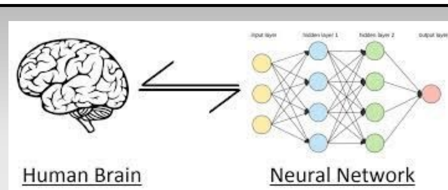
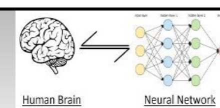
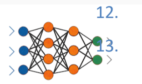


Artificial Neural Network and Deep Learning Lecture 2



Course Outlines

1. Introduction to NNs
2. Main characteristics of Neural Networks
3. Resenblatt's perceptron Single Layer Network
4. Least Mean Square algorithm for Single Layer Network
5. Multilayer Perceptron (MLP) Network
6. Optimization of Back-Propagation Algorithm
7. Deep Learning
8. Convolutional Neural Networks (CNNs)
9. Regularization and CNNs
10. YOLO for Object Detection
11. Fully CNNs and U-Net for Image Segmentation, Generative Models
12. Recurrent Neural Networks (RNNs) and Transformers
13. Graph Neural Networks



The main characteristics of Neural Network

- **Activation function**:
  - Are mathematical functions that **limit the range of output values** of a node.
- **Architecture or Structure** :
  - The **connectivity of neurons** (nodes) determines the neural network structure (architecture).
- **Learning algorithm**, or training method:
  - Method for **determining weights of the connections**
  - The **manner in which the neurons of neural network are structured** is intimately linked with the learning algorithm used **to train the network**.

Agenda

- Activation Function
  - Types of Activation Function
  - Stochastic Model of a Neuron
- Neural Networks Architectures
- Learning in Neural Networks
  - Learning Methods
  - Learning rules

Activation Function and Squashing

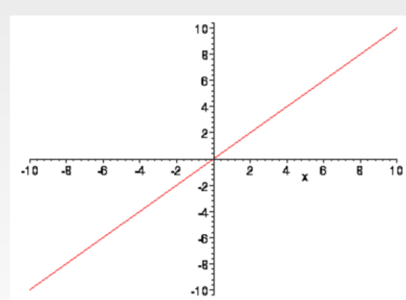
- The <u>mapping</u> from net unit activation to output may be characterized by activation or squashing function.
- Why do we need activation functions?
  - Non-linearity is achieved through the use of activation functions, which limit or squash the range of values a neuron can express.
  - The squashing function serves to limit the domain (0 to 1 or -1 to 2).
- In general, there are many different kinds of activation functions.
- To make the neuron learnable, some kind of continuous function is needed.

1- Identity function (linear function)

The simplest example is that of a linear unit, where

$$y_k = f(net_k) = net_k$$

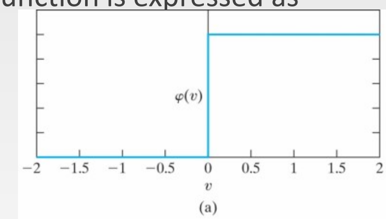
In this case, the activation function is the identity mapping.



2- Threshold (Step) function

❑The output of neuron **k** employing such a threshold function is expressed as

$$y_k = \varphi(v_k) = \begin{cases} 1 & if v_k \geq 0 \\ 0 & if v_k < 0 \end{cases}$$



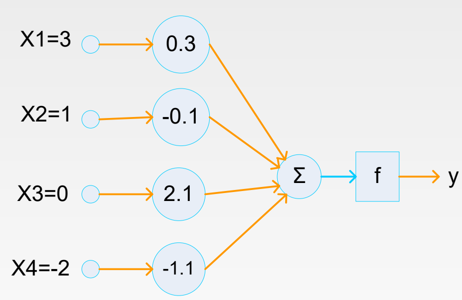
- The model uses this function is proposed by **McCulloch and Pitts** in 1943.
- That is, the neuron will have output signal only if its activation potential is non-negative, a property known as ***all-or-none*.**
- **Is it continuous function?**

2- Threshold (Step) function (Cont.) Example

Using the step function with threshold equal 3 for neuron has an input (3, 1, 0, -2) and weight (0.3, -0.1, 2.1, -1.1), calculate the output of that neuron?

$$\begin{aligned} u_k = net_k &= W^T X \\ &= 3(0.3)+1(-0.1)+0(2.1)+-2(-1.1) \\ &= 0.9+(-0.1)+2.2 \\ &= 3 \end{aligned}$$

$$f(u_k) = f(3) = 1$$

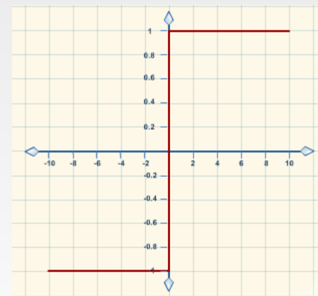


3- Symmetric Threshold function with bi-polar output

The Symmetric Threshold function has the form:

$$y_k = \varphi(v_k) = \begin{cases} 1 & \text{if } v_k \geq 0 \\ -1 & \text{if } v_k < 0 \end{cases}$$

which is commonly referred to as the ***signum function***.



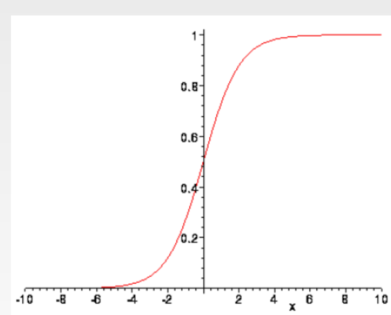
4- Sigmoid function

• The math of some neural nets requires that the activation function be *continuously differentiable.*

• It is defined as a strictly increasing function.

• The sigmoid function has a s-shaped graph.

• An example of the sigmoid function is the logistic function, defined by.



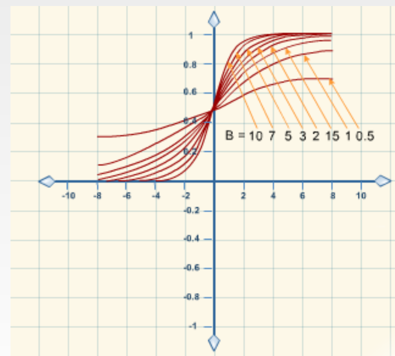
$$y_k = \varphi(v_k) = \frac{1}{1 + \exp(-a v_k)}$$

where ***a*** is the slop parameter of the sigmoid function.

5- Sigmoid function, cont.

By varying the parameter a, we obtain sigmoid functions of different slopes, as illustrated in the next figure:

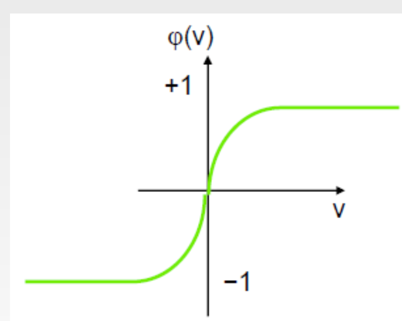
- Slop at the origin equals a/4.
- A sigmoid function often used to approximate the step function. If the slop parameter approaches infinity, the sigmoid function becomes a threshold function.



5- Hyperbolic tangent sigmoid function with bi-polar O/P

We may use the Hyperbolic tangent function as the corresponding form of a sigmoid function that has a range from -1 to +1.

$$\varphi(v) = \tanh(av) = \frac{1-\exp(-av)}{1+\exp(-av)}$$

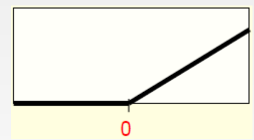


6- Ramp function or Rectified Linear Unit (ReLU) (sometimes called linear threshold neurons)

It can be expressed by numerous definitions, for example "0 for negative inputs, output equals input for non-negative inputs".

$$v = b + \sum_i w_i x_i$$

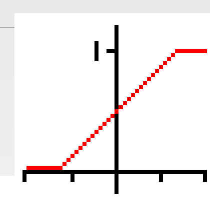
$$y = \begin{cases} v & if \;\; v \geq 0 \\ 0 & otherwise \end{cases}$$



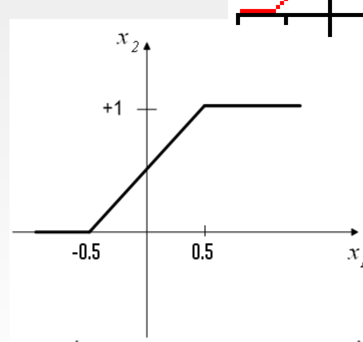
7- Piecewise-Linear function

- It is a function whose graph is composed of straight-line sections.

$$f(x) = \begin{cases} 0 & \text{if } x \leq x_{min} \\ mx+b & \text{if } x_{max} > x > x_{min} \\ 1 & \text{if } x \geq x_{max} \end{cases}$$



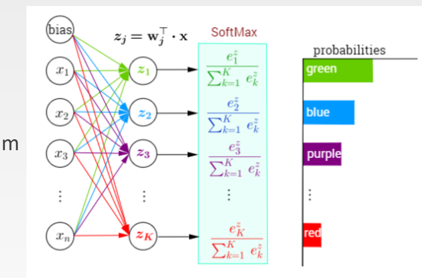
$$y_k = \varphi(v_k) = \begin{cases} 1 & if v_k \geq \frac{1}{2} \\ v_k + 0.5 & if -\frac{1}{2} < v_k < \frac{1}{2} \\ 0 & if v_k \leq -\frac{1}{2} \end{cases}$$



8- Softmax function

- Also called the normalized exponential function. **Generalization of the logistic function**.
- The softmax function takes as input a **vector z of K real numbers** and normalizes it into a probability distribution consisting of K probabilities proportional to **the exponentials of the input numbers**.
- Softmax converts the outputs to probabilities by dividing the output by the sum of all the output values, where each output is in the range (0,1) and sum of all the outputs was 1.

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad \text{for } j = 1, \ldots, K.$$

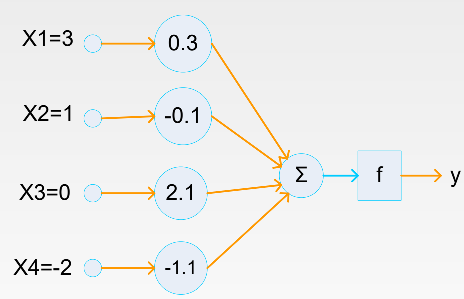


- The softmax function is used in various multiclass classification methods.

Example

Using the sigmoid function with slop equal 2 for neuron have an input (3, 1, 0, -2) and weight (0.3, -0.1, 2.1, -1.1), calculate the output of that neuron?

$$u_k = net_k = W^T X$$
$$= 3(0.3) + 1(-0.1) + 0(2.1) + -2(-1.1)$$
$$= 0.9 + (-0.1) + 2.2 = 3$$
$$y_k = \varphi(3) = \frac{1}{1+e^{-2*3}} = 0.998$$



Example

Given a two-input neuron with the following parameters: b = 1.2, W=[3 2], and X=[-5 6]^T, calculate the neuron output for the following transfer functions:

i. A symmetrical Threshold transfer function

ii. A linear transfer function

iii. A hyperbolic tangent sigmoid transfer function

Solution:

$$net = \begin{bmatrix} 3 & 2 \end{bmatrix} \begin{bmatrix} -5 \\ 6 \end{bmatrix} + (1.2)$$

$$= -15 + 12 + 1.2 \qquad = -1.8$$

$$i. \quad y = symmetric\,threshold(-1.8) = -1$$

$$ii. \quad y = linear(-1.8) = -1.8$$

$$iii. \quad y = \text{hyperbolic tangent sigmoid}(-1.8) = (1 - \exp(1.8))/(1 + \exp(1.8)) = -0.7163$$

Range of activation function

- The **range** of the threshold piecewise-Linear and sigmoid activation functions is from 0 to +1.

◦ This type of activation functions generates **uni-polar** output signals.

- It is sometimes desirable to have the activation function range from -1 to +1, in which case the activation function assumes an antisymmetric form with respect to the origin.

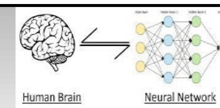
◦ This type of activation functions generates **bi-polar** output signals.

Agenda

- **Activation Function**
  - Types of Activation Function
  - Stochastic Model of a Neuron
- Neural Networks Architectures
- Learning in Neural Networks
  - Learning Methods
  - Learning rules

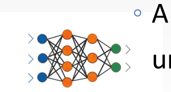
Stochastic Model of a Neuron

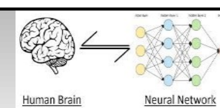
- The previous neuron models are ***deterministic***, that is, its output is precisely known for any input signal.
- *Deterministic modeling gives you* ***the same exact results for a particular set of inputs****,* ***no matter how many times you re-calculate the model****.*
  - *The mathematical properties are known.*
  - *None of them is random.*
  - *There is only one set of specific values and only one answer or solution to a problem.*
  - *With a deterministic model, the uncertain factors are external to the model.*



Stochastic Model of a Neuron

- In some applications it is desirable to have a *stochastic neuron model*.
  - Stochastic modeling, on the other hand, **<u>is inherently random</u>**.
  - The uncertain factors are built into the model.
  - That is, the neuron is *permitted* to reside in only one of two states; +1 and -1. The *decision* for a neuron to *fire* (i.e., switch its state from "*off*" to "*on*") is *probabilistic*.
    - Example 1: Predict how company balance sheets will look at a given point in the future
    - Example 2: Profitability ratio in the stock investing in the future coming six months.
  - A stochastic model incorporates random variables to produce many different outcomes under diverse conditions.





Stochastic Model of a Neuron

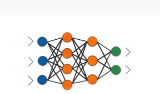
- Let *x* denote the *state* of the neuron, and *P(v)* denotes the *probability* of *firing*, where *v* is the activation potential, then we may write:

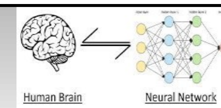
$$x = \begin{cases} +1 & \text{with probability } \mathrm{P}(v) \\ -1 & \text{with probability } 1\text{-}\mathrm{P}(v) \end{cases}$$

- A standard choice of *P(v)* is the sigmoid function

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

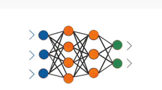
- Where *T* is a parameter used to control the *noise level* and therefore the *uncertainty* in *firing*. When *T → 0*, the model reduces to the deterministic model. The softmax function is often used in the final layer of a neural network-based classifier.

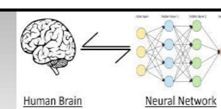




Agenda

- Activation Function
  - Types of Activation Function
  - Stochastic Model of a Neuron
- <u>Neural Networks Architectures</u>
- Learning in Neural Networks
  - Learning Methods
  - Learning rules

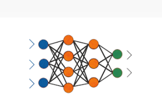




Classes of Neural Network Structures

- An architecture is the way in which the neurons are connected together.
- We may identify two fundamentally different classes of network architectures (Structures):

1. **Feedforward Networks**
2. **Backforward or Recurrent Networks**

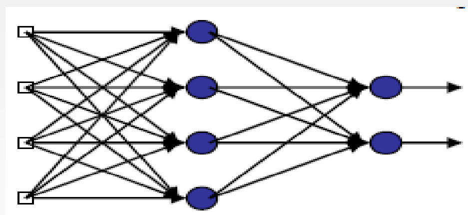


1- Feedforward Networks

These are the commonest type of neural network in practical applications.

– The first layer is the input and the last layer is the output.

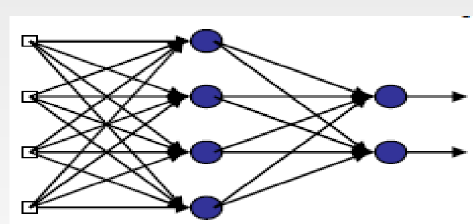
– If there is more than one hidden layer, we call them “deep” neural networks.



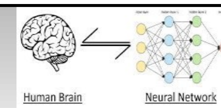
input units hidden units output units

1- Feedforward Networks
Characteristics

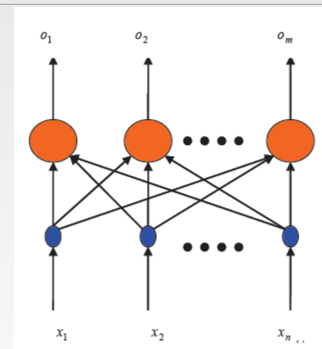
- Hierarchical: the neurons are arranged in **separate layers**
- There is **no connection between the neurons in the same layer**
- This network is strictly a Feedforward, in which graphs have **no loops**.
- The connections are **unidirectional**
- The neurons in one layer receive inputs from the previous layer
- The neurons in one layer delivers its output to the next layer.



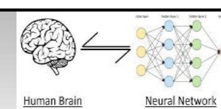
1- Feedforward Networks, cont.
Types



1. Single-layer Feedforward Networks
   - **Single-Layer Feedforward** Network is the simplest form of layered network.
   - The "signal-layer" referring to the output layer.
   - It has an input layer of source nodes that projects onto an output layer of neurons (computation nodes), but not vice versa.
   - A network is called a *single-layer network*, because we do not count the input layer since no computation is performed there.

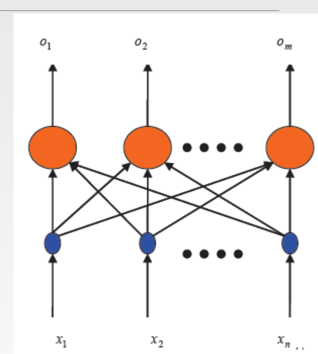


1- Feedforward Networks, cont.
Types



1. Single-layer Feedforward Networks, cont.:
   - The input-output relation of a single layer neural network is given by

$$\begin{bmatrix} o_1 \\ o_2 \\ \vdots \\ o_m \end{bmatrix} = F \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ w_{m1} & w_{m2} & \cdots & w_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$



Where:

- O_1, O_2 ,…,O_m are the output nodes.
- X_1, X_2, …, X_n are the input nodes.
- **F** is the activation (transfer) function.

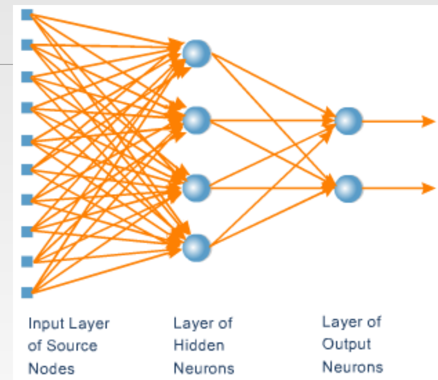
row *i* in the weight matrix, W_{i1}, W_{i2}, …, W_{in} , represent the weight **associative with output node O_i** .

and column j in the weight matrix, W_{1j}, W_{2j}, …, W_{mj} , represent the weight **associative with input node X_j** .

1- Feedforward Networks, cont.
Types

2. Multilayer Feedforward Networks

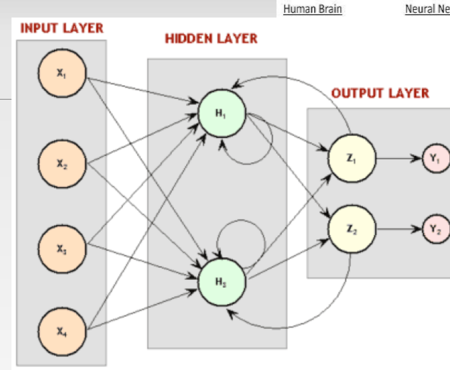
- Contains:
  - Input layer (source nodes)
  - one or more hidden layers, whose computation nodes are called hidden neurons or hidden units.
  - One output layer
- Example: 10-4-2 network, because it has 10 source node, 4 hidden neurons, and 2 output neurons.



➢ It is said to be ***fully connected*** in the sense that every node in each layer of the network is connected to every other node in the adjacent forward layer; otherwise, it is called ***partially connected*** if some of the weight connections are missing from the network.

2- Recurrent Networks

- It has at least one *feedback* loop.
- The network may have or not hidden neurons.
- There could be neurons with self-feedback links; that is the output of a neuron is fed back into its self as input.
- They are more biologically realistic.



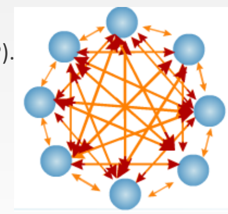
➢ Recurrent neural networks are a very natural way to model sequential data. For example, it is the most appropriate for predicting the price of a stock.

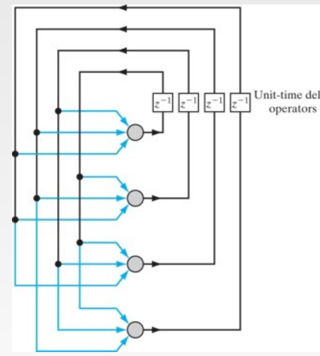
➢ Feedback connection takes the output of the previous data in a series as its next input.

➢ They have the ability to remember information in their hidden state for a long time.

2- Recurrent Networks, cont.

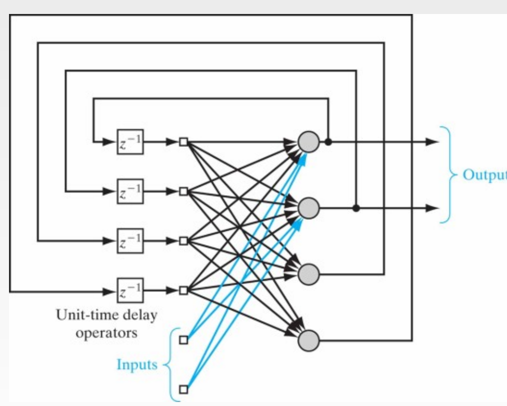
- Example: Hopfield network is a special kind of the recurrent network:
  - The connections between units are symmetrical (they have the same weight in both directions).
  - The connections is bidirectional.
    - There is no hierarchical arrangement.
    - Consist of a single layer of neurons (I/P & O/P).
    - No hidden layer.
    - No self-feedback loops.





2- Recurrent Networks, cont.

- Another example: A recurrent networks with hidden neurons and self feedback connections.



2- Recurrent Networks, cont.

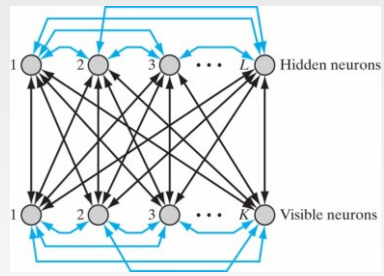
Another example: Boltzmann machines

- It is a Symmetrically connected networks with hidden units.

– They are much more powerful models than Hopfield nets.

– They are less powerful than recurrent neural networks.

– They have a beautifully simple learning algorithm.



Feedback connection

Feedback **is said to exist in *dynamic* systems whenever the output of a node influences in part the input of that particular node. (Very important in the study of *recurrent networks*).**

$$y_k(n) = \mathbf{A}[x'_j(n)]$$

and $$x'_j(n) = x_j(n) + \mathbf{B}[y_k(n)]$$

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

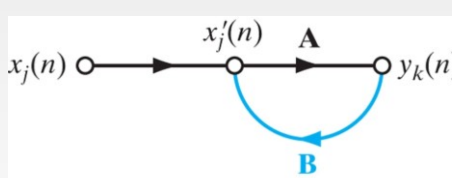


Figure 12 Signal-flow graph of a single-loop feedback system.

Agenda

- Activation Function
  - Types of Activation Function
  - Stochastic Model of a Neuron
- Neural Networks Architectures
- Learning in Neural Networks
  - Learning Methods
  - Learning rules

Learning in Neural Networks

Learning

- is a process to store the information into the network.
- defines how the system "adapts" to new knowledge.
- On-line learning and off-line learning.

Learning rules, for a connectionist system, are algorithms or equations which govern changes in the weights of the connections in a network.

Learning Approaches in NNs

- The learning methods in Neural Networks are classified into two basic types:

1- Learning with a Teacher (Supervised Learning)

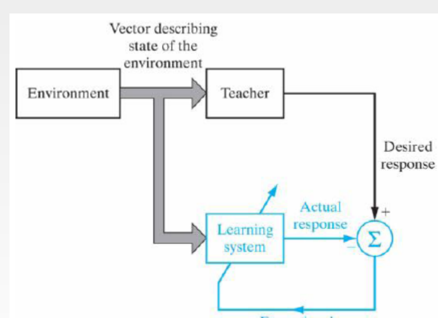
2- Learning without Teacher (Unsupervised Learning)

- These two types are classified based on:
  - **presence or absence** of teacher and
  - the **information** provided for the system to learn.

1- Learning with a teacher (Supervised Learning)

- Supervised learning is the problem of finding an input-output mapping from empirical data.

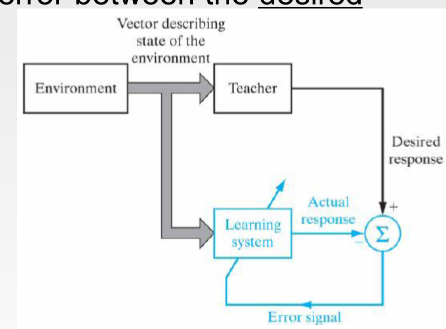
- The teacher has knowledge of the environment.



- The teacher is able to provide the neural network with a desired response for the training vector.
- So the NN is supplied with a sequence of *labeled training* patterns representing different classes.
- Each *training Labeled pattern* contains input signals (features), and the desires output class, (x_i, d_i).

1- Learning with a teacher, cont.

- The learning algorithm tries to minimize the error between the <u>desired response *t*</u> and the <u>actual output *y*</u>.



- In this way, the connection strengths of NN (Weights) are modified depending on
  - the input signal receives,
  - its output value (actual response)
  - and the desired response.

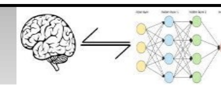
- The supervised learning process constitutes a *closed-loop feedback system*.

1- Learning with a teacher, cont.

Properties:
- Adaptively
  - Adapt weights to the environment
- Generalization ability

Supervised Learning algorithms:
- Perceptron learning algorithm.
- Error Correction Learning
  - Least Mean Square (LMS).
  - Back-propagation algorithm.

Tasks:
- Pattern classification
- Object Recognition
- Regression

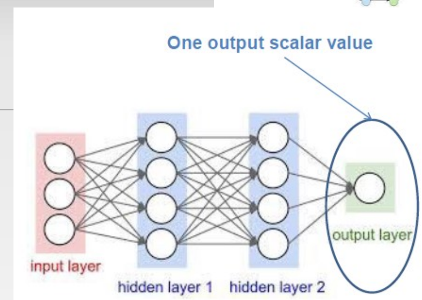


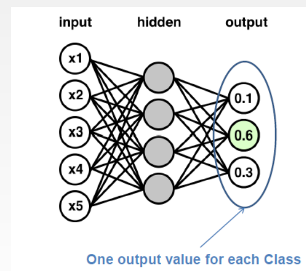
1- Learning with a teacher, cont.

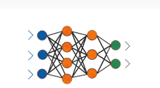
Neural Networks Classifier vs Regressor

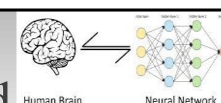
✓Label Data: Each training case consists of an input vector x and a target output t.

- Regression: The target output is a real number or a whole vector of real numbers.
  - – The price of a stock in 6 months time.
  - – The temperature at noon tomorrow.
- Classification: The target output is a class label.
  - – The simplest case is a choice between 1 and 0.
  - – We can also have multiple alternative labels.









2- Learning *without* a Teacher (Unsupervised Learning)

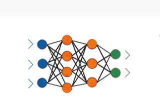
❖The teacher's response is not available

❖No error signal is available

❖When no teacher's response is available the NN will modify its weight **based only on the input**.

Unsupervised Learning is the problem of finding structure in data.

Tasks:
- Dimensionality reduction
- Clustering



2- Unsupervised Learning, cont.

Unsupervised Learning algorithms :

- Hebbian Learning
  - Used for Dimensionality reduction
  - Similar to Principal Component Analysis (PCA)
- Competitive learning
  - Used for Clustering - The NN must identify clusters in the input data and discover classes automatically
  - Self-organizing features map (SOFM) are neural network model for unsupervised learning.

2- Unsupervised Learning, cont.

Dimensionality reduction used as preprocessing step in Pattern Recognition schema.

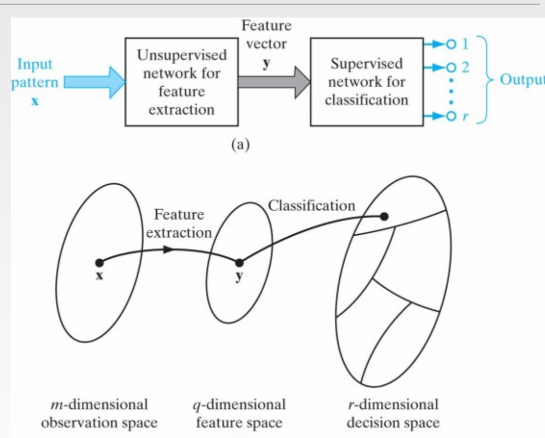


Illustration of the classical approach to pattern classification

Learning Tasks

Supervised

Learn to predict an output when given an input vector.

Data:

Labeled examples

(input , desired output)

Tasks:

pattern classification

object recognition

regression

NN models:

- **Perceptron**
- **Adaline**
- **Multilayer feed-forward NN**
- **Radial basis function**

Unsupervised

Discover a good internal representation of the input, i.e. find similarities and differences between data points.

Data:

Unlabeled examples

(different realizations of the input)

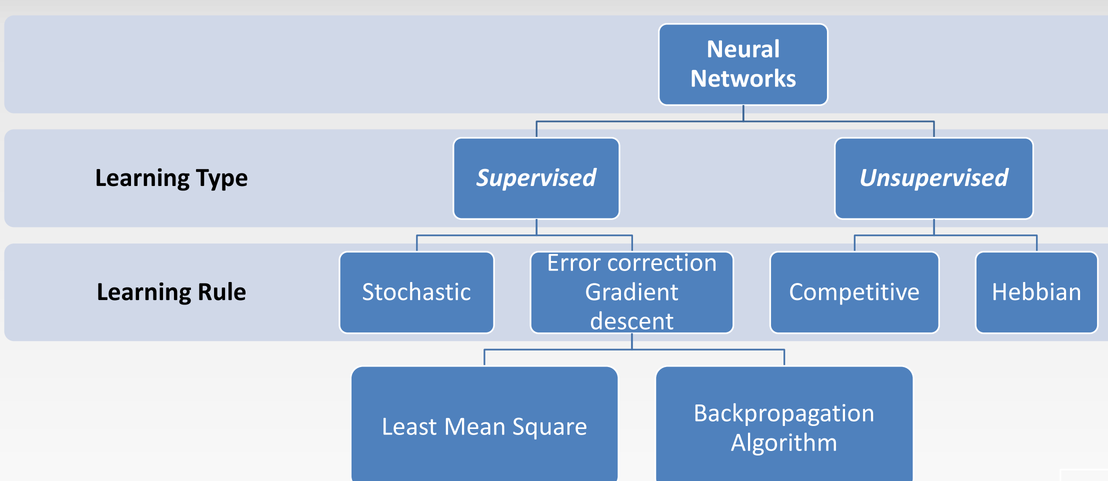
Tasks:

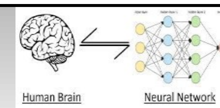
clustering

dimensionality reduction

NN models:

- **Self-organizing maps (SOM)**
- **Hopfield networks**
- **Principal Component Analysis (PCA)**





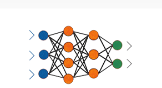
Learning rules

There are four basic types of learning rules:

- Hebbian,
- Error correction Gradient descent,
- Competitive and
- Stochastic learning (ex.:Boltzmann Machine**)**

Each of these can be trained with or without a teacher.

Have a particular architecture and learning algorithm.

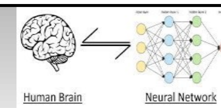


Taxonomy of neural networks

Based on architecture types and the learning methods

LAM=linear Associative Memory
BAM=bidirection Associative Memory
BP=Backpropagation algorithm

- **Feedforward**
  - **Unsupervised** **(PCA, LAM)**
  - **Supervised** **(Perceptron, LMS, MLP(BP))**
- **Recurrent**
  - **Unsupervised** **(SOFM, ART, Hopfield, BAM)**
  - **Supervised** **(Elman, Jordan, Bolzmann Machine)**



What’s Next Lecture

Single Layer Feedforward Network

(Rosenblatt’s Perceptron)