c_{x,y} \\ c_{y,x} & c_{y,y} \end{bmatrix}$

$c_{x,x} = \frac{\Sigma(x-\bar{x})^2}{n-1}$ $c_{x,y} = c_{y,x} = \frac{\Sigma(x-\bar{x})(y-\bar{y})}{n-1}$

$c_{y,y} = \frac{\Sigma(y-\bar{y})^2}{n-1}$

$$C = \begin{bmatrix} 0.6165 & 0.61544 \\ 0.61544 & 0.7165 \end{bmatrix}$$

$|C - \lambda I| = 0$

$$\left| \begin{bmatrix} 0.6165 & 0.61544 \\ 0.61544 & 0.7165 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right| = 0$$

By solving above determinant we will get quadratic eqn & solving that eqn will get two eigen values of $\lambda = \lambda_1 = 0.0489$ & $\lambda_2 = 1.284$

$C V_1 = \lambda_1 V_1$

18.1 / 10 = 1.81

$$\begin{bmatrix} 0.6165 & 0.61544 \\ 0.61544 & 0.7165 \end{bmatrix} \begin{bmatrix} v_{11} \\ v_{12} \end{bmatrix} = 0.0489 \begin{bmatrix} v_{11} \\ v_{12} \end{bmatrix}$$

$0.6165\, v_{11} + 0.61544\, v_{12} = 0.0489\, v_{11}$

$0.5676\, v_{11} + 0.61544\, v_{12}$ — (1)

$0.5676\, v_{11} = -0.61544\, v_{12}$

$v_{11} = -1.0842\, v_{12}$

$$v_1 = \begin{bmatrix} -1.0842 \\ 1 \end{bmatrix} \quad v_{1N} = \frac{1}{\sqrt{(-1.0842)^2 + 1^2}} \begin{bmatrix} -1.0842 \\ 1 \end{bmatrix}$$

$$v_{1N} = \begin{bmatrix} -0.735 \\ 0.677 \end{bmatrix}$$

$$CV_2 = \lambda_2 V_2$$

$$\begin{bmatrix} 0.6165 & 0.61544 \\ 0.61544 & 0.7165 \end{bmatrix} \begin{bmatrix} V_{21} \\ V_{22} \end{bmatrix} = 1.284 \begin{bmatrix} V_{21} \\ V_{22} \end{bmatrix}$$

$$0.6165 V_{21} + 0.61544 V_{22} = 1.284 V_{21}$$

$$-0.6675 V_{21} = -0.61544 V_{22}$$

$$V_{21} = 0.922 V_{22}$$

$$V_2 = \begin{bmatrix} 0.922 \\ 1 \end{bmatrix} \quad V_{2N} = \frac{1}{\sqrt{(0.922)^2 + 1^2}} \begin{bmatrix} 0.922 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0.677 \\ 0.735 \end{bmatrix}$$

$$V_1 = \begin{bmatrix} 0.735 \\ 0.677 \end{bmatrix} \quad V_2 = \begin{bmatrix} 0.677 \\ 0.735 \end{bmatrix}$$

max. eigen value is $\lambda_2 = 1.284$ ∴ Principle component is V_2

★ ③ COMPETITIVE LEARNING NETWORKS →

With no available information regarding the desired o/ps, unsupervised learning networks update wt's only on the basis of the input patterns. The competitive learning n/w is a popular scheme to achieve this type of unsupervised data clustering or classification. All i/p units i are connected to all o/p units j with wt. w_{ij}. The number of i/p's is the i/p dimension, while the number of o/p's is equal to the number of clusters that the data are to be divided into. A cluster's position is specified by the wt. vector connected to the corresponding o/p unit.

The activation value o_j of o/p unit j is then calculated by the inner product of i/p & wt. vector

$$o_j = \sum_{i=1}^{n} w_{ji}^t x_i$$

Next, the o/p unit with the highest activation must be selected for further processing, which is what is implied by competitive. The weights leading to the winner unit are updated according to the competitive or the so-called winner-take-all LR.

$$\Delta w_{mj} = \alpha(x_j - w_{mj})$$

★ 1) Kohonen Self-organising Networks →

KSOM, also known as Kohonen feature maps or topology preserving maps, are another competition-based n/w paradigm for data clustering. N/w's of this type impose a neighborhood constraint on the o/p units, such that a certain topological property in the i/p data is reflected in the o/p unit's w/s.

Figure 1 presents a relatively simple SOM n/w with 2 i/p & 49 o/p's. The learning procedure of SOM n/w is similar to that of competitive learning n/ws. That is a similarity (dissimilarity) measure is selected and the winning unit is considered to be the one with the largest (smallest)

activation. For Kohonen feature maps, however, we update not only the winning unit's wts but also all of the wt's in a neighborhood around the winning unit. The neighborhood's size generally decreases slowly with each iteration, as indicated in Figure 2.

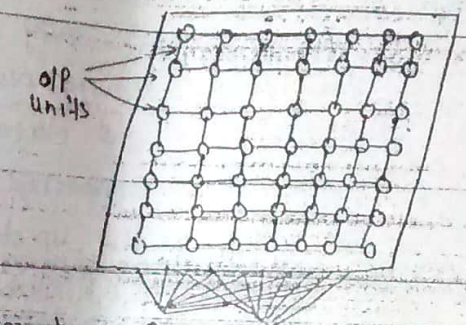


Figure 1

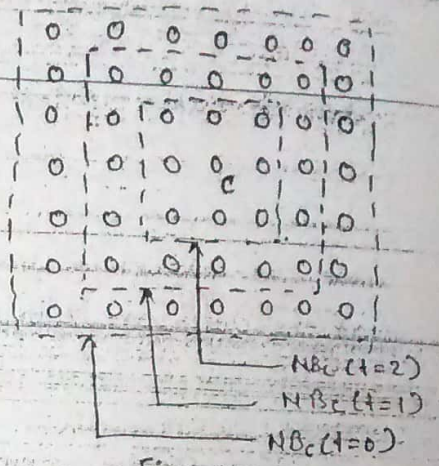


Figure 2

Training Steps:-

Step1 – Select the winning o/p unit as the one with largest similarity measure (or smallest dissimilarity measure) bet^n wt. vectors w_i & i/p vector x. If the Euclidean distance is chosen as dissimilarity measure, then the winning unit c satisfies the foll. equation.

$$\|x - w_c\| = \min_i \|x - w_i\|$$

where c → winning unit.

Step2 – Let NB_c denote a set of indices corresponding to a neighborhood around winner c. The wt's of the winner & it's neighboring units are then updated by :

$$\Delta w_i = \eta (x - w_i), \quad i \in NB_c.$$

where η → small +ve learning rate.

Instead of defining the neighborhood of a winning unit, we can use a neighborhood function $\Lambda_c(i)$ around a winning unit c. For instance, Gaussian function can be used as neighborhood function:

$$\Lambda_c(i) = \exp\left(\frac{-\|P_i - P_c\|^2}{2\sigma^2}\right)$$

where P_i & P_c → positions of the o/p units i & c resp.

σ → scope of neighborhood.

By using the neighborhood fun, the update formula can be written as,

$$\Delta w_i = \eta \Lambda_c(i)(x - w_i)$$

To achieve a better convergence, η & size of neighborhood (σ) should be decreased gradually with each iteration.

☆ e) Learning Vector Quantization →

LVQ is an adaptive data classification method based on training data with desired class information.

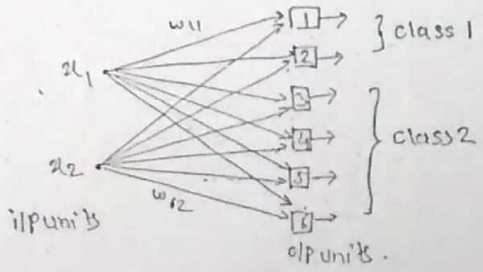


Figure 1

Figure 1 presents an eg, where the i/p dimension is 2 & the i/p space is divided into 6 clusters. The first 2 belong to class1 & rest four belong to class 2.

The LVQ algo, involves 2 steps In the first step, an unsupervised learning data clustering method is used to locate several cluster centers without using the class information. In the second step, the class information is used to fine-tune the clusters, centers to minimize the number of misclassified cases. After learning, an LVQ nlw classifies an i/p vector by assigning it to the same class as the o/p unit that has the wt vector (cluster center) closest to i/p vector.

Training steps:-

Step1 – Initialize the cluster centers by a clustering method.

step2 – Label each cluster by the voting method (ie a cluster is labeled class k if it has data points belonging to class k as a majority within the cluster).

Step3 – Randomly select a training i/p vector x and find k such that $\|x - w_k\|$ is a minimum.

(ie., wt vector w that is closest to i/p vector x must be found. If x & w belong to same class, we move w toward x; otherwise move w away from x)

Step4 – If x & w_k belong to same class, update w_k by

$$\Delta w_k = \eta(x - w_k)$$

otherwise, update w_k by

$$\Delta w_k = -\eta(x - w_k)$$

$\eta \rightarrow$ +ve small constant η should decrease with each iteration.

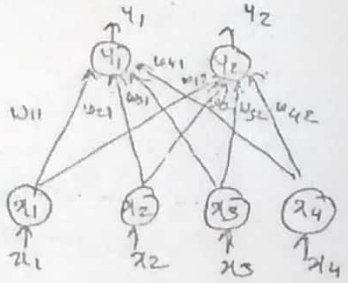
Step5 – If the maximum number of iterations is reached, stop – otherwise, return to step 3.

* Q) Construct a kohonean SOM to cluster the four given vectors, [0 0 1 1] [1 0 0 0] [0 1 1 0] & [0 0 0 1]. The number of clusters to be formed is two. Assume an initial learning rate of 0.5.

⟶ $n = 4, m = 2$, $\alpha(0) = 0.5$

initialise wt's randomly between 0 & 1

$$w_{ij} = \begin{bmatrix} 0.2 & 0.9 \\ 0.4 & 0.7 \\ 0.6 & 0.5 \\ 0.8 & 0.3 \end{bmatrix}_{4\times 2}$$



First i/p vector.

Step1 :- for $x = [0\ 0\ 1\ 1]$

Step2 :- Calculate Euclidean distance

$$D(j) = \sum_i (w_{ij} - x_i)^2$$

$$D(1) = \sum_{i=1}^{4} (w_{i1} - x_i)^2$$

$$= (0.2-0)^2 + (0.4-0)^2 + (0.6-1)^2 + (0.8-1)^2$$

$$D(1) = 0.4$$

$$D(2) = \sum_{i=1}^{4} (w_{i2} - x_i)^2$$

$= (0.9-0)^2+(0.7-0)^2+(0.5-1)^2+(0.3-1)^2 = 2.04$

Step3 - Since $D(1) < D(2)$ $\therefore$ $D(1)$ is winning cluster unit i.e. $J=1$

Step4 - Update the wt's of the winning cluster unit $J=1$

$w_{ij}(new) = w_{ij}(old) + \alpha [x_i - w_{ij}(old)]$

$w_{i1}(new) = w_{i1}(old) + 0.5 [x_i - w_{i1}(old)]$

$w_{11}(new) = w_{11}(old) + 0.5(x_1 - w_{11}(old)) = 0.2 + 0.5(0-0.2) = 0.1$

$w_{21}(new) = 0.2$, $w_{31} = 0.8$, $w_{41} = 0.9$

The updated wt. matrix after presentation of first i/p pattern is.

$$w_{ij} = \begin{bmatrix} 0.1 & 0.9 \\ 0.2 & 0.7 \\ 0.8 & 0.5 \\ 0.9 & 0.3 \end{bmatrix}$$

Second i/p vector

Step1 - for $x = [1000]$

Step2 - $D(1) = \sum_{i=1}^{4} (w_{i1} - x_i)^2$

$= (0.1-1)^2 + (0.2-0)^2 + (0.8-0)^2 + (0.9-0)^2$

$= 2.3$

$D(2) = \sum_{i=1}^{4} (w_{i2} - x_i)^2$

$= 0.94$

Step3 - Since $D(2) < D(1)$ $\therefore$ $D(2)$ is winning unit i.e. Y_2, $J=2$

Step4 - update wt's for $J=2$

$w_{12}(new) = 0.95$, $w_{22}(new) = 0.35$

$w_{32}(new) = 0.25$, $w_{42}(new) = 0.15$

$$w_{ij} = \begin{bmatrix} 0.1 & 0.95 \\ 0.2 & 0.35 \\ 0.8 & 0.25 \\ 0.9 & 0.15 \end{bmatrix}$$

Third i/p vector

Step1 - for $x = [0110]$

Step2 - $D(1) = 1.5$, $D(2) = 1.91$

Step3 - $D(1) < D(2)$ winning unit Y_1, $J=1$

Step4 - $w_{11} = 0.05$, $w_{21} = 0.6$, $w_{31} = 0.9$, $w_{41} = 0.95$

$$\begin{bmatrix} 0.05 & 0.95 \\ 0.6 & 0.35 \\ 0.9 & 0.25 \\ 0.95 & 0.15 \end{bmatrix}$$

Fourth i/p vector

Step1 - for x = [0001]

Step2 - D(1) = 1.175, D(2) = 1.81

Step3 - ∵ D(1) < D(2) winning unit is y1, J = 1

Step4 - $w_{11} = 0.025$, $w_{21} = 0.3$, $w_{31} = 0.495$, $w_{41} = 0.975$

$$w_{ij} = \begin{bmatrix} 0.025 & 0.975 \\ 0.3 & 0.35 \\ 0.495 & 0.25 \\ 0.975 & 0.15 \end{bmatrix}$$

End of 1st epoch

Now the learning rate is updated as, $\alpha(t+1) = 0.5\,\alpha(t)$.

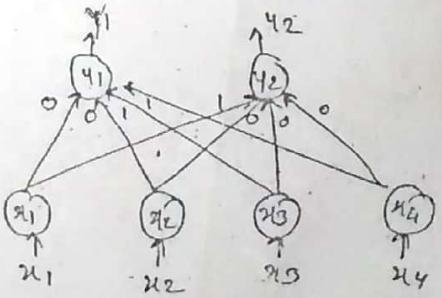
$\alpha(1) = 0.5\,\alpha(0) = 0.5 \times 0.5$

$= 0.25$

Now the procedure is repeated.

* Q2) Construct & test an LVQ net with five vectors assigned to 2 classes. The given vectors along with the classes are as shown in table below.

vector	class
0011	1
1000	2
0001	2
1100	1
0110	1



In the given 5 vectors first 2 vectors are used as initial weight vectors. 4 remaining 3 are used as i/p vector

$w_1 = 0011$

$w_2 = 1000$

$\alpha = 0.1$

First i/p vector for [0001], T = 2

$$D(j) = \sum_{i=1}^{4} (w_{ij} - x_i)^2$$

for j = 1 to 2

$D(1) = (0-0)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 = 1$

$D(2) = (1-0)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 = 2$

Since D(1) < D(2), D(1) is winner, J = 1

Now T = J w_ij's are updated as.

$w_J(new) = w_J(old) - \alpha[x - w_J(old)]$

$w_{11}(new) = w_{11}(old) - \alpha[x_1 - w_{11}(old)]$

$= 0 - 0.1(0-0) = 0$

$w_{21} = 0$, $w_{31} = 1.1$, $w_{41} = 1$

Required weight: $w = \begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 1.1 & 0 \\ 1 & 0 \end{bmatrix}$

8879415940