Introduction

CSE 6025, MSCSE
Lecture-1
Neural Network and Fuzzy
Dr. Mohammad Nurul Huda

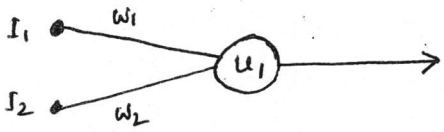
Neural Network:

- Simple processing unit.
- can store knowledge.
- is available for use.

NN 1

Similarities between Neural Network and human brain:

1. Through learning knowledge is acquired.
2. Synaptic weights are used to store knowledge.



w_1, w_2 synaptic weights

u_1 is neuron

I_1, I_2 inputs.

Human Brain:

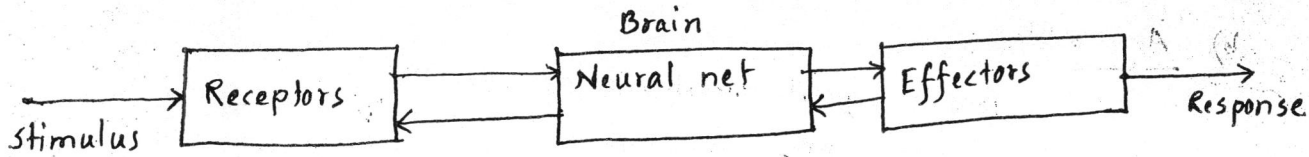


Fig: Human nervous system

Receptors:

- Convert stimuli from the human body/external environment into electrical pulses.
- Electrical pulses convey information to the neural net (brain).

Neural net (Brain):

Process information and generates electrical impulses.

Effectors:

Convert electrical impulses from brain into responses as system output.

Models of a Neuron:

A neuron is an information-processing unit that is fundamental operation of a neural network.

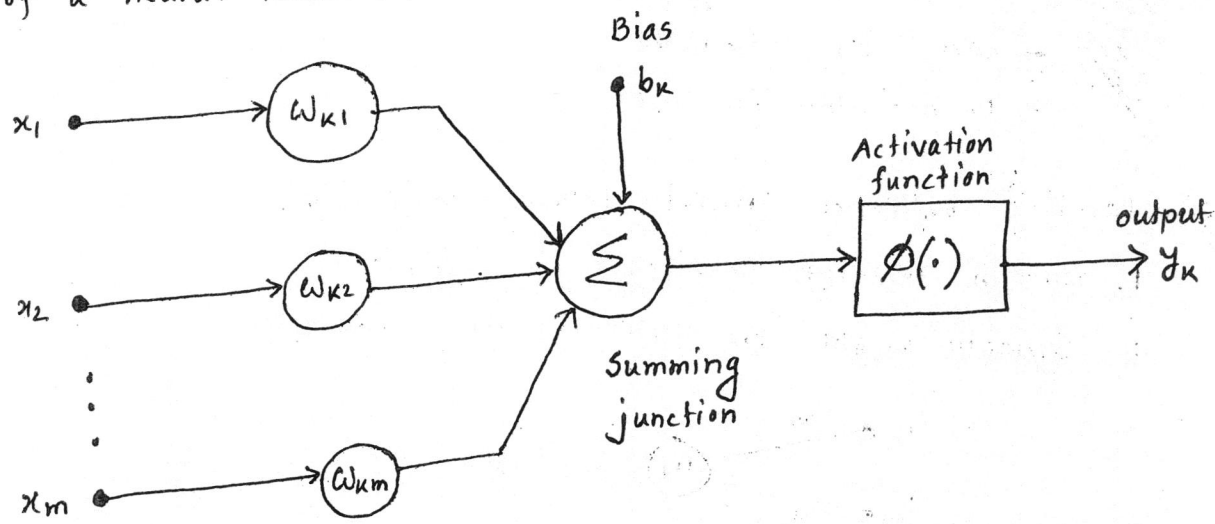


Fig: Model of a neuron.

a) A set of synapses or connecting links:

Weights $(w_{k1}, w_{k2}, \ldots\ldots w_{km})$

b) An adder:

input signal weighted by synapses.

$$u_k = \sum_{i=1}^{m} x_i w_{ki}$$

An activation function:

c) Bias:

- Increases or lowers the net input of the activation function
- May be positive or negative.

$$\text{net input} = u_k + b_k$$

An activation function :

- Limits the amplitude of the output of a neuron.
- It is called squashing function.

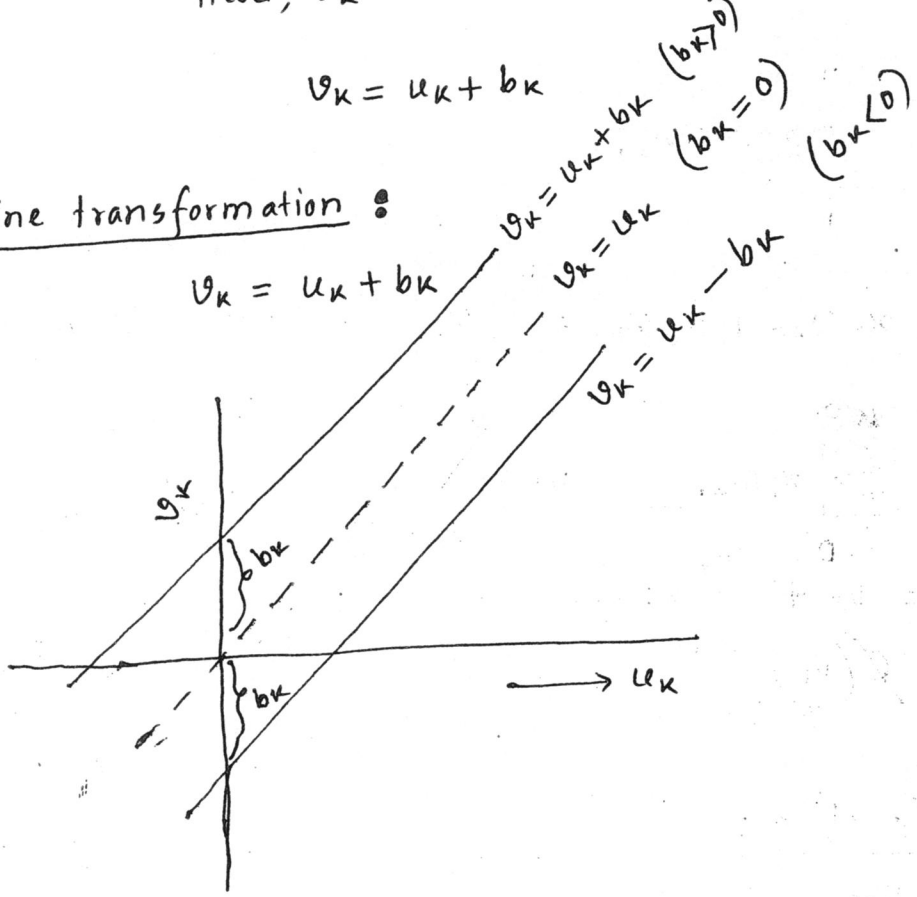
$$y_k = \phi(\text{net input})$$
$$= \phi(u_k + b_k)$$
$$= \phi(v_k) \quad \text{where} \quad v_k = u_k + b_k$$
ϕ is activation function

Here, v_k is induced local field or activation potential.

$$v_k = u_k + b_k$$

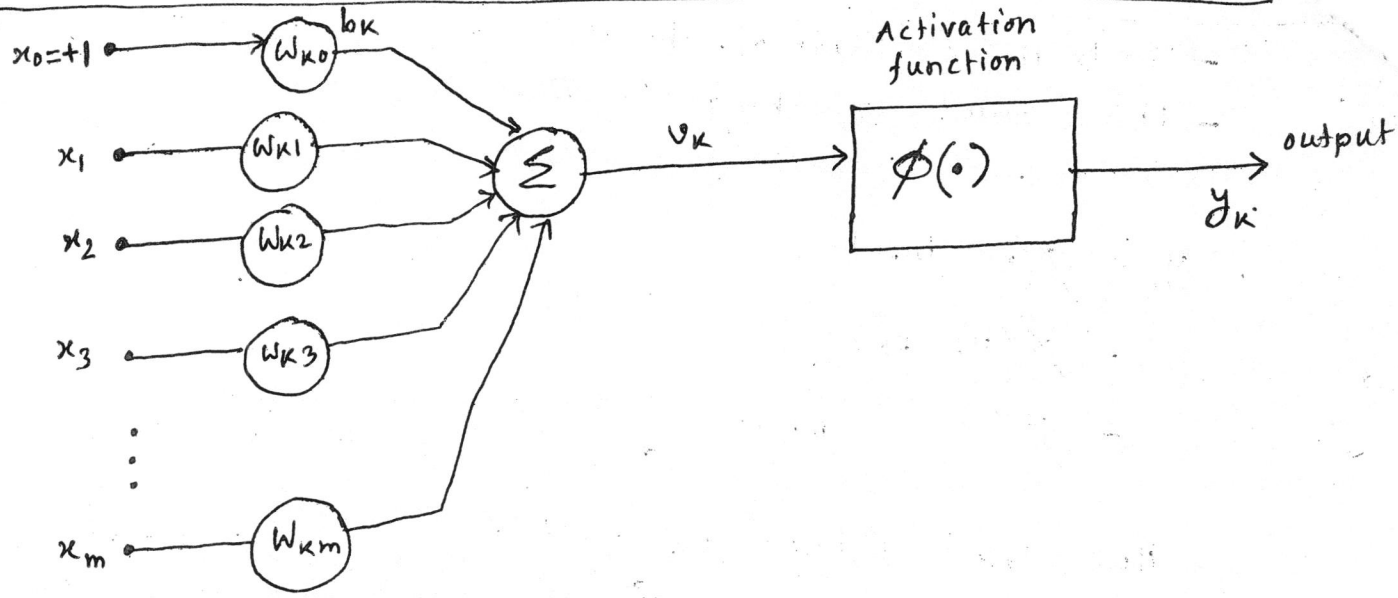
Affine transformation :

$v_k = u_k + b_k$

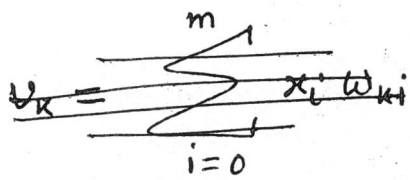


This diagram shows how induced local field is modified depending on the values of b_k.

Models of a neuron considering bias as a synaptic weight



$$v_k = \sum_{i=0}^{m} x_i w_{ki}$$



$$v_k = x_0 w_{k0} + x_1 w_{k1} + x_2 w_{k2} + \cdots + x_m w_{km}$$

$$= \sum_{i=0}^{m} x_i w_{ki} \qquad \text{where } x_0 = 1 \text{ and } w_{k0} = b_k$$

$$= b_k + \sum_{i=1}^{m} x_i w_{ki}$$

$$y_k = \phi(v_k)$$

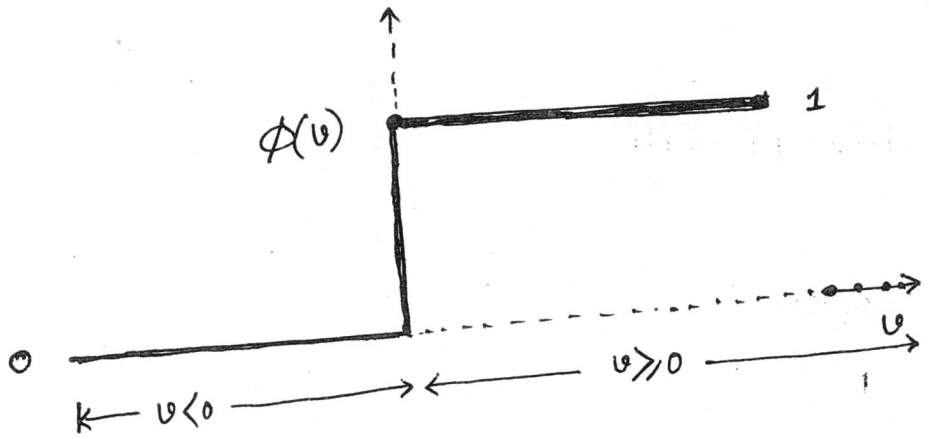
Types of Activation function :

- Theshold function.
- Piecewise-Linear function.
- Sigmoid function.
- Signum function.

(All models are deterministic)

id function:

$$\phi(v) = \begin{cases} 1 & v \geq 0 \\ 0 & v < 0 \end{cases}$$



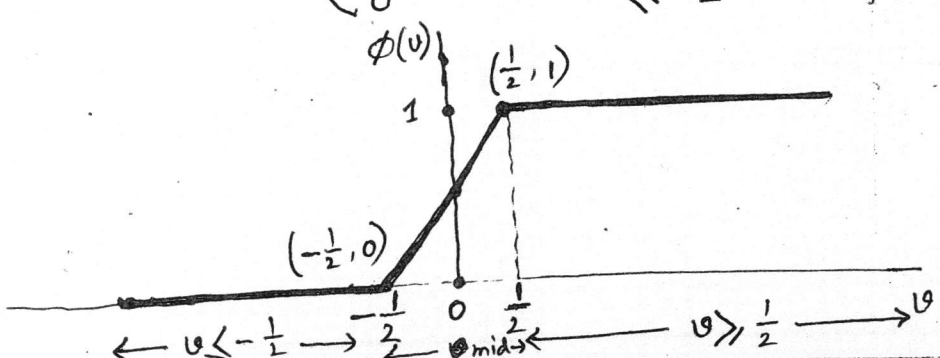
This is also called Heaviside function.

For neural network, output, $y_k = \begin{cases} 1 & \text{if } v_k \geq 0 \\ 0 & \text{if } v_k < 0 \end{cases}$ [McCulloch-Pitts Model]

Where $v_k = \sum_{i=1}^{m} x_i w_{ki} + b_k$

Piecewise-linear function:

$$\phi(v) = \begin{cases} 1 & v \geq \frac{1}{2} \quad \text{high} \\ v & \frac{1}{2} > v > -\frac{1}{2} \quad \text{mid} \\ 0 & v \leq -\frac{1}{2} \quad \text{low} \end{cases}$$



Mid part:

$$\frac{\phi(v) - 1}{1 - 0} = \frac{v - \frac{1}{2}}{\frac{1}{2} - (-\frac{1}{2})}$$

$$\Rightarrow \phi(v) - 1 = v - \frac{1}{2}$$

Sigmoid function:

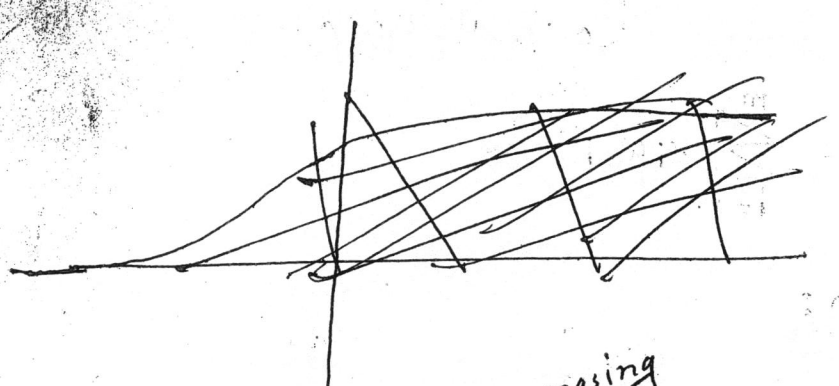
— Used in artificial neural network.

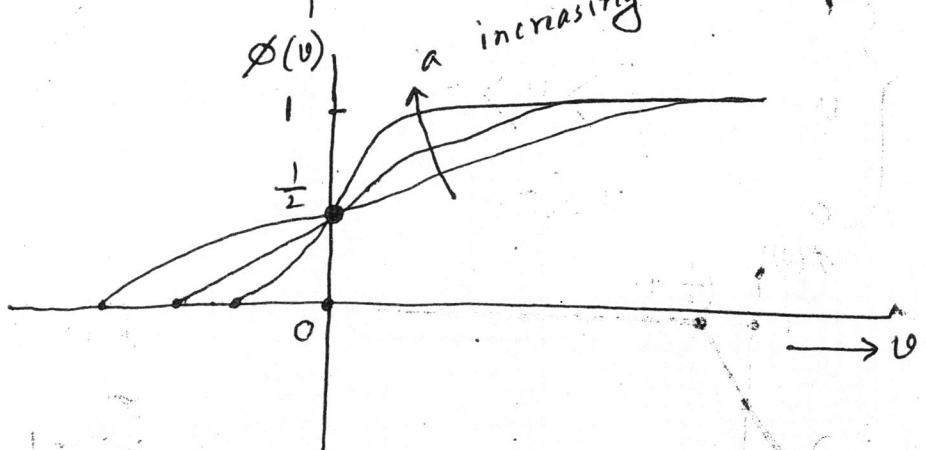
Logistic function

$$\phi(v) = \frac{1}{1 + e^{-av}}$$

Where a is slope parameter

v	$\phi(v)$
∞	1
0	$\frac{1}{2}$
$-\infty$	0





aw $\phi(v)$ using excel for the following values

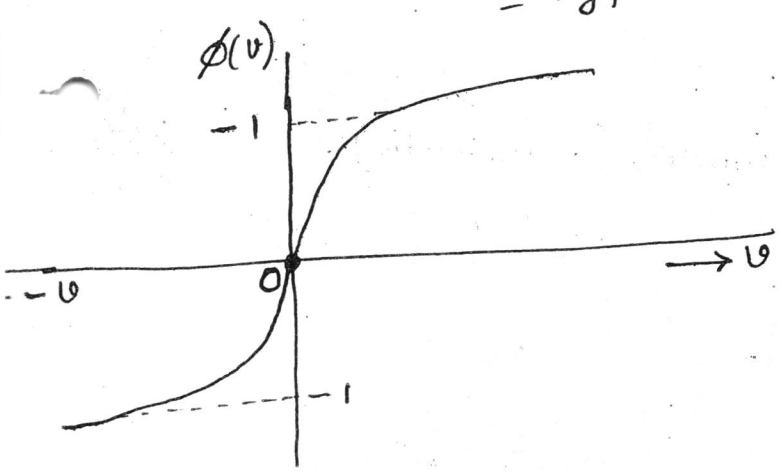
v	$\phi(v)$ at $a=1$	$\phi(v)$ at $a=2$	$\phi(v)$ at $a=3$
−10			
−8			
−6			
−4			
−2			
0			
2			
4			
6			
8			
10			

Function:

$$\phi(v) = \begin{cases} 1 & v > 0 \\ 0 & v = 0 \\ -1 & v < 0 \end{cases}$$

where $\phi(v) = \tanh(v)$

= hyperbolic tangent function



(4)

Stochastic Model of a neuron :

A neuron is permitted to reside in only one of two states +1

$$x = \begin{cases} +1 & \text{with probability } P(v) \\ -1 & \text{with probability } 1 - P(v) \end{cases}$$

x = state of the neuron.

The decision for a neuron to fire (switch its state from off to on) is probabilistic.

$P(v)$ = Probability of firing.

$$P(v) = \frac{1}{1 + e^{-\frac{v}{T}}}$$

Where v is induced local field.

T = Pseudo temperature

T is used to control the noise level and therefore the uncertainty in firing.

$$T \to 0 \qquad P(v) = \frac{1}{1 + e^{-\infty}}$$

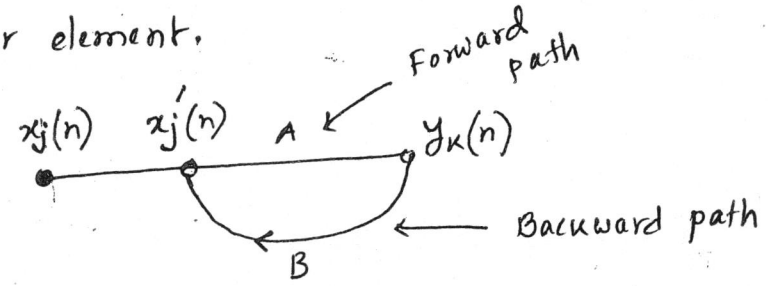
$$= \frac{1}{1}$$

$$= 1$$

The model is noiseless (deterministic).

in neural Network :

Feedback is said to exist in a dynamic system whenever the output of an element in the system in part the input applied to that particular element.



$x_j(n)$ = input signal

$x_j'(n)$ = Internal signal

$y_k(n)$ = Output signal

2(a)

$$x_j'(n) = x_j(n) + B[y_k(n)] \quad \cdots\cdots (1)$$

$$y_k(n) = A[x_j'(n)] \quad \cdots\cdots (11)$$

From eq (1) and eq (11) $\Rightarrow$

$$x_j'(n) = x_j(n) + B[y_k(n)]$$

$$x_j'(n) = A^{-1}[y_k(n)]$$

$$\therefore A^{-1}[y_k(n)] = x_j(n) + B[y_k(n)]$$

$$\Rightarrow y_k(n) = A[x_j(n) + B[y_k(n)]]$$

$$\Rightarrow y_k(n) = A[x_j(n)] + AB[y_k(n)]$$

$$\Rightarrow y_k(n) - AB[y_k(n)] = A[x_j(n)]$$

$$\Rightarrow (1-AB)\, y_k(n) = A[x_j(n)]$$

$$y_k(n) = \frac{A}{1-AB}[x_j(n)]$$

5)

This is input-output relationship for single-loop feedback system.

$$\frac{A}{1-AB} = \frac{\omega}{1-\omega z^{-1}}$$

$$= \omega \left(1-\omega z^{-1}\right)^{-1}$$

$$= \omega \left[1+\omega^1 z^{-1} + \omega^2 z^{-2} + \omega^3 z^{-3} + \cdots\right]$$

$$= \omega \sum_{l=0}^{\infty} \omega^l z^{-l}$$

$$y_k(n) = \frac{A}{1-AB}\left[x_j(n)\right]$$

$$= \omega \sum_{l=0}^{\infty} \omega^l z^{-l}\left[x_j(n)\right]$$

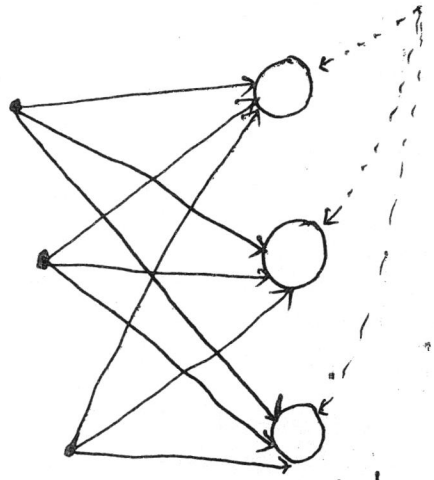
$$= \sum_{l=0}^{\infty} \omega^{l+1} z^{-l}\, x_j(n)$$

$$= \sum_{l=0}^{\infty} \omega^{l+1} x_j(n-l)$$

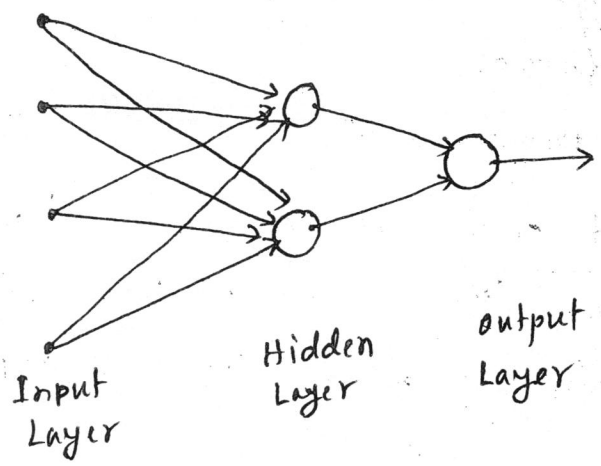
Output signal $y_k(n)$ is a weighted summation of present and past samples of the input signal $x_j(n)$.

work Architecture :

Single-Layer Feed forward Networks :

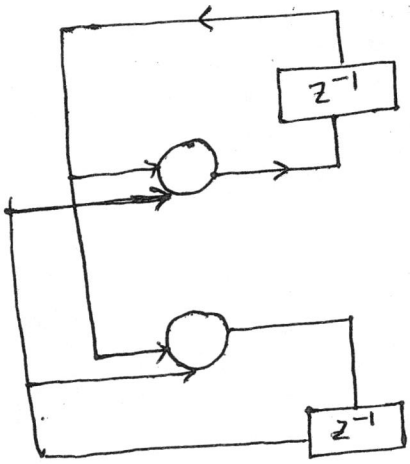


Multi layer feed forward network :



Recurrent Neural Network :

A recurrent network consists of a single layer of neurons with each neuron feeding its output signal back to the inputs of all the other neurons.



z^{-1} unit delay operator.

Knowledge Representation:

Knowledge:

stored information used by a person or machine in response to the outside world.

Two charc of Knowledge representation:

- Explicit information
- Encoded information for subsequent use.

Knowledge of the world consists of two kinds of information:

- Prior information. (Known state of the world)
- Examples to train the network. (Training samples)

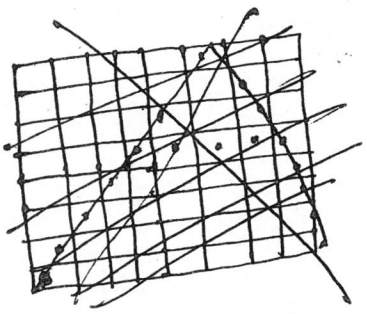
Examples
- Labelled. (input signal, output/target signal).
- Unlabeled. (Different realization of input by itself)

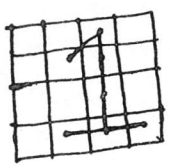
...iem 8:
Handwritten digit recognition problem.

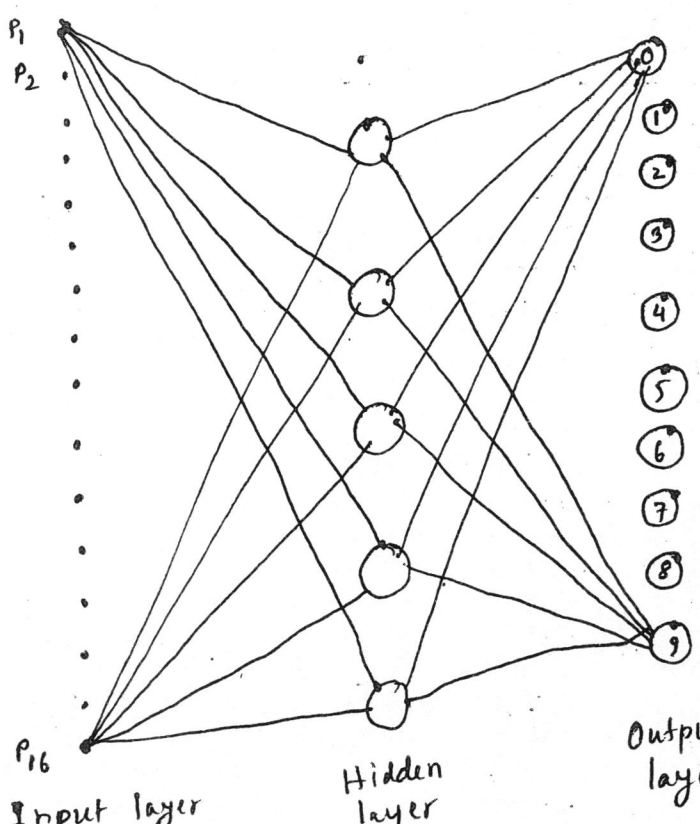
Training samples:

A large variety of handwritten digits

input: image with black or white pixels.
Output: one of 10 digits.





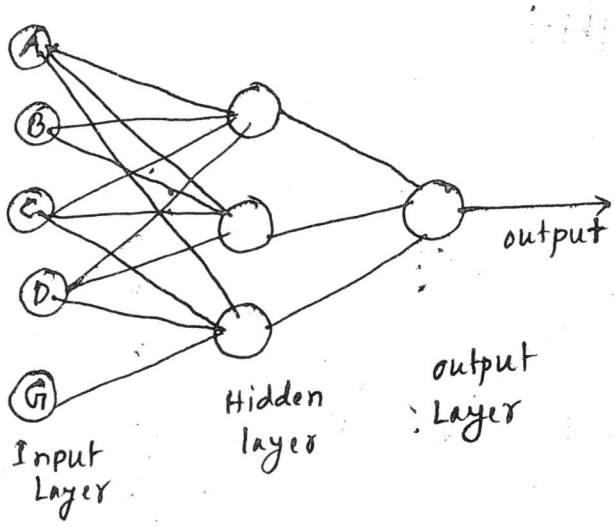


Two pages: - learning (Training with known input/output)
- Generalization (Testing for unknown data)

Problem 2:

A bank has four managers and one general manager. If managers or the general manager are present lock of the bank will be opened. Design neural network modeling for the problem.

Input					Output
A	B	C	D	G	Lock
Not	Not	Not	Not	Not	CLOSE
Not	Not	Not	Not	OK	OPEN
Not	Not	Not	OK	Not	CLOSE
Not	Not	Not	OK	OK	OPEN
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
OK	OK	OK	OK	OK	OPEN



Pattern classification:

Euclean distance:

$$\text{distance} = \sqrt{(\mu_1 - x_1)^2 + (\mu_2 - x_2)^2 + (\mu_3 - x_3)^2 + \cdots + (\mu_n - x_n)^2}$$

Problem Solving:

Pattern A: (x_{1A}, x_{2A}, x_{3A})

$(x'_{1A}, x'_{2A}, x'_{3A})$

$(x''_{1A}, x''_{2A}, x''_{3A})$

Pattern B:

x_{1B}, x_{2B}, x_{3B}

$x'_{1B}, x'_{2B}, x'_{3B}$

$x''_{1B}, x''_{2B}, x''_{3B}$

$x'''_{1B}, x'''_{2B}, x'''_{3B}$

classify the pattern C for data (x_{1C}, x_{2C}, x_{3C}) using Euclidean distance.

Ans:

step1: Find mean of Pattern A and B

For A:

x_{1A}, x_{2A}, x_{3A}

$x'_{1A}, x'_{2A}, x'_{3A}$

$x''_{1A}, x''_{2A}, x''_{3A}$

μ_{1A} μ_{2A} μ_{3A}

$\mu_{1A} = \frac{1}{3}(x_{1A} + x'_{1A} + x''_{1A})$

$\mu_{2A} = \frac{1}{3}($

$\mu_{3A} = \frac{1}{3}($

For B:

x_{1B}, x_{2B}, x_{3B}

$x'_{1B}, x'_{2B}, x'_{3B}$

$x''_{1B}, x''_{2B}, x''_{3B}$

$x'''_{1B}, x'''_{2B}, x'''_{3B}$

μ_{1B} μ_{2B} μ_{3B}

$\mu_{1B} = \frac{1}{4}(x_{1B} + x'_{1B} + x''_{1B} + x'''_{1B})$

$\mu_{2B} =$

$\mu_{3B} =$

step2: Calculate distance

$$dist_A = \sqrt{(\mu_{1A} - x_{1C})^2 + (\mu_{2A} - x_{2C})^2 + (\mu_{3A} - x_{3C})^2}$$

$$dist_B = \sqrt{(\mu_{1B} - x_{1C})^2 + (\mu_{2B} - x_{2C})^2 + (\mu_{3B} - x_{3C})^2}$$

step3 : Find minimum

minimum = $\min(dist_A, dist_B)$

Mahalanobis distance :

$$distance = (x-\mu)^T \Sigma^{-1} (x-\mu)$$

Problem solving :

pattern A : x_{1A}, x_{2A}, x_{3A}

$x'_{1A}, x'_{2A}, x'_{3A}$

$x''_{1A}, x''_{2A}, x'''_{3A}$

pattern B :

x_{1B}, x_{2B}, x_{3B}

$x'_{1B}, x'_{2B}, x'_{3B}$

$x''_{1B}, x''_{2B}, x''_{3B}$

$x'''_{1B}, x'''_{2B}, x'''_{3B}$

classify pattern C for data (x_{1C}, x_{2C}, x_{3C}) using Mahalanobis distance.

Ans :

step1 : Find mean for pattern A and B

Use previous step1.

A → $\mu_{1A}, \mu_{2A}, \mu_{3A}$

B → $\mu_{1B}, \mu_{2B}, \mu_{3B}$

step2 : Find covariances of pattern A and B

For A, $\Sigma_{1A} = \frac{1}{3}\left[(\mu_{1A} - x_{1A})^2 + (\mu_{1A} - x'_{1A})^2 + (\mu_{1A} - x''_{1A})^2 \right]$

$\Sigma_{2A} = \frac{1}{3}\left[(\mu_{2A} - x_{2A})^2 + (\mu_{2A} - x'_{2A})^2 + (\mu_{2A} - x''_{2A})^2 \right]$

$\Sigma_{3A} = \frac{1}{3}\left[(\mu_{3A} - x_{3A})^2 + (\mu_{3A} - x'_{3A})^2 + (\mu_{3A} - x''_{3A})^2 \right]$

For B, $\Sigma_{1B} = \frac{1}{4}\left[(\mu_{1B} - x_{1B})^2 + (\mu_{1B} - x'_{1B})^2 + (\mu_{1B} - x''_{1B})^2 + (\mu_{1B} - x'''_{1B})^2 \right]$

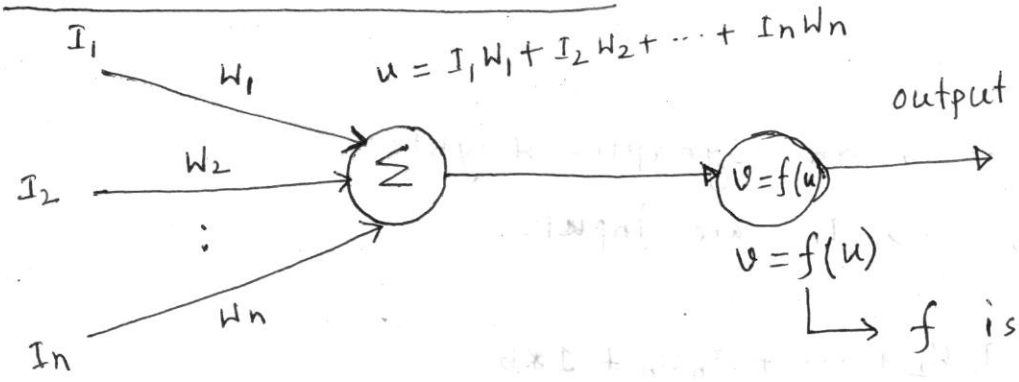
$\Sigma_{2B} =$

$\Sigma_{3B} =$

Neuro computing

- Artificial Neural Network
  - Basic unit neuron
    - Processing unit

Neuron model without bias



- f is activation function
  - sigmoid function
  - Relu
  - signum "
  - Relu (for deep Learning)
  - Piecewise linear function

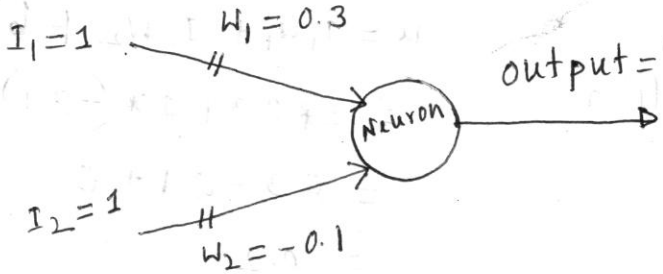
$W_1, W_2, W_3 \cdots W_n$ are synaptic weights
$I_1, I_2, I_3 \cdots I_n$ are inputs

$$u = I_1W_1 + I_2W_2 + \cdots + I_nW_n$$

$$= \sum_{i=1}^{n} I_iW_i$$

$$v = f(u)$$

For example:



$$u = I_1W_1 + I_2W_2$$
$$= 1 * 0.3 + 1 * (-0.1)$$
$$= 0.3 - 0.1$$
$$= 0.2$$

sigmoid is activation function

output, $v = \dfrac{1}{1+e^{-u}}$

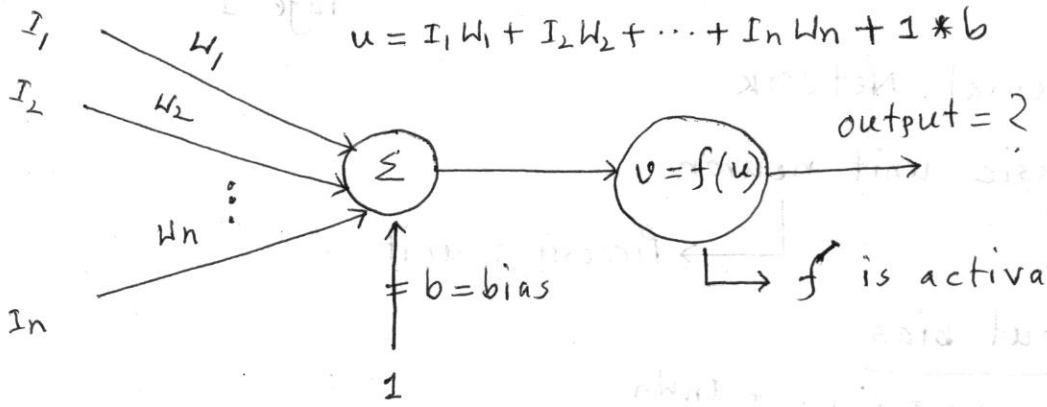
$$= \frac{1}{1+e^{-0.2}}$$

$$= \frac{1}{1+0.819} = \frac{1}{1.819}$$

~~$= 0.00122$~~

$= 0.55$

Neuron model with bias



$u = I_1 W_1 + I_2 W_2 + \cdots + I_n W_n + 1 * b$

f is activation function

$W_1, W_2, \ldots, W_n$ synaptic weights

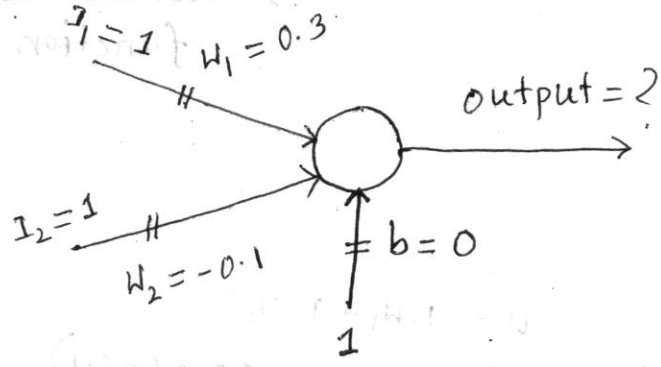
$I_1, I_2, \ldots, I_n$ are inputs

$$u = I_1 W_1 + I_2 W_2 + \cdots + I_n W_n + 1 * b$$

$$= \left(\sum_{i=1}^{n} I_i W_i \right) + b$$

$$v = f(u)$$

For example:



$u = I_1 W_1 + I_2 W_2 + b$

$= 1 * 0.3 + 1 * (-0.1) + 0$

$= 0.3 - 0.1 + 0$

$= 0.2$

sigmoid is activation function

output, $v = \dfrac{1}{1 + e^{-u}}$

$= \dfrac{1}{1 + e^{-0.2}}$

$= \dfrac{1}{1 + 0.819} = 0.55$

Ques:

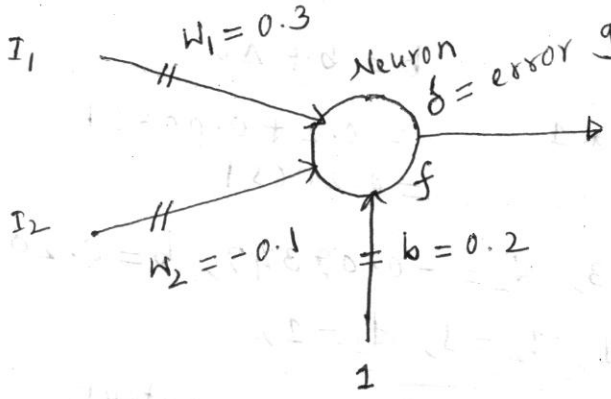
	Input		output	
	I_1	I_2		
Training data	0	1	$1 = d_1$ ←	Pattern 1
	1	1	$1 = d_2$ ←	Pattern 2
	1	0	?	Testing data

Activation function sigmoid

Learning rate, $\eta = 0.1$

Momentum coefficient, $\mu = 0.55$

Neural Network Architecture:



Complete one epoch.

Answer:

Training Model:

Using Pattern 1 ($I_1 = 0$, $I_2 = 1$, $d_1 = 1$)

Forward Pass:

- $u_1 = I_1 W_1 + I_2 W_2 + b$
  $= 0 * 0.3 + 1 * (-0.1) + 0.2$
  $= 0.1$

output, $f_1 = \frac{1}{1 + e^{-u_1}}$

$= \frac{1}{1 + e^{-0.1}}$

$= \frac{1}{1 + 0.905}$

$= \frac{1}{1.905}$

$= 0.525$

Backpropagation:

δ_1 calculation:

$\delta_1 = f_1'(d_1 - f_1)$

$= f_1(1 - f_1)(d_1 - f_1)$

$= 0.525(1 - 0.525)(1 - 0.525)$

Formula

$f_1' = \frac{d}{du_1}(f_1)$

$= f_1(1 - f_1)$

$= 0.525 * 0.475 * 0.475$

$= 0.1185$

Weight Update

$\Delta W_1 = \mu * \eta * \delta_1 * I_1$

$= 0.55 * 0.1 * 0.1185 * 0$

$= 0$

$\therefore W_1 = W_1 + \Delta W_1$

$= 0.3 + 0$

$= 0.3$

$\Delta W_2 = \mu * \eta * \delta_1 * I_2$

~~$= 0.55 * 0.1185 * 1$~~

$= 0.55 * 0.1 * 0.1185 * 1$

$= 0.00651$

$W_2 = W_2 + \Delta W_2$

$= -0.1 + 0.00651$

$= -0.09349$

$\Delta b = \mu * \eta * \delta_1 * 1$

$= 0.55 * 0.1 * 0.1185 * 1$

$= 0.00651$

$b = b + \Delta b$

$= 0.2 + 0.00651$

$= 0.20651$

New weights are $W_1 = 0.3$, $W_2 = -0.09349$, $b = 0.20651$

Using Pattern 2 ($I_1 = 1$, $I_2 = 1$, $d_2 = 1$)

Forward pass:

$u_2 = I_1 W_1 + I_2 W_2 + b$

$= 1 * 0.3 + 1 * (-0.09349) + 0.20651$

$= 0.3 - 0.09349 + 0.20651 = 0.41302$

output,

$f_2 = \frac{1}{1 + e^{-u_2}}$

$= \frac{1}{1 + e^{-0.41302}}$

$= \frac{1}{1 + 0.662} = 0.602$

$= 0.517004$ ↓

output, $f = \frac{1}{1 + e^{-u}} = 1/1.6 = 0.625$ Ans:

Backpropagation:

δ_2 calculation: $\delta_2 = f_2' (d_2 - f_2)$

$= f_2 (1 - f_2)(d_2 - f_2) = 0.602 (1 - 0.602)(1 - 0.602) = 0.0954$

Weight Update:

$\Delta W_1 = \mu * \eta * \delta_2 * I_1$

$= 0.55 * 0.1 * 0.0954 * 1 = 0.005247$

$W_1 = W_1 + \Delta W_1$

$= 0.3 + 0.005247$

$= 0.305247$

$\Delta W_2 = \mu * \eta * \delta_2 * I_2$

$= 0.55 * 0.1 * 0.0954 * 1 = 0.005247$

$W_2 = W_2 + \Delta W_2$

$= -0.09349 + 0.005247$

$= -0.088243$

$\Delta b = \mu * \eta * \delta_2 * 1 = 0.005247$

$b = b + \Delta b$

$= 0.20651 + 0.005247$

$= 0.211757$

New weights are $W_1 = 0.305247$, $W_2 = -0.088243$, $b = 0.211757$

one epoch complete

Testing

$I_1 = 1$, $I_2 = 0$

Forward pass

$u = I_1 W_1 + I_2 W_2 + b$

$= 1 * 0.305247 + 0 * (-0.088243) + 0.211757$

Multilayer Neural Network:

Ques:

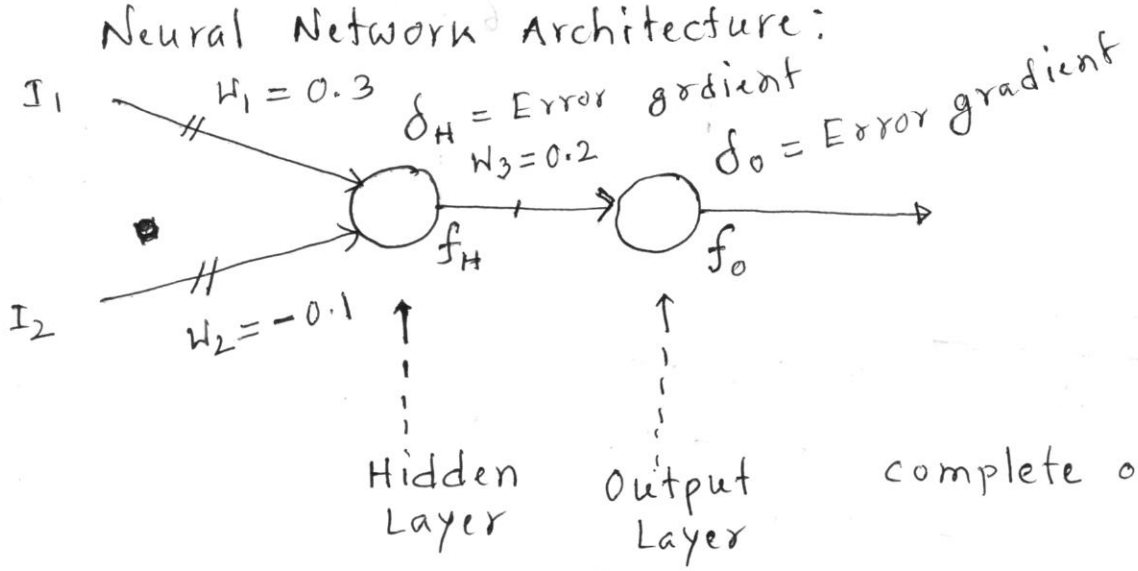
Input		Output	
I_1	I_2		
0	1	$1 = d_1$	← Pattern 1 (Training Data)
1	1	$1 = d_2$	← Pattern 2 (Training Data)
1	0	?	} Testing data

Activation function sigmoid

Learning rate, $\eta = 0.1$

Momentum coefficient, $\mu = 0.55$

Neural Network Architecture:



complete one epoch.

Answer:

Training Model:

Using Pattern 1 ($I_1 = 0$, $I_2 = 1$, $d_1 = 1$)

Forward Pass:

For Hidden Layer:

$$u_1 = I_1 W_1 + I_2 W_2$$
$$= 0 * 0.3 + 1 * (-0.1)$$
$$= -0.1$$

$$f_H = \frac{1}{1 + e^{-u_1}}$$
$$= \frac{1}{1 + e^{-(-0.1)}}$$
$$= 0.475$$

Output Layer:

$$u_1 = f_H \times W_3$$
$$= 0.475 \times 0.2$$
$$= 0.095$$

$$f_o = \frac{1}{1+e^{-u_1}}$$
$$= \frac{1}{1+e^{-0.095}}$$
$$= 0.524$$

Backpropagation:

δ calculation:

Output layer:

$$\delta_o = f_o'(d_1 - f_o)$$
$$= f_o(1-f_o)(d_1 - f_o)$$
$$= 0.524(1-0.524)(1-0.524)$$
$$= 0.524 \times 0.476 \times 0.476$$
$$= 0.11873$$

Hidden Layer:

$$\delta_H = f_H'(\delta_o \times W_3)$$
$$= f_H(1-f_H)(\delta_o \times W_3)$$
$$= 0.475 * (1-0.475) * (0.11873 \times 0.2)$$
$$= 0.475 * 0.525 * 0.023746$$
$$= 0.005922$$

Weight Update:

$$\Delta W_3 = \mu * \eta * \delta_o * f_H$$
$$= 0.55 * 0.1 * 0.11873 * 0.475$$
$$= 0.0031$$

$$W_3 = W_3 + \Delta W_3$$
$$= 0.2 + 0.0031$$
$$= 0.2031$$

$$\Delta W_1 = \mu * \eta * \delta_H * I_1$$
$$= 0.55 * 0.1 * 0.005922 * 0$$
$$= 0$$

$$W_1 = W_1 + \Delta W_1$$
$$= 0.3 + 0$$
$$= 0.3$$

$$\Delta W_2 = \mu * \eta * \delta_H * I_2$$
$$= 0.55 * 0.1 * 0.005922 * 1$$
$$= 0.000326$$

$$W_2 = W_2 + \Delta W_2$$
$$= -0.1 + 0.000326$$
$$= -0.099674$$

Updated weights are $W_1 = 0.3$, $W_2 = -0.099674$, $W_3 = 0.2031$

Using Pattern 2 ($I_1 = 1$, $I_2 = 1$, $d_2 = 1$)

Forward Pass:

For Hidden Layer

$u_2 = I_1 W_1 + I_2 W_2$

$= 1 * 0.3 + 1 * (-0.099674)$

$= 0.20033$

$f_H = \frac{1}{1 + e^{-u_2}}$

$= \frac{1}{1 + e^{-0.20033}}$

$= 0.55$

For Output Layer

$u_2 = f_H \times W_3$

$= 0.55 \times 0.2031$

$= 0.111705$

$f_o = \frac{1}{1 + e^{-u_2}}$

$= \frac{1}{1 + e^{-0.111705}}$

$= 0.53$

Back Propagation:

δ calculation:

Output layer:

$\delta_o = f_o'(d_2 - f_o)$

$= f_o(1 - f_o)(d_2 - f_o)$

$= 0.53(1 - 0.53)(1 - 0.53)$

$= 0.117077$

Hidden Layer:

$\delta_H = f_H'(\delta_o \times W_3)$

$= f_H(1 - f_H)(\delta_o \times W_3)$

$= 0.55(1 - 0.55)(0.117077 * 0.2031)$

$= 0.00589$

Weight Update:

$\Delta W_3 = \mu * \eta * \delta_o * f_H$

$= 0.55 * 0.1 * 0.117077 * 0.55$

$= 0.00644$

$W_3 = W_3 + \Delta W_3$

$= 0.2031 + 0.00644$

$= 0.20954$

$\Delta W_1 = \mu * \eta * \delta_H * I_1$

$= 0.55 * 0.1 * 0.00589 * 1 = 0.000324$

$W_1 = W_1 + \Delta W_1$

$= 0.3 + 0.000324$

$= 0.300324$

$\Delta W_2 = \mu * \eta * \delta_H * I_2$

$= 0.55 * 0.1 * 0.00589 * 1 = 0.000324$

$W_2 = W_2 + \Delta W_2$

$= -0.099674 + 0.000324$

$= -0.09935$

Updated weights are $W_1 = 0.300324$ $W_2 = -0.09935$, $W_3 = 0.20954$

One epoch completed

Testing the Model

$I_1 = 1$, $I_2 = 0$ output = ?

Forward Pass:

For Hidden Layer:

$u = I_1 W_1 + I_2 W_2$

$= 1 * 0.300324 + 0 * (-0.09935)$

$= 0.300324$

$f_H = \frac{1}{1 + e^{-u}}$

$= \frac{1}{1 + e^{-0.300324}}$

$= 0.5744$

For Output Layer:

$u = f_H \times W_3$

$= 0.5744 \times 0.20954$

$= 0.12036$

$f_o = \frac{1}{1 + e^{-u}}$

$= \frac{1}{1 + e^{-0.12036}}$

$= 0.53$

If threshold is 0.5

Since $f_o \geq$ threshold

$\Rightarrow 0.53 \geq 0.5$, output = 1

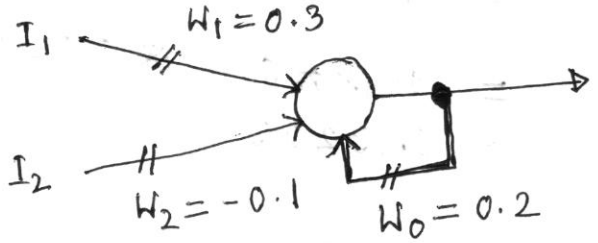
$\therefore$ When $I_1 = 1$, $I_2 = 0$, output = 1 Ans:

Recurrent Neural Network (RNN)

Ques:

	Input		Output
	I_1	I_2	
Training Data	0	1	$1 = d_1$ ← Pattern 1 (time 0)
	1	1	$1 = d_2$ ← Pattern 2 (time 1)
Testing data	1	0	?

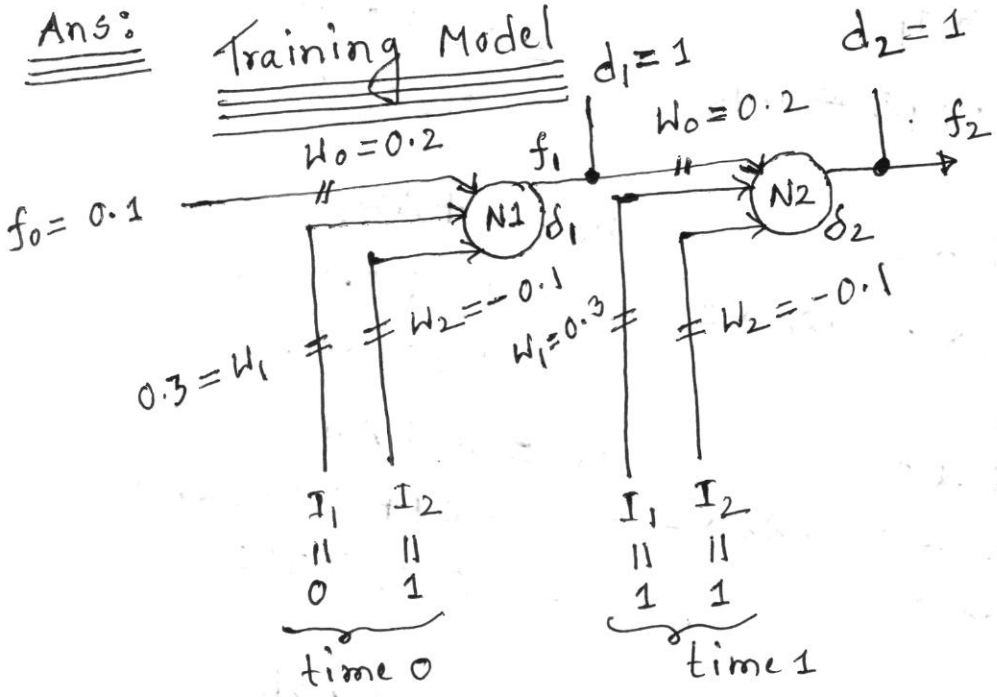
Activation function sigmoid
Learning rate, $\eta = 0.1$
Momentum coefficient, $\mu = 0.55$
Neural Network Architecture:



Assume initial value of output is 0.1

Ans:

Training Model



Forward Pass

For N1 :

$u_1 = 0.1 * W_0 + I_1 W_1 + I_2 W_2$

$= 0.1 * 0.2 + 0 * 0.3 + 1 * (-0.1)$

$= -0.08$

$$f_1 = \frac{1}{1 + e^{-u_1}}$$

$$= \frac{1}{1 + e^{-(-0.08)}}$$

$= 0.48$

For N2:

$u_2 = f_1 * W_0 + I_1 W_1 + I_2 W_2$

$= 0.48 * 0.2 + 1 * 0.3 + 1 * (-0.1)$

$= 0.296$

$f_2 = \frac{1}{1+e^{-u_2}}$

$= \frac{1}{1+e^{-0.296}}$

$= 0.744$

Backpropagation through time

δ calculation: For N2

$\delta_2 = f_2'(d_2 - f_2)$

$= f_2(1-f_2)(d_2-f_2)$

$= 0.744(1-0.744)(1-0.744)$

$= 0.0488$

For N1

$\delta_1 = f_1'(d_1 - f_1 + \delta_2 W_0)$

$= f_1(1-f_1)(d_1 - f_1 + \delta_2 W_0)$

$= 0.48(1-0.48)(1-0.48+0.0488 * 0.2)$

$= 0.48 * 0.52 * 0.52976$

$= 0.13223$

Weight Update:

For N2

$\Delta W_0 = \mu * \eta * \delta_2 * f_1$

$= 0.55 * 0.1 * 0.0488 * 0.48$

$= 0.00129$

$W_0 = W_0 + \Delta W_0$

$= 0.2 + 0.00129$

$= 0.20129$

$\Delta W_1 = \mu * \eta * \delta_2 * I_1$

$= 0.55 * 0.1 * 0.0488 * 1$

$= 0.002684$

$W_1 = W_1 + \Delta W_1$

$= 0.3 + 0.002684$

$= 0.302684$

$\Delta W_2 = \mu * \eta * \delta_2 * I_2$

$= 0.55 * 0.1 * 0.0488 * 1$

$= 0.002684$

$W_2 = W_2 + \Delta W_2$

$= -0.1 + 0.002684$

$= -0.097316$

For N1

$\Delta W_0 = \mu * \eta * \delta_1 * f_0$

$= 0.55 * 0.1 * 0.13223 * 0.1$

$= 0.00073$

$W_0 = W_0 + \Delta W_0$

$= 0.20129 + 0.00073$

$= 0.20202$

$\Delta W_1 = \mu * \eta * \delta_1 * I_1$

$= 0.55 * 0.1 * 0.13223 * 0$

$= 0$

$W_1 = W_1 + \Delta W_1$

$= 0.302684 + 0$

$= 0.302684$

$\Delta W_2 = \mu * \eta * \delta_1 * I_2$

$= 0.55 * 0.1 * 0.13223 * 1$

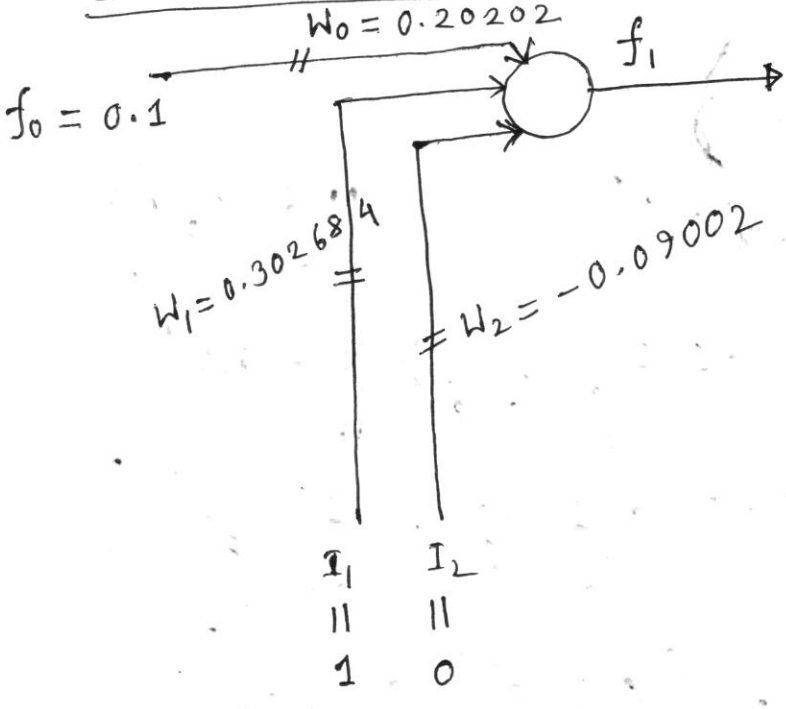
$= 0.0073$

$W_2 = W_2 + \Delta W_2$

$= -0.097316 + 0.0073$

$= -0.09002$

Testing the Model



Forward Pass

$u = f_0 W_0 + I_1 W_1 + I_2 W_2$

$= 0.1 * 0.20202 + 1 * 0.302684 + 0 * (-0.09002)$

$= 0.020202 + 0.302684 + 0$

$= 0.322886$

$$f_1 = \frac{1}{1+e^{-u}} = \frac{1}{1+e^{-0.322886}} = 0.58$$

If threshold is 0.5,

Since $f_o \geq$ threshold

$\Rightarrow 0.58 \geq 0.5$, output = 1

When $I_1 = 1$, $I_2 = 0$ output = 1 Ans:

Why $f' = f(1-f)$ when $f = \frac{1}{1+e^{-x}}$

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

$$\frac{df}{dx} = \frac{d}{dx}\left(\frac{1}{1+e^{-x}}\right)$$

$$\Rightarrow f' = \frac{d}{dx}\left\{(1+e^{-x})^{-1}\right\} \quad \left\| \frac{d}{dx}(x^n) = nx^{n-1} \right.$$

$$= -1 \times (1+e^{-x})^{-1-1} \frac{d}{dx}(1+e^{-x})$$

$$= -1 \times (1+e^{-x})^{-2} \left\{\frac{d}{dx}(1) + \frac{d}{dx}(e^{-x})\right\}$$

$$= -1 \times \frac{1}{(1+e^{-x})^2} \left\{0 + e^{-x} \frac{d}{dx}(-x)\right\}$$

$$= -\frac{1}{(1+e^{-x})^2} \left\{0 + e^{-x} * (-1)\right\}$$

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

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

$$= f(1-f)$$

GRNN: Generalized Regression Neural Networks

Input	Output
2	3
4	5
6	7
8	9

What will be the output of 5?

Ans:

Step 1:

$$d_1 = (5-2)^2 = 9$$

$$d_2 = (5-4)^2 = 1$$

$$d_3 = (5-6)^2 = 1$$

$$d_4 = (5-8)^2 = 9$$

Step 2:

Calculate weights using the activation function:

$$e^{-\frac{d_i^2}{2\sigma^2}}$$

Assume, $\sigma = 1$

$$\therefore W_1 = e^{-\frac{d_1^2}{2}} = 0.01$$

$$W_2 = e^{-\frac{d_2^2}{2}} = 0.6$$

$$W_3 = e^{-\frac{d_3^2}{2}} = 0.6$$

$$W_4 = e^{-\frac{d_4^2}{2}} = 0.01$$

Step 3:

Summation of $W_i = \sum W_i = W_1 + W_2 + W_3 + W_4 = 1.22$

So, denominator is 1.22

Numerator $= YW = W_1 * y_1 + W_2 y_2 + W_3 y_3 + W_4 y_4$

$= 0.01 * 3 + 0.6 * 5 + 0.6 * 7 + 0.01 * 9$

$= 7.32$

Step 4:

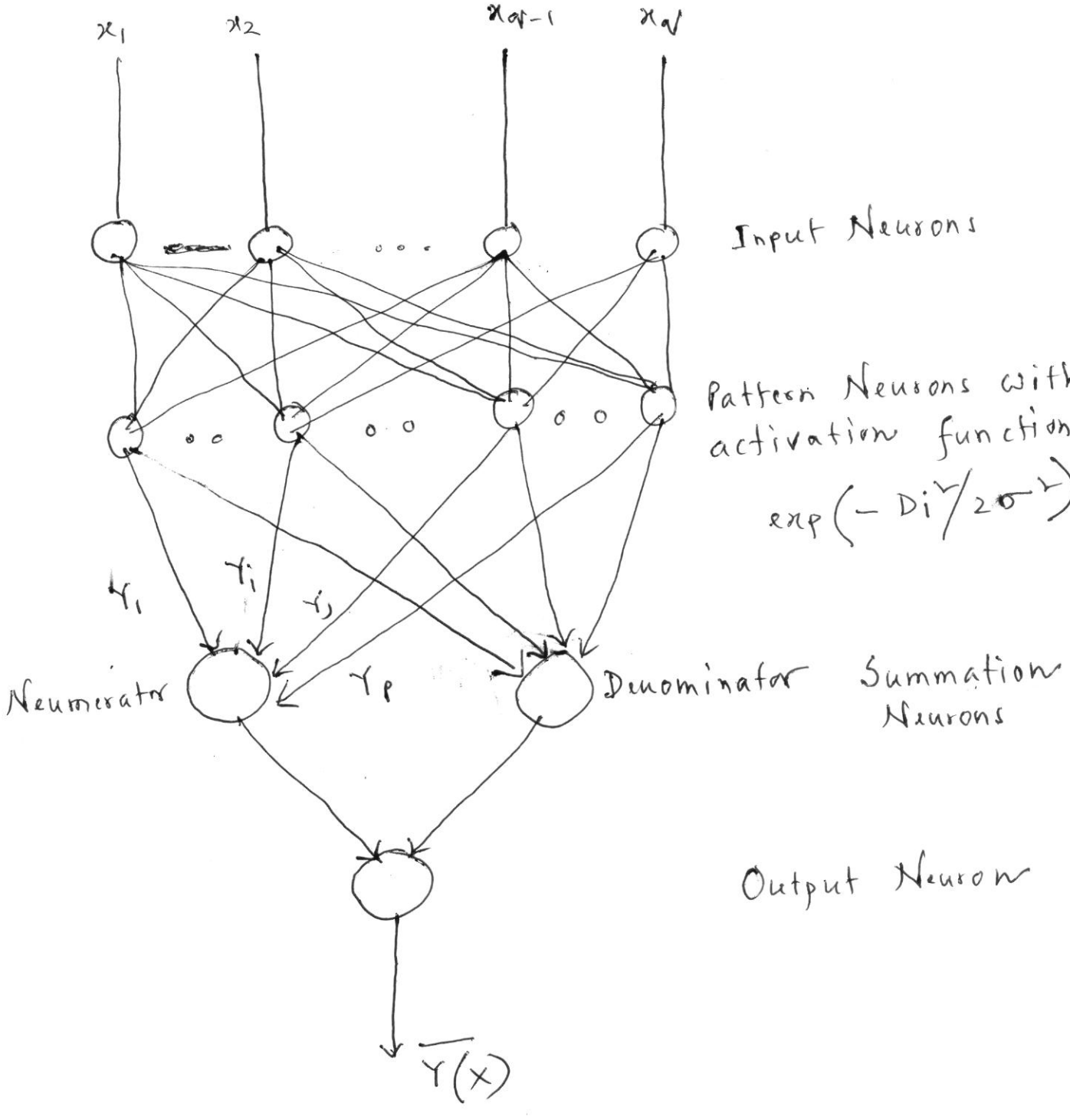
So the output is : $= \frac{\text{Neumerator}}{\text{Denominator}}$

$$= \frac{YW}{W}$$

$$= \frac{7.32}{1.22}$$

$$= 6$$

So predicted output is 6



Input layer:

Input layer feeds the input to the next layer

Pattern layer:

Pattern layer calculates the Euclidean distance and activation function.

Summation layer:

Summation layer:

- Numerator Part (Training output * activation function) sumation
- Dinominator Part (summation of all activation function)

This layer feeds both the Numerator and Denominator to the next output layer.

Output layer:

$$\text{Output} = \frac{\text{Numerator}}{\text{Denominator}}$$

$$= \frac{\sum Y_i e^{-\frac{d_i^2}{2\sigma^2}}}{\sum e^{-\frac{d_i^2}{2\sigma^2}}}$$

where $d_i^2 = (x - x_i)^T (x - x_i)$