Artificial Neural Networks Basics

History of Neural Nets

1943: The McCulloch-Pitts neuron

1949: “The Organization of Behavior” by Hebb

1958: The ANN Perceptron (by Rosenblatt)

1960-62: Widrow-Hoff’s learning rule and the ANN Adaline

1968: Book “Perceptrons” by Minsky and Papert

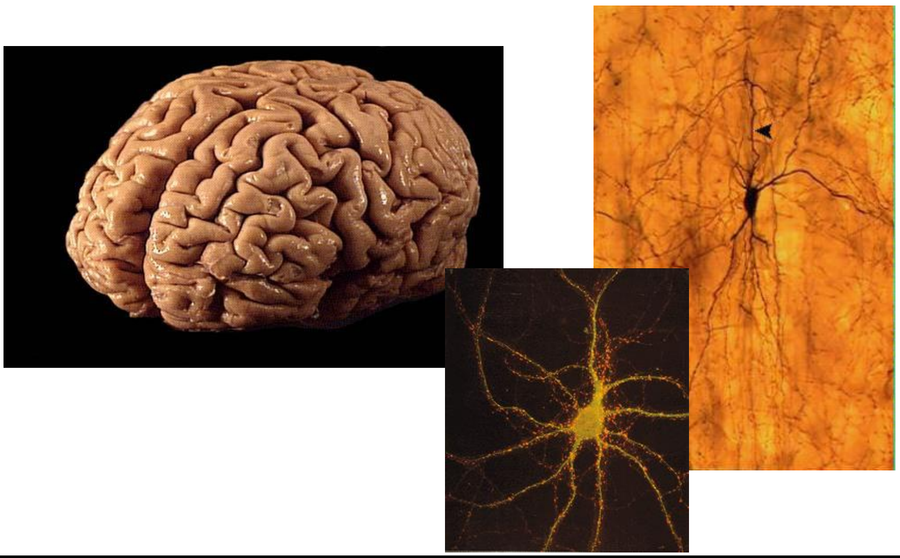
1970-80: ANN Winter: Anderson, Kohonen, Grossberg, Fukushima, Amari and others kept the field alive.

1982: Hopfield’s neural net

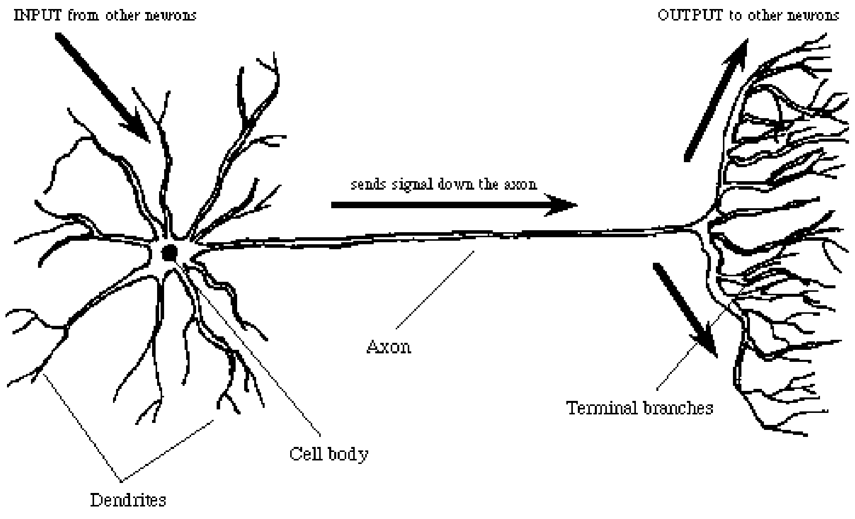
1985: The back-propagation algorithm for training MLPs (Werbos [1974], Rumelhart [1985] and LeCun [1985])

1990's: Convolutional Neural Networks (LeCun and others)

Human brain and neurons



Biological Neurons



Human Learning

Number of neurons:	~ 10^{10} - 10^{11}
Connections per neuron:	~ 10^4
Neuron switching time:	~ 1 – 10 ms
Scene recognition time:	~ 100 ms

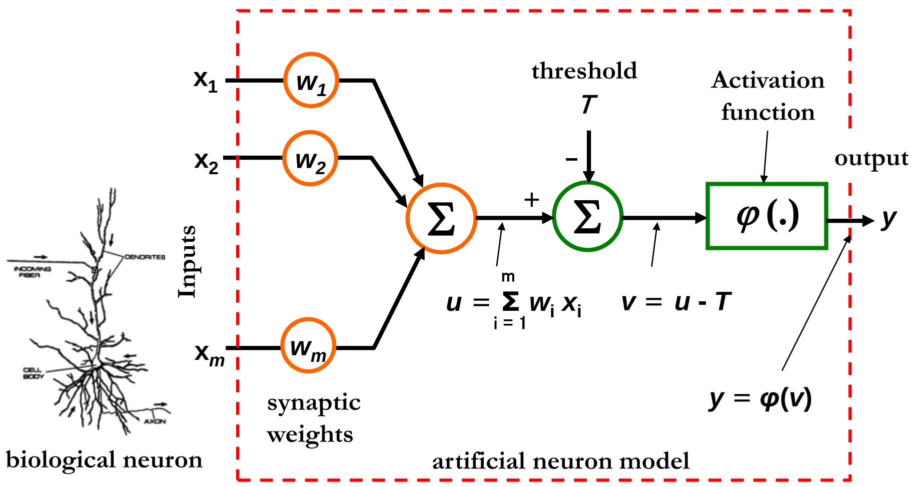
100 inference steps doesn’t seem much

ANN Definition

ANNs are massively parallel distributed processors that have a natural propensity for storing experiential knowledge and making it available for use. They resemble the brain in two respects:

- *Knowledge is acquired through a learning process.*
- *Interneuron connection strengths, known as synaptic weights, are used to store the knowledge.*

From biological neurons to the artificial neuron model



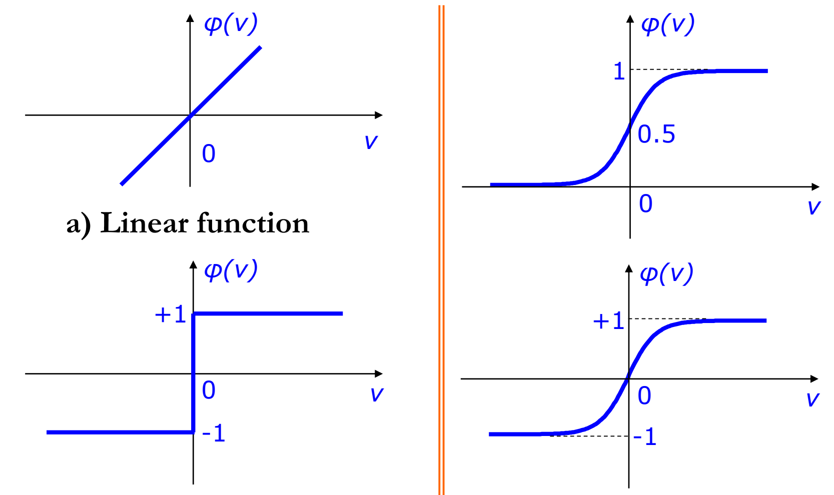
Neuron model

- Inputs: $\mathbf{x} = (x_1, x_2, \ldots, x_m)^T$
- Synaptic weights: $\mathbf{w} = (w_1, w_2, \ldots, w_m)^T$
- Input: $u = w_1 x_1 + \ldots + w_m x_m =$

$$= \sum_{i=1}^{m} w_i x_i = \mathbf{w}^T \mathbf{x}$$

- Total input: $v = u - T$
- Output of neuron: $y = \varphi(v)$

Common activation functions

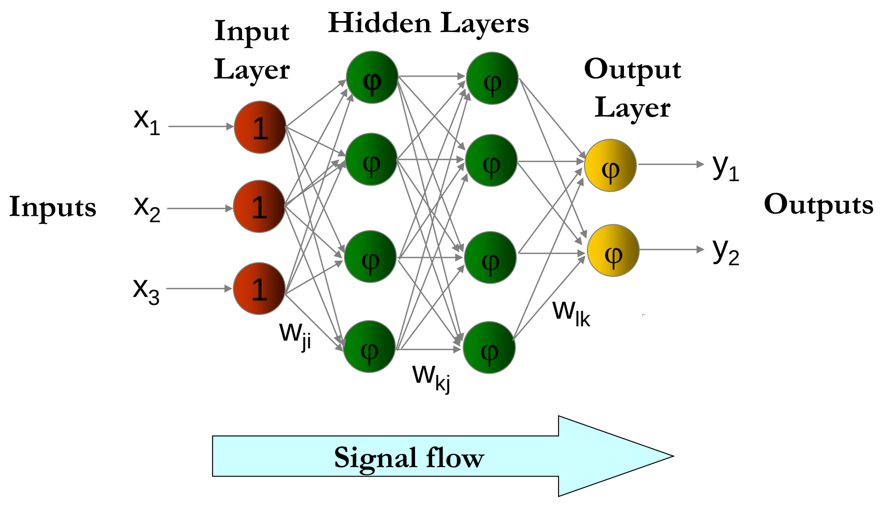


a) Linear function

b) Hard-limiter (step) function

c) Sigmoid functions

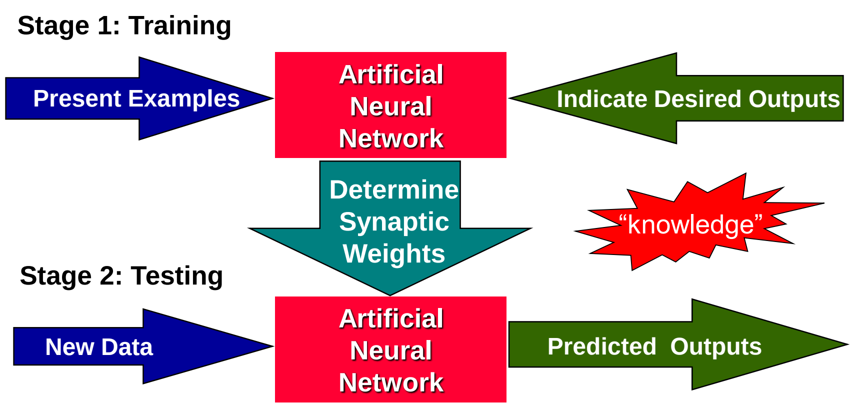
A typical ANN architecture

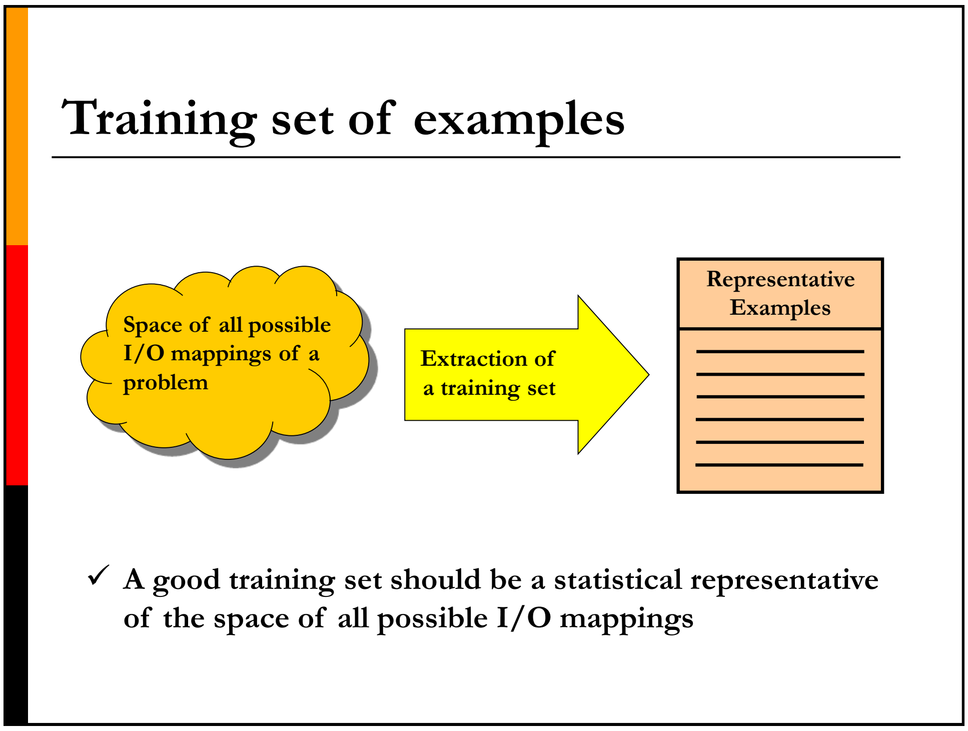


ANN paradigm

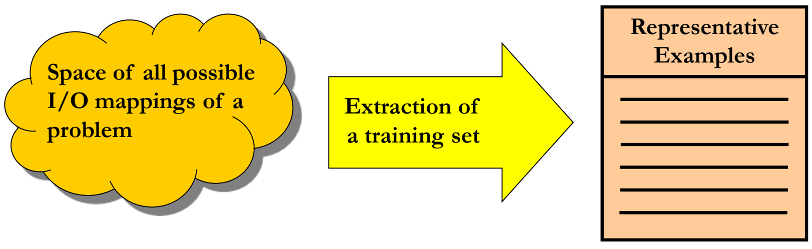
- Information flows from input to output.
- Usually, all neurons use the same activation function $\varphi(\cdot)$
- Connections between neurons are realized through the so called “synaptic weights” (parameters) of the network.
- In the previous network every neuron is shown connected to every neuron of the previous layer. However, neuronal connectivity and activation function may differ.
- The synaptic weights and thresholds are found using a learning (or training) algorithm. These parameters are modified through an iterative procedure so that the ANN realizes the desired input/output mapping.

General Supervised Learning Paradigm

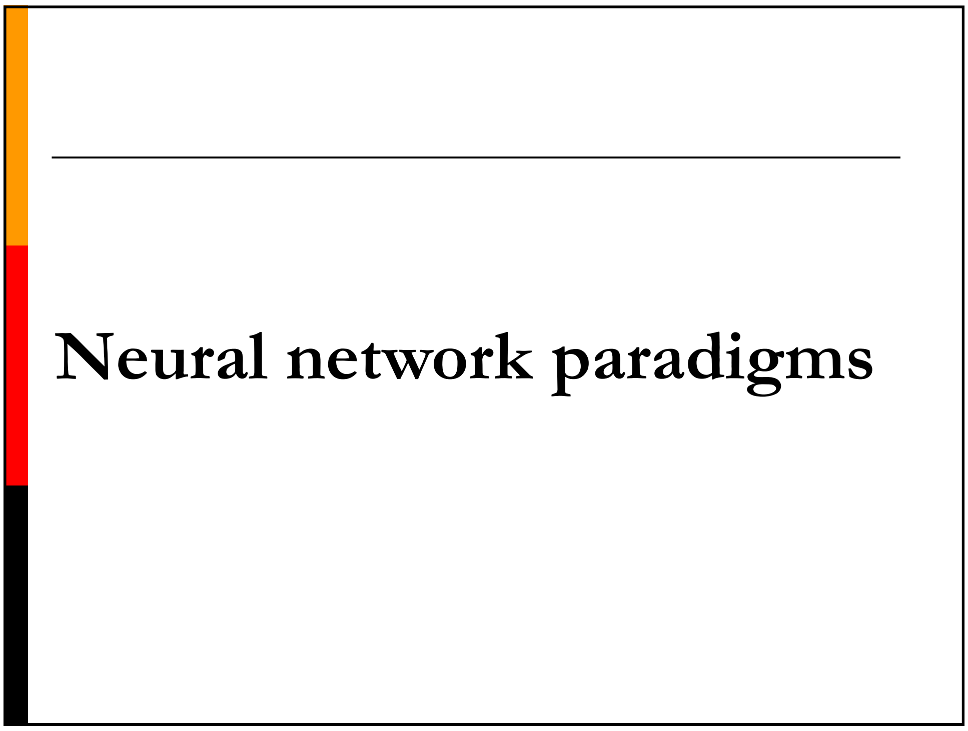




Training set of examples


Space of all possible I/O mappings of a problem
Extraction of a training set
Representative Examples

✓ A good training set should be a statistical representative of the space of all possible I/O mappings

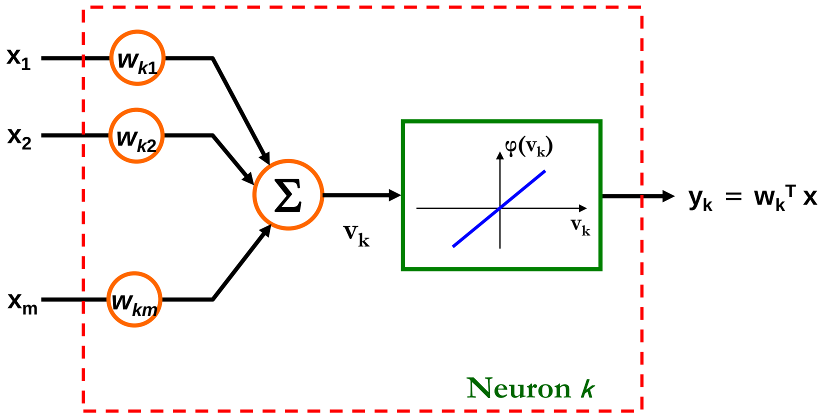


Neural network paradigms

Linear Associative Memories

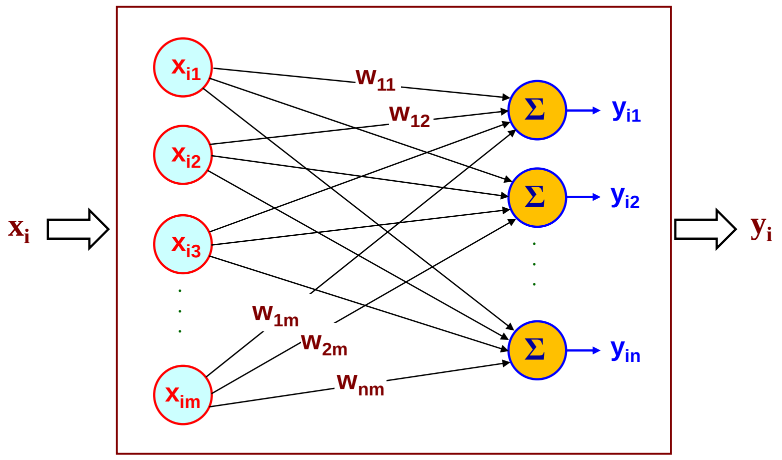
- They constitute ANN paradigms using the linear neuron model
- Two popular Linear Associative Memories:
  - The *Generalized Inverse Memory*
  - The *Correlation Matrix Memory* (also known as the *Outer Product Memory*)

The linear neuron model



no threshold, linear activation function

Linear Associative Memories



Linear Associative Memories

- The linear associative memories belong to the supervised learning paradigms
- They associate p input prototype patterns x_i with p output prototype patterns y_i. The $<x_i , y_i>$ pairs for i = 1, ..., p constitute the training set used for finding the synaptic weights of the network through a repetitive training procedure.

Symbolisms

- Input patterns: $x_i = (x_{i1}, x_{i2}, \ldots, x_{im})^T$ for $i = 1, 2, \ldots, p$
- Output patterns: $y_i = (y_{i1}, y_{i2}, \ldots, y_{in})^T$ for $i = 1, 2, \ldots, p$
- Input Matrix: $X = [x_1 \mid x_2 \mid \ldots \mid x_p]$ (mxp matrix)
- Output Matrix: $Y = [y_1 \mid y_2 \mid \ldots \mid y_p]$ (nxp matrix)
- Weight Matrix: $W^T = [w_1 \mid w_2 \mid \ldots \mid w_n]$ (mxn matrix)

GOAL

- Find matrix **W** (contains all network's parameters) that satisfies the system of equations:

$$y_i = W x_i \qquad \text{for every} \quad i = 1, 2, \ldots, p$$

- Equivalently, the above system of p equations is substituted by the following matrix equation:

$$Y = \underset{\textit{unknown}}{\underset{\uparrow}{W}} X$$

The Generalized Inverse Memory

- In general, there is no exact solution to the above system, i.e. there is no $\mathbf{W}$ that satisfies $\mathbf{Y} = \mathbf{W}\,\mathbf{X}$.
- The minimum square error approximate solution, i.e. that minimizes the Frobenius norm $\|\mathbf{Y} - \mathbf{W}\,\mathbf{X}\|^2$, is known as the Generalized Inverse Memory:

$$\mathbf{W} = \mathbf{Y}\,\mathbf{X}^+$$

where $\mathbf{X}^+$ $(\mathbf{p} \times \mathbf{n})$ is the generalized inverse (or pseudoinverse) of $\mathbf{X}$.

- If the columns of $\mathbf{X}$ are linearly independent:

$$\mathbf{X}^+ = (\mathbf{X}^T\mathbf{X})^{-1}\,\mathbf{X}^T$$

In this case, the GI memory gives the exact solution:

$$\mathbf{W} = \mathbf{Y}\,\mathbf{X}^+ = \mathbf{Y}\,(\mathbf{X}^T\mathbf{X})^{-1}\,\mathbf{X}^T$$

Verification of exact solution:

$$\mathbf{W}\,\mathbf{X} = \mathbf{Y}\,(\mathbf{X}^T\mathbf{X})^{-1}\,\mathbf{X}^T\mathbf{X} = \mathbf{Y}\,\mathbf{I} = \mathbf{Y}$$

where $\mathbf{I}$ is the identity matrix.

Associative Memory Recall

If we present pattern $\mathbf{x}_k$ at the input, then we obtain pattern $\mathbf{y}_k$ at the output in the case of exact solution. This is known as the memory recall phase:

$$\mathbf{y} = \mathbf{W}\,\mathbf{x}_k = \mathbf{W}\,\mathbf{X}\,\mathbf{e}_k = \mathbf{Y}\,\mathbf{e}_k = \mathbf{y}_k$$

where $\mathbf{e}_k$ is the *k*-th column of the identity matrix $\mathbf{I}_{pxp}$ which when multiplied with $\mathbf{X}$ gives the *k*-th column of $\mathbf{X}$.

Correlation Matrix Memory (CMM)

- The memory $\mathbf{W}$ is given by the outer product of the input and output matrices:

$$\mathbf{W} = \mathbf{Y}\,\mathbf{X}^T$$

- In the case of orthonormal (orthogonal and of unit length) input patterns, i.e. satisfying:

$$\mathbf{x}_i^T\mathbf{x}_j = \delta_{ij} \quad \text{where} \quad \delta_{ij} = \begin{cases} 1 & \text{αv } i = j \\ 0 & \text{αv } i \neq j \end{cases}$$

the above outer product is an exact solution.

Correlation Matrix Memory (CMM)

- Exact memory recall for orthonormal patterns.

  Let $\mathbf{x} = \mathbf{x}_k$. Then, the recalled output pattern will be:

  $$\mathbf{y} = \mathbf{W}\,\mathbf{x} = \mathbf{W}\,\mathbf{x}_k = \mathbf{Y}\,\mathbf{X}^T\mathbf{x}_k = \mathbf{Y}\,\mathbf{e}_k = \mathbf{y}_k$$

  where $\mathbf{e}_k$ is the ***k***-th column of the unit matrx $\mathbf{I}_{pxp}$.

- If $\mathbf{x}_k$ is corrupted by additive noise $\mathbf{n}_i$, i.e. $\mathbf{x} = \mathbf{x}_k + \mathbf{n}_i$:

  $$\mathbf{y} = \mathbf{W}\,(\mathbf{x}_k + \mathbf{n}_i) = \mathbf{W}\,\mathbf{x}_k + \mathbf{W}\,\mathbf{n}_i = \mathbf{y}_k + \mathbf{n}_o$$

  where $\mathbf{n}_o$ is the output noise. It can be shown that for auto-associative memories ($\mathbf{y}_k = \mathbf{x}_k\ \forall k$) the output noise power $\sigma_o^2 = E[n_o^2]$ is smaller than the input noise power $\sigma_i^2 = E[n_i^2]$.

Hebb's learning rule

- Donald O. Hebb, *Organization of Behavior* (1949)
- "When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased."

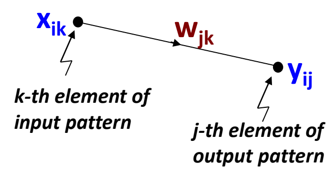
CMM construction using Hebb's rule

- Present one-by-one the I/O pairs $< x_i , y_i > \forall i = 1, .., p$ and update each synaptic weight w_{jk} according to:

  $w_{jk}^{new} = w_{jk}^{old} + \Delta w_{jk}(i)$

  where

  $\Delta w_{jk}(i) = x_{ik}\ y_{ij}$



  i.e. weight changes are proportional to the presynaptic (x_{ik}) and postsynaptic (y_{ik}) activities in accordance to Hebb's rule

- Hence, after training: $w_{jk} = \sum_{i=1}^{p} \Delta w_{jk}(i)$ and $W = Y X^T$

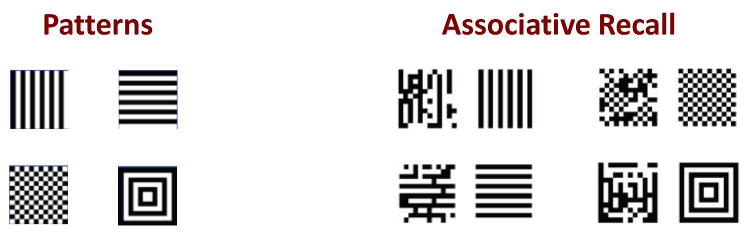
Types of Associative Memories

1. Auto-associative memories

In this case every input pattern is associated with itself (Y = X):

$$W = X X^+ \quad \text{or} \quad W = X X^T$$

Example: Orthogonal bipolar (+1/-1) prototype patterns



Types of Associative Memories

2. Hetero-associative memories

General case, input and output patterns are different **($Y \neq X$):**

$$W = Y X^{+} \quad \text{or} \quad W = Y X^{T}$$

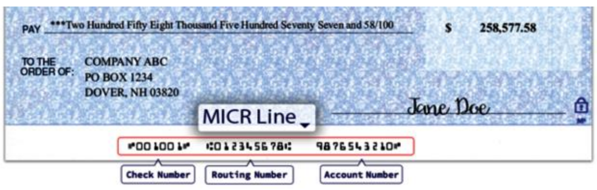
3. Classifiers

In this case, the ***k****-th input pattern (x_k) is associated with the* ***k****-th column of the unit matrx* ($y_k = e_k$) *i.e.* $Y = \mathrm{I}$:

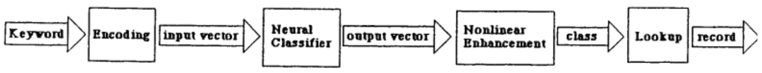
$$W = X^{+} \quad \text{or} \quad W = X^{T}$$

Applications

- **Automatic digit recognition**



- **Spelling correction**



Generic neural network structure for an associative spelling checker.