EE7207 Neural and Fuzzy Systems

Mao Kezhi

School of Electrical and Electronic Engineering
Nanyang Technological University

Tel: 67904284
Email: ekzmao@ntu.edu.sg
Office: S2.1-B2-14

Continuous Assessments and Exam

<u>Continuous Assessments:</u>

- 20%
- Two assessments (1 from NN, 1 from FS)
- For neural system part:
  - ✓ Homework assignment (2 weeks to finish)
  - ✓ Work on a real problem/data
  - ✓ Release the question on Week 7

<u>Exam:</u>

- 80%
- Five questions, 20 marks each

References

1. Simon Haykin, Neural Networks and Learning Machines, 3rd Edition, Prentice Hall, 2009.

2. Robert J. Schalkoff, Artificial Neural Networks, McGRAW-HILL, 2005.

3. Materials on the Internet.

Lecture Organization of Neural Systems

- Introduction to Neural Networks;
- Hopfield and Bi-directional Associative Memory Neural Network;
- Self-organizing Map;
- Radial Basis Function Neural Network;
- Support Vector Machines;
- Multilayer Perceptron Neural Network;
- Deep Neural Networks.

Introduction to Neural Networks

- An Overview of Neural Networks
- Neuron Models and Network Architectures
- Neural Network Learning

Why Artificial Neural Networks?

As modern computers become ever more powerful, scientists continue to be challenged to use machines effectively for some tasks that are relatively simple to humans. A good example is the processing of visual information. A one-year-old baby is much better and faster at recognizing objects, faces and so on than many AI systems running on the fastest supercomputer.

Traditional, sequential, logic-based computers excel in arithmetic, but is less effective than human brains in many fields. Human brains have many other features that would be desirable in artificial systems.

Let's look at the features of human brain.

Features of Human Brain

(1) Human brain is robust and fault tolerant. Nerve cells in the brain die every day without affecting its performance significantly;

(2) Human brain is flexible. It can easily adjust to a new environment by "learning"---it does not have to be programmed in any language;

(3) Human brain can deal with information that is fuzzy, probabilistic, noisy, or inconsistent;

(4) Human brain is highly parallel and highly nonlinear.

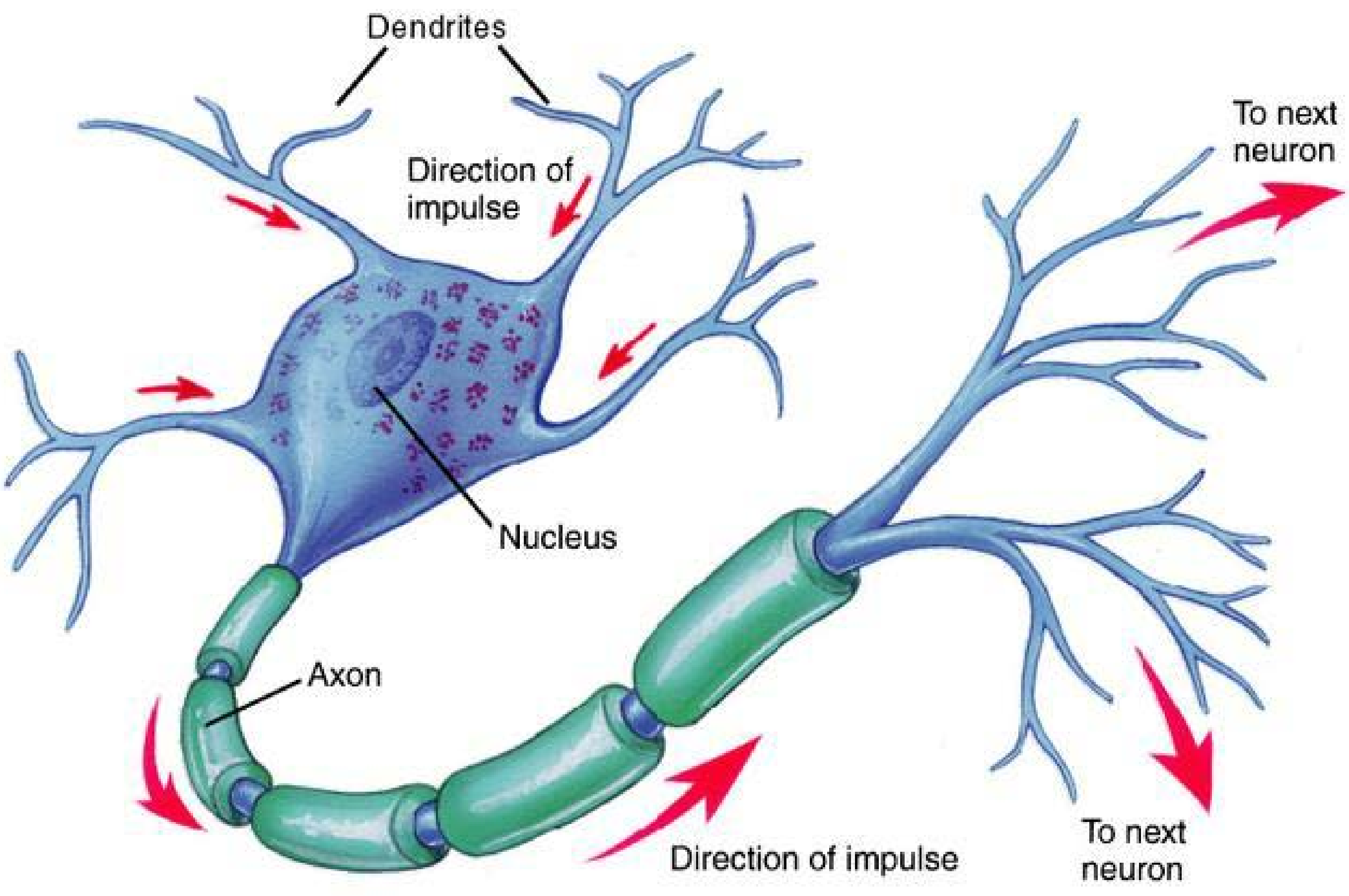
It can be said that human brains outperform computers in many areas. This is the motivation for studying neural computation.

Biological Origin of Artificial Neural Networks

The nervous system consists of two classes of cells: neurons (nerve cells) and glia (glial cells). Neurons are the basic building blocks of biological information processing systems, and the glial cells perform more of a support function. Therefore neurons are of major concern.

A biological neuron consists of three major portions:

(1) *Cell body*: information processing unit;
(2) *Axon*: the main conduction mechanism of neurons;
(3) *Dendrites*: facilitate excitatory and inhibitory functions in axon signal generation.

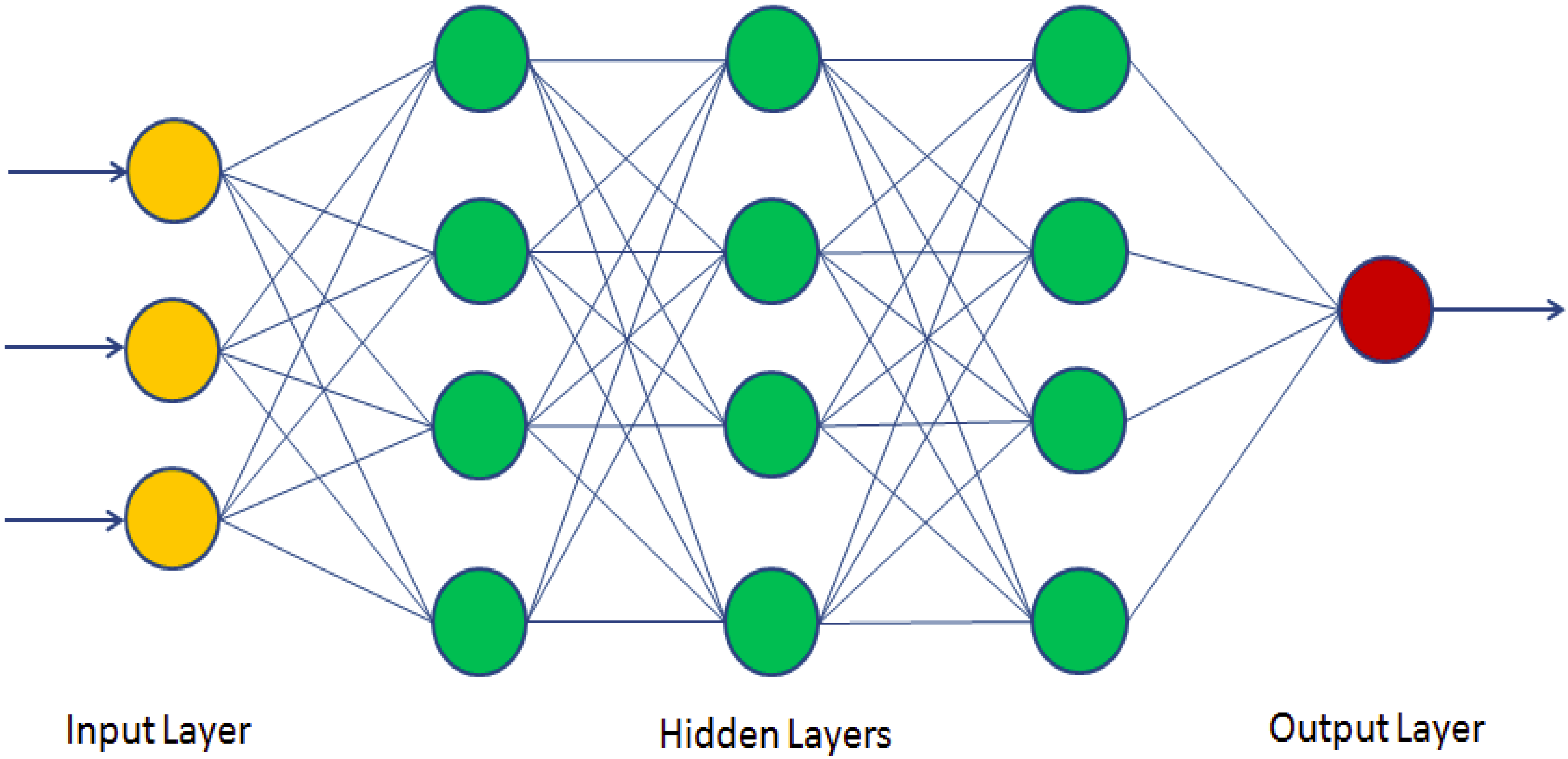


Sketch of a biological neuron

What is An Artificial Neural Network?

An artificial neural network is an information processing system that has certain performance characteristics in common with biological neuron networks. Artificial neural networks have been developed as generalisations of mathematical models of human neural biology, based on the following assumptions:

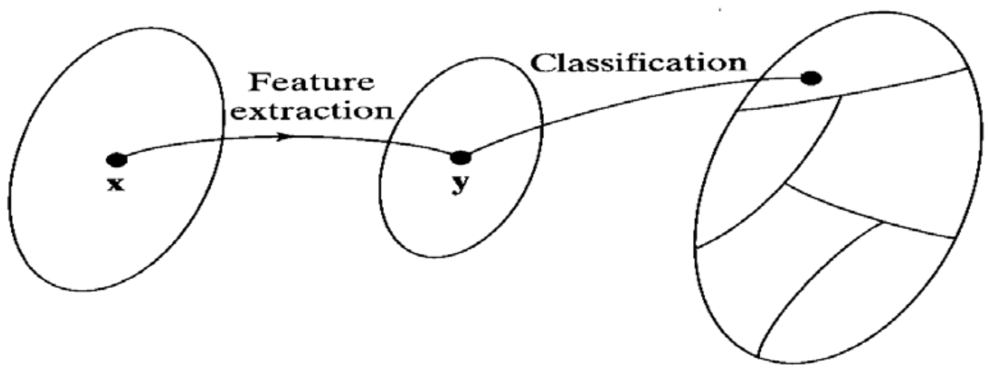
(1) Information processing occurs at many simple elements called neurons;
(2) Signals are passed between neurons over connection links;
(3) Each connection link has an associated weight, which multiplies the signal transmitted;
(4) Each neuron applies an activation function to the input to determine the output signal.



The diagram of a typical artificial neural network

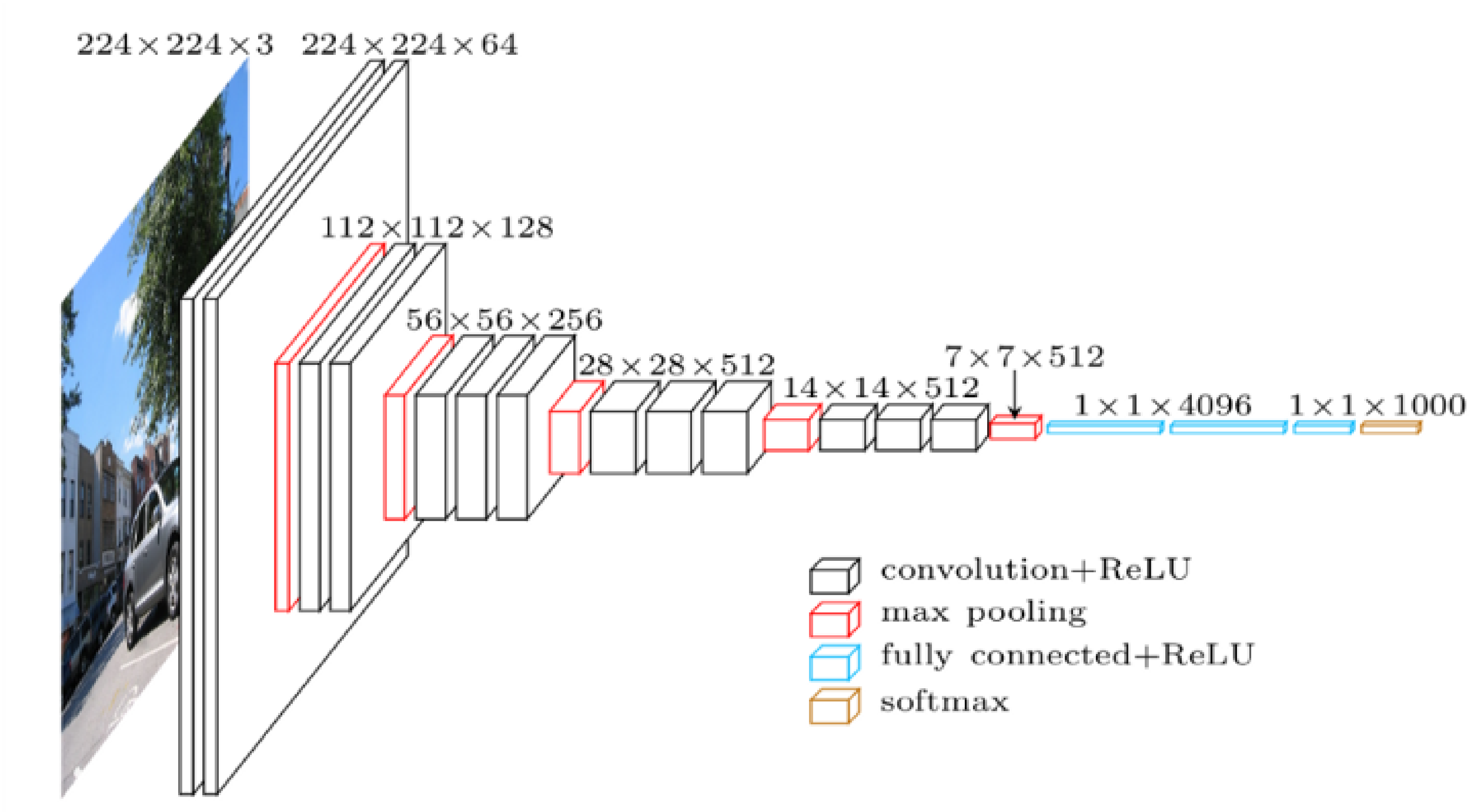
Where are Artificial Neural Networks Used?

1. Pattern Recognition



Feature vector **y** represents main characteristics of the input **x**. For example, the features of a face:

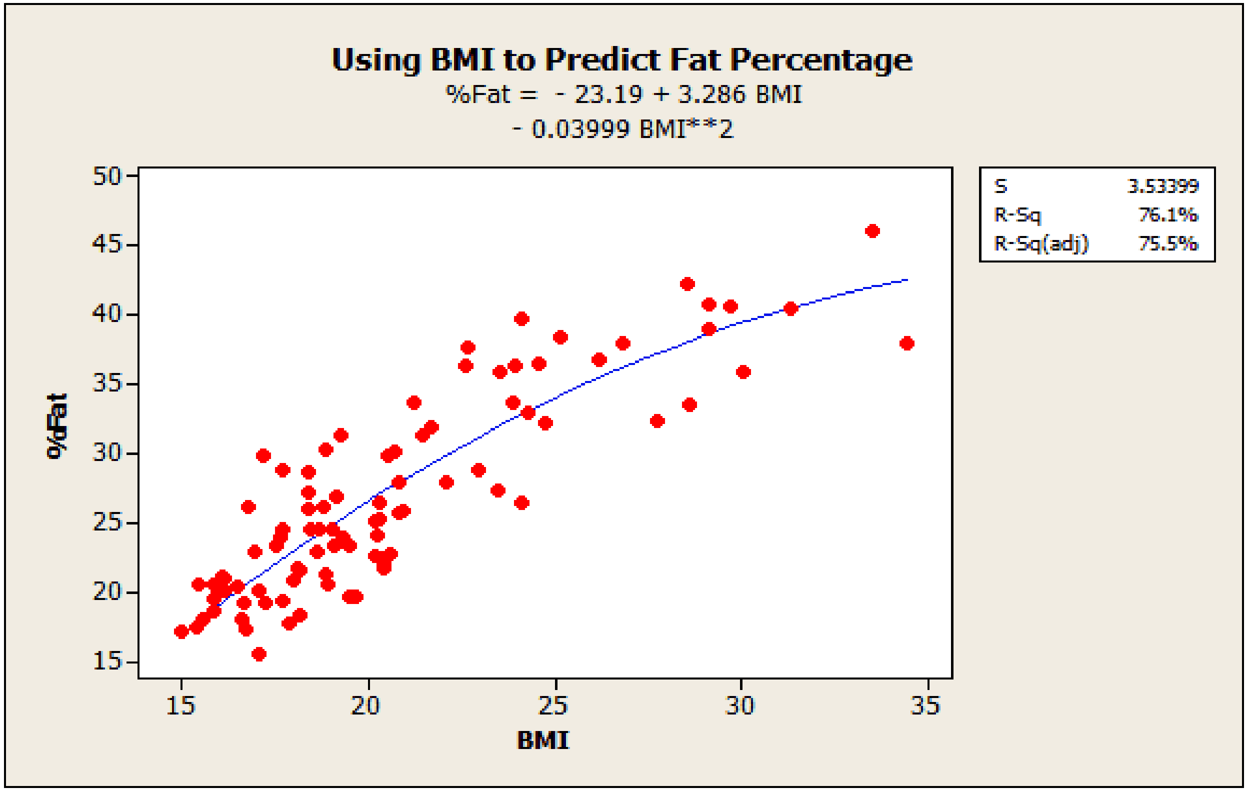
Traditionally, feature extraction and pattern classification are done separately using different techniques. The state-of-the-art technology is to integrate the two parts into one deep neural network, e.g. VGG16 shown below:



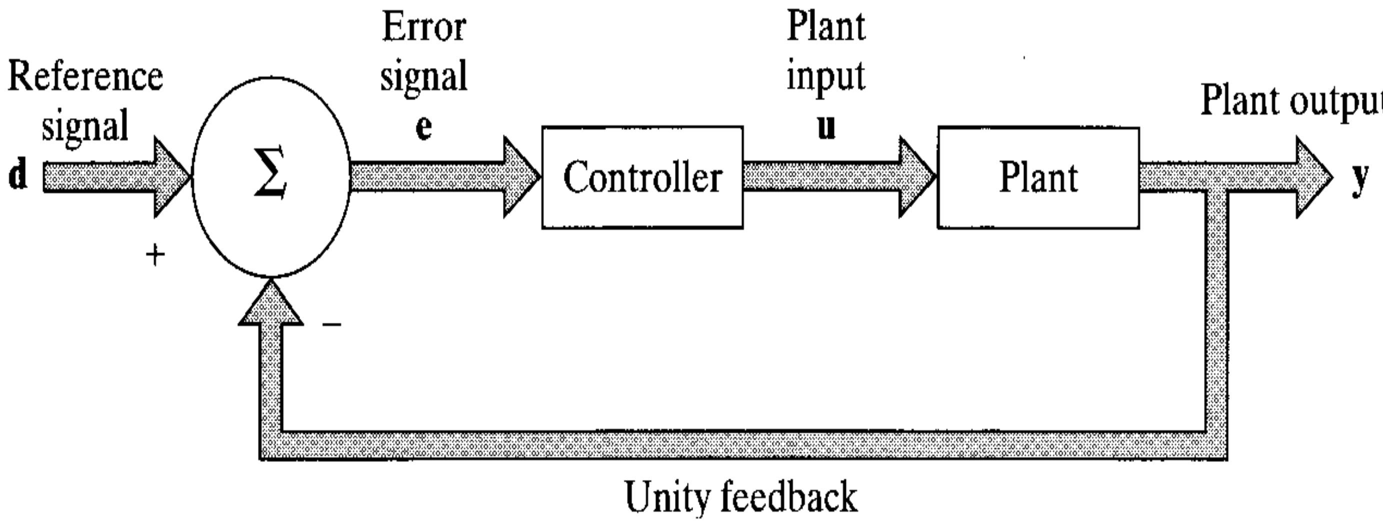
2. Function Approximation

We have a function *f*, but lack of knowledge about the function. However, we have a set of observations:

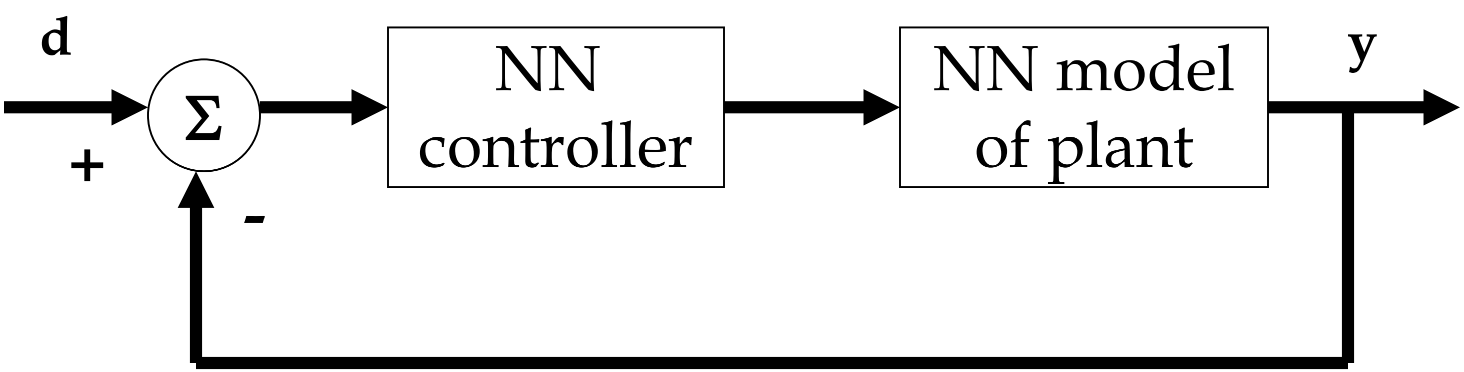
$$\{\mathbf{x}_1, d_1\}, \{\mathbf{x}_2, d_2\}, \ldots, \{\mathbf{x}_N, d_N\}$$



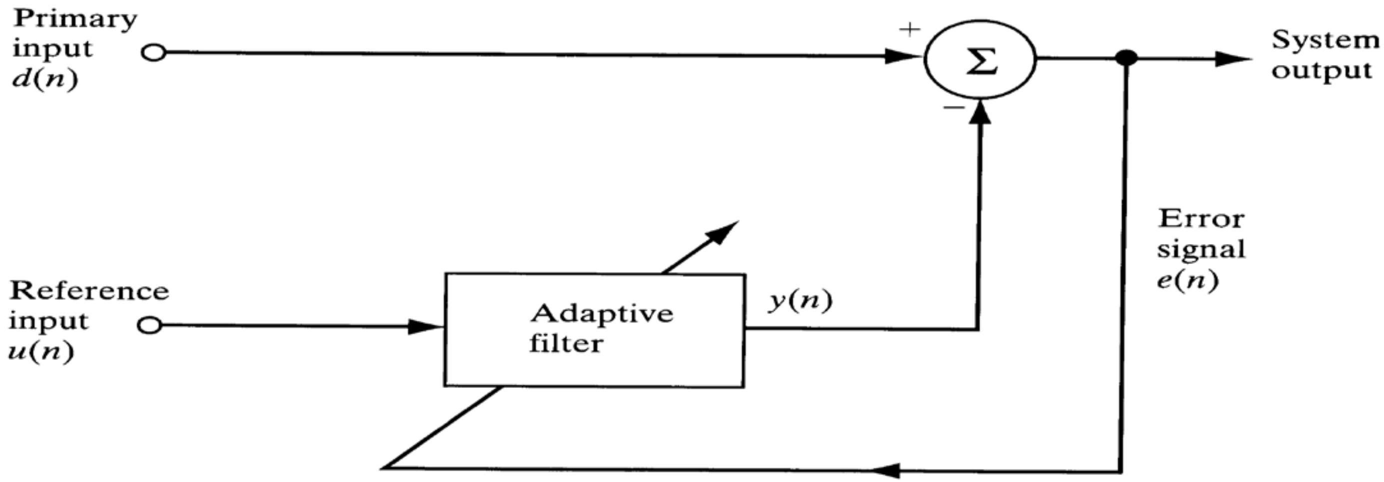
3. Control



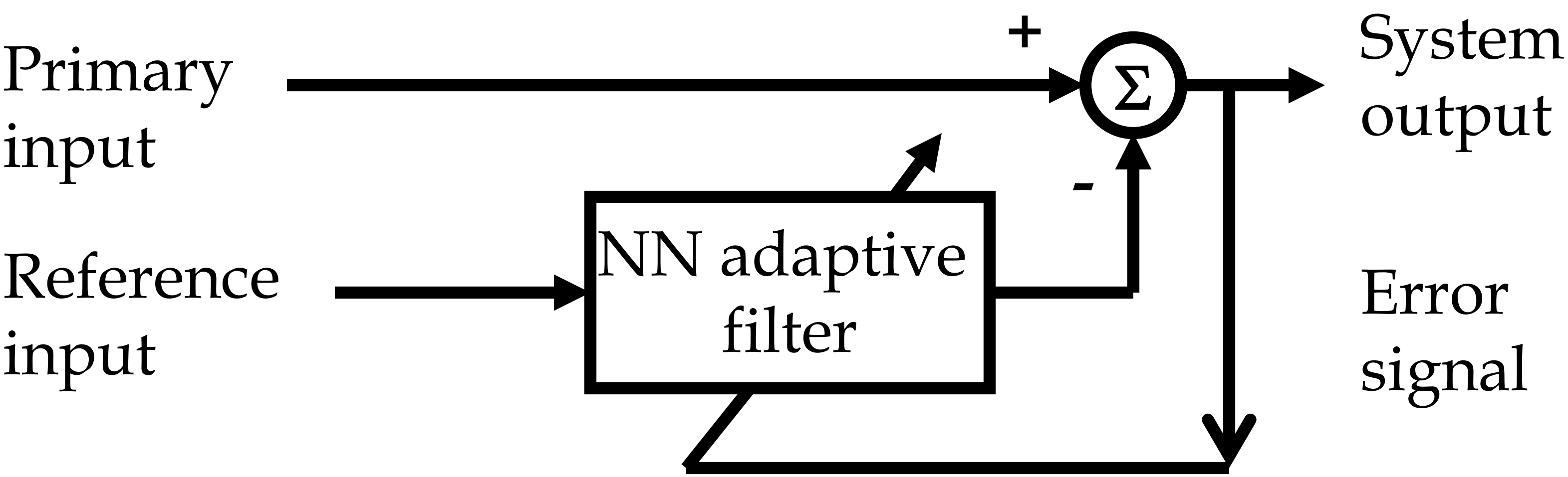
This can be implemented using neural networks as follows:



4. Signal Processing



The neural network implementation is:



5. Business Intelligence

(1) Credit card fraud detection;

(2) Finance data modeling and prediction;

(3) Customer analysis and classification.

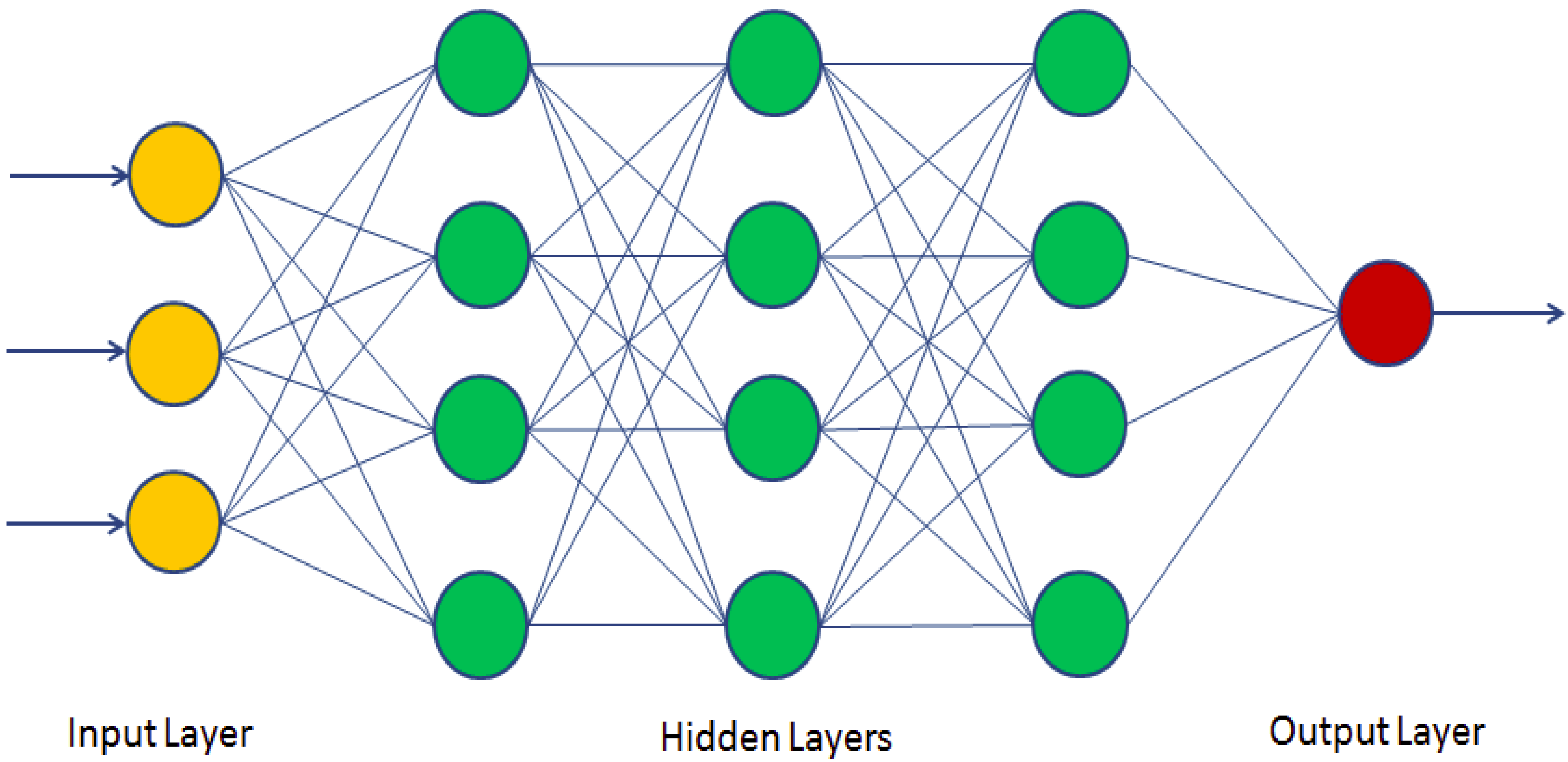
6. Applications in Medicine and Biology

(1) Medical image and signal processing;

(2) Computer assisted diagnosis (CAD);

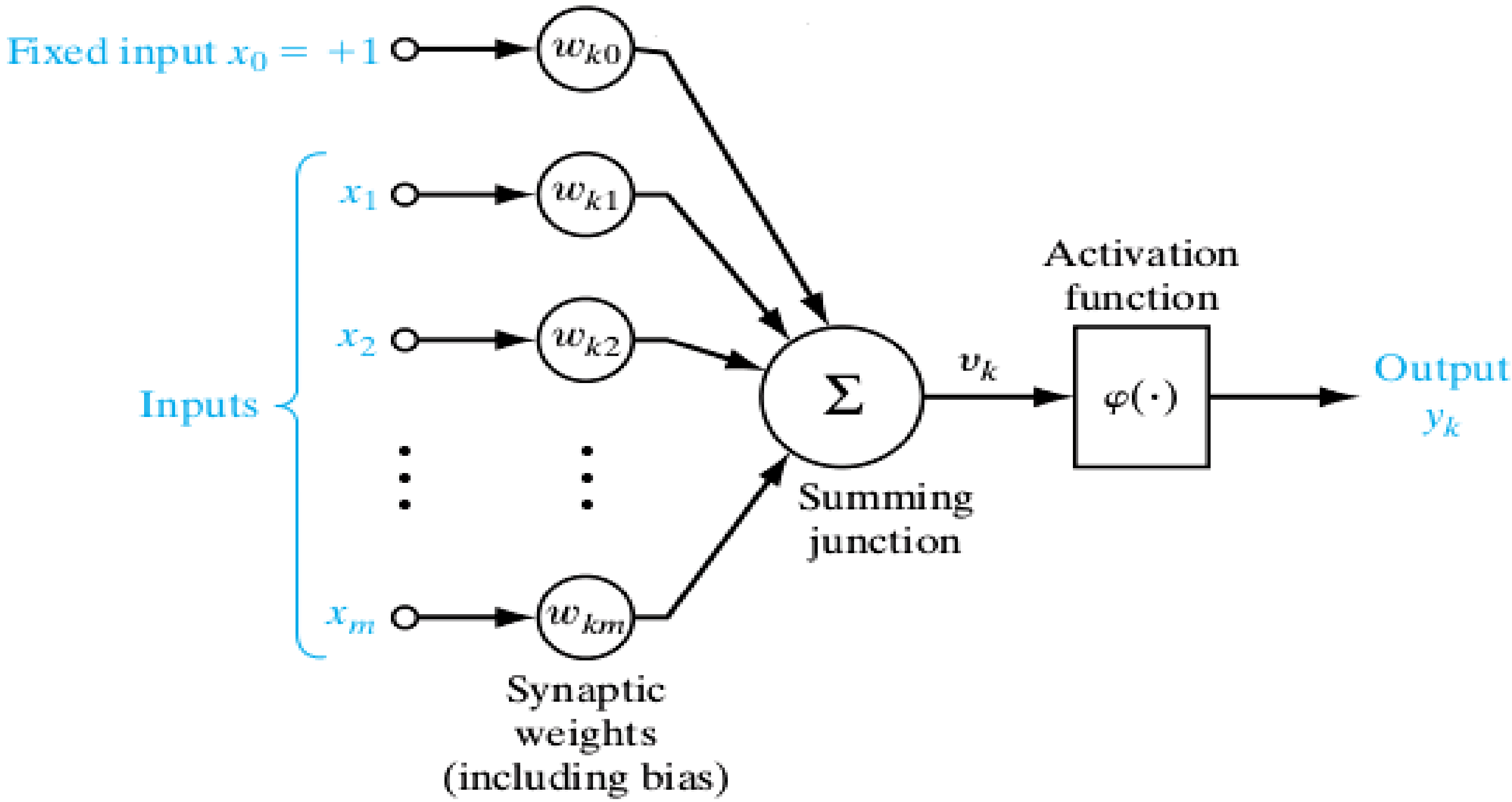
(3) Bioinformatics.

2. Neuron Models and Network Architectures



Neuron Models

A neuron is an information processing unit that is fundamental to the operation of a neural network. The model of a neuron is shown below:



We may identify three basic elements of the neuron model:

(1) *A set of synapses or connecting links*, each of which is characterised by a weight or strength of its own. A signal x_j at the input of synapse j connected to the neuron k is multiplied by the weight w_{kj};

(2) *An adder* for summing the input signal. The adder actually performs a linear combination;

(3) *An activation function* for limiting the amplitude of the neuron output. Typically, the normalised amplitude of the neuron output is in the range [0,1] or [-1 1], depending on the activation function used in the neuron.

In mathematical terms, we can describe the neuron as follows:

$$v_k = \sum_{j=0}^{m} w_{kj} x_j$$

$$y_k = \varphi(v_k)$$

Where

$x_0, x_1, \ldots, x_m$ are the input signals,

$w_{k0}, w_{k1}, \ldots, w_{kp}$ are the synaptic weights of neuron k,

v_k is the activation signal of neuron k,

$\varphi(.)$ is the activation function

y_k is the output of neuron k.

Activation Functions

The activation function defines the output of a neuron in terms of the activation level at its input. We may identify a few types of activation functions:

<u>Binary Function:</u>

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

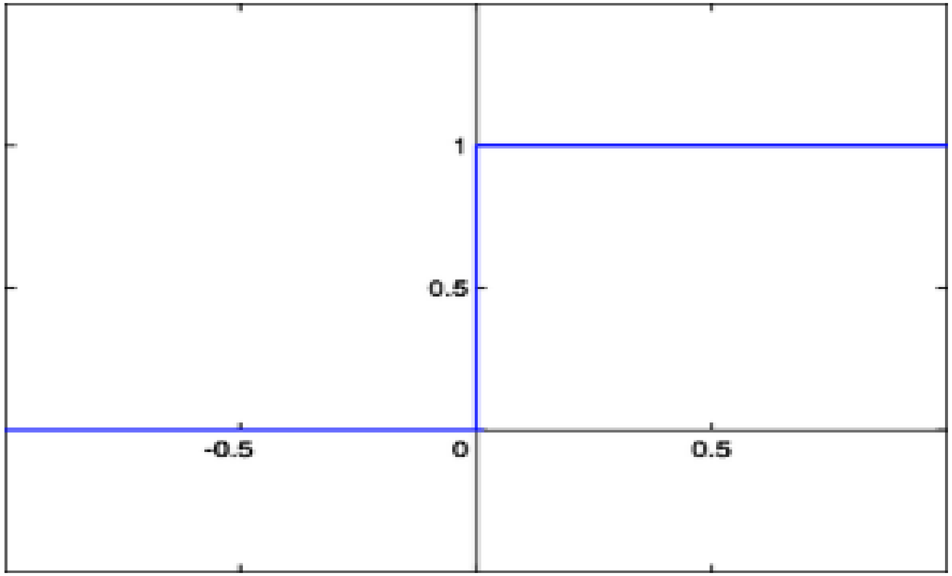
$$f(x) = \begin{cases} 1 & x \geq 0 \\ -1 & x < 0 \end{cases}$$ ---bi-polar binary

When bi-polar binary neuron is used, the model is often referred to as **McCulloch-Pitts (M-P) model**.

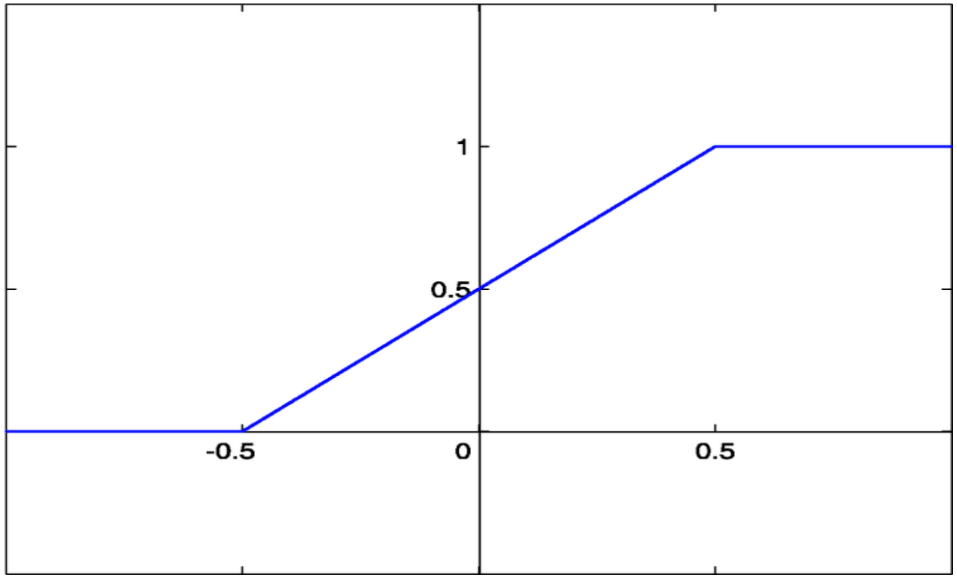
Piecewise-Linear Function

$$f(x) = \begin{cases} 1 & x \geq 0.5 \\ x + 0.5 & -0.5 < x < 0.5 \\ 0 & x \leq -0.5 \end{cases}$$

Where the amplification factor inside the linear region is set to unity.



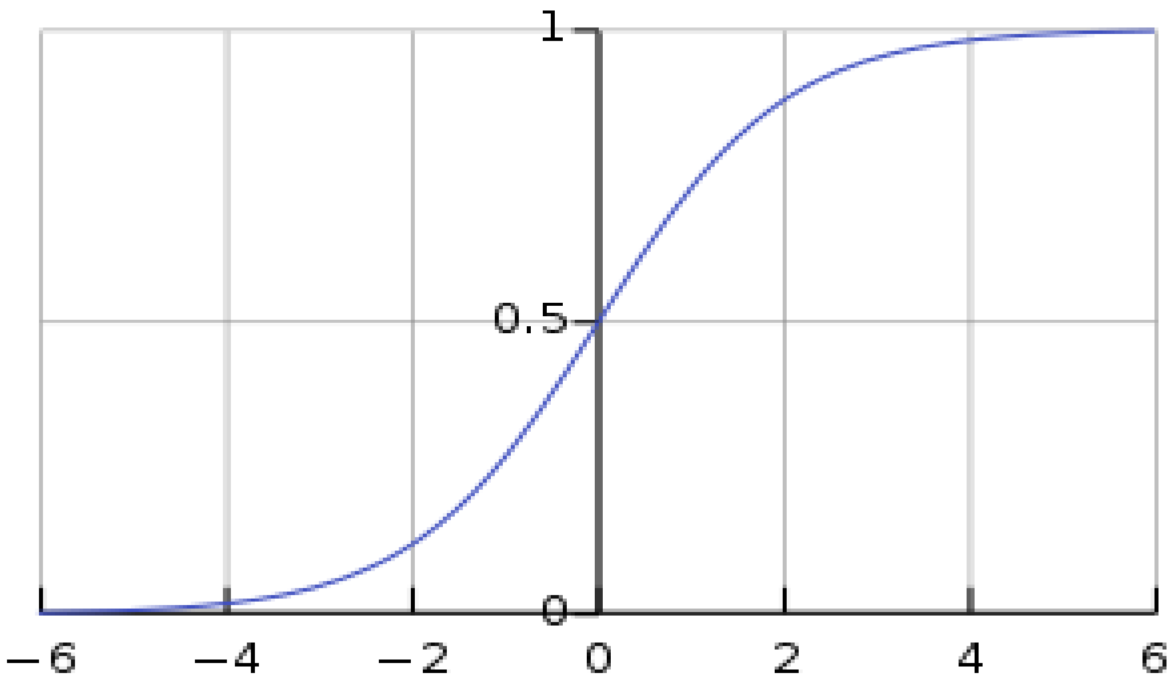
uni-polar binary



piecewise linear

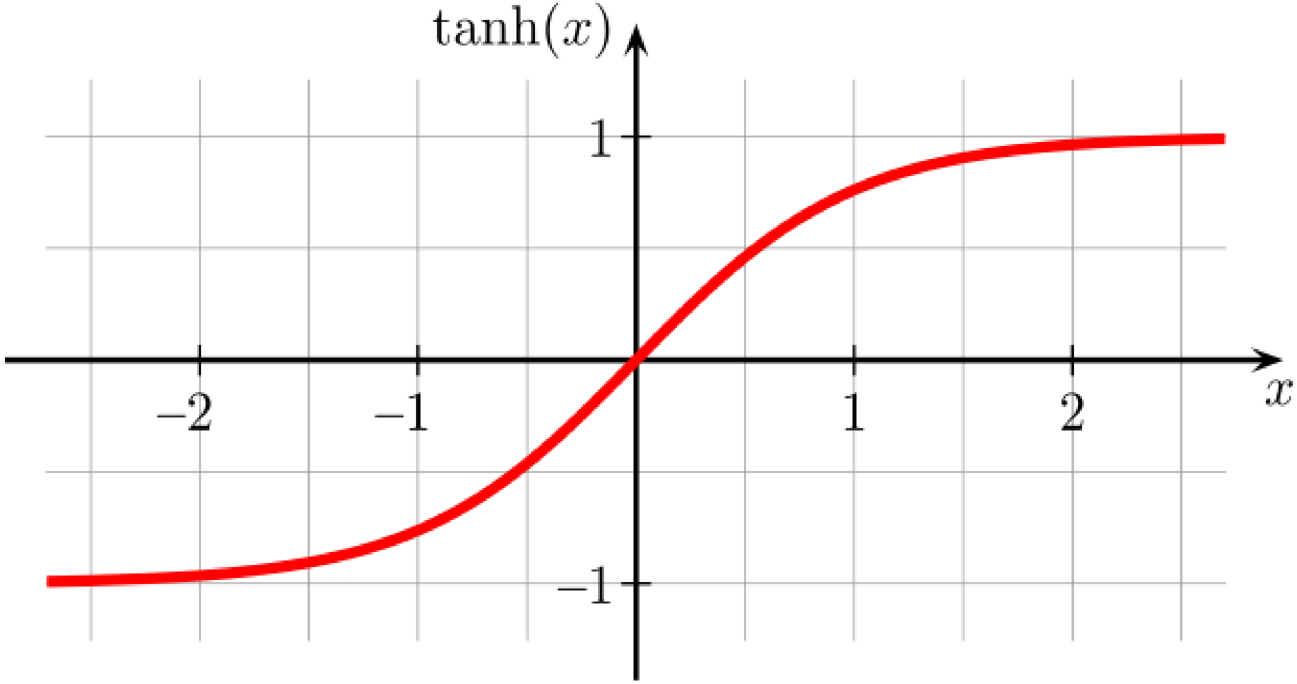
Sigmoid Function

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



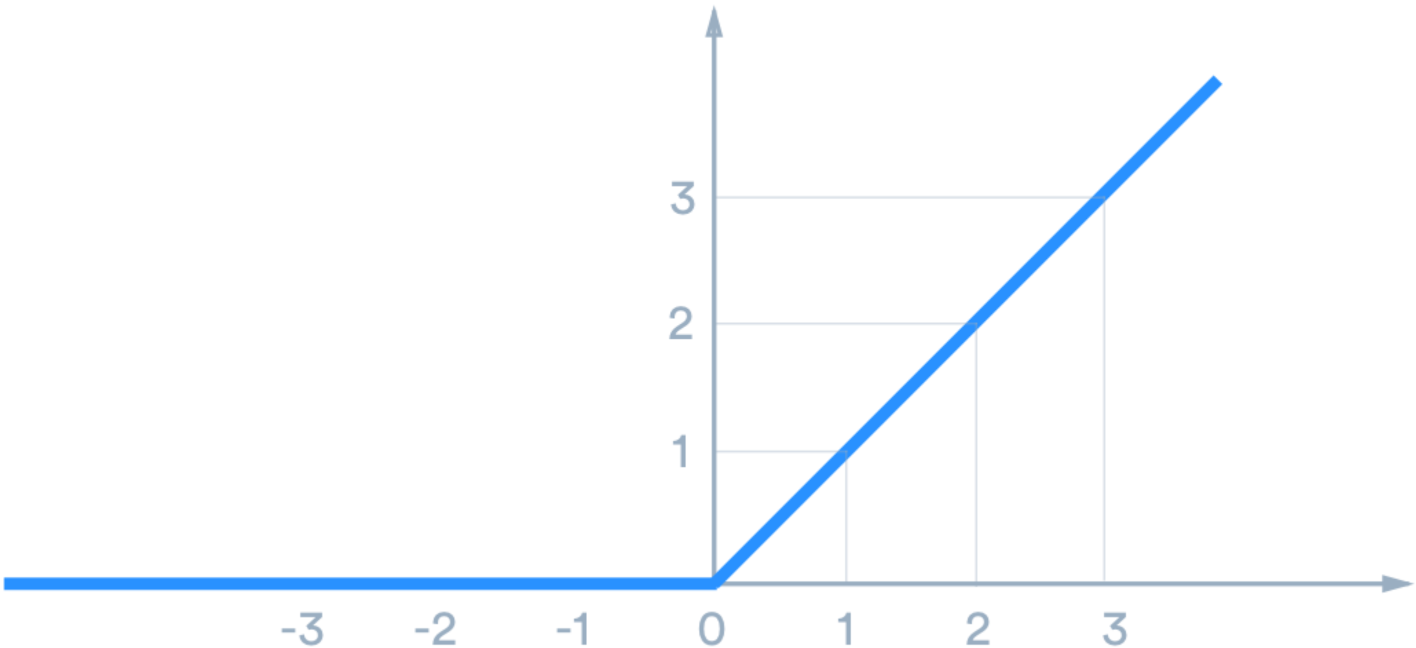
Hyperbolic Tangent Function (tanh)

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$$



Rectified Linear Unit (ReLU)

$$\mathrm{ReLU}(x) = \max\left(0, x\right)$$

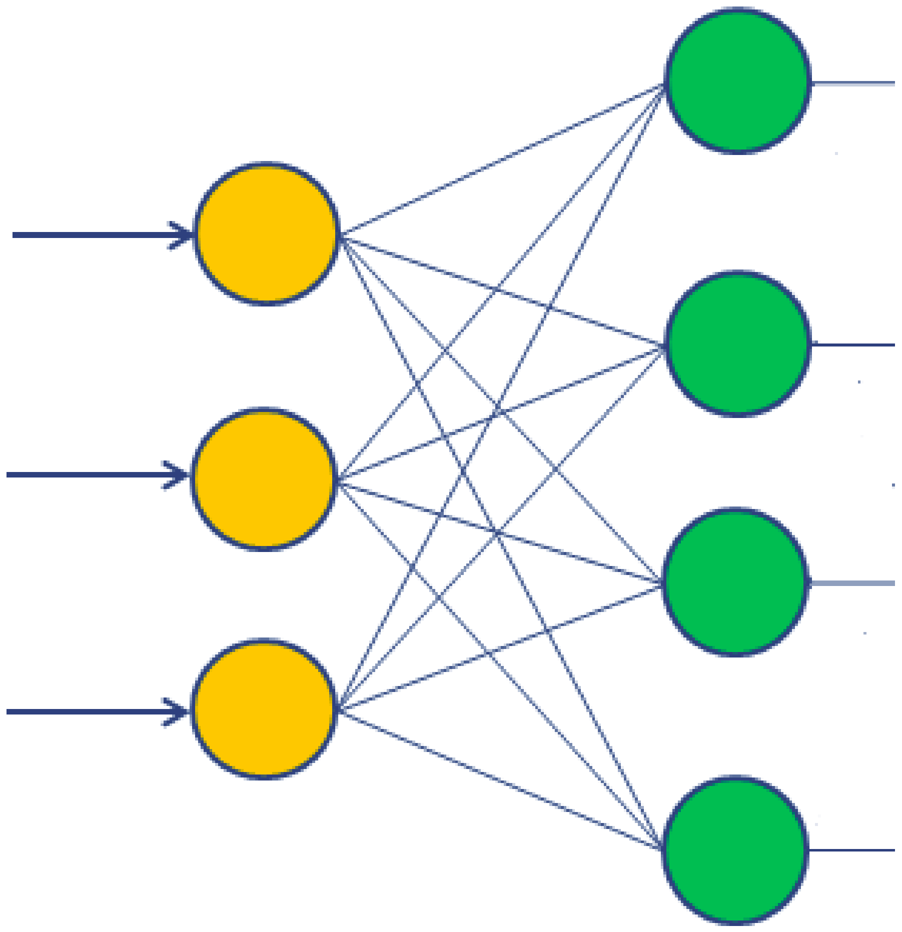


Neural Network Architectures

In general, we may identify two different classes of network architectures.

Feed-Forward Neural Networks

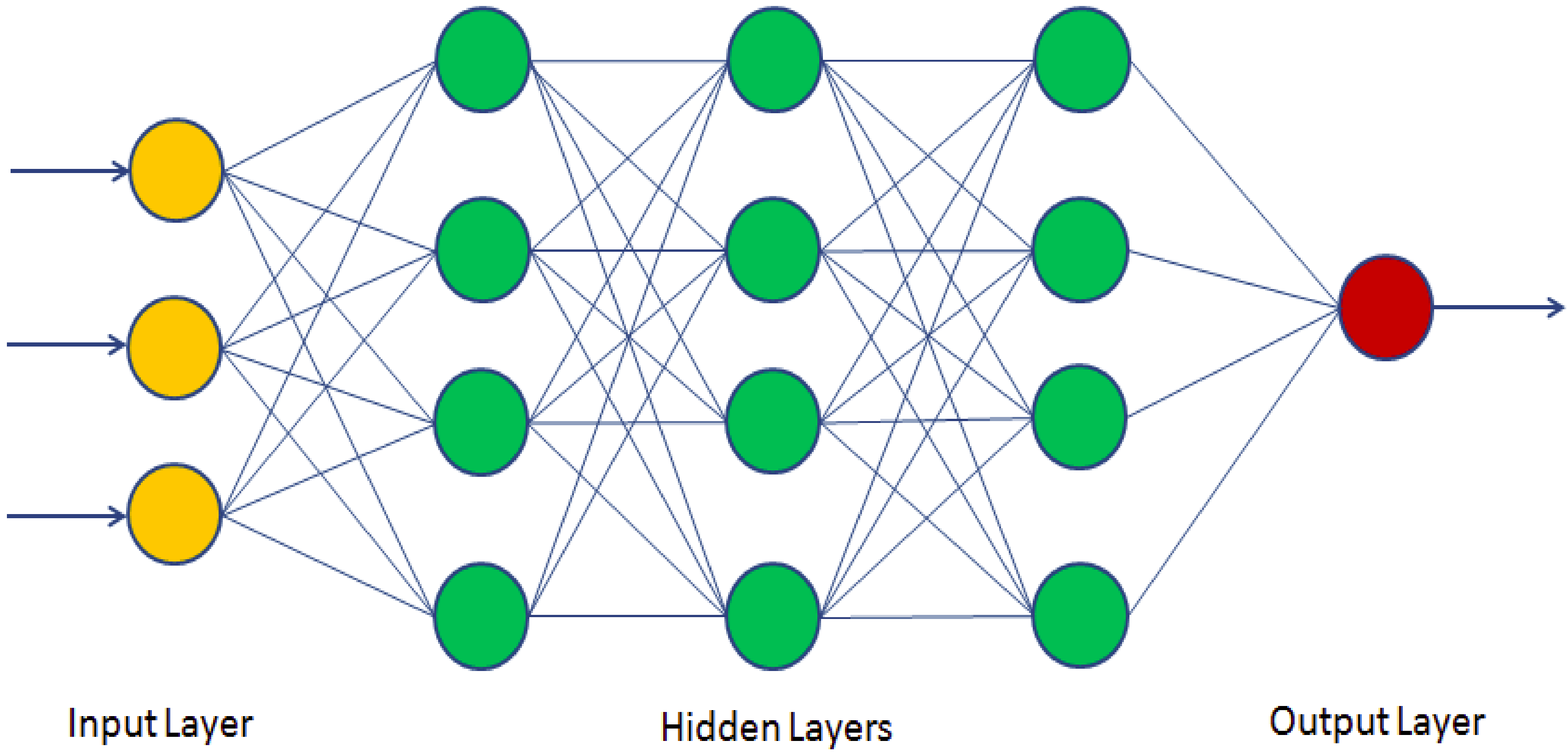
A layered neural network is a network of neurons organized in the form of layers. In the simplest form of a layered network, we just have an input layer of source nodes that projects onto the output layer of neurons. This kind of structure is called single-layer feed-forward network. The term *single layer* refers to the output layer of the computation neurons. In other words, we do not count the input layer of source neurons because the input layer neurons just distribute the input signals to neurons of the output layer and do not perform any computation.



The single layer feed-forward network

The second class of the feed-forward network is the multi-layer neural network. The multi-layer feed-forward neural network distinguishes itself from the single layer network by the presence of one or more hidden layers, whose computational units are the hidden layer neurons (nodes). By adding one or more hidden layer, the network is able to approximate severe non-linearity.

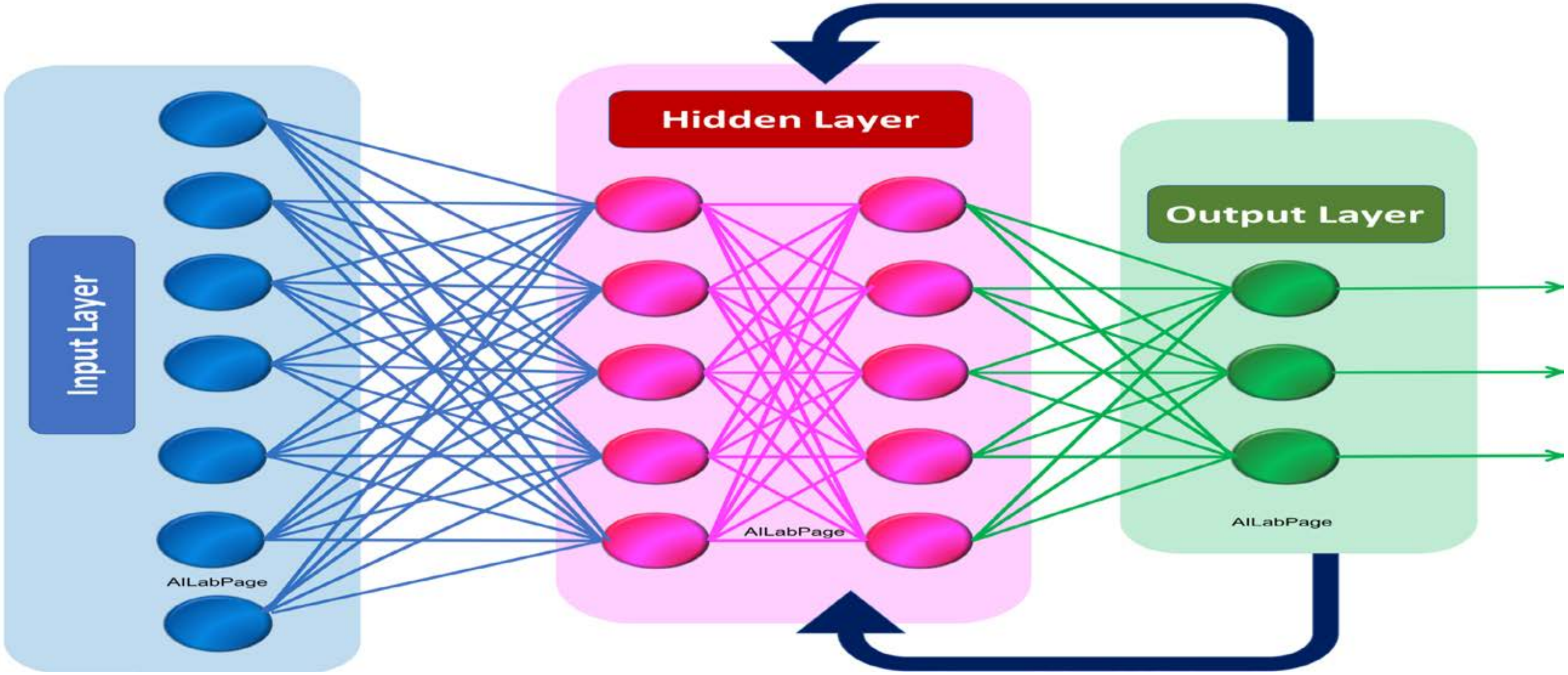
In the following architecture, the hidden layer neuron outputs are used as inputs to the third layer, and so on for the rest of the network. However, it is not necessary to use all external input signals or hidden layer neuron outputs to the input of neurons at the next layer.



Multi-layer feed-forward neural network

Recurrent Neural Networks

A recurrent neural network distinguishes itself from the feed-forward network in that it has at least one feedback loop as shown below:



Recurrent neural network

How to Train Neural Networks?

The major task of a neural network is to learn a model of the world based on the known knowledge of the world. Knowledge of the world consists of two kinds of information:

(1) The known world state, represented by facts about what is and what has been known. This form of knowledge is often referred to as *prior information*.

(2) Observations (measurements) of the world, obtained by sensors designed to probe the world. The observations so obtained provide the pool of information from which neural networks learn. These observations are often referred to as *examples or data or samples*.

Labelled and Unlabelled Data

(a) In labeled samples, each sample representing an input signal is paired with a target output, which is called class label in pattern classification.

Example 1: Object Recognition

Dog

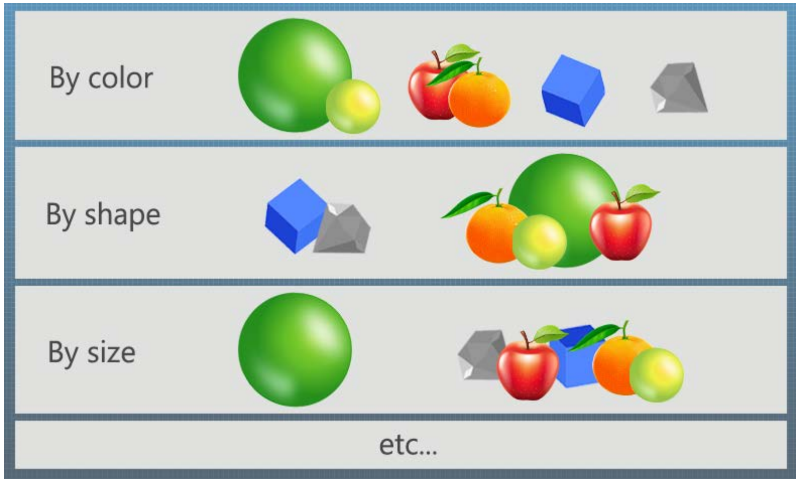
Cat

Example 2: Sentiment Analysis

Sentiment	Tweets
Negative	@united is the worst. Nonrefundable First class tickets? Oh because when you select Global/FC their system auto selects economy w/upgrade.
	@united I will not be flying you again
Neutral	@VirginAmerica my drivers license is expired by a little over a month. Can I fly Friday morning using my expired license?
	@VirginAmerica any plans to start flying direct from DAL to LAS?
Positive	@VirginAmerica done! Thank you for the quick response, apparently faster than sitting on hold ;)
	@united I appreciate your efforts getting me home!

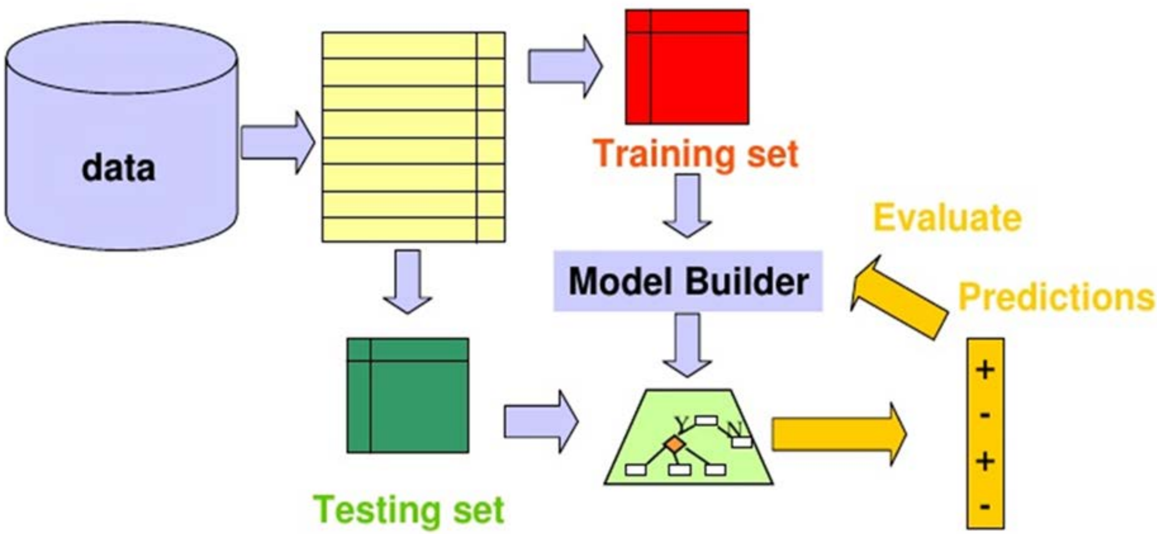
(b) In unlabeled samples, just the input signal is available, the target output is unknown. For example, a basket of fruits:

The fruits can be grouped based on different features:



Training and Testing Data

When a set of samples is provided, we often divide the data into two subsets. One subset is called *training set*, and another subset is called *testing set*.



The training set is used to train the model (neural network), and the testing set is used to evaluate the model learned.

The design of a neural network may be done as follows:

(1) First, an appropriate architecture is selected for the neural network. The training set is then used to train the network using a suitable algorithm. This phase of the neural network design is called *training* or *learning*.

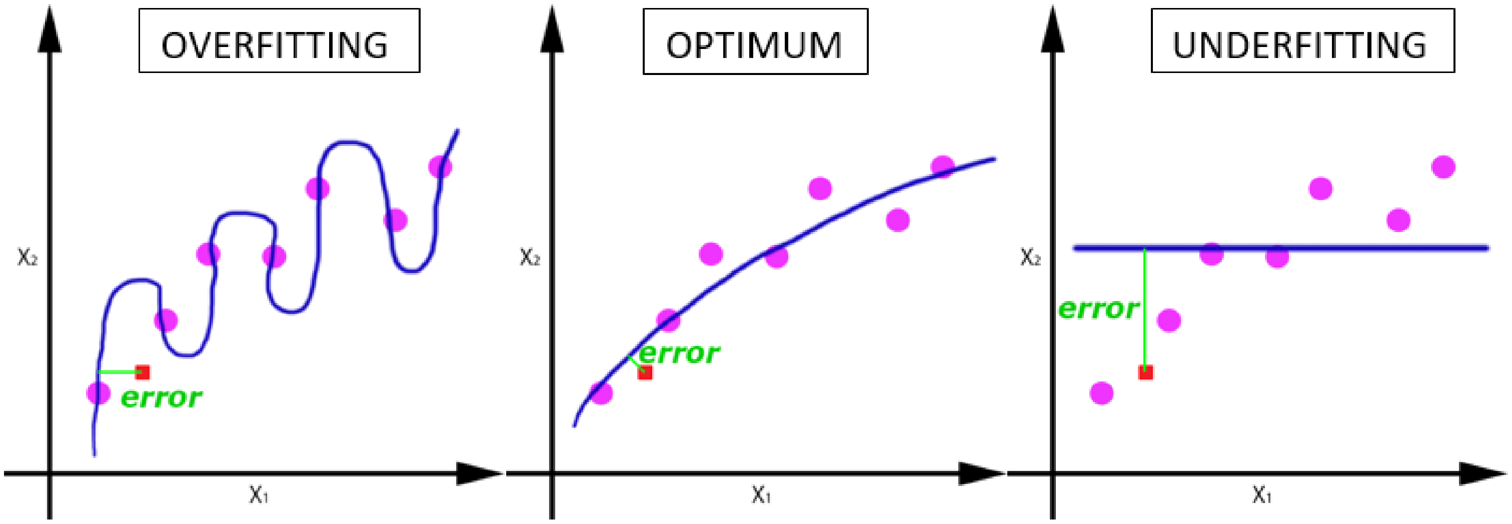
(2) Second, the performance of the trained network is evaluated with the testing set, which are examples that are not used in the training stage. The second phase of the design is called *testing*.

Quite often, part of the training set is used as *validation set*, which is used for setting hyper-parameters of the model or the leaning algorithms.

Overfitting and Underfitting

If a network performs well on the training data but very badly on the testing set, the network might be *over-trained (overfitting)*.

On the other hand, if the network performs bad on the training data, the network might be *under-trained (underfitting)*. An under-trained network also performs badly on the testing set.



3. Neural Network Learning

Among the many interesting properties of a neural network, the property that is of primary significance is the ability of *learning* from its environment. What is *learning*?

In the context of neural networks, *learning* is defined as a process by which the free parameters of a neural network are adapted through a continuing process of stimulation by the environment. The type of *learning* is determined by the manner in which the parameter changes take place.

The definition implies that:

(1) The network is stimulated by the environment;
(2) The network changes as a result of stimulation;

(3) The network responds to the environment in a new way after the change.

Next we will introduce three basic *learning rules* and two *learning paradigms*.

Three learning rules:

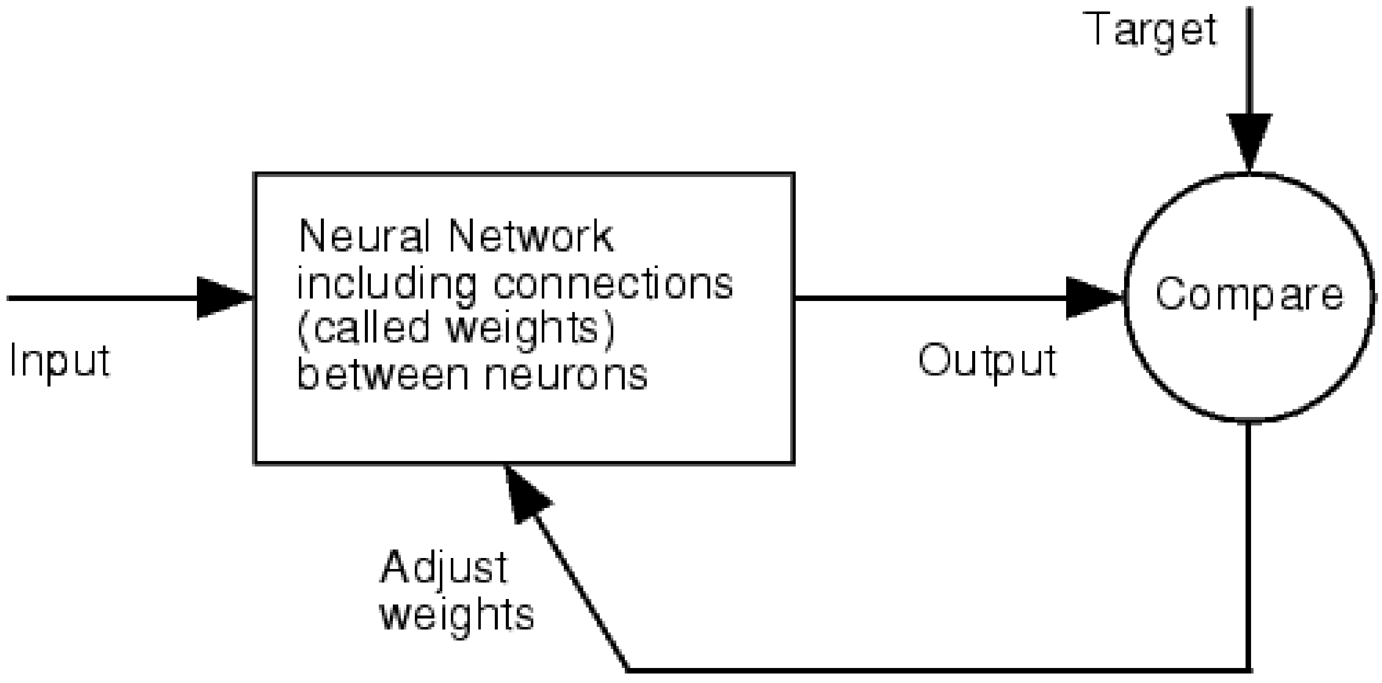
(1) Error-correction learning;
(2) Hebbian learning;
(3) Competitive learning.

Two learning paradigms:

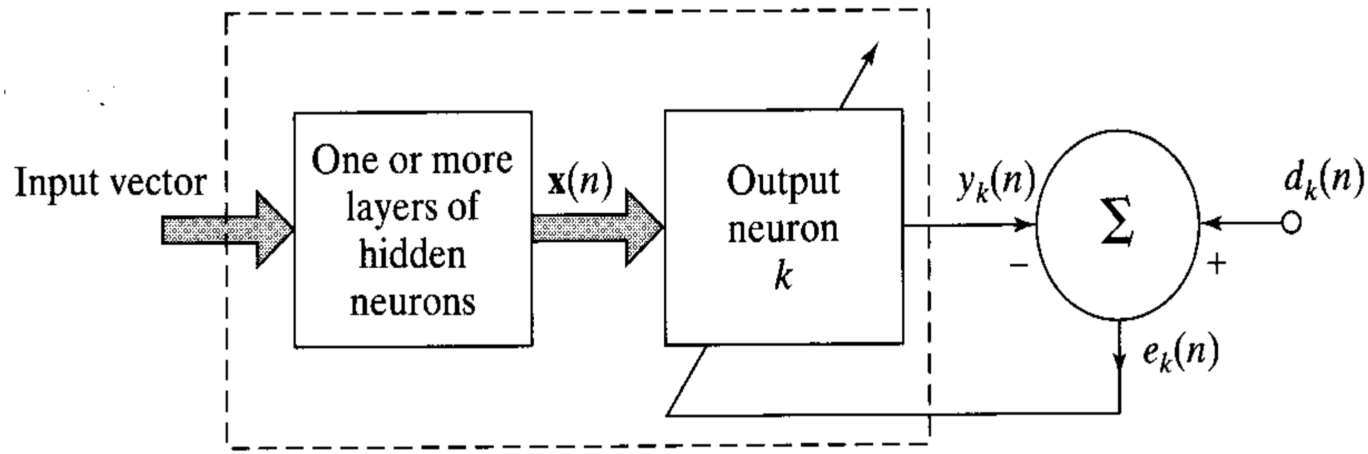
(1) Supervised learning;
(2) Unsupervised learning.

Error-correction Learning

The learning of a neural network is the process of determining the weights on all connections of the network. In error correction learning, the weight adjustment is based on the error, which is defined as the difference between the network output and the target value:



To illustrate the error-correction learning rule, let's consider the simple case that the output layer of a feed-forward neural network consists of only one neuron, say neuron *k*, as shown below:



Where neuron *k* is driven by a signal vector $\mathbf{x}$(n) produced by one or more layers of hidden neurons. The argument *n* denotes the index. The output and desired response of neuron *k* at *n* are denoted by $y_k(n)$ and $d_k(n)$ respectively.

Typically, the actual response is different from the desired response, and the difference between them, *i.e.* error signal, is defined as:

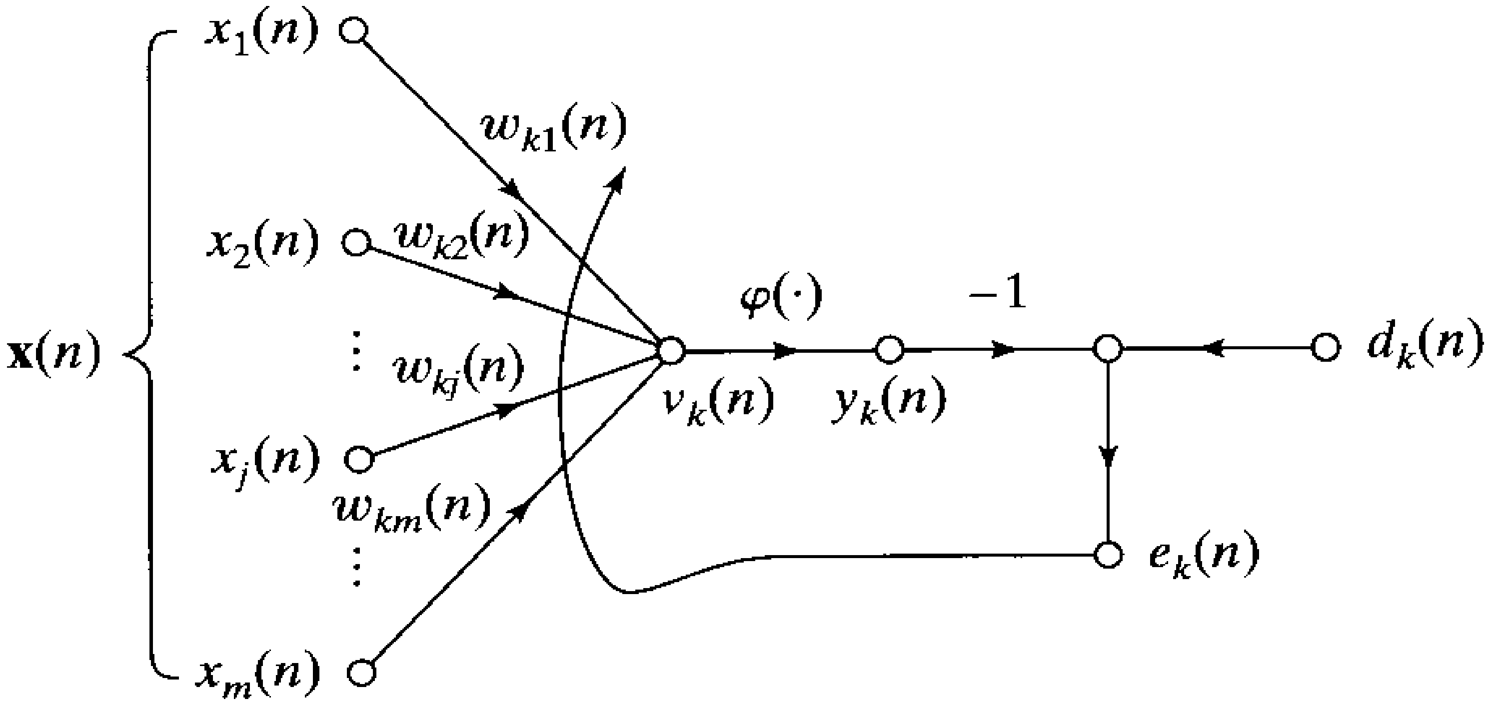
$$e_k(n) = d_k(n) - y_k(n)$$

The error signal actuates a control mechanism, based on which weights are adjusted. The adjustment of weights are designed to make the output signal come closer to the desired signal in a step-by-step manner. This goal is achieved by minimizing the cost function:

$$J(n) = \frac{1}{2} e_k^2(n) = \frac{1}{2}[d_k(n) - y_k(n)]^2$$

Where J(n) is the instantaneous value of error energy. The step-by-step adjustments to the weights of neuron k are continued until the weights are stabilised. At that point, the learning process finishes. Minimisation of cost function of J(n) leads to a learning rule commonly referred to as the *Widrow-Hoff rule*.

Let's consider the signal-flow graph of the output neuron k.



Let $w_{kj}(n)$ denote the value of weight w_{kj} of neuron *k* excited by signal $x_j(n)$ at step *n*. According to the Widrow-Huff rule, the adjustment applied to weight w_{jk} at step *n* is defined as:

$$\Delta w_{kj}(n) = \eta e_k(n) x_j(n)$$

where η is a positive constant that determines the *rate of learning*. The adjustment of the weight is proportional to the product of the error signal and the corresponding input signal.

Thus, the updated value of the weight w_{kj} is determined by the following rule:

$$w_{kj}(n+1) = w_{kj}(n) + \Delta w_{kj}(n)$$

As shown in the above diagram, an important feature of the error-correction learning is that it employs a closed-loop feedback mechanism. From feedback control theory, we know the stability of such a system is determined by those parameters that constitute the feedback loops of the system. In the error-correction learning, we only have a single feedback loop, and one of those parameters of particular interest to us is the learning rate parameter η. It is therefore important to carefully select η to ensure stability or convergence of the iterative learning process. The choice of parameter η also has a profound influence on the accuracy and other aspects of the learning process. It should be said that learning rate η plays a key role in determining the performance of the error-correction learning in practice.

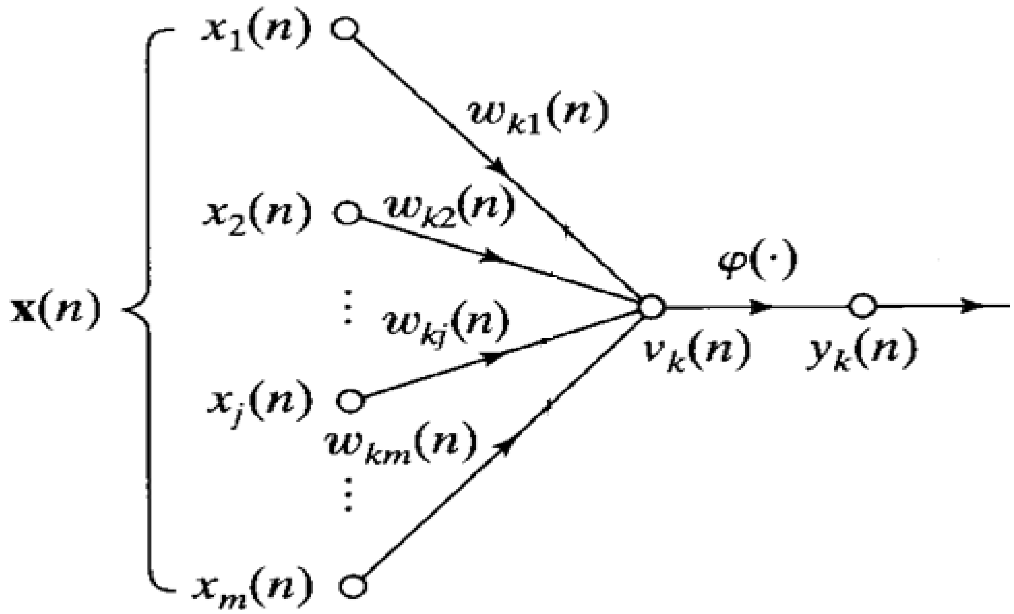
Hebbian Learning

Hebbian learning rule can be stated as:

(1) If two neurons on either side of a synapse, *i.e.* connection, are activated simultaneously, then the strength of that synapse is selectively increased.

(2) If two neurons on either side of a synapse are activated asynchronously, then that synapse is selectively weakened or eliminated

To formulate Hebbian learning in mathematical terms, consider a weight w_{kj} from input signal x_j to neuron k. The adjustment to the weight at time instant n is expressed in the general form:



$$\Delta w_{kj}(n) = f[x_j(n), y_k(n)]$$

Where f is a function of post-synaptic and pre-synaptic signals. The above formula has many forms, but only the simplest form is introduced here.

The simplest form of Hebbian learning is described by:

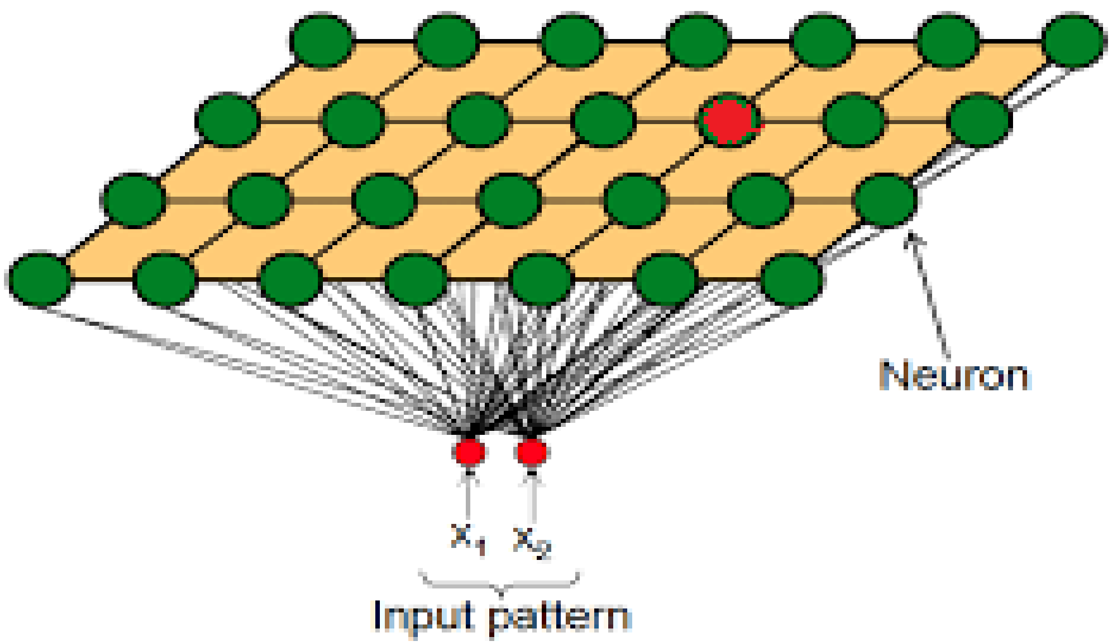
$$\Delta w_{kj}(n) = \eta y_k(n) x_j(n)$$

Where η is a positive constant that determines the *rate of learning*.

The Hebbian learning rule clearly emphasizes the correlation nature of a Hebbian synapse.

Competitive Learning

As its name implies, in competitive learning, the output neurons of a neural network compete among themselves for being the one to be active. Whereas in a neural network based on Hebbian learning several output neurons may be active simultaneously, in the case of competitive learning only one neuron is active.



There are three basic elements in a competitive learning rule:

(1) *A set of neurons* that are all the same except for some randomly distributed synaptic weights, and which therefore respond differently to a given set of input patterns;

(2) *A limit* imposed on the weights;

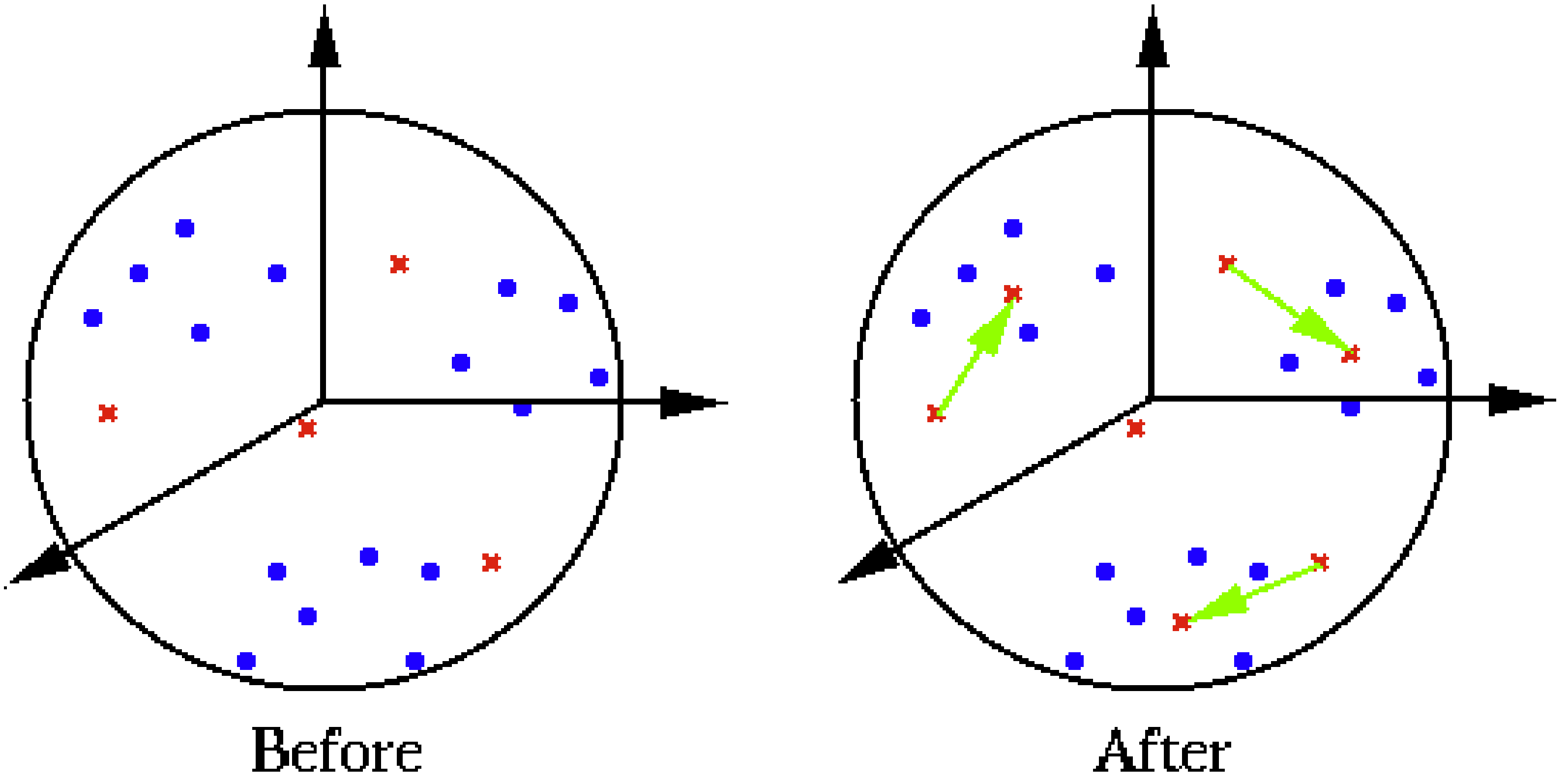
(3) *A mechanism* that permits the neurons to compete for the right to respond to a given subset of input such that only one neuron is active at a time. The neuron that wins the competition is called a *winner-takes-all* neuron.

Whether a neuron will be active depends on the activation signal received. The neuron that receives the largest activation will win and will be active.

The weights of neuron k are updated in the following way:

$$\Delta w_{kj} = \begin{cases} \eta(x_j - w_{kj}) & \text{if neuron k wins the competition} \\ 0 & \text{if neuron k loses the competition} \end{cases}$$

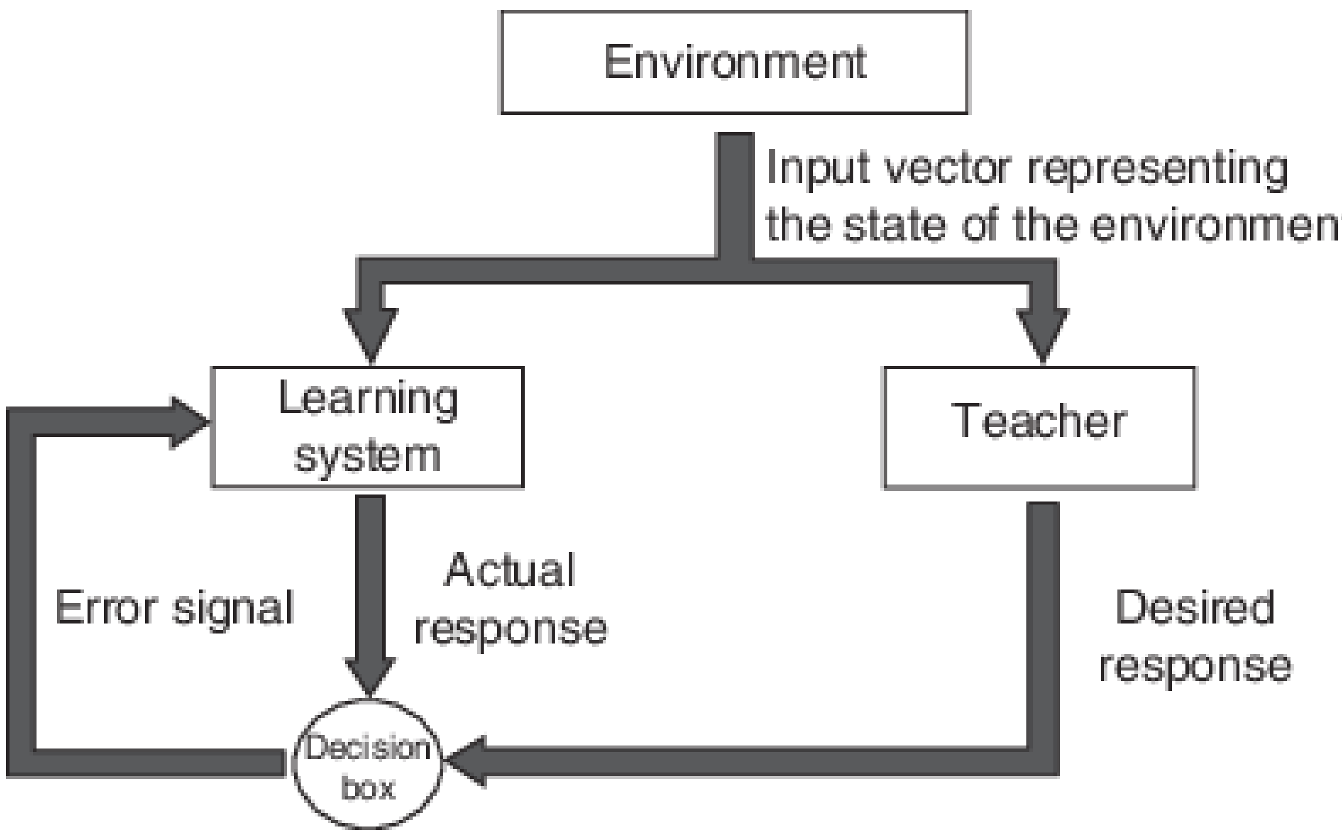
Where η is the learning rate parameter. The effect of the learning rule is to move the weight vector $\mathbf{w}_k$ of winning neuron k toward the input pattern $\mathbf{x}$ as shown below.


Before
After

Supervised Learning

During the supervised learning of a neural network, an input is applied to the network, and a response of the network is obtained. The response is compared with a target value. If the actual response differs from the target value, the neural network generates an error signal, which is then used to compute the adjustments that should be made to the network's synaptic weights so that the actual response matches the target value. In other words, the error is minimised, possibly to zero. Since the minimisation process requires a teacher (supervisor), this kind of training is named *supervised learning*.

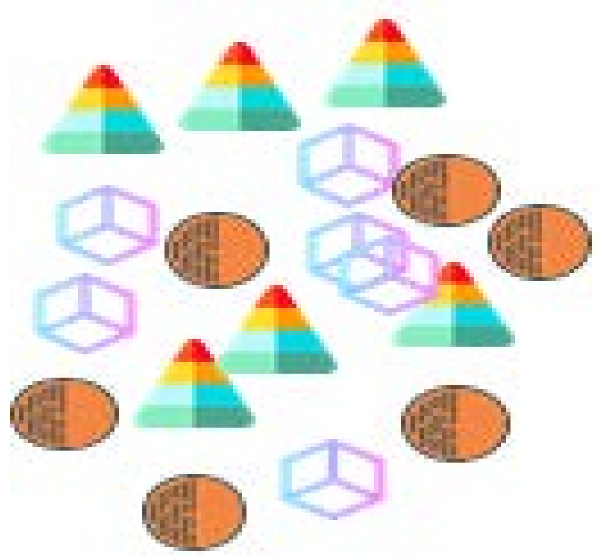
The notion of teacher comes from biological observations. For example, when learning a language, we hear the sound of a word from a teacher. The sound is stored in the memory banks of our brain, and we try to reproduce the sound. When we hear our own sound, we mentally compare it (actual response) with the stored sound (desired response) and note the error. If the error is large, we try again and again until it becomes significantly small.



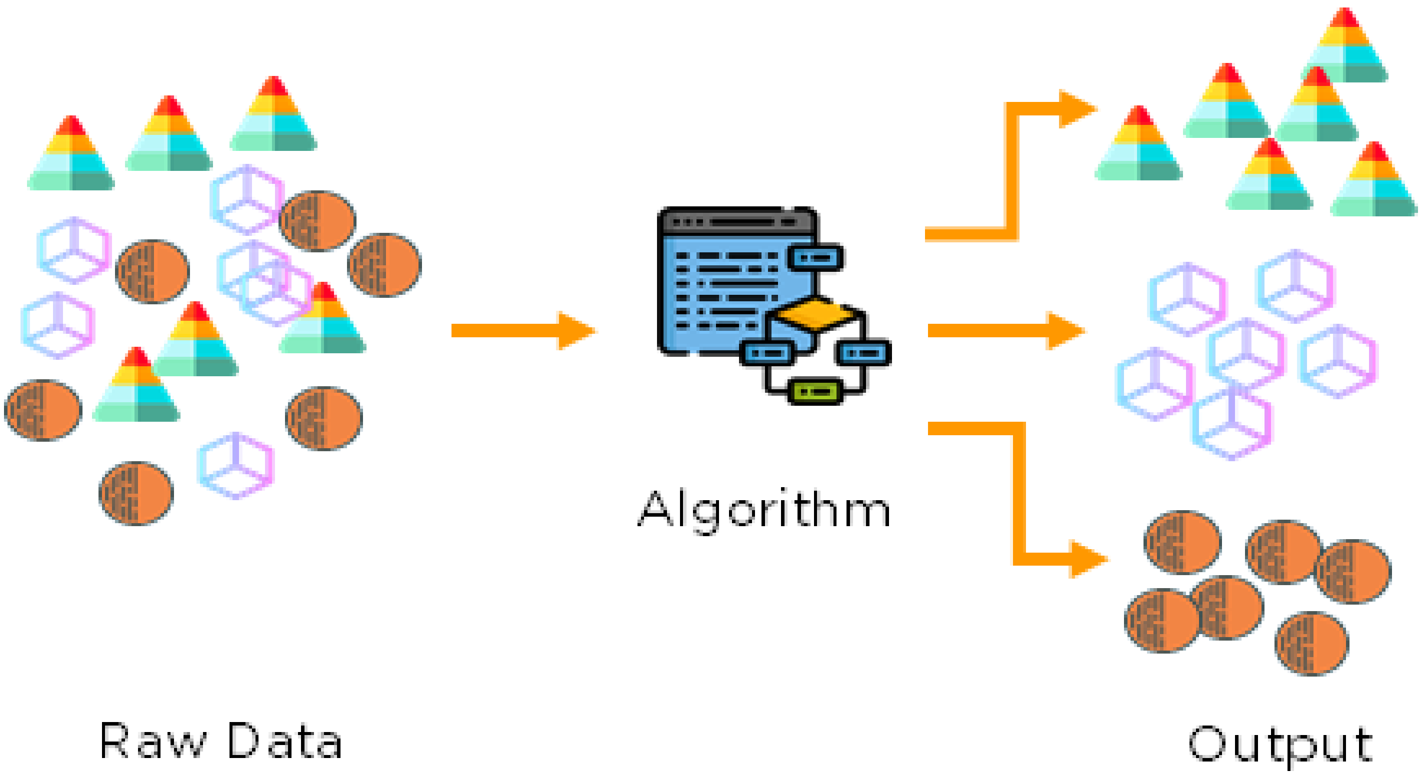
Unsupervised Learning

In contrast to supervised learning, *unsupervised learning* does not require a teacher, *i.e.* there is no target value. The goal of unsupervised learning is to group data based on their intrinsic properties.

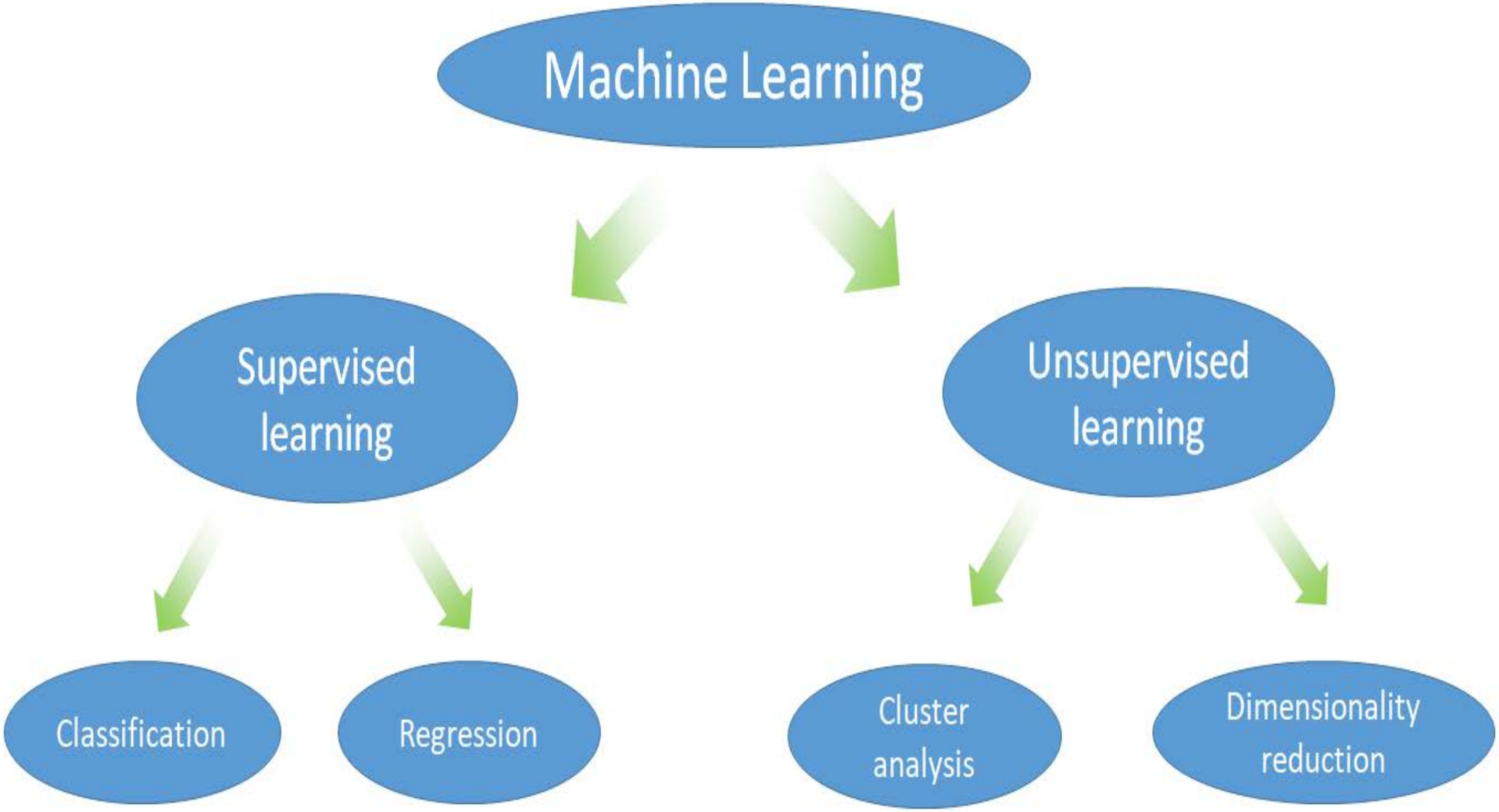
For example, a set of objects (without knowing their names):



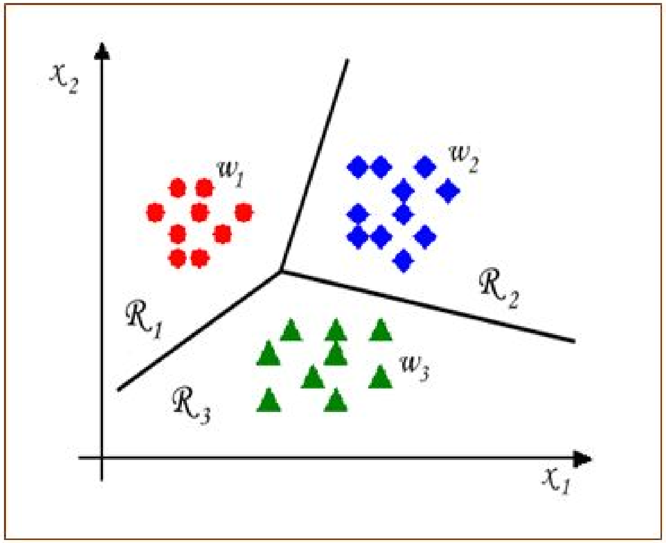
The unsupervised learning algorithm organizes the objects into three groups:

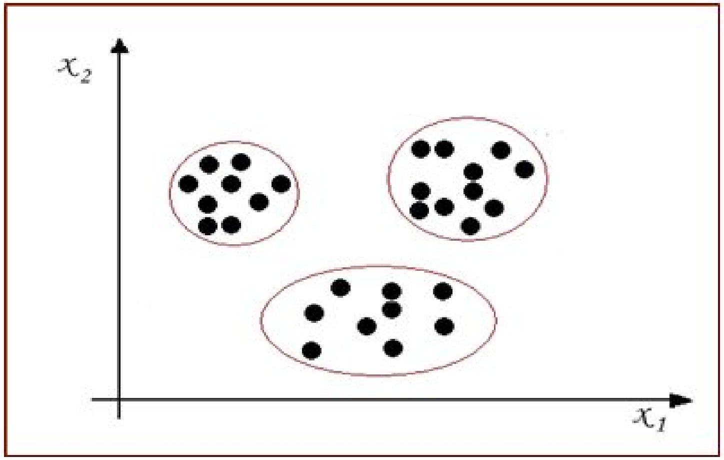


Supervised vs Unsupervised Learning



Classification vs Clustering





Given some training patterns from each class, the goal is to construct decision boundaries or to partition the feature space

Given some patterns, the goal is to discover the underlying structure (categories) in the data based on inter-pattern similarities