UNIT 1

Introduction

- Introduction to soft computing;
- introduction to biological and artificial neural network,
- introduction to fuzzy sets and fuzzy logic systems.

Introduction

Basics of Soft Computing

What is Soft Computing?

- The idea of soft computing was initiated in 1981 when Lotfi A. *Zadeh* published his first paper on soft data analysis “What is Soft Computing”, Soft Computing. Springer-Verlag Germany/USA 1997.]

- Zadeh, defined Soft Computing into one multidisciplinary system as the fusion of the fields of Fuzzy Logic, Neuro-Computing, Evolutionary and Genetic Computing, and Probabilistic Computing.

- Soft Computing is the fusion of methodologies designed to model and enable solutions to real world problems, which are not modeled or too difficult to model mathematically.

- The aim of Soft Computing is to exploit the tolerance for imprecision, uncertainty, approximate reasoning, and partial truth in order to achieve close resemblance with human like decision making.
- The Soft Computing – development history

SC	**=**	***EC***	**+**	***NN***	**+**	***FL***
Soft		**Evolutionary**		**Neural**		**Fuzzy**
Computing		**Computing**		**Network**		**Logic**
Zadeh		**Rechenberg**		**McCulloch**		**Zadeh**
1981		**1960**		**1943**		**1965**

EC	**=**	***GP***	**+**	***ES***	**+**	***EP***	**+**	***GA***
Evolutionary		***Genetic***		***Evolution***		***Evolutionary***		***Genetic***
Computing		***Programming***		***Strategies***		***Programming***		***Algorithms***
Rechenberg		**Koza**		**Rechenberg**		**Fogel**		**Holland**
1960		**1992**		**1965**		**1962**		**1970**

Definitions of Soft Computing (SC)

Lotfi A. Zadeh, 1992 : "Soft Computing is an emerging approach to computing which parallel the remarkable ability of the human mind to reason and learn in a environment of uncertainty and imprecision".

The Soft Computing consists of several computing paradigms mainly :

Fuzzy Systems, Neural Networks, and Genetic Algorithms.

- Fuzzy set : for knowledge representation via fuzzy If – Then rules.
- Neural Networks : for learning and adaptation
- Genetic Algorithms : for evolutionary computation

These methodologies form the core of SC.

Hybridization of these three creates a successful synergic effect; that is, hybridization creates a situation where different entities cooperate advantageously for a final outcome.

Soft Computing is still growing and developing.

Hence, a clear definite agreement on what comprises Soft Computing has not yet been reached. More new sciences are still merging into Soft Computing.

Goals of Soft Computing

Soft Computing is a new multidisciplinary field, to construct new generation of Artificial Intelligence, known as **Computational Intelligence**.

- The main goal of Soft Computing is to develop intelligent machines to provide solutions to real world problems, which are not modeled, or too difficult to model mathematically.

- Its aim is to exploit the tolerance for **Approximation, Uncertainty**, **Imprecision**, and **Partial Truth** in order to achieve close resemblance with human like decision making.

  Approximation : here the model features are similar to the real ones, but not the same.

Uncertainty : here we are not sure that the features of the model are the same as that of the entity (belief).

Imprecision : here the model features (quantities) are not the same as that of the real ones, but close to them.

Importance of Soft Computing

Soft computing differs from hard (conventional) computing. Unlike hard computing, the soft computing is **tolerant of imprecision, uncertainty, partial truth, and approximation**. The guiding principle of soft computing is to exploit these tolerance to achieve tractability, robustness and low solution cost. In effect, the role model for soft computing is the human mind.

The four fields that constitute Soft Computing (SC) are : **Fuzzy Computing** (FC), **Evolutionary Computing** (EC), **Neural computing** (NC), and **Probabilistic Computing** (PC), with the latter subsuming belief networks, chaos theory and parts of learning theory.

Soft computing is not a concoction, mixture, or combination, rather, **Soft computing is a partnership** in which each of the partners contributes a distinct methodology for addressing problems in its domain. In principal the constituent methodologies in Soft computing are complementary rather than competitive.

Soft computing may be viewed as a foundation component for the emerging field of Conceptual Intelligence.

Fuzzy Computing

In the real world there exists much fuzzy knowledge, that is, knowledge which is vague, imprecise, uncertain, ambiguous, inexact, or probabilistic in nature.

Human can use such information because the human thinking and reasoning frequently involve fuzzy information, possibly originating from inherently inexact human concepts and matching of similar rather then identical experiences.

The computing systems, based upon classical set theory and two-valued logic, can not answer to some questions, as human does, because they do not have completely true answers.

We want, the computing systems should not only give human like answers but also describe their reality levels. These levels need to be calculated using imprecision and the uncertainty of facts and rules that were applied.

Fuzzy Sets

Introduced by Lotfi Zadeh in 1965, the fuzzy set theory is an extension of classical set theory where elements have degrees of membership.

- **Classical Set Theory**
  - **Sets** are defined by a simple statement describing whether an element having a certain property belongs to a particular set.
  - When set **A** is contained in an universal space **X**,

then we can state explicitly whether each element **x** of space **X** "is or is not" an element of **A.**

– Set **A** is well described by a function called **characteristic function A**. This function, defined on the universal space **X**, assumes :

value **1** for those elements **x** that belong to set **A**, and

value **0** for those elements **x** that do not belong to set **A.**

The notations used to express these mathematically are

$$\left.\begin{array}{l} A : \mathbf{X} \rightarrow \mathbf{[0, 1]} \\ \mathbf{A(x) \;\; = 1 \;, \; x \;\; is \; a \; member \; of \; A} \\ \mathbf{A(x) \;\; = 0 \;, \; x \;\; is \; not \; a \; member \; of \; A} \end{array}\right\} \quad \text{Eq.(1)}$$

Alternatively, the set **A** can be represented for all elements **x** ∈ **X** by its **characteristic function** $\propto_A$ **(x)** defined as

$$\propto_{\mathbf{A}} \mathbf{(x)} = \begin{cases} \mathbf{1} \quad \mathbf{if} \quad \mathbf{x \in X} \\ \mathbf{0} \quad \mathbf{otherwise} \end{cases} \qquad \text{Eq.(2)}$$

– Thus, in classical set theory $\propto_A$ **(x)** has only the values **0** ('false') and **1** ('true''). Such sets are called **crisp sets.**

- **Crisp and Non-crisp Set**

  – As said before, in classical set theory, the **characteristic function $\propto_A(x)$** of Eq.(2) has only values **0** ('false') and **1** ('true'').

  Such sets are **crisp sets.**

  – For Non-crisp sets the characteristic function $\propto_A(x)$ can be defined.

  - The characteristic function $\propto_A(x)$ of Eq. (2) for the crisp set is generalized for the Non-crisp sets.
  - This generalized characteristic function $\propto_A(x)$ of Eq.(2) is called **membership function**.

  Such Non-crisp sets are called **Fuzzy Sets**.

  – Crisp set theory is not capable of representing descriptions and classifications in many cases; In fact, Crisp set does not provide adequate representation for most cases.

  – The proposition of Fuzzy Sets are motivated by the need to capture and represent real world data with uncertainty due to imprecise measurement.

  – The uncertainties are also caused by vagueness in the language.

- **Example 1 : Heap Paradox**

  This example represents a situation where vagueness and uncertainty are inevitable.

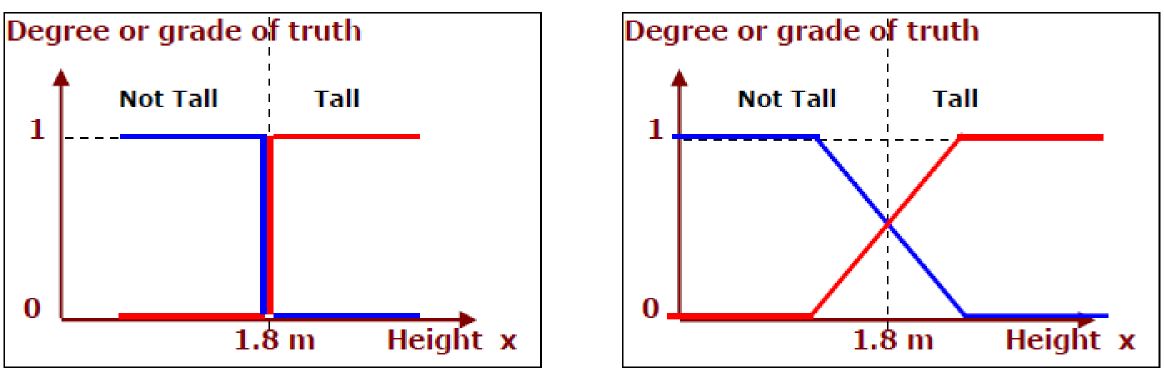
  - If we remove one grain from a heap of grains, we will still have a heap.
  - However, if we keep removing one-by-one grain from a heap of grains, there will be a time when we do not have a heap anymore.
  - The question is, at what time does the heap turn into a countable collection of grains that do not form a heap? There is no one correct answer to this question.

- **Example 2 : Classify Students for a basketball team** This example explains the grade of truth value.

  - **tall students** qualify and **not tall students** do not qualify
  - if students 1.8 m tall are to be qualified, then

    should we exclude a student who is ^{1}/10" less? or should we exclude a student who is 1" shorter?

  - Non-Crisp Representation to represent the notion of a tall person.



Crisp logic Non-crisp logic

Fig. 1 Set Representation – Degree or grade of truth

A student of height 1.79m would belong to both tall and not tall sets with a particular degree of membership.As the height increases the membership grade within the tall set would increase whilst the membership grade within the not-tall set would decrease.

- **Capturing Uncertainty**

  Instead of avoiding or ignoring uncertainty, Lotfi Zadeh introduced Fuzzy Set theory that captures uncertainty.

- A fuzzy set is described by a **membership function** $\mu_A(x)$ of **A**. This membership function associates to each element $x_\sigma \in X$ a number as $\mu_A(x_\sigma)$ in the closed unit interval **[0, 1]**.

  The number $\mu_A(x_\sigma)$ represents the **degree of membership** of x_σ in **A**.
- The notation used for membership function $\mu_A(x)$ of a fuzzy set **A** is

  $A : X \to [0, 1]$

- Each membership function maps elements of a given universal base set **X** , which is itself a crisp set, into real numbers in **[0, 1]** .

- Example

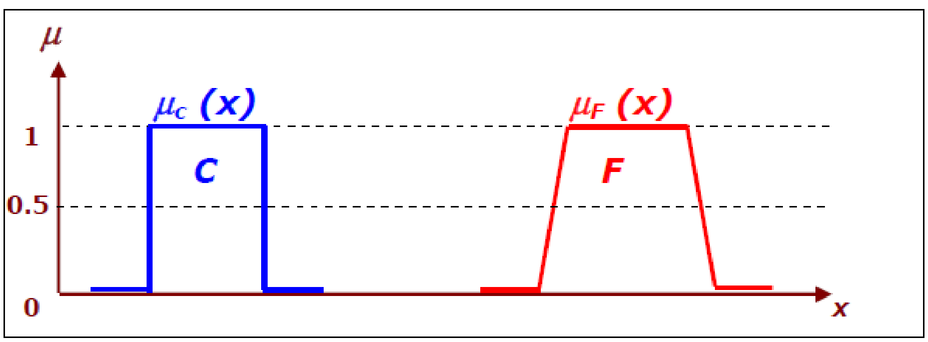


Fig. 2 Membership function of a Crisp set C and Fuzzy set F

- In the case of Crisp Sets the members of a set are : either out of the set, with membership of degree " **0** ", or in the set, with membership of degree " **1** ",

Therefore, **Crisp Sets** $\subseteq$ **Fuzzy Sets** In other words, Crisp Sets are Special cases of Fuzzy Sets.

Example 2: Set of SMALL (as non-crisp set) **Example 1: Set of prime numbers** (a crisp set)

If we consider space **X** consisting of natural numbers ≤ **12**

ie **X = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12}**

Then, the set of prime numbers could be described as follows.

PRIME = {x contained in X | x is a prime number} = {2, 3, 5, 6, 7, 11}

A Set **X** that consists of SMALL cannot be described;

for example **1** is a member of SMALL and **12** is not a member of SMALL.

Set **A**, as SMALL, has un-sharp boundaries, can be characterized by a function that assigns a real number from the closed interval from **0** to **1** to each element **x** in the set **X**.

- **Definition of Fuzzy Set**

A **fuzzy set A** defined in the universal space **X** is a function defined in **X** which assumes values in the range **[0, 1]**.

A fuzzy set **A** is written as a set of pairs **{x, A(x)}** as

A = {{x , A(x)}} , x in the set X

where **x** is an element of the universal space **X**, and
A(x) is the value of the function **A** for this element.

The value **A(x)** is the **membership grade** of the element **x** in a fuzzy set **A**.

Example : Set **SMALL** in set **X** consisting of natural numbers **to 12**.

Assume:
SMALL(1) = 1, SMALL(2) = 1, SMALL(3) = 0.9, SMALL(4) = 0.6,

SMALL(5) = 0.4, SMALL(6) = 0.3, SMALL(7) = 0.2, SMALL(8) = 0.1, SMALL(u) = 0 for u >= 9.

Then, following the notations described in the definition above :

Set SMALL = {{1, 1 }, {2, 1 }, {3, 0.9}, {4, 0.6}, {5, 0.4}, {6, 0.3}, {7, 0.2}, {8, 0.1}, {9, 0 }, {10, 0 }, {11, 0}, {12, 0}}

Note that a fuzzy set can be defined precisely by associating with each **x** , its grade of membership in **SMALL**.

- **Definition of Universal Space**

Originally the universal space for fuzzy sets in fuzzy logic was defined only on the integers. Now, the universal space for fuzzy sets and fuzzy relations is defined with three numbers. The first two numbers specify the start and end of the universal space, and the third argument specifies the increment between elements. This gives the user more flexibility in choosing the universal space.

Example : The fuzzy set of numbers, defined in the universal space

X = { x_i } = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12} is presented as

SetOption [FuzzySet, UniversalSpace → {1, 12, 1}]

- **Graphic Interpretation of Fuzzy Sets SMALL**

The fuzzy set SMALL of small numbers, defined in the universal space

X = { xi } = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12} is presented as

SetOption [FuzzySet, UniversalSpace → {1, 12, 1}]

The Set **SMALL** in set **X** is :

SMALL = FuzzySet {{1, 1 }, {2, 1 }, {3, 0.9}, {4, 0.6}, {5, 0.4}, {6, 0.3},
{7, 0.2}, {8, 0.1}, {9, 0 }, {10, 0 }, {11, 0}, {12, 0}}

Therefore **SetSmall** is represented as

SetSmall = **FuzzySet [{{1,1},{2,1}, {3,0.9}, {4,0.6}, {5,0.4},{6,0.3}, {7,0.2},**
{8, 0.1}, {9, 0}, {10, 0}, {11, 0}, {12, 0}} , **UniversalSpace → {1, 12, 1}]**

FuzzyPlot [SMALL, AxesLable → {"X", "SMALL"}]

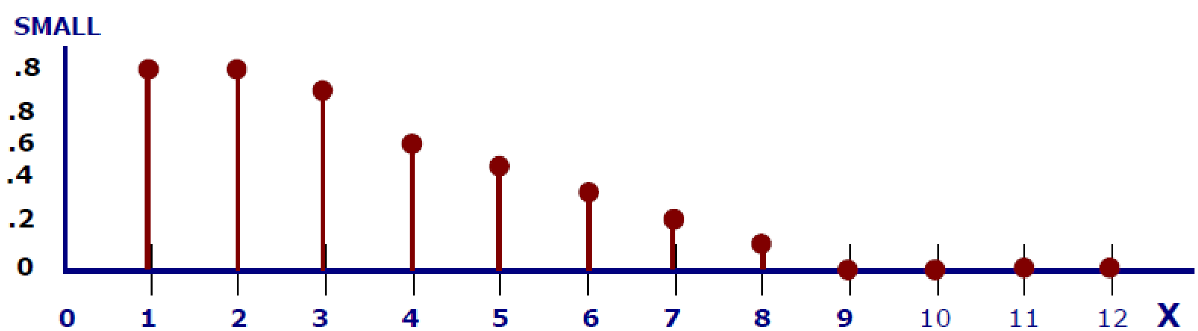


Fig Graphic Interpretation of Fuzzy Sets SMALL

- **Graphic Interpretation of Fuzzy Sets PRIME Numbers**

The fuzzy set PRIME numbers, defined in the universal space

X = { x_i } = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12} is presented as

SetOption [FuzzySet, UniversalSpace → {1, 12, 1}]

The Set **PRIME** in set **X** is :

PRIME = FuzzySet {{1, 0}, {2, 1}, {3, 1}, {4, 0}, {5, 1}, {6, 0}, {7, 1}, {8, 0}, {9, 0}, {10, 0}, {11, 1}, {12, 0}}

Therefore **SetPrime** is represented as

SetPrime = **FuzzySet [{{1,0},{2,1}, {3,1}, {4,0}, {5,1},{6,0}, {7,1},**

{8, 0}, {9, 0}, {10, 0}, {11, 1}, {12, 0}} , UniversalSpace → {1, 12, 1}]

FuzzyPlot [PRIME, AxesLable → {"X", "PRIME"}]

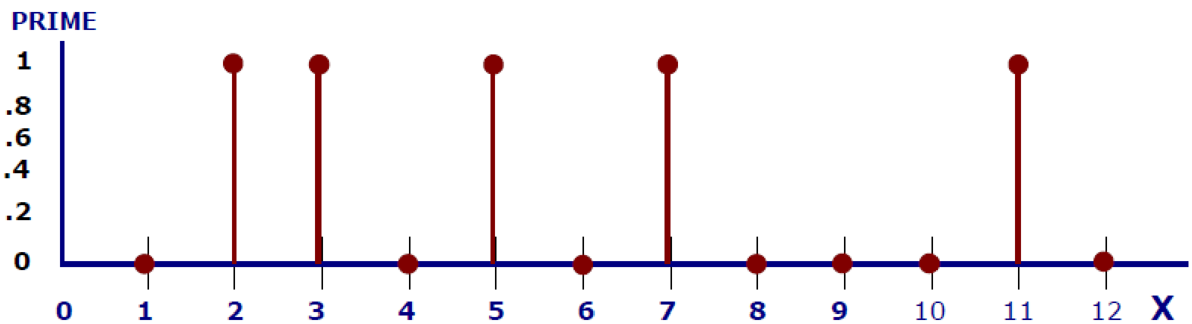


Fig Graphic Interpretation of Fuzzy Sets PRIME

- **Graphic Interpretation of Fuzzy Sets UNIVERSALSPACE**

In any application of sets or fuzzy sets theory, all sets are subsets of

a fixed set called universal space or universe of discourse denoted by **X**. Universal space **X** as a fuzzy set is a function equal to **1** for all elements.

The fuzzy set **UNIVERSALSPACE** numbers, defined in the universal space **X = { xi } = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12}** is presented as

SetOption [FuzzySet, UniversalSpace → {1, 12, 1}]

The Set **UNIVERSALSPACE** in set **X** is :

UNIVERSALSPACE = FuzzySet {{1, 1}, {2, 1}, {3, 1}, {4, 1}, {5, 1}, {6, 1}, {7, 1}, {8, 1}, {9, 1}, {10, 1}, {11, 1}, {12, 1}}

Therefore **SetUniversal** is represented as

SetUniversal = **FuzzySet [{{1,1},{2,1}, {3,1}, {4,1}, {5,1},{6,1}, {7,1}, {8, 1}, {9, 1}, {10, 1}, {11, 1}, {12, 1}} , UniversalSpace → {1, 12, 1}]**

FuzzyPlot [UNIVERSALSPACE, AxesLable → {"X", " UNIVERSAL SPACE "}]

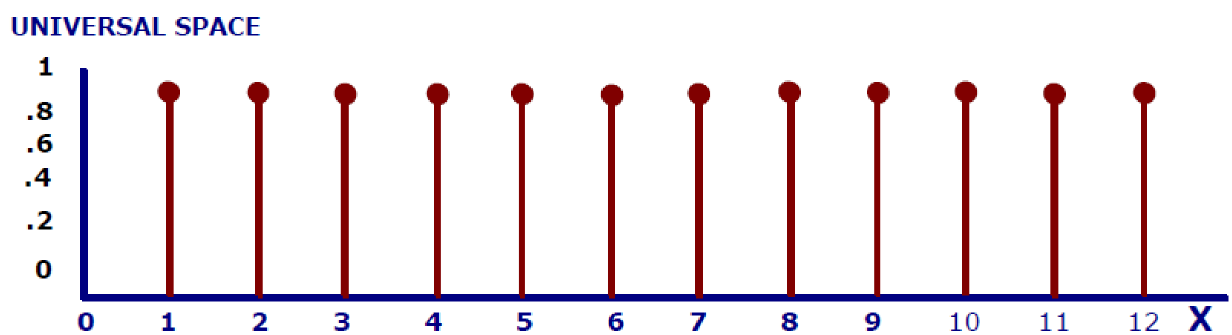


Fig Graphic Interpretation of Fuzzy Set UNIVERSALSPACE

Finite and Infinite Universal Space

Universal sets can be finite or infinite.

Any universal set is finite if it consists of a specific number of different elements, that is, if in counting the different elements of the set, the counting can come to an end, else the set is infinite.

Examples:

1. Let **N** be the universal space of the days of the week.

   N = {Mo, Tu, We, Th, Fr, Sa, Su}. **N** is finite.

2. Let **M = {1, 3, 5, 7, 9, ...}.** **M** is infinite.

3. Let **L = {u | u is a lake in a city }.** **L** is finite.

   (Although it may be difficult to count the number of lakes in a city, but **L** is still a finite universal set.)

- **Graphic Interpretation of Fuzzy Sets EMPTY**

An empty set is a set that contains only elements with a grade of membership equal to **0**.

Example: Let EMPTY be a set of people, in Minnesota, older than 120. The Empty set is also called the Null set.

The fuzzy set **EMPTY** , defined in the universal space

X = { x_i } = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12} is presented as

SetOption [FuzzySet, UniversalSpace →{1, 12, 1}]

The Set **EMPTY** in set **X** is :

EMPTY = FuzzySet {{1, 0}, {2, 0}, {3, 0}, {4, 0}, {5, 0}, {6, 0}, {7, 0}, {8, 0}, {9, 0}, {10, 0}, {11, 0}, {12, 0}}

Therefore **SetEmpty** is represented as

SetEmpty = **FuzzySet [{{1,0},{2,0}, {3,0}, {4,0}, {5,0},{6,0}, {7,0}, {8, 0}, {9, 0}, {10, 0}, {11, 0}, {12, 0}}** , **UniversalSpace → {1, 12, 1}]**

FuzzyPlot [EMPTY, AxesLable → {"X", " UNIVERSAL SPACE "}]

Fuzzy Operations

A fuzzy set operations are the operations on fuzzy sets. The fuzzy set operations are generalization of crisp set operations. Zadeh [1965] formulated the fuzzy set theory in the terms of standard operations: Complement, Union, Intersection, and Difference.

In this section, the graphical interpretation of the following standard fuzzy set terms and the Fuzzy Logic operations are illustrated:

Inclusion : **FuzzyInclude [VERYSMALL, SMALL]**

Equality : **FuzzyEQUALITY [SMALL, STILLSMALL]**

Complement : **FuzzyNOTSMALL = FuzzyCompliment [Small]**

Union : **FuzzyUNION = [SMALL ∪ MEDIUM]**

Intersection : **FUZZYINTERSECTON = [SMALL ∩ MEDIUM]**

- **Inclusion**

Let **A** and **B** be fuzzy sets defined in the same universal space **X.**

The fuzzy set **A** is included in the fuzzy set **B** if and only if for every **x** in the set **X** we have **A(x)** ≤ **B(x)**

Example :

The fuzzy set **UNIVERSALSPACE** numbers, defined in the universal space **X = { x**$_i$ **} = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12}** is presented as

SetOption [FuzzySet, UniversalSpace → {1, 12, 1}]

The fuzzy set B SMALL

The Set **SMALL** in set **X** is :

SMALL = FuzzySet {{1, 1 }, {2, 1 }, {3, 0.9}, {4, 0.6}, {5, 0.4}, {6, 0.3}, {7, 0.2}, {8, 0.1}, {9, 0 }, {10, 0 }, {11, 0}, {12, 0}}

Therefore **SetSmall** is represented as

SetSmall = **FuzzySet [{{1,1},{2,1}, {3,0.9}, {4,0.6}, {5,0.4},{6,0.3}, {7,0.2}, {8, 0.1}, {9, 0}, {10, 0}, {11, 0}, {12, 0}}** , **UniversalSpace → {1, 12, 1}]**

The fuzzy set A VERYSMALL

The Set **VERYSMALL** in set **X** is :

VERYSMALL = FuzzySet {{1, 1 }, {2, 0.8 }, {3, 0.7}, {4, 0.4}, {5, 0.2}, {6, 0.1}, {7, 0 }, {8, 0 }, {9, 0 }, {10, 0 }, {11, 0}, {12, 0}}

Therefore **SetVerySmall** is represented as

SetVerySmall = **FuzzySet [{{1,1},{2,0.8}, {3,0.7}, {4,0.4}, {5,0.2},{6,0.1}, {7,0}, {8, 0}, {9, 0}, {10, 0}, {11, 0}, {12, 0}} , UniversalSpace → {1, 12, 1}]**

The Fuzzy Operation :

nclusion

Include

**[VERYSMALL,
SMALL]**

- **Comparability**

Two fuzzy sets **A** and **B** are comparable

if the condition **A** ⊂ **B or B** ⊂ **A** holds, ie,

if one of the fuzzy sets is a subset of the other set, they are comparable.

Two fuzzy sets **A** and **B** are incomparable

if the condition **A** ⊄ **B or B** ⊄ **A** holds.

Example 1:

Let **A = {{a, 1}, {b, 1}, {c, 0}} and**

B = {{a, 1}, {b, 1}, {c, 1}}.

Then **A** is comparable to **B**, since **A** is a subset of **B**.

Example 2 :

Let **C = {{a, 1}, {b, 1}, {c, 0.5}} and**

D = {{a, 1}, {b, 0.9}, {c, 0.6}}.

Then **C** and **D** are not comparable since

C is not a subset of **D** and

D is not a subset of **C**.

Property Related to Inclusion :

for all **x** in the set **X**, if **A(x)** ⊂ **B(x)** ⊂ **C(x), then accordingly A** ⊂ **C.**

- **Equality**

Let **A** and **B** be fuzzy sets defined in the same space **X**.
Then **A** and **B** if and only if are equal, which is denoted **X = Y**

for all **x** in the set **X**, **A(x) = B(x).**

Example.

The fuzzy set B SMALL

SMALL = FuzzySet {{1, 1 }, {2, 1 }, {3, 0.9}, {4, 0.6}, {5, 0.4}, {6, 0.3}, {7, 0.2}, {8, 0.1}, {9, 0 }, {10, 0 }, {11, 0}, {12, 0}}

The fuzzy set A STILLSMALL

STILLSMALL = FuzzySet {{1, 1 }, {2, 1 }, {3, 0.9}, {4, 0.6}, {5, 0.4}, {6, 0.3}, {7, 0.2}, {8, 0.1}, {9, 0 }, {10, 0 }, {11, 0}, {12, 0}}

The Fuzzy Operation : Equality

Equality [SMALL, STILLSMALL]

the set **X**, then we say that **A** is not equal to **B**.

- **Complement**

Let **A** be a fuzzy set defined in the space **X.**

Then the fuzzy set **B** is a complement of the fuzzy set **A**, if and only if,

for all **x** in the set **X**, **B(x) = 1 - A(x).**

The complement of the fuzzy set **A** is often denoted by **A'** or **Ac** or $\bar{A}$

Fuzzy Complement : **Ac(x) = 1 – A(x)**

Example 1.

The fuzzy set A SMALL

SMALL = FuzzySet {{1, 1 }, {2, 1 }, {3, 0.9}, {4, 0.6}, {5, 0.4}, {6, 0.3}, {7, 0.2}, {8, 0.1}, {9, 0 }, {10, 0 }, {11, 0}, {12, 0}}

The fuzzy set Ac NOTSMALL

NOTSMALL = FuzzySet {{1, 0 }, {2, 0 }, {3, 0.1}, {4, 0.4}, {5, 0.6}, {6, 0.7}, {7, 0.8}, {8, 0.9}, {9, 1 }, {10, 1 }, {11, 1}, {12, 1}}

The Fuzzy Operation : Compliment

NOTSMALL = Compliment [SMALL]

Example 2.

The empty set Φ and the universal set **X**, as fuzzy sets, are complements of one another.

$$\Phi' = X \quad , \quad X' = \Phi$$

The fuzzy set B EMPTY

Empty = FuzzySet {{1, 0 }, {2, 0 }, {3, 0}, {4, 0}, {5, 0}, {6, 0}, {7, 0}, {8, 0}, {9, 0 }, {10, 0 }, {11, 0}, {12, 0}}

The fuzzy set A UNIVERSAL

Universal = FuzzySet {{1, 1 }, {2, 1 }, {3, 1}, {4, 1}, {5, 1}, {6, 1}, {7, 1}, {8, 1}, {9, 1 }, {10, 1 }, {11, 1}, {12, 1}}

The fuzzy operation : Compliment

EMPTY = Compliment [UNIVERSALSPACE]

- **Union**

Let **A** and **B** be fuzzy sets defined in the space **X**.

The union is defined as the smallest fuzzy set that contains both **A** and **B**. The union of **A** and **B** is denoted by **A ∪ B.**

The following relation must be satisfied for the union operation : **for all x in the set X, (A ∪ B)(x) = Max (A(x), B(x)).**

Fuzzy Union : (A ∪ B)(x) = max [A(x), B(x)] for all x ∈ X

Example 1 : Union of Fuzzy **A** and **B**

A(x) = 0.6 and B(x) = 0.4 ∴ (A ∪ B)(x) = max [0.6, 0.4] = 0.6

Example 2 : Union of **SMALL** and **MEDIUM**

The fuzzy set A SMALL

SMALL = FuzzySet {{1, 1 }, {2, 1 }, {3, 0.9}, {4, 0.6}, {5, 0.4}, {6, 0.3}, {7, 0.2}, {8, 0.1}, {9, 0 }, {10, 0 }, {11, 0}, {12, 0}}

The fuzzy set B MEDIUM

MEDIUM = FuzzySet {{1, 0 }, {2, 0 }, {3, 0}, {4, 0.2}, {5, 0.5}, {6, 0.8}, {7, 1}, {8, 1}, {9, 0.7 }, {10, 0.4 }, {11, 0.1}, {12, 0}}

The fuzzy operation : Union

FUZZYUNION = [SMALL ∪ MEDIUM]

SetSmallUNIONMedium = FuzzySet [{{1,1},{2,1}, {3,0.9}, {4,0.6}, {5,0.5}, {6,0.8}, {7,1}, {8, 1}, {9, 0.7}, {10, 0.4}, {11, 0.1}, {12, 0}} , UniversalSpace → {1, 12, 1}]

The notion of the union is closely related to that of the connective "or".

Let **A** is a class of "Young" men, **B** is a class of "Bald" men.

If "David is Young" or "David is Bald," then David is associated with the union of **A** and **B.** Implies David is a member of **A** ∪ **B.**

- **Properties Related to Union**

The properties related to union are :

Identity, Idempotence, Commutativity and Associativity.

- **Identity:**

A ∪ Φ **= A**

input = Equality [SMALL ∪ EMPTY , SMALL]

output = True

A ∪ **X = X**

input = Equality [SMALL ∪ UnivrsalSpace , UnivrsalSpace]

output = True

- **Idempotence :**

A ∪ **A = A**

input = Equality [SMALL ∪ SMALL , SMALL]

output = True

- **Commutativity :**

A ∪ **B = B** ∪ **A**

input = Equality [SMALL ∪ MEDIUM, MEDIUM ∪ SMALL]

output = True

- **Associativity:**

A ∪ (B ∪ C) = (A ∪ B) ∪ C

input = Equality [SMALL ∪ (MEDIUM ∪ BIG) , (SMALL ∪ MEDIUM) ∪ BIG]

output = True

SMALL = FuzzySet {{1, 1 }, {2, 1 }, {3, 0.9}, {4, 0.6}, {5, 0.4}, {6, 0.3}, {7, 0.2}, {8, 0.1}, {9, 0.7 }, {10, 0.4 }, {11, 0}, {12, 0}}

MEDIUM = FuzzySet {{1, 0 }, {2, 0 }, {3, 0}, {4, 0.2}, {5, 0.5}, {6, 0.8}, {7, 1}, {8, 1}, {9, 0 }, {10, 0 }, {11, 0.1}, {12, 0}}

BIG = FuzzySet [{{1,0}, {2,0}, {3,0}, {4,0}, {5,0}, {6,0.1}, {7,0.2}, {8,0.4}, {9,0.6}, {10,0.8}, {11,1}, {12,1}}]

Medium ∪ **BIG** = **FuzzySet [{1,0},{2,0}, {3,0}, {4,0.2}, {5,0.5}, {6,0.8}, {7,1}, {8, 1}, {9, 0.6}, {10, 0.8}, {11, 1}, {12, 1}]**

Small ∪ **Medium** = **FuzzySet [{1,1},{2,1}, {3,0.9}, {4,0.6}, {5,0.5}, {6,0.8}, {7,1}, {8, 1}, {9, 0.7}, {10, 0.4}, {11, 0.1}, {12, 0}]**

27

- **Intersection**

Let **A** and **B** be fuzzy sets defined in the space **X**.

The intersection is defined as the greatest fuzzy set included both **A** and **B**. The intersection of **A** and **B** is denoted by **A ∩ B.**

The following relation must be satisfied for the union operation :

for all x in the set X, (A ∩ B)(x) = Min (A(x), B(x)).

Fuzzy Intersection : (A ∩ B)(x) = min [A(x), B(x)] **for all x ∈ X**

Example 1 : Intersection of Fuzzy **A** and **B**

A(x) = 0.6 and B(x) = 0.4 ∴ (A ∩ B)(x) = min [0.6, 0.4] = 0.4

Example 2 : Union of **SMALL** and **MEDIUM**

The fuzzy set A SMALL

SMALL = FuzzySet {{1, 1 }, {2, 1 }, {3, 0.9}, {4, 0.6}, {5, 0.4}, {6, 0.3}, {7, 0.2}, {8, 0.1}, {9, 0 }, {10, 0 }, {11, 0}, {12, 0}}

The fuzzy set B MEDIUM

MEDIUM = FuzzySet {{1, 0 }, {2, 0 }, {3, 0}, {4, 0.2}, {5, 0.5}, {6, 0.8}, {7, 1}, {8, 1}, {9, 0.7 }, {10, 0.4 }, {11, 0.1}, {12, 0}}

The fuzzy operation : Intersection FUZZYINTERSECTION = min [SMALL ∩ MEDIUM] SetSmallINTERSECTIONMedium = FuzzySet [{{1,0},{2,0}, {3,0}, {4,0.2},

{5,0.4}, {6,0.3}, {7,0.2}, {8, 0.1}, {9, 0},{10, 0}, {11, 0}, {12, 0}} , UniversalSpace → {1, 12, 1}]

Neural Computing

Neural Computers mimic certain processing capabilities of the human brain.

- Neural Computing is an information processing paradigm, inspired by biological system, composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems.

- A neural net is an artificial representation of the human brain that tries to simulate its learning process. The term "artificial" means that neural nets are implemented in computer programs that are able to handle the large number of necessary calculations during the learning process.

- Artificial Neural Networks (ANNs), like people, learn by example.

- An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process.

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

Biological Model:

The human brain consists of a large number (more than a billion) of neural cells that process information. Each cell works like a simple processor. The massive interaction between all cells and their parallel processing, makes the brain's abilities possible. The structure of neuron is shown below.

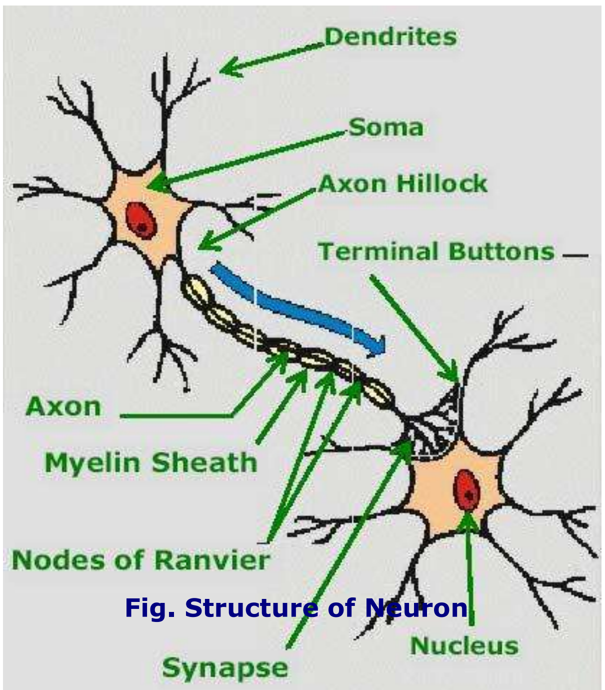


Fig. Structure of Neuron

Dendrites are the branching fibers extending from the cell body or soma. **Soma or cell body** of a neuron contains the nucleus and other structures, support chemical processing and production of neurotransmitters.

Axon is a singular fiber carries information away from the soma to the synaptic sites of other neurons (dendrites and somas), muscles, or glands.

Axon hillock is the site of summation for incoming information. At any moment, the collective influence of all neurons, that conduct as impulses to a given neuron, will determine whether or not an action potential will be initiated at the axon hillock and propagated along the axon.

Myelin Sheath consists of fat-containing cells that insulate the axon from electrical activity. This insulation acts to increase the rate of transmission of signals. A gap exists between each myelin sheath cell along the axon. Since fat inhibits the propagation of electricity, the signals jump from one gap to the next.

Nodes of Ranvier are the gaps (about 1 ∝m) between myelin sheath cells long axons. Since fat serves as a good insulator, the myelin sheaths speed the rate of transmission of an electrical impulse along the axon.

Synapse is the point of connection between two neurons or a neuron and a muscle or a gland. Electrochemical communication between neurons takes place at these junctions.

Terminal Buttons of a neuron are the small knobs at the end of an axon that release chemicals called neurotransmitters.

- **Information flow in a Neural Cell**

The input /output and the propagation of information are shown below.

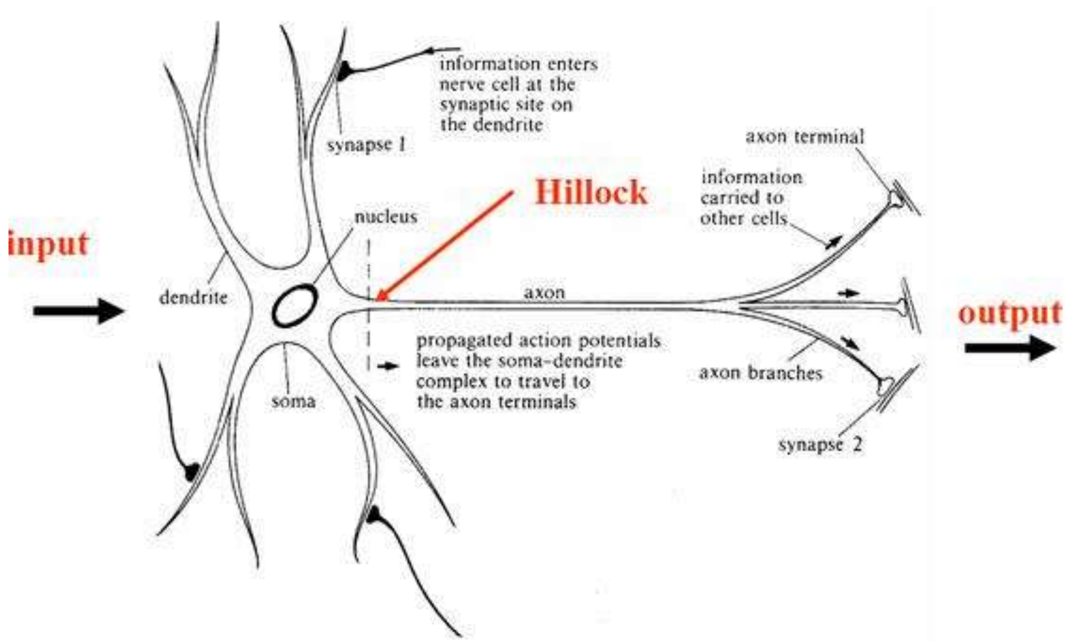


Fig. Structure of a neural cell in the human brain

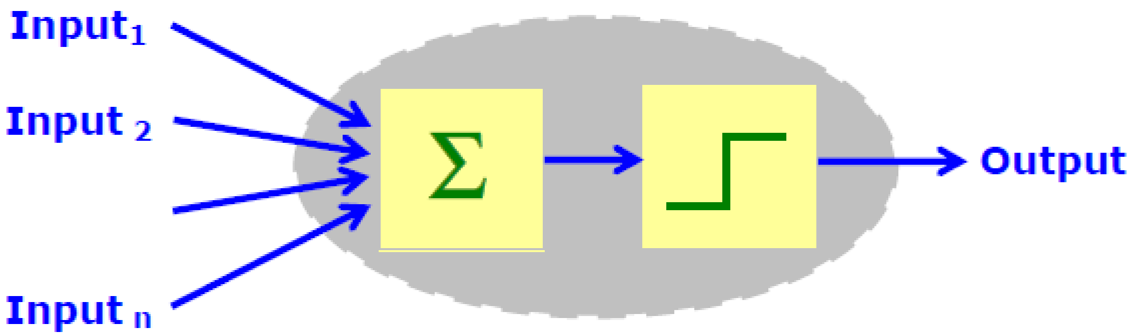
- Dendrites receive activation from other neurons.
- Soma processes the incoming activations and converts them into output activations.
- Axons act as transmission lines to send activation to other neurons.
- Synapses the junctions allow signal transmission between the axons and dendrites.
- The process of transmission is by diffusion of chemicals called neuro-transmitters.

McCulloch-Pitts introduced a simplified model of this real neurons.

Artificial Neuron

- **The McCulloch-Pitts Neuron**

This is a simplified model of real neurons, known as a Threshold Logic Unit.



- A set of synapses (ie connections) brings in activations from other neurons.

- A processing unit sums the inputs, and then applies a non-linear activation function (i.e. transfer / threshold function).

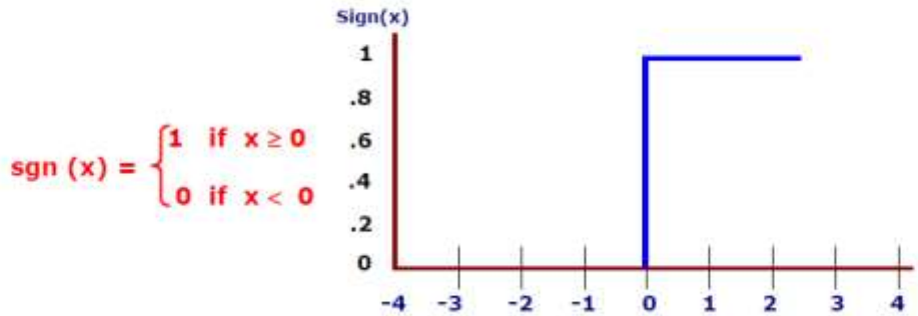
- An output line transmits the result to other neurons.

In other words, the input to a neuron arrives in the form of signals. The signals build up in the cell. Finally the cell fires (discharges) through the output. The cell can start building up signals again.

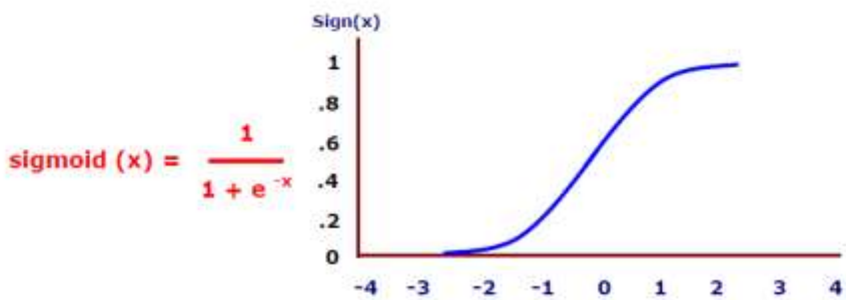
- **Functions :**

The function **y = f(x)** describes a relationship, an input-output mapping, from **x** to **y**.

- **Threshold or Sign function sgn(x)** : defined as

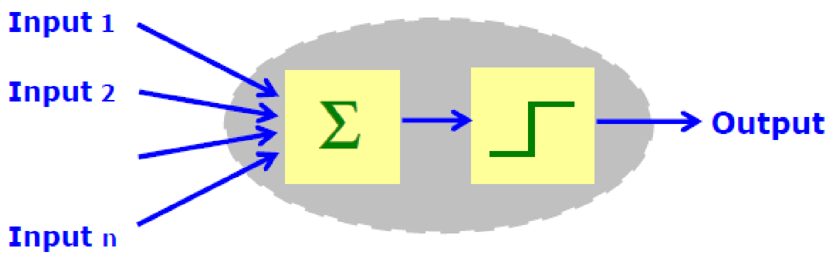


- **Threshold or Sign function sigmoid (x)** : defined as a smoothed (differentiable) form of the threshold function



- **McCulloch-Pitts (M-P) Neuron Equation**

Fig below is the same previously shown simplified model of a real neuron, as a threshold Logic Unit.



The equation for the output of a McCulloch-Pitts neuron as a function of **1** to **n** inputs is written as :

$$\textbf{Output} = \textbf{sgn}\left(\sum_{i=1}^{n} \textbf{Input}_i - \Phi\right)$$

where Φ is the neuron's activation threshold.

$$\textbf{If} \sum_{i=1}^{n} \textbf{Input}_i \geq \Phi \quad \textbf{then Output} = 1$$

$$\textbf{If} \sum_{i=1}^{n} \textbf{Input}_i < \Phi \quad \textbf{then Output} = 0$$

Note : The McCulloch-Pitts neuron is an extremely simplified model of real biological neurons. Some of its missing features include: non-binary input and output, non-linear summation, smooth thresholding, stochastic (non-deterministic), and temporal information processing.

- **Basic Elements of an Artificial Neuron**

It consists of three basic components - weights, thresholds, and a single activation function.

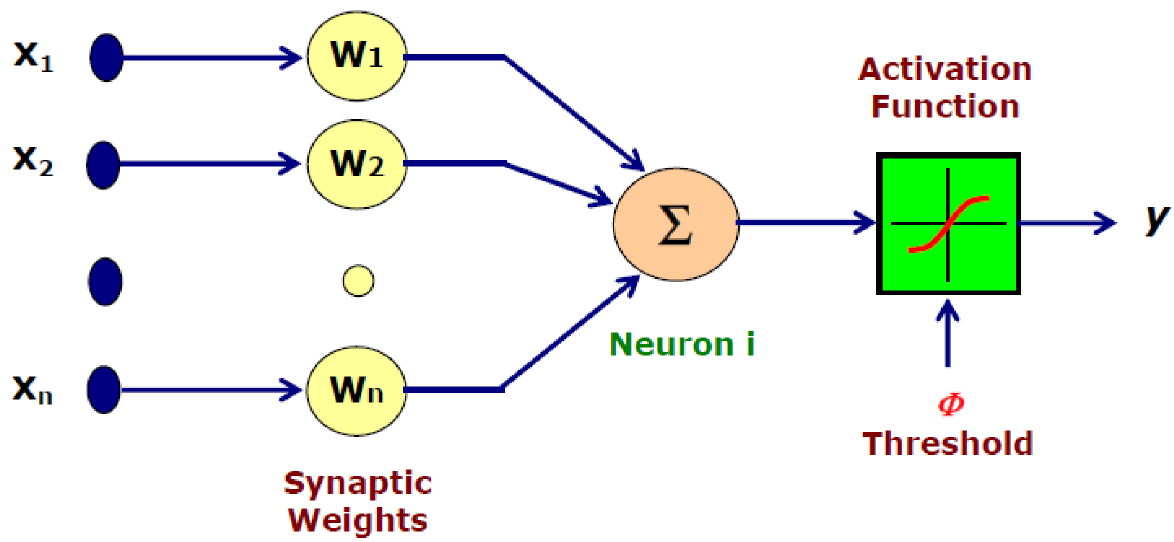


Fig Basic Elements of an Artificial Neuron

Weighting Factors

The values $W_1, W_2, \ldots W_n$ are weights to determine the strength of input row vector $X = [x_1, x_2, \ldots, x_n]^T$. Each input is multiplied by the associated weight of the neuron connection $X^T W$. The **+ve** weight excites and the **-ve** weight inhibits the node output.

Threshold

The node's internal threshold Φ is the magnitude offset. It affects the activation of the node output **y** as:

$$y = \sum_{i=1}^{n} X_i W_i - \Phi_k$$

Activation Function

An activation function performs a mathematical operation on the signal output. The most common activation functions are, Linear

Function, Threshold Function, Piecewise Linear Function, Sigmoidal (S shaped) function, Tangent hyperbolic function and are chose depending upon the type of problem to be solved by the network.

- **Example :**

A neural network consists four inputs with the weights as shown.

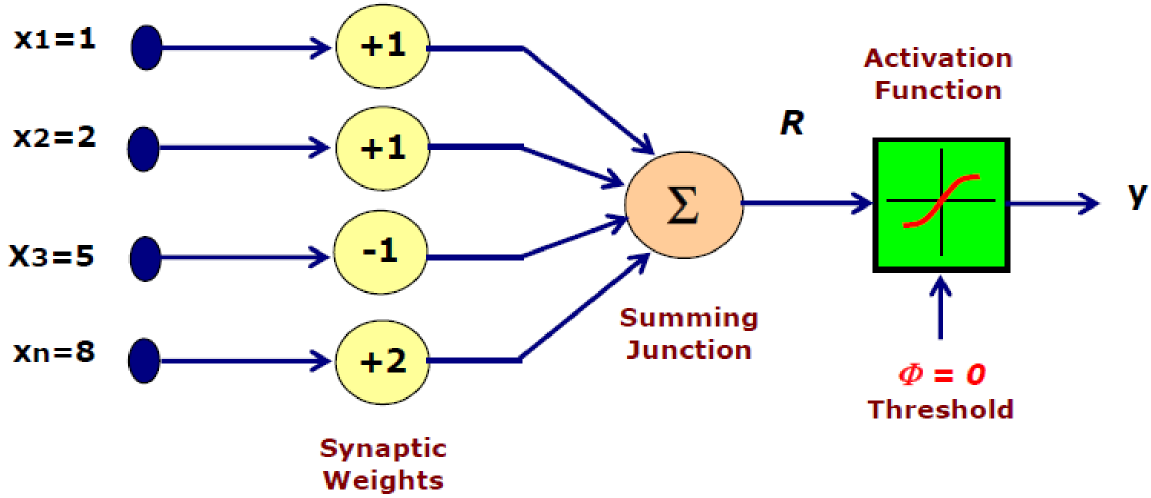


Fig Neuron Structure of Example

The output **R** of the network, prior to the activation function stage, is

$$R = W^T . X = \begin{bmatrix} 1 & 1 & -1 & 2 \end{bmatrix} \bullet \begin{pmatrix} 1 \\ 2 \\ 5 \\ 8 \end{pmatrix} = 14$$

$$= (1 \times 1) + (1 \times 2) + (-1 \times 5) + (2 \times 8) = 14$$

With a binary activation function, the outputs of the neuron is:

y (threshold) = 1

- **Single and Multi - Layer Perceptrons**

A perceptron is a name for simulated neuron in the computer program. The usually way to represent a neuron model is described below.

The neurons are shown as circles in the diagram. It has several inputs and a single output. The neurons have gone under various names.

- Each individual cell is called either a **node** or a **perceptron**.
- A neural network consisting of a layer of nodes or perceptrons between the input and the output is called a **single layer perceptron**.
- A network consisting of several layers of single layer perceptron stacked on top of other, between input and output , is called a **multi-layer perceptron**

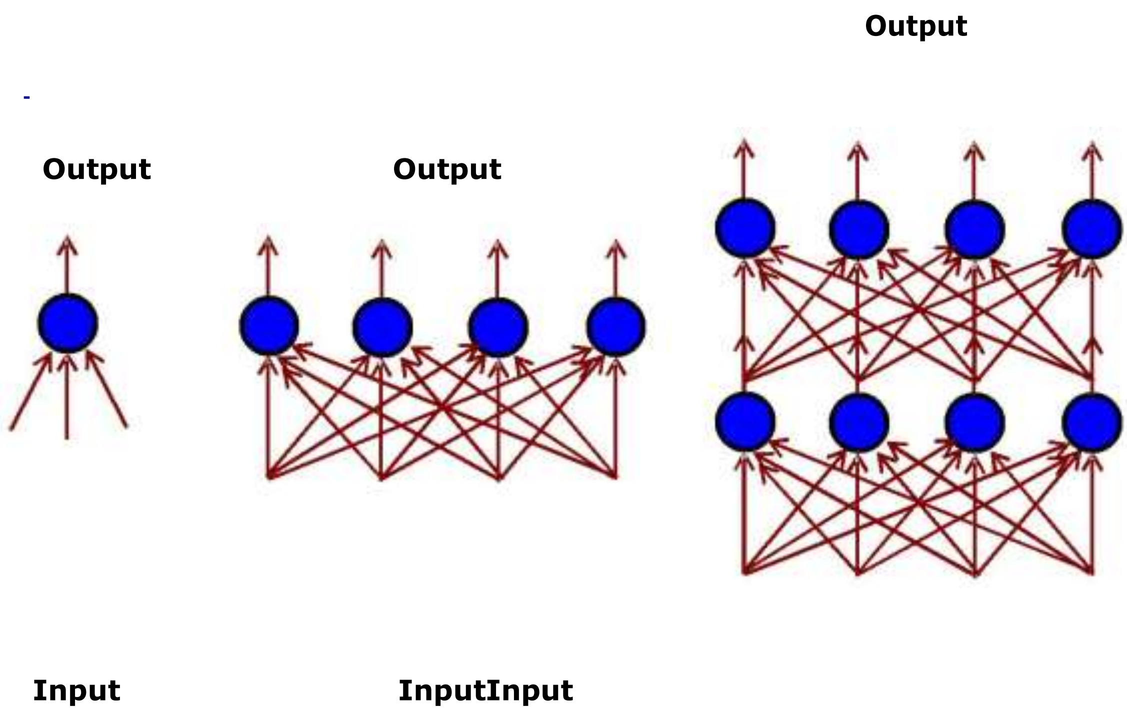


Fig Single and Multi - Layer Perceptrons

Multi-layer perceptrons are more powerful than single-layer perceptrons.

- **Perceptron**

Any number of McCulloch-Pitts neurons can be connected together in any way.

Definition : An arrangement of one input layer of McCulloch-Pitts neurons, that is feeding forward to one output layer of McCulloch-Pitts neurons is known as a Perceptron.

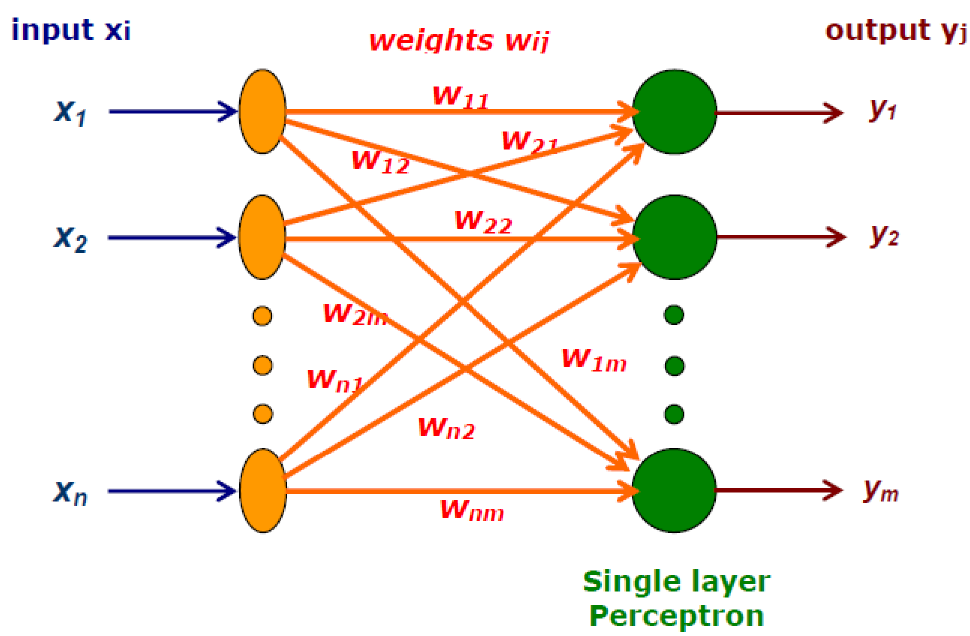


Fig. Simple Perceptron Model

$$y_j = f(net_j) = \begin{cases} 1 & \text{if } net_j \geq 0 \\ 0 & \text{if } net_j < 0 \end{cases} \quad \text{where } net_j = \sum_{i=1}^{n} x_i \, w_{ij}$$

A Perceptron is a powerful computational device.

Neural Network History

- **<u>McCulloch and Pitts (1943)</u>** are generally recognized as the designers of the first neural network.
- They combined many simple processing units together that could lead to an overall increase in computational power.
- They suggested many ideas like : a neuron has a threshold level and once that level is reached the neuron fires.
- It is still the fundamental way in which ANNs operate.
- The McCulloch and Pitts's network had a fixed set of weights.
- Hebb (1949) developed the first learning rule, that is if two neurons are active at the same time then the strength between them should be increased.
- In the 1950 and 60's, many researchers (Block, Minsky, Papert, and Rosenblatt worked on perceptron. The neural network model could be proved to converge to the correct weights, that will solve the problem. The weight adjustment (learning algorithm) used in the perceptron was found more powerful than the learning rules used by Hebb.
- Minsky & Papert (1969) showed that perceptron could not learn those functions which are not linearly separable.
- The neural networks research declined throughout the 1970 and until mid 80's because the perceptron could not learn certain important functions.
- Neural network regained importance in 1985-86.
- The researchers, Parker and LeCun discovered a learning algorithm for multi-layer networks called back propagation that could solve problems that were not linearly separable.

Activation Function

- Activation function is nothing but a mathematical function that takes an input and produces an output.
- The input in this instance is the weighted sum plus bias:

$$Output = activation\ function(x_1w_1 + x_2w_2 + \cdots + x_nw_n + bias)$$

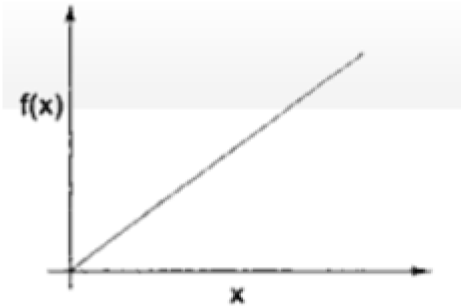
Think of the activation function as a mathematical operation that normalizes the input and produces an output. The output is then passed forward onto the neurons on the subsequent layer.

- The thresholds are pre-defined numerical values in the function.
- **Activation functions** are important for a Artificial **Neural Network** to learn and understand the complex patterns.
- The main **function** of it is to introduce non-linear properties into the **network**.
- The non linear **activation function** will help the model to understand the complexity and give accurate results.

- Some activation functions are:
  - Identity function
  - Binary step function
  - Sigmoidal Functions
    - Binary and
    - Bipolar

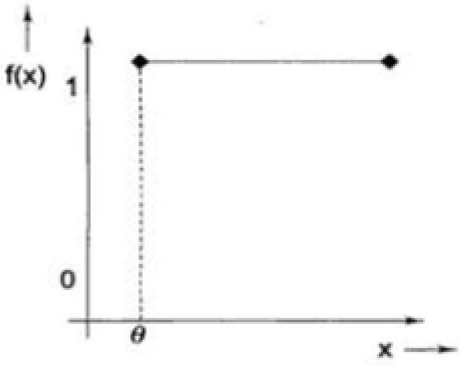
- **<u>Identity Function</u>: Linear function and output is same as input.**
  - The function is given by,

    $F(x) = x$; for all x



- **<u>Binary Step Function</u>: used in single layer to convert network input to output.**
  - The function is given by

$$f(x) = \begin{cases} 1; & \text{if} \quad x \geq \theta \\ 0; & \text{if} \quad x < \theta \end{cases}$$

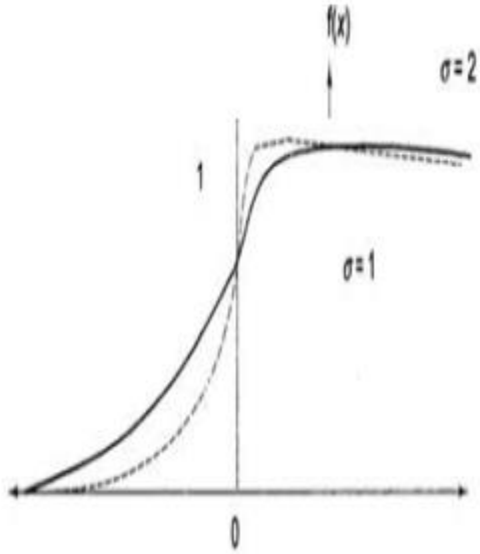


- **Sigmoidal Functions**
  - These functions are usually S-shaped curve.
  - These are used in multilayer nets like back propagation network, radial basis function network etc.
  - There are two main types of Sigmoidal fuctions:
    - Binary Sigmoidal Function
    - Bipolar Sigmoidal Function

Binary Sigmoidal Function

- This is called logistic function
- It ranges between 0 to 1.
- This function can be represented as

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



- Where □ is called the steepness parameters. If f(x) is differentiated we g

- $$f'(x) = \sigma f(x) [1 - f(x)].$$

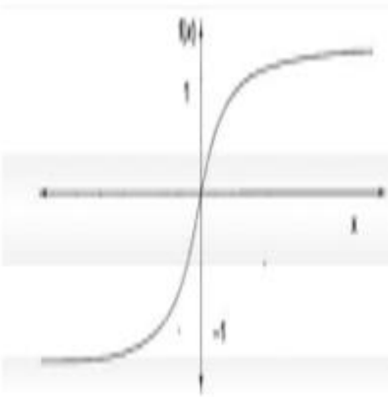
Bipolar Sigmoidal Function

- The desired range here is between +1 and -1 .
- This function is related to the hyperbolic tangent function.
- The bipolar sigmoidal function is $b(x) = 2f(x) - 1$

$$b(x) = 2 \times \frac{1}{1 + \exp(-\sigma x)} - 1$$

$$= \frac{2 - 1 - \exp(-\sigma x)}{1 + \exp(-\sigma x)}$$

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



On differentiating the function

$$b'(x) = \frac{\sigma}{2} [(1 + b(x)) (1 - b(x))]$$

Artificial neural network Vs. Biological neural network

Characteristics	**Artificial Neural Network**	**Biological Neural Network**
Speed	Faster in processing information. Response time is in nanoseconds.	Slower in processing information. The response time is in milliseconds.

Processing	Serial processing	Massively parallel processing
Size and complexity	Small size usually 10^2 to 10^4 nodes, mainly depends on the type of application and network designer and less complex	Highly complex and dense network of interconnected neurons containing neurons of the order of 10^{11} neurons and 10^{15} interconnections.
Storage	Information storage is replaceable means new data can be added by deleting an old one.	Information storage is adaptable means new information is added by adjusting the interconnection strengths without destroying old information.
Fault Tolerance	It is capable of robust performance, hence has the potential to be fault tolerant	Performance degrades with even partial damage.
Control Mechanism	There is a control unit for controlling computing activities.	No specific control mechanism external to the computing task.

Advantages of ANN

I. Neural Networks have the ability to learn by themselves and produce the output that is not limited to the input provided to them.

II. The input is stored in its own networks instead of a database, hence the loss of data does not affect its working.

III. These networks can learn from examples and apply them when a similar event arises, making them able to work through real-time events.

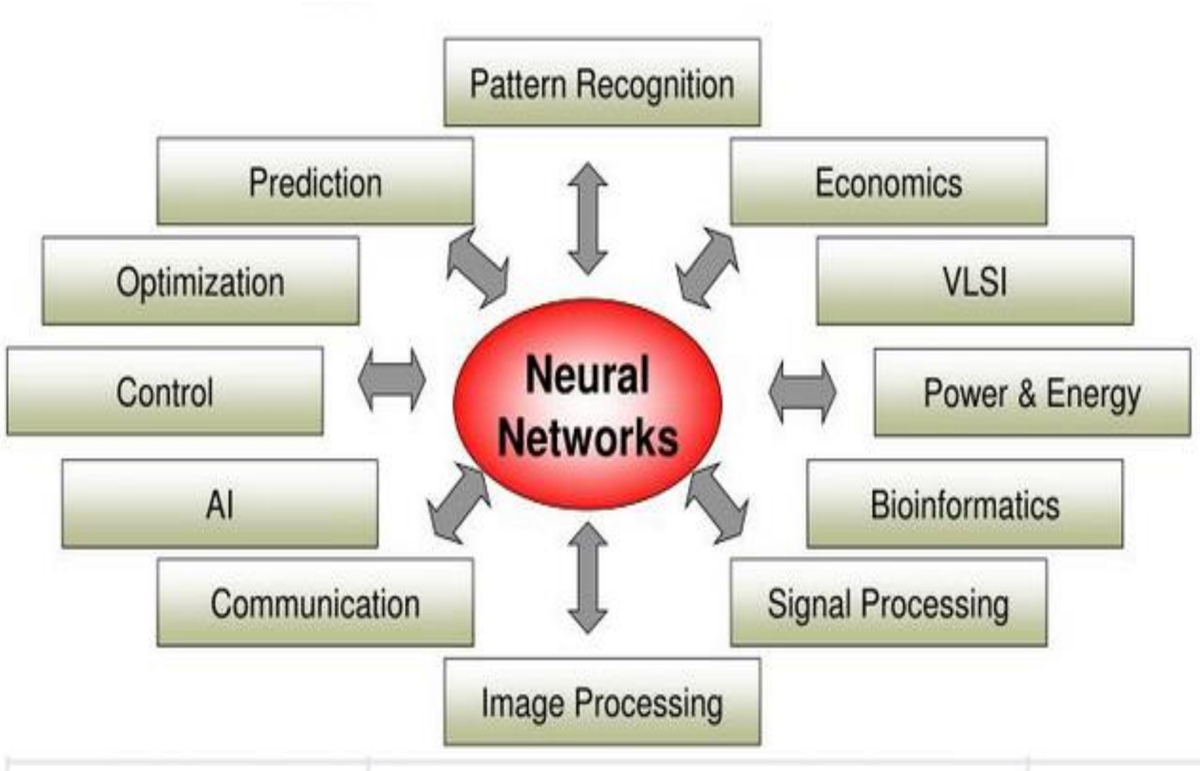
IV. Even if a neuron is not responding or a piece of information is missing, the network can detect the fault and still produce the output.

Disadvantages of ANN

1. Hardware Dependence:
    1. Artificial Neural Networks require processors with parallel processing power, by their structure.
    2. For this reason, the realization of the equipment is dependent.
2. Unexplained functioning of the network:
    1. This the most important problem of ANN.

    2. When ANN gives a probing solution, it does not give a clue as to why and how.
    3. This reduces trust in the network.
3. Assurance of proper network structure:
    1. There is no specific rule for determining the structure of artificial neural networks.
    2. The appropriate network structure is achieved through experience and trial and error.
4. The difficulty of showing the problem to the network:
    1. ANNs can work with numerical information.
    2. Problems have to be translated into numerical values before being introduced to ANN.
    3. The display mechanism to be determined will directly influence the performance of the network.
    4. This is dependent on the user's ability.
5. The duration of the network is unknown:
    1. The network is reduced to a certain value of the error on the sample means that the training has been completed.
    2. The value does not give us optimum results.

Applications of ANN



Type of Neural Network Architecture

1. Feed Forward Network

- Single Layer Feed forward
- Multi Layer Feed forward

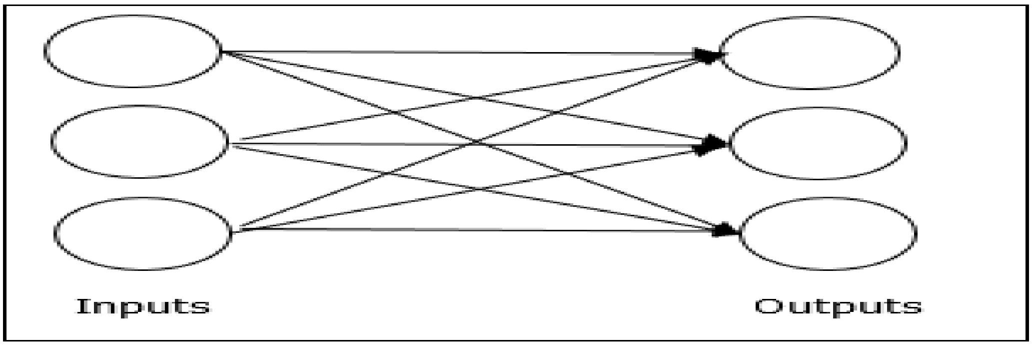
2. Feedback Network
    - Fully recurrent network
    - Jordan network

Feed-forward Network

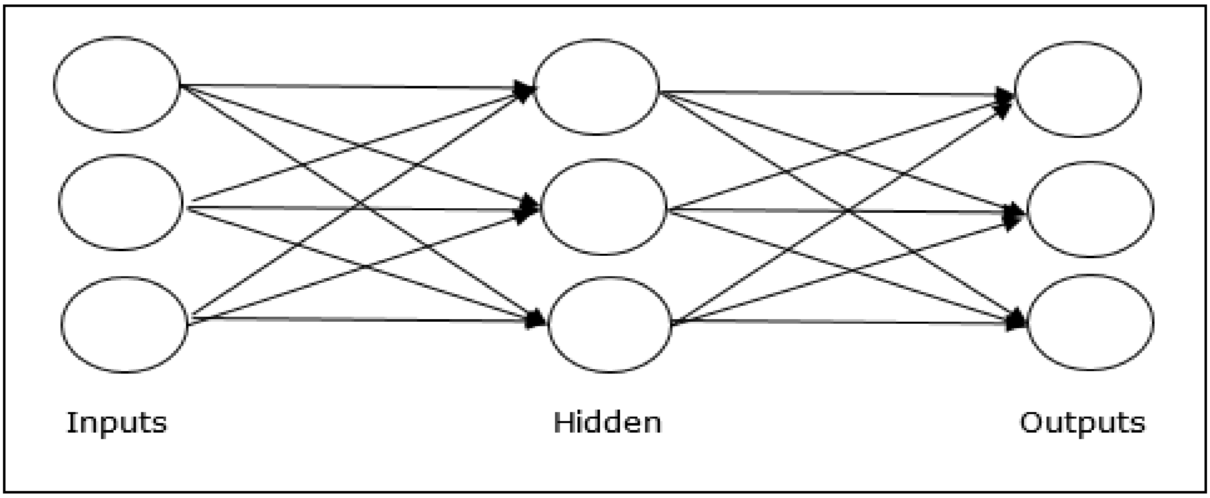
- Feed-forward neural network are the basic type of neural networks.
- The information in this network travels in a unidirectional manner, that is, only from input to processing node to output.
- The hidden layers may or may not be present in this type, making it more explicable.

Single Layer Feed forward Networks

I. In a layered neural network the neurons are organized in the form of layers.
II. In the simplest form of a layered network, we have an input layer of source nodes that projects on to an output layer of neurons, but not vice versa.
III. In other words, this network is strictly a feed forward or acyclic type.
IV. In the below figure, the case of four nodes in both the input and output layers. Such a network is called a single layer network, with the designation "single layer" referring to the output layer of computation nodes (neurons).



Multi Layer Feed Forward



Multi Layer Feed Forward

I. The net where the signal flow from the input units to the output units in a forward direction.
II. The multi-layer net pose one or more layers of nodes between the input and output units.

III. This is advantageous over single layer net in the sense that, it can be used to solve more complicated problems.

Feedback Network

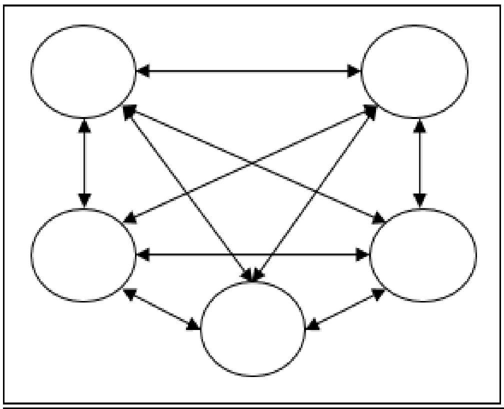
1. As the name suggests, a feedback network has feedback paths, which means the signal can flow in both directions using loops.
2. This makes it a non-linear dynamic system, which changes continuously until it reaches a state of equilibrium.

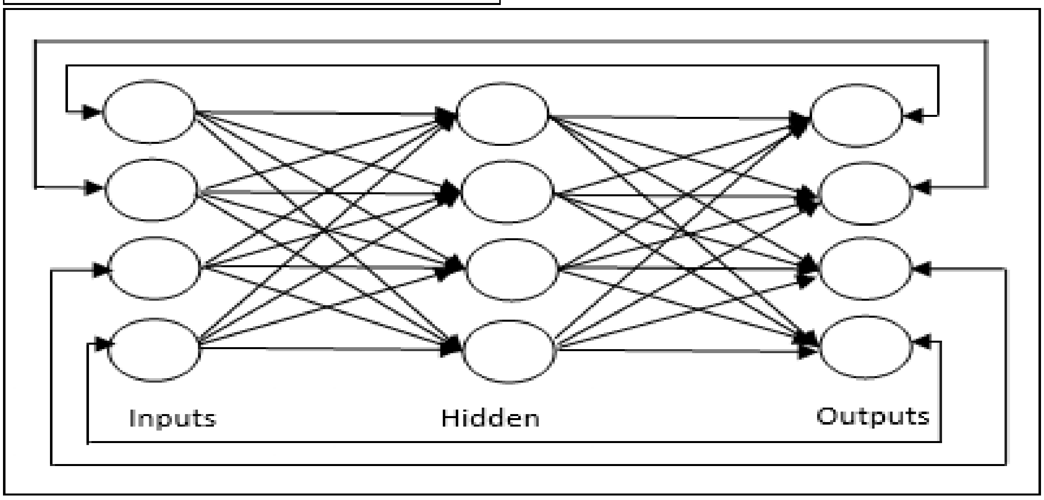
Types of feedback networks

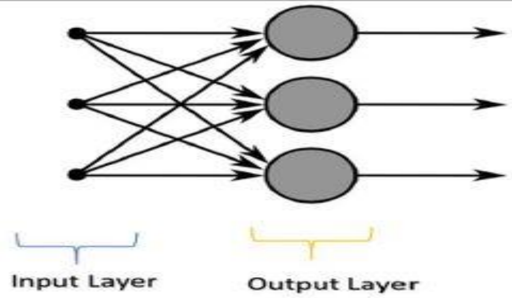
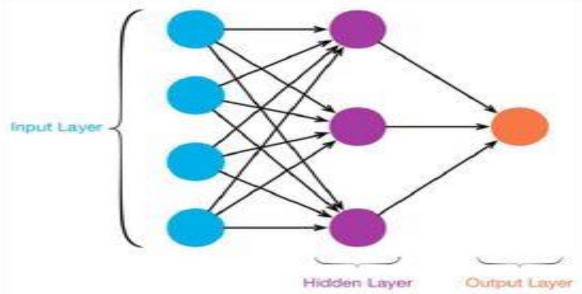
i. **Recurrent networks** − Recurrent neural networks are much more complex and most widely used networks. The data flows in multiple directions in this network. They store the output data of the processing nodes and learn to improve their functioning. Following are the two types of recurrent networks.

ii. **Fully recurrent network** − It is the simplest neural network architecture because all nodes are connected to all other nodes and each node works as both input and output.

iii. **Jordan network** − It is a closed loop network in which the output will go to the input again as feedback as shown in the following diagram.

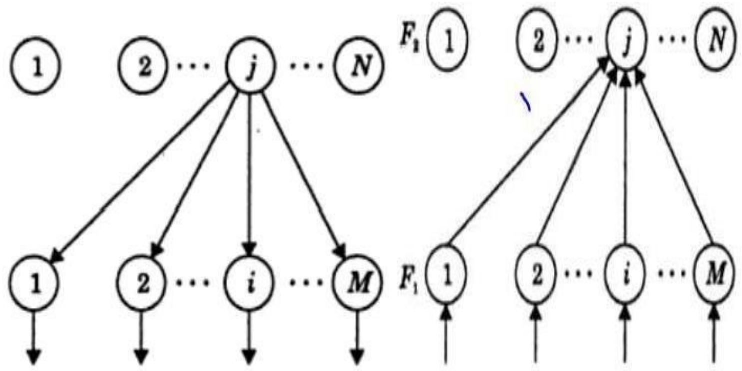




Single Layer Feed-Forward Neural Network	Multi Layer Feed-Forward Neural Network
Layer is formed by taking processing element & combining it with other processing element.	It is formed by interconnection of several layers.
Input & output are linked with each other.	There are multiple layers between input & output layers which are known as hidden layers.
Inputs are connected to the processing nodes with various weights resulting series of output one per node.	Input layers receives input & buffers input signal, output layer generates output.
Zero hidden layers are present.	Zero to several hidden layers are in a network.
Not efficient in certain areas.	More the hidden layers, more the complexity of networks, but efficient output is produced.
	

Topology

- Artificial neural network are organized into layers of processing units.
- The units of layer are similar in the sense that they all have the same activation dynamics and output function.
- The arrangement of the processing units, connections, and pattern input/output is referred to as **<u>topology.</u>**
- The connections can be made either from the units of one layer to the units of another layer (interlayer connections) or among the units within layer (intralayer connection) or both interlayer and lntralayer connection.
- Some basic structure of Artificial neural network are:
    - Instar
    - Outstar
    - Group of instars
    - Group of outstars
    - Bidirectional associative memory
    - Auto associative memory

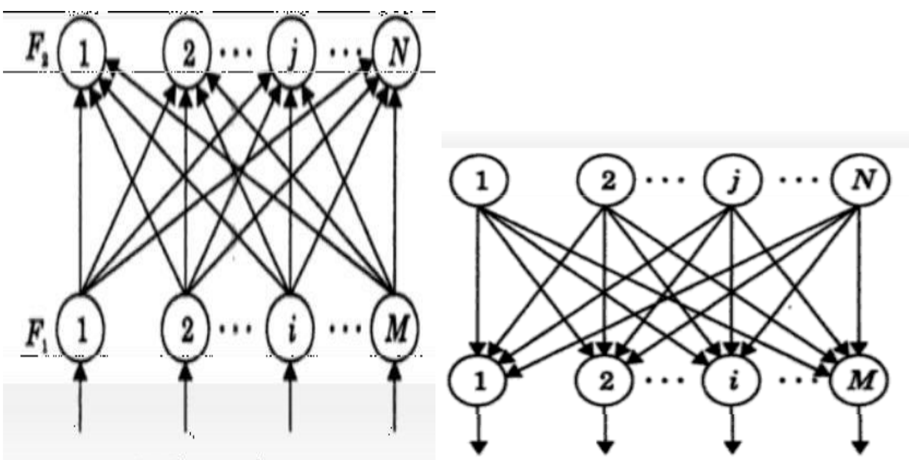


(b) Outstar (a) Instar

Instar and Outstar

1. Let us consider two layers F1 and F2 with M and N processing units respectively.
2. By providing connections to the jth unit in the F2 layer from all the units in the F1 layer, we get two network structures instar and outstar, which have fan-in and fan-out geometries respectively.
3. Whenever the input is given to F1 then the jth unit of F2 will be activated to the maximum extent thus the operation of an instar can be viewed as content addressing the memory.
4. In the case of an outstar, during learning, the weight vector for the connections from the jth unit in F2 approaches the activity pattern in F1, when j is activated, the signal pattern will be transmitted to F1, S_j is the output of the jth unit.

Group of Instars and Group of Outstars

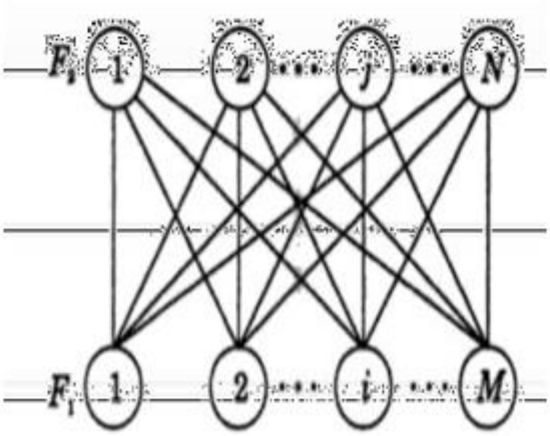


(c) Group of instars (d) Group of outstars

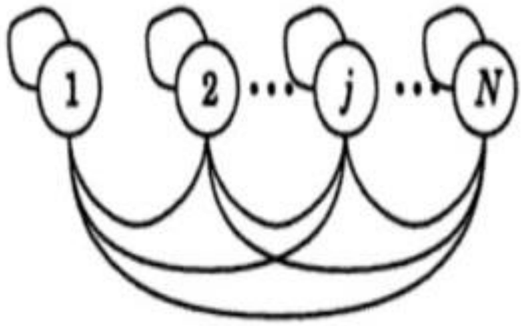
Group of Instars and Group of Outstars

1. In the group of instars, the flow is from F1 to F2. It is also called as hetro-association network.
2. In the group of outstars the flow is from F2 to F1.

Associative and Bidirectional associative memory



(e) Bidirectional associative memory



(f) Autoassociative memory

1. When the flow is bidirectional, we get a bidirectional associative memory, where either of the layers can be used as input/output.
2. If the two layers F1 and F2 coincide and the weights are symmetric, then we obtain and auto associative memory in which each unit is connected to every other unit and to itself