ARTIFICIAL NEURAL NETWRK

1. Introduction of ANN
2. Perceptrons and its applications
3. Backpropagation and its applications

Indra Deo RAM
Dept of Comp Sc
BIT Mesra, Ranchi

WHAT IS NEURAL NET

- Neural Nets are named after the cell in the human brain that perform intelligent operations

- The brain is maid up of billions of neuron cells.Each of these cells is like a tiny computer with extremely limited capabilities; however , connected together , these cells form the most intelligence system.

- Neural Nets are formed from hundreds or thousands of simulated neurons connected together in much the same way as the brain’ neurons

WHAT IS NEURAL NET

- The key element of this paradigm is the novel structure of the information processing system

- It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems

- Learning in biological systems involves adjustment to the synaptic connections that exist between the neurons. This is true of ANNs as well.

Contd.

CHARACTERISTICS OF ANNs

ANNs exhibit a surprising no. of the brain's characteristics . E.g.

Learning :

– Human brain has ability to learn from experience.Similarly ANNs can modify their behavior in response to their environment. For set of I/Ps(perhaps with desired O/P), they self-adjust to produce consistent responses.This depends what things a network can be trained to do , and how the training should be performed.

Generalization :

– As human brain has ability to generalize the knowledge it possess, similarly NNs can also generalize from previous example to new ones. Once the networks has been trained to a degree and we are giving partially incorrect input in that case even network can give correct output automatically.This reflect that it has ability to see through noise and distortion to pattern that lies within.

CHARACTERISTICS OF ANNs

Abstraction:

– Some ANNs are capable of abstracting the essence of a set of inputs. For example , a network can be trained on a sequence of distorted versions of the letter A. after adequate training, application of such a distorted example will cause the network to produce a perfectly formed letter. In some sense it has learned to produce some thing that it has never seen before.

Error:

– As human make error, similarly neural networks also make errors.

Contd.

APPLICATION OF ANNS

- Aerospace
  - High performance aircraft autopilots, flight path simulations, aircraft control systems, autopilot enhancements, aircraft component simulations, aircraft component fault detectors
- Automotive
  - Automobile automatic guidance systems, warranty activity analyzes
- Banking
  - Check and other document readers, credit application evaluators
- Defense
  - Weapon steering, target tracking, object discrimination, facial recognition, new kinds of sensors, sonar, radar and image signal processing including data compression, feature extraction and noise suppression, signal/image identification
- Electronics
  - Code sequence prediction, integrated circuit chip layout, process control, chip failure analysis, machine vision, voice synthesis, nonlinear modeling

APPLICATION OF ANNS

- Financial
  - Real estate appraisal, loan advisor, mortgage screening, corporate bond rating, credit line use analysis, portfolio trading program, corporate financial analysis, currency price prediction
- Manufacturing
  - Manufacturing process control, product design and analysis, process and machine diagnosis, real-time particle identification, visual quality inspection systems, beer testing, welding quality analysis, paper quality prediction, computer chip quality analysis, analysis of grinding operations, chemical product design analysis, machine maintenance analysis, project bidding, planning and management, dynamic modeling of chemical process systems
- Medical
  - Breast cancer cell analysis, EEG and ECG analysis, prosthesis design, optimization of transplant times, hospital expense education, hospital quality improvement, emergency room test advisement

Contd.

APPLICATION OF ANNS

- **Robotics**
  - Trajectory control, forklift robot, manipulator controllers, vision systems
- **Speech**
  - Speech recognition, speech compression, vowel classification, text to speech synthesis
- **Securities**
  - Market analysis, automatic bond rating, stock trading advisory systems
- **Telecommunications**
  - Image and data compression, automated information services, real-time translation of spoken language, customer payment processing systems
- **Transportation**
  - Truck brake diagnosis systems, vehicle scheduling, routing systems

Contd

COMPARISON OF COMPUTING APPROACH

CHARACTERISTICS	TRADITIONAL COMPUTING (including Expert system)	ARTIFICIAL NEURAL NETWORK
Processing style Functions	Sequential Logically (left brained) via Rules concepts Calculation	Parallel Gestalt (right brained) Via Images Pictures Controls
Learning method applications	By rules (didactically) Accounting word processing math inventory digital communications	By example (Socratic ally) sensor processing speech processing pattern recognition text recognition

COMPARISON OF EXPERT SYSTEM AND NEURAL NETWORK

Characteristics	Von Neumann Architecture used for expert system	Artificial Neural Networks
Processors	VLSI(traditional processors)	ANNs; variety of Technologies; H/w development is on going
Processing Approach	Processes problem rule at a one time; sequential	Multiple, Simultaneous
Connections	Externally programmable	Dynamically self Programmable
Self Learning	Only algorithmic parameters modified	Continuously adaptable

COMPARISON OF EXPERT SYSTEM AND NEURAL NETWORK

Characteristics	Von Neumann Architecture used for expert system	Artificial Neural Networks
Fault tolerance	Non without special Processors	Significant in the very nature of the interconnected neurons
Programming	Through a rule based	Self-programming
Ability to be fast	Require big processors	Require multiple custom-built chips

Contd.

HISTORICAL BACKGROUND

- Neural network simulation appear to be a recent development. However , this field was established before the advent of computer, and has survived at least one major setback and several eras.

- The first artificial neuron was produced in 1943 by the neurophysiologist Warren McCulloch and the logician Walter Pits.But the technology available at that time did not allow them to do too much

Historical Sketch

- Pre-1940: von Hemholtz, Mach, Pavlov, etc.
  - No specific mathematical models of neuron operation
  - General theories of learning, vision, conditioning
- 1940s: Hebb, McCulloch and Pitts
  - Mechanism for learning in biological neurons
  - Neural-like networks can compute any arithmetic function
- 1950s: Rosenblatt, Widrow and Hoff
  - First practical networks and learning rules
- 1960s: Minsky and Papert
  - Demonstrated limitations of existing neural networks, new learning algorithms are not forthcoming, some research suspended
- 1970s: Amari, Anderson, Fukushima, Grossberg, Kohonen
  - Progress continues, although at a slower pace
- 1980s: Grossberg, Hopfield, Kohonen, Rumelhart, etc.
  - Important new developments cause a resurgence in the field

Contd.

Artificial neural networks

Analogy to Brain

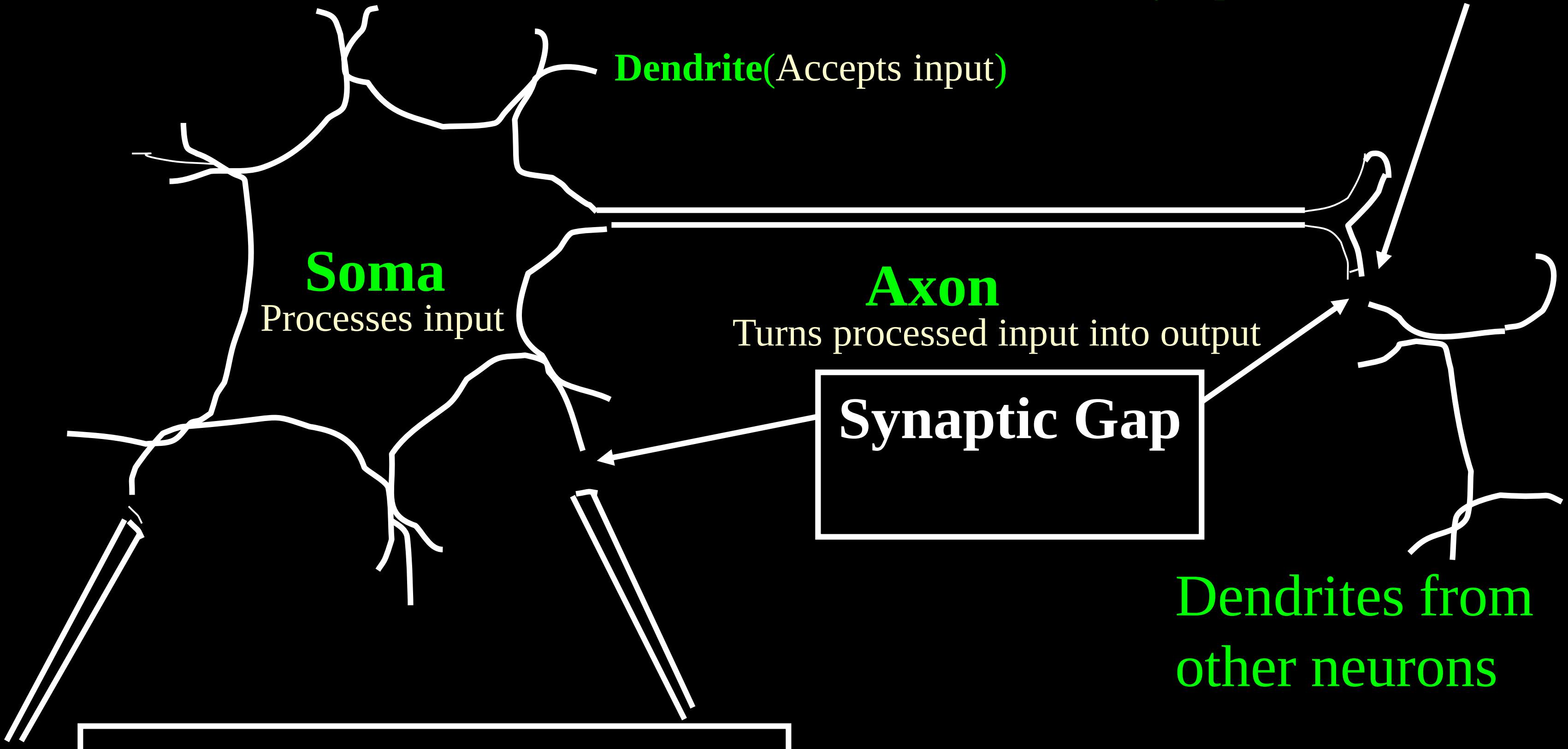
Human Brain :

- Contains approx. 10^{11} neurons
- Each neuron is connected to approx. 10^{4} others
- Neurons have real-valued input and output
- Activity is inhibited/excited via connections to others.
- processing operations result from highly parallel processes, distributed over many neurons.

Contd.

Artificial neural networks

Biological neuron



Synapse (Contact between neurons)

Dendrite(Accepts input)

Soma
Processes input

Axon
Turns processed input into output

Synaptic Gap

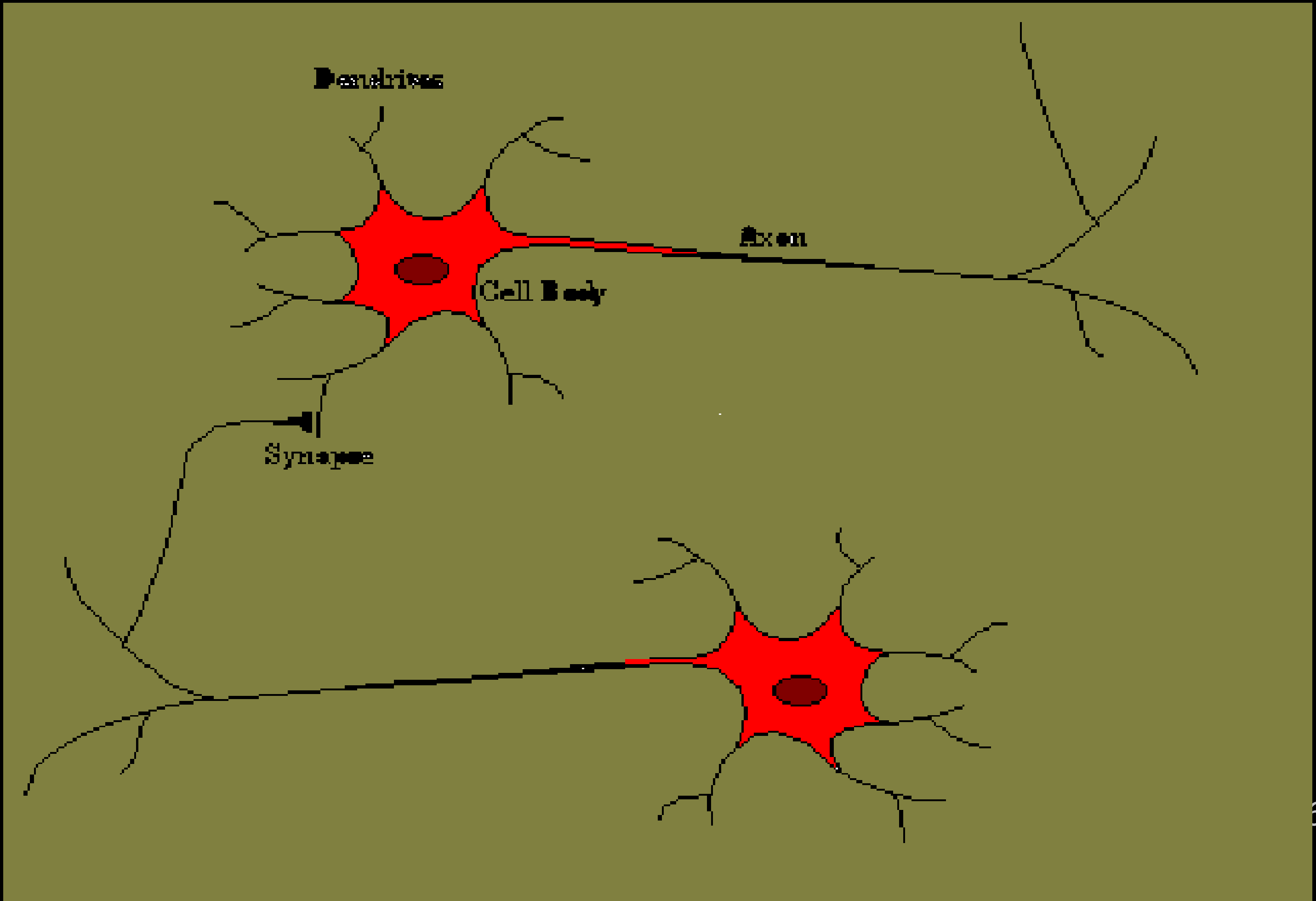
Dendrites from other neurons

Axons from other neurons

Contd.

Artificial neural networks

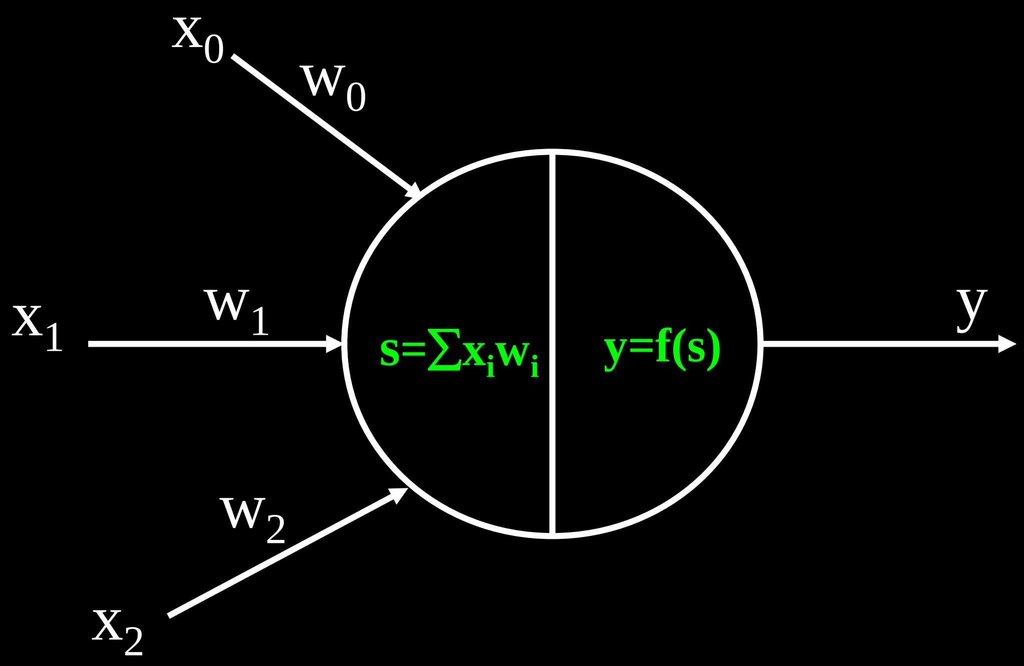
- Neurons respond slowly
  - – 10^{-3} s compared to 10^{-9} s for electrical circuits



Contd

Artificial neural networks

Basic Artificial neuron



x_0 , x_1 , x_2 : Inputs

W_0 , W_1, W_2 :Weights

S - Weighted sum

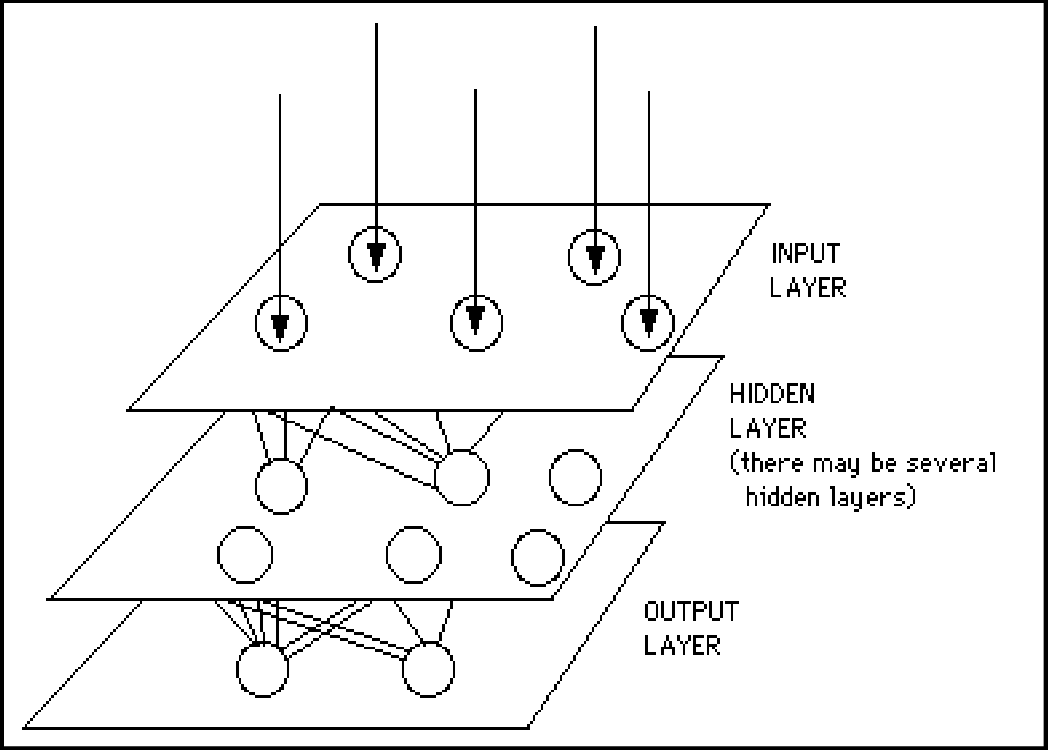
Y – Output

f – Activtion function

Contd.

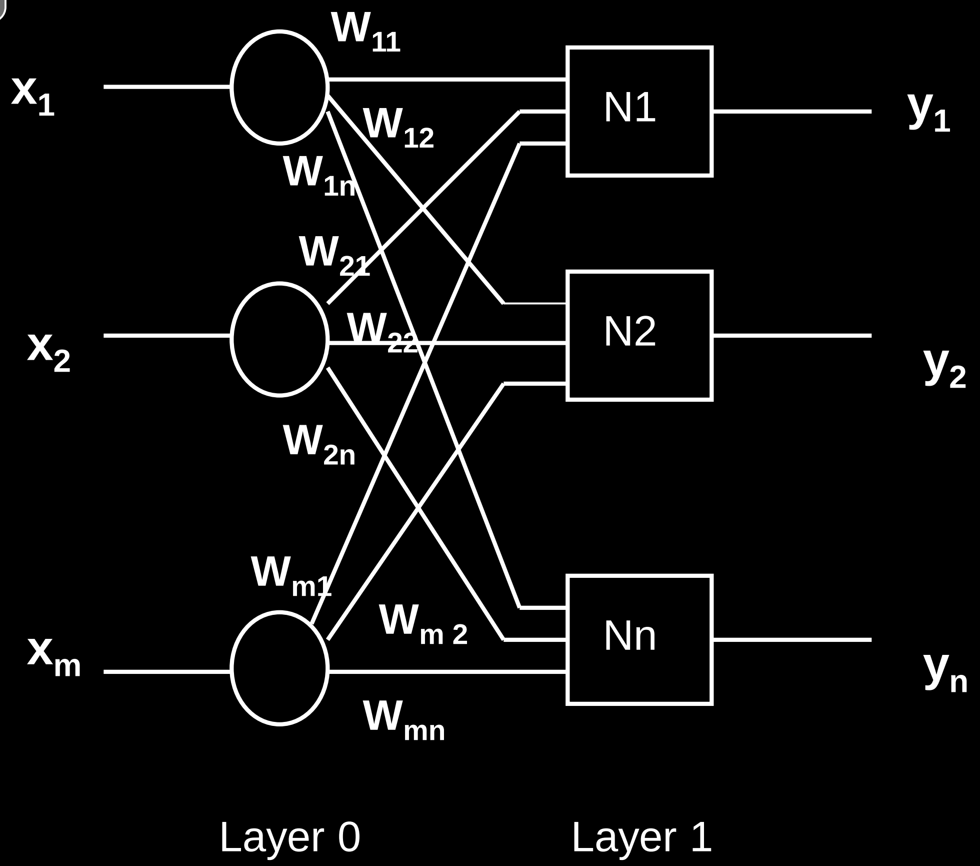
Artificial neural networks

A Simple Neural Network Diagram



Contd.

Artificial neural networks



Layer 0- Distribution layer

Does no processing

Input vector - X

X =[x1,x2,…. ,xm]

Weight matrix – W of order

mXn

m-No. of inputs

n-No. of neurons

N Neurons

For each neuron NET = XW

Y- Output vector

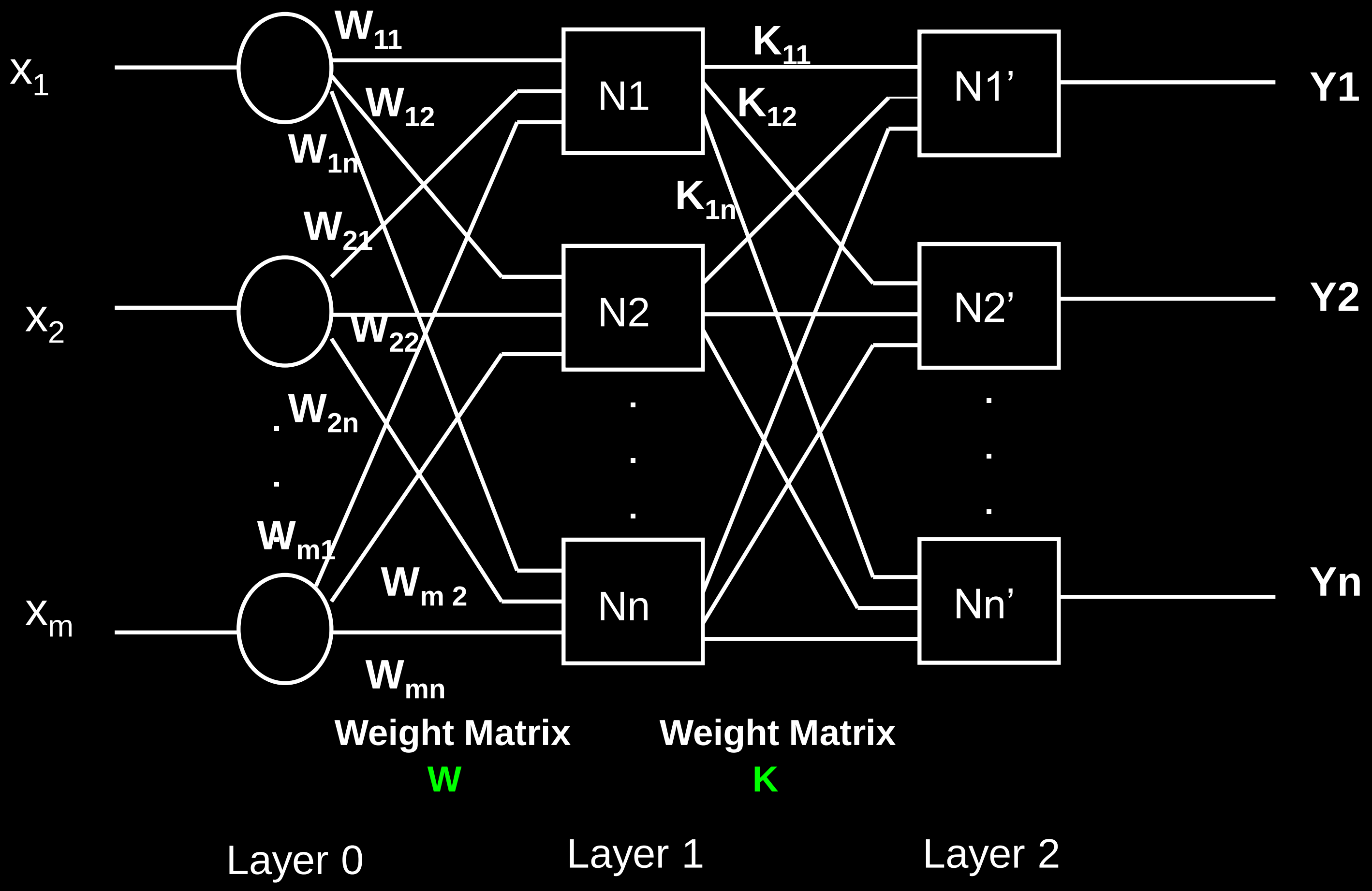
Y=F(NET)

Single layer neural networks

Suitable only for linearly separable problems.

Contd.

Artificial neural networks



Two-layer neural network

Contd.

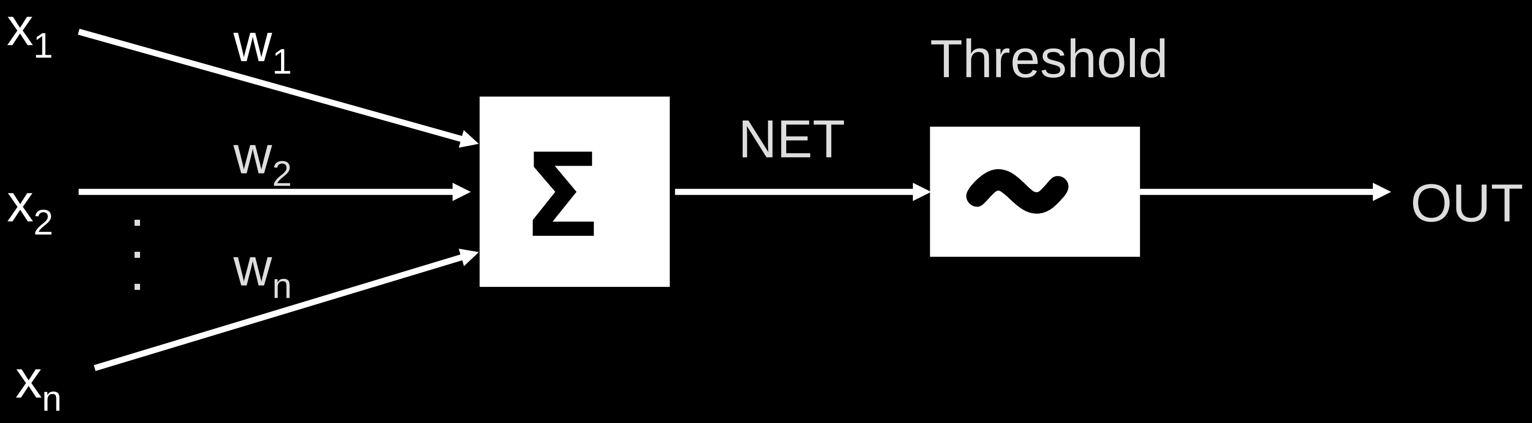
Artificial neural networks

Connections in Neurons:

- Inter-layer connections
- Intra-layer connections

PERCEPTRON

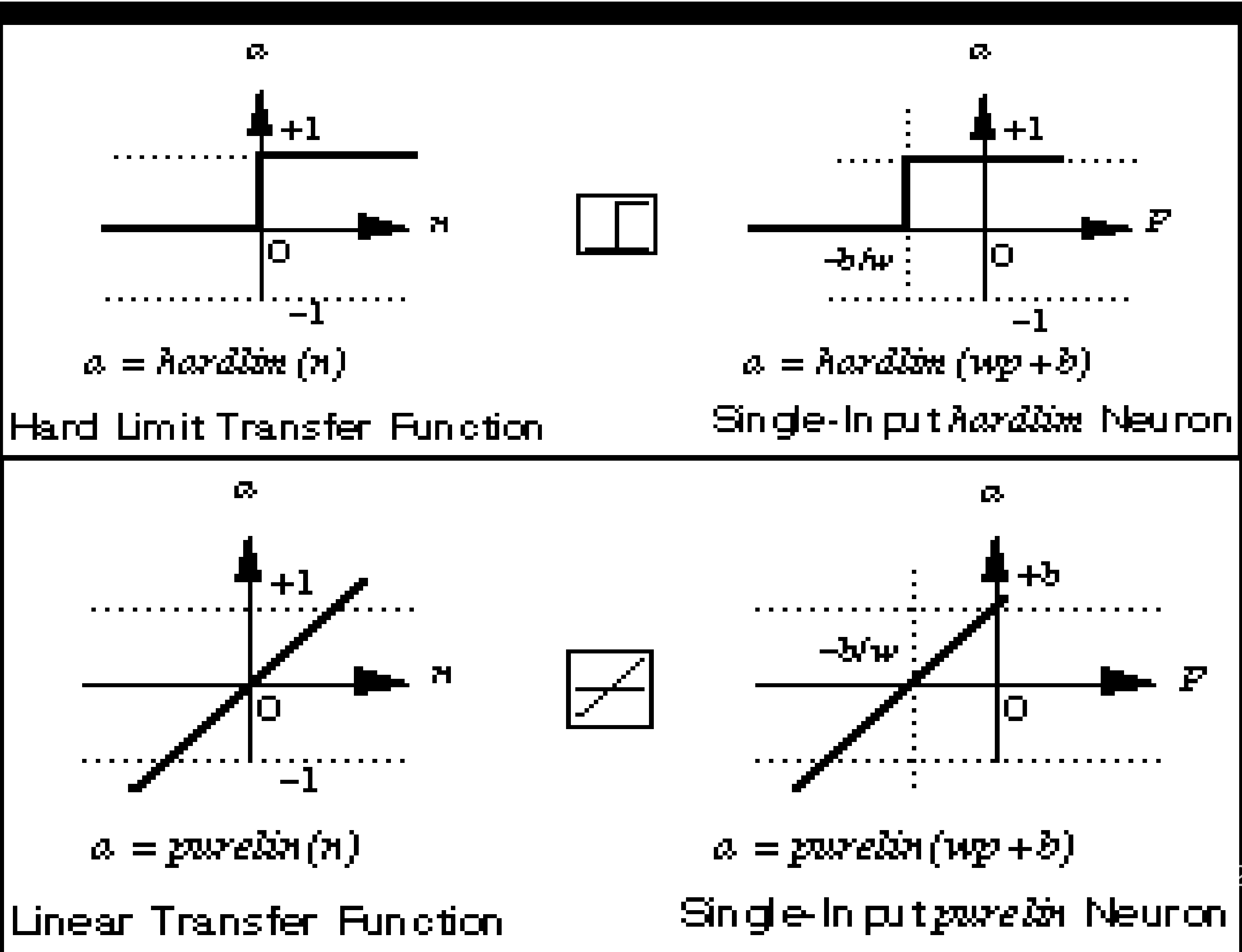
- **Marwin minsky, Frank, Bernard, and others developed networks consisting of a single layer of artificial neurons often called Perceptrons.They are used for weather prediction,Electrocardiogram analysis, and artificial vision i.e..pattern recognition.**



$$NET = X_1W_1+X_2W_2+....+X_nW_n$$

$$NET = XW^T$$

Transfer Functions



$a = \mathrm{hardlim}(n)$

Hard Limit Transfer Function

$a = \mathrm{hardlim}(wp + b)$

Single-Input *hardlim* Neuron

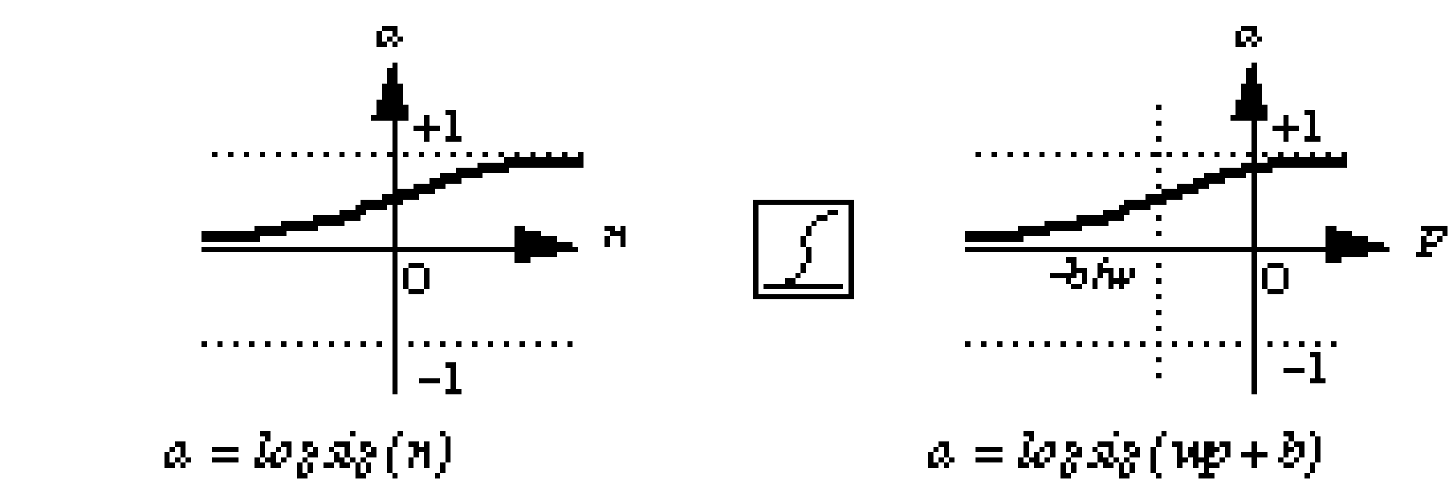
$a = \mathrm{purelin}(n)$

Linear Transfer Function

$a = \mathrm{purelin}(wp + b)$

Single-Input *purelin* Neuron

Transfer Functions



$a = \mathit{logsig}(n)$ $\qquad$ $a = \mathit{logsig}(wp + b)$

Log-Sigmoid Transfer Function $\qquad$ Single-Input *logsig* Neuron

Contd.

PERCEPTRON FEATURES

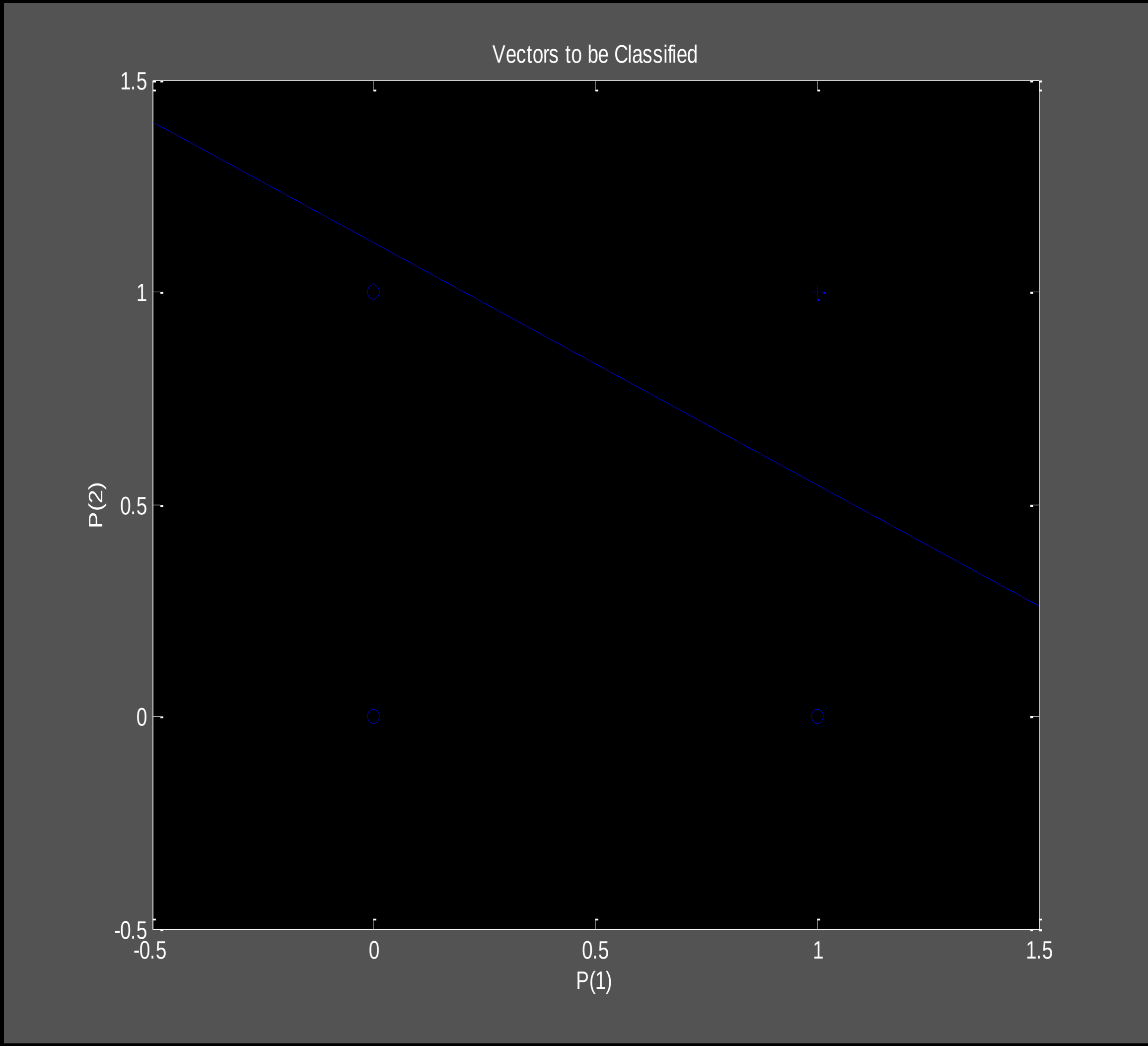
- **The Perceptron has shown itself worthy of study despite its severe limitations. It has many features that attract attention**
  - **Its linearity**
  - **Its intriguing learning theorem**
  - **Its clear paradigmatic simplicity as a kind of parallel computation** Etc.

LINEAR SEPARABILITY

- Linear separability limits single-layer networks to classification problems in which the sets of points(corresponding to the I/p values can be separated geometrically.

- For two I/p case, the separator is a straight line

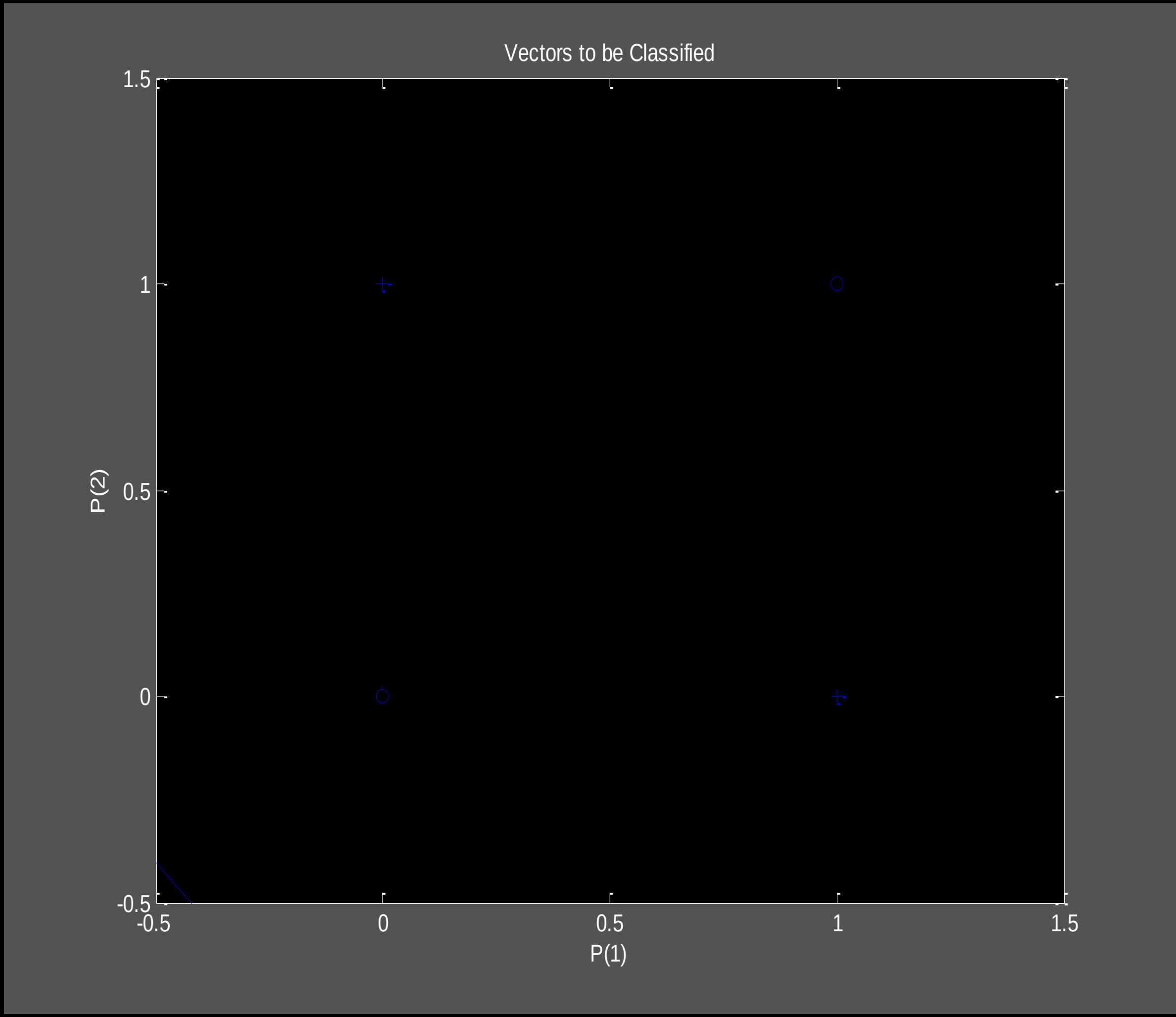
- For three I/ps , a flat plane cutting through the resulting 3-D space.



PERCEPTRON LIMITATIONS

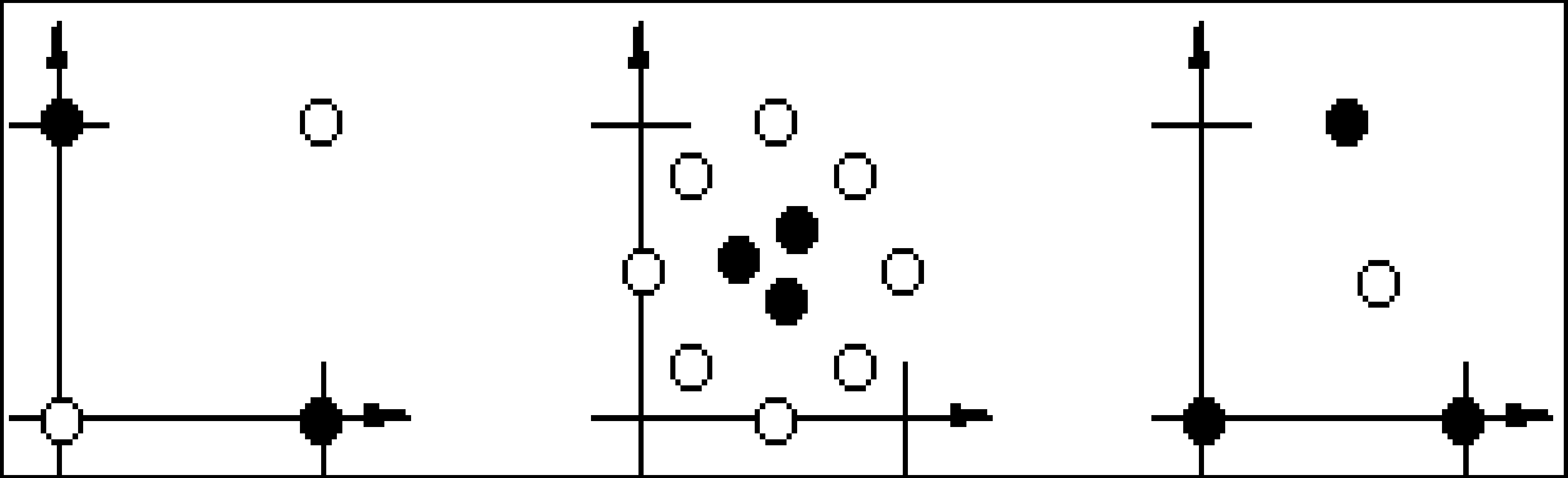
- There exists a large class of functions that can’t be represented by a single-layered network.

- These functions are said to be linear inseparable, and they set definite bounds on the capabilities of single layer networks



PERCEPTRON LIMITATIONS

- Linearly Inseparable Problems



Training A Network

Learning:

The process of adjusting the weights to make the network model the relationship between input and target is called learning or training.

Objective of training :

A network is trained so that application of a set of inputs produces the desired (or at least consistent) set of outputs

Types of learning:

Supervised learning

Unsupervised learning/Adaptive learning

Contd.

Training A Network

Supervised training :

- Both inputs and target outputs are provided to the network.
- Network produces output based on the given inputs.
- Outputs are compared against the desired outputs.
- Errors are then propagated back through the system.
- System adjusts the weights which control the network.

Contd.

Training A Network

Unsupervised training :

- The network is provided with only the inputs.
- Desired outputs are not provided.
- System decides itself what features it will use to group input data.
- This is called self-organization or adoption.

Learning laws

Mathematical algorithms used to update the connection weights.

Various laws :

- Hebb's Rule
- The Delta Rule
- The Gradient Descent Rule
- Hopfield Law
- Kohonen's Learning Law

Contd.

Learning laws

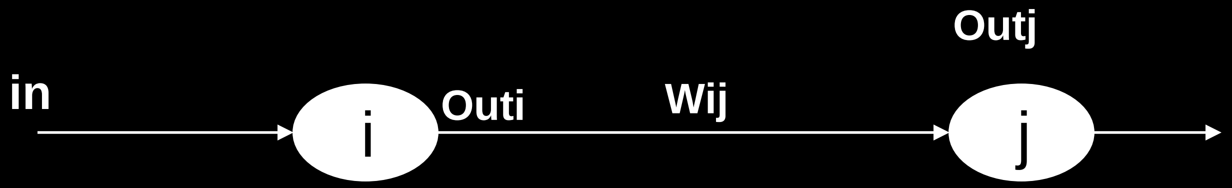
Hebb’s Rule :

Proposed by D.O.Hebb

For unsupervised learning

Statement :

Synaptic strength should be increased if both the source and destination neurons are activated.



Weight change formula **$Wij(n+1)=wij(n) + \alpha$ outi outj**

wij(n) – weight prior to adjustment from neuron i to neuron j

Wij(n+1) - weight after adjustment from neuron i to neuron j

Contd.

Perceptron Training Algo

- The training method can be summarized as :

  1. Apply an input pattern X and calculate the o/p Y

  2. a. If the o/p is correct, go to step 1

     b. If the o/p incorrect, and is zero, add each I/p to its corresponding weight; or

     c. If the o/p is incorrect and one, subtract each I/p from its corresponding weight.

  3. Go to step 1

Learning laws

- An important generalization of the Perceptron training algorithm, called the delta rule.
- For generalization consider a term δ, which is difference between desired or target output **T** and the actual o/p **A**.

$$\delta = T - A$$

- The case in which **$\delta=0$** corresponds to step 2a, in which output is correct and nothing is done. Step 2b corresponds to **$\delta>0$** while step 2c corresponds to **$\delta<0$.**
- To generalize this the learning rate coefficient η multiplies the δx_i product to allow control of the average size of weight changes

  $\triangle_i = \eta\delta x_i$ where $\triangle_i$ is correction associated with i^{th} I/p

  $W_i(n+1) = W_i(n) + \triangle_i$

Learning laws

Delta Rule :

Variation of Hebb’s rule.

- Based on idea of continuously modifying the strength of input connections to reduce the difference between desired output and actual output of a neuron
- Delta which is error actually is transformed by the derivative of transfer function and is then used in the previous neural layer to adjust input connection weights.
- This error is backpropagated into the previous layers one layer at a time.

Contd.

Example

$x_1 = [1\ \ -2\ 0\ 1]$ $\qquad x_2 = [0\ \ 1.5\ \ -0.5\ \ -1]$

$x_3 = [-1\ 1\ 0.5\ \ -1]$ $\qquad w_1 = [1\ \ -1\ \ 0\ 0.5]$

$d_1 = -1, d_2 = -1, d_3 = 1$

$\gamma = 0.1$

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

Backpropagation

Backpropagation

- ✓ Supervised learning algorithm.
- ✓ Back propagation algorithm is used to train multilayered feed-forward networks.
- ✓ It uses delta rule to calculate the error between desired and actual output values.

King of Neural networks : because of its

Ease of use

Wide applicability

Contd.

Backpropagation

Training the network

Back propagation network requires following steps :

1) Apply input vector to the network

2) Calculate the output of the network

3) Calculate the error between the network output and desired output

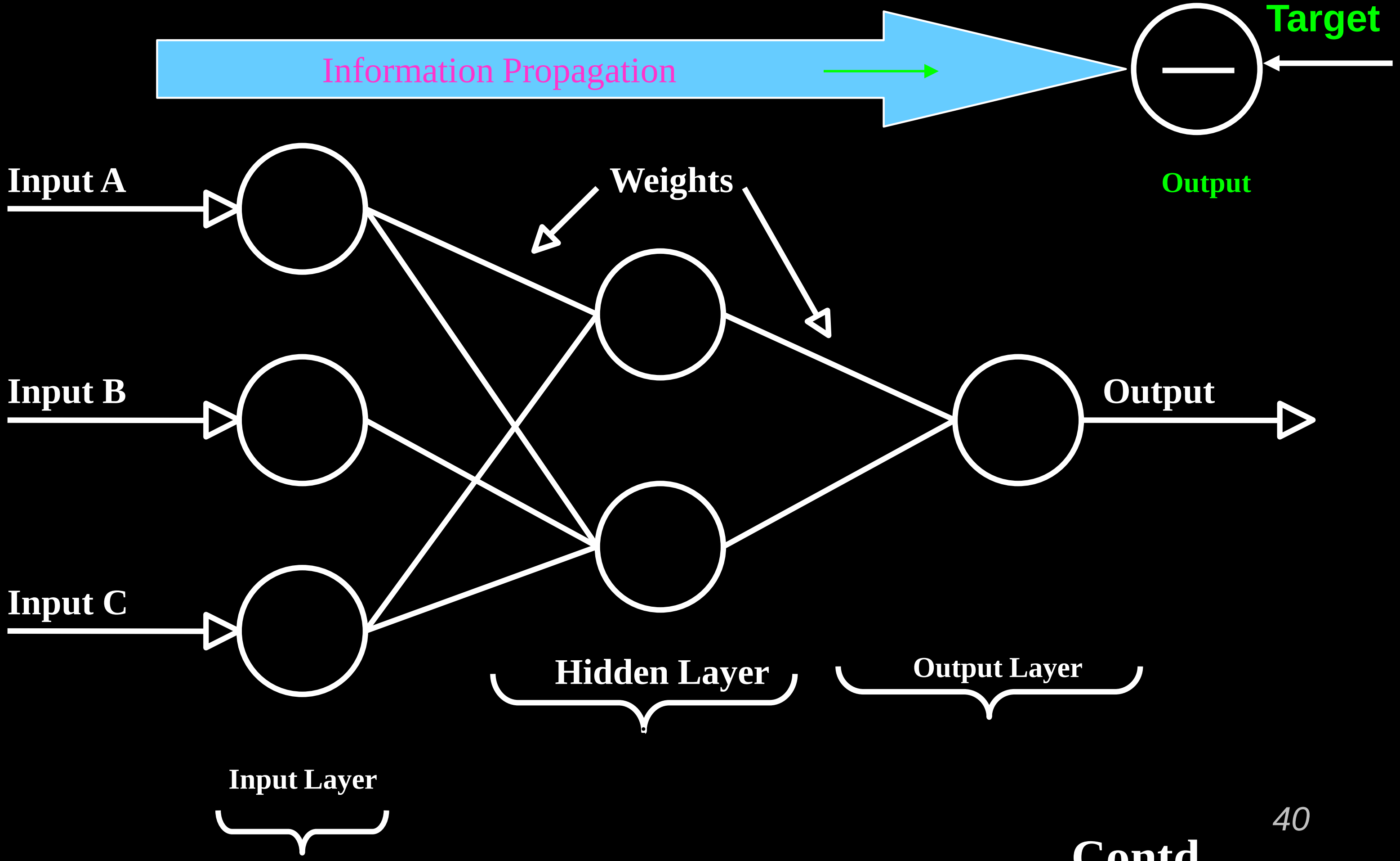
4) Adjust the weights of the network in a way that minimizes the error

Steps – 1 & 2	FORWARD PASS
Steps – 3 & 4	REVERSE PASS

Contd.

Backpropagation

Forward Pass Demo



Contd.

Backpropagation

Reverse Pass

Steps :

- ➢ Calculate error between network output and desired output

(T – A)

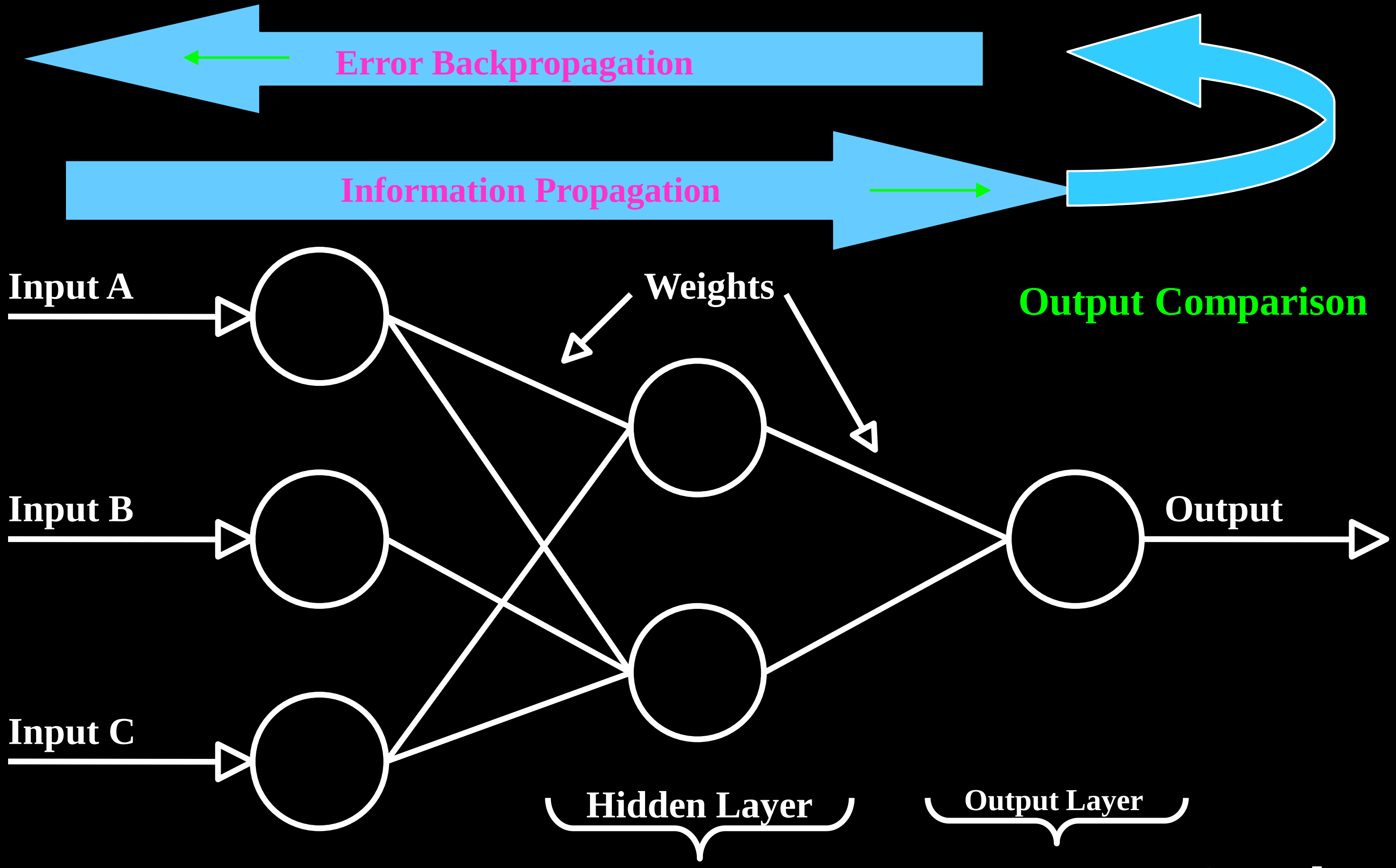
- ➢ Adjust weights of the network to minimize the error

Repeat these steps for each vector in the training set untill the error for the entire set is acceptably low.

Contd.

Backpropagation

Reverse Pass Demo



Contd.

Backpropagation

Reverse Pass (Weight adjustment)

- Adjusting the weights of output layer
- Adjusting the weights of hidden layer

Training the output layer

- Calculate error = TARGET - OUT
- Calculate the derivative of squashing function
- $F(NET)=1/(1+e^{-NET})$
- $F'(NET) = \partial OUT / \partial NET = OUT(1 - OUT)$
- Calculate δ

δ = error x derivative of squashing function

δ =(TARGET-OUT) x OUT(1-OUT) ------------------(1)

Contd.

Backpropagation

- Previous weight

$$\Delta Wpq,k = \eta\delta q,k\ OUTpj$$

- weight after adjustment

$$Wpq,k(n+1) = Wpq,k(n) + \Delta\ Wpq,k \text{ --------------------(2)}$$

Where **Wpq,k (n)** = the value of weight from neuron p in the hidden layer to neuron q in the output layer at step n (before adjustment)

Wpq,k (n+1) = the value of the weight at step n+1 after adjustment

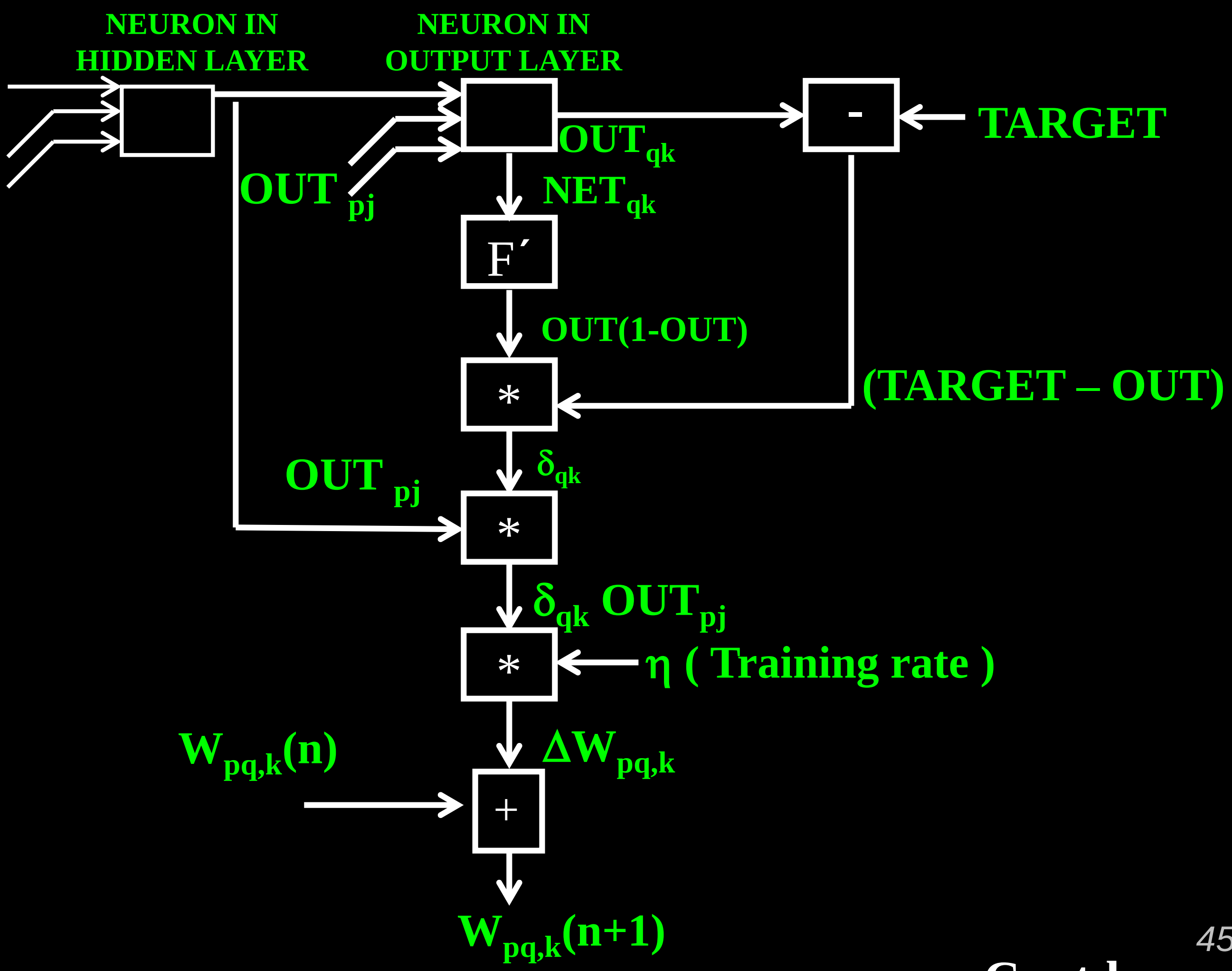
δq,k = the value of δ for neuron q in the output layer k

OUTpj = the value of OUT for neuron p in the hidden layer j

Contd.

Backpropagation

Training the output layer neuron



Contd.

Backpropagation

- Equation(2) are used for all layers both output and hidden

- δ Must be generated independent from target vector

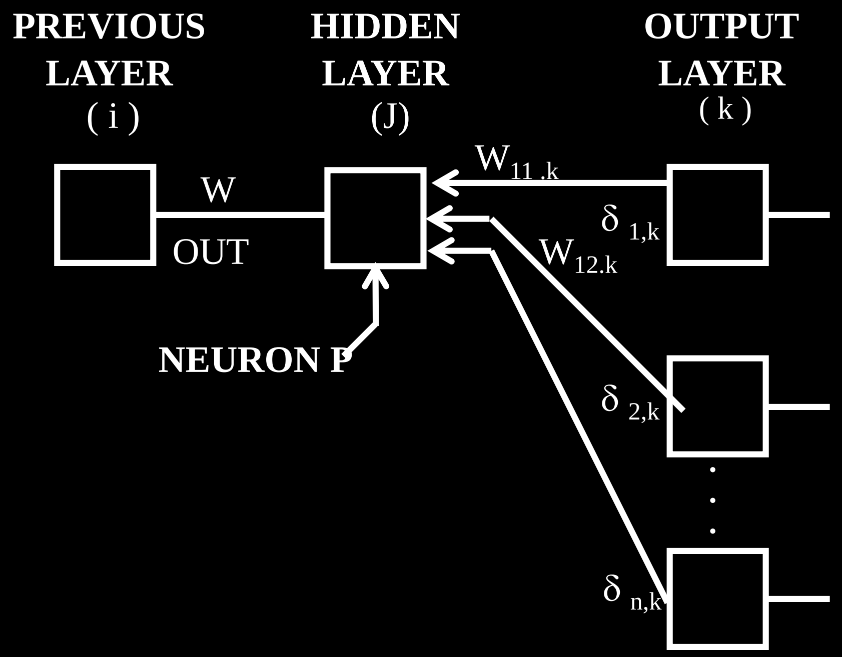
- First, δ is calculated for each neuron in the o/p layer as in eq^n(1)

- It is used to adjust the wts. Feeding into the output layer, then it is propagated back through the same wts.to generate a value for δ for each neuron in the first hidden layer.

- $\delta_{pj} = OUT_{pj}\,(1-OUT_{pj})(\Sigma_q\ \delta_{q,k}\ w_{pq,k})$

Backpropagation

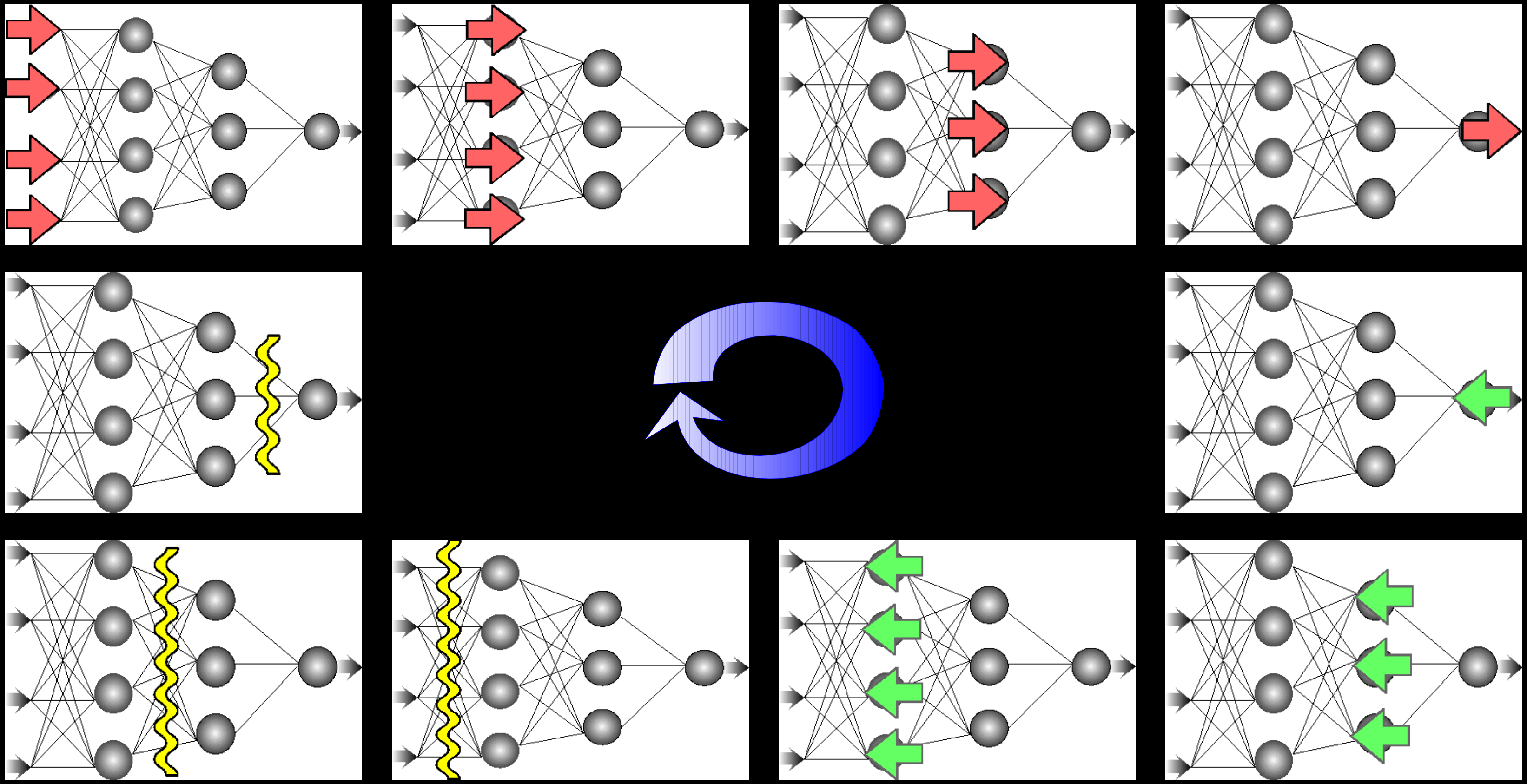
Training the hidden layer neuron



Contd.

Backpropagation

Backpropagation training cycle



THANK YOU