MODULE 1

CHAPTER 1

Uncertainty

University Prescribed Syllabus w.e.f Academic Year 2022-2023

Uncertainty in AI, Inference using full joint distributions, Bayes Theorem, the semantics of Bayesian Networks, Inference in Bayesian networks, Decision Theory, Markov Decision Processes.

Self-learning Topics : Hidden Markov Model (HMM), Gaussian Mixture Model (GMM).

1.1 DEFINITION OF UNCERTAINTY

- **Uncertainty** is a state of doubt about the future or about what is the right thing to do.
- It a situation in which something is not known, or something that is not known or certain.
- Uncertainty is defined as doubt.
- When you feel as if you are not sure if you want to take a new job or not, this is an example of uncertainty.
- When the economy is going bad and causing everyone to worry about what will happen next, this is an example of an uncertainty.

1.2 UNCERTAINTY IN AI

- In AI, we have learned knowledge representation using first-order logic and propositional logic with certainty, which means we were sure about the predicates.
- With this knowledge representation, we might write A→B, which means if A is true then B is true, but consider a situation where we are not sure about whether A is true or not then we cannot express this statement, this situation is called **uncertainty**.
- So, to represent uncertain knowledge, where we are not sure about the predicates, we need uncertain reasoning or probabilistic reasoning.
- AI agents may be required to deal with uncertainty, whether due to partial observability, lack of determinism, or a combination of the two. An agent may never be certain of its current state or where it will end up after a series of actions.

1. **Uncertain data :** When data is missing, unreliable, ambiguous, inconsistent, subjective, noisy, or derived from defaults, it represents an expert's guess; it is referred to as uncertain data.
2. **Uncertain knowledge :** Uncertain knowledge exists when available knowledge has multiple causes leading to multiple effects or when knowledge of causality in the domain is incomplete and cannot be defined in advance.
3. **Uncertain knowledge representation :** Uncertain knowledge representation refers to representations that provide a restricted model of the real system, limited expressiveness of the representation mechanism, and data with imprecise representation.

1.2.1 Causes of Uncertainty

Following are some leading causes of uncertainty to occur in the real world.

- Information occurred from unreliable sources
- Experimental Errors
- Equipment fault
- Temperature variation
- Climate change

1.3 PROBABILISTIC REASONING

- Probabilistic reasoning is a way of knowledge representation where we apply the concept of probability to indicate the uncertainty in knowledge.
- In probabilistic reasoning, we combine probability theory with logic to handle the uncertainty.
- We use probability in probabilistic reasoning because it provides a way to handle the uncertainty that is the result of someone's laziness and ignorance.
- In the real world, there are lots of scenarios, where the certainty of something is not confirmed, such as "It will rain today", "behavior of someone for some situations", "A match between two teams or two players".
- These are probable sentences for which we can assume that it will happen but not sure about it, so here we use probabilistic reasoning.
- **Need of probabilistic reasoning in AI**
  1. When there are unpredictable outcomes.
  2. When specifications or possibilities of predicates becomes too large to handle.
  3. When an unknown error occurs during an experiment.
- In probabilistic reasoning, there are two ways to solve problems with uncertain knowledge:
  1. Baye's rule
  2. Bayesian Statistics

- As probabilistic reasoning uses probability and related terms, so before understanding probabilistic reasoning, let's understand some common terms :

1. **Probability :** Probability can be defined as a chance that an uncertain event will occur. It is the numerical measure of the likelihood that an event will occur. The value of probability always remains between 0 and 1 that represent ideal uncertainties.
   - $0 \leq P(A) \leq 1$, where P (A) is the probability of an event A.
   - P (A) = 0, indicates total uncertainty in an event A.
   - P (A) = 1, indicates total certainty in an event A.
   - We can find the probability of an uncertain event by using the below formula.

$$\text{Probability of Occurrence} = \frac{\text{Number of desired outcomes}}{\text{Total number of outcomes}}$$

   - $P(\bar{A})$ = probability of event A not happening.
   - $P(\bar{A}) + P(A) = 1$.
2. **Event :** Each possible outcome of a variable is called an event.
3. **Sample Space :** The collection of all possible events is called sample space.
4. **Random Variables :** Random variables are used to represent the events and objects in the real world.
5. **Prior Probability :** The prior probability of an event is probability computed before observing new information. For example, if the prior probability that I have cavity is 0.2, then we would write

   P (cavity = true) = 0.2 or P (cavity) = 0.2
6. **Posterior Probability :** The probability that is calculated after all evidence or information has taken into account is called the posterior probability. It is a combination of prior probability and new information.
7. **Conditional Probability :** Conditional probability is a probability of occurring an event when another event has already happened.

   Let's suppose, we want to calculate the event A when event B has already occurred, "the probability of A under the conditions of B", it can be written as:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

where, $P(A \cap B)$ = Joint Probability of A and B.

P (B) = Marginal Probability of B and P (B) > 0.

We can write, $P(A \cap B) = P(A \mid B)\, P(B)$

1.4 INFERENCE USING FULL JOINT DISTRIBUTIONS

- Probability inference means computation from observed evidence of posterior probabilities.
- The knowledge based answering the query is represented as full joint distribution.
- The probability distribution on a single variable must sum to 1.
- It is also true that any joint probability distribution on any set of variables must sum to 1.
- Any proposition 'a' is equivalent to the disjunction of all the atomic events in which 'a' holds.
- Call this set of events e (a).
- Atomic events are mutually exclusive, so the probability of any conjunction of atomic events is zero.
- We have,

$$P(a) = \sum_{e_i \in e(a)} P(e_i)$$

- Given a full joint distribution that specifies the probabilities of all atomic events, this equation provides a simple method for computing the probability of any proposition.

	toothache		¬toothache	
	catch	¬catch	catch	¬catch
cavity	0.108	0.012	0.072	0.008
¬cavity	0.016	0.064	0.144	0.576

- For example, there are six atomic events for **(cavity V toothache) :**

  0.108 + 0.012 + 0.072 + 0.008 + 0.016 + 0.064 = 0.28
- Extracting the distribution over a variable (or some subset of variables), marginal probability, is attained by adding the entries in the corresponding rows or columns.

- For example,

  $P(\text{cavity}) = 0.108 + 0.012 + 0.072 + 0.008 = 0.2$

- We can write the following general marginalization (summing out) rule for any sets of variables Y and Z :

$$P(Y) = \sum_{z \in Z} P(Y, z)$$

For example,

$$P(\text{cavity}) = \sum_{z \in \{\text{cavity, to othache}\}} P(\text{cavity}, z)$$

- A variant of this rule involves conditional probabilities instead of joint probabilities, using the product rule:

$$P(Y) = \sum_{z \in Z} P(Y \mid z)\, P(z)$$

- This rule is called **conditioning.**
- Marginalization and conditioning turn out to be useful rules for all kinds of derivations involving probability expressions.
- Computing a conditional probability

$$P(\text{cavity} \mid \text{toothache}) = \frac{P(\text{cavity} \wedge \text{toothache})}{P(\text{toothache})}$$

$$= \frac{0.108 + 0.012}{0.108 + 0.012 + 0.016 + 0.064} = \frac{0.12}{0.2} = 0.6$$

Similarly,

$$P(\neg \text{cavity} \mid \text{toothache}) = \frac{0.016 + 0.064}{0.2} = 0.4$$

- The two probabilities sum up to one, as they should.
- In both the cases, $\frac{1}{P(\text{toothache})} = \frac{1}{0.2} = 5$ remains constant, no matter which value of cavity we calculate. It is a normalization constant ensuring that the distribution P (cavity | toothache) adds up to 1.
- Let α denote the normalization constant.

$$P(\text{cavity} \mid \text{toothache}) = \alpha\, P(\text{cavity, toothache})$$

$$= \alpha\, [P(\text{cavity, toothache, catch}) + P(\text{cavity, toothache, } \neg \text{ catch})]$$

$$= \alpha\, [(0.108, 0.016) + (0.012, 0.064)]$$

$$= \alpha\, [0.12, 0.08] = [0.6, 0.04]$$

- In other words, we can calculate the conditional probability distribution without knowing P (toothache) using normalization.

Ex. 1.4.1 : In a class, there are 80% of the students who like English and 30% of the students who likes English and Mathematics, and then what is the percentage of students those who like English, also like mathematics ?

☑ **Soln. :**

Let, A is an event that a student likes Mathematics

B is an event that a student likes English

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)} = \frac{0.3}{0.8} = 0.375$$

Hence, 37.5% are the students who like English also like Mathematics.

Ex. 1.4.2 : The probability that it will be sunny on Friday is 4/5. The probability that an ice cream shop will sell ice creams on a sunny Friday is 2/3 and the probability that the ice cream shop sells ice creams on a non-sunny Friday is 1/3. Then find the probability that it will be sunny and the ice cream shop sells the ice creams on Friday.

☑ **Soln. :**

Let us assume that the probabilities for a Friday to be sunny and for the ice cream shop to sell ice creams be S and I respectively. Then,

$$P(S) = 4/5$$

$$P(I \mid S) = 2/3$$

$$P(I \mid \bar{S}) = 1/3.$$

We have to find $P(S \cap I)$.

We can see that S and I are dependent events. By using the dependent events' formula of conditional probability.

$$P(S \cap I) = P(I \mid S) \cdot P(S) = (2/3) \cdot (4/5) = 8/15.$$

Answer : The required probability = 8/15.

Ex. 1.4.3 : The table below shows the occurrence of diabetes in 100 people. Let D and N be the events where a randomly selected person "has diabetes" and "not overweight". Then find P (D | N).

	Diabetes (D)	No Diabetes ($\bar{D}$)
Not overweight (N)	5	45
Overweight ($\bar{N}$)	17	33

☑ **Soln. :**

From the given table,

$$P(N) = \frac{5+45}{100} = \frac{50}{100} = 0.5$$

$$P(D \cap N) = \frac{5}{100} = 0.05$$

By the conditional probability formula,

$$P(D \mid N) = \frac{P(D \cap N)}{P(N)} = \frac{0.05}{0.5} = 0.1$$

Answer : P (D | N) = 0.1

1.5 BAYES THEOREM

- Reverend Thomas Bayes, that helps in determining the probability of an event that is based on some event that has already occurred.
- Bayes theorem has many applications such as Bayesian interference, in the healthcare sector, to determine the chances of developing health problems with an increase in age and many others. In finance, Bayes' Theorem can be used to rate the risk of lending money to potential borrowers.
- Bayes theorem, in simple words, determines the conditional probability of an event A given that event B has already occurred.
- Bayes theorem is also known as the Bayes Rule or Bayes Law.
- It is a method to determine the probability of an event based on the occurrences of prior events. It is used to calculate conditional probability.
- Bayes theorem calculates the probability based on the hypothesis.
- Now, let us state the theorem and its proof. Bayes theorem states that the conditional probability of an event A, given the occurrence of another event B, is equal to the product of the likelihood of B, given A and the probability of A.
- It is given as :

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)}$$

- Here, P (A) = how likely A happens (Prior knowledge) i.e., the probability of a hypothesis is true before any evidence is present.
- P (B) = how likely B happens (Marginalization) i.e., the probability of observing the evidence.
- P (A|B) = how likely A happens given that B has happened(Posterior) i.e., the probability of a hypothesis is true given the evidence.
- P (B|A) = how likely B happens given that A has happened (Likelihood) i.e., the probability of seeing the evidence if the hypothesis is true.

Ex. 1.5.1 : Assume that the chances of a person having a skin disease are 40%. Assuming that skin creams and drinking enough water reduces the risk of skin disease by 30% and prescription of a certain drug reduces its chance by 20%. At a time, a patient can choose any one of the two options with equal probabilities. It is given that after picking one of the options, the patient selected at random has the skin disease. Find the probability that the patient picked the option of skin screams and drinking enough water using the Bayes theorem.

☑ **Soln. :**

Assume E_1: The patient uses skin creams and drinks enough water; E_2: The patient uses the drug; A: The selected patient has the skin disease

$$P(E_1) = P(E_2) = ½$$

Using the probabilities known to us, we have

$$P(A \mid E_1) = 0.4 \times (1 - 0.3) = 0.28$$

$$P(A \mid E_2) = 0.4 \times (1 - 0.2) = 0.32$$

Using Bayes Theorem, the probability that the selected patient uses skin creams and drinks enough water is given by,

$$P(E_1 \mid A) = \frac{P(A \mid E_1)\, P(E_1)}{P(A \mid E_1)\, P(E_1) + P(A \mid E_2)\, P(E_2)}$$

$$= \frac{0.28 \times 0.5}{0.28 \times 0.5 + 0.32 \times 0.5} = \frac{0.28}{0.60} = 0.47$$

The probability that the patient picked the first option is 0.47

Ex. 1.5.2 : Three identical boxes contain red and white balls. The first box contains 3 red and 2 white balls, the second box has 4 red and 5 white balls, and the third box has 2 red and 4 white balls. A box is chosen very randomly and a ball is drawn from it. If the ball that is drawn out is red, what will be the probability that the second box is chosen?

Soln. :

Let A_1, A_2 and A_3 represent the events of choosing the first, second, and third box respectively, and let X be the event of drawing a red ball from the chosen box.

Then we are to find the value of $P(A_2 | X)$.

Since the boxes are identical, hence

$$P(A_1) = P(A_2) = P(A_3) = \frac{1}{3}$$

Again, by the problem

$$P(X | A_1) = \frac{3}{3+2} = \frac{3}{5};$$

$$P(X | A_2) = \frac{4}{4+5} = \frac{4}{9};$$

$$P(X | A_3) = \frac{2}{2+4} = \frac{1}{3}$$

Now, event X occurs if one of the mutually exclusive and exhaustive events A_1, A_2 and A_3 occurs. Therefore, using Bayes' theorem formula we get,

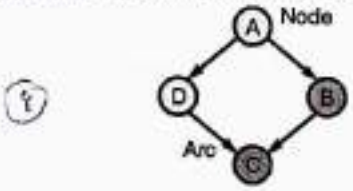
$$P(A_2 | X) = \frac{P(A_2) P(X | A_2)}{P(A_1) P(X | A_1) + P(A_2) P(X | A_2) + P(A_3) P(X | A_3)}$$

$$= \frac{\frac{1}{3} \times \frac{4}{9}}{\frac{1}{3} \times \frac{3}{5} + \frac{1}{3} \times \frac{4}{9} + \frac{1}{3} \times \frac{1}{3}} = \frac{\frac{4}{9}}{\frac{62}{45}} = \frac{10}{31}$$

1.6 BAYESIAN BELIEF NETWORKS

- Bayesian belief network is key computer technology for dealing with probabilistic events and to solve a problem which has uncertainty.
- We can define a Bayesian network as : "A Bayesian network is a probabilistic graphical model which represents a set of variables and their conditional dependencies using a directed acyclic graph."
- It is also called a Bayes network, belief network, decision network, or Bayesian model.
- Bayesian networks are probabilistic, because these networks are built from a probability distribution, and also use probability theory for prediction and anomaly detection.
- Real world applications are probabilistic in nature, and to represent the relationship between multiple events, we need a Bayesian network.
- It can also be used in various tasks including prediction, anomaly detection, diagnostics, automated insight, reasoning, time series prediction, and decision making under uncertainty.
- Bayesian Network can be used for building models from data and experts opinions, and it consists of two parts:
  - (a) Directed Acyclic Graph
  - (b) Table of conditional probabilities.
- The generalized form of Bayesian network that represents and solve decision problems under uncertain knowledge is known as an **Influence diagram**.
- A Bayesian network graph is made up of nodes and Arcs (directed links), where:
  - (a) Each node corresponds to the random variables, and a variable can be continuous or discrete.
  - (b) Arc or directed arrows represent the causal relationship or conditional probabilities between random variables. These directed links or arrows connect the pair of nodes in the graph. These links represent that one node directly influence the other node, and if there is no directed link that means that nodes are independent with each other.
- In Fig. 1.6.1 below, A, B, C, and D are random variables represented by the nodes of the network graph. If we are considering node B, which is connected with node A by a directed arrow, then node A is called the parent of Node B. Node C is independent of node A.
- The Bayesian network graph does not contain any cyclic graph. Hence, it is known as a **directed acyclic graph or DAG.**
- The Bayesian network has mainly two components: Causal Component and Actual numbers.

- Each node in the Bayesian network has condition probability distribution $P(X_i \mid Parent(X_i))$, which determines the effect of the parent on that node.
- Bayesian network is based on Joint probability distribution and conditional probability. So let's first understand the joint probability distribution :



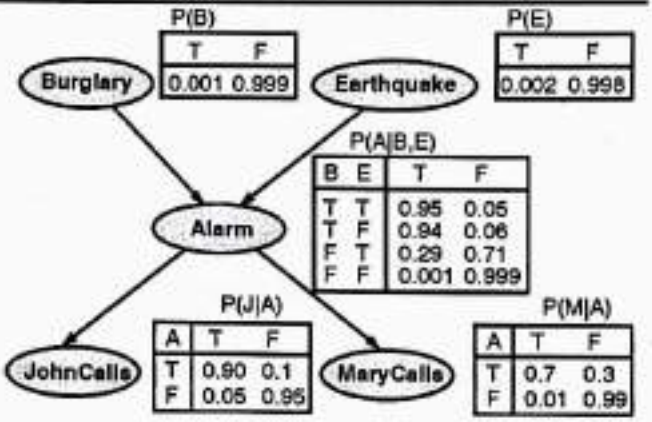
(1A1)Fig. 1.6.1 : Directed Acyclic Graph

- Let's understand the Bayesian network through an example by creating a directed acyclic graph.
- **Example :** Harry installed a new burglar alarm at his home to detect burglary. The alarm reliably responds at detecting a burglary but also responds for minor earthquakes. Harry has two neighbors John and Marry, who have taken a responsibility to inform Harry at work when they hear the alarm. John always calls Harry when he hears the alarm, but sometimes he got confused with the phone ringing and calls at that time too. On the other hand, Mary likes to listen to high music, so sometimes she misses to hear the alarm. Here we would like to compute the probability of Burglary Alarm.

Ex. 1.6.1 : Calculate the probability that alarm has sounded, but there is neither a burglary, nor an earthquake occurred, and John and Mary both called the Harry.

☑ **Soln. :**

- The Bayesian network for the above problem is given below.
- The network structure is showing that burglary and earthquake is the parent node of the alarm and directly affecting the probability of alarm's going off, but John and Mary's calls depend on alarm probability.



(1A2)Fig. P. 1.6.1

- The network is representing that our assumptions do not directly perceive the burglary and also do not notice the minor earthquake, and they also not confer before calling.
- The conditional distributions for each node are given as conditional probabilities table or CPT.
- Each row in the CPT must be sum to 1 because all the entries in the table represent an exhaustive set of cases for the variable.
- In CPT, a Boolean variable with k Boolean parents contains 2^K probabilities. Hence, if there are two parents, then CPT will contain 4 probability values.
- List of all events occurring in this network :
  - Burglary (B)
  - Earthquake (E)
  - Alarm (A)
  - John Calls (J)
  - Marry calls (M)
- We can write the events of problem statement in the form of probability :

$$P(M, J, A, \bar{B}, \bar{E}) = P(M \mid A) \times P(J \mid A) \times P(A \mid \bar{B} \cap \bar{E}) \times P(\bar{B}) \times P(\bar{E})$$

$$= 0.70 \times 0.90 \times 0.001 \times 0.999 \times 0.998$$

$$= \mathbf{0.0006281}$$

- Hence, a Bayesian network can answer any query about the domain by using Joint distribution.

1.6.1 The semantics of Bayesian Network

There are two ways to understand the semantics of the Bayesian network, which is given below :

1. **To understand the network as the representation of the Joint probability distribution.**

- It is helpful to understand how to construct the network.
- One way to define what the network means is to define how it represents a particular joint distribution over all variables. To do so, we must first retract what we said earlier about the parameters associated with each node.
- We stated that those parameters correspond to conditional probabilities $P(X_i \mid Parents\ (X_i))$; while this is true, we should think of them as numbers $\theta\ (X_i \mid Parents\ (X_i))$ until we assign semantics to the network as a whole.
- A generic entry in the joint distribution is the probability of a conjunction of particular assignments to each variable, such as $P\ (X_1 = x_1 \wedge \ldots \wedge X_n = x_n)$. We use the notation $P\ (x_1, \ldots, x_n)$ as an abbreviation for this.
- The value of this entry is given by the formula

$$P(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} \theta(x_i \mid \text{parents}(X_i))$$

- Where $parents\ (X_i)$ denotes the values of $Parents\ (X_i)$ that appear in $x_1, \ldots, x_n$. Thus, each entry in the joint distribution is represented by the product of the appropriate elements of the conditional probability tables (CPTs) in the Bayesian network.
- We can rewrite the above equation as

$$P(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid \text{parents}(X_i))$$

- The next step is to explain how to construct a Bayesian network in such a way that the resulting joint distribution is a good representation of a given domain. First, we rewrite the entries in the joint distribution in terms of conditional probability, using the product rule.

$$P(x_1, x_2, \ldots, x_n) = P(x_n \mid x_{n-1}, \ldots, x_1)\, P(x_{n-1}, \ldots, x_1)$$

- Then we repeat the process, reducing each conjunctive probability to a conditional probability and a smaller conjunction. We end up with one big product:

$$\begin{aligned} P(x_1, x_2, \ldots, x_n) &= P(x_n \mid x_{n-1}, \ldots, x_1) \\ &\quad P(x_{n-1} \mid x_{n-2}, \ldots, x_1) \\ &\quad \ldots P(x_2 \mid x_1)\, P(x_1) \\ &= \prod_{i=1}^{n} P(x_i \mid x_{i-1}, \ldots, x_1) \end{aligned}$$

- This is known as the **chain rule**. It holds true for any set of random variables.
- For every variable X_i in the network,
  $P(X_i \mid X_{i-1}, \ldots, X_1) = P(X_i \mid \text{Parents}(X_i))$, provided that $\text{Parents}(X_i) \subseteq \{X_{i-1}, \ldots, X_1\}$.
- The above equation says that the Bayesian network is a correct representation of the domain only if each node is conditionally independent of its other predecessors in the node ordering, given its parents.

2. **To understand the network as an encoding of a collection of conditional independence statements.**

- It is helpful in designing inference procedure.

1.6.2 Advantages and Disadvantages of Bayesian Belief Network

There are many Bayesian belief network advantages and disadvantages. They are listed below :

Advantages

There are a few advantages of Bayesian belief networks as it visualizes different probabilities of the variables. Some of them are:

- Graphical and visual networks provide a model to visualize the structure of the probabilities and develop designs for new models as well.
- Relationships determine the type of relationship and the presence or absence of it between variables.
- Computations calculate complex probability problems efficiently.

- Bayesian networks can investigate and tell you whether a particular feature is taken into a note for the decision-making process and can force it to include that feature if necessary. This network will ensure that all known features are investigated for deciding on a problem.
- Bayesian Networks are more extensible than other networks and learning methods. Adding a new piece in the network requires only a few probabilities and a few edges in the graph. So, it is an excellent network for adding a new piece of data to an existing probabilistic model.
- The graph of a Bayesian Network is useful. It is readable to both computers and humans; both can interpret the information, unlike some networks like neural networks, which humans can't read.

☞ Disadvantages

- The most significant disadvantage is that there is no universally acknowledged method for constructing networks from data. There have been many developments in this regard, but there hasn't been a conqueror in a long time. The design of Bayesian Networks is hard to make compared to other networks. It needs a lot of effort. Hence, only the person creating the network can exploit causal influences. Neural networks are an advantage compared to this, as they learn different patterns and aren't limited to only the creator.
- The Bayesian network fails to define cyclic relationships for example, deflection of airplane wings and fluid pressure field around it. The deflection depends on the pressure, and the pressure is dependent on the deflection. It is a tightly coupled problem which this network fails to define and make decisions.
- The network is expensive to build.
- It performs poorly on high dimensional data.
- It is tough to interpret and require copula functions to separate effects and causes.

⏭ 1.7 DECISION THEORY

- Decision theory is based on the axioms of probability and utility.
- Where probability theory provides a framework for coherent assignment of beliefs with incomplete information, utility theory introduces a set of principles for consistency among preferences and decisions.
- A decision is an irrevocable allocation of resources under control of the decision maker.
- Preferences describe a decision maker's relative valuations for possible states of the world, or outcomes.
- The valuation of an outcome may be based on the traditional attributes of money and time, as well as on other dimensions of value including pleasure, pain, life-years, and computational effort.
- Utility theory is based on a set of simple axioms or rules concerning choices under uncertainty. Like the axioms of probability theory, these rules are fairly intuitive.
- The first set of axioms concerns preferences for **outcomes under certainty**.

1. The axiom of **order ability** asserts that all outcomes are comparable, even if described by many attributes. Thus, for any two possible outcomes x and y, either one prefers x to y or one prefers y to x, or one is indifferent between them.
2. The axiom of **transitivity** asserts that these orderings are consistent; that is, if one prefers x to y and y to z, then one prefers x to z.

- The second set of axioms describes **preferences under uncertainty**. They involve the notion of a lottery, an uncertain situation with more than one possible outcome.
- Each outcome has an assignable probability of occurrence.

1. The **monotonicity** axiom says that, when comparing two lotteries, each with the same two alternative outcomes but different probabilities, a decision maker should prefer the lottery that has the higher probability of the preferred outcome.
2. The **decomposability** axiom says that a decision maker should be indifferent between lotteries that have the same set of eventual outcomes each with the same probabilities, even if they are reached by different means. For example, a lottery whose outcomes are other lotteries can be decomposed into an equivalent one-stage lottery using the standard rules of probability.

3. The **substitutability** axiom asserts that if a decision maker is indifferent between a lottery and some certain outcome (the certainty equivalent of the lottery), then substituting one for the other as a possible outcome in some more complex lottery should not affect her preference for that lottery.
4. Finally, the **continuity** axiom says that if one prefers outcome x to y, and y to z, then there is some probability p that one is indifferent between getting the intermediate outcome y for sure and a lottery with a p chance of x (the best outcome) and (1 - p) chance of z (the worst outcome).

- The consistency criteria embodied in classical decision theory can be stated as follows : Given a set of preferences expressed as a utility function, belief expressed as probability distributions, and a set of decision alternatives, a decision maker should choose that course of action that maximizes expected utility.
- The power of this result is that it allows preferences for complex and uncertain combinations of outcomes with multiple attributes to be computed from preferences expressed for simple components. Thus, it may be used as a tool to help people think about complex choices by decomposing them into simpler choices.
- Decision theory provides an appealing approach to **analytic tasks**, particularly to those involving inference and decision making under uncertainty. Consequently, we focus on expert systems for analytic tasks.
- Decision theory also can be relevant to **synthetic tasks**, because useful alternatives often must be selected from large numbers of options.

1.7.1 Types of Decision Making

1. **Decision making under certainty :** The outcome of a decision alternative is known (i.e., there is only one state of nature).
2. **Decision making under risk :** The outcome of a decision alternative is not known, but its probability is known.
3. **Decision making under uncertainty :** The outcome of a decision alternative is not known, and even its probability is not known.

A few criteria (approaches) are available for the decision makers to select according to their preferences and personalities under uncertainty

1. Maximax Criterion

- An adventurous and aggressive decision maker may choose the act that would result in the maximum payoff possible.
- This is viewed as an optimistic approach, "Best of bests".

▶ **Step 1 :** Pick maximum payoff of each alternative.

▶ **Step 2 :** Pick maximum of those maximums in Step 1; its corresponding alternative is the decision.

2. Maximin Criterion

- This is also called Waldian criterion.
- This criterion of decision making stands for choice between alternative courses of action assuming pessimistic view of nature.
- This is viewed as a pessimistic approach, "Best of worsts"

▶ **Step 1 :** Pick minimum payoff of each alternative

▶ **Step 2 :** Pick the maximum of those minimums in Step 1, its corresponding alternative is the decision

3. Minimax Criterion

- Application of the minimax criterion requires a table of losses or table of regret instead of gains.
- Regret is amount you give up due to not picking the best alternative in a given state of nature.

▶ **Step 1 :** Construct a 'regret table',

▶ **Step 2 :** Pick maximum regret of each row in regret table,

▶ **Step 3 :** Pick minimum of those maximums in Step 2. its corresponding alternative is the decision.

4. Laplace Criterion (Equally likelihood)

- The decision maker makes a simple assumption that each state of nature is equally likely to occur & compute average payoff for each.
- Choose decision with highest average payoff.

- **Step 1 :** Calculate the average payoff for each alternative.
- **Step 2 :** The alternative with highest average is the decision.

5. Hurwiczalpha Criterion (Rationality or Realism)

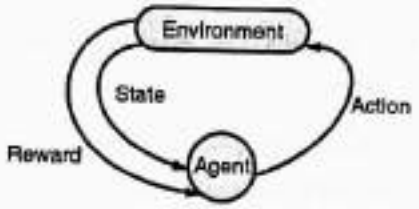
- This method is a combination of Maximin and Maximax criterion.
- Also known as criterion of rationality. neither too optimistic nor too pessimistic
- **Step 1 :** Calculate Hurwicz value for each alternative
- **Step 2 :** Pick the alternative of largest Hurwicz value as the decision.

Hurwicz value of an alternative = (row max) (α) + (row min) ($1 - \alpha$) where α ($0 \leq \alpha \leq 1$) is called coefficient of realism.

1.8 MARKOV DECISION PROCESS

- The Markov decision process (MDP) is a model of predicting outcomes.
- Markov Decision Process (MDP) is a mathematical framework to describe an environment in reinforcement learning.
- Markov decision processes give us a way to formalize sequential decision making. This formalization is the basis for structuring problems that are solved with reinforcement learning.

Components of an MDP



(1A3)**Fig. 1.8.1 : Components of Markov Decision Process**

1. **Agent :** An agent is the entity which we are training to make correct decisions (for example, a Robot that is being trained to move around a house without crashing).
2. **Environment :** The environment is the surrounding with which the agent interacts (for example, the house where the Robot moves). The agent cannot manipulate the environment; it can only control its own actions (for example, the Robot cannot control where a table is kept in the house, but it can walk around it to avoid crashing).
3. **State :** The state defines the current situation of the agent (for example, it can be the exact position of the Robot in the house, or the alignment of its two legs, or its current posture; it depends on how you address the problem).
4. **Action :** The choice that the agent makes at the current time step (for example, it can move its right or left leg, or raise its arm, or lift an object, turn right or left, etc.). We know the set of actions (decisions) that the agent can perform in advance.
5. **Reward :** A reward is the thought process behind picking an action. In practice, it is a probability distribution assigned to the set of actions. Highly rewarding actions will have a high probability and vice versa.

- In an MDP, we have a decision maker, called an *agent*, that interacts with the *environment* it's placed in. These interactions occur sequentially over time. At each time step, the agent will get some representation of the environment's *state*. Given this representation, the agent selects an *action* to take. The environment is then transitioned into a new state, and the agent is given a *reward* as a consequence of the previous action.
- This process of selecting an action from a given state, transitioning to a new state, and receiving a reward happens sequentially over and over again, which creates something called a *trajectory* that shows the sequence of states, actions, and rewards. Throughout this process, it is the agent's goal to maximize the total amount of rewards that it receives from taking actions in given states. This means that the agent wants to maximize not just the immediate reward, but the *cumulative* rewards it receives over time.
- In an MDP, we have a set of states S, a set of actions A, and a set of rewards R. We'll assume that each of these sets has a finite number of elements.
- At each time step $t = 0, 1, 2, \cdots$, the agent receives some representation of the environment's state

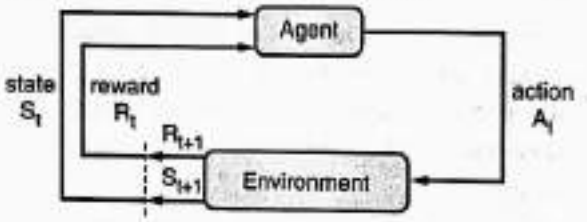
$S_t \in S$. Based on this state, the agent selects an action $A_t \in A$. This gives us the state-action pair (S_t, A_t).

- Time is then incremented to the next time step $t + 1$, and the environment is transitioned to a new state $S_{t+1} \in S$. At this time, the agent receives a numerical reward $R_{t+1} \in R$ for the action A_t taken from state S_t.
- We can think of the process of receiving a reward as an arbitrary function f that maps state-action pairs to rewards. At each time t, we have

$$f(S_t, A_t) = R_{t+1}$$

- The trajectory representing the sequential process of selecting an action from a state, transitioning to a new state, and receiving a reward can be represented as

$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \ldots$



(1A4)Fig. 1.8.2 : Agent- Environment Interaction

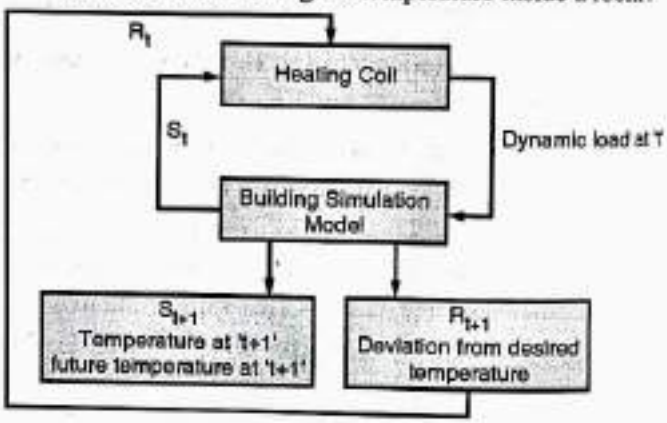
- The random variables R_t and S_t have well defined discrete probability distributions. These probability distributions are dependent only on the preceding state and action by virtue of Markov Property.
- Let S, A, and R be the sets of states, actions, and rewards. Then the probability that the values of S_t, R_t and A_t taking values s', r and a with previous state s is given by,

$$p(s', r \mid s, a) = P\{S_t = s', R_t = r \mid S_{t-1} = s, A_{t-1} = a\}$$

- The function p controls the dynamics of the process.

Example :

- Let us now discuss a simple example where Markov Decision Process with Reinforcement Learning can be used to implement a control strategy for a heating process.
- The idea is to control the temperature of a room within the specified temperature limits.
- The temperature inside the room is influenced by external factors such as outside temperature, the internal heat generated, etc.
- The agent, in this case, is the heating coil which has to decide the amount of heat required to control the temperature inside the room by interacting with the environment and ensure that the temperature inside the room is within the specified range.
- The reward, in this case, is basically the cost paid for deviating from the optimal temperature limits.
- The action for the agent is the dynamic load. This dynamic load is then fed to the room simulator which is basically a heat transfer model that calculates the temperature based on the dynamic load. So, in this case, the environment is the simulation model. The state variable S_t contains the present as well as future rewards.
- The following block diagram explains how MDP can be used for controlling the temperature inside a room :



(1A5)Fig. 1.8.3 : MDP for Controlling Temperature Inside Room

1.8.1 Limitations of Markov Decision Process

- Markov Decision Process uses Reinforcement learning that learns from the state. The state is the input for policymaking. Hence, the state inputs should be correctly given.
- Also as we have seen, there are multiple variables and the dimensionality is huge. So using it for real physical systems would be difficult!

1.9 MULTIPLE CHOICE QUESTIONS

Q. 1.1 Which of the following is not the leading cause of uncertainty to occur in the real world ?

(a) Information occurred from reliable sources
(b) Experimental Errors
(c) Equipment fault
(d) Temperature variation

✓Ans. : (a)

Q. 1.2 What is the major problem for AI systems while solving the real world problems?

(a) Uncertainty in Environment
(b) Poor battery life of the system
(c) Improper training time
(d) Set-up cost

✓Ans. : (a)

Q. 1.3 What would be the probability of an event 'G' if H denotes its complement, according to the axioms of probability?

(a) $P(G) = 1 / P(H)$
(b) $P(G) = 1 - P(H)$
(c) $P(G) = 1 + P(H)$
(d) $P(G) = P(H)$

✓Ans. : (b)

Q. 1.4 _____ is the process of calculating a probability distribution of interest e.g. $P(A \mid B = True)$, or $P(A, B \mid C, D = True)$.

(a) Diagnostics
(b) Supervised anomaly detection
(c) Inference
(d) Prediction

✓Ans. : (c)

Q. 1.5 Naina receives emails that consists of 18% spam of those emails. The spam filter is 93% reliable i.e., 93% of the mails it marks as spam are actually a spam and 93% of spam mails are correctly labelled as spam. If a mail marked spam by her spam filter, determine the probability that it is really spam.

(a) 50% (b) 84%
(c) 39% (d) 63%

✓Ans. : (a)

Q. 1.6 Mangoes numbered 1 through 18 are placed in a bag for delivery. Two mangoes are drawn out of the bag without replacement. Find the probability such that all the mangoes have even numbers on them?

(a) 43.7% (b) 34%
(c) 6.8% (d) 9.3%

✓Ans. : (c)

Q. 1.7 A bucket contains 6 blue, 8 red and 9 black pens. If six pens are drawn one by one without replacement, find the probability of getting all black pens?

(a) 8/213 (b) 8/4807
(c) 5/1204 (d) 7/4328

✓Ans. : (b)

Q. 1.8 Where can the Bayes rule can be used?

(a) Solving queries
(b) Increasing complexity
(c) Decreasing complexity
(d) Answering probabilistic query

✓Ans. : (d)

Q. 1.9 What does the Bayesian network provides?

(a) Complete description of the domain
(b) Partial description of the domain
(c) Complete description of the problem
(d) Partial description of the problem

✓Ans. : (a)

Q. 1.10 How can the Bayesian network be used to answer any query?

(a) Full distribution
(b) Joint distribution
(c) Partial distribution
(d) All of the mentioned

✓Ans. : (b)

Q. 1.11 How many terms are required for building a bayes model?

(a) 1 (b) 2 (c) 3 (d) 4

✓Ans. : (c)

Q. 1.12 Bayesian Belief Network is also known as _____.

(a) belief network
(b) decision network
(c) Bayesian model
(d) All of the above

✓Ans. : (d)

Q. 1.13 If we have variables x_1, x_2, x_3,..... x_n, then the probabilities of a different combination of x_1, x_2, $x_3 \ldots x_n$, are known as?

(a) Table of conditional probabilities
(b) Causal Component
(c) Actual numbers
(d) Joint probability distribution

✓Ans. : (d)

Q. 1.14 Suppose you are creating a Bayesian network. Which of the following is the outcome between a node and its predecessors?

(a) Conditionally independent
(b) Dependant
(c) Functionally dependent
(d) Both Conditionally dependant & Dependant

✓Ans. : (a)

Q. 1.15 The generalized form of Bayesian network that represents and solve decision problems under uncertain knowledge is known as _____.

(a) Directed Acyclic Graph
(b) Table of conditional probabilities
(c) Influence diagram
(d) Directed Cyclic Graph

✓**Ans. : (c)**

Q. 1.16 Which is true for Decision theory?

(a) Decision Theory = Probability theory + utility theory
(b) Decision Theory = Inference theory + utility theory
(c) Decision Theory = Uncertainty + utility theory
(d) Decision Theory = Probability theory + preference

✓**Ans. : (c)**

Q. 1.17 What does MDP stand for?

(a) Markov Delicate Program
(b) Markov Decision Process
(c) Modern Derivative Parallel
(d) Modern Design Process

✓**Ans. (b)**

Q. 1.18 The state and reward at time t depend on which of the following?

(a) State-action pair for all time instances before t
(b) State-action pair for time (t-1)
(c) Agent Dynamics
(d) Cumulative reward at time t

✓**Ans. : (b)**

Q. 1.19 In an MDP, the _____'s objective is to maximize the total reward.

(a) Environment (b) Action
(c) Agent (d) State

✓**Ans. : (c)**

Q. 1.20 What is the correct order for the components of an MDP?

(a) Environment, Agent, State, Action, Reward
(b) State, Agent, Environment, Action, Reward
(c) Agent, State, Environment, Action, Reward
(d) Agent, Environment, State, Action, Reward

✓**Ans. : (d)**

DESCRIPTIVE QUESTIONS

Q. 1 Explain the concept of uncertainty in AI.

Q. 2 Illustrate inference with joint distribution using suitable example.

Q. 3 Describe Baye's theorem with suitable example.

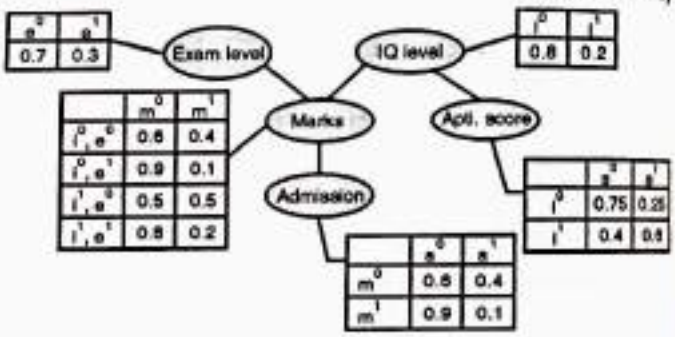
Q. 4 Explain Bayesian Belief Network with suitable example.

Q. 5 Imagine that we are given the task of modeling a student's marks (*m*) for an exam he has just given. From the given Bayesian Network Graph below:

(a) Calculate the probability that in spite of the exam level being difficult, the student having a low IQ level and a low Aptitude Score, manages to pass the exam and secure admission to the university. **(Ans. : 0.0018)**

(b) Calculate the probability that the student has a High IQ level and Aptitude Score, the exam being easy yet fails to pass and does not secure admission to the university.

(Ans. : 0.0252)



(1A8) **Fig. Q. 5**

Q. 6 Explain the concept of decision theory under uncertainty.

Q. 7 State and explain the components of Markov Decision Process (MDP).

Q. 8 Explain Markov Decision Process with suitable example.

☞ Self-Learning Topics

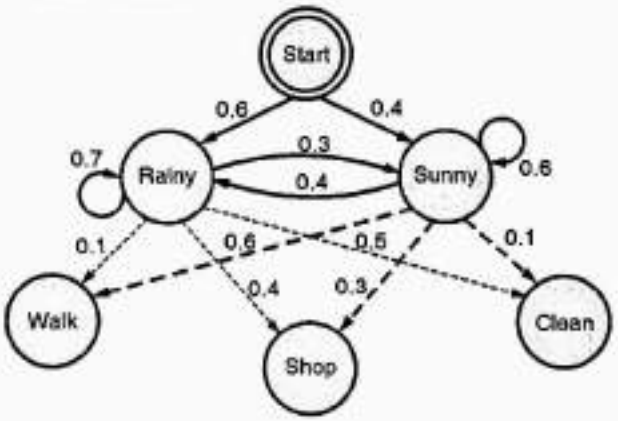
1. Hidden Markov Model

- A Hidden Markov Model (HMM) is a statistical model which is also used in machine learning.
- It can be used to describe the evolution of observable events that depend on internal factors, which are not directly observable.
- These are a class of probabilistic graphical models that allow us to predict a sequence of unknown variables from a set of observed variables.

- The main goal of HMM is to learn about a Markov chain by observing its hidden states.
- Considering a Markov process X with hidden states Y here the HMM solidifies that for each time stamp the probability distribution of Y must not depend on the history of X according to that time.
- Hidden Markov models are especially known for their application in reinforcement learning and temporal pattern recognition such as speech, handwriting, gesture recognition, part-of-speech tagging, musical score following, partial discharges and bioinformatics.

Terminology in HMM

- The term hidden refers to the first order Markov process behind the observation.
- Observation refers to the data we know and can observe.
- Markov process is shown by the interaction between "Rainy" and "Sunny" in the below diagram and each of these are **HIDDEN STATES**.



(IA7)Fig. (a)

- **OBSERVATIONS** are known data and refers to "Walk", "Shop", and "Clean" in the above diagram. In machine learning sense, observation is our training data, and the number of hidden states is our hyper parameter for our model.
- **State transition probabilities** are the arrows pointing to each hidden state.
- **Observation probability matrix** are the arrows pointing to each observation from each hidden state. The matrix is row stochastic meaning the rows add up to 1.
- The states and observation are:
  states = ('Rainy', 'Sunny')
  observations = ('walk', 'shop', 'clean')
- And the start probability is:
  start_probability = {'Rainy': 0.6, 'Sunny': 0.4}
- Now the distribution of the probability has the weightage more on the rainy day stateside so we can say there will be more chances for a day to being rainy again and the probabilities for next day weather states are as following:

```
transition_probability = {
    'Rainy' : {'Rainy': 0.7, 'Sunny': 0.3},
    'Sunny' : {'Rainy': 0.4, 'Sunny': 0.6},
    }
```

- From the above we can say the changes in the probability for a day is transition probabilities and according to the transition probability the emitted results for the probability of work that will be performed is

```
emission_probability = {
'Rainy' : {'jog': 0.1, 'work': 0.4, 'clean': 0.5},
'Sunny' : {'jog': 0.6, 'work: 0.3, 'clean': 0.1},
}
```

- This probability can be considered as the emission probability. Using the emission probability one can predict the states of the weather.

2. Gaussian Mixture Model

- Gaussian mixture models (GMMs) are a probabilistic model for representing normally distributed subpopulations within an overall population.
- Mixture models in general don't require knowing which subpopulation a data point belongs to, allowing the model to learn the subpopulations automatically.
- Since subpopulation assignment is not known, this constitutes a form of unsupervised learning.
- For example, in modeling human height data, height is typically modeled as a normal distribution for each gender with a mean of approximately 5'10" for males and 5'5" for females.

- Given only the height data and not the gender assignments for each data point, the distribution of all heights would follow the sum of two scaled (different variance) and shifted (different mean) normal distributions.
- A model making this assumption is an example of a Gaussian mixture model (GMM), though in general a GMM may have more than two components.
- Estimating the parameters of the individual normal distribution components is a canonical problem in modeling data with GMMs.
- GMMs have been used for feature extraction from speech data, and have also been used extensively in object tracking of multiple objects, where the number of mixture components and their means predict object locations at each frame in a video sequence.
- A Gaussian mixture model is parameterized by two types of values, the mixture **component weights** and the component **means** and **variances/covariances**. For a Gaussian mixture model with K components, the k^{th} component has a mean of μ_k and variance of σ_k for the univariate case and a mean of $\vec{\mu_k}$ and covariance matrix of Σ_k for the multivariate case.
- The mixture component weights are defined as ϕ_k for component C_k, with the constraint that $\sum_{i=1}^{k} \phi_i = 1$ so that the total probability distribution normalizes to 1.
- If the component weights aren't learned, they can be viewed as an a-priori distribution over components such that p (x generated by component C_k) = ϕ_k.
- If they are instead learned, they are the a-posteriori estimates of the component probabilities given the data.

Top of Form

One-dimensional Model	Multi-dimensional Model
$p(x) = \sum_{i=1}^{k} \phi_i N(x \mid \mu_i, \sigma_i)$	$p\left(\vec{x}\right) = \sum_{i=1}^{k} \phi_i N\left(\vec{x} \mid \vec{\mu_i}, \Sigma_i\right)$
$N(x \mid \mu_i, \sigma_i) = \frac{1}{\sigma_i \sqrt{2\pi}} \exp\left(-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right)$	$N\left(\vec{x} \mid \vec{\mu_i}, \Sigma_i\right) = \frac{1}{\sqrt{(2\pi)^k \lvert\Sigma_i\rvert}} \exp\left(-\frac{1}{2}\left(\vec{x}-\vec{\mu_i}\right)^T \Sigma_i^{-1}\left(\vec{x}-\vec{\mu_i}\right)\right)$
$\sum_{i=1}^{k} \phi_i = 1$	$\sum_{i=1}^{k} \phi_i = 1$

Chapter Ends...

□□□

MODULE 2

CHAPTER 2

Cognitive Computing

University Prescribed Syllabus w.e.f Academic Year 2022-2023

Foundation of Cognitive Computing, Design Principles for Cognitive Systems, Natural Language Processing in Support of a Cognitive System, Representing Knowledge in Taxonomies and Ontologies, Applying Advanced Analytics to Cognitive Computing, The Process of Building a Cognitive Application.

Self-learning Topics : Cognitive Systems such as IBM's Watson.

2.1 FOUNDATION OF COGNITIVE COMPUTING

- **Cognitive computing** is the use of computerized models to simulate the human thought process in complex situations where the answers may be ambiguous and uncertain.
- The phrase is closely associated with IBM's cognitive computer system, Watson.
- Computers are faster than humans at processing and calculating, but they have yet to master some tasks, such as understanding natural language and recognizing objects in an image. Cognitive computing is an attempt to have computers mimic the way a human brain works.
- To accomplish this, cognitive computing makes use of artificial intelligence (AI) and other underlying technologies, including expert systems, neural networks, machine learning, deep learning, natural language processing, speech recognition, object recognition and robotics.
- Cognitive computing uses these processes in conjunction with self-learning algorithms, data analysis and pattern recognition to teach computing systems.
- The learning technology can be used for speech recognition, sentiment analysis, risk assessments, face detection and more.
- **Siri, Google Assistant, Cortana, and Alexa** are few of the best examples of Cognitive computing.
- In addition, it is particularly useful in fields such as healthcare, banking, finance and retail.
- Cognitive computing works on the principle of :

(1) Learn
(2) Model
(3) Generate Hypothesis

(1) Learn

- A cognitive system can learn.
- Based on training and observations from all types, volumes, and speeds of data, the system uses data to draw inferences about a domain, a topic, a person, or a problem.

(2) Model

- In order for the system to learn, it must first construct a model or representation of the domain (which contains both internal and external data) as well as assumptions that determine which learning methods are utilized.
- A cognitive system's ability to comprehend the context of how data fits into the model is critical.

(3) Generate Hypothesis

- There is no single proper solution, according to a cognitive system. The best response is determined by the data.
- As a result, a probabilistic cognitive system exists. A hypothesis is a potential explanation for some previously known data. To train, test, or score a hypothesis, a cognitive system uses data.

2.1.1 Distinguishing Features of a Cognitive System

There are some traits that cognitive systems share, despite the fact that there are many diverse approaches on how they will be created. They consist of the ability to :

(1) Without reprogramming, it may learn from experience and use data/evidence to increase its own knowledge and performance.

(2) Based on existing level of knowledge, generate and/or evaluate contradictory hypotheses.

(3) Report findings in a way that validates conclusions based on the evidence's reliability.

(4) Discover patterns in data, either with or without explicit user guidance on the pattern's nature.

(5) Emulate natural learning systems' processes or architecture (that is, memory management, knowledge organization processes, or modelling the neurosynaptic brain structures and processes).

(6) Use natural language processing (NLP) to extract meaning from text, and deep learning to extract features from pictures, video, speech, and sensors.

(7) Make use of a number of statistical and predictive analytics algorithms.

2.1.2 Working of Cognitive System

- Systems used in the cognitive sciences combine data from various sources while weighing context and conflicting evidence to suggest the best possible answers.
- To achieve this, cognitive systems include self-learning technologies that use data mining, pattern recognition and NLP to mimic human intelligence.
- Using computer systems to solve the types of problems that humans are typically tasked with requires vast amounts of structured and unstructured data fed to machine learning algorithms.
- Over time, cognitive systems are able to refine the way they identify patterns and the way they process data. They become capable of anticipating new problems and modeling possible solutions.
- For example, by storing thousands of pictures of dogs in a database, an AI system can be taught how to identify pictures of dogs. The more data a system is exposed to, the more it is able to learn and the more accurate it becomes over time.

To achieve those capabilities, cognitive computing systems must have the following attributes :

(1) Adaptive	(2) Interactive
(3) Iterative and stateful	(4) Contextual

(1) Adaptive

- These systems must be flexible enough to learn as information changes and as goals evolve.
- They must digest dynamic data in real time and adjust as the data and environment change.

(2) Interactive

- Human-computer interaction is a critical component in cognitive systems.
- Users must be able to interact with cognitive machines and define their needs as those needs change.
- The technologies must also be able to interact with other processors, devices and cloud platforms.

(3) Iterative and stateful

- Cognitive computing technologies can ask questions and pull in additional data to identify or clarify a problem.
- They must be stateful in that they keep information about similar situations that have previously occurred.

(4) Contextual

- Understanding context is critical in thought processes.
- Cognitive systems must understand, identify and mine contextual data, such as syntax, time, location, domain, requirements and a user's profile, tasks and goals.
- The systems may draw on multiple sources of information, including structured and unstructured data and visual, auditory and sensor data.

2.1.3 Examples and Applications of Cognitive Computing

Cognitive computing systems are typically used to accomplish tasks that require the parsing of large amounts of data. For example, in computer science, cognitive computing aids in big data analytics, identifying trends and patterns, understanding human language and interacting with customers.

Examples of how cognitive computing is used in various industries include the following :

(1) Healthcare	(2) Retail
(3) Banking and finance	(4) Logistics

(1) Healthcare

- Cognitive computing can deal with large amounts of unstructured healthcare data such as patient histories, diagnoses, conditions and journal research articles to make recommendations to medical professionals.
- This is done with the goal of helping doctors make better treatment decisions.
- Cognitive technology expands a doctor's capabilities and assists with decision-making.

▶ **(2) Retail**

- In retail environments, these technologies analyze basic information about the customer, along with details about the product the customer is looking at.
- The system then provides the customer with personalized suggestions.

▶ **(3) Banking and finance**

- Cognitive computing in the banking and finance industry analyzes unstructured data from different sources to gain more knowledge about customers.
- NLP is used to create chatbots that communicate with customers. This improves operational efficiency and customer engagement.

▶ **(4) Logistics**

- Cognitive computing aids in areas such as warehouse management, warehouse automation, networking and IoT devices.
- IBM's Watson for Oncology is an example of a cognitive computing system.
- It provides oncologists at Memorial Sloan Kettering Cancer Center in New York with evidence-based treatment options for cancer patients.
- When medical staff input questions, Watson generates a list of hypotheses and offers treatment options for doctors to consider.
- Watson Health is another IBM tool that helps clients in medical and clinical research.

2.1.4 Advantages of Cognitive Computing

Advantages of cognitive computing include positive outcomes in the following areas :

(1) **Analytical accuracy** : Cognitive computing is proficient at juxtaposing and cross-referencing structured and unstructured data.

(2) **Business process efficiency** : Cognitive technology can recognize patterns when analyzing large data sets.

(3) **Customer interaction and experience** : The contextual and relevant information that cognitive computing provides to customers through tools like chatbots improves customer interactions. A combination of cognitive assistants, personalized recommendations and behavioral predictions enhances customer experience.

(4) **Employee productivity and service quality** : Cognitive systems help employees analyze structured or unstructured data and identify data patterns and trends.

2.1.5 Disadvantages of Cognitive Computing

Cognitive technology also has downsides, including the following :

(1) **Security challenges** : Cognitive systems need large amounts of data to learn from. Organizations using the systems must properly protect that data -- especially if it is health, customer or any type of personal data.

(2) **Long development cycle length** : These systems require skilled development teams and a considerable amount of time to develop software for them. The systems themselves need extensive and detailed training with large data sets to understand given tasks and processes.

(3) **Slow adoption** : The slow development lifecycle is one reason for slow adoption rates. Smaller organizations may have more difficulty implementing cognitive systems and therefore avoid them.

(4) **Negative environmental impact** : The process of training cognitive systems and neural networks consumes a lot of power and has a sizable carbon footprint.

2.1.6 AI Vs Cognitive Computing

Sr. No.	Artificial Intelligence	Cognitive Computing
1.	Algorithm of AI generates the most accurate result without the utilization of human input.	Cognitive Computing is based on human input i.e. thinking, reasoning, and belief to generate output.
2.	AI is autonomous.	Cognitive is dependent.
3.	Machine as an author of its own actions. It does the work of the human brain.	Machine as an agent of some business process or intention of a human being. It is just an information tool.
4.	It reflects the reality.	It copies human behavior.
5.	AI itself generates the algorithm to produce end results and decisions.	It generates only the information and allows the end result to be interpreted by humans itself.
6.	Utilizes pre-trained algorithms.	Utilizes prediction and analysis as a basic tool.
7.	To produce results, AI finds the hidden information and uses a specific pattern.	To generate solutions, it imitates the human thought process. Helps for smarter decisions.
8.	Retail, finance, and manufacturing security are a few areas that use AI.	Cognitive Computing enhances process across various fields viz. industries, customer service, health care.
9.	Job of AI is to make our work easier.	If we complex human-like decisions, Cognitive computing comes into hand.
10.	Technologies, where AI is utilized, are NLP, speech recognition, image processing, video analytics, chatbots.	Cognitive shines when there is a need for sentiment analysis, facial recognition, fraud detection, risk assessment.

2.1.7 Elements of a Cognitive System

A cognitive system is made up of several components, ranging from hardware and deployment models to machine learning and applications. Although there are several techniques to create a cognitive system, some common elements must be incorporated. Fig. 2.1.1 shows an overview of the architecture for a cognitive system.

(1) Infrastructure and Deployment Modalities

- It is important in a cognitive system to have a flexible and agile infrastructure to accommodate applications that evolve over time.
- A wide range of public and private data must be maintained and processed as the market for cognitive technologies grows.
- Furthermore, enterprises can use Software as a Service (SaaS) applications and services to satisfy industry needs.
- A highly parallelized and distributed environment must be provided, including computing and storage cloud services.

(2) Data Access, Metadata, and Management Services

- Cognitive computing is data-centric. So data procurement, access, and management play a vital role. Therefore, before adding and using such data, we need to have a range of underlying services.
- Preparing to use the captured data requires an understanding of the origins and derivation of that data. Therefore, there must be a way to classify the characteristics of that data, such as when that text or data source was created and by whom.
- In a cognitive system these data sources are not static. There will be a variety of internal and external data sources that will be included in the corpus.
- To make sense of these data sources, we need to have a set of management services that prepare the data for use within the corpus. Therefore, as in a traditional system, the data must be examined, cleaned and monitored to verify its accuracy.

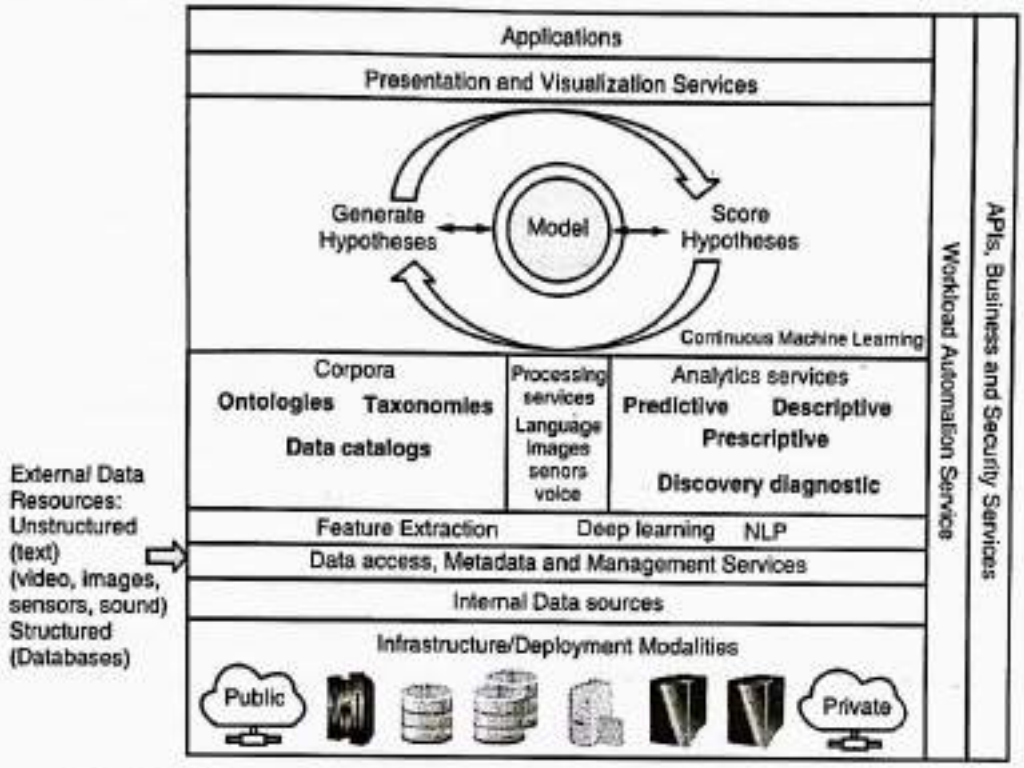


Fig. 2.1.1 : Elements of a cognitive system (image credit: Horovitz, Kaufman, Bowles)

(3) The Corpus, Taxonomies, and Data Catalogs

- The corpus and data analytics services are tightly associated with the data access and management layer.
- A corpus is a repository of data that is used to manage codified knowledge. This corpus contains the information necessary to establish the domain of the system.
- Fig. 2.1.1 shows the different types of data fed into the system. The majority of this data comes from text-based sources (documents, textbooks, patient notes, customer reports, etc.).
- Cognitive systems use a variety of unstructured and semi-structured inputs including video, images, sensors, and sound.
- In addition, the corpus may contain ontologies that describe certain entities and their relationships.
- Ontologies are frequently created by industry organizations to identify industry-specific items, such as conventional chemical compounds, machine parts, or medical disorders and treatments.
- Frequently, cognitive systems must use a subset of an industry-based ontology to contain just the facts relevant to the cognitive system's objectives.
- Ontologies and taxonomies go hand in hand. Within an ontology, a taxonomy gives context.

(4) Data Analytics Services

- Incorporating data analytics into the corpus involves techniques for gaining a deeper understanding of it.
- By using structured, unstructured, and semi-structured data that has been ingested, users can begin to predict outcomes, discover patterns, and determine next best actions.
- These services do not live in isolation. They retrieve data continuously from the data access layer and pull data from the corpus.
- Several advanced algorithms are used to develop the cognitive system model.

(5) Continuous Machine Learning

- Machine learning is a method for data to learn without having to be actively programmed.

- The cognitive system does not exist in a static state. Models are regularly updated as new data, analytics, and interactions become available.
- The two fundamental components of the machine learning process are hypothesis generation and hypothesis evaluation.

(6) Hypothesis Generation and Evaluation

- A hypothesis is an evidence-based testable statement that explains some observable occurrence.
- In a cognitive computing system, evidence is sought to support or disprove assumptions.
- You must collect data from diverse sources, build models, and then evaluate how well the models perform. This is accomplished through an iterative data training procedure.
- Training can take place automatically based on data analysis by the system, or it might include human end users.
- After training, it becomes evident whether the hypothesis is supported by the data. If the data does not support the hypothesis, the user has numerous alternatives.
- The user, for example, can refine the data by adding to the corpus or changing the hypothesis.
- To evaluate the hypothesis, constituents that utilize the cognitive system must work together.
- The evaluation of results, like the formation of the hypothesis, refines those results and trains them again.

(7) Tools and The Learning Process

- To learn from data, you must have tools that can handle both structured and unstructured data.
- NLP services can evaluate and find patterns in unstructured textual input to help a cognitive system.
- Deep learning tools are required for unstructured data such as photos, videos, and audio.
- Sensor data is critical in evolving cognitive systems. Sensor data is used in industries ranging from transportation to healthcare to monitor speed, performance, failure rates, and other parameters, and this data is then captured and analyzed in real time to anticipate behavior and influence results.

(8) Presentation and Visualization Services

- New visualization interfaces are required to analyze sophisticated and often large amounts of data.
- Data visualization is the visual representation of data as well as the visual analysis of data.
- A bar chart or pie chart, for example, is a visual representation of underlying data.
- When data patterns and relationships are depicted with structure, color, and other elements, they are simpler to detect and comprehend.
- Static and dynamic data visualizations are the two fundamental forms. Interactivity may be required in either or both circumstances.
- Looking at a data visualization isn't always enough. You must drill down, reposition, expand and contract, and so on. This interaction allows you to "personalize" the data views so that you may seek nonobvious representations of data, relationships, and alternatives.
- Color, position, and proximity may all influence visualization. Shape, size, and motion are all important factors in visualization.
- Presentation services prepare findings for output.
- Visualization services aid in the communication of results by demonstrating the relationships between data points.

(9) Cognitive Applications

- To allow the development of applications that handle business challenges in vertical domains, the Cognitive system must harness all of these fundamental capacities.
- These apps may need to incorporate procedures to acquire insight into a complex business (ex: healthcare, logistics, preventive maintenance, etc.).
- It should give consumers with insights so they may make better decisions based on data that exists but is not easily accessible.

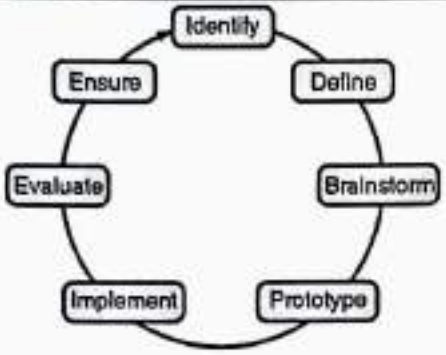


Fig. 2.2.1 : Cognitive System Design Principles

- The maturity of a cognitive model is measured by the confidence level of the recommendations or the actions taken by a cognitive process.
- Hence, improving the confidence level of a system plays a vital role in making the cognitive process successful.
- There are multiple factors which can influence the confidence level depending on the data it has access to.
- In a normal scenario, the confidence level can be improved based on the human actions taken to the recommendations suggested by the cognitive system.
- These are self-evolving and require human intervention for updating the context and improving the confidence level.

2.3 NATURAL LANGUAGE PROCESSING IN SUPPORT OF A COGNITIVE SYSTEM

- Cognitive computing is a technology that helps in mimicking the human thought process.
- Most of the people believe that cognitive computing is a standalone technology. However, cognitive computing is a combination of multiple technologies. The key technologies that fuel cognitive computing are machine learning, NLP, machine reasoning, speech recognition, object recognition or computer vision, dialog systems, and human-computer interaction.
- NLP plays a vital role in building cognitive systems as natural language understanding and natural language generation (NLG) have been an inevitable human feature.
- Natural language processing (NLP) is a core ability of cognitive computing systems and is often defined as helping computers process and understand human language.
- On a very basic level, NLP enables computers to understand language by putting words together in meaningful phrases, assigning meaning to those phrases and drawing inferences from them.
- Some of the most well-known components of NLP are part-of-speech tagging, named entity resolution, word sense disambiguation and coreference resolution, each of which plays a vital role in identifying and characterizing the core text that carries the primary meaning of a phrase or sentence.
- Other deep technical processes behind NLP include machine learning techniques, computational linguistics and statistics across training corpora.
- The ability to process language naturally allows NLP applications to summarize documents, auto-classify text, conduct sentiment analysis and provide search results with enhanced relevance ranking.
- Both NLP and cognitive computing rely on each other. NLP aids cognitive computing and cognitive computing aids NLP.
- NLP itself can be seen as a cognitive technology because it uses sensory perceptions such as audio and visual perceptions as the primary step for an NLP task.
- Fig 2.3.1 shows the cognitive approach to NLP.

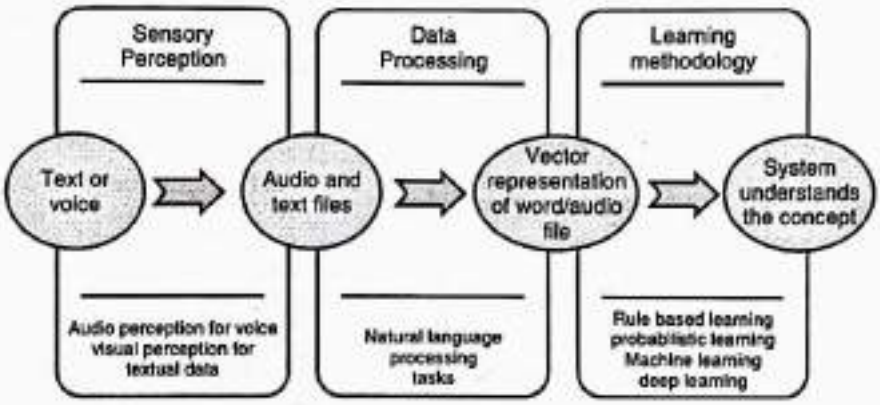


Fig. 2.3.1 : Cognitive Approach to NLP

2.3.1 Concepts of NLP

For understanding NLP, one needs to know the basics of natural language, that is, linguistics and the steps involved in processing it.

2.3.1.1 Linguistic Terms

1. Phonology/ Phonetics

- Phonology is the study of speech sounds used in a particular language.
- Every alphabet has a sound associated with it, and a word is pronounced by combining those sounds.
- Word pronunciation can be explained through phonetics.
- For example, in the English language, "read" in the present tense (reed) and "read" in the past tense (red) have different meanings.

2. Morphology

- Morphology is the study of structures of words/formation of words.
- A morpheme is the smallest individual unit of language that has a specific meaning. Morphemes can be words, prefixes, or even suffixes.
- For example, the word "unfairness" means a lack of justice or inequality.
- Morphemes out of this would be:

  Un (not) – prefix

  fair (treating people equally) - the root word

  ness (being in a state) - suffix

3. Syntax

- The syntax is nothing but the structure of language.
- Syntax analysis is the study of the structural relationships among words in a sentence or how words are grouped to form sentences.
- Every language follows a rule for creating meaningful sentences.
- Subject, verb, object, parts of speech (POS), etc., help in the formation of meaningful sentences.
- In English linguistics, subject-verb-object is a sentence structure where the subject comes first, the verb second, and the object third.
- The POS categorize words according to then usage.
- For example, noun is used to represent the name of a place, person, or thing.

4. Semantics

- Semantics is the study of the meaning of words in a sentence and how these words are combined to form meaningful sentences.
- Lexical semantics analyze the relation between words such as synonyms, hyponyms, and others.
- Semantics try to interpret the meaning of a sentence by combining and finding a relationship between word meanings.
- For example, consider a sentence extracted from a

paragraph. "That company is facing a huge financial crisis now, and in May it may vary." In this case, "May" represents a month and "may" represent a word. This type of words can confuse the machine in finding the proper meaning of a sentence.

5. Pragmatics

- Pragmatics studies the situational use of language sentences.
- It is slightly different from semantics. Semantics try to interpret the meaning of a sentence by combining the meaning of words, whereas pragmatics try to find the meaning of a sentence based on the situation.
- Consider the proverb, "Don't judge a book by its cover." Its semantic meaning is the same as the word meaning in the sentence, but its pragmatic meaning is "Don't judge someone or something by appearance alone."

6. Discourse

- Discourse is a group of sentences, and it studies or finds the actual meaning of the context by connecting component sentences.
- "Albin took a book from the library. Then he went to the coffee shop, and he left the book there."
- The above context leads to more than one inferences, and it can answer different questions. Consider the question, "Where is the book now?" The answer is, "Book is in the coffee shop."

7. Word Sense Disambiguation (WSD)

- Word sense disambiguation, in natural language processing (NLP), may be defined as the ability to determine which meaning of word is activated by the use of word in a particular context.
- Lexical ambiguity, syntactic or semantic, is one of the very first problem that any NLP system faces.
- Part-of-speech (POS) taggers with high level of accuracy can solve Word's syntactic ambiguity.
- On the other hand, the problem of resolving semantic ambiguity is called WSD (word sense disambiguation). Resolving semantic ambiguity is harder than resolving syntactic ambiguity.
- For example, consider the two examples of the distinct sense that exist for the word *"bass"*:
  (i) I can hear bass sound.
  (ii) He likes to eat grilled bass.
- The occurrence of the word **bass** clearly denotes the distinct meaning. In first sentence, it means **frequency** and in second, it means **fish**.
- Hence, if it would be disambiguated by WSD then the correct meaning to the above sentences can be assigned as follows:
  (i) I can hear bass/frequency sound.
  (ii) He likes to eat grilled bass/fish.

2.3.1.2 Phases in NLP

In this section we will see what are typical steps involved while performing NLP tasks. We should keep in mind that the below section describes some standard workflow, it may however differ drastically as we do real life implementations basis on our problem statement or requirements.

The source of Natural Language could be Speech (sound) or Text.

(1) Phonological Analysis

- This level is applied only if the text origin is a speech.
- It deals with the interpretation of speech sounds within and across words.
- Speech sound might give a big hint about the meaning of a word or a sentence.
- It is study of organizing sound systematically.

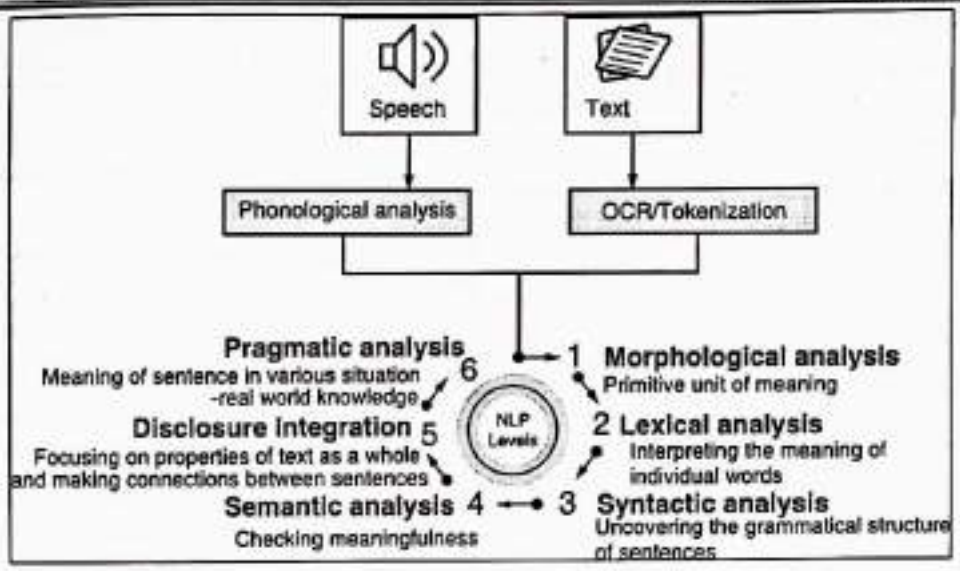


Fig. 2.3.2 : Phases in NLP

(2) Morphological Analysis

- Deals with understanding distinct words according to their morphemes (the smallest units of meanings).
- Taking, for example, the word: "unhappiness". It can be broken down into three morphemes (prefix, stem, and suffix), with each conveying some form of meaning: the prefix un- refers to "not being", while the suffix -ness refers to "a state of being". The stem happy is considered as a free morpheme since it is a "word" in its own right.
- Bound morphemes (prefixes and suffixes) require a free morpheme to which it can be attached to, and can therefore not appear as a "word" on their own.

(3) Lexical Analysis

- It involves identifying and analysing the structure of words.
- Lexicon of a language means the collection of words and phrases in a language.
- Lexical analysis is dividing the whole chunk of text into paragraphs, sentences, and words.
- In order to deal with lexical analysis, we often need to perform Lexicon Normalization.
- The most common lexicon normalization practices are Stemming and Lemmatization.
  - (i) **Stemming** : Stemming is a rudimentary rule-based process of stripping the suffixes ("ing", "ly", "es", "s" etc) from a word.
  - (ii) **Lemmatization** : Lemmatization, on the other hand, is an organized and step by step procedure of obtaining the root form of the word, it makes use of vocabulary (dictionary importance of words) and morphological analysis (word structure and grammar relations).

(4) Syntactic Analysis

- Deals with analysing the words of a sentence so as to uncover the grammatical structure of the sentence.
- **Example** : "Colourless green idea." This would be rejected by the Semantic analysis as colourless here; green doesn't make any sense.
- Syntactical parsing involves the analysis of words in the sentence for grammar and their arrangement in a manner that shows the relationships among the words.
- Dependency Grammar and Part of Speech tags are the important attributes of text syntactic.

(5) Semantic Analysis

- Determines the possible meanings of a sentence by focusing on the interactions among word-level meanings in the sentence.
- Some people may think it's the level which determines the meaning, but actually all the level do.
- The semantic analyzer disregards sentence such as "hot ice-cream".

(6) Discourse Integration

- Focuses on the properties of the text as a whole that convey meaning by making connections between component sentences.
- It means a sense of the context. The meaning of any single sentence which depends upon those sentences. It also considers the meaning of the following sentence.
- For example, the word "that" in the sentence "He wanted that" depends upon the prior discourse context.

(7) Pragmatic Analysis

- Explains how extra meaning is read into texts without actually being encoded in them.
- This requires much world knowledge, including the understanding of intentions, plans, and goals.
- Consider the following two sentences:
  (i) The city police refused the demonstrators a permit because they feared violence.
  (ii) The city police refused the demonstrators a permit because they advocated revolution.
- The meaning of "they" in the two sentences is different.
- In order to figure out the difference, world knowledge in knowledge bases and inference modules should be utilized.
- Pragmatic analysis helps users to discover this intended effect by applying a set of rules that characterize cooperative dialogues.
- Example: "close the window?" should be interpreted as a request instead of an order.

2.4 REPRESENTING KNOWLEDGE IN TAXONOMIES AND ONTOLOGIES

- In Artificial Intelligence, knowledge representation studies the formalization of knowledge and its processing within machines.
- Techniques of automated reasoning allow a computer system to draw conclusions from knowledge represented in a machine-interpretable form.
- Knowledge may be represented in a number of ways. It might be as basic as a wall chart or as complex as a whole lexicon of terminology used in a discipline, with representations and meanings.
- Taxonomies and ontologies are discussed in this section.

2.4.1 Taxonomies

- Taxonomies are basic classification systems that enable us to describe concepts and their dependencies, typically in a hierarchical fashion.
- Taxonomies provide machines ordered representations.
- According to Bowles, a Taxonomy represents the formal structure of classes or types of objects within a domain.

Taxonomies :

(1) Follow a hierarchic format and provides names for each object in relation to other objects.
(2) May also capture the membership properties of each object in relation to other objects.
(3) Have specific rules used to classify or categorize any object in a domain. These rules must be complete, consistent and unambiguous
(4) Apply rigor in specification, ensuring any newly discovered object must fit into one and only one category or object
(5) Inherits all the properties of the class above it, but can also have additional properties.

- Fig. 2.4.1 shows the example of a Taxonomy.

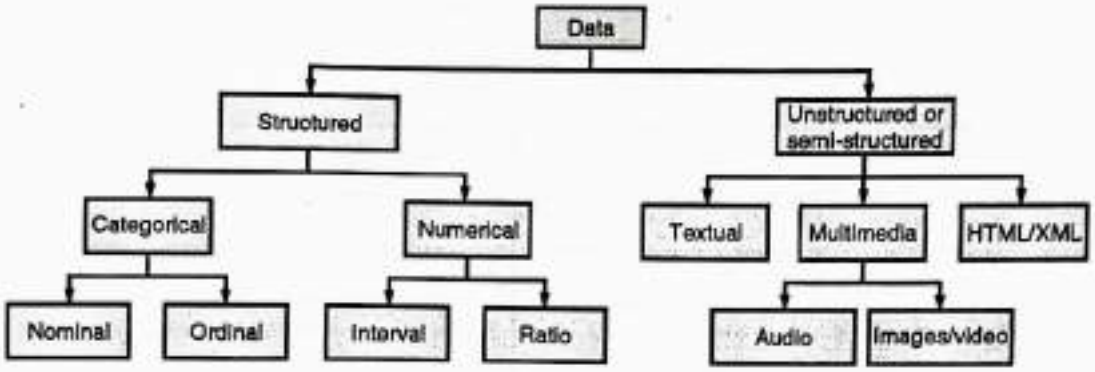


Fig. 2.4.1 : Taxonomy of Data

- The reference taxonomy in a cognitive computing system might be represented as objects in an object-oriented programming language or as conventional data structures like tables and trees.
- These taxonomies are made up of rules and structures that are unlikely to change in the near future.
- Resource Description Framework (RDF) and RDF Schema (RDFS) are used to represent a taxonomy.

2.4.2 Ontologies

- An ontology is a vocabulary used to describe some domain. This includes describing the entities in the domain, and their relationships.
- Ontology is a subset of Taxonomy, but with more information about the behavior of the entities and the relationships between them.
- An ontology is a model that organizes structured and unstructured information through entities, their properties, and the way they relate to one another.
- This model can be seen as a semantic structure that ties together the organization's information and data to provide enhanced content management through :
  (1) Improved findability and discoverability of information;
  (2) Reuse of unknown and forgotten organizational information; and
- Improved SEO on external search engines.
- The anatomy of an ontology consists of:
  (1) **Class** : It is the type of entity. Example: People, Places, Companies, etc.
  (2) **Relationship** : It is the link between objects in the ontology. Example: Albinworks_for TIAA Company.
  (3) **Attribute** : It is the data associated with an object in the ontology. Example: TIAA Company has_PhoneNumber571-403-1109.
- The components of an ontology allow us to thoroughly define a domain of knowledge and business context through the entities defined in classes, relationships between classes, and data associated with classes.
- Fig. 2.4.2 below shows a graphical relationship of a simple ontology that shows how two classes (Person and Company) are related and an attribute (phone number) associated with the company class:

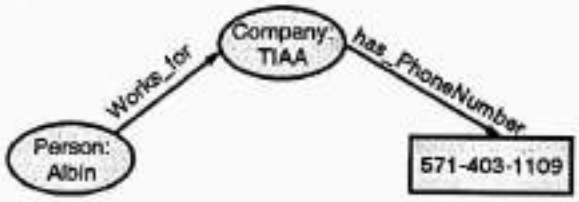


Fig. 2.4.2 : Ontology Example

- List Processing (LISP) and Web Ontology Language (OWL) are the programming languages used for ontology creation.

2.4.3 Comparison of Taxonomy and Ontology

- Table 2.4.1 below compares the two knowledge representation techniques, namely, taxonomy and ontology.

Table 2.4.1 : Comparison of Taxonomy and Ontology

Sr. No.	Taxonomy	Ontology
1.	A taxonomy formalizes the hierarchical relationships among concepts and specifies the term to be used to refer to each.	An ontology identifies and distinguishes concepts and their relationships based on a domain.
2.	A taxonomy is static.	An ontology is dynamic and domain-centric.
3.	It prescribes structure and terminology.	It describes content and relationships in the context of a specific domain.
4.	Taxonomy takes into consideration one type of relationship.	Ontology takes into account many different complex relationships between the concepts.
5.	Add structure/context to unstructured information to make that information more easily searchable.	Create a more sophisticated information model that might be deployed to do advanced natural language processing or text analytics.
6.	Taxonomies tend to relate classes using only is_a, e.g. kidney is_a organ is_a anatomical structure.	Ontologies use a more expressive set of pre-defined relationships such as part_of, develops_from.
7.	Resource Description Framework (RDF) and RDF Schema (RDFS) are used to represent a taxonomy.	LISt Processing (LISP) and Web Ontology Language (OWL) are the programming languages used for ontology creation.
8.	Taxonomy is like a tree with branches.	Ontology is like a web, with everything interconnected

2.4.4 Other Methods of Knowledge Representation

There are other approaches to knowledge representation, in addition to ontologies. Two examples are described here.

(1) Simple Trees

- A simple tree is a logical data structure for representing parent-child relationships.
- A simple tree, expressed as a table with (element, parent) fields in every row, is an effective approach to represent information in a model where the connections are rigid and codified.
- In data analytic tools and catalogues, simple trees are often employed. A retailer's catalogue, for example, can have 30 or 40 product categories. Each category would have a set of elements that belong to it.

(1) The Semantic Web

- The Semantic Web is a mesh of data that are associated in such a way that they can easily be processed by machines instead of human operators.
- The term Semantic Web was coined by Tim Berners-Lee.
- The Semantic Web is driven by the World Wide Web Consortium (W3C).
- The technology stack that supports the Semantic Web is designed to enable computers, software systems, and people to work together in a network. It consists of a wide array of technologies, the most important of which are: **RDF, SPARQL and OWL.**
- The inclusion of data to RDF files enables computer programs or Web spiders to search, discover, collect, assess and process the data on the Web.
- The key goal of the Semantic Web is to trigger the evolution of the existing Web to enable users to search, discover, share and join information with less effort.

- The Semantic Web leads to smarter, more effortless customer experiences by giving content the ability to understand and present itself in the most useful forms matched to a customer's need.
- Humans can use the Semantic Web to execute multiple tasks, such as booking online tickets, searching for different information, using online dictionaries, etc.
- The Semantic Web is a process that allows machines to quickly understand and react to complicated human requests subject to their meaning.
- This kind of understanding mandates that the appropriate information sources are semantically structured, which is a difficult task.
- Semantic Web content structures form an essential basis for a reliable graph, or map of knowledge, necessary for true artificial intelligence (AI) beyond basic Natural Language Processing (NLP) and Natural Language Understanding (NLU).
- Using Semantic Web Technologies, publishers can :
  (1) Build smart digital content infrastructures
  (2) Connect content silos across a huge organization
  (3) Leverage metadata to provide richer experiences
  (4) Curate and reuse content more efficiently
  (5) Connect internal and external content sets
  (6) Build towards real augmented and artificial intelligence
  (7) Power-up authoring experiences and workflow processes

2.5 APPLYING ADVANCED ANALYTICS TO COGNITIVE COMPUTING

- Advanced analytics refers to a collection of techniques and algorithms to identify patterns in large, complex, or high-velocity data sets with different levels of structures.
- Complex statistical models, predictive analytics, machine learning, neural networks, text analytics, and other advanced data mining techniques are all part of advanced analytics.
- Some of the statistical techniques used in advanced analytics include linear regression analysis, logistic regression analysis, decision tree analysis, social network analysis, and time series analysis.
- These analytical techniques aid in the discovery of patterns and anomalies in massive datasets that may be used to forecast and anticipate business outcomes.

2.5.1 Analytics Maturity Levels

1. Descriptive Analytics

- Descriptive analytics answers the question of *what happened*.
- Descriptive analytics looks at data statistically to tell you what happened in the past.
- Descriptive analytics helps a business understand how it is performing by providing context to help stakeholders interpret information.
- This can be in the form of data visualizations like graphs, charts, reports, and dashboards.
- It helps companies understand things such as:
  (1) How much did we sell as a company?
  (2) What was our overall productivity?
  (3) How many customers churned in the last quarter?

2. Diagnostic Analytics

- At this stage, historical data can be measured against other data to answer the question of *why something happened*.
- Often, diagnostic analysis is referred to as root cause analysis.
- This includes using processes such as data discovery, data mining, and drill down and drill through.
- This type of analytics helps companies answer questions such as:
  (a) Why did our company sales decrease in the previous quarter?
  (b) Why are we seeing an increase in customer churn?
  (c) Why is a specific basket of products vastly outperforming their prior year sales figures?

3. Predictive Analytics

- Predictive analytics tells *what is likely to happen*.
- It uses the findings of descriptive and diagnostic analytics to detect clusters and exceptions, and to predict future trends, which makes it a valuable tool for forecasting.

- Predictive analytics belongs to advanced analytics types and brings many advantages like sophisticated analysis based on machine or deep learning and proactive approach that predictions enable.
- Historical data that comprises the bulk of descriptive and diagnostic analytics is used as the basis of building predictive analytics models.
- Predictive analytics helps companies address use cases such as:
  (a) Predicting maintenance issues and part breakdown in machines.
  (b) Determining credit risk and identifying potential fraud.
  (c) Predict and avoid customer churn by identifying signs of customer dissatisfaction.

4. Prescriptive Analytics

- The purpose of prescriptive analytics is to literally prescribe ***what action to take*** to eliminate a future problem or take full advantage of a promising trend.
- Prescriptive analytics uses advanced tools and technologies, like machine learning, business rules and algorithms, which makes it sophisticated to implement and manage.
- Besides, this state-of-the-art type of data analytics requires not only historical internal data but also external information due to the nature of algorithms it's based on.
- Prescriptive analytics requires strong competencies in descriptive, diagnostic, and predictive analytics which is why it tends to be found in highly specialized industries (oil and gas, clinical healthcare, finance, and insurance to name a few) where use cases are well defined.
- Prescriptive analytics help to address use cases such as:
  (a) Automatic adjustment of product pricing based on anticipated customer demand and external factors.
  (b) Flagging select employees for additional training based on incident reports in the field.

5. Cognitive Analytics

- Cognitive analytics brings together a number of intelligent technologies to accomplish bringing together all of the above analytics and data and the software learns by itself without us telling it what to do, including semantics, artificial intelligence algorithms and a number of learning techniques such as deep learning and machine learning.
- Applying such techniques, a cognitive application can get smarter and self-heal and become more effective over time by learning from its interactions with data and with humans.
- Cognitive analytics helps companies address use cases such as :
  (a) How secure is the city environment?
  (b) Which combination of drugs will provide the best outcome for the cancer patient based on the specific characteristics of the tumor and genetic sequencing?

2.5.2 Key Capabilities in Advanced Analytics

- It's impossible to create a cognitive system without combining predictive analytics, text analytics, and machine learning.
- Data scientists can identify and comprehend the significance of patterns and anomalies in massive amounts of structured and unstructured data by applying modern analytics components.
- These patterns are used to create models and algorithms that assist decision-makers in determining the best course of action.
- The analytics process aids in comprehending the relationships between data elements as well as the data's context.
- Machine learning is used to increase the model's accuracy and create more accurate predictions. It's an essential tool for advanced analytics, especially when it comes to analyzing massive data sources that are largely unstructured.

2.5.3 The Relationship Between Statistics, Data Mining and Machine Learning

- Advanced analytics includes statistics, data mining, and machine learning.
- Understanding data, characterizing the features of a data set, detecting relationships and patterns in that data, constructing a model, and generating predictions are all responsibilities of these disciplines.
- When it comes to solving business challenges, there is a lot of overlap in how the various strategies and instruments are used.
- Many popular data mining and machine learning methods have their roots in traditional statistical analysis.

2.5.3.1 Statistics

- Statistics is the base of all Data Mining and Machine learning algorithms.
- Statistics is the study of collecting, analyzing and studying data and come up with inferences and prediction about future.
- Major task of a statistician is to estimate population from sample metrics. Statistics also deal with designing surveys and experiments in order to get quality data which can further be used to make estimation about the population.
- Statistics is used to summarize numbers for example finding out descriptive statistics like Mean, Median, Mode, Standard Deviation, Variance, Percentiles, Testing hypotheses etc.

2.5.3.2 Data Mining

- Data mining is a field where we try to identify patterns in data and come up with initial insights.
- Example: You got the data and you identified missing values, then you saw that missing values are mostly coming from recordings taken manually.
- Data mining comes into play after data is collected.
- Data mining uses power of machine learning, statistics and database techniques to mine large databases and come up with patterns.
- Mostly data mining uses cluster analysis, anomaly detection, association rule mining etc. to find out patterns in data.
- In short, Data Mining is finding out hidden and interesting patterns stored in large data warehouses using the power of statistics, artificial intelligence, machine learning and database management techniques.

2.5.3.3 Machine Learning

- Machine learning is a part of data science which majorly focuses on writing algorithms in a way such that machines (Computers) are able to learn on their own and use the learnings to tell about new dataset whenever it comes in.
- Machine learning uses power of statistics and learns from the training dataset.
- For example, we use regressions, classifications etc. to learn from training data and use those learnings to estimate test dataset.

2.5.3.4 Data Mining vs. Statistics - Similarities and Differences

The objective of Data Mining and Statistics is to perform data analysis but both are different tools. Data mining process involve modelling, predicting and optimizing a dataset while Statistics describes how efficient a dataset is –more or less.

Table 2.5.1 : Data Mining Vs Statistics

Sr. No.	Data Mining	Statistics
1.	Explorative – Dig out the data first, discover novel patterns and then make theories.	Confirmative – Provide theory first and then test it using various statistical tools.
2.	Involves Data Cleaning	Statistical methods applied on Clean Data
3.	Usually involves working with large datasets.	Usually involves working with small datasets.
4.	Makes generous use of heuristics thinking	There is no scope for heuristics thinking.
5.	Inductive process	Deductive (Does not involve making any predictions)

Sr. No.	Data Mining	Statistics
6.	Numeric and Non-Numeric Data	Numeric Data
7.	Less concerned about data collection.	More concerned about data collection.
8.	Some of the popular data mining methods include – Estimation, Classification, Neural Networks, Clustering, Association, and Visualization.	Some of the popular statistical methods include –Inferential and Descriptive Statistics.

2.5.3.5 Machine Learning vs. Statistics

Table 2.5.2 : Machine Learning Vs Statistics

Sr. No.	Machine Learning	Statistics
1	Machine learning focuses on optimization and performance.	Statistics is more concerned about the inference that can be drawn from the model
2	In machine learning models, the machine learning algorithms are directly run on the model making the data speak instead of guiding it in a specific direction with initial hypothesis.	Statistical learning involves forming a hypothesis (making assumptions that are validated before building models) before building a model.
3	Machine learning is all about predictions, supervised and unsupervised learning.	Statistics is all about sample, population, and hypothesis.
4	Machine Learning is about building algorithms that help machines emulate human learning	Statistics is about converting the data into aggregate numbers which help understand the structure in data.

2.3.5.6 Data Mining vs Machine Learning

- Data mining and machine learning both come under the common umbrella of Data Science, since they both involve processing and analysis of large amounts of data.
- Both the techniques are used to solve complex real-world problems. Machine learning can be used as a means of conducting data mining and the data gathered from data mining can be used to train models to apply machine learning techniques.

Table 2.5.3 : Data Mining Vs Machine Learning

Sr. No.	Data Mining	Machine learning
1	Data mining involves extraction of information from large amounts of unstructured data.	Machine learning is about using algorithms to build a model and train it so that new information can be introduced based on data from previous occurrences.
2	In data mining, rules are obtained from the data available.	In machine learning, the algorithm used teaches the computer to learn and comprehend the rules.
3	Data mining requires human intervention and is created so that the data can be further processed by people.	The idea of machine learning is to teach itself so that there is no dependence on human influence. Human interference in the case of machine learning is mostly limited to setting up the initial algorithms.
4	In the case of data mining, there is no concept of the system adapting. Data mining is as smart as the users who specify the parameters.	The entire goal of Machine learning is to teach itself to adapt based on the algorithms and new data inputs.
5	Data mining is all about working on large amounts of raw data to make forecasts for the business.	Machine learning is about applying algorithms to structured data.

2.5.4 Using Machine Learning in the Analytics Process

- Machine learning is a method of data analysis that automates analytical model building.
- It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention.
- Machine learning is the science of designing algorithms that learn on their own from data and adapt without human correction.
- As we feed data to these algorithms, they build their own logic and, as a result, create solutions relevant to aspects of our world as diverse as fraud detection, web searches, tumor classification, and price prediction.
- In deep learning, a subset of machine learning, programs discover intricate concepts by building them out of simpler ones. These algorithms work by exposing multilayered (hence "deep") neural networks to vast amounts of data.
- Applications for machine learning, such as natural language processing, dramatically improve performance through the use of deep learning.
- Machine learning constitutes model-building automation for data analysis.
- When we assign machines tasks like classification, clustering, and anomaly detection i.e. tasks at the core of data analysis, we are employing machine learning.
- We can design self-improving learning algorithms that take data as input and offer statistical inferences.
- Without relying on hard-coded programming, the algorithms make decisions whenever they detect a change in pattern.
- In the real world, the dataset present will never be clean and perfect. It means each dataset contains impurities, noisy data, outliers, missing data, or imbalanced data.
- Due to these impurities, different problems occur that affect the accuracy and the performance of the model. One of such problems is Overfitting in Machine Learning. *Overfitting is a problem that a model can exhibit.*

2.5.4.1 Supervised Learning

- Supervised learning is the types of machine learning in which machines are trained using well "labelled" training data, and on basis of that data, machines predict the output.
- The labelled data means some input data is already tagged with the correct output.
- In supervised learning, the training data provided to the machines work as the supervisor that teaches the machines to predict the output correctly.
- It applies the same concept as a student learns in the supervision of the teacher.
- Supervised learning is a process of providing input data as well as correct output data to the machine learning model.
- The aim of a supervised learning algorithm is to find a mapping function to map the input variable(x) with the output variable(y).
- In the real-world, supervised learning can be used for Risk Assessment, Image classification, Fraud Detection, spam filtering, etc.

Algorithms included in Supervised Learning are as follows :

(1) Linear Regression
(2) Logistic Regression
(3) Decision Tree
(4) Random Forest
(5) Gradient-boosted Decision Tree
(6) Neural Networks
(7) Support Vector Machine (SVM)
(8) k-Nearest Neighbor (k-NN)

(1) Linear Regression

- In this process, a relationship is established between independent and dependent variables by fitting them to a line.
- This line is known as the regression line and represented by a linear equation $Y = a*X + b$. **LASSO** is a type of linear regression that minimize the sum of squared errors.
- It stands for Least Absolute and Selection Operator. It is also called as L1 regularization. **Ridge regression** is one of the types of linear regression in which a small amount of bias is introduced so that we can get better long-term predictions.

- Ridge regression is mostly used to reduce the overfitting in the model, and it includes all the features present in the model.
- It reduces the complexity of the model by shrinking the coefficients. It is also called as L2 regularization.

(2) Logistic Regression

- Logistic regression is used to estimate discrete values (usually binary values like 0 or 1) from a set of independent variables.
- It helps predict the probability of an event by fitting data to a logit function.

(3) Decision Tree

- This is a supervised learning algorithm that is used for classifying problems.
- In this algorithm, we split the population into two or more homogenous sets based on the most significant attributes/ independent variables.

(4) Random Forest

- A collection of decision trees is called a Random Forest.
- To classify a new object based its attributes, each tree is classified, and the tree "votes" for that class.
- The forest chooses the classification having the most votes.

(5) Gradient-boosted Decision Tree

- Gradient-boosted decision trees are a popular method for solving prediction problems in both classification and regression domains.
- The approach improves the learning process by simplifying the objective and reducing the number of iterations to get to a sufficiently optimal solution.
- Each iteration of the decision tree involves adjusting the values of the coefficients, weights, or biases applied to each of the input variables being used to predict the target value, with the goal of minimizing the loss function (the measure of difference between the predicted and actual target values).
- The gradient is the incremental adjustment made in each step of the process; boosting is a method of accelerating the improvement in predictive accuracy to a sufficiently optimum value.

(6) Neural Networks

- Neural networks, also known as artificial neural networks (ANNs) or simulated neural networks (SNNs), are a subset of machine learning and are at the heart of deep learning algorithms.
- Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another.
- Artificial neural networks (ANNs) are comprised of a node layers, containing an input layer, one or more hidden layers, and an output layer.
- Each node, or artificial neuron, connects to another and has an associated weight and threshold.
- If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network.
- Otherwise, no data is passed along to the next layer of the network.

(7) Support Vector Machine (SVM)

- Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems.
- However, primarily, it is used for Classification problems in Machine Learning.
- The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future.
- This best decision boundary is called a hyperplane. SVM chooses the extreme points/vectors that help in creating the hyperplane.
- These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.

(8) k-Nearest Neighbor (k-NN)

- This algorithm can be applied to both classification and regression problems.
- It stores all available cases and classifies any new cases by taking a majority vote of its k neighbors.
- The case is then assigned to the class with which it has the most in common.

2.5.4.2 Unsupervised Learning

- As the name suggests, unsupervised learning is a machine learning technique in which models are not supervised using training dataset. Instead, models itself find the hidden patterns and insights from the given data.
- It can be compared to learning which takes place in the human brain while learning new things.
- Unsupervised learning is a type of machine learning in which models are trained using unlabeled dataset and are allowed to act on that data without any supervision.
- Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data.
- The goal of unsupervised learning is to **find the underlying structure of dataset, group that data according to similarities, and represent that dataset in a compressed format**.

Example

- Suppose the unsupervised learning algorithm is given an input dataset containing images of different types of cats and dogs.
- The algorithm is never trained upon the given dataset, which means it does not have any idea about the features of the dataset.
- The task of the unsupervised learning algorithm is to identify the image features on their own.
- Unsupervised learning algorithm will perform this task by clustering the image dataset into the groups according to similarities between images.

Algorithms included in Unsupervised Learning are as follows :

(1) Clustering
(2) Principal Component Analysis (PCA)
(3) Singular Value Decomposition (SVD)
(4) Kernel Density Elimination (KDE)
(5) Nonnegative Matrix Factorization (NMF)
(6) Self-Organizing Map (SOM)

(1) Clustering

- Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups.
- It is basically a collection of objects on the basis of similarity and dissimilarity between them. **K-means clustering** is the simplest unsupervised learning algorithm that solves clustering problem.
- K-means algorithm partitions n observations into k clusters where each observation belongs to the cluster with the nearest mean serving as a prototype of the cluster.
- K-means clustering is a simple local optimization algorithm.
- The **EM algorithm** finds maximum likelihood estimates of parameters in probabilistic models. EM is an iterative method which alternates between two steps, expectation (E) and maximization (M).
- For clustering, EM makes use of the finite Gaussian mixtures model and estimates a set of parameters iteratively until a desired convergence value is achieved.
- The mixture is defined as a set of K probability distributions and each distribution corresponds to one cluster.
- An instance is assigned with a membership probability for each cluster.

(2) Principal Component Analysis (PCA)

- Principal Component Analysis is an unsupervised learning algorithm that is used for the dimensionality reduction in machine learning.
- It is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation.
- These new transformed features are called the Principal Components. It is one of the popular tools that is used for exploratory data analysis and predictive modeling.
- It is a technique to draw strong patterns from the given dataset by reducing the variances.
- PCA generally tries to find the lower-dimensional surface to project the high-dimensional data.
- PCA works by considering the variance of each attribute because the high attribute shows the good split between the classes, and hence it reduces the dimensionality.
- Some real-world applications of PCA are image processing, movie recommendation system, optimizing the power allocation in various communication channels.

- It is a feature extraction technique, so it contains the important variables and drops the least important variable.

(3) Singular Value Decomposition (SVD)

- Singular value decomposition (SVD) is a method of representing a matrix as a series of linear approximations that expose the underlying meaning-structure of the matrix.
- The goal of SVD is to find the optimal set of factors that best predict the outcome.
- In more technical terms, SVD is closely related to principal components analysis in that it reduces the overall dimensionality of the input matrix (number of input documents by number of extracted terms) to a lower dimensional space (a matrix of much smaller size with fewer variables), where each consecutive dimension represents the largest degree of variability (between terms and documents) possible. SVD is often used in recommendation engines.

(4) Kernel Density Elimination (KDE)

- Kernel density estimation extrapolates data to an estimated population probability density function.
- It's called kernel density estimation because each data point is replaced with a kernel i.e. a weighting function to estimate the pdf.
- The function spreads the influence of any point around a narrow region surrounding the point.
- The resulting probability density function is a summation of every kernel. KDE is used in analytics for risk management and financial modeling.

(5) Nonnegative Matrix Factorization (NMF)

- Nonnegative matrix factorization (NMF) is a family of methods widely used for information retrieval across domains including text, images, and audio.
- Within music processing, NMF has been used for tasks such as transcription, source separation, and structure analysis.
- It is a technique for obtaining low rank representation of matrices with non-negative or positive elements. NMF is useful in pattern recognition, gene expression analysis and social network analysis.

(6) Self-Organizing Map (SOM)

- A Self-organizing Map is a data visualization technique developed by Professor Teuvo Kohonen in the early 1980's.
- SOMs map multidimensional data onto lower dimensional subspaces where geometric relationships between points indicate their similarity.
- The reduction in dimensionality that SOMs provide allows people to visualize and interpret what would otherwise be, for all intents and purposes, indecipherable data.
- SOMs generate subspaces with an unsupervised learning neural network trained with a competitive learning algorithm.
- Neuron weights are adjusted based on their proximity to "winning" neurons (i.e. neurons that most closely resemble a sample input).
- Training over several iterations of input data sets results in similar neurons grouping together and vice versa.
- SOMs are widely used in manufacturing processes.

2.5.5 Predictive Analytics

- Predictive analytics is a branch of advanced analytics that makes predictions about future outcomes using historical data combined with statistical modeling, data mining techniques and machine learning.
- Companies employ predictive analytics to find patterns in this data to identify risks and opportunities.
- Predictive analytics is often associated with big data and data science.
- Companies are generating huge volume of data residing across transactional databases, equipment log files, images, video, sensors or other data sources.
- To gain insights from this data, data scientists use deep learning and machine learning algorithms to find patterns and make predictions about future events.
- These include linear and nonlinear regression, neural networks, support vector machines and decision trees.
- Learnings obtained through predictive analytics can then be used further within prescriptive analytics to drive actions based on predictive insights.

- Predictive analytics can be applied to structured, unstructured, and semi-structured data.
- In predictive analytics, the algorithm uses some sort of objective function.
- Predictive models are applied to business activities to better understand customers, with the goal of predicting buying patterns, potential risks, and likely opportunities.

2.5.5.1 Predictive Analytics Tools

(1) **Data Science Platform :** IBM Watson® Studio helps operationalize AI by providing the tools to prepare data and build models anywhere using open source code or visual modelling.

(2) **Statistical Analysis Software :** IBM® SPSS® Statistics is designed to solve business and research problems using ad hoc analysis, hypothesis testing, geospatial analysis and predictive analytics.

(3) **Visual Modeling Tool :** The IBM SPSS Modeler solution can help you tap into data assets and modern applications, with complete algorithms and models that are ready for immediate use.

(4) **Decision Optimization Solutions :** IBM Decision Optimization optimizes outcomes by offering prescriptive analytics capabilities to augment predictive insights from machine learning models.

2.5.5.2 Predictive Analytics Use Cases

(1) **Banking :** Financial services use machine learning and quantitative tools to predict credit risk and detect fraud.

(2) **Healthcare :** Predictive analytics in health care is used to detect and manage the care of chronically ill patients.

(3) **Human resources (HR) :** HR teams use predictive analytics to identify and hire employees, determine labor markets and predict an employee's performance level.

(4) **Marketing and sales :** Predictive analytics can be used for marketing campaigns throughout the customer lifecycle and in cross-sell strategies.

(5) **Retail :** Retailers use predictive analytics to identify product recommendations, forecast sales, analyze markets and manage seasonal inventory.

(6) **Supply chain :** Businesses use predictive analytics to make inventory management more efficient, helping to meet demand while minimizing stock.

2.5.6 Text Analytics

- Text analytics is the process of transforming unstructured text documents into usable, structured data.
- Text analysis works by breaking apart sentences and phrases into their components, and then evaluating each part's role and meaning using complex software rules and machine learning algorithms.
- Text analytics forms the foundation of numerous natural language processing (NLP) features, including named entity recognition, categorization, and sentiment analysis. In broad terms, these NLP features aim to answer four questions:
  (1) Who is talking?
  (2) What are they talking about?
  (3) What are they saying about those subjects?
  (4) How do they feel?
- Data analysts and other professionals use text mining tools to derive useful information and context-rich insights from large volumes of raw text, such as social media comments, online reviews, and news articles.
- In this way, text analytics software forms the backbone of business intelligence programs, including voice of customer/customer experience management, social listening and media monitoring, and voice of employee/workforce analytics.
- Text mining, text analysis, and text analytics are often used interchangeably, with the end goal of analyzing unstructured text to obtain insights.
- However, while text mining (or text analysis) provides insights of a qualitative nature, text analytics aggregates these results and turns them into something that can be quantified and visualized through charts and reports.
- Text analysis and text analytics often work together to provide a complete understanding of all kinds of text, like emails, social media posts, surveys, customer support tickets, and more.

- For example, you can use text analysis tools to find out how people feel toward a brand on social media (sentiment analysis), or understand the main topics in product reviews (topic detection).
- Text analytics, on the other hand, leverages the results of text analysis to identify patterns, such as a spike in negative feedback, and provides you with actionable insights you can use to make improvements, like fixing a bug that's frustrating your users.

2.5.6.1 Working of Text Analytics

- Text analytics starts by breaking down each sentence and phrase into its basic parts.
- Each of these components, including parts of speech, tokens, and chunks, serve a vital role in accomplishing deeper natural language processing and contextual analysis.

There are seven computational steps involved in preparing an unstructured text document for deeper analysis :

(1) Language Identification

- The first step in text analytics is identifying what language the text is written in. Spanish? Russian? Arabic? Chinese?
- Together, these languages include a complex tangle of alphabets, abjads and logographies.
- Each language has its own idiosyncrasies and unique rules of grammar. So, as basic as it might seem, language identification determines the whole process for every other text analytics function.

(2) Tokenization

- Tokenization is the process of breaking apart a sentence or phrase into its component pieces. Tokens are usually words or numbers.
- Depending on the type of unstructured text you're processing, however, tokens can also be:
  (i) Punctuation (exclamation points amplify sentiment)
  (ii) Hyperlinks (https://...)
  (iii) Possessive markers (apostrophes)
- Tokenization is language-specific, so it's important to know which language you're analyzing.
- Most alphabetic languages use whitespace and punctuation to denote tokens within a phrase or sentence.
- Logographic (character-based) languages such as Chinese, however, use other systems.

(3) Sentence breaking

- Small text documents, such as tweets, usually contain a single sentence. But longer documents require sentence breaking to separate each unique statement.
- In some documents, each sentence is separated by a punctuation mark.
- But some sentences contain punctuation marks that don't mean the end of the statement (like the period in "Dr.")

(4) Part of Speech tagging

- Part of Speech tagging (or PoS tagging) is the process of determining the part of speech of every token in a document, and then tagging it as such.
- Most languages follow some basic rules and patterns that can be written into a basic Part of Speech tagger.
- When shown a text document, the tagger figures out whether a given token represents a proper noun or a common noun, or if it's a verb, an adjective, or something else entirely.

(5) Chunking

- Chunking refers to a range of sentence-breaking systems that splinter a sentence into its component phrases (noun phrases, verb phrases, and so on).
- Chunking in text analytics is different than Part of Speech tagging. PoS tagging means assigning parts of speech to tokens while Chunking means assigning PoS-tagged tokens to phrases.
- For example, take the sentence : The tall man is going to quickly walk under the ladder. PoS tagging will identify man and ladder as nouns and walk as a verb.
- Chunking will return: [the tall man]_np [is going to quickly walk]_vp [under the ladder]_pp where np stands for "noun phrase," vp stands for "verb phrase," and pp stands for "prepositional phrase."

(6) Syntax parsing

- Syntax parsing is the analysis of how a sentence is formed.
- Syntax parsing is a critical preparatory step in sentiment analysis and other natural language processing features.
- The same sentence can have multiple meanings depending on how it's structured:

  Apple was doing poorly until *Steve Jobs*...

  Because **Apple** was doing poorly, *Steve Jobs*...

  Apple was doing poorly because *Steve Jobs*...
- In the first sentence, **Apple** is negative, whereas *Steve Jobs* is positive.
- In the second, **Apple** is still negative, but *Steve Jobs* is now neutral.
- In the final example, both **Apple** and *Steve Jobs* are negative.
- Syntax parsing is one of the most computationally-intensive steps in text analytics.

(7) Sentence chaining

- The final step in preparing unstructured text for deeper analysis is sentence chaining.
- Sentence chaining uses a technique called lexical chaining to connect individual sentences based on their association to a larger topic. Take the sentences :

  I like beer.

  Miller just launched a new pilsner.

  But I only drink Belgian ale.

  Even if these sentences don't appear near each other in a body of text, they are still connected to each other through the topics of beer → pilsner → ale.

  Lexical chaining allows us to make these kinds of connections. The "score" of a lexical chain is directly related to the length of the chain and the relationships between the chaining nouns (same words, antonyms, synonyms, homonyms, meronyms, hypernyms or holonyms).Lexical chains flow through the document and help a machine detect over-arching topics and quantify the overall "feel".

2.5.6.2 Text Analytics Use Cases

(1) **Manufacturing** : Identify root causes of product issue quicker, Identify trends in market segments, Understand competitor's products.

(2) **Government** : Identify fraud, Understand public sentiments about unmet needs, Find emerging concerns that can shape policy.

(3) **Financial Institutions** : Use contact center transcriptions, Understand customers, Identify money laundering or other fraudulent situation.

(4) **Retail** : Identify profitable customers and understand the reasons for their loyalty, Manage the brand on social media.

(5) **Legal** : Identify topics and keywords in discovery documents, Find patterns in defendant's communication.

(6) **Healthcare** : Find similar patterns in doctor's reports, Use social media to detect outbreaks earlier, Identify patterns in patient claims data.

2.5.7 Image Analytics

- Image analytics is the technique of incorporating various technologies and automatic algorithms used for extracting and processing data from images.
- Image analytics also make use of logical analysis to interpret information from visuals and graphics.
- In simple words, image analytics is the ability of computers to recognize various elements pictured in images. Image analytics is also known as "computer vision" or "image recognition".
- With an explosion of image data, which makes up about 80 percent of all unstructured big data, there is a growing need of analytical systems to interpret images, which is unstructured data to machine readable format. For example, the use of bar codes and QR codes are simple and popular examples of image analytics that we encounter in our day-to-day lives.
- The main purpose of image analytics is to convert the unstructured form of images and videos into a machine analyzable representation of a set of variables or making it analytically prepared data.

In the first step, images are segmented into structured elements and prepped up for feature extraction or simply the identification of low-level features in the image.

Image segmentation is the process of partitioning an image into a collection of connected sets of pixels. Image segmentation helps to identify certain features in the image.

During the second transformation step, the application detects the relationships between the features, variables and time.

The third transformation step is the extraction of variables with time-stamped values. In the image processing application, a variable is represented by a series of values related to an entity (for example, emotion or customer sentiment).

- Each value is time stamped, so that it makes it possible to treat a variable as a time series.
- Essentially, image analytics transforms the image-input, adds value and create a rich set of time series as analytically prepared data output.

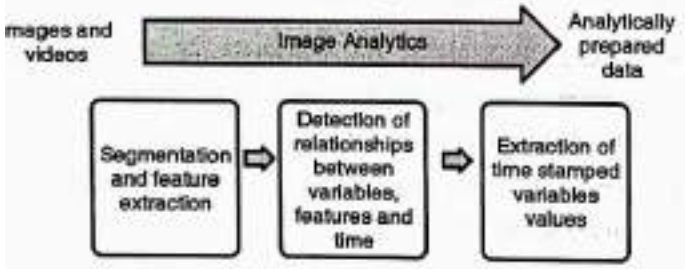


Fig. 2.5.1 : Image Analytics Process

2.5.7.1 Image Analytics Use Cases

(1) **Object Detection and Recognition** : Detect and identify specific object classifications such as people, bikes, packages, buses, and automobiles in images and video feeds in real time.

(2) **Image Classification** : Categorize, tag, and label images into groups based on rules or automatically identified similarities, reducing manual efforts in image analysis.

(3) **Logo Detection and Recognition** : Scour the web, broadcast video, and social media to identify where/when a logo appears to understand brand value and mitigate negative exposure.

(4) **Facial Recognition, Detection and Redaction** : Image and video analytics identify faces within images and compare the detected face to a database of known individuals and redacts other images.

(5) **Optical Character Recognition** : Optical character recognition and detection is used to extract text from image files for PII mapping and data analysis.

2.5.8 Speech Analytics

- Speech analytics is a software technology that transcribes 100% of voice calls and derives deep insights, trends, and metrics from each call.
- It utilizes AI services including transcription, natural language processing, and speech technologies to understand, analyze, and derive insights from a voice conversation.
- These insights are then used to evaluate agent performance, assess customer experience, and monitor organization-wide strengths and shortcomings on every voice interaction.
- Speech analytics is also referred to as audio mining. It is a technology that identifies human speech and text and converts that into data. The data is then structured so actionable insights can be drawn from it.
- In its most basic form, a speech analytics solution consists of a speech engine that converts speech to data, an indexing layer that searches the information, a query engine that lets users query the data, and finally, a reporting application that enables you to present the analytical findings.

2.5.8.1 How does Speech Analytics Work ?

(1) **Data Processing** : It uses a number of AI-services including automatic speech recognition, transcription and tonality-based sentiment analysis to analyze both the audio recording and call metadata.

(2) **Analysis** : Once the call recordings are analyzed, speech analytics then categorizes, keyword spots, redacts (for compliance purposes), and reports its analysis of the call.

(3) **Insights** : The speech analytics platform then delivers detailed reporting on the analysis, including call quality, sentiment, agent performance, and compliance monitoring.

2.5.8.2 Benefits of Speech Analytics

(1) **Significantly increases call coverage** : Historically, QA teams in call centers on average quality check 2-4 voice calls per agent, per month. With speech analytics, organizations can review up to 100% of voice calls.

(2) **Monitor key KPIs** : Speech analytics empowers customer service and support teams to set up analysis on any number of customer interactions moments. This is anything from supervisor escalations and compliance violations, to customer satisfaction and average handle time (AHT).

(3) **Provide near-real-time speech analytics feedback** : With faster analysis and 100% call coverage, supervisors can deliver tailored feedback almost immediately to agents.

(4) **Uncover hidden inefficiencies** : By monitoring a variety of contact center KPIs, leadership can better understand what's impacting those KPIs and unearth inefficiencies causing them.

(5) **Personalized training** : With deep insights on 100% of customer calls per agent, supervisors and L&D teams can create custom tailored coaching sessions for individual agents.

(6) **Improve customer experiences** : With sentiment analysis, teams can look at the things driving positive customer experiences (e.g. empathy statements), and indicators of negative ones (e.g. supervisor escalations), and in turn, reduce customer churn.

2.5.8.3 Speech Analytics Use Cases

(1) Speech analytics in Banking
(2) Speech analytics in Insurance
(3) Speech analytics in Telecom
(4) Speech analytics in BPO

(1) Speech analytics in Banking

- If you asked a banking customer during the height of the COVID pandemic, one of their biggest complaints was not being able to talk to a human agent.
- Most of the banking contact centers were inundated with customer calls leaving them helpless to provide the right customer service.
- Speech analytics in banking can help to change all that. Banking speech analytics solutions can assist human agents, thanks to Natural Language Processing (NLP) capabilities.
- It helps to understand the customers better and even anticipate their needs due to the latest advancements in banking speech analytics software solutions.
- For those banking calls that need human attention, speech analytics solutions can help too. Banking speech analytics can provide human agents with the right in-call assistance, alerts, notifications, and more.
- Thus, it is possible to drive a superior customer experience. Human empathy is the most significant factor in building long-term banking relationships, and banking speech analytics solutions help to do just that.
- Speech analytics banking solutions can also provide hitherto unheard insights into reducing customer call volumes, achieving first contact resolution (FCR), and reducing average handle time (AHT).
- Predictive speech analytics banking software does play a significant role in the banking and financial services sector since there is a constant threat from upstart and nimble fintech players that can poach the legacy banking customers overnight with better customer service.
- AI, automation, and speech analytics can significantly reduce customer churn by predicting customer behavior and guiding the banks in implementing the right corrective actions in advance.

(2) Speech analytics in Insurance

- Insurance is a sector that faces a huge number of challenges.
- Due to the increased competition, higher regulation, and bigger customer expectations, the insurance industry urgently needs help and conversational AI technology can help.
- Customer service is one aspect where the insurance industry could look at speech analytics to bring about a positive change.

- Insurance speech analytics can go a long way in helping meet customer expectations and ensuring insurance players can grow rapidly.
- Speech analytics in insurance lets you monitor and analyze customer conversation data to anticipate customer needs, to be better equipped to improve compliance, reduce fraud, lower costs, and deliver a better customer experience.
- Insurers spend ample time and money to keep customer support up to date. With speech analytics in insurance, it is possible to create structured responses based on past customer interactions. Insurance speech analytics solutions can support insurance contact centers to handle multiple customer-facing processes.
- Insurance speech analytics software can help human agents with the necessary real-time assistance, understanding the real intent of the customers, automating mundane and repetitive tasks, and ensuring that the agents comply with the regulations and disclosure norms set forth by various governing bodies.
- Thus, speech analytics in insurance can render the agents free to do what they do best – empathize with the customers.
- Top insurance players are already implementing conversational AI and speech analytics software to improve throughput, efficiency, and insurance customer service delivery.

(3) Speech analytics in Telecom

- Telecom speech analytics solutions are revolutionizing the telecom sector, which is finding it increasingly difficult to cater to customers' needs of late.
- Telecom speech analytics solutions can help to resolve customer queries quickly and satisfactorily thanks to machine's ability to understand human conversations using ML.
- It is a fact that telecoms face one of the highest churn rates of any industry.
- The chief reason being customers feeling their voice not getting heard by the brand. Speech analytics telecom solutions can help to address the problem.
- Thanks to predictive analytics, it is possible to anticipate customer problems and take corrective actions before it inconveniences the customer.
- Telecom speech analytics solutions are powered by AI and Machine Learning and hence they are context aware, making such solutions highly desirable to telecoms that are facing a deluge of customer calls.
- With the right speech analytics telecom technology, it is possible to leverage pre-built telecom intent models to upgrade telecom customer experiences.
- Delivering easy, personalized conversations your customers will love, is vital to reducing telecom customer churn. Telecom speech analytics solutions help you do just that.
- Knowing if a customer is likely to pay or not, finding the optimal way to approach an individual customer, are all crucial for any telecom player.
- The AI-powered telecom speech analytics solutions can be of immense help in such scenarios too.

(4) Speech analytics in BPO

- The BPO industry is seeing increased deployment of technology tools to help contact center agents understand the customers better.
- BPO speech analytics solution is one of them. Since speech analytics BPO solutions understand human speech, they are better suited to resolve simpler customer queries using AI-based chatbots.
- Speech analytics BPO technology driven by AI can now identify the customer's real intent in real-time and offer suitable alerts to the agent.
- Speech analytics BPO solutions can now monitor 100% of the customer calls, derive actionable insights, raise red flags when there are areas of concern, assist during agent QA process, etc.
- Thus conversational AI-driven solutions are truly disrupting BPO's voice business.
- Training the BPO agents with the domain knowledge, analyzing call disposition across multiple tags and increasing first call resolution can all be realized with speech analytics BPO software.
- The agent performance can be continuously monitored with BPO speech analytics tools, and the right feedback be sent to agent supervisors in real-time.

- It improves the overall call quality and even helps BPOs in adhering to regulatory compliance in the BPO industry.
- BPOs for long have been struggling with manual operations of front-office tasks.
- It led to agent fatigue and inability to focus on customer problems and ultimately leading to lack of empathy towards customers.
- Luckily, speech analytics BPO tools can help to streamline the processes and help BPOs adhere to the business and customer metrics that matter.

2.5.9 Impact of Open Source Tools on Advanced Analytics

- Predictive analytics is a significant analytical approach used by many firms to assess risk, forecast future business trends, and predict when maintenance is required.
- Data scientists use historical data as their source and utilize various regression models and machine learning techniques to detect patterns and trends in the data.
- The basic goal of predictive analytics is to forecast what will happen in the future with a high degree of certainty.
- This distinguishes predictive analytics from descriptive analytics, which assists analysts in analyzing what has previously occurred, and prescriptive analytics, which uses optimization techniques to detect optimal solutions to address the trends revealed by predictive analytics.
- Predictive analytics tools use data to help you predict the future. Instead, it informs you of the probability of various scenarios. Knowing these possibilities might assist you in planning various parts of your business.
- There are a number of tools that are available in the market which can help us with the predictive analytics.
- Some notable open source tools are :

(1) **scikit-learn** : Scikit-learn is an open source machine learning library for the Python programming language.

(2) **KNIME** : KNIME is an open source data analytics, reporting and integration platform that integrates various components for machine learning and data mining.

(3) **OpenNN** : OpenNN is a software library which is able to learn from both datasets and mathematical models. The software can be used to solve pattern recognition problems.

(4) **Orange** : Orange is a component-based data mining and machine learning software suite. It includes a set of components for data pre-processing, modelling, model evaluation, and exploration techniques.

(5) **R** : R is a free software programming language and software environment for statistical computing and graphics.

(6) **Rapid Miner** : Rapid Miner is a software platform that provides an integrated environment for machine learning, data mining, text mining, predictive analytics and business analytics.

(7) **Weka** : Weka is a popular suite of machine learning software written in Java, developed at the University of Waikato, New Zealand. It contains a collection of visualization tools and algorithms for analysis and predictive modelling.

(8) **GNU Octave** : GNU Octave is a high-level programming language, primarily intended for numerical computations.

(9) **Apache Mahout** : Apache Mahout is a project to produce free implementations of distributed or otherwise scalable machine learning algorithms focused primarily in the areas of collaborative filtering, clustering and classification.

2.6 THE PROCESS OF BUILDING A COGNITIVE APPLICATION

Designing a typical cognitive application involves basic seven steps as discussed below :

(1) Defining the objective
(2) Defining the domain
(3) Understanding the intended users and their attributes
(4) Defining questions and exploring insights
(5) Acquiring the relevant data sources
(6) Creating and refining the corpora
(7) Training and Testing

(1) Defining the objective

- The first step in developing a cognitive solution is understanding the type of problem a cognitive application will solve.

- The objective should take into consideration the types of users one will be appealing to, whether the users will be from different communities, what issues users are really interested in and what they need to know?
- A cognitive application should not only provide answers to questions, but also go deeper and investigate the context of how and why something happened.
- In a cognitive application, you must create an objective based on knowledge and data.
- The objective should probably focus on a specific segment of an industry rather than attempting to tackle all the problems for a particular industry.
- Some of the examples of objectives for cognitive health care applications are as follows :
  (a) Help determine whether the plan selected for treatment of a particular patient is the best and cost-effective.
  (b) Help patients optimize their health and wellness by providing personalized information and social support.

(2) Defining the domain

- The next step is to determine the domain or the subject area for the cognitive application.
- Identifying the domain helps to define and assess the data sources needed for the application.
- Domain definition also helps in determining the domain experts who can help in training the system.
- Some of the examples of cognitive application domains are as follows :
  (a) **Domain** : Medicine; Data Source: Electronic Medical Records (EMR); Domain Experts: Physicians
  (b) **Domain** : Retail; Data Source: Customer and product data; Domain Experts: Sales Associates

(3) Understanding the intended users and their attributes

- While developing cognitive application, one need to understand the types of users who will access that application.
- Expectations for user and system interactions will influence the corpus' creation, the user interface's design, and how the system is trained.
- The degree of precision necessary in a cognitive application is determined by the use case.
- A scientist, for example, demands far more precision than a customer service professional addressing concerns about replacement components.
- The best practices listed below can assist guarantee that your cognitive application has the flexibility it requires to give the appropriate level of support to its users.
  (a) Analyze how well the users understand the domain.
  (b) Prepare for a wide range of questions and forms of analysis.
  (c) Keep the application's scope broad enough to accommodate a variety of users.

(4) Defining questions and exploring insights

- Cognitive systems can provide insights about a domain, a topic, an individual, or an issue based on training and observations of all types, volumes, and velocities of data.
- A cognitive system generates and scores hypotheses to answer questions or provide insight based on models of the domain.
- When designing a cognitive application, one should start by mapping out the types of questions the users may ask.
- The questions users will ask can be placed in two general categories :
  (a) **Question-Answer Pairs** : The answers to these questions can be found in a data source. There may be contradictory answers within the data sources, and the cognitive system will analyze the alternatives to provide multiple responses with corresponding confidence levels.
  (b) **Anticipatory Analytics** : The user interacts with the cognitive application. The user may ask some but not all of the questions. Predictive models will be used by the cognitive application to anticipate the user's next question or series of questions.

(5) Acquiring the relevant data sources

- When creating a corpus, choose the most relevant data sources. This is difficult because one cannot predict what type of insights users will require as their needs change over time.

- Taking the time to evaluate data sources one already has and those one might want to acquire, on the other hand, offers tremendous opportunity.
- One may also discover that they have internal data resources that, when tapped by a cognitive system, can provide new insights.
- One may also want to include data from social media or other external sources.
- Cognitive systems enable the use of data sources in new directions. To begin building the corpus, one must first understand needs for various internal and external data sources.
- Identify what **internal data sources** are going to be meaningful. For example, a healthcare application needs data on patient's current health status, medical history and hospital admission.
- Analyze **dark data**. Dark data refers to the data that has been stored over many years and sometimes decades. Such data might not be previously analyzed. For example, dark data could be data about performance of a company's stock over a decade.
- Leverage **external data** from sources like industry-specific technical magazines or journals focusing on new research findings. For example, in medical domain, the results from clinical trials can provide insights into drug interactions.

(6) Creating and refining the corpora

- The process of creating a corpus of data includes preparing the data, ingesting the data, refining the data, and governing the data throughout its life cycle.

(i) Preparing the data

- All data entered into the corpus must be validated first to ensure that it is readable, searchable, and understandable.
- Prior to ingestion into a corpus, all data sources must be evaluated to determine if any transformations or enhancements are required.
- Check whether text-based resources are properly annotated and tagged.

(ii) Ingesting the data

- Data ingestion is not something that occurs only once during the system's development.
- Existing data sources are constantly updated and refined to ensure their accuracy and timeliness. Furthermore, changing user expectations are likely to result in new corpora additions.
- Delays in making necessary corpora updates will reduce the system's effectiveness and accuracy.
- As a result, in order for the cognitive system to remain viable, data sources may need to be ingested in near real time.

(iii) Refining and Expanding the Corpora

- A corpus must be constantly refined to ensure that the cognitive application provides accurate information and the appropriate level of insight.
- Despite extensive preparation for ingesting content required to provide a good knowledge base for the cognitive application, it is difficult to anticipate all data requirements at the outset.
- One may discover early in the training process that the accuracy of an answer to a specific question is below the acceptable threshold.
- One should improve accuracy by increasing coverage (adding more data) for specific topic areas in their domain.
- Plan on repeating the process of training, observing results, and then adding to the corpus several times.
- As one progress through the testing process and after the application is operational, one must establish an ongoing process of updating data requirements and adding to the corpus.
- One can use expansion algorithms to determine which additional information would best fill in gaps and add nuance to the corpus's information sources.

(iv) Governance of Data

- A cognitive application will incorporate a wide variety of data sources.
- It is possible that personal data in the organization may be subject to the same data privacy rules as data used in other systems.
- As a result, one must adhere to the same privacy and security requirements as any system.
- There will be data ingested into the corpus that may have restrictions on use due to governance requirements.

- It is possible that the corpus might contain content or images that are copyrighted. Thus, one should ensure that you have a license to use the content.

(7) Training and Testing

- A cognitive system begins to learn through an iterative process of model development, analysis, training, and testing.
- Training and testing can ensure that your application works as intended when it becomes operational.
- In order to determine what level of accuracy is acceptable, one need to measure responses.
- By developing well-designed cognitive systems, business knowledge can be uncovered in a way that has never been possible before.

☞ Self-Learning Topic

Cognitive Systems such as IBM's Watson

- Watson is a question-answering computer system capable of answering questions posed in natural language, developed in IBM's DeepQA project by a research team led by principal investigator David Ferrucci.
- Watson was named after IBM's founder and first CEO, industrialist Thomas J. Watson.
- Watson was created as a question answering (QA) computing system that IBM built to apply advanced natural language processing, information retrieval, knowledge representation, automated reasoning, and machine learning technologies to the field of open domain question answering.
- IBM stated that in Watson "more than 100 different techniques are used to analyze natural language, identify sources, find and generate hypotheses, find and score evidence, and merge and rank hypotheses."
- IBM Watson Analytics is a data analysis and visualization app that discovers patterns and insights out of the user's data.
- This is an intelligent and self-service application that users can use as a guide through the insight discovery process.
- The related cognitive processes and predictive analysis that come afterward are then automated by the system.
- The application uses natural language processing, which allows its users to interact with the system as if they were conversing with it. This makes it easy to extract answers from both structured and unstructured data.
- IBM Watson Analytics also gives the users information about new and emerging trends in their data.

IBM Watson Analytics Features

Main features of IBM Watson Analytics are :

(1) Advanced Analytics
(2) Natural Language Dialogue
(3) Automated Predictive Analysis
(4) Smart Data Discovery
(5) Self-Service Dashboards
(6) One-Click Analysis

IBM Watson Analytics Benefits

The main benefits of IBM Watson Analytics are its natural language processing capability, accessibility from different devices, easy pattern detection, visually engaging information format and reliable insights for decision-making. To know more, here are its benefits :

(1) Natural language processing capability
(2) Accessible from different devices
(3) Easy detection of patterns
(4) Visually engaging format of information
(5) Helps make future business decisions

(1) Natural language processing capability

- IBM Watson Analytics employs natural language processing, which enables users to have conversations with their data.
- This allows the users to use their own words when asking about the information on their work that they can fully understand.

(2) Accessible from different devices

- The application is not necessarily required to be installed in desktops or laptops since it's also accessible from iPads.
- This would allow users some mobility while still being productive at the same time.

- Regardless of their current location, users can still operate the app and find or process data on the go.

(3) Easy detection of patterns

- IBM Watson Analytics can easily detect patterns on the users' business data.
- It shows the users the factors that affect their business such as what makes the business appealing to the customers and what to do to make it more so.
- The application also shows users the process of how the discovery was made so that they can get to know more about their businesses.

(4) Visually engaging format of information

- Instead of just plain text format, IBM Watson Analytics offers the users the option to present their data in a more visually engaging way.
- It is accomplished by utilizing one of the pre-ready templates which can be customized further to fit the subject and to structure the report properly.
- This is a great way of delivering a report across an audience through a more interesting and comprehensible manner.

(5) Helps make future business decisions

- IBM Watson Analytics has the capability to execute predictive analytics which companies can use to get their selves ready from potential trouble that may come in the way.
- This would allow them to overcome such challenges and become more ready for facing future troubles and issues which might've otherwise taken them by surprise.

2.7 MULTIPLE CHOICE QUESTIONS

Q. 2.1 _____ algorithms try to mimic a human brain by analyzing text/speech/images/objects in a manner that a human does and tries to give the desired output.

(a) Cognitive Computing
(b) Artificial Intelligence
(c) Data Structure
(d) Genetic

✓**Ans. : (a)**

Q. 2.2 _____ is about creating a "Natural, human-like interaction".

(a) Artificial Intelligence
(b) Cognitive Computing
(c) Data Mining
(d) Deep Learning

✓**Ans. : (b)**

Q. 2.3 Siri, Google Assistant, Cortana, and Alexa are few of the best examples of _____.

(a) Artificial Intelligence
(b) Cognitive Computing
(c) Data Mining
(d) Deep Learning

✓**Ans. : (b)**

Q. 2.4 Which of the following is not an attribute of cognitive computing?

(a) Adaptive (b) Interactive
(c) Stateless (d) Contextual

✓**Ans. : (c)**

Q. 2.5 A _____ is the knowledge base of ingested data and is used to manage codified knowledge.

(a) Data Catalog (b) Corpus
(c) Taxonomy (d) Ontology

✓**Ans. : (b)**

Q. 2.6 _____ is the study of the physical sounds of a language and its utterance.

(a) Phonology (b) Morphology
(c) Discourse Analysis (d) Pragmatics

✓**Ans. : (a)**

Q. 2.7 _____ is a technique that connects each word with its corresponding dictionary meaning.

(a) Lexical Analysis
(b) Syntax Analysis
(c) Discourse Analysis
(d) Pragmatics

✓**Ans. : (a)**

Q. 2.8 A _____ is the smallest individual unit of language that has a specific meaning.

(a) Phonemes (b) Morphemes
(c) Grapheme (d) Lexeme

✓**Ans. : (b)**

Q. 2.9 _____ studies or finds the actual meaning of the context by connecting component sentences.

(a) Phonology (b) Morphology
(c) Pragmatics (d) Discourse

✓**Ans. : (d)**

Q. 2.10 RDF and RDFS are used to represent a _____.

(a) Taxonomy (b) Ontology
(c) Pragmatics (d) Discourse

✓**Ans. : (a)**

Q. 2.11 _____ is a vocabulary used to describe some domain.
(a) Taxonomy (b) Ontology
(c) Pragmatics (d) Discourse ✓**Ans. : (b)**

Q. 2.12 NLU stands for _____.
(a) Natural Language Unit
(b) Natural Language Understanding
(c) Natural Library Unit
(d) Natural Level Understanding ✓**Ans. : (b)**

Q. 2.13 _____ analytics answers the question of what happened.
(a) Descriptive (b) Diagnostic
(c) Predictive (d) Prescriptive ✓**Ans. : (a)**

Q. 2.14 _____ analytics answers the question of why something happened.
(a) Descriptive (b) Diagnostic
(c) Predictive (d) Prescriptive ✓**Ans. : (b)**

Q. 2.15 _____ analytics answers the question of what is likely to happen.
(a) Descriptive (b) Diagnostic
(c) Predictive (d) Prescriptive ✓**Ans. : (c)**

Q. 2.16 _____ analytics answers the question of what action to take.
(a) Descriptive (b) Diagnostic
(c) Predictive (d) Prescriptive ✓**Ans. : (d)**

Q. 2.17 Companies employ _____ analytics to find patterns in this data to identify risks and opportunities.
(a) Predictive (b) Text
(c) Image (d) Speech ✓**Ans. : (a)**

Q. 2.18 _____ analytics is the process of transforming unstructured text documents into usable, structured data.
(a) Predictive (b) Text
(c) Image (d) Speech ✓**Ans. : (b)**

Q. 2.19 _____ analytics make use of logical analysis to interpret information from visuals and graphics.
(a) Predictive (b) Text
(c) Image (d) Speech ✓**Ans. : (c)**

Q. 2.20 _____ analytics software takes spoken word data in multiple languages and dialects and transcribes it into text for analysis.
(a) Predictive (b) Text
(c) Image (d) Speech ✓**Ans. : (d)**

DESCRIPTIVE QUESTIONS

Q. 1 Define Cognitive Computing. List the features of a cognitive system.

Q. 2 State the advantages and disadvantages of cognitive computing.

Q. 3 Compare AI and Cognitive Computing.

Q. 4 Explain the elements present in a cognitive system with neat diagram.

Q. 5 Discuss the design principles for cognitive system.

Q. 6 Explain how NLP supports cognitive system.

Q. 7 With neat diagram, explain the phases involved in Natural Language Processing (NLP).

Q. 8 Differentiate between Taxonomy and Ontology.

Q. 9 Discuss in detail the maturity levels of analytics.

Q. 10 Explain how machine learning is used in the analytics process.

Q. 11 Write a note on Predictive Analytics.

Q. 12 Briefly explain Text Analytics.

Q. 13 Write a note on Image Analytics.

Q. 14 Write a note on Speech Analytics.

Q. 15 Explain the steps involved in building a cognitive application.

Chapter Ends...
□□□

MODULE 3

CHAPTER 3

Fuzzy Logic & Its Applications

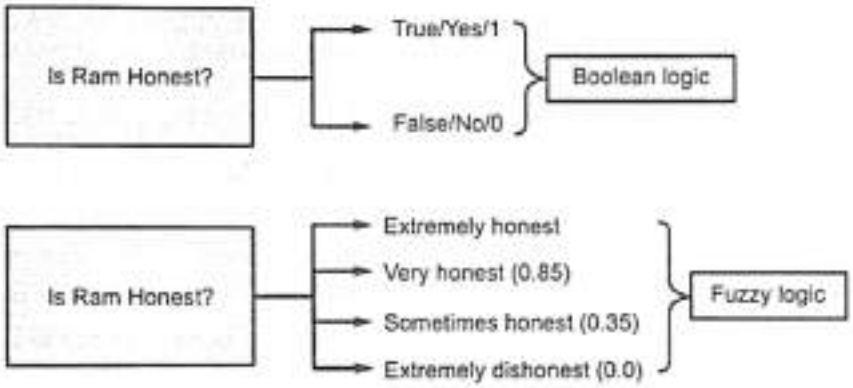
University Prescribed Syllabus w.e.f Academic Year 2022-2023

Introduction to Fuzzy Sets, Properties of Fuzzy Sets, Operations on Fuzzy Sets, Fuzzy Membership Functions, Fuzzy Relations with Operations and its Properties, Fuzzy Composition: Max-Min Composition, Max-Product Composition, Defuzzification Methods, Architecture of Mamdani Type Fuzzy Control System, Design of Fuzzy Controllers like Domestic Shower Controller, Washing Machine Controller, Water Purifier Controller, etc.

Self-learning Topics : Other Fuzzy Composition Operations, Fuzzy Inference System (FIS) and ANFIS.

3.1 INTRODUCTION TO FUZZY LOGIC

- The word **'Fuzzy'** means the things that are not clear or are vague.
- Sometimes, we cannot decide in real life that the given problem or statement is either true or false.
- At that time, this concept provides many values between the true and false and gives the flexibility to find the best solution to that problem.
- Fuzzy Logic resembles the human decision-making methodology.
- Fuzzy logic is an approach to computing based on "degrees of truth" rather than the usual "true or false" (1 or 0) Boolean logic on which the modern computer is based.
- The idea of fuzzy logic was first advanced by Lotfi Zadeh of the University of California at Berkeley in the 1960s.
- This concept provides the possibilities which are not given by computers, but similar to the range of possibilities generated by humans.
- In the Boolean system, only two possibilities (0 and 1) exist, where 1 denotes the absolute truth value and 0 denotes the absolute false value. But in the fuzzy system, there are multiple possibilities present between the 0 and 1, which are partially false and partially true.
- The Fuzzy logic can be implemented in systems such as micro-controllers, workstation-based or large network-based systems for achieving the definite output. It can also be implemented in both hardware or software.
- In artificial intelligence (AI) systems, fuzzy logic is used to imitate human reasoning and cognition. Rather than strictly binary cases of truth, fuzzy logic includes 0 and 1 as extreme cases of truth but with various intermediate degrees of truth.



(ICT)Fig. 3.1.1 : Boolean Logic Vs Fuzzy Logic

3.1.1 Applications of Fuzzy Logic

Various types of AI systems and technologies use fuzzy logic. This includes **vehicle intelligence**, consumer electronics, medicine, software, chemicals and aerospace.

1. In automobiles, fuzzy logic is used for gear selection and is based on factors such as engine load, road conditions and style of driving.
2. In copy machines, fuzzy logic is used to adjust drum voltage based on factors such as humidity, picture density and temperature.
3. In medicine, fuzzy logic is used for computer-aided diagnoses, based on factors such as symptoms and medical history.
4. In chemical distillation, fuzzy logic is used to control pH and temperature variables.
5. In dishwashers, fuzzy logic is used to determine the washing strategy and power needed, which is based on factors such as the number of dishes and the level of food residue on the dishes.
6. In aerospace, fuzzy logic is used to manage altitude control for satellites and spacecraft's based on environmental factors.

7. In natural language processing, fuzzy logic is used to determine semantic relations between concepts represented by words and other linguistic variables.
8. In environmental control systems, such as air conditioners and heaters, fuzzy logic determines output based on factors such as current temperature and target temperature.
9. In a business rules engine, fuzzy logic may be used to streamline decision-making according to predetermined criteria.

3.1.2 Architecture of a Fuzzy Logic System

The architecture consists of the different four components which are given below :

1. Rule Base
2. Fuzzification
3. Inference Engine
4. Defuzzification

Fig. 3.1.2 shows the architecture or process of a Fuzzy Logic system.

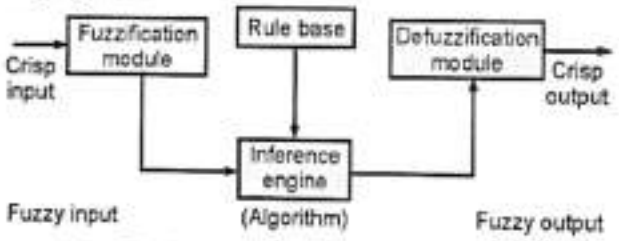


Fig. 3.1.2 : Fuzzyy Logic Architecture

1. Rule Base

- This contains the rules and membership function that regulate or control decision-making in the fuzzy logic system.
- It also contains the 'IF-THEN' conditions used for condition programming and controlling the system.

2. Fuzzification

- Fuzzification is a module or component for transforming the system inputs, i.e., it converts the crisp number into fuzzy steps.
- The crisp numbers are those inputs which are measured by the sensors and then fuzzification passed them into the control systems for further processing.
- This component divides the input signals into following five states in any Fuzzy Logic system:
  i. Large Positive (LP)
  ii. Medium Positive (MP)
  iii. Small (S)
  iv. Medium Negative (MN)
  v. Large negative (LN)

3. Inference Engine

- This component is a main component in any Fuzzy Logic system (FLS), because all the information is processed in the Inference Engine.
- It allows users to find the matching degree between the current fuzzy input and the rules.
- After the matching degree, this system determines which rule is to be added according to the given input field.
- When all rules are fired, then they are combined for developing the control actions.

4. Defuzzification

- Defuzzification is a module or component, which takes the fuzzy set inputs generated by the Inference Engine, and then transforms them into a crisp value.
- It is the last step in the process of a fuzzy logic system. The crisp value is a type of value which is acceptable by the user.
- Various techniques are present to do this, but the user has to select the best one for reducing the errors.

3.1.3 Advantages of Fuzzy Logic System

1. The structure of Fuzzy Logic Systems is easy and understandable.
2. Fuzzy logic is widely used for commercial and practical purposes.
3. Fuzzy logic in AI helps you to control machines and consumer products.
4. It may not offer accurate reasoning, but the only acceptable reasoning.
5. Fuzzy logic in Data Mining helps you to deal with the uncertainty in engineering.
6. Mostly robust as no precise inputs required.
7. It can be programmed to in the situation when feedback sensor stops working.
8. It can easily be modified to improve or alter system performance.
9. Inexpensive sensors can be used which helps you to keep the overall system cost and complexity low.
10. It provides a most effective solution to complex issues.

3.1.4 Disadvantages of Fuzzy Logic Systems

1. Fuzzy logic is not always accurate, so the results are perceived based on assumption, and so it may not be widely accepted.
2. Fuzzy systems don't have the capability of machine learning as-well-as neural network type pattern recognition.
3. Validation and Verification of a fuzzy knowledge-based system needs extensive testing with hardware.
4. Setting exact fuzzy rules and, membership functions is a difficult task.
5. Some fuzzy time logic is confused with probability theory and the terms.

3.1.5 Fuzzy Logic Membership Function

Definition : A membership function is a graphical representation of a fuzzy set. It indicates how values **between 0 and 1** are mapped to inputs. Inputs are generally represented by universe set u (or X).

The membership function for a given fuzzy set is in the form :

$\mu_A : X \rightarrow [0, 1]$, where A is fuzzy set and X is universe

Remark

(1) Any value within the range of 0 and 1 indicates a **degree of membership.**

(2) Each element of the universe X is given a specific degree of membership.

(3) In simple terms, the membership function is used to estimate or compute the degree of membership of a certain input element in a specified fuzzy set.

(4) The universe set X is plotted on X-axis and the degrees of membership are on Y-axis.

(5) The fuzzy set A can be expressed as :

$$A = \{(x, \mu_A(x)), x \in X\}$$

(6) The values in between 0 and 1 represent fuzziness.

3.1.6 Fuzzy Rules

(1) Fuzzy sets form the building blocks for fuzzy IF-THEN rules. If A and B are fuzzy sets then 'IF X is A THEN Y is B'.

The 'IF' part is called as the 'antecedent' and the 'THEN' part is called a 'consequent'.

(2) A fuzzy system is a set of fuzzy rules that converts inputs to outputs.

(3) The fuzzy inference algorithm combines fuzzy 'IF-THEN' rules into a mapping from space X to space Y using fuzzy logic principles.

Where X is the input space and Y is outer space.

(4) A fuzzy 'IF-THEN' rule is a scheme for capturing knowledge that involves imprecision.

3.1.7 Crisp Sets (Classical Sets)

- A classical set is a collection of distinct objects. The classical set is defined in such a way that the universe of discourse is splitted into two groups : Members and non-members.
- Let x be an object in a crisp set A. This is either a member or a non-member of the set A. this is either a member or a non-member of set A.
- Let U be the universal set. The collection of elements in the universe is called whole set. The total number of elements in the universe u is called cardinal number, denoted by n_U.
- Collections of elements within a universe set are called sets, and collection of elements within a set are called subsets.
- Let A be a crisp set in the universe U. Then we have the following :

(1) An object x is a member of a set A, i.e. $x \in A$: (x belongs to A).

(2) If $x \notin A$ (i.e. x does not belong to A).

(3) The member function for a set A is defined by

$$\mu_A(x) = \begin{cases} 1, & \text{if } x \in A \\ 0, & \text{if } x \notin A \end{cases}$$

The membership is denoted by letter μ.

(4) The set with no elements is denoted by ϕ, called as empty set.

(5) The set containing all possible subsets of a given set A is called power set and is denoted by

$$P(A) = \{x \mid x \subseteq A\}$$

(6) Let A and B be crisp sets containing some elements in the universe X, then

$x \in A \Rightarrow$ x belongs to A

$x \notin A \Rightarrow$ x does not belong to A.

(7) Let A and B be two crisp sets

$A \subset B \Rightarrow$ if $x \in A$ then $x \in B$

$A \subseteq B \Rightarrow$ A is contained in B or is equivalent to B

$A = B \Rightarrow A \subseteq B$ and $B \subseteq A$.

3.1.8 Set-theoretical Operations on Classical Sets

(1) **Union :** The union between two sets is all the element in the universe that are either in A or in B or in both A and B.

$A \cup B = \{x \mid x \in A \text{ or } x \in B\}$

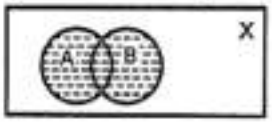


Fig. 3.1.3 : Union of two sets

The union operation is a logical 'OR' operation and written symbolically as '∪'.

(2) **Intersection :** It is the collection of all the elements which are both in A and B.

$A \cap B = \{x \mid x \in A \text{ and } x \in B\}$

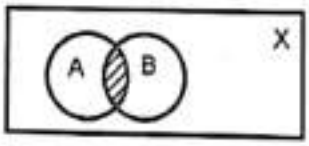


Fig. 3.1.4 : Intersection of two sets

The intersection operation is logical 'AND' operation and symbolically it is written as '∩'.

(3) **Complements :** The complement of a set A is the collection of all elements which do not Delong to A and is denoted by $\bar{A}$.

i.e. $\bar{A} = \{x \mid x \notin A, x \in X\}$

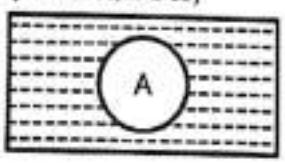


Fig. 3.1.5 : Complement of A i.e. $\bar{A}$

(4) **Difference of two sets :** Let A and B be two sets.

The difference of set A with reference to set B is the collection of all elements in A which are not in B, and denoted by A/B or A – B,

i.e. $A - B = \{x \mid x \in A \text{ and } x \notin B\} = A - (A \cap B)$

Also, $B - A = \{x \mid x \in B \text{ and } x \notin A\} = B - (A \cap B)$

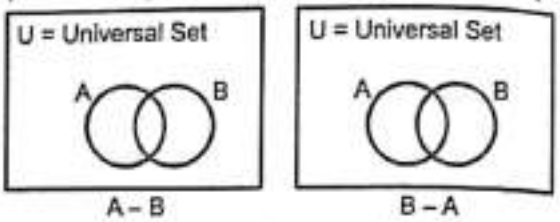


Fig. 3.1.6 : Difference Operation

3.1.9 Properties of Classical Sets

We mention below the properties of classical sets :

(1)	**Commutativity**	$A \cup B = B \cup A ; A \cap B = B \cap A$
(2)	**Associativity**	$A \cup (B \cup C) = (A \cup B) \cup C$ $A \cap (B \cap C) = (A \cap B) \cap C$
(3)	**Idempotency**	$A \cup A = A, \quad A \cap A = A$
(4)	**Distributivity**	$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$ $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
(5)	**Transitivity**	If $A \subseteq B$ and $B \subseteq C$, then $A \subseteq C$
(6)	**Identity**	$A \cup \phi = A, \quad A \cap \phi = A$ $A \cup X = X, \quad A \cap X = A$
(7)	**Double negation (Involution)**	$\bar{\bar{A}} = A$
(8)	**Law of excluded middle**	$A \cup \bar{A} = X$
(9)	**Law of contradiction**	$A \cap \bar{A} = \phi$
(10)	**De Morgan's Laws**	$\overline{(A \cap B)} = \bar{A} \cup \bar{B}$ and $\overline{(A \cup B)} = \bar{A} \cap \bar{B}$

3.1.10 Function Mapping of Classical Sets

- A classical set is represented by it's characteristic function X (x) where x is an element in the universe.

- Let $x \in X$ correspond to an element Y in [0, 1], then it is called mapping from X to [0, 1] i.e. $f : X \rightarrow [0, 1]$. On the basis of this mapping, the characteristic function is defined as,

$$X_A(x) = \begin{cases} 1, & \text{if } x \in A \\ 0, & \text{if } x \notin A \end{cases}$$

where X_A is the membership in set A for element x in the universe.

- The membership concept represents mapping from an element x in universe X to one of the two elements [either to elements 0 or to 1].

3.1.11 The Function-Theoretic Form

Let A and B be two sets on universe X. We define function-theoretic form of operations between two sets as follows :

1.	**Union ($A \cup B$)**	$X_{A \cup B}(x) = X_A(x) \vee X_B(x) = \max \cdot \{X_A(x), X_B(x)\}$
2.	**Intersection ($A \cap B$)**	$X_{A \cap B}(x) = X_A(x) \wedge X_B(x) = \min \cdot \{X_A(x), X_B(x)\}$
3.	**Complement ($\bar{A}$)**	$X_{\bar{A}(x)} = 1 - X_A(x)$
4.	**Containment**	If $A \subseteq B$ then $X_A(x) \leq X_B(x)$

3.2 FUZZY SETS

Definition : A fuzzy set A in the universe of discourse u can be defined as a set of **ordered pairs** and it is give by,

$$\underline{A} = \{(x, \mu_{\underline{A}}(x)) \mid x \in u\}$$

Where $\mu_{\underline{A}}(x)$ is the degree of membership of x in $\underline{A}$ and it is the degree that x belongs to $\underline{A}$.

Note that $\mu_{\underline{A}}(x) \in [0, 1]$

3.2.1 Methods of Representation of Fuzzy Sets

(A) When the universe of discourse U is discrete and finite, we write $\underline{A}$ as :,

$$\underline{A} = \left\{ \frac{\mu_{\underline{A}_1}(x_1)}{x_1} + \frac{\mu_{\underline{A}_2}(x_2)}{x_2} + \frac{\mu_{\underline{A}_3}(x_3)}{x_3} + \ldots \right\}$$

$$= \left\{ \sum_{i=0}^{n} \frac{\mu_{\underline{A}_i}(x_i)}{x_i} \right\} ; \text{ where n is finite value}$$

(B) When the universe of discourse U is continuous and infinite, then fuzzy set $\underline{A}$ is given by,

$$\underline{A} = \left\{ \int \frac{\mu_{\underline{A}}(x)}{x} \right\}$$

3.2.2 Remarks

(1) The horizontal bar in the above two representation of fuzzy sets for discrete and continuous universe is not a quotient (or division) but the representative of the corresponding variable, and called as a 'delimiter'

(2) The numerator in each representation is the membership value in set $\underline{A}$.

(3) The summation symbol '+' in the representation of fuzzy set is not 'addition' but it is a discrete function union.

(4) A fuzzy set is universal fuzzy set if

$$\mu_u(x) = 1, \quad \forall x \in u$$

and it is empty set if

$$\mu_\phi(x) = 0, \quad \forall x \in u$$

(5) Two fuzzy sets $\underline{A}$ and $\underline{B}$ are said to be equal if

$$\mu_{\underline{A}}(x) = \mu_B(x), \quad \forall x \in u.$$

Also, $\because A \subseteq u, \quad \therefore \mu_A(x) \subseteq \mu_u(x) = 1, \quad \forall x \in u$

3.2.3 Different Notations for Representing a Fuzzy Set

1. Ordered Pairs

$$\underline{A} = \{(x, \mu_{\underline{A}}(x)) \mid x \in X\}$$

Example : $\underline{A} = \{(1, 0, 3), (2, 1, 0), (3, 0, 3), (4, 0, 0)\}$

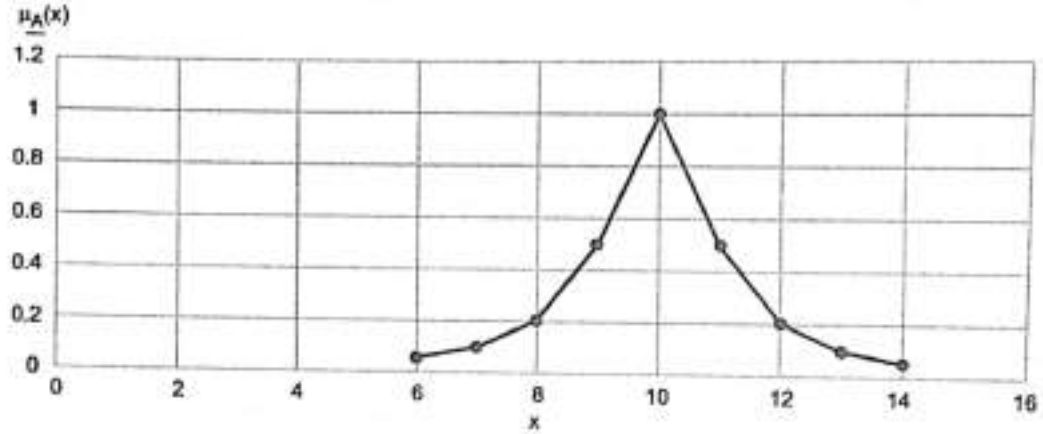
In each tuple, first value represents element of the set, and second value represents its membership in the set.

2. Using Membership Function

Let us represent the concept : real number close to 10 by fuzzy set $\underline{A}$. This concept can be modeled using following function :

$$\mu_{\underline{A}}(x) = \frac{1}{1 + (x - 10)^2}$$

x	6	7	8	9	10	11	12	13	14
$\mu_{\underline{A}}(x)$	0.058824	0.1	0.2	0.5	1	0.5	0.2	0.1	0.058824



(1C7) Fig. 3.2.1

Let us represent the concept : real number greater than 10 by fuzzy set $\underline{A}$. This concept can be modeled using following function :

$$\mu_{\underline{A}}(x) = \frac{1}{1 + \frac{1}{(x - 10)^2}}$$

x	10	11	12	13	14	15	16	17	18
$\mu_{\underline{A}}(x)$	0	0.5	0.8	0.9	0.9411	0.9615	0.9729	0.98	0.9846

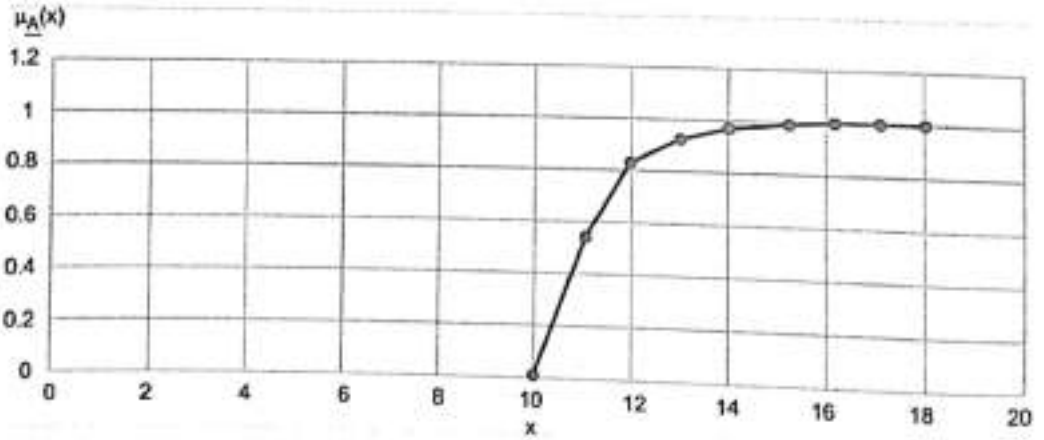


Fig. 3.2.2

3. Using Sum of Membership Function

$$\underline{A} = \sum_{i=1}^{n} \frac{\mu_{\underline{A}}(x_i)}{x_i} = \frac{\mu_{\underline{A}}(x_1)}{x_1} + \frac{\mu_{\underline{A}}(x_2)}{x_2} + \ldots.$$

Example :

$$\underline{A} = \frac{0.3}{1} + \frac{1.0}{2} + \frac{0.3}{3} + \frac{0.0}{4}$$

3.2.4 Operations on Fuzzy Sets

GQ. Discuss operations on sets.

Let $\underline{A}$ and $\underline{B}$ be fuzzy sets on u.

(I) **Union :** The union of $\underline{A}$ and $\underline{B}$ is denoted by $\underline{A} \cup \underline{B}$,

$$\therefore \mu_{\underline{A} \cup \underline{B}}(x) = \max [\mu_A(x), \mu_B(x)]$$

$$= \mu_A(x) \cup \mu_B(x), \ \forall x \in u$$

Where $\cup$ is 'OR' or 'max' operation

(II) **Intersection :** The intersection of fuzzy sets $\underline{A}$ and $\underline{B}$ denoted by $\underline{A} \cap \underline{B}$ and is defined by,

$$\mu_{\underline{A} \cap \underline{B}}(x) = \min [\mu_{\underline{A}}(x), \mu_{\underline{B}}(x)]$$

$$= \mu_A(x) \cap \mu_B(x), \forall x \in u.$$

Where $\cap$ is 'and' symbol and here it is minimum operator.

(III) **Complement :** The complement of $\underline{A}$ is denote by $\overline{\underline{A}}$ and is given by

$$\mu_{\overline{\underline{A}}}(x) = 1 - \mu_{\underline{A}}(x)$$

(IV) **Algebraic sum :** The algebraic sum of $(\underline{A} + \underline{B})$ of fuzzy sets is defined as

$$\mu_{\underline{A}+\underline{B}}(x) = \mu_{\underline{A}}(x) + \mu_{\underline{B}}(x) - \mu_{\underline{A}}(x) \cdot \mu_{\underline{B}}(x)$$

(V) **Algebraic product :** The algebraic product $(\underline{A} \cdot \underline{B})$ of two fuzzy sets $\underline{A}$ and $\underline{B}$ is defined as

$$\mu_{\underline{A} \cdot \underline{B}}(x) = \mu_{\underline{A}}(x) \cdot \mu_{\underline{B}}(x)$$

(VI) **Bounded sum :** The bounded sum $(A \oplus B)$ of two fuzzy sets $\underline{A}$ and $\underline{B}$ is defined as

$$\mu_{\underline{A} \oplus \underline{B}}(x) = \min [1, \mu_{\underline{A}}(x) + \mu_{\underline{B}}(x)]$$

(VII) **Bounded difference :** The bounded difference $(\underline{A} \odot \underline{B})$ of two fuzzy sets $\underline{A}$ and $\underline{B}$ is defined as

$$\mu_{\underline{A} \odot \underline{B}}(x) = \max \cdot [0, \mu_{\underline{A}}(x) - \mu_{\underline{B}}(x)]$$

3.2.5 Properties of Fuzzy Sets

UQ. What are the different properties of Fuzzy sets. MU - Q. 1(d), May 18, 5 Marks

Let $\underline{A}$ be a fuzzy set

(I) $\underline{A} \cup \overline{\underline{A}} \neq u$ and $\underline{A} \cap \overline{\underline{A}} \neq \phi$

(Note that member of one fuzzy set can also be member of other fuzzy set in the same universe, Also sharp boundaries are eliminated in fuzzy sets).

Following properties are frequently used :

(II)	**Commutitivity**	$\underline{A} \cup \underline{B} = \underline{B} \cup \underline{A}$; $\underline{A} \cap \underline{B} = \underline{B} \cap \underline{A}$
(III)	**Associativity**	$\underline{A} \cup (\underline{B} \cup \underline{C}) = (\underline{A} \cup \underline{B}) \cup \underline{C}$ $\underline{A} \cap (\underline{B} \cap \underline{C}) = (\underline{A} \cap \underline{B}) \cap \underline{C}$
(IV)	**Distributivity**	$\underline{A} \cup (\underline{B} \cap \underline{C}) = (\underline{A} \cup \underline{B}) \cap (\underline{A} \cup \underline{C})$ $\underline{A} \cap (\underline{B} \cup \underline{C}) = (\underline{A} \cap \underline{B}) \cap (\underline{A} \cap \underline{C})$
(V)	**Idempotency**	$\underline{A} \cup \underline{A} = \underline{A}$ and $\underline{A} \cap \underline{A} = \underline{A}$
(VI)	**Identity**	$\underline{A} \cup \phi = \underline{A}$; $\underline{A} \cup u = u$ (universal set) $\underline{A} \cap \phi = \phi$ and $\underline{A} \cap u = \underline{A}$
(VII)	**Involution or double negation**	$\overline{\overline{\underline{A}}} = \underline{A}$
(VIII)	**Transitivity**	If $\underline{A} \subseteq \underline{B} \subseteq \underline{C}$ then $\underline{A} \subseteq \underline{C}$
(IX)	**De Morgan's laws**	$\overline{\underline{A} \cup \underline{B}} = \overline{\underline{A}} \cap \overline{\underline{B}}$; $\overline{\underline{A} \cap \underline{B}} = \overline{\underline{A}} \cup \overline{\underline{B}}$

3.2.6 Fuzzy Terminologies

Fuzzy terminologies define the properties of fuzzy sets. A complete set of fuzzy terminologies is discussed here.

1. **Support :** The support of a fuzzy set A is the set of all points $x \in X$ such that $\mu_{\underline{A}}(x) > 0$.

   Support $(\underline{A}) = \{ x \mid \mu_{\underline{A}}(x) > 0, x \in X \}$

> Note : Support of fuzzy set is its Strong 0-cut.

Graphically, we can define support of fuzzy set as,

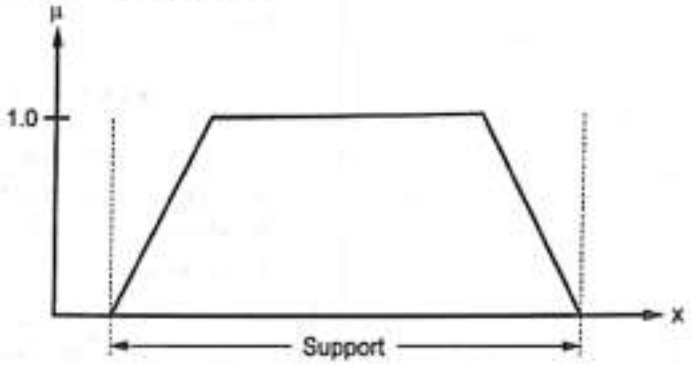


Fig. 3.2.3 : Support of Fuzzy Set

2. **Core :** The core of a fuzzy set $\underline{A}$ is the set of all points $x \in X$ such that $\mu_{\underline{A}}(x) = 1$.

   Core$(\underline{A}) = \{ x \mid \mu_{\underline{A}}(x) = 1, x \in X \}$

   All fuzzy sets might not have core present in it.

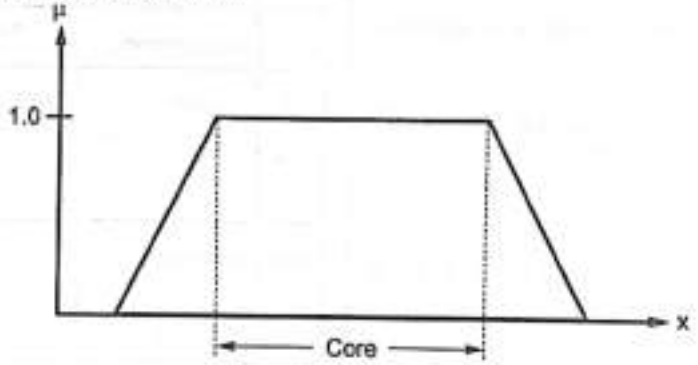


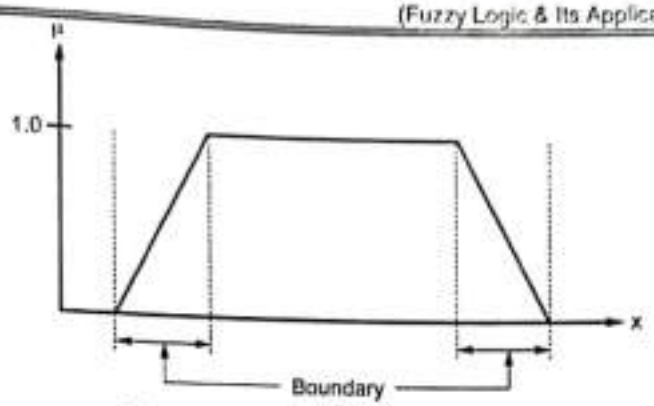
Fig. 3.2.4 : Core of Fuzzy Set

3. **Height of Fuzzy Set :** It is defined as the largest membership values of the elements contained in that set. It may not be 1 always. If core of fuzzy set is non empty, then height of fuzzy set is 1.
4. **Boundary : Boundary comprises those elements x of the universe such that $0 < \mu_{\underline{A}}(x) < 1$**

   Boundary $(\underline{A}) = \{ x \mid 0 < \mu_{\underline{A}}(x) < 1, x \in X \}$

   We can treat boundary as the difference of support and core.

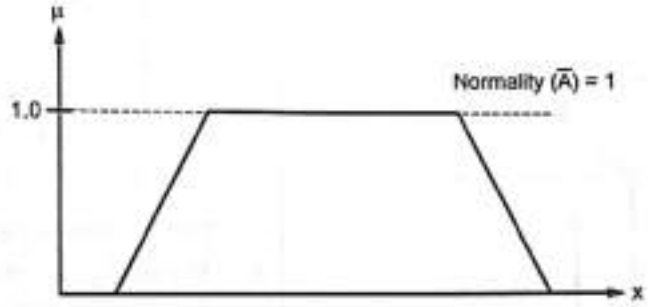
   Graphically, it is represented as



(1C10)**Fig. 3.2.5 : Boundary of Fuzzy Set**

5. **Normality : A fuzzy set $\underline{A}$ is normal if its core is non empty.** In other words, fuzzy set is normal if its height is 1.

Sub-normal Fuzzy set : For a sub-normal fuzzy set, $h(\underline{A}) < 1$, where $h(\underline{A})$ represents the height of fuzzy set / highest membership value in the fuzzy set.

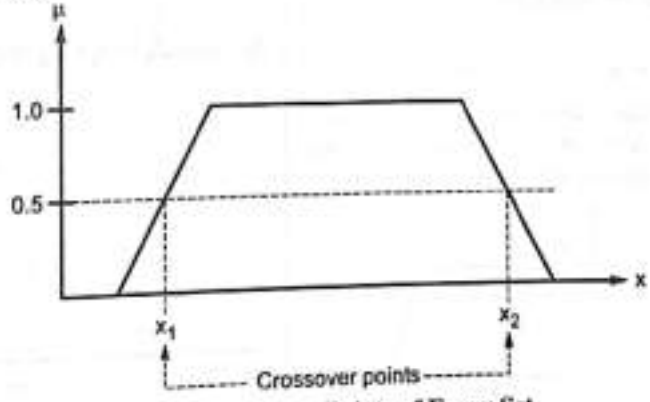


(1C11)**Fig. 3.2.6 : Normality of Fuzzy Set**

6. **Crossover Points : A crossover point of a fuzzy set $\underline{A}$ is a point $x \in X$ at which $\mu_{\underline{A}}(x) = 0.5$**

Crossover $(\underline{A}) = \{ x \mid \mu_{\underline{A}}(x) = 0.5 \}$

Graphically, we can represent it as

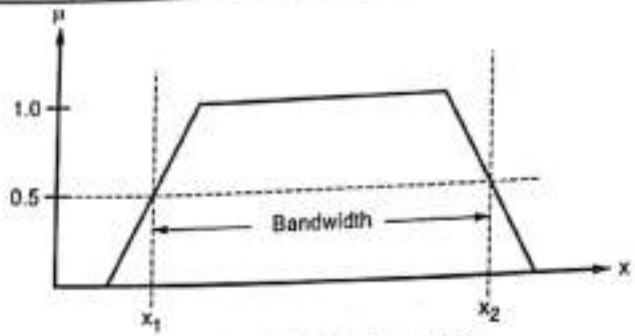


(1C12)**Fig. 3.2.7 : Crossover Points of Fuzzy Set**

7. **Bandwidth : For a fuzzy set, the bandwidth (or width) is defined as the distance between the two unique crossover points.**

Bandwidth $(\underline{A}) = |x_1 - x_2|$ where, $\mu_{\underline{A}}(x_1) = \mu_{\underline{A}}(x_2) = 0.5$

Graphically, we can represent it as

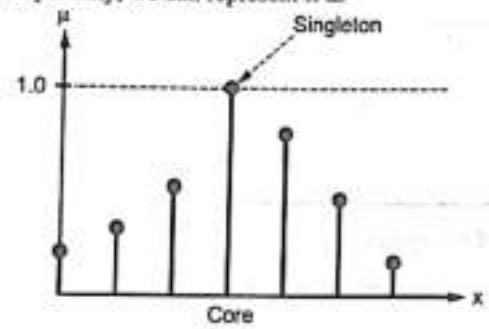


(IC13)Fig. 3.2.8 : Bandwidth of Fuzzy Set

8. **Fuzzy Singleton : A fuzzy set whose core is single point in X with $\mu_{\underline{A}}(x) = 1$, is called a fuzzy singleton. In other words, if fuzzy set is having only one element with membership value 1, then it is called fuzzy singleton.**

$$|A| = \{ \mu_{\underline{A}}(x) = 1 \}$$

Graphically, we can represent it as

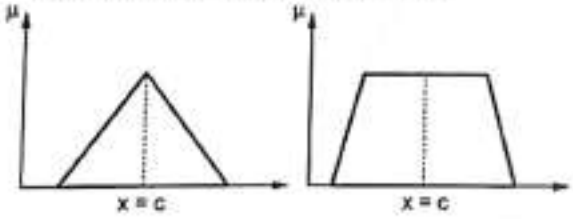


(IC14)Fig. 3.2.9 : Fuzzy Singleton

9. **Symmetry : Fuzzy set $\underline{A}$ is symmetric if its membership function around a centre point x = c is symmetric**

$$\text{i.e. } \mu_{\underline{A}}(x + c) = \mu_{\underline{A}}(x - c), \forall x \in X$$

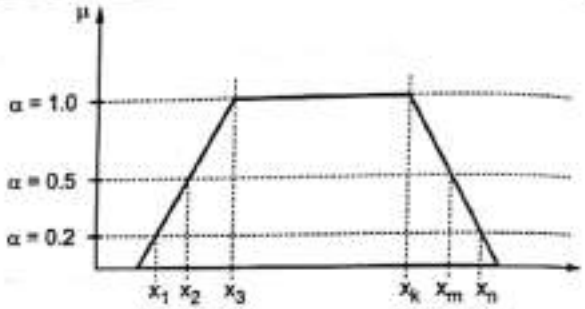
Triangular, Trapezoidal, Gaussian etc. are mostly symmetric. This is more natural to represent the membership then non-symmetric shape.



(IC15)Fig. 3.2.10 : Symmetric Fuzzy Set

10. **Alpha Cut : The α-cut of a fuzzy set $\underline{A}$ is a crisp set defined by $A_\alpha = \{ x \mid \mu_{\underline{A}}(x) \geq \alpha \}$**

Strong α-cut of a fuzzy set $\underline{A}$ is a crisp set defined by $A_{\alpha+} = \{ x \mid \mu_A(x) > \alpha \}$

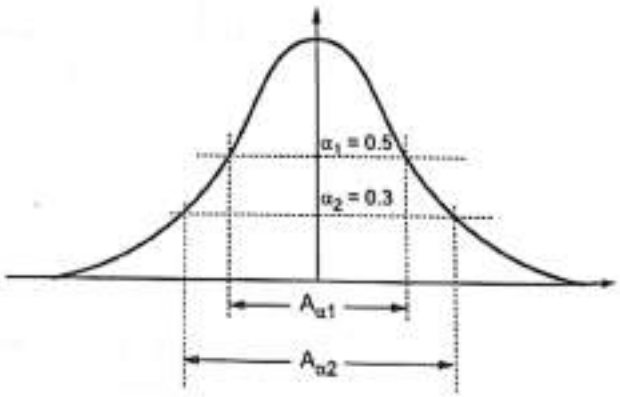


(IC16)Fig. 3.2.11 : Alpha Cut in Fuzzy Set

For the above diagram,

- **The set $A_{\alpha = 0.2}$ contains all the elements from x_1 to x_n, including both end values**
- **The set $A_{\alpha = 0.5}$ contains all the elements from x_2 to x_m, including both end values**
- **The set $A_{\alpha = 1.0}$ contains all the elements from x_3 to x_k, including both end values**

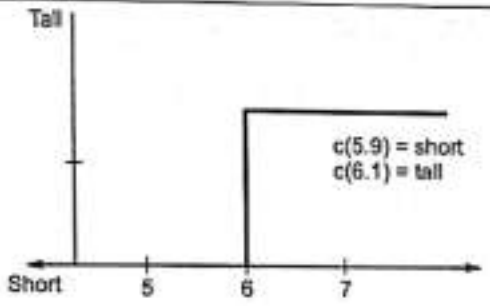
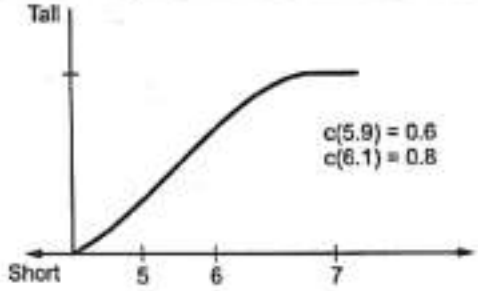
For different values of α, we get different crisp sets. In general, if $\alpha_1 > \alpha_2$ then $A_{\alpha 1} \subseteq A_{\alpha 2}$



$\alpha_1 > \alpha_2 \Rightarrow A_{\alpha 1} \subseteq A_{\alpha 2}$

(IC17)Fig. 3.2.12 : Relation Between Different α- Values

3.2.7 Crisp Set Vs Fuzzy Set

Sr. No.	Crisp Set	Fuzzy Set
1	Crisp set defines the value is either 0 or 1.	Fuzzy set defines the value between 0 and 1 including both 0 and 1.
2	It is also called a classical set.	It specifies the degree to which something is true.
3	It shows full membership	It shows partial membership.
4	Eg1. She is 18 years old. Eg2. Rahul is 1.6m tall	Eg1. She is about 18 years old. Eg2. Rahul is about 1.6m tall.
5	Crisp set application used for digital design.	Fuzzy set used in the fuzzy controller.
6	It is bi-valued function logic.	It is infinite valued function logic
7	Full membership means totally true/false, yes/no, 0/1.	Partial membership means true to false, yes to no, 0 to 1.
8	 (1C18)Fig. 3.2.13	 (1C19)Fig. 3.2.14

3.3 SOLVED EXAMPLES

Ex. 3.3.1 : Consider two given fuzzy sets

$$\underline{A} = \left\{\frac{1}{2} + \frac{0.3}{4} + \frac{0.5}{6} + \frac{0.2}{8}\right\}, \underline{B} = \left\{\frac{0.5}{2} + \frac{0.4}{4} + \frac{0.1}{6} + \frac{1}{8}\right\}.$$

Perform union, intersection, difference and complement over the fuzzy sets $\underline{A}$ and $\underline{B}$.

Soln. :

(i) Union : $\underline{A} \cup \underline{B} = \max\{\mu_{\underline{A}}(x), \mu_{\underline{B}}(x)\}$

$$= \left\{\frac{1}{2} + \frac{0.4}{4} + \frac{0.5}{6} + \frac{1}{8}\right\}$$

Note : The number 2, 4, 6, 8 in the denominator are delimiters, so to find maximum, numerators of first terms in $\underline{A}$ and $\underline{B}$, it is 1, and so on. Again '+' is a symbol and not an addition.

(ii) Intersection

$$\underline{A} \cap \underline{B} = \min\{\mu_{\underline{A}}(x), \mu_{\underline{B}}(x)\} = \left\{\frac{0.5}{2} + \frac{0.3}{4} + \frac{0.1}{6} + \frac{0.2}{8}\right\}$$

(iii) Complement

$$\overline{\underline{A}} = 1 - \mu_{\underline{A}}(x) = \left\{\frac{1-1}{2} + \frac{1-0.3}{4} + \frac{1-0.5}{6} + \frac{1-0.2}{8}\right\}$$

$$= \left\{\frac{0}{2} + \frac{0.7}{4} + \frac{0.5}{6} + \frac{0.8}{8}\right\}$$

and $$\overline{\underline{B}} = 1 - \mu_{\underline{B}}(x) = \left\{\frac{1-0.5}{2} + \frac{1-0.4}{4} + \frac{1-0.1}{6} + \frac{1-1}{8}\right\}$$

$$= \left\{\frac{0.5}{2} + \frac{0.6}{4} + \frac{0.9}{6} + \frac{0}{8}\right\}$$

(iv) Difference

$$\underline{A} / \underline{B} = \underline{A} - \underline{B} = \underline{A} \cap \overline{\underline{B}} = \left\{\frac{0.5}{2} + \frac{0.3}{4} + \frac{0.5}{6} + \frac{0}{8}\right\}$$

And $$B / A = B - A = B \cap \overline{A} = \left\{\frac{0}{2} + \frac{0.4}{4} + \frac{0.1}{6} + \frac{0.8}{8}\right\}$$

Ex. 3.3.2 : Given two fuzzy sets

$$B_1 = \left\{ \frac{1}{1.0} + \frac{0.75}{1.5} + \frac{0.3}{2.0} + \frac{0.15}{2.5} + \frac{0}{3.0} \right\}, \quad B_2 = \left\{ \frac{1}{1.0} + \frac{0.6}{1.5} + \frac{0.2}{2.0} + \frac{0.1}{2.5} + \frac{0}{3.0} \right\}$$

Find the following

(a) $B_1 \cup B_2$ (b) $B_1 \cap B_2$ (c) $\overline{B_1}$ (d) $\overline{B_2}$ (e) B_1 / B_2 (f) $\overline{B_1 \cup B_2}$

(g) $\overline{B_1 \cap B_2}$ (h) $B_1 \cap \overline{B_1}$ (i) $B_1 \cup \overline{B_1}$ (j) $B_2 \cap \overline{B_2}$ (k) $B_2 \cup \overline{B_2}$

☑ **Soln. :** From the given fuzzy sets.

[Again note that the denominators of the terms of B_1 and B_2 are delimiters and '+' is a symbol and not arithmetic addition].

(a) $B_1 \cup B_2 \max \{ \mu_{B_1}(x), \mu_{B_2}(x) \} = \left\{ \frac{1}{1.0} + \frac{0.75}{1.5} + \frac{0.3}{2.0} + \frac{0.15}{2.5} + \frac{0}{3.0} \right\}$

(b) $B_1 \cap B_2 = \left\{ \frac{1}{1.0} + \frac{0.6}{1.5} + \frac{0.2}{2.0} + \frac{0.1}{2.5} + \frac{0}{3.0} \right\}$

(c) $\overline{B}_1 = 1 - \mu_{B_1}(x) = \left\{ \frac{1-1}{1.0} + \frac{1-0.75}{1.5} + \frac{1-0.3}{2.0} + \frac{1-0.15}{2.5} + \frac{1-0}{3} \right\} = \left\{ \frac{0}{1.0} + \frac{0.25}{1.5} + \frac{0.7}{2.0} + \frac{0.85}{2.5} + \frac{1}{3} \right\}$

(d) $\overline{B}_2 = 1 - \mu_{B_2}(x) = \left\{ \frac{1-1}{1.0} + \frac{1-0.6}{1.5} + \frac{1-0.2}{2.0} + \frac{1-0.1}{2.5} + \frac{1-0}{3.0} \right\} = \left\{ \frac{0}{1.0} + \frac{0.4}{1.5} + \frac{0.8}{2.0} + \frac{0.9}{2.5} + \frac{1}{3.0} \right\}$

(e) $B_1/B_2 = B_1 - B_2 = B_1 \cap \overline{B}_2 = \min(\mu_{B_1}(x), \mu_{B_2}(x)) = \left\{ \frac{0}{1.0} + \frac{0.4}{1.5} + \frac{0.3}{2.0} + \frac{0.15}{2.5} + \frac{0}{3.0} \right\}$

(f) $\overline{B_1 \cup B_2} = \overline{B}_1 \cap \overline{B}_2$ (by De Morgan's laws)

$$= \min \{ \mu_{\overline{B_1}}(x), \mu_{\overline{B_2}}(x) \} = \left\{ \frac{0}{1.0} + \frac{0.25}{1.5} + \frac{0.7}{2.0} + \frac{0.85}{2.5} + \frac{1}{3.0} \right\}$$

(g) $\overline{B_1 \cap B_2} = \overline{B}_1 \cup \overline{B}_2$ (by De Morgan's laws)

$$= \max \{ \mu_{\overline{B_1}}(x), \mu_{\overline{B_2}}(x) \} = \left\{ \frac{0}{1.0} + \frac{0.4}{1.5} + \frac{0.8}{2.0} + \frac{0.9}{2.5} + \frac{1}{3.0} \right\}$$

(h) $B_1 \cap \overline{B}_1 = \min \{ \mu_{B_1}(x), \mu_{\overline{B}_1}(x) \} = \left\{ \frac{0}{1.0} + \frac{0.25}{1.5} + \frac{0.3}{2.0} + \frac{0.15}{2.5} + \frac{0}{3.0} \right\}$

(i) $B_1 \cap \overline{B}_1 = \max \cdot \{ \mu_{B_1}(x), \mu_{\overline{B_1}}(x) \} = \left\{ \frac{1}{1.0} + \frac{0.75}{1.5} + \frac{0.7}{2.0} + \frac{0.85}{2.5} + \frac{1}{3.0} \right\}$

(j) $B_2 \cap \overline{B}_2 = \min \cdot \{ \mu_{B_2}(x), \mu_{\overline{B_2}}(x) \} = \left\{ \frac{0}{1.0} + \frac{0.4}{1.5} + \frac{0.2}{2.0} + \frac{0.1}{2.5} + \frac{0}{3.0} \right\}$

(k) $B_2 \cap \overline{B}_2 = \max \cdot \{ \mu_{B_2}(x), \mu_{\overline{B_2}}(x) \} = \left\{ \frac{1}{1.0} + \frac{0.6}{1.5} + \frac{0.8}{2.0} + \frac{0.9}{2.5} + \frac{1}{3.0} \right\}$

Ex. 3.3.3 : Consider two fuzzy sets :

$\underline{A} = \left\{\frac{0.2}{1} + \frac{0.3}{2} + \frac{0.4}{3} + \frac{0.5}{4}\right\}$ and $\underline{B} = \left\{\frac{0.1}{1} + \frac{0.2}{2} + \frac{0.2}{3} + \frac{1}{4}\right\}$

Find the algebraic sum, algebraic product, bounded sum and bounded difference of the given fuzzy sets.

☑ **Soln. :**

(i) The algebraic sum is given by

$$\mu_{\underline{A}+\underline{B}}(x) = \mu_{\underline{A}}(x) + \mu_{\underline{B}}(x) - [\mu_{\underline{A}}(x) \cdot \mu_{\underline{B}}(x)] \quad ...(i)$$

Now, $\mu_{\underline{A}}(x) \cdot \mu_{\underline{B}}(x) = \left\{\frac{0.02}{1} + \frac{0.06}{2} + \frac{0.08}{3} + \frac{0.05}{4}\right\}$ ...(ii)

$\therefore \mu_{\underline{A}+\underline{B}}(x)$

$$= \left\{\frac{0.2+0.1-0.2}{1} + \frac{0.3+0.2-0.6}{2} + \frac{0.4+0.2-0.08}{3} + \frac{0.5+0.1-0.05}{4}\right\}$$

$$= \left\{\frac{0.28}{1} + \frac{0.44}{2} + \frac{0.52}{3} + \frac{0.55}{4}\right\}$$

(ii) Bounded sum

$$= \min \cdot [1, \mu_{\underline{A}}(x) + \mu_{\underline{B}}(x)]$$

$$= \min \cdot \left\{1, \left\{\frac{0.3}{1} + \frac{0.5}{2} + \frac{0.6}{3} + \frac{0.5}{4}\right\}\right\}$$

$$= \left\{\frac{0.3}{1} + \frac{0.5}{2} + \frac{0.6}{3} + \frac{0.5}{4}\right\}$$

(iii) Bounded difference

$$\mu_{\underline{A} \ominus \underline{B}}(x) = \max \{0, \mu_{\underline{A}}(x) - \mu_{\underline{B}}(x)]$$

$$= \max \left\{0, \left\{\frac{0.1}{1} + \frac{0.1}{2} + \frac{0.2}{3} + \frac{0.5}{4}\right\}\right\}$$

$$= \left\{\frac{0.1}{1} + \frac{0.1}{2} + \frac{0.2}{3} + \frac{0.5}{4}\right\}$$

UEx. 3.3.4 MU - Q. 1(a), Dec. 19, 3 Marks

Find (a) $\underline{A} \cap \underline{B}$ (ii) $\underline{A} \cup \underline{B}$ for the given fuzzy sets :

$A = \left\{\frac{0.4}{a} + \frac{0.2}{b} + \frac{0.9}{c}\right\}$, $B = \left\{\frac{0.1}{a} + \frac{0.5}{b} + \frac{0.8}{c}\right\}$

☑ **Soln. :**

(i) $\underline{A} \cap \underline{B} = \min \{\mu_A(x), \mu_B(x)\} = \left\{\frac{0.1}{a} + \frac{0.2}{b} + \frac{0.8}{c}\right\}$

(ii) $\underline{A} \cup \underline{B} = \max \{\mu_A(x), \mu_B(x)\} = \left\{\frac{0.4}{a} + \frac{0.5}{b} + \frac{0.9}{c}\right\}$

UEx. 3.3.5 MU - Q. 1(b), Dec. 19, 4 Marks

Prove De Morgan's theorem for the following fuzzy sets :

$A = \left\{\frac{0}{0} + \frac{0.5}{20} + \frac{0.65}{40} + \frac{0.85}{60}\right\}$; $B = \left\{\frac{0}{0} + \frac{0.45}{20} + \frac{0.6}{40} + \frac{0.8}{60}\right\}$

To show that $\overline{(A_1 \cup B_2)} = \bar{A} \cap \bar{B}$

☑ **Soln. :**

▶ **Step I :**

We have $A \cup B = \max \{\mu_A(x), \mu_B(x)\}$

$$\therefore \mu(A \cup B) = \left\{\frac{0}{0} + \frac{0.5}{20} + \frac{0.65}{40} + \frac{0.85}{60}\right\}$$

$$\overline{A \cup B} = 1 - \mu_{A \cup B} = 1 - \max \{\mu_A(x), \mu_B(x)\}$$

$$= 1 - \left\{\frac{0}{0} + \frac{0.5}{20} + \frac{0.65}{40} + \frac{0.85}{60}\right\}$$

$$= \left\{\frac{1}{0} + \frac{0.5}{20} + \frac{0.35}{40} + \frac{0.15}{60}\right\} \quad ...(i)$$

Now, $\mu_{\bar{A}} = 1 - \mu_A = 1 - \left\{\frac{0}{0} + \frac{0.5}{20} + \frac{0.65}{40} + \frac{0.85}{60}\right\}$

$$= \left\{\frac{1}{0} + \frac{0.5}{20} + \frac{0.35}{40} + \frac{0.15}{60}\right\}$$

$$\mu_{\bar{B}} = 1 - \mu_B = 1 - \left\{\frac{0}{0} + \frac{0.45}{20} + \frac{0.6}{40} + \frac{0.8}{60}\right\}$$

$$= \left\{\frac{1}{0} + \frac{0.55}{20} + \frac{0.4}{40} + \frac{0.2}{60}\right\}$$

$$\therefore \bar{A} \cap \bar{B} = \min \{\mu_{\bar{A}}, \mu_{\bar{B}}\}$$

$$= \left\{\frac{1}{0} + \frac{0.5}{20} + \frac{0.35}{40} + \frac{0.15}{60}\right\} \quad ...(ii)$$

From (i), (ii), the theorem is verified,

UEx. 3.3.6 MU - Q. 2(b), Dec. 19, 5 Marks

For the two fuzzy sets :

Consider two fuzzy sets given by :

$\underline{A} = \left\{\frac{0.5}{2} + \frac{0.1}{3} + \frac{0.6}{4}\right\}$, $\underline{B} = \left\{\frac{0.7}{2} + \frac{0.2}{3} + \frac{0.4}{4}\right\}$

Find (i) $\underline{A} \cup \underline{B}$ (ii) $\underline{A} \cap \underline{B}$ (iii) $\bar{A}$ (iv) $\bar{A} \cup B$ of the fuzzy sets

☑ **Soln. :**

(i) $\underline{A} \cup \underline{B} = \max \cdot \{\mu_{\underline{A}}(x), \mu_{\underline{B}}(x)\} = \left\{\frac{0.7}{2} + \frac{0.2}{3} + \frac{0.6}{4}\right\}$

(ii) $\underline{A} \cap \underline{B} = \min \cdot \{\mu_{\underline{A}}(x), \mu_{\underline{B}}(x)\} = \left\{\frac{0.5}{2} + \frac{0.1}{3} + \frac{0.4}{4}\right\}$

(iii) $\bar{\underline{A}} = 1 - \underline{A} = 1 - \left\{\frac{0.5}{2} + \frac{0.1}{3} + \frac{0.6}{4}\right\} = \left\{\frac{0.5}{2} + \frac{0.9}{3} + \frac{0.4}{4}\right\}$

(iv) $\bar{A} \cup B = \max \{\mu_{\bar{A}}(x), \mu_B(x)\} = \left\{\frac{0.7}{2} + \frac{0.9}{3} + \frac{0.4}{4}\right\}$

UEx. 3.3.7 MU - Q. 1(d), Dec. 18, 5 Marks

Consider two fuzzy set given by

$$\underline{A} = \left\{\frac{1}{2} + \frac{0.2}{3} + \frac{0.5}{4}\right\}, \underline{B} = \left\{\frac{0.9}{2} + \frac{0.4}{3} + \frac{0.8}{4}\right\}$$

Find (i) $\underline{A} \cup \underline{B}$ (ii) $A \cap B$ (iii) $\bar{A}$, (iv) $\bar{A} \cup B$ of the fuzzy sets

☑ **Soln. :**

We have $\underline{A} = \left\{\frac{1}{2} + \frac{0.2}{3} + \frac{0.5}{4}\right\}$; $\underline{B} = \left\{\frac{0.9}{2} + \frac{0.4}{3} + \frac{0.8}{4}\right\}$

(i) $\underline{A} \cup \underline{B} = \max\{\mu_{\underline{A}}(x), \mu_{\underline{B}}(x)\} = \left\{\frac{1}{2} + \frac{0.4}{3} + \frac{0.8}{4}\right\}$

(ii) $\underline{A} \cap \underline{B} = \min\{\mu_{\underline{A}}(x), \mu_{\underline{B}}(x)\} = \left\{\frac{0.9}{2} + \frac{0.2}{3} + \frac{0.5}{4}\right\}$

(iii) $\bar{\underline{A}} = 1 - \underline{A} = 1 - \left\{\frac{1}{2} + \frac{0.2}{3} + \frac{0.5}{4}\right\} = \left\{\frac{0}{2} + \frac{0.8}{3} + \frac{0.5}{4}\right\}$

(iv) $\bar{A} \cup B = \max\cdot\{\mu_{\bar{\underline{A}}}(x), \mu_{\underline{B}}(x)\} = \left\{\frac{0.9}{2} + \frac{0.8}{3} + \frac{0.8}{4}\right\}$

UEx. 3.3.8 MU - Q. 1(b), Dec. 17, 5 Marks

For a Fuzzy set, $\underline{A} = \left\{\frac{0.5}{x_1} + \frac{0.4}{x_2} + \frac{0.7}{x_3} + \frac{0.8}{x_4} + \frac{1}{x_5}\right\}$, perform Fuzzy complement on $\underline{A}$.

☑ **Soln. :**

Complement

$$\bar{\underline{A}} = 1 - \mu_{\underline{A}}(x)$$

$$= \left\{\frac{1-0.5}{x_1} + \frac{1-0.4}{x_2} + \frac{1-0.7}{x_3} + \frac{1-0.8}{x_4} + \frac{1-1}{x_5}\right\}$$

$$= \left\{\frac{0.5}{x_1} + \frac{0.6}{x_2} + \frac{0.3}{x_3} + \frac{0.2}{x_4} + \frac{0}{x_5}\right\}$$

Ex. 3.3.9 : Consider the set X = {2, 4, 6, 8, 10}. Find it's power set, and cardinality of power set.

☑ **Soln. :** Cardinality of set X = number of elements in X.

$\therefore n_X = 5$

and cardinality of power set of X is

$2^{n_X} = 2^5 = 32$

P(X) = collection of all subsets of X

= {ϕ, {2}, {4}, {6}, {8}, {10}, {2, 4}, {2, 6} {2, 8}, {2, 10}, {4, 6}, {4, 8}, {4, 10}, {6, 8}, {6, 10}, {8, 10}, {2, 4, 6}, {2, 4, 8}, {2, 4, 10}, {2, 6, 8}, {2, 6, 10}, {2, 8, 10}, {4, 6, 8}, {4, 6, 10}, {4, 8, 10}, {6, 8, 10}, {2, 4, 6, 8}, {2, 4, 6, 10}, {2, 4, 8, 10}, {2, 6, 8, 10}, {4, 6, 8, 10}, {2, 4, 6, 8, 10}}

Ex. 3.3.10 : It is necessary to compare two sensors based upon their detection levels and gain settings. The table of gain settings and sensor detection levels with a standard item being monitored providing typical membership values to represent the detection levels for each sensor is given in Table :

Gain setting	Detection level of sensor 1	Detection level of sensor 2
1	0	0
10	0.2	0.35
20	0.35	0.25
30	0.65	0.8
40	0.85	0.95
50	1	1

Given the universe of discourse X = {0, 10, 20, 30, 40, 50} and the membership function for the two sensors in discrete form as

$$\underline{D_1} = \left\{\frac{0}{0} + \frac{0.2}{10} + \frac{0.35}{20} + \frac{0.65}{30} + \frac{0.85}{40} + \frac{1}{50}\right\} \quad \underline{D_2} = \left\{\frac{0}{0} + \frac{0.35}{10} + \frac{0.25}{20} + \frac{0.8}{30} + \frac{0.95}{40} + \frac{1}{50}\right\}$$

Find the following membership functions

(a) $\mu_{\underline{D_1} \cup \underline{D_2}}(x)$ (b) $\mu_{\underline{D_1} \cap \underline{D_2}}(x)$ (c) $\mu_{\bar{\underline{D_1}}}(x)$ (d) $\mu_{\bar{\underline{D_2}}}(x)$ (e) $\mu_{\underline{D_1} \cup \bar{\underline{D_1}}}(x)$ (f) $\mu_{\underline{D_1} \cap \bar{\underline{D_1}}}(x)$

(g) $\mu_{\underline{D_2} \cup \bar{\underline{D_2}}}(x)$ (h) $\mu_{\bar{\underline{D_2}} \cap \bar{\underline{D_2}}}(x)$ (i) $\mu_{\underline{D_1}/\underline{D_2}}(x)$ (j) $\mu_{\underline{D_2}/\underline{D_1}}(x)$

☑ **Soln. :**

(a) From the given fuzzy sets, we have

$$\mu_{D_1 \cup D_2} = \max\{\mu_{D_1}(x), \mu_{D_2}(x)\} = \left\{\frac{0}{0} + \frac{0.35}{10} + \frac{0.35}{20} + \frac{0.8}{30} + \frac{0.95}{40} + \frac{1}{50}\right\}$$

(b) And

$$\mu_{D_1 \cap D_2}(x) = \min\{\mu_{D_1}(x), \mu_{D_2}(x)\} = \left\{\frac{0}{0} + \frac{0.2}{10} + \frac{0.25}{20} + \frac{0.65}{30} + \frac{0.85}{40} + \frac{1}{50}\right\}$$

(c) $$\mu_{\overline{D_1}}(x) = 1 - \mu_{D_1}(x) = \left\{\frac{1-0}{0} + \frac{1-0.2}{10} + \frac{1-0.35}{20} + \frac{1-0.65}{30} + \frac{1-0.85}{40} + \frac{1-1}{50}\right\}$$

$$= \left\{\frac{1}{0} + \frac{0.8}{10} + \frac{0.65}{20} + \frac{0.35}{30} + \frac{0.15}{40} + \frac{0}{50}\right\}$$

(d) $$\mu_{\overline{D_2}}(x) = -1\,\mu_{D_2}(x)$$

$$= \left\{\frac{1-0}{0} + \frac{1-0.35}{10} + \frac{1-0.25}{20} + \frac{1-0.8}{30} + \frac{1-0.95}{40} + \frac{1-1}{50}\right\} = \left\{\frac{1}{0} + \frac{0.65}{10} + \frac{0.75}{20} + \frac{0.2}{30} + \frac{0.05}{40} + \frac{0}{50}\right\}$$

(e) $$\mu_{D_1 \cup \overline{D_1}}(x) = \max[\mu_{D_1}(x), \mu_{\overline{D_1}}(x)] = \left\{\frac{1}{0} + \frac{0.8}{10} + \frac{0.65}{20} + \frac{0.65}{30} + \frac{0.85}{40} + \frac{1}{50}\right\}$$

(f) $$\mu_{D_1 \cap \overline{D_1}}(x) = \min[\mu_{D_1}(x), \mu_{\overline{D_1}}(x)] = \left\{\frac{0}{0} + \frac{0.2}{10} + \frac{0.35}{20} + \frac{0.35}{30} + \frac{0.15}{40} + \frac{0}{50}\right\}$$

(g) $$\mu_{D_2 \cup \overline{D_2}}(x) = \max[\mu_{D_2}(x), \mu_{\overline{D_2}}(x)] = \left\{\frac{1}{0} + \frac{0.65}{10} + \frac{0.75}{20} + \frac{0.8}{30} + \frac{0.95}{40} + \frac{1}{50}\right\}$$

(h) $$\mu_{D_2 \cap \overline{D_2}}(x) = \min[\mu_{D_2}(x), \mu_{\overline{D_2}}(x)] = \left\{\frac{0}{0} + \frac{0.35}{10} + \frac{0.25}{20} + \frac{0.2}{30} + \frac{0.05}{40} + \frac{0}{50}\right\}$$

(i) $$\mu_{D_1/D_2}(x) = \mu_{D_1 \cap \overline{D_2}}(x) = \min[\mu_{D_1}(x), \mu_{\overline{D_2}}(x)] = \left\{\frac{0}{0} + \frac{0.2}{10} + \frac{0.35}{20} + \frac{0.2}{30} + \frac{0.05}{40} + \frac{0}{50}\right\}$$

(j) $$\mu_{D_2/D_1}(x) = \mu_{D_2 \cap \overline{D_1}}(x) = \min[\mu_{D_2}(x), \mu_{\overline{D_1}}(x)] = \left\{\frac{0}{0} + \frac{0.35}{10} + \frac{0.25}{20} + \frac{0.35}{30} + \frac{0.15}{40} + \frac{0}{50}\right\}$$

Ex. 3.3.11 : Design a computer software to perform image processing to locate object within a scene. The two fuzzy sets representing a plane and a train image :

$$\text{Plane} = \left\{\frac{0.2}{\text{train}} + \frac{0.5}{\text{bike}} + \frac{0.3}{\text{boat}} + \frac{0.8}{\text{plane}} + \frac{0.1}{\text{horse}}\right\}$$

$$\text{Plane} = \left\{\frac{1}{\text{train}} + \frac{0.2}{\text{bike}} + \frac{0.4}{\text{boat}} + \frac{0.5}{\text{plane}} + \frac{0.2}{\text{horse}}\right\}$$

Find the following :

(a) Plane ∪ Train (b) Plane∩ Train

(c) $\overline{\text{Plane}}$ (d) $\overline{\text{Train}}$ (e) Plane/Train

(f) $\overline{\text{Plane} \cup \text{Train}}$ (g) $\overline{\text{Plane} \cap \text{Train}}$

(h) Plane ∪ $\overline{\text{Plane}}$ (i) Plane ∩ $\overline{\text{Plane}}$

(j) Train ∪ $\overline{\text{Train}}$ (k) Train ∩ $\overline{\text{Train}}$

☑ **Soln. :**

Using the given fuzzy sets, we have the following :

(a) Plane ∪ Train $= \max[\mu_{\text{plane}}(x), \mu_{\text{train}}(x)]$

$$= \left\{\frac{1.0}{\text{train}} + \frac{0.5}{\text{bike}} + \frac{0.4}{\text{boat}} + \frac{0.8}{\text{plane}} + \frac{0.2}{\text{horse}}\right\}$$

(b) Plane ∩ Train $= \min[\mu_{\text{plane}}(x), \mu_{\text{train}}(x)]$

$$= \left\{\frac{0.2}{\text{train}} + \frac{0.2}{\text{bike}} + \frac{0.3}{\text{boat}} + \frac{0.5}{\text{plane}} + \frac{0.1}{\text{horse}}\right\}$$

(c) $\overline{\text{Plane}} = 1 - \mu_{\text{plane}}(x)$

$$= \left\{\frac{1-0.2}{\text{train}} + \frac{1-0.5}{\text{bike}} + \frac{1-0.3}{\text{boat}} + \frac{1-0.8}{\text{plane}} + \frac{1-0.1}{\text{horse}}\right\}$$

$$= \left\{\frac{0.8}{\text{train}} + \frac{0.5}{\text{bike}} + \frac{0.7}{\text{boat}} + \frac{0.2}{\text{plane}} + \frac{0.9}{\text{horse}}\right\}$$

(d) $\overline{\text{Train}} = 1 - \mu_{\text{train}}(x)$

$$= \left\{ \frac{1-1}{\text{train}} + \frac{1-0.2}{\text{bike}} + \frac{1-0.4}{\text{boat}} + \frac{1-0.5}{\text{plane}} + \frac{1-0.2}{\text{horse}} \right\}$$

$$= \left\{ \frac{0}{\text{train}} + \frac{0.8}{\text{bike}} + \frac{0.6}{\text{boat}} + \frac{0.5}{\text{plane}} + \frac{0.8}{\text{horse}} \right\}$$

(e) Plane/Train

$$= \text{Plane} \cap \overline{\text{Train}} = \min [\mu_{\text{plane}}(x), \mu_{\overline{\text{train}}}(x)]$$

$$= \left\{ \frac{0}{\text{train}} + \frac{0.5}{\text{bike}} + \frac{0.3}{\text{boat}} + \frac{0.5}{\text{plane}} + \frac{0.1}{\text{horse}} \right\}$$

(f) $\overline{\text{Plane} \cup \text{Train}} = 1 - \max [\mu_{\text{plane}}(x), \mu_{\text{train}}(x)]$

$$= \left\{ \frac{1-1}{\text{train}} + \frac{1-0.5}{\text{bike}} + \frac{1-0.4}{\text{boat}} + \frac{1-0.8}{\text{plane}} + \frac{1-0.2}{\text{horse}} \right\}$$

$$= \left\{ \frac{0}{\text{train}} + \frac{0.5}{\text{bike}} + \frac{0.6}{\text{boat}} + \frac{0.2}{\text{plane}} + \frac{0.8}{\text{horse}} \right\}$$

(g) $\overline{\text{Plane} \cap \text{Train}} = 1 - \min [\mu_{\text{plane}}(x), \mu_{\text{train}}(x)]$

$$= \left\{ \frac{1-0.2}{\text{train}} + \frac{1-0.2}{\text{bike}} + \frac{1-0.3}{\text{boat}} + \frac{1-0.5}{\text{plane}} + \frac{1-0.1}{\text{horse}} \right\}$$

$$= \left\{ \frac{0.8}{\text{train}} + \frac{0.8}{\text{bike}} + \frac{0.7}{\text{boat}} + \frac{0.5}{\text{plane}} + \frac{0.9}{\text{horse}} \right\}$$

(h) $\text{Plane} \cup \overline{\text{Plane}} = \max [\mu_{\text{plane}}(x), \mu_{\overline{\text{plane}}}(x)]$

$$= \left\{ \frac{0.8}{\text{train}} + \frac{0.5}{\text{bike}} + \frac{0.7}{\text{boat}} + \frac{0.8}{\text{plane}} + \frac{0.9}{\text{horse}} \right\}$$

(i) $\text{Plane} \cap \overline{\text{Plane}} = \min [\mu_{\text{plane}}(x), \mu_{\overline{\text{plain}}}(x)]$

$$= \left\{ \frac{0.2}{\text{train}} + \frac{0.5}{\text{bike}} + \frac{0.3}{\text{boat}} + \frac{0.2}{\text{plane}} + \frac{0.1}{\text{horse}} \right\}$$

(j) $\text{Train} \cup \overline{\text{Train}} = \max [\mu_{\text{train}}(x), \mu_{\overline{\text{train}}}(x)]$

$$= \left\{ \frac{1.0}{\text{train}} + \frac{0.8}{\text{bike}} + \frac{0.6}{\text{boat}} + \frac{0.5}{\text{plane}} + \frac{0.8}{\text{horse}} \right\}$$

(k) $\text{Train} \cap \overline{\text{Train}} = \min [\mu_{\text{train}}(x), \mu_{\overline{\text{train}}}(x)]$

$$= \left\{ \frac{0}{\text{train}} + \frac{0.2}{\text{bike}} + \frac{0.4}{\text{boat}} + \frac{0.5}{\text{plane}} + \frac{0.2}{\text{horse}} \right\}$$

Ex. 3.3.12 : Let U be the universe of military aircraft of interest as defined below :

$U = \{a10, b52, c130, f2, fa\}$

Let $\underline{A}$ be the fuzzy set of bomber class aircraft

$$\underline{A} = \left\{ \frac{0.3}{a10} + \frac{0.4}{b52} + \frac{0.2}{c130} + \frac{0.1}{f2} + \frac{1}{f9} \right\}$$

Let $\underline{B}$ be the fuzzy set of fighter class aircraft

$$\underline{B} = \left\{ \frac{0.1}{a10} + \frac{0.2}{b52} + \frac{0.8}{c130} + \frac{0.7}{f2} + \frac{0}{f9} \right\}$$

Find the following

(a) $\underline{A} \cup \underline{B}$, (b) $\underline{A} \cap \underline{B}$, (c) $\overline{\underline{A}}$ (d) $\overline{\underline{B}}$ (e) $\underline{A} / \underline{B}$, (f) $\underline{B} / \underline{A}$,

(g) $\overline{\underline{A} \cup \underline{B}}$ (h) $\overline{\underline{A} \cap \underline{B}}$, (i) $\overline{\underline{A}} \cup \overline{\underline{B}}$, (j) $\overline{\underline{B}} \cup \underline{A}$

☑ **Soln. :**

From the given data, we have

(a) $\underline{A} \cup \underline{B} = \max [\mu_{\underline{A}}(x), \mu_{\underline{B}}(x)]$

$$= \left\{ \frac{0.3}{a10} + \frac{0.4}{b52} + \frac{0.8}{c130} + \frac{0.7}{f2} + \frac{1}{f9} \right\}$$

(b) $\underline{A} \cap \underline{B} = \min [\mu_{\underline{A}}(x), \mu_{\underline{B}}(x)]$

$$= \left\{ \frac{0.1}{a10} + \frac{0.2}{b52} + \frac{0.2}{c130} + \frac{0.1}{f2} + \frac{0}{f9} \right\}$$

(c) $\overline{\underline{A}} = 1 - \mu_{\underline{A}}(x)$

$$= 1 - \left\{ \frac{0.3}{a10} + \frac{0.4}{b52} + \frac{0.2}{c130} + \frac{0.1}{f2} + \frac{1}{f9} \right\}$$

$$= \left\{ \frac{0.7}{a10} + \frac{0.6}{b52} + \frac{0.8}{c130} + \frac{0.9}{f2} + \frac{0}{f9} \right\}$$

(d) $\overline{\underline{B}} = 1 - \mu_{\underline{B}}(x)$

$$= 1 - \left\{ \frac{0-1}{a10} + \frac{0.2}{b52} + \frac{0.8}{c130} + \frac{0.7}{f2} + \frac{0}{f9} \right\}$$

$$= \left\{ \frac{0.9}{a10} + \frac{0.8}{b52} + \frac{0.2}{c130} + \frac{0.3}{f2} + \frac{1}{f9} \right\}$$

(e) $\underline{A} / \underline{B} = \underline{A} \cap \overline{\underline{B}} = \min [\mu_{\underline{A}}(x), \mu_{\overline{\underline{B}}}(x)]$

$$= \left\{ \frac{0.3}{a10} + \frac{0.4}{b52} + \frac{0.2}{c130} + \frac{0.1}{f2} + \frac{1}{f9} \right\}$$

(f) $\underline{B} / \underline{A} = B \cap \overline{A} = \min \cdot \{\mu_{\underline{B}}(x), \mu_{\overline{\underline{A}}}(x)]$

$$= \left\{ \frac{0.1}{a10} + \frac{0.2}{b52} + \frac{0.8}{c130} + \frac{0.7}{f2} + \frac{0}{f9} \right\}$$

(g) $\overline{\underline{A} \cup \underline{B}} = 1 - \max [\mu_{\underline{A}}(x), \mu_{\underline{B}}(x)]$

$$= \left\{ \frac{1-0.3}{a10} + \frac{1-0.4}{b52} + \frac{1-0.8}{c130} + \frac{1-0.7}{f2} + \frac{1-1}{f9} \right\}$$

$$= \left\{ \frac{0.7}{a10} + \frac{0.6}{b52} + \frac{0.2}{c130} + \frac{0.3}{f2} + \frac{0}{f9} \right\}$$

(h) $\overline{\underline{A} \cap \underline{B}} = 1 - \min [\mu_{\underline{A}}(x), \mu_{\underline{B}}(x)]$

$$= \left\{ \frac{1-0.1}{a10} + \frac{1-0.2}{b52} + \frac{1-0.2}{c130} + \frac{1-0.1}{f2} + \frac{1-0}{f9} \right\}$$

$$= \left\{ \frac{0.9}{a10} + \frac{0.8}{b52} + \frac{0.8}{c130} + \frac{0.9}{f2} + \frac{1}{f9} \right\}$$

(i) $\bar{A} \cup \bar{B} = \max \{\mu_{\bar{A}}(x), \mu_{\bar{B}}(x)\}$

$$= \left\{\frac{0.9}{a10} + \frac{0.8}{b52} + \frac{0.8}{c130} + \frac{0.9}{f2} + \frac{1}{f9}\right\}$$

(j) $\bar{B} \cup A = \max \{\mu_{\bar{B}}(x), \mu_A(x)\}$

$$= \left\{\frac{0.9}{a10} + \frac{0.8}{b52} + \frac{0.2}{c130} + \frac{0.3}{f2} + \frac{1}{f9}\right\}$$

Ex. 3.3.13 : The discretised membership functions for a transistor and a resistor are as given :

$$\mu_T = \left\{\frac{0}{0} + \frac{0.2}{1} + \frac{0.7}{2} + \frac{0.8}{3} + \frac{0.9}{4} + \frac{1}{5}\right\}$$

$$\mu_R = \left\{\frac{0}{0} + \frac{0.1}{1} + \frac{0.3}{2} + \frac{0.2}{3} + \frac{0.4}{4} + \frac{0.5}{5}\right\}$$

Find the following

(a) Algebraic sum (b) Algebraic product

(c) bounded sum (d) bounded difference

☑ **Soln. :**

(a) Algebraic sum = $\mu_{T+R}(x)$

$$= \mu_T(x) + \mu_R(x) - [\mu_T(x) \cdot \mu_R(x)]$$

(by definition) ...(1)

Now, $\mu_T(x) \cdot \mu_R(x) = \left\{\frac{0}{0} + \frac{0.02}{1} + \frac{0.21}{2} + \frac{0.16}{3} + \frac{0.36}{4} + \frac{0.5}{5}\right\}$

$$\mu_{T+R}(x) = \left\{\frac{0}{0} + \frac{0.3}{1} + \frac{1.0}{2} + \frac{1.1}{3} + \frac{1.3}{4} + \frac{1.5}{5}\right\} - \left\{\frac{0}{0} + \frac{0.02}{1} + \frac{0.21}{2} + \frac{0.16}{3} + \frac{0.36}{4} + \frac{0.5}{5}\right\}$$

$$= \left\{\frac{0}{0} + \frac{0.28}{1} + \frac{0.79}{2} + \frac{0.94}{3} + \frac{0.94}{4} + \frac{1}{5}\right\}$$

(b) Algebraic product

$$\mu_{T \cdot R}(x) = \mu_T(x) \cdot \mu_R(x) = \left\{\frac{0}{0} + \frac{0.02}{1} + \frac{0.21}{2} + \frac{0.16}{3} + \frac{0.36}{4} + \frac{0.5}{5}\right\}$$

(c) Bounded sum

$$\mu_{T \oplus R}(x) = \min \{1, \mu_T(x) + \mu_R(x)\} \quad ...(1)$$

Now, $\mu_T(x) + \mu_R(x) = \left\{\frac{0}{0} + \frac{0.3}{1} + \frac{1.0}{2} + \frac{1.0}{3} + \frac{1.3}{4} + \frac{1.5}{5}\right\}$

$$\therefore \mu_{T \oplus R}(x) = \min \left\{1, \left\{\frac{0}{0} + \frac{0.3}{1} + \frac{1.0}{2} + \frac{1.0}{3} + \frac{1.3}{4} + \frac{1.5}{5}\right\}\right\}$$

$$= \left\{\frac{0}{0} + \frac{0.3}{1} + \frac{1.0}{2} + \frac{1.0}{3} + \frac{1.0}{4} + \frac{1.0}{5}\right\}$$

(d) Bounded difference

$$\mu_{T \ominus R}(x) = \max [0, \mu_T(x) - \mu_R(x)] \quad ...(2)$$

Now, $\mu_T(x) - \mu_R(x) = \left\{\frac{0}{0} + \frac{0.1}{1} + \frac{0.4}{2} + \frac{0.6}{3} + \frac{0.5}{4} + \frac{0.5}{5}\right\}$

From (2) $\therefore \mu_{T \ominus R}(x) = \left\{\frac{0}{0} + \frac{0.1}{1} + \frac{0.4}{2} + \frac{0.6}{3} + \frac{0.5}{4} + \frac{0.5}{5}\right\}$

Ex. 3.3.14 : Consider the two membership functions as follows :

For fuzzy set $\underline{A}$: $\mu_A(x) = \frac{|(60 - x)|}{8} + 1$

For fuzzy set $\underline{B}$: $\mu_B(x) = \frac{|(40 - x)|}{8} + 1$

Find the following

(a) $\underline{A} \cup \underline{B}$ (b) $\underline{A} \cap \underline{B}$ (c) $\bar{\underline{A}}$

(d) $\bar{\underline{B}}$, (e) $\overline{\underline{A} \cup \underline{B}}$ (f) $\overline{\underline{A} \cap \underline{B}}$

☑ **Soln. :**

(a) By definition

$$\underline{A} \cup \underline{B} = \max [\mu_{\underline{A}}(x), \mu_{\underline{B}}(x)] = \frac{|(60 - x)|}{8} + 1 \quad ...[\because |60 - x| > |40 - x| \text{ for } \forall x]$$

(b) $\underline{A} \cap \underline{B} = \min \{\mu_{\underline{A}}(x), \mu_{\underline{B}}(x)\} = \frac{|40 - x|}{8} + 1$

$$...[\because |40 - x| < |60 - x| \ \forall \ x]$$

(c) $\bar{\underline{A}} = 1 - \mu_{\underline{A}}(x) = 1 - \left[\frac{|60 - x|}{8} + 1\right] = -\frac{|60 - x|}{8}$

(d) $\bar{\underline{B}} = 1 - \mu_{\underline{B}}(x) = 1 - \left[\frac{|40 - x|}{8} + 1\right] = -\frac{|40 - x|}{8}$

(e) $\overline{\underline{A} \cup \underline{B}} = 1 - \mu_{\underline{A} \cup \underline{B}} = 1 - \left[\frac{|60 - x|}{8} + 1\right] = -\frac{|60 - x|}{8}$

(f) $\overline{\underline{A} \cap \underline{B}} = 1 - \mu_{\underline{A} \cap \underline{B}} = 1 - \left[\frac{|40 - x|}{8} + 1\right] = -\frac{|40 - x|}{8}$

Ex. 3.3.15 : The Discretized membership function (in non-dimensional units) for a UJT (unit junction transistor) and BJT (bipolar junction transistor) are given below :

$$\mu_{\underline{J1}} = \left\{\frac{0}{0} + \frac{0.2}{1} + \frac{0.3}{2} + \frac{0.6}{3} + \frac{0.9}{4} + \frac{1}{5}\right\}$$

$$\mu_{\underline{J2}} = \left\{\frac{0}{0} + \frac{0.1}{1} + \frac{0.2}{2} + \frac{0.3}{3} + \frac{0.4}{4} + \frac{0.5}{5}\right\}$$

For these two fuzzy sets, perform the following calculations :

(a) $\mu_{\underline{J1}} \vee \mu_{\underline{J2}}$ (b) $\mu_{\underline{J1}} \wedge \mu_{\underline{J2}}$

(c) $\bar{\mu}_{\underline{J1}}$ (d) $\bar{\mu}_{\underline{J2}}$ (e) $\overline{\mu_{\underline{J1}} \wedge \mu_{\underline{J2}}} = \bar{\mu}_{\underline{J1}} \vee \bar{\mu}_{\underline{J2}}$

☑ **Soln. :**

☞ **Remark** : Let A and B be two sets, then

$A \wedge B = A \cap B$; ($\wedge$ = and) and $A \vee B = A \cup B$; ($\vee$= or)

(a) $\mu_{\underline{J}1} \vee \mu_{\underline{J}2} = \max [\mu_{\underline{J}1}(x), \mu_{\underline{J}2}(x)]$

$$= \left\{\frac{0}{0} + \frac{0.2}{1} + \frac{0.3}{2} + \frac{0.6}{3} + \frac{0.9}{4} + \frac{1}{5}\right\}$$

(b) $\mu_{\underline{J}1} \wedge \mu_{\underline{J}2} = \min [\mu_{\underline{J}1}(x), \mu_{\underline{J}2}(x)]$

$$= \left\{\frac{0}{0} + \frac{0.1}{1} + \frac{0.2}{2} + \frac{0.3}{3} + \frac{0.4}{4} + \frac{0.7}{5}\right\}$$

(c) $\bar{\mu}_{\underline{J}1} = 1 - \mu_{\underline{J}1} = 1 - \left\{\frac{0}{0} + \frac{0.2}{1} + \frac{0.3}{2} + \frac{0.6}{3} + \frac{0.9}{4} + \frac{1}{5}\right\}$

$$= \left\{\frac{1}{0} + \frac{0.8}{1} + \frac{0.7}{2} + \frac{0.4}{3} + \frac{0.1}{4} + \frac{0}{5}\right\}$$

(d) $\bar{\mu}_{\underline{J}2} = 1 - \mu_{\underline{J}2} = 1 - \left\{\frac{0}{0} + \frac{0.1}{1} + \frac{0.2}{2} + \frac{0.3}{3} + \frac{0.4}{4} + \frac{0.7}{5}\right\}$

$$= \left\{\frac{1}{0} + \frac{0.9}{1} + \frac{0.8}{2} + \frac{0.7}{3} + \frac{0.6}{4} + \frac{0.3}{5}\right\}$$

(e) $\overline{\mu_{\underline{J}1} \wedge \mu_{\underline{J}2}} = 1 - \mu_{J1} \wedge \mu_2 = 1 - \min [\mu_{J1}(x), \mu_{J2}(x)]$

$$= 1 - \left\{\frac{0}{0} + \frac{0.1}{1} + \frac{0.2}{2} + \frac{0.3}{3} + \frac{0.4}{4} + \frac{0.7}{5}\right\}$$

$$= \left\{\frac{1}{0} + \frac{0.9}{1} + \frac{0.8}{2} + \frac{0.7}{3} + \frac{0.6}{4} + \frac{0.3}{5}\right\} \quad ...(i)$$

And $\bar{\mu}_{\underline{J}1} \vee \bar{\mu}_{\underline{J}2} = \max [\bar{\mu}_{\underline{J}1}(x), \bar{\mu}_{\underline{J}2}(x)]$

$$= \left\{\frac{1}{0} + \frac{0.9}{1} + \frac{0.8}{2} + \frac{0.7}{3} + \frac{0.6}{4} + \frac{0.3}{5}\right\} \quad ...(ii)$$

From (i), (ii)

$$\overline{\mu_{\underline{J}1} \wedge \mu_{\underline{J}2}} = \bar{\mu}_{\underline{J}1} \vee \bar{\mu}_{\underline{J}2}$$

Ex. 3.3.16 : Consider a local area network (LAN) of interconnected workstations that communicate using Ethernet protocols at a maximum rate of 12 Mbit/s. The two fuzzy sets given below represent the loading of the LAN :

$$\mu_S(x) = \left\{\frac{1.0}{0} + \frac{1.0}{1} + \frac{0.8}{2} + \frac{0.2}{5} + \frac{0.1}{7} + \frac{0.0}{9} + \frac{0.0}{10}\right\}$$

and $\mu_C(x) = \left\{\frac{0.0}{0} + \frac{0.0}{1} + \frac{0.0}{2} + \frac{0.5}{5} + \frac{0.7}{7} + \frac{0.8}{9} + \frac{1.0}{10}\right\}$

Where S represent silent and C represent congestion. Perform algebraic sum, algebraic product, bounded sum and bounded difference over the two fuzzy sets,

☑ **Soln. :**

(a) **Algebraic sum** of $\mu_S(x)$ and $\mu_C(x)$ is given by

$$\mu_{S+C}(x) = \mu_S(x) + \mu_C(x) - [\mu_S(x) \cdot \mu_C(x)] \quad ...(i)$$

From the given data, we first calculate

$$\mu_S(x) \cdot \mu_C(x) = \left\{\frac{0}{0} + \frac{0}{1} + \frac{0}{2} + \frac{0.1}{5} + \frac{0.07}{7} + \frac{0}{9} + \frac{0}{10}\right\} \quad ...(ii)$$

and $\mu_S(x) + \mu_C(x) = \left\{\frac{1.0}{0} + \frac{1.0}{1} + \frac{0.8}{2} + \frac{0.7}{5} + \frac{0.8}{7} + \frac{0.8}{9} + \frac{1.0}{10}\right\}$

...(iii)

Substituting (ii) and (iii) in (i);

$$\mu_{S+C}(x) = \left\{\frac{1.0}{0} + \frac{1.0}{1} + \frac{0.8}{2} + \frac{0.6}{5} + \frac{0.73}{7} + \frac{0.8}{9} + \frac{0.0}{10}\right\}$$

(b) Algebraic product

$$\mu_{S \cdot C}(x) = \mu_S(x) \cdot \mu_C(x)$$

$$= \left\{\frac{0}{0} + \frac{0}{1} + \frac{0}{2} + \frac{0.1}{5} + \frac{0.07}{7} + \frac{0}{9} + \frac{0}{10}\right\}$$

...From (ii)

(c) Bounded sum

$$\mu_{S \oplus C}(x) = \min [1, \mu_S(x) + \mu_C(x)]$$

$$= \min \left\{1, \left\{\frac{1.0}{0} + \frac{1.0}{1} + \frac{0.8}{2} + \frac{0.7}{5} + \frac{0.8}{7} + \frac{0.8}{9} + \frac{1.0}{10}\right\}\right\}$$

$$= \left\{\frac{1.0}{0} + \frac{1.0}{1} + \frac{0.8}{2} + \frac{0.7}{5} + \frac{0.8}{7} + \frac{0.8}{9} + \frac{1.0}{10}\right\}$$

(d) Bounded difference : It is given by

$$\mu_{S \ominus C}(x) = \max [0, \mu_S(x) - \mu_C(x)]$$

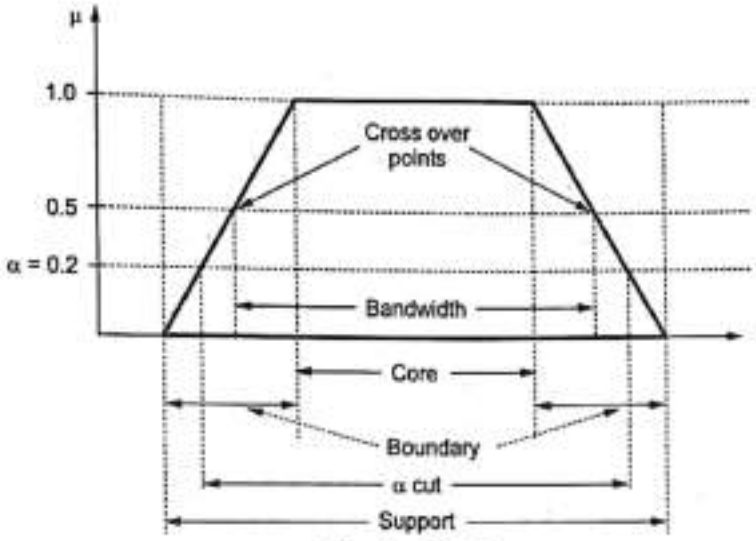
$$= \max \left\{0, \left\{\frac{1.0}{0} + \frac{1.0}{1} + \frac{0.8}{2} + \frac{(-0.3)}{5} + \frac{(-0.6)}{7} + \frac{(-0.8)}{9} + \frac{(-1.0)}{10}\right\}\right\}$$

$$= \left\{\frac{1.0}{0} + \frac{1.0}{1} + \frac{0.8}{2} + \frac{0}{5} + \frac{0}{7} + \frac{0}{9} + \frac{0}{10}\right\}$$

Ex. 3.3.17 : Let $\underline{A}$ = { $(x_1, 0), (x_2, 0.2), (x_3, 0.5), (x_4, 1), (x_5, 1), (x_6, 1), (x_7, 0.5), (x_8, 0.2), (x_9, 0)$ }

Find support, core, crossover points, alpha cut and strong alpha cut for $\alpha = 0.2$, boundary, bandwidth, normality of given fuzzy set.

☑ **Soln. :**



(IC19)**Fig. Ex. 3.3.17**

- **Support :** $\{(x_2, 0.2), (x_3, 0.5), (x_4, 1), (x_5, 1), (x_6, 1), (x_7, 0.5), (x_8, 0.2)\}$
- **Core :** $\{(x_4, 1), (x_5, 1), (x_6, 1)\}$
- **Crossover Points :** $\{(x_3, 0.5), (x_7, 0.5)\}$
- **Alpha Cut$_{0.2}$:** $\{(x_2, 0.2), (x_3, 0.5), (x_4, 1), (x_5, 1), (x_6, 1), (x_7, 0.5), (x_8, 0.2)\}$
- **Strong Alpha Cut$_{0.2}^{+}$:** $\{(x_3, 0.5), (x_4, 1), (x_5, 1), (x_6, 1), (x_7, 0.5)\}$
- **Boundary :** $\{(x_2, 0.2), (x_3, 0.5), (x_7, 0.5), (x_8, 0.2)\}$
- **Bandwidth :** $|x_7 - x_3|$
- **Normality :** True

Ex. 3.3.18 : For the fuzzy set $\underline{A}$ = { $(x_1, 0), (x_2, 0.3), (x_3, 0.4), (x_4, 0.5), (x_5, 0.8), (x_6, 1), (x_7, 1), (x_8, 1), (x_9, 1), (x_{10}, 0.7), (x_{11}, 0.5)$, $(x_{12}, 0.1), (x_{13}, 0)$ }, find following: Support, Core, Crossover points, Alpha cut for $\alpha = 0.3$, Strong Alpha cut for $\alpha = 0.4$, Boundary, Normality

☑ **Soln. :**

- **Support** = $\{(x_2, 0.3), (x_3, 0.4), (x_4, 0.5), (x_5, 0.8), (x_6, 1), (x_7, 1), (x_8, 1), (x_9, 1), (x_{10}, 0.7), (x_{11}, 0.5), (x_{12}, 0.1)\}$
- **Core** = $\{(x_6, 1), (x_7, 1), (x_8, 1), (x_9, 1), (x_{12}, 0.1)\}$
- **Crossover** = $\{(x_4, 0.5), (x_{11}, 0.5)\}$
- **Alpha cut for $\alpha = 0.3$** = $\{(x_2, 0.3), (x_3, 0.4), (x_4, 0.5), (x_5, 0.8), (x_6, 1), (x_7, 1), (x_8, 1), (x_9, 1), (x_{10}, 0.7), (x_{11}, 0.5)\}$
- **Strong Alpha cut for $\alpha = 0.4$** = $\{(x_4, 0.5), (x_5, 0.8), (x_6, 1), (x_7, 1), (x_8, 1), (x_9, 1), (x_{10}, 0.7), (x_{11}, 0.5)\}$
- **Boundary** = $\{(x_2, 0.3), (x_3, 0.4), (x_4, 0.5), (x_5, 0.8), (x_{10}, 0.7), (x_{11}, 0.5)$, $(x_{12}, 0.1)\}$
- **Normality** = True

UEx. 3.3.19

MU - Q. 1(d) May 19, Q. 1(e) Dec. 19, 5 Marks

Let us consider the discrete Fuzzy set,

$\underline{A} = \left\{\frac{1}{a} + \frac{0.9}{b} + \frac{0.6}{c} + \frac{0.3}{d} + \frac{0.01}{e} + \frac{0}{f}\right\}$ using Zadeh's otation, defined on universe X = {a, b, c, d, e, f}. Compute/infer the α cut for: (a) $\alpha = 0.9$ (b) $\alpha = 0.3$.

☑ **Soln. :**

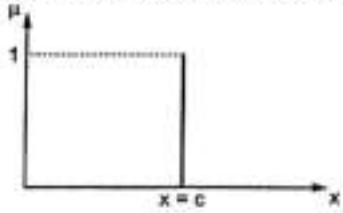
- α cut for $\alpha = 0.9 = \left\{\frac{1}{a} + \frac{0.9}{b}\right\}$
- α cut for $\alpha = 0.3 = \left\{\frac{1}{a} + \frac{0.9}{b} + \frac{0.6}{c} + \frac{0.3}{d}\right\}$

3.4 FUZZY MEMBERSHIP FUCNTIONS

- Fuzzy membership function is used to convert the crisp input provided to the fuzzy inference system.

- Fuzzy logic itself is not fuzzy, rather it deals with the fuzziness in the data. And this fuzziness in the data is best described by the fuzzy membership function.
- Fuzzy inference system is the core part of any fuzzy logic system. Fuzzification is the first step in Fuzzy Inference System.
- Formally, a membership function for a fuzzy set A on the universe of discourse X is defined as μ_A: $X \rightarrow [0, 1]$, where each element of X is mapped to a value between 0 and 1. This value, called membership value or degree of membership, quantifies the grade of membership of the element in X to the fuzzy set A. Here, X is the universal set and A is the fuzzy set derived from X.
- Fuzzy membership function is the graphical way of visualizing degree of membership of any value in given fuzzy set. In the graph, X axis represents the universe of discourse and Y axis represents the degree of membership in the range [0, 1].
- In following discussion, we will see various fuzzy membership functions. These functions are mathematically very simple.
- Fuzzy logic is meant to deal with the fuzziness, so use of complex membership function would not add much precision in final output.

1. **Singleton Membership Function :** Singleton membership function assigns membership value 1 to particular value of x, and assigns value 0 to rest of all. It is represented by impulse function as shown.



(IC20)**Fig. 3.4.1 : Singleton Membership Function**

Mathematically it is formulated as,

$$\mu_A(x) = \begin{cases} 1, & \text{if } x = c \\ 0, & \text{otherwise} \end{cases}$$

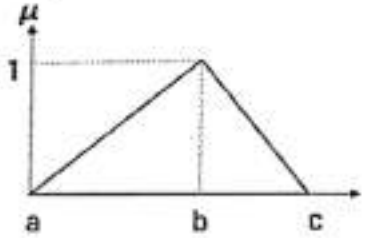
2. **Triangular Membership Function :** This is one of the most widely accepted and used membership function (MF) in fuzzy controller design. The triangle which fuzzifies the input can be defined by three parameters a, b and c, where a and c defines the base and b defines the height of the triangle.

Trivial Case : Here, in the diagram, X axis represents the input from the process (such as air conditioner, washing machine, etc.) and Y axis represents corresponding fuzzy value.

If input x = b, then it is having full membership in the given set. So, **$\mu(x) = 1$, if $x = b$.**

And if input is less than a or greater than b, then it does belong to fuzzy set at all, and its membership value will be 0.

$\mu(x) = 0$, if $x < a$ or $x > c$

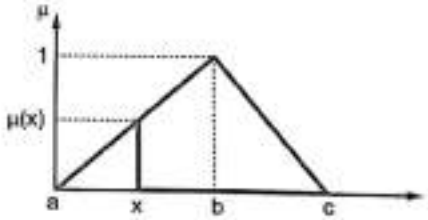


(IC21)**Fig. 3.4.2**

x is between a and b : If x is between a and b, as shown in the Fig. 3.4.3, its membership value varies from 0 to 1. If it is near a, its membership value is close to 0, and if x is near to b, its membership value gets close to 1.

We can compute the fuzzy value of x using similar triangle rule,

$$\mu(x) = \frac{(x-a)}{(b-a)}, \; a \le x \le b$$

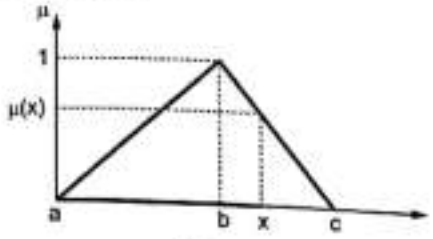


(IC22)**Fig. 3.4.3**

x is between b and c : If x is between b and c, as shown in the Fig. 3.4.4, its membership value varies from 0 to 1. If it is near b, its membership value is close to 1, and if x is near to c, its membership value gets close to 0.

We can compute the fuzzy value of x using similar triangle rule,

$$\mu(x) = \frac{(c-x)}{(c-b)}, b \le x \le c$$



(1C23)**Fig. 3.4.4**

Combine all together : We can combine all above scenario in single equation as,

$$\mu_{eriangle}(x; a, b, c) = \begin{cases} 0 & , x \le a \\ \frac{(x-a)}{(b-a)} & , a \le x \le b \\ \frac{(c-x)}{(c-b)} & , b \le x \le c \\ 0 & , c \le x \end{cases}$$

i.e. $\mu_{triangle}(x; a, b, c) = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right)$

Ex. 3.4.1 : Determine $\mu_{triangle}$ (x = 8; a = 2, b = 6, c = 10)

☑ **Soln. :**

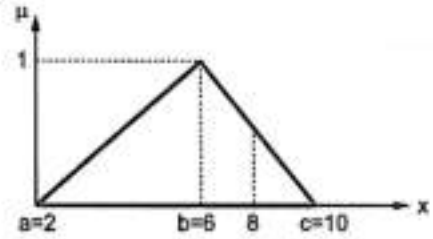


Fig. Ex. 3.4.1

$$\mu_{triangle}(x; a, b, c) = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right)$$

$\therefore$ $\mu_{triangle}$ (x = 8; a = 2, b = 6, c = 10)

$$= \max\left(\min\left(\frac{8-2}{6-2}, \frac{10-8}{10-6}\right), 0\right)$$

$$= \max\left(\min\left(\frac{3}{2}, \frac{1}{2}\right), 0\right) = \frac{1}{2}$$

3. **Trapezoidal Membership Function :** Trapezoidal membership function is defined by four parameters: a, b, c and d. Span b to c represents the highest membership value that element can take. And if x is between (a, b) or (c, d), then it will have membership value between 0 and 1.

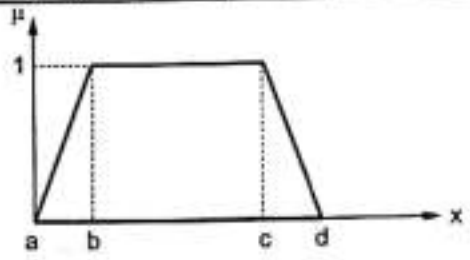


Fig. 3.4.5

We can apply the triangle MF if elements is in between a to b or c to d.

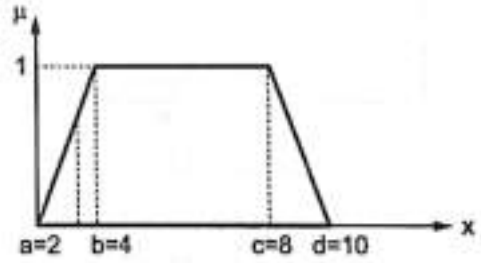
It is quite obvious to combine all together as,

$$\mu_{trapezoidal}(x; a, b, c, d) = \begin{cases} 0 & , x \le a \\ \frac{x-a}{b-a} & , a \le x \le b \\ 1 & , b \le x \le c \\ \frac{d-x}{d-c} & , c \le x \le d \\ 0 & , d \le x \end{cases}$$

$$= \max\left(\min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right), 0\right)$$

Ex. 3.4.2 : Determine $\mu_{trapezoidal}$ (x = 3.5; a = 2, b = 4, c = 8, d = 10).

☑ **Soln. :**



(1C24)**Fig. Ex. 3.4.2**

$$\mu_{trapezoidal}(x; a, b, c, d) = \max\left(\min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right), 0\right)$$

$\therefore$ $\mu_{trapezoidal}$ (x = 3.5; a = 2, b = 4. C = 8, d = 10)

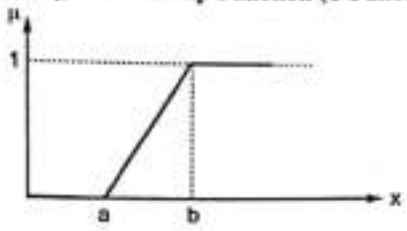
$$= \max\left(\min\left(\frac{3.5-2}{4-2}, 1, \frac{10-3.5}{10-8}\right), 0\right)$$

$$= \max\left(\min\left(\frac{3}{4}, 1, \frac{13}{4}\right), 0\right)$$

$$= \max(\min(0.75, 1, 3.25), 0)$$

$$= \max(0.75, 0) = 0.75$$

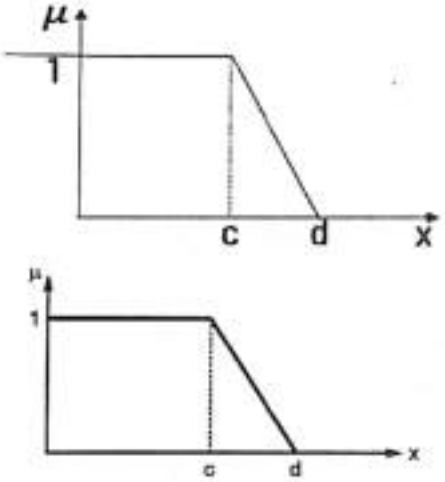
4. **Increasing Membership Function (T Function) :**



(1C25)**Fig. 3.4.6**

$$\mu_T(x; a, b) = \begin{cases} 0 & , x \le a \\ \frac{x-a}{b-a} & , a \le x \le b \\ 1 & , x \ge b \end{cases}$$

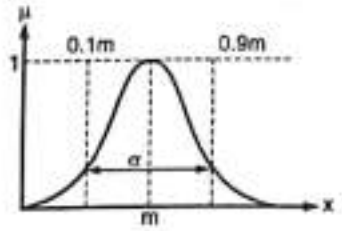
5. **Decreasing Membership Function (L Function)**



(1C26)**Fig. 3.4.7**

$$\mu_L(x; c, d) = \begin{cases} 1 & , x \le c \\ \frac{d-x}{d-c} & , c \le x \le d \\ 0 & , x \ge d \end{cases}$$

6. **Gaussian Membership Function :** A Gaussian MF is specified by two parameters {m, σ} and can be defined as follows:



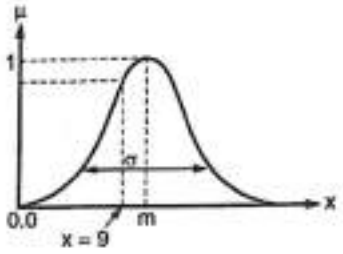
(1C27)**Fig. 3.4.8**

$$\mu_{gaussian}(x; m, \sigma) = e^{\frac{1}{2}\left(\frac{x-m}{\sigma}\right)^2}$$

In this function, m represents the mean / center of the gaussian curve and σ represents the spread of the curve. This is more natural way of representing the data distribution, but due to mathematical complexity it is not much used for fuzzification.

Ex. 3.4.3 : Determine $\mu_{gaussian}(x = 9; m = 10, \sigma = 3)$.

☑ **Soln. :**

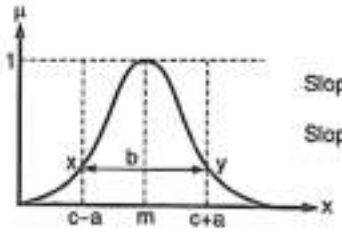


(1C28)**Fig. Ex. 3.4.3**

$$\mu_{gaussian}(x; m, \sigma) = e^{\frac{1}{2}\left(\frac{x-m}{\sigma}\right)^2}$$

$$\mu_{gaussian}(9; 10, 3) = e^{\frac{1}{2}\left(\frac{9-10}{3}\right)^2} = 0.9459$$

7. **Generalized Bell Shaped Function :** It is also called Cauchy MF. A generalized bell MF is specified by three parameters {a, b, c} and can be defined as follows:



Slope at $x = \frac{b}{2a}$

Slope at $y = -\frac{b}{2a}$

(1C29)**Fig. 3.4.9**

$$\mu_{bell}(x; a, b, c) = \frac{1}{1 + \left|\frac{x-c}{a}\right|^{2b}}$$

It is called generalized MF, because by changing the parameters a, b and c, we can produce a family of different membership functions.

The function $\mu(X) = 1/(1 + x^2)$ can be modelled by setting a = b = 1 and c = 0. Similarly, we can produce other shapes/functions by setting appropriate a, b and c.

8. **Sigmoid Membership Function :** Sigmoid functions are widely used in classification task in machine learning. Specifically, it is used in logistic regression and neural network, where it suppresses the input and maps it between 0 and 1. It is controlled by parameters a and c, where a controls the slope at the crossover point x = c.

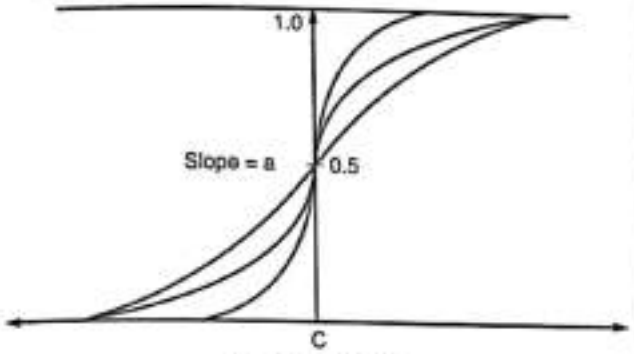


Fig. 3.4.10

$$\mu_{sigmoid}(x; a, c) = \frac{1}{1 + e^{-a(x-c)}}$$

3.5 FUZZY RELATION

- Fuzzy relation defines the mapping of variables from one fuzzy set to another. Like crisp relation, we can also define the relation over fuzzy sets.
- Let $\underline{A}$ be a fuzzy set on universe X and $\underline{B}$ be a fuzzy set on universe Y, then the Cartesian product between fuzzy sets $\underline{A}$ and $\underline{B}$ will result in a fuzzy relation $\underline{R}$ which is contained with the full Cartesian product space or it is subset of cartesian product of fuzzy subsets. Formally, we can define fuzzy relation as,

$$\underline{R} = \underline{A} \times \underline{B} \text{ and } \underline{R} \subset (X \times Y)$$

where the relation $\underline{R}$ has membership function, $\mu_{\underline{R}}(x, y) = \mu_{\underline{A} \times \underline{B}}(x, y) = \min(\mu_{\underline{A}}(x), \mu_{\underline{B}}(y))$

- A binary fuzzy relation $\underline{R}$ (X, Y) is called **bipartite graph** if $X \neq Y$.
- A binary fuzzy relation $\underline{R}$ (X, Y) is called **directed graph** or **digraph** if X = Y. , which is denoted as

$$\underline{R}(X, X) = \underline{R}(X^2)$$

- Let $\underline{A} = \{a_1, a_2, \ldots, a_n\}$ and $\underline{B} = \{b_1, b_2, \ldots, b_m\}$, then fuzzy relation between $\underline{A}$ and $\underline{B}$ is described by the **fuzzy relation matrix** as,

$$\begin{bmatrix} \mu_{R(a_1, b_1)} & \mu_{R(a_1, b_2)} & \cdot & \cdot & \mu_{R(a_1, b_m)} \\ \mu_{R(a_2, b_1)} & \mu_{R(a_2, b_2)} & \cdot & \cdot & \mu_{R(a_2, b_m)} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \mu_{R(a_n, b_1)} & \mu_{R(a_n, b_2)} & \cdot & \cdot & \mu_{R(a_n, b_m)} \end{bmatrix}$$

- We can also consider fuzzy relation as a mapping from the cartesian space (X, Y) to the interval [0, 1]. The strength of this mapping is represented by the membership function of the relation for every tuple $\mu_{R(x, y)}$

Ex. 3.5.1 : Given $\underline{A} = \{(a_1, 0.2), (a_2, 0.7), (a_3, 0.4)\}$ and $\underline{B} = \{(b_1, 0.5), (b_2, 0.6)\}$, find the relation over $\underline{A} \times \underline{B}$.

☑ **Soln. :**

$$\underline{R} = \underline{A} \times \underline{B} = \begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \end{array} \begin{array}{c} \begin{array}{cc} b_1 & b_2 \end{array} \\ \begin{bmatrix} 0.2 & 0.2 \\ 0.5 & 0.6 \\ 0.4 & 0.4 \end{bmatrix} \end{array}$$

3.5.1 Operations on Fuzzy Relation

- Just like crisp relations, following operations are possible on fuzzy relations as well.
- Fuzzy relation describes the interaction between variables in fuzzy controller.
- Four types of operations can be performed on fuzzy relation.
  1. Union
  2. Intersection
  3. Projection
  4. Cylindrical Extension

Let $\underline{R}$ and $\underline{S}$ be fuzzy relations on the cartesian space $X \times Y$.

1. **Union :** The union of $\underline{R}$ and $\underline{S}$ is defined as:

$$\forall (x, y) \in X \times Y : \mu_{\underline{R} \cup \underline{S}}(x, y) = \min(\mu_{\underline{R}}(x, y), \mu_{\underline{S}}(x, y))$$

2. **Intersection :** The intersection of $\underline{R}$ and $\underline{S}$ is defined as:

$$\forall (x, y) \in X \times Y : \mu_{\underline{R} \cup \underline{S}}(x, y) = \min(\mu_{\underline{R}}(x, y), \mu_{\underline{S}}(x, y))$$

3. **Projection**

Let $\underline{R} = \{((x, y), \mu_{\underline{R}}(x, y)) \mid (x, y) \in X \times Y)\}$ be a fuzzy relation.

The projection of $\underline{R}(x, y)$ on X denoted by R_1 is given by:

$$R_1 = \{(x, \max_y \mu_{\underline{R}}(x, y)) \mid (x, y) \in X \times Y\}$$

i.e. select the maximum value from each row.

The projection of $\underline{R}(x, y)$ on Y denoted by R_2 is given by:

$$R_2 = \{(y, \max_x \mu_{\underline{R}}(x, y)) \mid (x, y) \in X \times Y\}$$

i.e. select the maximum value from each column.

The projection relation brings a ternary relation back to a binary relation, or a binary relation to a fuzzy set, or a fuzzy set to a single crisp value.

4. Cylindrical Extension

- The cylindrical extension on X × Y of a fuzzy set A of X is a fuzzy relation *cyl*A whose membership function is equal to

$$cylA(x, y) = A(x), \forall x \in X, \forall y \in Y$$

- Cylindrical extension from X-projection means filling all the columns of the related matrix by the X -projection.
- Similarly cylindrical extension from Y projection means filling all the rows of the relational matrix by the Y -projection.

Ex. 3.5.2 : Given two relations $\underline{R}$ and $\underline{S}$

$\underline{R}$ =

	y_1	y_2	y_3	y_4
x_1	0.8	0.1	0.1	0.7
x_2	0.0	0.8	0.0	0.0
x_3	0.9	1.0	0.7	0.8

$\underline{S}$ =

	y_1	y_2	y_3	y_4
x_1	0.4	0.0	0.9	0.6
x_2	0.9	0.4	0.5	0.7
x_3	0.3	0.0	0.8	0.5

Find (a) $\underline{R} \cup \underline{S}$ (b) $\underline{R} \cap \underline{S}$ (c) Projection of $\underline{R}$ on X (d) Projection of $\underline{S}$ on Y (e) Cylindrical extension of $\underline{R}$

☑ **Soln. :**

$\underline{R} \cup \underline{S}$ =

	y_1	y_2	y_3	y_4
x_1	0.8	0.1	0.9	0.7
x_2	0.9	0.8	0.5	0.7
x_3	0.9	1.0	0.8	0.8

$\underline{R} \cap \underline{S}$ =

	y_1	y_2	y_3	y_4
x_1	0.4	0.0	0.1	0.6
x_2	0.0	0.4	0.0	0.0
x_3	0.3	0.0	0.7	0.5

Projection of $\underline{R}$ on X = $\{(0.8, x_1), (0.8, x_2), (1.0, x_3)\}$

$$= \frac{0.8}{x_1} + \frac{0.8}{x_2} + \frac{1.0}{x_3}$$

Projection of $\underline{S}$ on Y = $\{(0.9, y_1), (0.4, y_2), (0.9, y_3), (0.7, y_4)\}$

$$= \frac{0.9}{y_1} + \frac{0.4}{y_2} + \frac{0.9}{y_3} + \frac{0.7}{y_4}$$

cyl$\underline{R}$ on X =

	y_1	y_2	y_3	y_4
x_1	0.8	0.8	0.8	0.8
x_2	0.8	0.8	0.8	0.8
x_3	1.0	1.0	1.0	1.0

cyl$\underline{R}$ on Y =

	y_1	y_2	y_3	y_4
x_1	0.9	1.0	0.7	0.8
x_2	0.9	1.0	0.7	0.8
x_3	0.9	1.0	0.7	0.8

3.5.2 Composition of Fuzzy Relations

UQ. With suitable example, explain max-min composition and max-product composition. MU - Dec 17, 10 Marks

UQ. Explain max-min and max-product composition with example. MU - May 18, 5 Marks

- Composition operation is used to combine two fuzzy relations in different product spaces.
- The two composition operations commonly used are:
  1. Max – min Composition 2. Max- product Composition

1. Max – min Composition

- Let $\underline{R}$ be a fuzzy relation defined on X × Y.
- Let $\underline{S}$ be a fuzzy relation defined on Y × Z.

Then the max – min composition of two fuzzy relations $\underline{R}$ and $\underline{S}$ is denoted by $\underline{R} \cdot \underline{S}$ and defined as

$$\underline{R} \cdot \underline{S} = \left\{ \left[(x, z), \max_{y \in Y} \left(\min \left(\mu_{\underline{R}}(x, y), \mu_{\underline{S}}(y, z) \right) \right) \right] \middle| x \in X, y \in Y, z \in Z \right\}$$

OR

$$\mu_{\underline{R} \cdot \underline{S}}(x, z) = \max_{y \in Y} \left(\min \left(\mu_{\underline{R}}(x, y), \mu_{\underline{S}}(y, z) \right) \right)$$

2. Max – product Composition

- Let $\underline{R}$ be a fuzzy relation defined on $X \times Y$.
- Let $\underline{S}$ be a fuzzy relation defined on $Y \times Z$.
- Then the max – product composition of two fuzzy relations $\underline{R}$ and $\underline{S}$ is denoted by $\underline{R} \cdot \underline{S}$ and defined as

$$\underline{R} \cdot \underline{S} = \left\{ \left[(x, z), \max_{y \in Y} \left(\mu_{\underline{R}}(x, y) \cdot \mu_{\underline{S}}(y, z) \right) \right] \middle| x \in X, y \in Y, z \in Z \right\}$$

OR

$$\mu_{\underline{R} \cdot \underline{S}}(x, z) = \max_{y \in Y} \left(\left(\mu_{\underline{R}}(x, y) \cdot \mu_{\underline{S}}(y, z) \right) \right)$$

UEx. 3.5.3 (MU - Q. 4(B), Dec. 18, 10 Marks)

Two fuzzy relations are given by :

$\underline{R}$ =		b_1	b_2	b_3
	a_1	0.4	0.5	0.0
	a_2	0.2	0.8	0.2

$\underline{S}$ =		c_1	c_2
	b_1	0.2	0.7
	b_2	0.3	0.8
	b_3	1.0	0.0

Find T as a max-min composition and max-product composition between the fuzzy relations.

☑ **Soln. :**

1. Max – min Composition

$$T = \mu_{\underline{R} \cdot \underline{S}}(a, c) = \max_{b \in B} \left(\min \left(\mu_{\underline{R}}(a, b), \mu_{\underline{S}}(b, c) \right) \right)$$

$\mu_{\underline{R} \cdot \underline{S}}(a_1, c_1)$ = max (min (0.4, 0.2), min (0.5, 0.3), min (0.0, 1.0)) = max (0.2, 0.3, 0.0) = 0.3

$\mu_{\underline{R} \cdot \underline{S}}(a_1, c_2)$ = max (min (0.4, 0.7), min (0.5, 0.8), min (0.0, 0.0)) = max (0.4, 0.5, 0.0) = 0.5

$\mu_{\underline{R} \cdot \underline{S}}(a_2, c_1)$ = max (min (0.2, 0.2), min (0.8, 0.3), min (0.2, 1.0)) = max (0.2, 0.3, 0.2) = 0.3

$\mu_{\underline{R} \cdot \underline{S}}(a_2, c_2)$ = max (min 0.2, 0.7), min (0.8, 0.8), min (0.2, 0.0)) = max (0.2, 0.8, 0.0) = 0.8

T =		c_1	c_2
	a_1	0.3	0.5
	a_2	0.3	0.8

2. Max – product Composition

$$T = \mu_{\underline{R} \cdot \underline{S}}(a, c) = \max_{b \in B} \left(\mu_{\underline{R}}(a, b) \cdot \mu_{\underline{S}}(b, c) \right)$$

$\mu_{R \cdot S}(a_1, c_1) = \max((0.4 \times 0.2), (0.5 \times 0.3), (0.0 \times 1.0)) = \max(0.08, 0.15, 0.0) = 0.15$

$\mu_{R \cdot S}(a_1, c_2) = \max((0.4 \times 0.7), (0.5 \times 0.8), (0.0 \times 0.0)) = \max(0.28, 0.40, 0.0) = 0.40$

$\mu_{R \cdot S}(a_2, c_1) = \max((0.2 \times 0.2), (0.8 \times 0.3), (0.2 \times 1.0)) = \max(0.04, 0.24, 0.2) = 0.24$

$\mu_{R \cdot S}(a_2, c_2) = \max((0.2 \times 0.7), (0.8 \times 0.8), (0.2 \times 0.0)) = \max(0.14, 0.64, 0.0) = 0.64$

T =		c_1	c_2
	a_1	0.15	0.40
	a_2	0.24	0.64

UEx. 3.5.4 (MU - Q. 6(b), May. 19, Dec. 19, 10 Marks)

Let R and S be two fuzzy relations defined as:

R =		y_1	y_2	y_3
	x_1	0.0	0.2	0.8
	x_2	0.3	0.6	1.0

S =		z_1	z_2	z_3
	y_1	0.3	0.7	1.0
	y_2	0.5	1.0	0.6
	y_3	1.0	0.2	0.0

(a) Compute / Infer the result of R · S as a max-min composition.

(b) Compute / Infer the result of R · S as a max-product composition.

☑ **Soln. :**

1. Max – min Composition

$$\mu_{R \cdot S}(x, z) = \max_{y \in Y} \left(\min \left(\mu_R(x, y), \mu_S(y, z) \right) \right)$$

$\mu_{R \cdot S}(x_1, z_1) = \max(\min(0.0, 0.3), \min(0.2, 0.5), \min(0.8, 1.0)) = \max(0.0, 0.2, 0.8) = 0.8$

$\mu_{R \cdot S}(x_1, z_2) = \max(\min(0.0, 0.7), \min(0.2, 1.0), \min(0.8, 0.2)) = \max(0.0, 0.2, 0.2) = 0.2$

$\mu_{R \cdot S}(x_1, z_3) = \max(\min(0.0, 1.0), \min(0.2, 0.6), \min(0.8, 0.0)) = \max(0.0, 0.2, 0.0) = 0.2$

$\mu_{R \cdot S}(x_2, z_1) = \max(\min(0.3, 0.3), \min(0.6, 0.5), \min(1.0, 1.0)) = \max(0.3, 0.5, 1.0) = 1.0$

$\mu_{R \cdot S}(x_2, z_2) = \max(\min(0.3, 0.7), \min(0.6, 1.0), \min(1.0, 0.2)) = \max(0.3, 0.6, 1.2) = 0.6$

$\mu_{R \cdot S}(x_2, z_3) = \max(\min(0.3, 1.0), \min(0.6, 0.6), \min(1.0, 0.0)) = \max(0.3, 0.6, 0.0) = 0.6$

$T = \mu_{R \cdot S}(x, z)$		z_1	z_2	z_3
	x_1	0.8	0.2	0.2
	x_2	1.0	0.6	0.6

2. Max – product Composition

$$\mu_{R \cdot S}(x, z) = \max_{y \in Y} \left(\left(\mu_R(x, y) \cdot \mu_S(y, z) \right) \right)$$

$\mu_{R \cdot S}(x_1, z_1) = \max((0.0 \times 0.3), (0.2 \times 0.5), (0.8 \times 1.0)) = \max(0.0, 0.10, 0.8) = 0.8$

$\mu_{R \cdot S}(x_1, z_2) = \max((0.0 \times 0.7), (0.2 \times 1.0), (0.8 \times 0.2)) = \max(0.0, 0.2, 0.16) = 0.2$

$\mu_{R \cdot S}(x_1, z_3) = \max((0.0 \times 1.0), (0.2 \times 0.6), (0.8 \times 0.0)) = \max(0.0, 0.12, 0.0) = 0.12$

$\mu_{R \cdot S}(x_2, z_1) = \max((0.3 \times 0.3), (0.6 \times 0.5), (1.0 \times 1.0)) = \max(0.09, 0.30, 1.0) = 1.0$

$\mu_{R \cdot S}(x_2, z_2) = \max((0.3 \times 0.7), (0.6 \times 1.0), (1.0 \times 0.2)) = \max(0.21, 0.6, 0.2) = 0.6$

$\mu_{R \cdot S}(x_2, z_3) = \max((0.3 \times 1.0), (0.6 \times 0.6), (1.0 \times 0.0)) = \max(0.3, 0.36, 0.0) = 0.36$

$T = \mu_{R \cdot S}(x, z)$		z_1	z_2	z_3
	x_1	0.8	0.2	0.12
	x_2	1.0	0.6	0.36

3.5.3 Properties of Fuzzy Relations

Let $\underline{R}$, $\underline{S}$ and $\underline{T}$ be fuzzy relations defined on the universe $X \times Y$. Then, the properties of fuzzy relations are as below:

1.	**Commutativity**	$\underline{R} \cup \underline{S} = \underline{S} \cup \underline{R}$ $\underline{R} \cap \underline{S} = \underline{S} \cap \underline{R}$
2.	**Associativity**	$\underline{R} \cup (\underline{S} \cup \underline{T}) = (\underline{R} \cup \underline{S}) \cup \underline{T}$ $\underline{R} \cap (\underline{S} \cap \underline{T}) = (\underline{R} \cap \underline{S}) \cap \underline{T}$
3.	**Distributivity**	$\underline{R} \cup (\underline{S} \cap \underline{T}) = (\underline{R} \cup \underline{S}) \cap (\underline{R} \cup \underline{T})$ $\underline{R} \cap (\underline{S} \cup \underline{T}) = (\underline{R} \cap \underline{S}) \cup (\underline{R} \cap \underline{T})$
4.	**Idempotency**	$\underline{R} \cup \underline{R} = \underline{R}$ $\underline{R} \cap \underline{R} = \underline{R}$
5.	**Identity**	$\underline{R} \cup \phi = \underline{R}$ $\underline{R} \cap \phi = \phi$ $\underline{R} \cup X = X$ $\underline{R} \cap X = \underline{R}$ where ϕ is null relation (null matrix) and X is complete relation (unit matrix)
6.	**Involution**	$\overline{\overline{R}} = R$
7.	**De-Morgan's Law**	$\overline{\underline{R} \cap \underline{S}} = \underline{R} \cup \underline{S}$ $\overline{\underline{R} \cup \underline{S}} = \underline{R} \cap \underline{S}$
8.	**Law of excluded middle and law of contradiction are not satisfied**	$\underline{R} \cup \overline{\underline{R}} \neq X$ $\underline{R} \cap \overline{\underline{R}} \neq \phi$

3.5.4 Fuzzy Extension Principle

- The extension principle is one of the most basic ideas **in fuzzy set theory.**
- It **provides a general method for extending crisp mathematical concepts to address fuzzy quantities, such as real algebraic operations on fuzzy numbers.**
- **These operations are computationally effective generalizations of interval analysis.**
- **Suppose *f* is a function from X to Y and $\underline{R}$ is a fuzzy set on X defined as :**

$$\underline{R} = \left\{ \frac{\mu_{\underline{R}}(x_1)}{x_1} + \frac{\mu_{\underline{R}}(x_2)}{x_2} + \ldots + \frac{\mu_{\underline{R}}(x_n)}{x_n} \right\}$$

- **Then the extension principle states that the image of fuzzy set $\underline{R}$ under the mapping function *f(.)* can be expressed as a new fuzzy set $\underline{S}$, where**

$$\underline{S} = f(\underline{R}) = \left\{ \frac{\mu_{\underline{R}}(x_1)}{y_1} + \frac{\mu_{\underline{R}}(x_2)}{y_2} + \ldots + \frac{\mu_{\underline{R}}(x_n)}{y_n} \right\}$$

where $y_i = f(x_i)$; $i = 1, 2, \ldots, n$

- **Example :**

Let $\underline{R} = \left\{ \frac{0.1}{-2} + \frac{0.4}{-1} + \frac{0.8}{0} + \frac{0.9}{1} + \frac{0.3}{2} \right\}$

and $f(x) = x^2 + x - 3$

Upon applying the fuzzy extension principle, we have,

$$\underline{S} = \left\{ \frac{0.1}{-1} + \frac{0.4}{-3} + \frac{0.8}{-3} + \frac{0.9}{-1} + \frac{0.3}{3} \right\}$$

$$= \left\{ \frac{0.4 \cup 0.8}{-3} + \frac{0.1 \cup 0.9}{-1} + \frac{0.3}{3} \right\}$$

$$= \left\{ \frac{0.8}{-3} + \frac{0.9}{-1} + \frac{0.3}{3} \right\}$$

where $\cup$ indicates max operation.

3.6 DEFUZZIFICATION METHODS

Fuzzification

- It is the method of transforming a crisp quantity(set) into a fuzzy quantity(set).
- This can be achieved by identifying the various known crisp and deterministic quantities as completely nondeterministic and quite uncertain in nature.
- This uncertainty may have emerged because of vagueness and imprecision which then lead the variables to be represented by a membership function, as they can be fuzzy in nature.
- For example, when I say the temperature is 45° Celsius the viewer converts the crisp input value into a linguistic variable like favorable temperature for the human body, hot or cold.

Defuzzification

- It is the inverse of fuzzification. The former one was used to convert the crisp results into fuzzy results but here the mapping is done to convert the fuzzy results into crisp results.
- This process is capable of generating a non-fuzzy control action which illustrates the possibility distribution of an inferred fuzzy control action.

Defuzzification process can also be treated as the rounding off process, where fuzzy set having a group of membership values on the unit interval reduced to a single scalar quantity.

Difference between Fuzzification and Defuzzification

Sr. No.	Key	Fuzzification	Defuzzification
1	Definition	Fuzzification is the process of transforming a crisp set to a fuzzy set.	Defuzzification is the process of reducing a fuzzy set into a crisp set or converting a fuzzy member into a crisp member.
2	Purpose	Fuzzification converts a precise data into imprecise data.	Defuzzification converts an imprecise data into precise data.
3	Example	Voltmeter.	Stepper motor, D/A converter.
4	Methods used	Inference, Rank ordering, Angular fuzzy sets, Neural network.	Maximum membership principle, Centroid method, Weighted average method, Centre of sums.
5	Complexity	Fuzzification is easy.	Defuzzification is quite complex to implement.
6	Approach	Fuzzification uses if-then rules to fuzzify the crisp value.	Defuzzification uses centre of gravity methods to get centroid of sets.

Defuzzification Methods

UQ. Define Defuzzification. Discuss any two methods of assigning membership value.
MU - Q. 1(b), May 17, 5 Marks

UQ. Write a note on: Defuzzification.
MU - Q. 6(d), Dec. 17, 5 Marks

UQ. Explain different defuzzification techniques.
MU - Q. 1(d), Dec. 18, 5 Marks

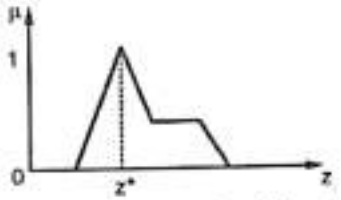
UQ. What are defuzzification methods in fuzzy logic? Explain any one with example.
MU - Q. 1(b), May 19, 5 Marks

- **Defuzzification is the process of producing a quantifiable result in Crisp logic, given fuzzy sets and corresponding membership degrees.**
- **It is the process that maps a fuzzy set to a crisp set. It is typically needed in fuzzy control systems.**
- **These will have a number of rules that transform a number of variables into a fuzzy result, that is, the result is described in terms of membership in fuzzy sets.**
- **Defuzzification is the conversion of a fuzzy quantity to a precise quantity, just as fuzzification is the conversion of a precise quantity to a fuzzy quantity μ.**
- Among the many methods of defuzzification, we shall be studying seven methods for defuzzifying fuzzy output functions.
- They are as follows :
  1. Max-membership principle
  2. Centroid method
  3. Weighted average method
  4. Mean-max membership
  5. Centre of sums
  6. Centre of largest area
  7. First of maxima, last of maxima.

1. Max membership principle

Also known as the **height method.** It is limited to peaked output function. This method is given by algebraic expression :

$$\mu_{\underline{C}}(x^*) \geq \mu_{\underline{C}}(x) \text{ for all } z \in Z$$



(IC31)Fig. 3.6.1 : Max-member ship principle

Where Z* is defuzzified value, as shown in Fig. 3.6.1.

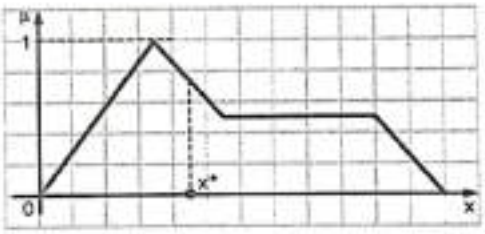
2. Centroid method

This method is also known as centre of gravity or centre of area or centre of mass method, as it obtains the centre of area (x^*) occupied by the fuzzy set.

It is given by, $x^* = \dfrac{\int \mu(x) \cdot x \cdot dx}{\int \mu(x) \cdot dx}$

for a continuous membership function and

$$x^* = \frac{\sum_{i=1}^{n} x_i \mu(x_i)}{\sum_{i=1}^{n} \mu(x_i)}$$; for a discrete membership function.



(IC32)Fig. 3.6.2 : Centroid method

Here n is the number of elements in the sample, x_i are the elements, and $\mu(x_i)$ is its membership function. The method is illustrated in the Fig. 3.6.2.

3. Weighted average method

This method is frequently used in fuzzy applications since it is computationally efficient method. But it is **restricted** to **symmetrical output membership functions.**

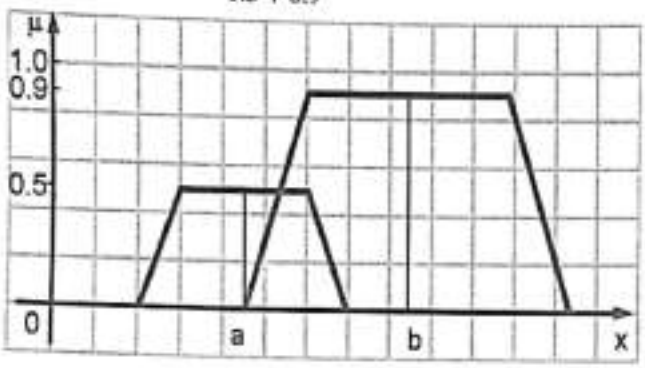
It is given by algebraic expression.

$$x^* = \frac{\sum \mu_c(\bar{x}) \cdot \bar{x}}{\sum \mu_c(\bar{x})}$$

where $\bar{x}$ is the centroid of each symmetric membership function.

The weighted average method is formed by weighing each membership function in the output by its respective maxium membership value. For example, the two functions shown in the Fig. 3.6.3 would result for defuzzified value

$$x^* = \frac{a\,(0.5) + b\,(0.9)}{0.5 + 0.9}$$



(1C33)**Fig. 3.6.3 : Weighted average method**

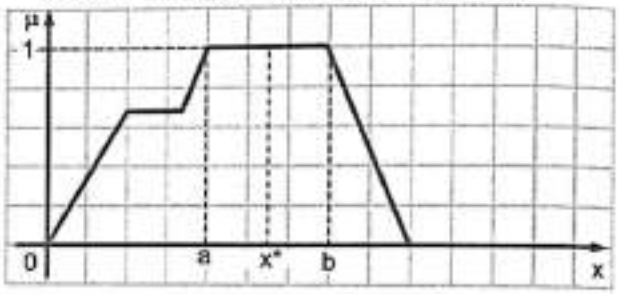
4. Mean-max membership

This method is also called as 'middle-of-maxima' and is closely related to the first method, except that the locations of maximum membership can be nonunique.

The defuzzified output is given by,

$$x^* = \frac{a + b}{2}$$

where a and b are as shown in the Fig. 3.6.4.

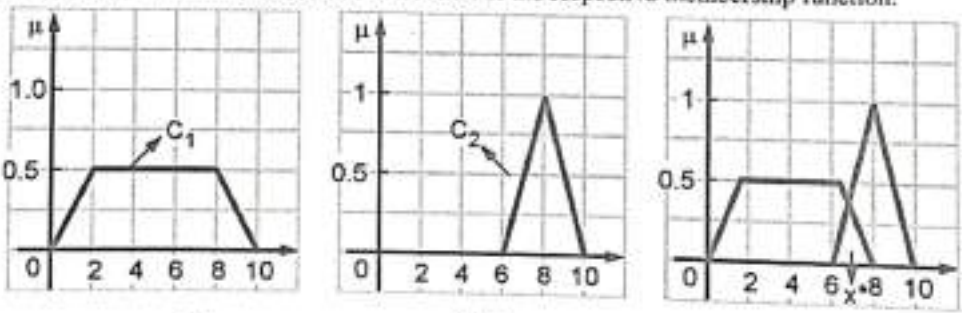


(1C34)**Fig. 3.6.4 : Mean-max membership**

5. Centre of sums : This method is faster than many other defuzzification method. And this method is not restricted to symmetric membership functions. This process involves the algebraic sum of individual output fuzzy sets, say C_1 and C_2. The defuzzified value x^* is given as,

$$x^* = \frac{\sum_{k=1}^{n} \mu_{c_k}(\bar{x}) \int_z \bar{x}\, dz}{\sum_{k=1}^{n} \mu_{c_k}(x) \int_x dx}$$

where the symbol $\bar{x}$ is the distance to the centroid of each of the respective membership function.



(a) (b) (c)

(1C35)**Fig. 3.6.5 : Centre of sums**

(a) **First membership function** $\rightarrow C_1$

(b) **Second membership function** $\rightarrow C_2$

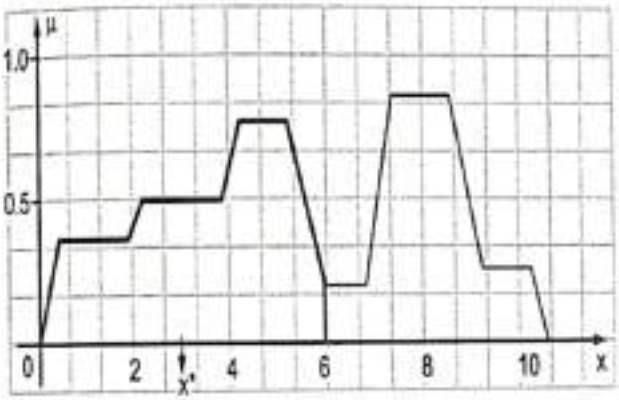
(c) **Defuzzification step.**

6. Centre of largest area

This method is used when the output consists of at least two convex fuzzy subsets which are not overlapping. The centre of gravity of the convex fuzzy subregion having the largest are gives the defuzzified value x^*. This value is given by,

$$x^* = \frac{\int \mu_{Cj}(x) \cdot x \cdot dx}{\int \mu_{Cj}(x) \cdot dx}$$

where C_j is the convex subregion that has the largest area making up C_j. Fig. 3.6.6 illustrates the centre of largest area.



(IC38)**Fig. 3.6.6 : Centre of largest area method (outlined with bold lines)**

7. First (or last) of maxima :

This method uses the overall output of all individual output fuzzy sets C_k to determine the smallest value of the domain.

The expression for x^* is as follows :

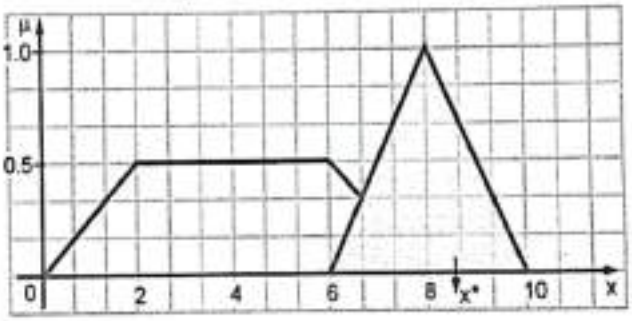
(a) First determine the largest height in the union [denoted as $hgt(C_k)$]

(b) The first of the maxima is found,

$$x^* = \underset{x}{\text{int}} \left[x \mid \mu_{Ck}(x) = hgt(C_k)\right]$$

(c) An alternative to this method is called the 'last of maxima' and it is given as,

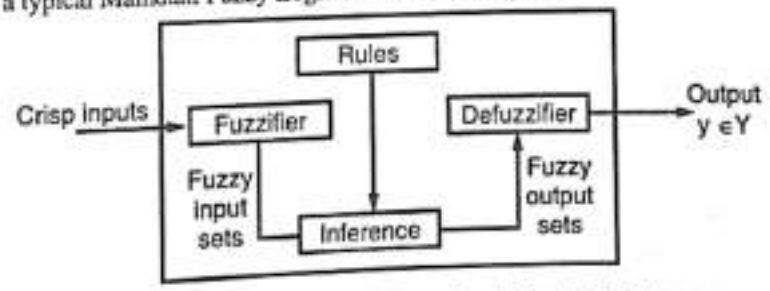
$$x^* = \underset{x}{\sup} \left[x \mid \mu_{Ck}(x) = hgt(C_k)\right]$$



(IC37)**Fig. 3.6.7 : First of max (and last of max) method**

3.7 ARCHITECTURE OF MAMDANI TYPE FUZZY CONTROL SYSTEM

- The most commonly used fuzzy inference technique is the so-called Mamdani method.
- In 1975, Professor Ebrahim Mamdani of London University built one of the first fuzzy systems to control a steam engine and boiler combination.
- The original goal was to control a steam engine & boiler combination by a set of linguistic control rules obtained from experienced human operators.
- Fig. 3.7.1 illustrates a typical Mamdani Fuzzy Logic Control (FLC) System.

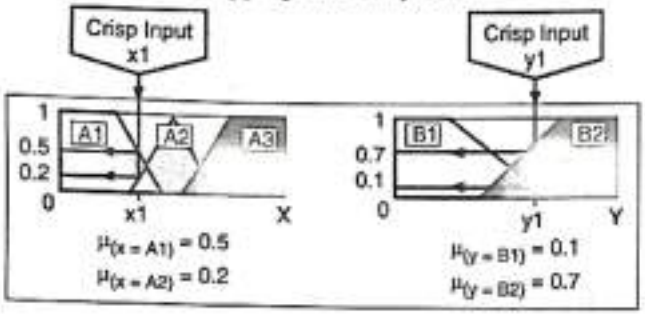


(IC39)**Fig. 3.7.1 : Mamdani Fuzzy Logic Control System**

- The Mamdani-style fuzzy inference process is performed in four steps:
  1. Fuzzification of the input variables,
  2. Rule evaluation,
  3. Aggregation of the rule outputs, and finally
  4. De-fuzzification

1. Fuzzification

- **Take the crisp inputs, x1 and y1**
- **Determine the degree to which these inputs belong to each of the appropriate fuzzy sets.**



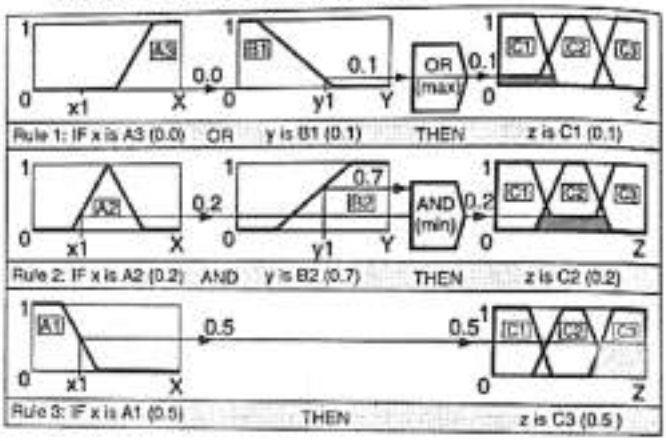
(IC39)**Fig. 3.7.2 : Crisp Inputs**

2. Rule Evaluation

- Take the fuzzified inputs, $\mu_{(x=A1)} = 0.5$, $\mu_{(x=A2)} = 0.2$, $\mu_{(y=B1)} = 0.1$ and $\mu_{(y=B2)} = 0.7$
- Apply them to the antecedents of the fuzzy rules.
- If a given fuzzy rule has multiple antecedents, the fuzzy operator (AND or OR) is used to obtain a single number that represents the result of the antecedent evaluation.
- This number (the truth value) is then applied to the consequent membership function.
- To evaluate the disjunction of the rule antecedents, we use the OR fuzzy operation. Typically, fuzzy expert systems make use of the classical fuzzy operation union:

  $\mu_{A \cup B}(x) = \max [\mu_A(x), \mu_B(x)]$
- Similarly, in order to evaluate the conjunction of the rule antecedents, we apply the AND fuzzy operation intersection:

  $\mu_{A \cap B}(x) = \min [\mu_A(x), \mu_B(x)]$
- The rule base comprises a knowledge of the application domain and the attendant control goals.
- It consists of a "data base" and a "linguistic (fuzzy) control rule base":
  (a) the data base provides necessary definitions which are used to define linguistic control rules and fuzzy data manipulation in a FLC.
  (b) the rule base characterizes the control goals and the control policy of the domain experts by means of a set of linguistic control rules.
- The fuzzy inference engine is the kernel of a FLC; it has the capability of simulating human decision-making based of fuzzy concepts and of inferring fuzzy control actions employing fuzzy implication and the rules of inference in fuzzy logic.

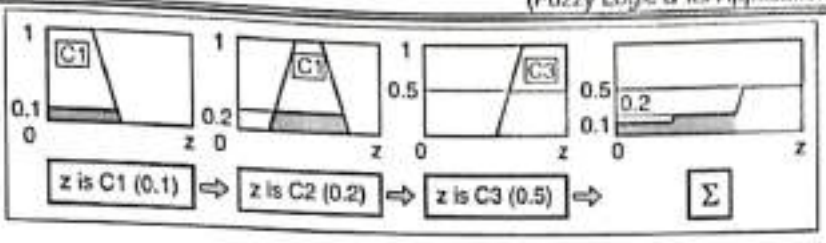


(IC40)**Fig. 3.7.3 : Rule Evaluation**

3. Aggregation of the Rules Output

- Aggregation is the process of unification of the outputs of all rules.
- We take the membership functions of all rule consequents previously clipped or scaled and combine them into a single fuzzy set.
- There are several defuzzification methods, but probably the most popular one is the centroid technique.
- It finds the point where a vertical line would slice the aggregate set into two equal masses.
- Mathematically this centre of gravity (COG) can be expressed as:

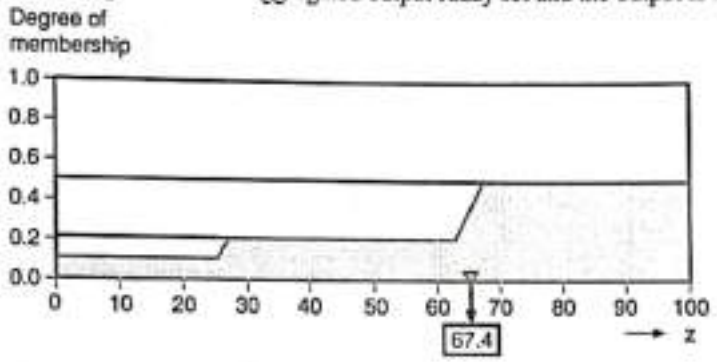
$$x^* = \frac{\int \mu(x) \cdot x \cdot dx}{\int \mu(x) \cdot dx}$$



(1C41)**Fig. 3.7.4 :Aggregation of the Rules Output**

4. Defuzzification

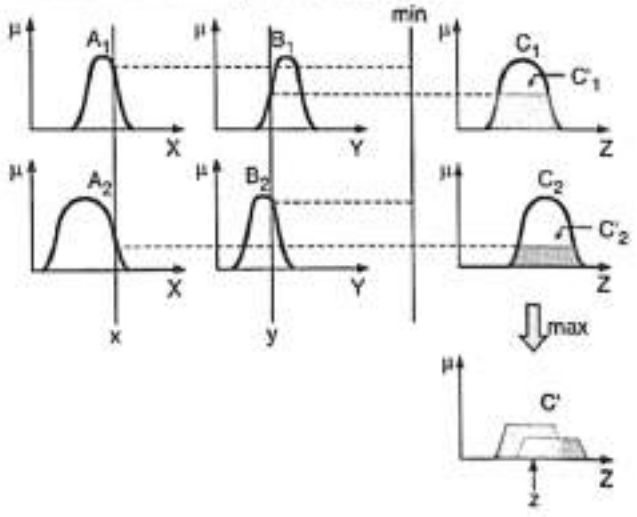
- Fuzziness helps us to evaluate the rules, but the final output of a fuzzy system has to be a crisp number.
- The input for the defuzzification process is the aggregated output fuzzy set and the output is a single number.



(1C42)**Fig. 3.7.5 : Defuzzification**

$$x^* = \frac{(0+10+20)\times 0.1 + (30+40+50+60)\times 0.2 + (70+80+90+100)\times 0.5}{0.1+0.1+0.1+0.2+0.2+0.2+0.2+0.5+0.5+0.5+0.5} = 67.4$$

- Mamdani model also supports min-max and max-product composition.



(1C43)**Fig. 3.7.6 : Mamdani Fuzzy Inference System using Min-max Composition**

3.8 DESIGN OF FUZZY CONTROLLERS

Steps in Designing a Fuzzy Logic Control System

1. Identify the system input variables, their ranges, and membership functions.
2. Identify the output variables, their ranges, and membership functions.
3. Identify the rules that describe the relations of the inputs to the outputs.
4. Determine the de-fuzzifier method of combining fuzzy rules into system outputs.

UEx. 3.8.1 MU - May-17, May-18, May-19, Dec.-18, Dec.-19, 10 Marks

Using Mamdani Fuzzy model, design a fuzzy logic controller to determine the wash time of domestic washing machine. Assume that the inputs are dirt and grease on clothes. Use three descriptors for each input variables and five descriptors for output variables. Derive necessary membership function and required fuzzy rules for the application.

Soln. :

Step 1 : Identify input and output variables and their descriptors.

Given that the inputs are dirt and grease on clothes. We assume that they are measured in percentage (%).

The output is the wash time measured in minutes.

We use three descriptors for the input variables.

Descriptor for dirt : Small Dirt (SD), Medium Dirt (MD), and Large Dirt (LD).

Descriptor for grease : No Grease (NG), Medium Grease (MG), and Large Grease (LG).

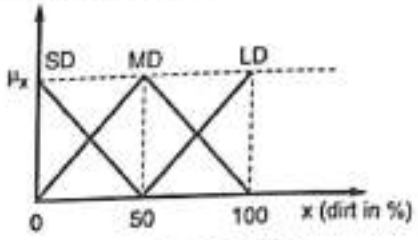
We use five descriptors for the output variable.

Descriptor for wash time : Very Short (VS), Short (S), Medium (M), Large (L), and Very Large (VL)

Step 2 : Define membership functions for each of the input and output variables.

We use triangular membership functions.

1. Membership Function for Dirt



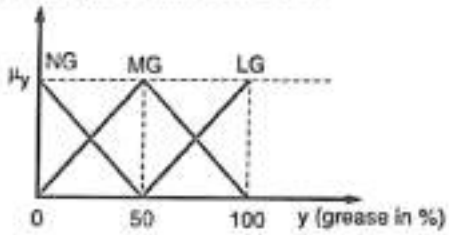
(1C44)**Fig. Ex. 3.8.1**

$$\mu_{SD}(x) = \frac{50 - x}{50}, \quad 0 \le x \le 50$$

$$\mu_{MD}(x) = \begin{cases} \frac{x}{50}, & 0 \le x \le 50 \\ \frac{100 - x}{50}, & 50 < x \le 100 \end{cases}$$

$$\mu_{LD}(x) = \frac{x - 50}{50}, \quad 50 \le x \le 100$$

2. Membership Function for Grease



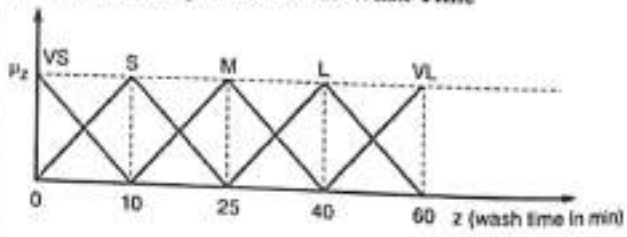
(1C45)**Fig. Ex. 3.8.1(a)**

$$\mu_{NG}(y) = \frac{50 - y}{50}, \quad 0 \le y \le 50$$

$$\mu_{MG}(y) = \begin{cases} \frac{y}{50}, & 0 \le y \le 50 \\ \frac{100 - y}{50}, & 50 < y \le 100 \end{cases}$$

$$\mu_{LG}(y) = \frac{y - 50}{50}, \quad 50 \le y \le 100$$

3. Membership Function for Wash Time



(1C46)**Fig. Ex. 3.8.1(b)**

$$\mu_{VS}(z) = \frac{10-z}{10},\ 0 \le z \le 10$$

$$\mu_{S}(z) = \begin{cases} \frac{z}{10},\ 0 \le z \le 10 \\ \frac{25-z}{15},\ 10 < z \le 25 \end{cases}$$

$$\mu_{M}(z) = \begin{cases} \frac{z-10}{15},\ 10 \le z \le 25 \\ \frac{40-z}{15},\ 25 < z \le 40 \end{cases}$$

$$\mu_{L}(z) = \begin{cases} \frac{z-25}{15},\ 25 \le z \le 40 \\ \frac{60-z}{20},\ 40 < z \le 60 \end{cases}$$

$$\mu_{VL}(z) = \frac{z-40}{20},\ 40 \le z \le 60$$

▶ **Step 3 : Form a rule base.**

	NG	MG	LG
SD	VS	M	L
MD	S	M	L
LD	M	L	VL

There are total nine rules defined in above matrix. For example, **"If dirt is small and medium grease then wash time is medium."** Similarly, we can also define all rules using if – then.

▶ **Step 4 : Rule Evaluation**

Assume that dirt = 80% and grease = 90%

Dirt = 80% maps to the following two MFs of "dirt" variable.

$$\mu_{MD}(x) = \frac{100-x}{50} \quad \text{and} \quad \mu_{LD}(x) = \frac{x-50}{50}$$

Evaluate $\mu_{MD}(x)$ and $\mu_{LD}(x)$ for x = 80.

$$\mu_{MD}(80) = \frac{100-80}{50} = \frac{2}{5} \qquad ...(1)$$

$$\mu_{LD}(80) = \frac{80-50}{50} = \frac{3}{5} \qquad ...(2)$$

Similarly, grease = 90% maps to the following two MFs of "grease" variable.

$$\mu_{MG}(y) = \frac{100-y}{50} \quad \text{and} \quad \mu_{LG}(y) = \frac{y-50}{50}$$

Evaluate $\mu_{MG}(y)$ and $\mu_{LG}(y)$ for y = 90.

$$\mu_{MG}(90) = \frac{100-90}{50} = \frac{1}{5} \qquad ...(3)$$

$$\mu_{LG}(90) = \frac{90-50}{50} = \frac{4}{5} \qquad ...(4)$$

The above four equations represent the following four rules that we need to evaluate.

1. Dirt is **medium** and grease is **medium**.
2. Dirt is **medium** and grease is **large**.
3. Dirt is **large** and grease is **medium**.
4. Dirt is **large** and grease is **large**.

The antecedent part of each of the above rule is connected by **and** operator. So, we use **min** operator to evaluate the strength of each rule.

Strength of rule 1: $S_1 = \min(\mu_{MD}(80), \mu_{MG}(90)) = \min\left(\frac{2}{5}, \frac{1}{5}\right) = \frac{1}{5}$

Strength of rule 2: $S_2 = \min(\mu_{MD}(80), \mu_{LG}(90)) = \min\left(\frac{2}{5}, \frac{4}{5}\right) = \frac{2}{5}$

Strength of rule 3: $S_3 = \min(\mu_{LD}(80), \mu_{MG}(90)) = \min\left(\frac{3}{5}, \frac{1}{5}\right) = \frac{1}{5}$

Strength of rule 4: $S_4 = \min(\mu_{MD}(80), \mu_{LG}(90)) = \min\left(\frac{3}{5}, \frac{4}{5}\right) = \frac{3}{5}$

▶ **Step 5 : Defuzzification**

We use **"Mean of max"** technique for defuzzification. So, we find the rule with maximum strength.

$$\text{Max}(S_1, S_2, S_3, S_4) = \text{Max}\left(\frac{1}{5}, \frac{2}{5}, \frac{1}{5}, \frac{3}{5}\right) = \frac{3}{5}$$

This corresponds to rule 4, i.e. Dirt is **large** and grease is **large**.

From rule base, If Dirt is **large** and grease is **large**, then wash time is **very large**.

This corresponds to the output MF $\mu_{VL}(z) = \frac{z-40}{20}$

$$\mu_{VL}(z) = \frac{3}{5}$$

$$\therefore \frac{3}{5} = \frac{z-40}{20} \Rightarrow z = \frac{3 \times 20}{5} + 40 = 52 \text{ minutes}$$

$$\therefore \quad \mathbf{z = 52 \text{ minutes}}$$

Ex. 3.8.2 : Using Mamdani Fuzzy model, design a fuzzy logic controller to regulate the temperature of a domestic shower. Assume that the input is the position of mixer tap. Use five descriptors for both input and output variable. Derive necessary membership function and required fuzzy rules for the application.

☑ **Soln. :**

▶ **Step 1: Identify input and output variables and their descriptors.**

Given that the input is the position of mixer tap. We assume that the position of mixer tap is measured in degrees (0° to 180°). 0° indicates tap is closed and 180° indicates tap is fully opened.

The output is the temperature of water measured in °C as per the position of mixer tap.

We use five descriptors for each input and output variables.

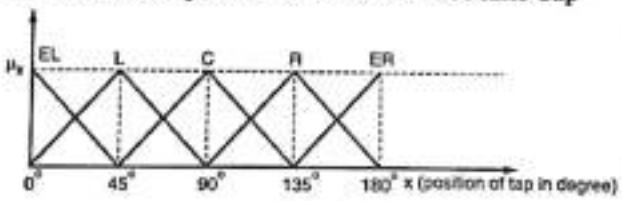
Descriptor for position of mixer tap : Extreme Left (EL), Left (L), Centre (C), Right ®, Extreme Right (ER)

Descriptor for temperature : Very Cold (VC), Cold (C), Medium (M), Hot (H), very Hot (VH).

▶ **Step 2 : Define membership functions for input and output variables.**

We use triangular membership functions.

1. Membership Function for Position of Mixer Tap



(1C47)**Fig. Ex. 3.8.2**

$$\mu_{EL}(x) = \frac{45 - x}{45},\ 0 \le x \le 45$$

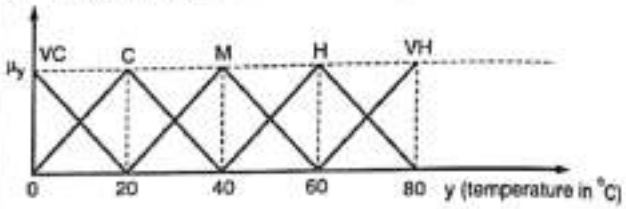
$$\mu_L(x) = \begin{cases} \frac{x}{45},\ 0 \le x \le 45 \\ \frac{90 - x}{45},\ 45 < x \le 90 \end{cases}$$

$$\mu_C(x) = \begin{cases} \frac{x - 45}{45},\ 45 \le x \le 90 \\ \frac{135 - x}{45},\ 90 < x \le 135 \end{cases}$$

$$\mu_R(x) = \begin{cases} \frac{x - 90}{45},\ 90 \le x \le 135 \\ \frac{180 - x}{45},\ 135 < x \le 180 \end{cases}$$

$$\mu_{ER}(x) = \frac{x - 135}{45},\ 135 \le x \le 180$$

2. Membership Function for Temperature



(1C48)**Fig. Ex. 3.8.2(a)**

$$\mu_{VC}(y) = \frac{20 - y}{20},\ 0 \le y \le 20$$

$$\mu_C(y) = \begin{cases} \frac{y}{20},\ 0 \le y \le 20 \\ \frac{40 - y}{20},\ 20 < y \le 40 \end{cases}$$

$$\mu_M(y) = \begin{cases} \frac{y - 20}{20},\ 20 \le y \le 40 \\ \frac{60 - y}{20},\ 40 < y \le 60 \end{cases}$$

$$\mu_H(y) = \begin{cases} \frac{y - 40}{20},\ 40 \le y \le 60 \\ \frac{80 - y}{20},\ 60 < y \le 80 \end{cases}$$

$$\mu_{VH}(y) = \frac{y - 60}{20},\ 60 \le y \le 80$$

▶ **Step 3 : Form a rule base.**

Input Position of Mixer Tap	Output Temperature of Water
EL	VC
L	C
C	M
R	H
ER	VH

There are total five rules defined in above table. For example, "**If position of mixer tap is left then temperature of water is cold.**" Similarly, we can also define all rules using if – then.

Step 4 : Rule Evaluation

Assume position of mixer tap is 80°.

This value x = 80° maps to the following two MFs.

$$\mu_L(x) = \frac{90-x}{45} \quad \text{and} \quad \mu_C(x) = \frac{x-45}{45}$$

Evaluate $\mu_L(x)$ and $\mu_C(x)$ for x = 80.

$$\mu_L(80) = \frac{90-80}{45} = \frac{2}{9}$$

$$\mu_C(80) = \frac{80-45}{45} = \frac{7}{9}$$

Step 5 : Defuzzification

We use **"Mean of max"** technique for defuzzification. So, we find the rule with maximum strength.

$$\text{Max}(\mu_L(x), \mu_C(x)) = \max\left(\frac{2}{9}, \frac{7}{9}\right) = \frac{7}{9}$$

This corresponds to rule 3 in rule base, i.e. If the position of mixer tap is left, temperature of water is medium.

We have following two equations for medium water temperature.

$$\mu_M(y) = \frac{y-20}{20} \quad \text{and} \quad \mu_M(y) = \frac{60-y}{20}$$

The strength of rule is $\frac{7}{9}$

$$\mu_M(y) = \frac{y-20}{20}$$

$$\therefore \frac{7}{9} = \frac{y-20}{20} \Rightarrow y = \frac{7\times 20}{9} + 20 = 35.55$$

$$\mu_M(y) = \frac{60-y}{20}$$

$$\therefore \frac{7}{9} = \frac{60-y}{20} \Rightarrow y = 60 - \frac{7\times 20}{9} = 44.44$$

To find the final defuzzified value, we now take the average of $\mu_M(y)$.

$$\therefore \quad y^* = \frac{35.55 + 44.44}{2} = 39.995$$

$$\therefore \quad y^* = 40°C$$

Ex. 3.8.3 : Using Mamdani fuzzy model, design a fuzzy logic controller to the feed amount of purifier for the water purification plant. Assume input as water temperature and grade of water. Use three descriptors for both input and output variables. Derive necessary membership function and required fuzzy rules for the application.

☑ **Soln. :**

Step 1 : Identify input and output variables and their descriptors.

Given that the input variables are water temperature and grade of water.

We assume that the temperature of water measured in °C and the grade of water is measured in percentage (%).

Also, we assume that amount of purifier is measured in grams.

We use three descriptors for each input and output variables.

Descriptor for water temperature : Cold (C), Medium (M), High (H)

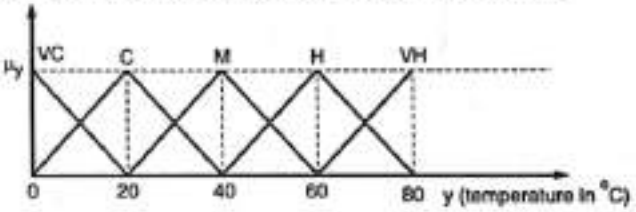
Descriptor for grade : Low (L), Medium (M), High (H)

Descriptor for amount of purifier : Small (S), Medium (M), Large (L)

Step 2 : Define membership functions for input and output variables.

We use triangular membership functions.

1. Membership Function for Water Temperature



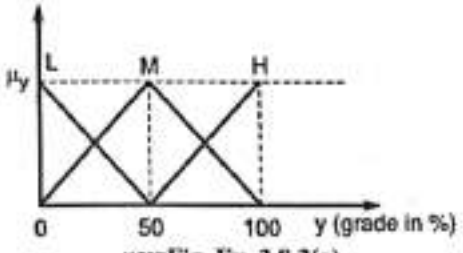
(1C49)**Fig. Ex. 3.8.3**

$$\mu_c(x) = \frac{50-x}{50}, \quad 0 \le x \le 50$$

$$\mu_M(x) = \begin{cases} \frac{x}{50}, & 0 \le x \le 50 \\ \frac{100-x}{50}, & 50 < x \le 100 \end{cases}$$

$$\mu_H(x) = \frac{x-50}{50}, \quad 50 \le x \le 100$$

2. Membership Function for Grade of Water



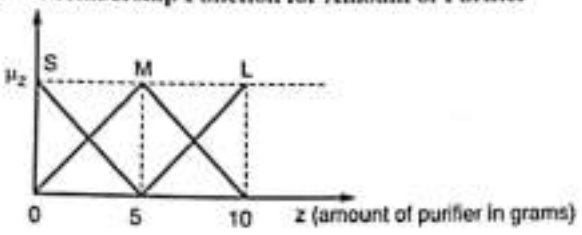
(1C50)**Fig. Ex. 3.8.3(a)**

$$\mu_L(y) = \frac{50-y}{50}, 0 \le y \le 50$$

$$\mu_M(y) = \begin{cases} \frac{y}{50}, & 0 \le y \le 50 \\ \frac{100-y}{50}, & 50 < y \le 100 \end{cases}$$

$$\mu_H(y) = \frac{y-50}{50}, 50 \le y \le 100$$

3. **Membership Function for Amount of Purifier**



(new)**Fig. Ex. 3.8.3(b)**

$$\mu_S(z) = \frac{5-z}{5}, 0 \le z \le 5$$

$$\mu_M(z) = \begin{cases} \frac{z}{5}, & 0 \le z \le 5 \\ \frac{10-z}{5}, & 5 < z \le 10 \end{cases}$$

$$\mu_L(z) = \frac{z-5}{5}, 5 \le z \le 10$$

▶ **Step 3 : Form a rule base.**

		Grade of Water		
		L	**M**	**H**
Temperature of water	**C**	L	M	S
	M	L	M	M
	H	M	S	S

The above matrix represents total nine rules. For example, **"If temperature is cold and grade of water is low then amount of purifier required is large"**. Similarly, we can define all the rules using if - then.

▶ **Step 4 : Rule Evaluation**

Assume water temperature = 5°C and grade of water = 30%.

Water temperature = 5°C maps to the following two MFs of "water temperature" variable.

$$\mu_c(x) = \frac{50-x}{50} \text{ and } \mu_M(x) = \frac{x}{50}$$

Evaluate $\mu_c(x)$ and $\mu_M(x)$ for x = 5.

$$\mu_c(5) = \frac{50-5}{50} = \frac{9}{10} \quad \text{...(1)}$$

$$\mu_M(5) = \frac{5}{50} = \frac{1}{10} \quad \text{...(2)}$$

Similarly, grade = 30% maps to the following two MFs of "grade" variable.

$$\mu_L(y) = \frac{50-y}{50} \text{ and } \mu_M(y) = \frac{y}{50}$$

Evaluate $\mu_L(y)$ and $\mu_M(y)$ for y = 30.

$$\mu_L(30) = \frac{50-30}{50} = \frac{2}{5} \quad \text{...(3)}$$

$$\mu_M(30) = \frac{30}{50} = \frac{3}{5} \quad \text{...(4)}$$

The above four equations represent the following four rules that we need to evaluate.

1. If temperature is **cold** and grade is **low**.
2. If temperature is **cold** and grade is **medium**.
3. If temperature is **medium** and grade is **low**.
4. If temperature is **medium** and grade is **medium**.

The antecedent part of each of the above rule is connected by **and** operator. So, we use **min** operator to evaluate the strength of each rule.

Strength of rule 1: $S_1 = \min(\mu_c(5), \mu_L(30))$

$$= \min\left(\frac{9}{10}, \frac{2}{5}\right) = \frac{2}{5}$$

Strength of rule 2: $S_2 = \min(\mu_c(5), \mu_M(30))$

$$= \min\left(\frac{9}{10}, \frac{3}{5}\right) = \frac{3}{5}$$

Strength of rule 3: $S_3 = \min(\mu_M(5), \mu_L(30))$

$$= \min\left(\frac{1}{10}, \frac{2}{5}\right) = \frac{1}{10}$$

Strength of rule 4: $S_4 = \min(\mu_M(5), \mu_M(30))$

$$= \min\left(\frac{1}{10}, \frac{3}{5}\right) = \frac{1}{10}$$

▶ **Step 5 : Defuzzification**

We use **"Mean of max"** technique for defuzzification.

So, we find the rule with maximum strength.

$$\text{Max}(S_1, S_2, S_3, S_4) = \text{Max}\left(\frac{2}{5}, \frac{3}{5}, \frac{1}{10}, \frac{1}{10}\right) = \frac{3}{5}$$

This corresponds to rule 2, i.e. Temperature is **cold** and grade is **medium**.

From rule base, If Temperature is **cold** and grade is **medium**, then amount of purifier is **medium**.

We have following two equations for medium water temperature.

$$\mu_M(z) = \frac{z}{5} \quad \text{and} \quad \mu_M(z) = \frac{10-z}{5}$$

The strength of rule is $\frac{3}{5}$.

$$\mu_M(z) = \frac{z}{5}$$

$$\therefore \quad \frac{3}{5} = \frac{z}{5} \Rightarrow z = 3$$

$$\mu_M(z) = \frac{10-z}{5}$$

$$\therefore \quad \frac{3}{5} = \frac{10-z}{5} \Rightarrow z = 7$$

To find the final defuzzified value, we now take the average of $\mu_M(z)$.

$$\therefore \quad z^* = \frac{3+7}{2} = 5$$

$$\therefore \quad z^* = 5 \text{ gms}$$

3.9 MULTIPLE CHOICE QUESTIONS

Q. 3.1 Which of the following logic is the form of Fuzzy logic?
(a) Two-valued logic (b) Crisp set logic
(c) Binary set logic (d) Many-valued logic
✓**Ans. : (d)**

Q. 3.2 The truth values of traditional set theory is _____ and that of fuzzy set is _____.
(a) Either 0 or 1, between 0 & 1
(b) Between 0 & 1, either 0 or 1
(c) Between 0 & 1, between 0 & 1
(d) Either 0 or 1, either 0 or 1 ✓**Ans. : (a)**

Q. 3.3 The room temperature is hot. Here the hot (use of linguistic variable is used) can be represented by _____.
(a) Fuzzy Set (b) Crisp Set
(c) Fuzzy & Crisp Set (d) Classical Set ✓**Ans. : (a)**

Q. 3.4 The values of the set membership is represented by _____.
(a) Discrete Set
(b) Degree of truth
(c) Probabilities
(d) Both Degree of truth & Probabilities ✓**Ans. : (b)**

Q. 3.5 Fuzzy logic is usually represented as _____.
(a) IF-THEN-ELSE rules
(b) IF-THEN rules
(c) Both IF-THEN-ELSE rules & IF-THEN rules
(d) Unconditional Statement ✓**Ans. : (b)**

Q. 3.6 How many output Fuzzy Logic produce?
(a) 2 (b) 3 (c) 4 (d) 5 ✓**Ans. : (a)**

Q. 3.7 In Membership function graph x-axis represent?
(a) Universe of discourse
(b) Degrees of membership in the [0, 1] interval
(c) Degrees of discourse
(d) Universe of membership ✓**Ans. : (a)**

Q. 3.8 Which of the following represents the values of set membership?
(a) Degree of truth (b) Probabilities
(c) Discrete set (d) Both a & b ✓**Ans. : (d)**

Q. 3.9 Which of the following fuzzy operator is not utilized in fuzzy set theory?
(a) AND (b) OR
(c) NOT (d) EX-OR ✓**Ans. : (d)**

Q. 3.10 Uncertainty can be represented by _____
(a) Entropy (b) Fuzzy logic
(c) Probability (d) All of the above ✓**Ans. : (d)**

Q. 3.11 Fuzzy logic is
(a) Used to respond to questions in a human like way
(b) A new programming language used to program animation
(c) The result of fuzzy thinking
(d) A term that indicates logical values greater than one
✓**Ans. : (a)**

Q. 3.12 What are the following sequence of steps taken in designing a fuzzy logic machine?
(a) Rule evaluation → Fuzzification → Defuzzification
(b) Fuzzification → Rule evaluation → Defuzzification
(c) Fuzzy sets → Defuzzification → Rule evaluation
(d) Defuzzification → Rule evaluation → Fuzzification
✓**Ans. : (b)**

Q. 3.13 The height h (A) of a fuzzy set A is defined as h (A) = sup A (x) where

(a) h (A) = 0 (b) h (A) < 0

(c) h (A) = 1 (d) h (A) < 1 ✓**Ans. : (c)**

Q. 3.14 Fuzzy logic uses

(a) Global variables (b) Linguistic variables

(c) Local variables (d) Approximate variables

✓**Ans. : (b)**

Q. 3.15 Consider a fuzzy set A defined on the interval X = [0, 10] of integers by the membership function $\mu_A(x) = x / (x+2)$. Then the α cut corresponding to α = 0.5 will be

(a) {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

(b) {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

(c) {2, 3, 4, 5, 6, 7, 8, 9, 10}

(d) {0, 1, 2, 3, 4, 5, 6, 7, 8, 9} ✓**Ans. : (c)**

Q. 3.16 If A and B are two fuzzy sets with membership functions:

$\mu_a(x)$ = {0.2,0.5,0.6,0.1,0.9}

$\mu_b(x)$ = {0.1,0.5,0.2,0.7,0.8}

then the value of $\mu_a \cap \mu_b$ will be

(a) {0.2,0.5,0.6,0.7,0.9}

(b) {0.2, 0.5,0.2, 0.1,0.8}

(c) {0.1, 0.5, 0.6, 0.1,0.8}

(d) {0.1, 0.5, 0.2, 0.1,0.8} ✓**Ans. : (d)**

Q. 3.17 A _____ point of a fuzzy set A is a point $x \in X$ at which $\mu_A(x) = 0.5$.

(a) Core (b) Support

(c) Cross over (d) Alpha – cut ✓**Ans. : (c)**

Q. 3.18 Let R and S be two fuzzy relations defined as follows. Then, the resulting relation, T, which relates elements of universe x to the elements of universe z using max-min composition is given by :

$$R = \begin{matrix} \\ x_1 \\ x_2 \end{matrix} \begin{matrix} \begin{matrix} y_1 & y_2 \end{matrix} \\ \begin{bmatrix} 0.6 & 0.4 \\ 0.7 & 0.3 \end{bmatrix} \end{matrix} \text{ and } S = \begin{matrix} \\ y_1 \\ y_2 \end{matrix} \begin{matrix} \begin{matrix} z_1 & z_2 & z_3 \end{matrix} \\ \begin{bmatrix} 0.8 & 0.5 & 0.1 \\ 0.0 & 0.6 & 0.4 \end{bmatrix} \end{matrix}$$

(1) $T = \begin{matrix} \\ x_1 \\ x_2 \end{matrix} \begin{matrix} \begin{matrix} z_1 & z_2 & z_3 \end{matrix} \\ \begin{bmatrix} 0.4 & 0.6 & 0.4 \\ 0.7 & 0.7 & 0.7 \end{bmatrix} \end{matrix}$

(2) $T = \begin{matrix} \\ x_1 \\ x_2 \end{matrix} \begin{matrix} \begin{matrix} z_1 & z_2 & z_3 \end{matrix} \\ \begin{bmatrix} 0.4 & 0.6 & 0.4 \\ 0.8 & 0.5 & 0.4 \end{bmatrix} \end{matrix}$

(3) $T = \begin{matrix} \\ x_1 \\ x_2 \end{matrix} \begin{matrix} \begin{matrix} z_1 & z_2 & z_3 \end{matrix} \\ \begin{bmatrix} 0.6 & 0.5 & 0.4 \\ 0.7 & 0.5 & 0.3 \end{bmatrix} \end{matrix}$

(4) $T = \begin{matrix} \\ x_1 \\ x_2 \end{matrix} \begin{matrix} \begin{matrix} z_1 & z_2 & z_3 \end{matrix} \\ \begin{bmatrix} 0.6 & 0.5 & 0.5 \\ 0.7 & 0.7 & 0.7 \end{bmatrix} \end{matrix}$

(a) 1 (b) 2 (c) 3 (d) 4 ✓**Ans. : (c)**

Q. 3.19 Let A and B be two fuzzy integers defined as:

A = {(1,0.3), (2,0.6), (3,1), (4,0.7), (5,0.2)}

B = {(10,0.5), (11,1), (12,0.5)}

Using fuzzy arithmetic operation given by

$\mu_{A+B}(z) = x \overset{\oplus}{y} = z \; (\mu_A(x) \otimes \mu_B(y))$

f (A + B) is _____ [Note : $\oplus$ = max, $\otimes$ = min]

(a) {(11,0.8), (13,1), (15,1)}

(b) {(11,0.3), (12,0.5), (13,1), (14,1), (15,1), (16,0.5), (17,0.2)}

(c) {(11,0.3), (12,0.5), (13,0.6), (14,1), (15,1), (16,0.5), (17,0.2)}

(d) {(11,0.3), (12,0.5), (13,0.6), (14,1), (15,0.7), (16,0.5), (17,0.2)} ✓**Ans. : (d)**

Q. 3.20 Support of a fuzzy set given below, within a universal set X is given as

$$A = \left\{ \frac{x_1}{0.2}, \frac{x_2}{0.15}, \frac{x_3}{0.9}, \frac{x_4}{0.95}, \frac{x_5}{0.15} \right\}$$

(A) $\left\{ \frac{x_1}{0.15}, \frac{x_2}{0.15}, \frac{x_3}{0.15}, \frac{x_4}{0.15}, \frac{x_5}{0.15} \right\}$

(B) $\left\{ \frac{x_1}{0.95}, \frac{x_2}{0.95}, \frac{x_3}{0.95}, \frac{x_4}{0.95}, \frac{x_5}{0.95} \right\}$

(C) $\{x_3, x_4\}$

(D) $(x_1, x_2, x_3, x_4, x_5)$

(a) A (b) B (c) C (d) D ✓**Ans. : (d)**

Descriptive Questions

Q.1 Define classical sets and fuzzy sets.

Q. 2 State the importance of fuzzy sets.

Q. 3 Discuss the operations of crisp sets.

Q. 4 List the properties of classical sets.

Q. 5 What are different properties of fuzzy sets?

Q. 6 Compare and contrast classical logic and fuzzy logic.

Q. 7 With suitable example explain max-min composition and max-product composition.

Q. 8 Let $X = \{x_1, x_2, x_3\}$, $y = \{y_1, y_2\}$, $Z = \{z_1, z_2, z_3\}$ Let R be the fuzzy relation

$$\begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \end{array} \begin{array}{c} \begin{array}{cc} y_1 & y_2 \end{array} \\ \begin{bmatrix} 0.5 & 0.1 \\ 0.2 & 0.9 \\ 0.8 & 0.6 \end{bmatrix} \end{array}$$

and S be a fuzzy relation

$$\begin{array}{c} \\ y_1 \\ y_2 \end{array} \begin{array}{c} \begin{array}{ccc} z_1 & z_2 & z_3 \end{array} \\ \begin{bmatrix} 0.6 & 0.4 & 0.7 \\ 0.5 & 0.8 & 0.9 \end{bmatrix} \end{array}$$

(a) Find R o S by max-min composition

(b) Find R o S by max-product composition

Q. 9 Define defuzzification. Discuss any two methods of assigning membership value.

Q. 10 Explain different defuzzification techniques.

Q. 11 Draw the block diagram of Fuzzy Inference System and brief about the inferencecomponent.Implement a fuzzy controller for the control of break-power of a train approaching station.Inputs are speed and distance of train from station and output is break-power. Use triangularmembership function. Consider two descriptors for inputs and three descriptors for output.Device a set of rules for control action and defuzzification. The implementation needs besupported by figures wherever possible. Design a fuzzy controller for a train with high speedand small distance.

Q. 12 Three fuzzy sets are given as follows :

$$P = \left\{\frac{0.1}{2} + \frac{0.3}{4} + \frac{0.7}{6} + \frac{0.4}{8} + \frac{0.2}{10}\right\};$$

$$Q = \left\{\frac{0.1}{0.1} + \frac{0.3}{0.2} + \frac{0.3}{0.3} + \frac{0.4}{0.4} + \frac{0.5}{0.5} + \frac{0.2}{0.6}\right\}$$

$$T = \left\{\frac{0.1}{0} + \frac{0.7}{0.5} + \frac{0.3}{1}\right\}$$

Find **(a)** $R = P \times Q$, **(b)** $S = Q \times T$, **(c)** max-min composition $M = R \circ S$ **(d)** max-product composition $M = R \circ S$.

Q. 13 State the advantages and disadvantages of fuzzy logic.

Q. 14 Define the fuzzy terminologies: Support, Core, Height, Alpha-cut, Crossover, Symmetry, Normality.

Q. 15 Design a fuzzy controller for maintaining the temperature of water in the tank at fixed level. Input variables are the cold water flow into the tank and steam flow into the tank. Assume five descriptors for each of the input variables and seven descriptors for the output variable. Device a set of rules for control action and defuzzification. The implementation needs be supported by figures wherever possible.

SELF-LEARNING TOPICS

I. Various Forms of Fuzzy Composition Operations

If A and B are two fuzzy sets and R is a fuzzy relation such that $B = A \cdot R$, then the various forms of fuzzy composition operations are as follow:

Max-Min Composition

$$\mu_B(y) = \max_{x \in X} \left\{\min\left[\mu_A(x), \mu_R(x, y)\right]\right\}$$

Max-Product Composition

$$\mu_B(y) = \max_{(x \in X)} \left\{\mu_A(x) \cdot \mu_R(x, y)\right\}$$

Min-Max Composition

$$\mu_B(y) = \min_{(x \in X)} \left\{\max\left[\mu_A(x), \mu_R(x, y)\right]\right\}$$

Max-Max Composition

$$\mu_B(y) = \max_{(x \in X)} \left\{\max\left[\mu_A(x), \mu_R(x, y)\right]\right\}$$

Min-Min Composition:

$$\mu_B(y) = \min_{(x \in X)} \left\{\min\left[\mu_A(x), \mu_R(x, y)\right]\right\}$$

Max-Average Composition

$$\mu_B(y) = \frac{1}{2}\left[\min_{(x \in X)} \left\{\mu_A(x) + \mu_R(x, y)\right\}\right]$$

Sum-Product Composition

$$\mu_B(y) = f\left\{\sum_{x \in X} \left[\mu_A(x) \cdot \mu_R(x, y)\right]\right\}$$

where f is a logistic function like that of step, sigmoid, or linear.

I. Fuzzy Inference System

Fuzzy Inference System (FIS) is the key unit of a fuzzy logic system having decision making as its primary work. It uses the "IF...THEN" rules along with connectors "OR" or "AND" for drawing essential decision rules.

Characteristics of Fuzzy Inference System

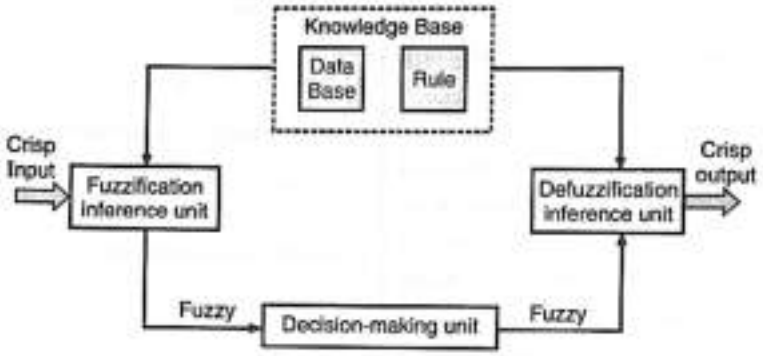
Following are some characteristics of FIS:

- The output from FIS is always a fuzzy set irrespective of its input which can be fuzzy or crisp.
- It is necessary to have fuzzy output when it is used as a controller.
- A defuzzification unit would be there with FIS to convert fuzzy variables into crisp variables.

Functional Blocks of FIS

The following five functional blocks will help you understand the construction of FIS :

- **Rule Base :** It contains fuzzy IF-THEN rules.
- **Database :** It defines the membership functions of fuzzy sets used in fuzzy rules.
- **Decision-making Unit :** It performs operation on rules.
- **Fuzzification Interface Unit :** It converts the crisp quantities into fuzzy quantities.
- **Defuzzification Interface Unit :** It converts the fuzzy quantities into crisp quantities. Following is a block diagram of fuzzy interference system.



(ICS2) Fig. 1

Working of FIS

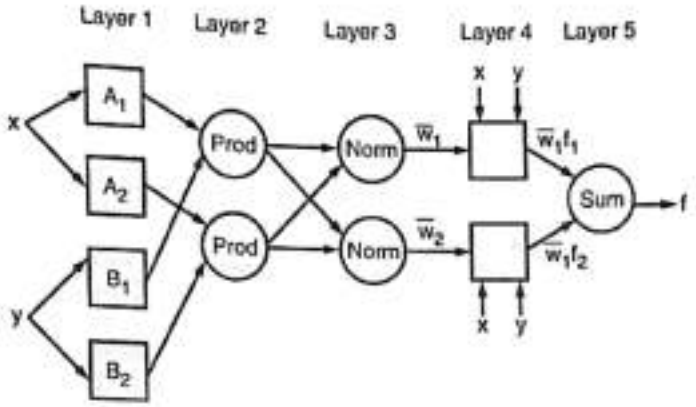
The working of the FIS consists of the following steps:

- A fuzzification unit supports the application of numerous fuzzification methods, and converts the crisp input into fuzzy input.
- A knowledge base - collection of rule base and database is formed upon the conversion of crisp input into fuzzy input.
- The defuzzification unit fuzzy input is finally converted into crisp output.

III. Adaptive Neuro-Fuzzy Inference System (ANFIS)

An adaptive neuro-fuzzy inference system or adaptive network-based fuzzy inference system (ANFIS) is a kind of artificial neural network that is based on Takagi–Sugeno fuzzy inference system. The technique was developed in the early 1990s.Since it integrates both neural networks and fuzzy logic principles, it has potential to capture the benefits of both in a single framework. Its inference system corresponds to a set of fuzzy IF–THEN rules that have learning capability to approximate nonlinear functions. Hence, ANFIS is considered to be a universal estimator. For using the ANFIS in a more efficient and optimal way, one can use the best parameters obtained by genetic algorithm. It has uses in intelligent situational aware energy management system.

ANFIS Architecture



(ncss)**Fig. 2**

Layer 1

$O_{1,i}$ is the output of the i^{th} node of the layer 1. Every node i in this layer is an adaptive node with a node function

$$O_{1,i} = \mu_{Ai}(x) \text{ for } i = 1, 2, \text{ or}$$

$$O_{1,i} = \mu_{Bi-2}(x) \text{ for } i = 3, 4$$

x (or y) is the input node i and A_i (or B_{i-2}) is a linguistic label associated with this node.

Therefore $O_{1,i}$ is the membership grade of a fuzzy set (A_1, A_2, B_1, B_2).

Typical membership function:

$$\mu_A(x) = \frac{1}{1 + \left|\frac{x - c_i}{a_i}\right| 2b_i}$$

a_i, b_i, c_i is the parameter set.

Parameters are referred to as premise parameters.

Layer 2

Every node in this layer is a fixed node labeled Prod.

The output is the product of all the incoming signals.

$O_{2,i} = w_i = \mu_{Ai}(x) \cdot \mu_{ni}(y), i = 1, 2$

Each node represents the fire strength of the rule.

Any other T-norm operator that perform the AND operator can be used.

Layer 3

Every node in this layer is a fixed node labeled Norm.

The i^{th} node calculates the ratio of the i^{th} rulet's firing strength to the sum of all rulet's firing strengths.

$$O_{3,i} = \bar{w}_i = \frac{w_i}{w_1 + w_2}, i = 1, 2$$

Outputs are called normalized firing strengths.

Layer 4

Every node i in this layer is an adaptive node with a node function:

$$O_{4,1} = \bar{w}_i f_i = \bar{w}_i (p_x + q_i y + r_i)$$

$\bar{w}_i$ is the normalized firing strength from layer 3.

$\{p_i, q_i, r_i\}$ is the parameter set of this node.

These are referred to as consequent parameters.

Layer 5

The single node in this layer is a fixed node labeled sum, which computes the overall output as the summation of all incoming signals:

$$\text{Overall output} = O_{5,1} = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i}$$

Chapter Ends...

□□□

MODULE 4

CHAPTER 4 Introduction to Deep Learning

University Prescribed Syllabus w.e.f Academic Year 2022-2023

Introduction to Deep Learning, ANN, Machine Learning Vs Deep Learning, Working of Deep Learning; Convolutional Neural Network: Introduction, Components of CNN Architecture, Properties of CNN, Architectures of CNN, Applications of CNN, Recurrent Neural Network: Introduction, Simple RNN, LSTM Implementation, Deep RNN, Autoencoder: Introduction, Features, Types, Applications of Deep Learning.

Self-learning Topics : Restricted Boltzmann Machine (RBM).

4.1 INTRODUCTION TO DEEP LEARNING

- **Deep Learning** is a type of machine learning that imitates the way humans gain certain type of knowledge. It is a field that is based on learning and improving on its own by examining computer algorithms.
- Deep Learning is a subset of Artificial Intelligence also an important element of data science, which includes statistics and predictive modelling.
- Deep learning gets its name from the fact that it involves going deep into several layers of network, which also includes a hidden layer.
- The deeper you dive; you more complex information you extract.
- Deep learning methods rely on various complex programs to imitate human intelligence. This particular method teaches machines to recognise ideas so that they can be classified into distinct categories.
- To understand deep learning, imagine a toddler whose first word is dog.
- The toddler learns what a dog is and is not by pointing to objects and saying the word dog.
- The parent says, "Yes, that is a dog," or, "No, that is not a dog." As the toddler continues to point to objects, he becomes more aware of the features that all dogs possess.
- What the toddler does, without knowing it, is clarify a complex abstraction "the concept of dog" by building a hierarchy in which each level of abstraction is created with knowledge that was gained from the preceding layer of the hierarchy.

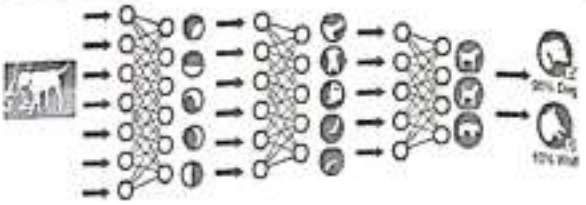


Fig. 4.1.1 : Understanding deep learning

4.2 MACHINE LEARNING VS DEEP LEARNING

- Deep learning is a type of machine learning that differs in how it solves problems. While traditional machine learning algorithms are linear, deep learning algorithms are highly complex and non-linear. To discover the most widely used features in machine learning, a domain expert is required.
- Deep learning, on the other hand, learns features incrementally, obviating the need for domain knowledge. Deep learning algorithms, on the other hand, require far longer to train than machine learning algorithms, which take anywhere from a few seconds to a few hours.
- During testing, on the other hand, the opposite is true. Deep learning algorithms conduct tests at a fraction of the time it takes machine learning algorithms, whose test time increases as the size of the data grows.
- Thus, deep learning is a specialised subset of machine learning. The Table 4.2.1 depicts the difference between Machine learning and Deep Learning.

Difference between Machine Learning & Deep Learning

Table 4.2.1 : difference between Machine learning and Deep Learning

Sr. No.	Machine Learning	Deep Learning
1.	Machine Learning is a superset of Deep Learning	Deep Learning is a subset of Machine Learning
2.	Uses various types of automated algorithms that turn to model functions and predict future action from data.	Uses neural network that passes data through processing layers to the interpret data features and relations.
3.	The data representation in Machine Learning is quite different as compared to Deep Learning as it uses structured data	The data representation is used in Deep Learning is quite different as it uses neural networks (ANN).
4.	Machine Learning algorithms are linear.	Deep Learning algorithms are complex and non-linear.
5.	Machine learning consists of thousands of data points.	Deep learning works on Big Data, so millions of data points.

Sr. No.	Machine Learning	Deep Learning
6.	To find best set of features in machine learning, domain expert is required.	Deep learning learns features incrementally, obviating the need for domain knowledge.
7.	Time required to train/test a machine learning model is comparatively very less than a deep learning model.	Time required to train/test a deep learning model is high as more layers are present.
8.	Not necessary to have costly high-end machines.	High-end machines and High performing GPUs are required.
9.	Outputs of a machine learning model can be some class label or numerical Value, like classification score.	Output of a deep learning model can be anything from numerical values to free-form elements, such as free text and sound.
10.	Algorithms are detected by data analysts to examine specific variables in data sets.	Algorithms are largely self-depicted on data analysis once they're put into production.

4.3 WORKING OF DEEP LEARNING NETWORKS

- Deep Learning networks are mathematical models that are used to represent human brains in order to solve tasks with unstructured data. These mathematical models are created in form of neural network that consists of neurons.
- The basic neural network is divided into three major layers that are input layer which is the first layer of neural network, hidden layer (all the middle layers of neural network) and the output layer which is the last layer of the neural network. To understand working of Deep learning, lets understand neural network first.
- The human brain is a highly complex and amazing processor. The most basic element of the human brain is a specific type of cell, known as **Neuron**, which does not regenerate.
- The power of human brain (min(d) comes from the number of Neurons and their multiple interconnections. A brain has the ability to **Learn** to build up its own rules through what we refer to as Experience.
- A **Neural-Network** is a machine that is designed to model the way in which the brain performs a particular task. Neural networks are those information processing systems, which are constructed and implemented to **model the human brain**.
- The concept that is of primary importance for a Neural Network is the ability of the network to **learn** from its environment, and to **improve** its performance through learning.
- To achieve good performance, Neural networks employ a massive interconnection of simple computing cells, referred to as "**Neurons**" or "**Processing units**".
- Let's have a look at biological neural network to understand ANN.

4.3.1 Biological Neural Network

Human brain consists of a huge number of neurons, approximately 1000 billion. Each neuron has an association point somewhere in the range of 1,000 and 100,000.

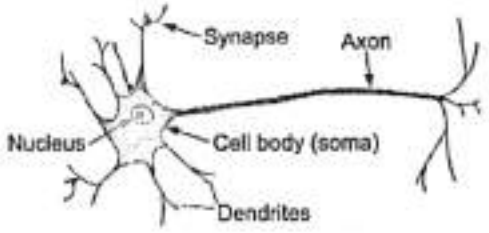


Fig. 4.3.1 : Diagram of a Biological Neuron

In the human brain, data is stored in such a manner as to be distributed, and we can extract more than one piece of this data whenever necessary from our memory parallelly. We can say that the human brain is made up of incredibly amazing parallel processors.

The biological neuron shown in the Fig. 4.3.1. consists of three main parts :

1. **Soma or cell body :** Here the cell nucleus is located.
2. **Dendrites :** Where the nerve is connected to the cell body.
3. **Axon :** It carries the impulses of the neuron.

(i) **Dendrites** are networks made of nerve-fibre and are connected to the cell body.

(ii) An Axon carries signals from the neuron. It is a single, long connection extending from the cell body.

(iii) The end of the axon splits into fine strands. Each strand terminates into a small-bulb-like organ called synapse. It is through synapse that the neuron introduces its signals to other nearby neurons. The receiving ends of these synapses on the nearby neurons can be found on the dendrites and also on the cell-body. Approximately there are 10^4 **synapses** per neuron in the human brain.

- Electric impulses are passed between synapse and the dendrites.
- This signal transmission involves a chemical process in which specific transmitter substances are releases from the sending side of the junction.
- This causes and increase or decrease in the electric potential inside the body of the receiving cell.
- If the electric potential reaches or crosses a threshold then the receiving cell fires and a pulse or action potential of fixed strength and duration is sent through the **axon** to the **synaptic** junctions of the other cells.
- The synapses are inhibitory it they hinder the firing of the receiving cell, or they are excitatory if they pass impulses which cause the firing of the receiving cell.

4.3.2 Artificial Neural Network (ANN)

- An artificial neural network (ANN) is an efficient information processing system which resembles in characteristics with a biological neutral network (BNN).
- ANN's behavior is similar to that of a human brain, so far as ability to learn, recall and generalize training patterns. Thus, the ANN processing elements are called **neurons** or **artificial neurons.**
- Dendrites from Biological Neural Network represent inputs in Artificial Neural Networks, cell nucleus represents Nodes, synapse represents Weights, and Axon represents Output.

Relationship between BNN and ANN

Sr. No.	Biological neural network	Artificial neural network
1.	Cell nucleus	Neuron / Nodes
2.	Synapse	Weights or interconnections
3.	Dendrites	Inputs
4.	Axon	Output

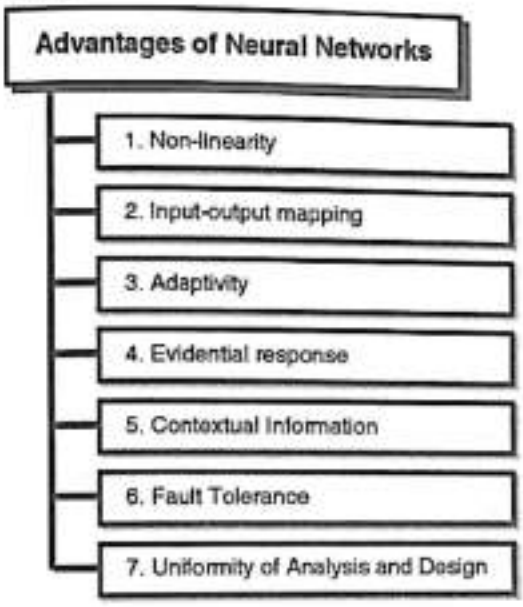
Comparison between Biological Neuron (BNN) and Artificial Neural Network (ANN)

Sr. No.	Characteristics	Artificial Neural Network	Biological (Real) Neural Network
1.	Speed	Faster in processing information. Response time is in nanoseconds.	Slower in processing information. The response time is in milliseconds.
2.	Processing	Serial processing	Massively parallel processing.
3.	Size and complexity	Less size and complexity. It does not perform complex pattern recognition tasks.	A highly complex and dense network of interconnected neurons containing neurons of the order of 1011 with 1015 of interconnections.
4.	Storage	New data with old data can be replaced, i.e. information storage is replaceable.	New information or data is added by adjusting the interconnection strength without destroying old information.
5.	Anism	There is a control unit for controlling computing activities.	No specific control mechanism external to the computing task.

Sr. No.	Characteristics	Artificial Neural Network	Biological (Real) Neural Network
6.	Resemblance	A neural network acquires knowledge through learning.	–
7.	Resemblance	A neural network's knowledge is a stored within inter neurons connection known as synaptic weights.	–

☞ Advantages of Neural Networks

The use of neural networks offers the following useful capabilities :



(1) **Non-linearity :** An artificial neuron can be **linear or non-linear**. A neural network consists of an interconnection of non-linear neurons, and hence it is itself non-linear. The non-linearity is distributed throughout the network.

(2) **Input-output mapping :** Each example consists of a unique **input signal** and corresponding desired response. Thus, the network learns from the examples by constructing an **input-output mapping** for the problem at hand.

(3) **Adaptivity :** A neural network is trained to operate in a specific environment can be easily retrained to deal with minor changes in the operating environmental conditions.

(4) **Evidential response :** A neural network can be designed to provide information not only about which particular pattern to **select,** but also about the **confidence** in the decision made.

(5) **Contextual Information :** Knowledge is represented by the very structure of a neural network. Every neuron in the network is potentially affected by the global activity of all other neurons in the network.

(6) **Fault Tolerance :** A neural network has the potential to be **fault tolerant.** In order to have neural network **fault tolerant,** it is necessary to take corrective measures in designing the algorithm used to develop the network.

(7) **Uniformity of Analysis and Design :** Neural networks are basically information processors. Hence the same notation is used in all domains involving the application of neural networks.

☞ Applications of Neural Networks

Variety of problems can be solved by applications of neural networks.

Some of the common applications are :

- Pattern Recognition.
- Image processing.

  Neural Networks are very useful in visual images, handwritten characters, printed characters, speech and other PR based tasks.
- Optimisation/Constraint Satisfactions.

  To obtain optimal solution satisfying given constraints, such problems can be solved by NN successfully.

Examples

1. Manufacturing scheduling,
2. Finding the shortest possible tour given a set of cities,
3. Problems of this nature arising out of industrial and manufacturing fields have been found acceptable solutions using NNS.

- **Forecasting and Risk Assessment :** Neural networks have shown capability to predict situations form past trends. There are ample applications in area such as meteorology, banking, stock market, econometrics with high success rates.
- **Control system :** NNS have gained commercial ground by applying applications in control systems. Companies incorporating NN technology have produced dozers of computer successfully.

There are many applications for the control of chemical plants, robots and so on.

Understanding working of Artificial Neural Network (ANN)

- To understand the concept of the architecture of an artificial neural network, we have to understand what a neural network consists of.
- In order to define a neural network that consists of a large number of artificial neurons, which are termed units arranged in a sequence of layers.
- Let's us look at various types of layers available in an artificial neural network.

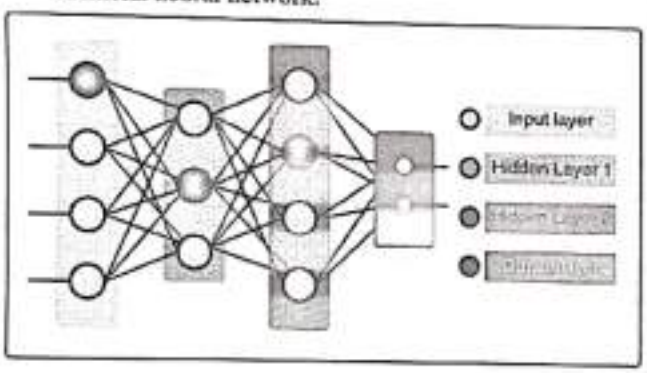


Fig. 4.3.2 : Basic architecture of ANN

ANN primarily consists of three layers :

1. Input Layer

- Input layer accepts inputs in several formats provided by the programmer.

2. Hidden Layer

- The hidden layer(s) is/are present between input and output layers.
- It performs all the calculations to find hidden features and patterns.

3. Output Layer

- The input goes through a series of transformations using the hidden layer, which finally results in output that is conveyed using this layer.
- The artificial neural network takes input and computes the weighted sum of the inputs and includes a bias. This computation is represented in the form of a transfer function.

$$Y = \sum_{i=1}^{n} x_i w_i + b$$

- It determines weighted total is passed as an input to an activation function to produce the output. Activation functions choose whether a node should fire or not.
- Only those nodes who are fired make it to the output layer. There are distinctive activation functions available that can be applied upon the sort of task we are performing.
- Artificial Neural Network can be best represented as a weighted directed graph, where the artificial neurons form the nodes.
- The association between the neurons outputs and neuron inputs can be viewed as the directed edges with weights.
- The Artificial Neural Network receives the input signal from the external source in the form of a pattern and image in the form of a vector.
- These inputs are then mathematically assigned by the notations x(n) for every n number of inputs.

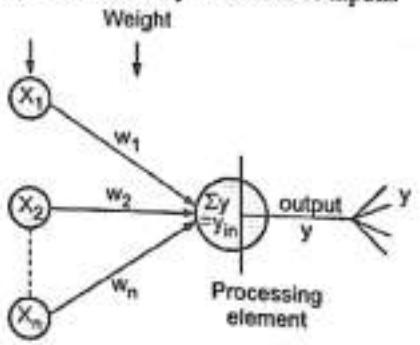


Fig. 4.3.3 : Mathematical model of artificial neuron

- In the model given in Fig. 4.3.3, the net input is given by,

$$y_{in} = x_1 w_1 + x_2 w_2 + \ldots + x_n w_n = \sum_{i=1}^{n} x_i w_i$$

Where i is i^{th} processing element.

- Based on this basic architecture, there can be similar architectures as provided in Figs. 4.3.4 and 4.3.5.
- Architecture provided in Fig. 4.3.5 shows that there are just two inputs, so it's equation for the output becomes:

$$Y = x_1 w_1 + x_2 w_2$$

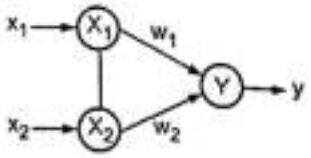


Fig. 4.3.4 : Architecture of a simple artificial neutron network

Fig. 4.3.5 : Neural net of pure linear equation

Note :

(i) Each neurons has its own internal state, and is called the **activation** or **activity level** of neuron.

(ii) The state of neuron is a function of inputs it receives, and this activation signal is transmitted to other neurons.

(iii) A neuron can send only one signal at a time and is transmitted to other several neurons.

Important Terminologies of ANN

1. Weights

(1) Each neuron, in the architecture of ANN, is connected to other neurons by means of direct communication links, and each links is associated with weights.

(2) Weights contain information about the input signal.

(3) This information is used by the net to solve a problem.

(4) The weight can be represented in terms of matrix. The weight matrix is also called as connection matrix.

(5) To make mathematical formulation, let there be 'n' processing elements in an ANN and each processing element has exactly m adaptive weights.

Thus the weight matrix w is defined as :

$$W = \begin{bmatrix} w_1^T \\ w_2^T \\ \vdots \\ w_n^T \end{bmatrix} \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1m} \\ w_{21} & w_{22} & \cdots & w_{2m} \\ \vdots & & & \\ w_{n1} & w_{n2} & \cdots & w_{nm} \end{bmatrix}$$

Where, $w_i = [w_{i1}, w_{i2}, \cdots, w_{im}]^T$

$i = 1$ to n. is the weight vector of processing element

And wij is the weight from processing element 'i' (source node) to processing element 'j' (destination node).

(6) The set of all W matrices will determine the set of all information for the ANN.

(7) The ANN can be realised by finding an appropriate matrix w.

(8) The weights encode long*term memory (LTM) and the activation states of neurons encode short-term memory (STM) in a neural network.

2. Bias

(1) The bias is included in the network.

(2) It has impact in calculating the net input.

(3) The input vector becomes, $X = \{1, X_1 X_2, ..., X_n\}$

(4) The bias is another weight, say $w = b_1$.

(5) Bias plays a major role in determining the output of the network.

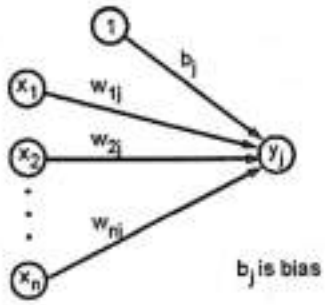


Fig. 4.3.6 : Net with bias

(6) The bias is of two types :

(a) Positive bias increases the net input of the network and

(b) The negative bias decreases the net input of the network.

(7) Due to bias-effect, the output of the network can be varied.

3. Threshold

(1) Threshold is a set value on which the final output of the network may be calculated.

(2) In activation function, threshold value is used.

(3) To obtain the network output, a comparison is made between the calculated net input and the threshold.

- **Forecasting and Risk Assessment :** Neural networks have shown capability to predict situations form past trends. There are ample applications in area such as metreorology, banking, stock market, econometrics with high success rates.
- **Control system :** NNS have gained commercial ground by applying applications in control systems. Companies incorporating NN technology have produced dozers of computer successfully.

There are many applications for the control of chemical plants, robots and so on.

Understanding working of Artificial Neural Network (ANN)

- To understand the concept of the architecture of an artificial neural network, we have to understand what a neural network consists of.
- In order to define a neural network that consists of a large number of artificial neurons, which are termed units arranged in a sequence of layers.
- Let's us look at various types of layers available in an artificial neural network.

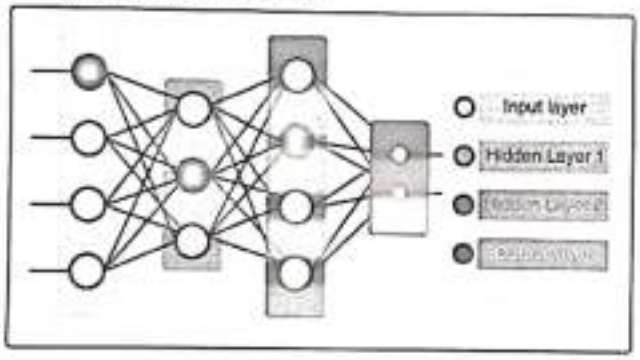


Fig. 4.3.2 : Basic architecture of ANN

ANN primarily consists of three layers :

1. Input Layer

- Input layer accepts inputs in several formats provided by the programmer.

2. Hidden Layer

- The hidden layer(s) is/are present between input and output layers.
- It performs all the calculations to find hidden features and patterns.

3. Output Layer

- The input goes through a series of transformations using the hidden layer, which finally results in output that is conveyed using this layer.
- The artificial neural network takes input and computes the weighted sum of the inputs and includes a bias. This computation is represented in the form of a transfer function.

$$Y = \sum_{i=1}^{n} x_i w_i + b$$

- It determines weighted total is passed as an input to an activation function to produce the output. Activation functions choose whether a node should fire or not.
- Only those nodes who are fired make it to the output layer. There are distinctive activation functions available that can be applied upon the sort of task we are performing.
- Artificial Neural Network can be best represented as a weighted directed graph, where the artificial neurons form the nodes.
- The association between the neurons outputs and neuron inputs can be viewed as the directed edges with weights.
- The Artificial Neural Network receives the input signal from the external source in the form of a pattern and image in the form of a vector.
- These inputs are then mathematically assigned by the notations x(n) for every n number of inputs.

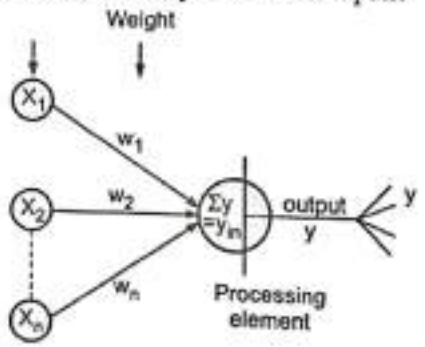


Fig. 4.3.3 : Mathematical model of artificial neuron

- In the model given in Fig. 4.3.3, the net input is given by,

$$y_{in} = x_1 w_1 + x_2 w_2 + \ldots + x_n w_n = \sum_{i=1}^{n} x_i w_i$$

Where i is i^{th} processing element.

- Based on this basic architecture, there can be similar architectures as provided in Figs. 4.3.4 and 4.3.5.
- Architecture provided in Fig. 4.3.5 shows that there are just two inputs, so it's equation for the output becomes:

$$Y = x_1 w_1 + x_2 w_2$$

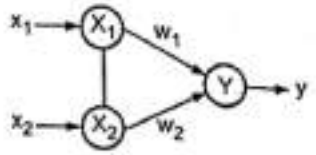


Fig. 4.3.4 : Architecture of a simple artificial neutron network

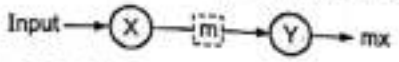


Fig. 4.3.5 : Neural net of pure linear equation

➡ **Note :**

(i) Each neurons has its own internal state, and is called the **activation** or **activity level** of neuron.

(ii) The state of neuron is a function of inputs it receives, and this activation signal is transmitted to other neurons.

(iii) A neuron can send only one signal at a time and is transmitted to other several neurons.

☞ Important Terminologies of ANN

1. Weights

(1) Each neuron, in the architecture of ANN, is connected to other neurons by means of direct communication links, and each links is associated with weights.

(2) Weights contain information about the input signal.

(3) This information is used by the net to solve a problem.

(4) The weight can be represented in terms of matrix. The weight matrix is also called as connection matrix.

(5) To make mathematical formulation, let there be 'n' processing elements in an ANN and each processing element has exactly m adaptive weights.

Thus the weight matrix w is defined as :

$$W = \begin{bmatrix} w_1^T \\ w_2^T \\ \vdots \\ w_n^T \end{bmatrix} \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1m} \\ w_{21} & w_{22} & \cdots & w_{2m} \\ \vdots & & & \\ w_{n1} & w_{n2} & \cdots & w_{nm} \end{bmatrix}$$

Where, $w_i = [w_{i1}, w_{i2}, \cdots, w_{im}]^T$

i = 1 to n. is the weight vector of processing element

And wij is the weight from processing element 'i' (source node) to processing element 'j' (destination node).

(6) The set of all W matrices will determine the set of all information for the ANN.

(7) The ANN can be realised by finding an appropriate matrix w.

(8) The weights encode long*term memory (LTM) and the activation states of neurons encode short-term memory (STM) in a neural network.

2. Bias

(1) The bias is included in the network.

(2) It has impact in calculating the net input.

(3) The input vector becomes, $X = (1, X_1 X_2, ..., X_n)$

(4) The bias is another weight, say $w = b_j$.

(5) Bias plays a major role in determining the output of the network.

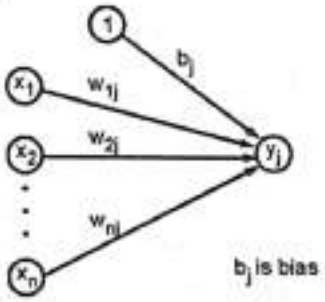


Fig. 4.3.6 : Net with bias

(6) The bias is of two types :

(a) Positive bias increases the net input of the network and

(b) The negative bias decreases the net input of the network.

(7) Due to bias-effect, the output of the network can be varied.

3. Threshold

(1) Threshold is a set value on which the final output of the network may be calculated.

(2) In activation function, threshold value is used.

(3) To obtain the network output, a comparison is made between the calculated net input and the threshold.

(4) There is a threshold limit for each and every applications.

(5) The activations function using threshold is defined as

$$f(net) = \begin{cases} 1 & \text{if net} \geq \theta \\ -1 & \text{if net} < \theta; \end{cases}$$

Where θ is fixed threshold value.

4.3.3 McCulloch-Pitts Neuron Architecture

The very first step towards the perceptron/neuron we use today was taken in 1943 by McCulloch and Pitts, by mimicking the functionality of a biological neuron. The Mc-Culloch-Pitts Neuron is the first computational model of a neuron. It is usually called a **m-p neuron**.

(1) The M – P neurons are **connected** by **directed weighted paths**.

(2) Activation of a M-P neuron is binary i.e 1 or 0, that is, at any time step, the neuron may fire or may not fire.

(3) The weight associated with the links are of **two types** :

(a) They may be **excitatory** (weight is positive) or

(b) They may be **inhibitory** (weight is negative).

(4) The threshold plays a major role as an activation function.

(5) For each neuron, threshold is fixed.

(6) If the net input to the neuron is greater than the threshold the neuron fires i.e. the output signal y_{out} is 1 if the input y_{sum} is greater than or equal to a given threshold value.

(7) Any non-zero inhibitory input will prevent the neuron from firing i.e. the output signal y_{out} is 0 if the input y_{sum} is less than or equal to a given threshold value.

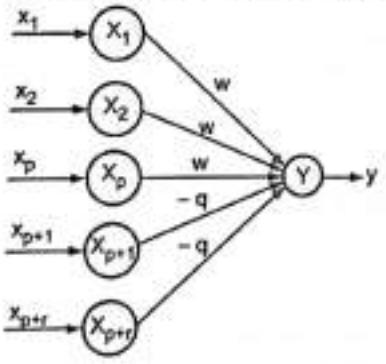


Fig. 4.3.7 : McCulloch-pitts neuron Architecture

Here the inputs refer Fig. 4.3.7 x_1 to x_p (in the figure) possess excitatory weighted connections and input from x_{p+1} to x_{p+r} possess inhibitory weighted connections.

We define activation function as,

$$f(y_{in}) = \begin{cases} 1 \text{ if } y_{in} \geq \theta \\ 0 \text{ if } y_p < \theta \end{cases} \quad ; \text{ Where } \theta > pw - q$$

The output will fire if it receives "k" or more excitatory inputs but no inhibitory inputs,

i.e., $kw \geq \theta \geq (k-1)\, w$.

4.3.3.1 Linear Separability

To separate positive and negative responses, a **decision line** is drawn. The decision line is also called as the decision making line or **'decision-support line'** or **'linear-separable line'.**

Generally the net input calculated to the output unit is given as :

$$y_{in} = b + \sum_{i=1}^{n} x_i w_i \qquad \ldots(i)$$

The value of the function is 1 for a +ve net input and –1 for a negative net input. Also there exists a boundary between the region. Where $y_{in} > 0$ and $y_{in} < 0$.

This region is called is called as **decision boundary region,** and can be determined by the relation.

$$b + \sum_{i=1}^{n} x_i w_i = 0 \qquad \ldots(ii)$$

Remarks

(1) Depending upon the number of input units in the network, the above equation (ii) may represent a line, a plane or a hyper plane.

(2) If there exist weights for which the input vectors having positive response, +1, lie on one side of the decision boundary line and all other vectors having negative response, –1, lie on the other side of the decision boundary, then the problem is **"linearly separable"**.

4.3.3.2 Single-Layer Network

Let us consider a Single-Layer Network

Now, the separating line is given by,

$$b + x_1 w_1 + x_2 w_2 = 0$$

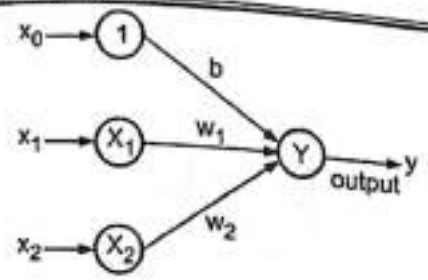


Fig. 4.3.8 : A single-layer neural-net

If $w_2 \neq 0$, then

$$x_2 = -\frac{w_1}{w_2} x_1 - \frac{b}{w_2}$$

Thus the requirement for the positive response is :

$b + x_1 w_1 + x_2 w_2 > 0$

Remarks

If the threshold value is being used, then the condition for the positive response from the output unit is

Net input received $>\theta$ (threshold value)

$\therefore y_{in} > \theta$; $\therefore x_1 w_1 + x_2 w_2 > \theta$

Then the equation of the separating line will be

$$x_1 w_1 + x_2 w_2 = \theta ;$$

$$\therefore x_2 = -\frac{w_1}{w_2} x_1 + \frac{\theta}{w_2}, \text{ (with } w_2 \neq 0)$$

Simple McCulloch-Pitts neurons can be used to design logical operations. For that purpose, the connection weights need to be correctly decided along with the threshold function (rather than the threshold value of the activation function).

Ex. 4.3.1 : Implement AND function using Mc-Culloch-Pitts neuron (take binary data).

☑ **Soln. :**

▶ **Step (I) : First we construct truth-table for AND function :**

Here we assume the weights to be $w_1 = 1$ and $w_2 = 1$.

With these assumed weights we calculate the net input for four inputs as follows :

Truth Table

x_1	x_2	y
1	1	1
1	0	0
0	1	0
0	0	0

For inputs :

$(1, 1)$, $y_{in} = x_1 w_1 + x_2 w_2 = 1 \times 1 + 1 \times 1 = 2$

$(1, 0)$, $y_{in} = x_1 w_1 + x_2 w_2 = 1 \times 1 + 1 \times 0 = 1$

$(0, 1)$, $y_{in} = x_1 w_1 + x_2 w_2 = 0 \times 1 + 1 \times 1 = 1$

$(0, 0)$, $y_{in} = x_1 w_1 + x_2 w_2 = 0 \times 1 + 0 \times 1 = 0$

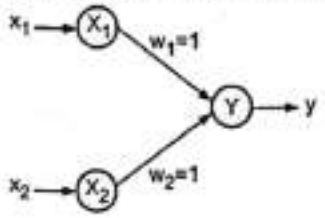


Fig. Ex. 4.3.1 : Network architecture for AND

▶ **Step (II) : For an AND function, the output is high if both the inputs are high, Hence we calculate the net input as 2.**

Based on this net input, the thresh is set. If the threshold is greater than or equal to 2, then the neuron fires, otherwise it does not fire.

$\therefore$ Threshold value is set equal to 2, $(\theta = 2)$. Also, we can use the formula : $\theta \geq nw - p$

Here, $n = 2$, $w = 1$ and $p = 0$

($\because$ no inhibitory weights).

$\therefore \theta \geq 2(1) - 0 = 2$

Thus, the output of neuron Y can be written as :

$$y = f(y_{in}) = \begin{cases} 1 \text{ if } y_{in} \geq 2 \\ 0 \text{ if } y_{in} < 2 \end{cases}$$

where '2' represents the threshold value.

Ex. 4.3.2 : Implement ANDNOT function using MC-Culloch-Pitts-neuron (use binary data representation).

☑ **Soln. :**

▶ **Step (I) : We prepare the truth-table for ANDNOT function.**

In this case, the response is true if the first input is true and the second input is false. For all other input variations, the response is false.

Truth Table

x_1	x_2	y
0	0	0
0	1	0
1	0	1
1	1	0

The given function gives an output when $x_1 = 1$ and $x_2 = 0$.

We represent the neural net :

Step (II) : Case(i) : Let w_1 and w_2 be excitatory, i.e.,

$$w_1 = w_2 = 1$$

Now, we calculate the net input using four inputs.

$$y_{in} = x_1 w_1 + x_2 w_2$$

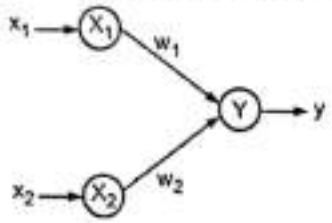


Fig. Ex. 4.3.2 : Network architecture for ANDNOT

For inputs

$(1, 1) : y_{in} = 1 \times 1 + 1 \times 1 = 2$;

$(1, 0) : y_{in} = 1 \times 1 + 0 \times 1 = 1$

$(0, 1) : y_{in} = 0 \times 1 + 1 \times 1 = 1$;

$(0, 0) : y_{in} = 0 \times 1 + 0 \times 1 = 0$

From the calculated net inputs, it is not possible to fire the neuron for input (1, 0) only. Hence these weights are not suitable.

So, we assume : one weight as excitatory and the other as inhibitory, i.e.,

$w_1 = 1, w_2 = -1$.

Now, we calculate the net input : For the input :

$(1, 1) : y_{in} = 1 \times 1 + 1 \times (-1) = 0$;

$(1, 0) : y_{in} = 1 \times 1 + 0 \times (-1) = 1$

$(0, 1) : y_{in} = 0 \times 1 + 1 \times (-1) = -1$;

$(0, 0) : y_{in} = 0 \times 1 + 0 \times (-1) = 0$

Now, it is possible to fire the neuron for input (1, 0) only by firing a threshold of 1 ;

i.e. $\theta \geq 1$. $\therefore$ $w_1 = 1$, $w_2 = -1$; $\theta \geq 1$

Thus the output of neuron Y can be written as,

$$y = f(y_{in}) = \begin{cases} 1, \text{ if } y_{in} \geq 1 \\ 0, \text{ if } y_{in} < 0 \end{cases}$$

Ex. 4.3.3 : Using the linear separability concept, obtain the response for OR function (taking bipolar inputs and bipolar targets).

Soln. :

Step (I) : We prepare truth-table for OR function with bipolar inputs and targets.

Noting the points, we plot the figure :

Truth-table

x_1	x_2	y
1	1	1
1	-1	1
-1	1	1
-1	-1	-1

Using co-ordinates (1, – 1) and (– 1, 1);

$$\text{Slope of line} = \frac{1-(-1)}{(-1)-(1)} = \frac{2}{-2} = -1$$

If the output is 1.

It is denoted as '+ve', otherwise '–ve'.

Using this value the equations for the line is,

$$y = mx + c = (-1)x - 1 = -x - 1$$

Note : Here the quadrants are x_1 and x_2 and not x and y.

$\therefore$ The above equation becomes $x_2 = -x_1 - 1$ ...(i)

This can be written as $x_2 = -\frac{w_1}{w_2} x_1 - \frac{b}{w_2}$ ...(ii)

Comparing (i) and (ii); $\frac{w_1}{w_2} = \frac{1}{1}$, $\frac{b}{w_2} = \frac{1}{1}$

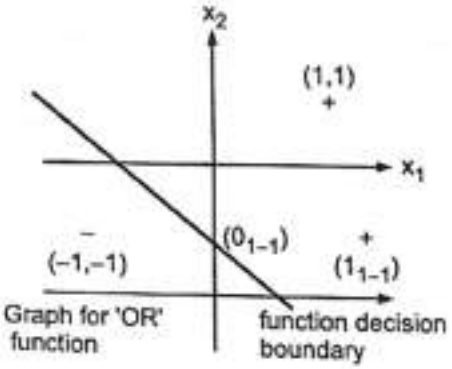


Fig. Ex. 4.3.3 : Graph for 'OR'

$\therefore$ $w_1 = 1$ and $w_2 = 1$, and $b = 1$

Step (II) : Calculating the not input and output of OR function on the basis of the above weights, we have the entries

Thus the output of the neuron Y can be written as

$$y = f(y_{in}) = \begin{cases} 1, \text{ if } y_n \geq 1 \\ 0, \text{ if } y_n < 1 \end{cases}$$

Truth-table

x_1	x_2	b	$y_{in} = b + x_1 w_1 + x_2 w_2$	y
1	1	1	3	1
1	−1	1	1	1
−1	1	1	1	1
−1	−1	1	−1	−1

Where the threshold is taken as "1" ($\theta = 1$) based on calculated net input. Hence using the linear separability concept, He response is obtained for "OR" function.

Ex. 4.3.4 : Implement XOR function using Mc-Culloch-Pitts neuron (consider binary data).

☑ **Soln. :**

▶ **Step (I) : Prepare truth-table for XOR function**

Truth-Table

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

In this case, only for add numbers of 1's the output is 'ON'. And for the rest it is 'OFF'.

Such a function cannot be represented by simple and single logic functions; so we represent it as follows:

$y = x_1\bar{x}_2 + \bar{x}_1 x_2$ or $y = z_1 + z_2$

where $z_1 = x_1\bar{x}_2$ (function 1)

$z_2 = \bar{x}_1 x_2$ (function 2)

and $y = z_1$ (OR) z_2 (function 3)

Here a single-layer is not sufficient to represent the function. An intermediate layer is necessary :

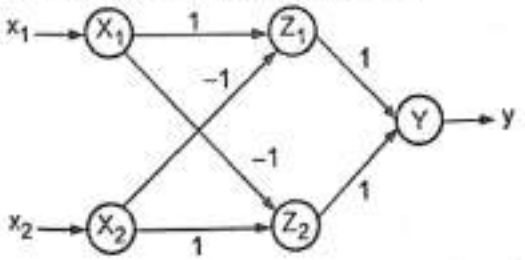


Fig. Ex. 4.3.4(a) : Neural net for XOR function the weights shown are obtained after analysis

▶ **Step (II) : Prepare truth-table for first function**

Truth-Table

x_1	x_2	z_1
0	0	0
0	1	0
1	0	1
1	1	0

$z = x_1\bar{x}_2$:

The net representation is given as :

Case (1) : Assume both weights as excitatory, i.e.

$w_{11} = w_{21} = 1$

We calculate the net inputs :

For (0, 0) ; $z_1(in) = 0 \times 1 + 0 \times 1 = 0 + 0 = 0$

For (0, 1) ; $z_1(in) = 0 \times 1 + 1 \times 1 = 0 + 1 = 1$

For (1, 0) ; $z_1(in) = 1 \times 1 + 0 \times 1 = 1 + 0 = 1$

For (1, 1) ; $z_1(in) = 1 \times 1 + 1 \times 1 = 1 + 1 = 2$

Hence, using these weights, it is not possible to obtain function z_1 ;

[∵ for $\theta > 2$, there is not excitatory].

▶ **Case (2) : We assume one weight as excitatory and the other as inhibitory, i.e.,**

$w_{11} = 1$; $w_{21} = -1$.

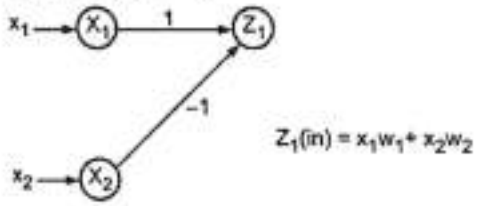


Fig. Ex. 4.3.4(b) : Neural for Z_1

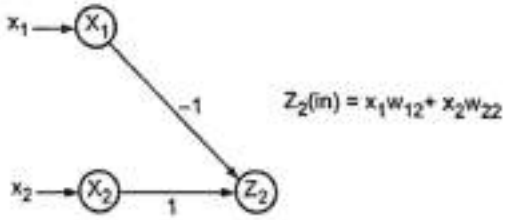


Fig. Ex. 4.3.4(c) : Neural for Z_2

We calculate the net inputs for inputs :

(0, 0), z_1 (in) $= 0 \times 1 + 0 \times (-1) = 0 + 0 = 0$,

(0, 1), z_1 (in) $= 0 \times 1 + 1 \times (-1) = -1$

(1, 0), z_1 (in) $= 1 \times 1 + 0 \times (-1) = 1$;

(1, 1), z_1 (in) $= 1 \times 1 + 1 \times (-1) = 0$

On the basis of this calculated net input, it is possible to get the required output. Hence.

$w_{11} = 1$, $w_{21} = -1$ and $\theta \geq 1$ for z_1 neuron.

▶ **Step (III) : Second function :** $z_2 = \bar{x}_1 x_2$

We prepare the truth-table for z_2 :

Truth-Table for z_2

x_1	x_2	z_2
0	0	0
0	1	1
1	0	0
1	1	0

▶ **Case (1) : Assume both weights as excitatory,**

i.e. $w_{12} = w_{22} = 1$.

With this, we calculate the inputs.

For the inputs :

(0, 0), z_2 (in) $= 0 \times 1 + 0 \times 1 = 0$;

(0, 1), z_2 (in) $= 0 \times 1 + 1 \times 1 = 1$

(1, 0), z_2 (in) $= 1 \times 1 + 0 \times 1 = 1$;

(1, 1), z_2 (in) $= 1 \times 1 + 1 \times 1 = 2$

Again, it is not possible to obtain function Z_2

Using these weights (as above case).

▶ **Case (2) : (As in the previous step) we assume one weight as excitatory and the other as inhibitory, i.e., $w_{12} = -1$; $w_{22} = 1$.**

Now calculate the net inputs. For the inputs :

(0, 0), z_2 (in) $= 0 \times (-1) + 0 \times 1 = 0$;

(0, 1), z_2 (in) $= 0 \times (-1) + 1 \times 1 = 1$

(1, 0), z_2 (in) $= 1 \times (-1) + 0 \times 1 = -1$;

(1, 1), z_2 (in) $= 1 \times (-1) + 1 \times (1) = 0$

Thus, based on this calculated net input, it is possible to get the required output, i.e.,

$w_{12} = -1$, $w_{22} = 2$. $\therefore \theta \geq 1$ for z_2 neuron.

▶ **Case (3) : Third function :**

($y = z_1$ Or z_2)

The truth table for this function is an follows

x_1	x_2	y	Z_1	z_2
0	0	0	0	0
0	1	1	0	1
1	0	1	1	0
1	1	0	0	0

The net input is to be calculated as :

$y_{in} = z_1 v_1 + z_2 v_2$;

Where V_1 and V_2 are weights to be assigned.

▶ **Case (i) : Assume both weights as excitatory, i.e.,**

$v_1 = v_2 = 1$

We calculate the net input. For inputs : (0, 0),

$y_{in} = 0 \times 1 + 0 \times 1 = 0$

(0, 1), $y_{in} = 0 \times 1 + 1 \times 1 = 1$;

(1, 0), $y_{in} = 1 \times 1 + 0 \times 1 = 1$

(1, 1), $y_{in} = 0 \times 1 + 0 \times 1 = 0$

[it is $\because$ for $x_1 = 1$, $x_2 = 1$; $z_1 = 0$ and $z_2 = 0$]

[from the table].

Setting a threshold of $\theta \geq 1$,

$v_1 = v_2 = 1$; it implies that the net is recognised.

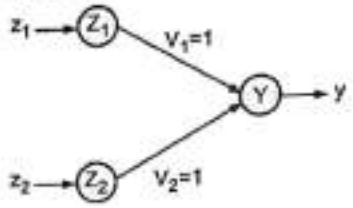


Fig. Ex. 4.3.4(d) : Neural net for Y(Z_1 or Z_2)

The analysis made for XOR function using M-P neurons.

Thus for XOR – function, the weights to be assigned are

$w_{11} = w_{22} = 1$ (excitatory) ;

$w_{12} = w_{21} = -1$ (inhibitory)

$v_1 = v_2 = 1$ (excitatory)

4.3.4 Activation Functions

(1) A model of the behaviour of a neuron can be presented as shown in Fig. 4.3.9.

Here x_1, x_2, ..., x_n are the n-inputs to the artificial neuron, w_1,

w_2, ..., w_n are the weights attached to the input links.

(2) **Biological** neuron receives all inputs through the dendrites, sums them and produces an output. If the sum is greater than a threshold value. The input signals are passed on to the cell body through the synapse which may accelerate or retard an arriving signal.

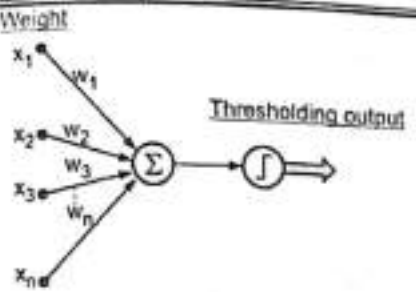


Fig. 4.3.9 : Simple model of an artificial neuron

(3) Here weights model the acceleration or retardation of the input signals. In short, weights are multiplicative factors of the inputs.

(4) The total input (say I) received by the soma (body) of the artificial neuron is

$$I = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n \quad \ldots(i)$$

$$= \sum_{i=1}^{n} w_i x_i$$

(5) This sum is passed on to a non-linear filter ϕ called **Activation function**, or **Transfer function**, or **Squash Function** which releases the outputs.

$$\text{i.e. } y = f(I) \quad \ldots(ii)$$

The obtain exact output, the activation function is applied over the net input of an ANN. There are several activation functions.

Types of Neural Networks Activation Functions

1. Binary step function (Threshold Function)
2. Linear Activation Function
3. Sigmoid/Logistic Activation Function
4. The derivative of the sigmoid Activation Function
5. Tanh Function (Hyperbolic Tangent)
6. Gradient of the Tanh Activation function
7. ReLu Activation Function

Lets discuss few of them here.

1. **Linear function :** It is defined as

$$f(x) = x \quad \text{for all } x$$

Here input = output

2. **Bipolar Step function :** The function is defined as :

$$f(x) = \begin{cases} 1, & \text{if } x \geq q \\ -1, & \text{if } x < q \end{cases}$$

Where 0 is threshold value. The function is also used in single-layer nets to convert the net input to an bipolar output (+ 1, – 1).

3. **Binary step function :** The function is defined as :

$$f(x) = \begin{cases} 1, & \text{if } x \geq q \\ 0, & \text{if } x < q \end{cases}$$

This function is used in single-layer nets to convert the net input to an output that is binary (0 or 1).

4. **Sigmoidal function :** This function is used in back-propagation nets. They are of two types :

(i) **Binary sigmoidal function :** It is also called as unipolar sigmoid function or a logistic sigmoid function, and defined as

$$f(x) = \frac{1}{1+e^{-\lambda x}}, \text{ where } \lambda \text{ is steepness parameter}$$

The derivative of f(x) is :

$$f'(x) = \frac{-1}{[1+e^{-\lambda x}]^2} \cdot (-\lambda e^{-\lambda x}) = \frac{\lambda (e^{-\lambda x})}{[1+e^{-\lambda x}]^2}$$

$$= \frac{\lambda [1+e^{-\lambda x} - 1]}{[1+e^{-\lambda x}]^2}$$

$$= \lambda \left\{ \frac{1}{(1+e^{-\lambda x})} - \frac{1}{(1+e^{-\lambda x})^2} \right\}$$

$$= \lambda \left[\frac{1}{(1+e^{-\lambda x})} \left\{ 1 - \frac{1}{(1+e^{-\lambda x})} \right\} \right]$$

$$= \lambda [f(x)(1 - f(x))] = \lambda f(x)[1 - f(x)]$$

Range of this sigmoid function is 0 to 1

[∵ denominator is always greater than or equal to 1]

(ii) Bipolar sigmoid function :

The function is given by

$$f(x) = \frac{2}{1+e^{-\lambda x}} - 1 = \frac{2 - 1 - e^{-\lambda x}}{1+e^{-\lambda x}} = \frac{1 - e^{-\lambda x}}{1+e^{-\lambda x}}$$

where again, λ is steepress parameter and the range is between – 1 and + 1.

Derivative of f(x) is :

$$f'(x) = \frac{[1+e^{-\lambda x}][\lambda e^{-\lambda x}] - (1 - e^{-\lambda x})(-\lambda e^{-\lambda x})}{(1+e^{-\lambda x})^2}$$

$$= \frac{\lambda e^{-\lambda x}[1 + e^{-\lambda x} + 1 - e^{-\lambda x}]}{(1+e^{-\lambda x})^2}$$

$$= \frac{2\lambda e^{-\lambda x}}{(1+e^{-\lambda x})^2} = \frac{\lambda}{2}\left[\frac{4e^{-\lambda x}}{(1+e^{-\lambda x})^2}\right]$$

$$= \frac{\lambda}{2}\left[\frac{(1+e^{-\lambda x})^2 - (1-e^{-\lambda x})^2}{(1+e^{-\lambda x})^2}\right]$$

$$= \frac{\lambda}{2}\left[1 - \left(\frac{1-e^{-\lambda x}}{1+e^{-\lambda x}}\right)^2\right] = \frac{\lambda}{2}[1 - f(x)^2]$$

$$= \frac{\lambda}{2}[(1+f(x))(1-f(x))]$$

$$[\because a^2 - b^2 = (a+(b)(a-b)]$$

Remark

(1) Here, $f(x) = \frac{1-e^{-\lambda x}}{1+e^{-\lambda x}} = \tanh\left(\frac{\lambda x}{2}\right)$

$$\therefore f'(x) = \frac{\lambda}{2}\operatorname{sech}^2\left(\frac{x}{2}\right)$$

$$= \frac{\lambda}{2}\left[1 - \tanh^2\left(\frac{x}{2}\right)\right]$$

$$= \frac{\lambda}{2}\left[\left(1+\tanh\frac{x}{2}\right)\left(1-\tanh\frac{x}{2}\right)\right]$$

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

(2) Sigmoid functions are so-called because their graphs are "S-shaped".

(i) Simple sigmoids defined to be odd, are monotone functions of one variable,

(ii) Hyperbolic sigmoids are subset of simple sigmoids and natural generalisation of the hyperbolic tangent function.

(5) Ramp function

It is defined as $f(x) = \begin{cases} 1, & \text{if } x > 1 \\ x, & 0 \le x \le 1 \\ 0, & x < 0 \end{cases}$

(6) ReLU

ReLU stands for Rectified Linear Unit. Although it gives an impression of a linear function, ReLU has a derivative function and allows for backpropagation while simultaneously making it computationally efficient.

The ReLU function does not activate all the neurons at the same time. The neurons will only be deactivated if the output of the linear transformation is less than 0.

Graphs of activation functions

(1) Linear function

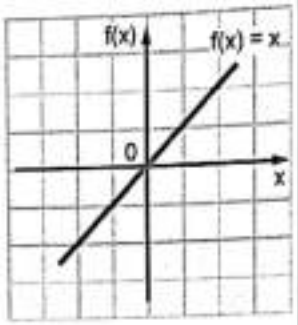


Fig. 4.3.10

(2) Bipolar Step Function

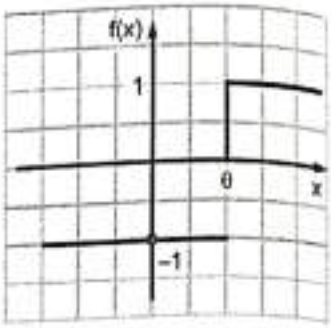


Fig. 4.3.11

(3) Binary-step function

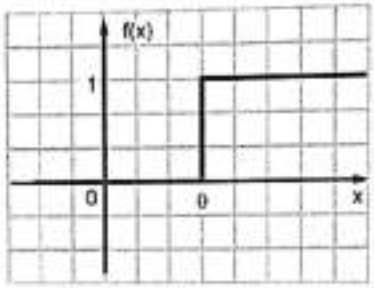


Fig. 4.3.12

(4)

(i) Binary sigmoidal function

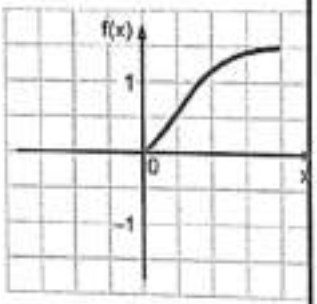


Fig. 4.3.13(i)

(ii) Bipolar sigmoidal function

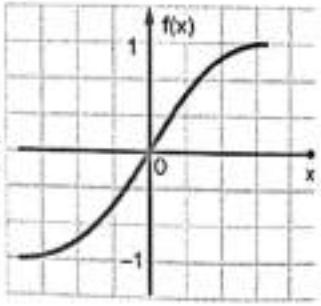


Fig. 4.3.13(ii)

(5) Ramp function

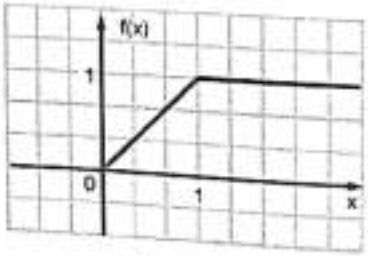


Fig. 4.3.14

(6) ReLu

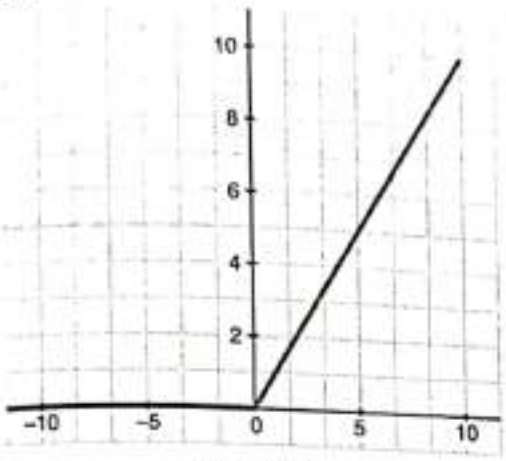


Fig. 4.3.15

Solved Examples

Ex. 4.3.5 : For the network shown in the Fig. Ex. 4.3.5 calculate the net input to the output neuron.

☑ **Soln. :**

The given neural net consists of three input neurons and one output neuron.

The inputs are : $x_1 = 0.3,\quad x_2 = 0.5,\quad x_3 = 0.6$

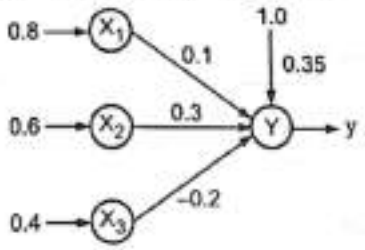


Fig. Ex. 4.3.5

The weights are : $w_1 = 0.2,\quad w_2 = 0.1,\quad w_3 = -0.3$

The net-input is given by,

$$y_{in} = x_1 w_1 + x_2 w_2 + x_3 w_3$$

$$= (0.3)(0.2) + (0.5)(0.1) + (0.6)(-0.3)$$

$$\therefore\ y_{in} = 0.06 + 0.05 - 0.18 = -0.07$$

Ex. 4.3.6 : Obtain the output of the neuron Y for the network shown in the Fig. 4.3.6, using activation functions as: (i) binary sigmoidal and (ii) bipolar sigmoidal.

☑ **Soln. :**

▶ **Step (I) :**

(i) The given network has three input neurons with bias and one output neuron.

These form a single – layer network.

(ii) The inputs are : $x_1 = 0.8\ ;\ x_2 = 0.6\ ;\ x_3 = 0.4$

and weights are $w_1 = 0.1\ ;\ w_2 = 0.3\ ;\ w_3 = -0.2$

and bias is $b = 0.35$ (its input is always 1)

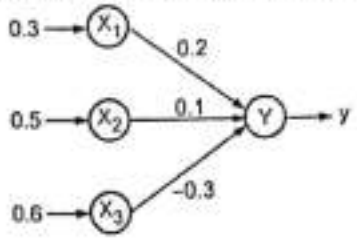


Fig. Ex. 4.3.6

▶ **Step (II) : Now the net input to the output neuron is**

$$y_{in} = b + \sum_{i=1}^{3} x_i w_i;$$

(∵ only 3 input neurons are given)

$$= b + x_1 w_1 + x_2 w_2 + x_3 w_3$$

$$= 0.35 + (0.8)(0.1) + (0.6)(0.3) + (0.4)(-0.2)$$

$$= 0.35 + 0.08 + 0.18 - 0.08 = 0.53$$

▶ **Step (III) : (i) Binary sigmoidal activation function given by,**

$$y = f(y_{in}) = \left(\frac{1}{1+e^{-y_{in}}}\right) = \frac{1}{1+e^{-0.53}} = 0.625$$

(ii) bipolar sigmoidal activation functions is given by,

$$y = f(y_{in}) = \left(\frac{2}{1+e^{-y_{in}}}\right) - 1$$

$$= \left(\frac{2}{1+e^{-0.53}}\right) - 1 = 0.259$$

Ex. 4.3.7 : Calculate the output of the neuron Y for the net given below. Use binary and bipolar sigmoidal activation functions :

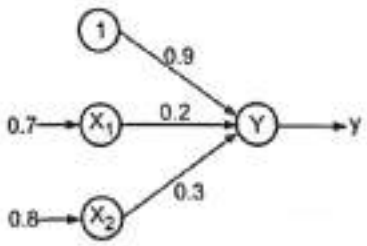


Fig. Ex. 4.3.7

☑ **Soln. :**

▶ **Step (I) :**

(i) The given network has two input neurons with bias and one output neuron.

These form a single-layer network.

(ii) The inputs are : $x_1 = 0.7$; $x_2 = 0.8$

and weights are : $w_1 = 0.2$; $w_2 = 0.3$ and

bias = $b = 0.9$ (its input is always 1)

▶ **Step (II) : Now the net input to the output neuron is**

$$y_{in} = b + \sum_{i=1}^{2} x_i w_i ;$$

(∵ only 2 input neurons are mentioned)

$$= b + (x_1 w_1 + x_2 w_2)$$

$$= 0.9 + [(0.7)(0.2) + (0.8)(0.3)]$$

$$= 0.9 + 0.14 + 0.24 = 1.28 \quad ...(i)$$

▶ **Step (III) : (i) Binary sigmoidal function is given by**

$$y = f(y_{in}) = \frac{1}{1 + e^{-y_{in}}}$$

$$= \frac{1}{1 + e^{-1.28}} = \frac{1}{1 + 0.278} = 0.7824$$

(ii) Bipolar sigmoidal activation function is,

$$y = f(y_{in}) = \left[\frac{2}{1 + e^{-y_{in}}}\right] - 1$$

$$= \left[\frac{2}{1 + e^{-1.28}}\right] - 1 = 0.5649$$

Ex. 4.3.8 : For the network shown in the Fig. Ex. 4.3.8, calculate the output Y for the network using activation function as :

(i) binary sigmoid and (ii) bipolar sigmoid

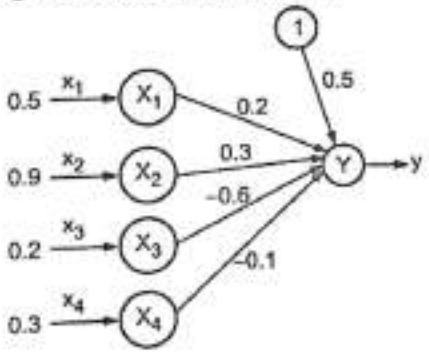


Fig. Ex. 4.3.8

☑ **Soln. :**

▶ **Step (I) : First we find y_{in}.**

Now, $y_{in} = b + x_1 w_1 + x_2 w_2 + x_3 w_3 + x_4 w_4$ ...(i)

Where

$x_1 = 0.5$, $x_2 = 0.9$, $x_3 = 0.2$, $x_4 = 0.3$

$w_1 = 0.2$, $w_2 = 0.3$, $w_3 = -0.6$, $w_4 = -0.1$

and $b = 0.5$ $(x_0 = 1)$

From Equation (i),

$$y_{in} = 0.5 + (0.5)(0.2) + (0.9)(0.3) + (0.2)(-0.6) + (0.3)(-0.1)$$

$$= 0.5 + 0.1 + 0.27 - 0.12 - 0.03$$

$$= 0.87 - 0.15 = 0.72$$

$$\therefore y_{in} = 0.72$$

▶ **Step (II) : (i) to Find y for binary sigmoidal activation function**

$$y = f(y_{in}) = \frac{1}{1 + e^{-y_{in}}} = \frac{1}{1 + e^{-0.72}} = 0.6726$$

(ii) For bipolar sigmoidal activation function

$$y = f(y_{in}) = \left(\frac{2}{1 + e^{-y_{in}}}\right) = \left(\frac{2}{1 + e^{-0.72}}\right) - 1$$

$$= 0.3452$$

Ex. 4.3.9 : Logistic function is given by,

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

Show that the derivative of f(v) w.r.t. v is given by,

$$\frac{df}{dv} = af(v)[1 - f(v)]$$

What is the value of this derivative at the origin ?

☑ **Soln. :**

▶ **Step (I) : We have, $f(v) = \frac{1}{1 + \exp(-av)}$**

Different w.r.t. v ; we get

$$f(v) = \frac{-1}{[1 + \exp(-av)]^2} [\exp(-av)(-a)]$$

$$= \frac{a \exp(-av)}{[1 + \exp(-av)]^2}$$

$$= \frac{a}{[1 + \exp(-av)]} \frac{\exp(-av)}{[1 + \exp(-av)]}$$

$$= a f(v) \left[\frac{\exp(-av)}{[1 + \exp(-av)]}\right]$$

$$= a\,f(v)\left[1 - 1 - \frac{\exp(-av)}{1+\exp(-av)}\right]$$

$$= a\,f(v)\left[1 - \left\{1 + \frac{\exp(-av)}{1+\exp(-av)}\right\}\right]$$

$$= a\,f(v)\left[1 - \left\{\frac{1+\exp(-av)-\exp(-av)}{1+\exp(-av)}\right\}\right]$$

$$= a\,f(v)\left[1 - \left\{\frac{1}{1+\exp(-av)}\right\}\right]$$

$\therefore f'(v) = a\,f(v)\,[1 - f(v)]$

▶ **Step (II) : As $V \to 0$; $\exp(-av) = \exp(0) = 1$**

$\therefore f(v) = \frac{1}{1+1} = \frac{1}{2}$

$\therefore f'(v)$ at $v = 0$ is,

$$= a \cdot \frac{1}{2}\left[1 - \frac{1}{2}\right] = \frac{9}{4}$$

Ex. 4.3.10 : An odd sigmoid function is given by

$$f(v) = \frac{1-\exp(-av)}{1+\exp(-av)} = \tan h\left(\frac{av}{2}\right).$$

Show that the derivative of f(v) w.r.t. V is given by,

$$\frac{df}{dv} = \frac{a}{2}[1 - f^2(v)].$$

What is the valve of this derivative at the origin ? Suppose that the slope parameter a is made infinitely large. What is the resulting form of f(v).

☑ **Soln. :**

▶ **Step (I) : For convenience we take, $f(v) = \tan h\left(\frac{av}{2}\right)$**

Differentiating w.r.t. V;

$$\frac{df}{dv} = \operatorname{sech}^2\left(\frac{av}{2}\right)\cdot\frac{a}{2} = \frac{a}{2}\left[1 - \tan h^2\left(\frac{av}{2}\right)\right]$$

$[\because \operatorname{sech}^2 x = 1 - \tanh^2 x]$

$$= \frac{a}{2}[1 - f^2(v)]$$

$$\therefore \frac{df}{dv} = \frac{a}{2}[1 - f^2(v)] \qquad ...(i)$$

▶ **Step (II) : We have $f(v) = \tanh\left(\frac{av}{2}\right)$**

At $v = 0$, $f(0) = \tan h(0) = 0$. $\therefore$ At origin, from Equation (i), $\frac{df}{dv} = \frac{a}{2}[1 - 0] = \frac{a}{2}$

As $a \to \infty$, $\tan h(\infty) = 1$. $\therefore$ As $a \to \infty$, $f(v) \to 1$.

Ex. 4.3.11 : The algebraic sigmoid function is given by,

$$f(v) = \frac{v}{\sqrt{1+v^2}}$$

Show that the derivative of f(v) w.r.t. v is given by,

$$\frac{df}{dv} = \frac{\phi^3(v)}{v^3}$$

What is the value of this derivative at origin ?

☑ **Soln. :**

▶ **Step (I) :**

Differentiating $f(v) = \frac{v}{\sqrt{1+v^2}}$; we get

$$f'(v) = \left[\frac{\sqrt{1+v^2}\cdot 1 - v\cdot\frac{1\cdot 2v}{2\sqrt{1+v^2}}}{(1+v^2)}\right]$$

$$= \left[\frac{\sqrt{1+v^2} - \frac{v^2}{\sqrt{1+v^2}}}{(1+v^2)}\right] = \left[\frac{(1+v^2)-v^2}{(1+v^2)^{3/2}}\right]$$

$$\therefore f'(v) = \frac{1}{(1+v^2)^{3/2}} \qquad ...(i)$$

$$\therefore f'(v) = \frac{1}{v^3}\left[\frac{v^3}{(1+v^2)^{3/2}}\right] = \frac{1}{v^3}\left[\frac{v}{(1+v^2)}\right]^3$$

$$= \frac{1}{v^3}[f(v)]^3 = \frac{f^3(v)}{v^3} \qquad ...(ii)$$

▶ **Step (II) : As $v \to 0$ $f'(v) = \frac{1}{(1+v^2)^{3/2}} \to 1$.**

$\therefore$ At origin, $f'(v) = 1$.

4.4 TYPES OF DEEP LEARNING NETWORKS

Following are the different types of the Deep learning networks:

(1) Feed-forward neural network
(2) Radial basis function neural network
(3) Multi-layer perceptron
(4) Convolution neural network (CNN)
(5) Recurrent neural network (RNN)
(6) Modular neural network
(7) Sequence to sequence models.

1. Feed Forward Neural Networks (FFNN)

- A feed forward neural network is an artificial neural network wherein connections between the nodes do not form a cycle.
- As such it is different from its descendent : random neural network. The feed forward neural network was the first and simplest type of neural network device.
- In an artificial feed forward network (FFN), information always moves in one direction, i.e. it never goes backwards.
- A (FNN) is an artificial neural network (ANN). FFNN is a biologically inspired classification algorithm. It contains organised layers, consisting of a number of simple neuron – like processing units.
- Every unit in a layer is connected with all the units in the previous layer. These connections are not all equal. Each connection may have a different weight.
- The weights on these connections encode the knowledge of a network. Often the units in a neural network are called 'nodes'.
- Data enters at the inputs and passes through the network, layer by layer, until it arrives at the outputs.
- When it acts as a classifier, there is no feedback between layers. Hence they are called as feed forward neural networks.

2. Radial basis Functional Neural Network

- A radial basis function is a real valued function whose value depends only on the distance between the input and **some fixed point**, either the origin.
- Radial Basis Neural Network (RBNN) transforms the input signals into another form, which can then be fed into network to obtain linear separability.
- If fixed point is origin, then radial basis function is $\psi(x) = \hat{\psi}(\|x\|)$ and if the fixed point is some other point say 'C' then

$$\psi(x) = \hat{\psi}(\|x - c\|)$$

- Any function that satisfies the property $\psi(x) = \hat{\psi}(\|x - c\|)$ is called radial basis function (RBF).
- R.B.F. networks are implemented to resolve domain problems having a training data set, that is quite small.

3. Multi-layer Perceptron

- A multi-layer perceptron (MLP) is a class of feed forward algorithm neural network. The terms multiple-layer perceptron consists of networks compose of multiple layers of perceptron. MLP consists of three types of layers- the initial layer, the output layer and hidden layer.
- The input layer receives the input signal to be processed. And the classification is performed by output layer. The (number of) hidden layers that are placed in between the input and output layers carry the computational work of MLP.
- Similar to feed forward network (FFN) in MLP the data flows in the forward direction form input to output layer. The neurons in the MLP are trained with the back propagation algorithm. The major use of MLP are pattern classification and recognition.
- Multilayer perceptron and CNN are two fundamental concepts in machine learning.

4. Convolution Neural Network (CNN)

- A CNN is a types of artificial neural network used in major recognition. CNNs are a category of neural networks that have proven very effective in areas such as image recognition and classification.
- Convolutional networks have been successful in identifying faces, objects and traffic signs apart from powering vision in robots and self-driving cars.
- CNNs are important tools for most machine learning practitioners today.
- CNN is a deep learning neural network designed for processing structured arrays of data such as images.

5. Recurrent Neural Network (RNN)

- Recurrent neural network is a class of artificial neural network where connections between nodes form a directed graph along a temporal sequence. This makes it to show temporal dynamic behaviour.
- It uses sequential data or time series data. These data are commonly used for ordinal or temporal problems, such as languages translation, natural language processing, speech recognition and image captioning. they are incorporated into popular applications such as voice search and Google translation.

- Like CNNs, recurrent neural networks utilise training data by learning. They are distinguished by their 'memory' as they take information from prior inputs to influence the current input and output.
- While traditional deep neural networks assume that inputs and outputs are independent of each other. The output of recurrent neural networks depends on the prior elements within the sequence, while future events would be also helpful in determining the output of a given sequence, unidirectional recurrent neural networks cannot account for these events in their predictions.

6. Modular Neural Network (MNN)

- Modular neural network is an artificial neural network characterised by a series of independent neural network moderated by some intermediary.
- Each independent neural network serves as a module and operates on separate inputs to accomplish the task, that network is suppose to execute. MNNs use a number of neural network for problem solving.
- Here the various neural networks behave as modules to solve a part of the problem. The whole task of generating the final output of the system after dividing the problems into various modules and also integrating the responses of the modules is done by an integrator.

There are two major models :

- The first model would cluster the entire input space and the other model would makes different neural networks work over the same problem. here we use a 'response integration technique' for the final output of the system.

7. Sequence to Sequence Models

- Sequence to sequence leaning is about training models to convert sequences from one domain to sequences into another domain, (e.g. some sentences to be translated from English to German).
- Sequences to sequences is a family of machines learning approaches used for layer processing. Applications include language translation, image captioning, conversational models and text summarization.
- In inference mode, when we want to decode unknown input sequences, we go through slightly different process :
  (1) Encode the input sequence into state vectors
  (2) Start with a target sequence, say to size 1 i.e., just the start-of-sequence character.
  (3) Feed the state vector and 1-character target sequence to the decoder to produce predictions for the next character.
  (4) Sample the next character using these predictions (we simply use 'arg max ').
  (5) Append the sampled character to the target sequence.
  (6) Repeat until we generate the end of sequence character or we hit character limit.
  (7) And the sequence is encoded.
- The three types of neural networks in deep learning such as convolution neural networks (CNN), Recurrent neural networks (RNN) and ANN etc. are changing the way we interact with the world.
- These different types are the base of deep learning revolution, like powering applications, unmanned aerial vehicles, self-driving cars, speech recognition etc.

4.5 CLASSES OF NEURAL NETWORKS

According to their learning mechanism, Neural Network are classified into several classes. We identify three fundamentally different classes of networks. All these three classes employ the digraph structure for their representation. The arrangement of neurons to form layers to from layers and the connection pattern formed within and between layers is called **network architecture.** There exist five basic types of neuron connection architectures. They are :

1. Single-layer feed-forward network
2. Multilayer feed-forward network;
3. Single node with its own feedback
4. Single-layer recurrent network
5. Multilayer recurrent network

1. Single Layer Feed Forward Network

(1) This type of network comprises two layers, namely the **input layer** and the **output layer.**

(2) The **input layer neurons** receive' the input signals,

(3) The **output layer neurons** receive the output signals.

(4) The synaptic links carrying the weights connect every input neuron to the output neuron but not conversely.

(5) This network is called as feed forward in type or acyclic in nature.

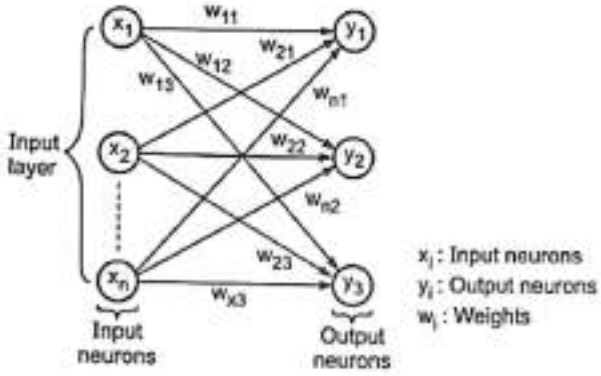


Fig. 4.5.1 : Single layer feedforward network

(6) The network is termed as single layer since the alone output layer, alone which performs computation.

(7) Even though there are two layers, it is called as single layer because of output layer.

(8) The input layer merely transmits the signals to the output layer.

2. Multilayer Feedforward Network

(1) As the name indicates, this network is made up of multiple layers.

(2) It has intermediary layers besides possessing input and output layers. These layers are called as **hidden layers.**

(3) The computational units of the hidden layer are called as hidden neurons or hidden units.

(4) The hidden layers adds in performing intermediary computations and then input layers are directed to the output layers.

(5) The weights on the links of input layer neurons and hidden layers are called as **Input-hidden layer weights.**

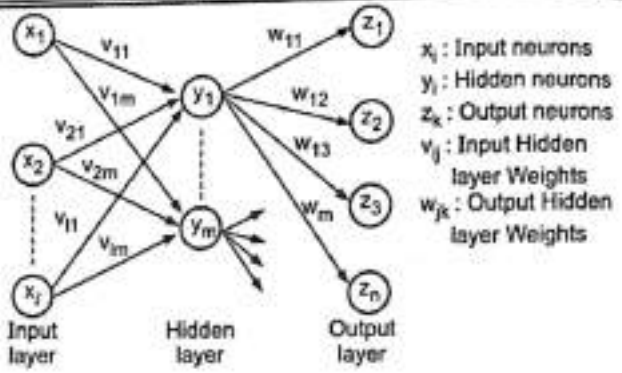


Fig. 4.5.2 : A multilayer feed forward network

3. Single Node with its Own Feedback

- When outputs can be directed back as inputs to the same layer or preceding layer nodes, then it results in feedback networks.
- If the feedback of the output of the processing element is directed back as input to the processing element in the same layer then it is called lateral feedback.

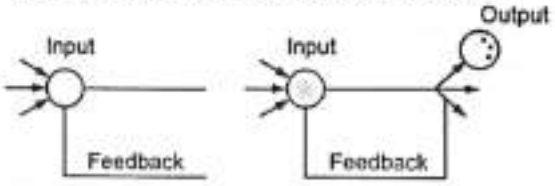


Fig. 4.5.3

4. Single Layer Recurrent Network

- Recurrent networks are feedback networks with closed loop.
- Fig. 4.5.4 shows a single-layer network with a feedback connection in which a processing elements output can be directed back to the processing element itself or to the other processing element or to both.

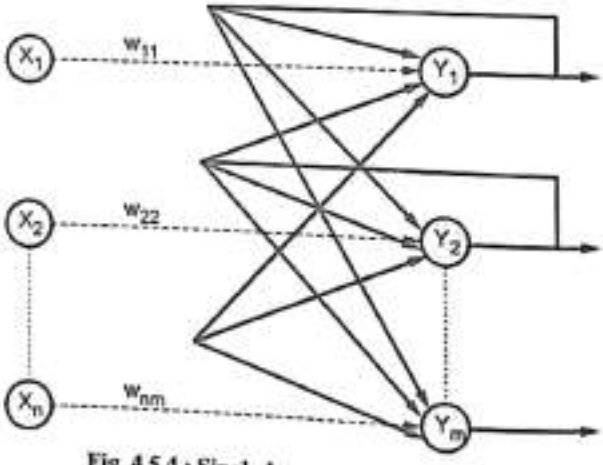


Fig. 4.5.4 : Single-layer recurrent network

5. Multilayer Recurrent Network

- The main feature of a multiplayer recurrent network is its hidden state, which captures information about a sequence.
- To form a multilayer recurrent network, a processing element output can be directed back to the nodes in a preceding layer. Also, in these networks, a processing element output can be directed back to the processing element itself and to other processing elements in the same layer.

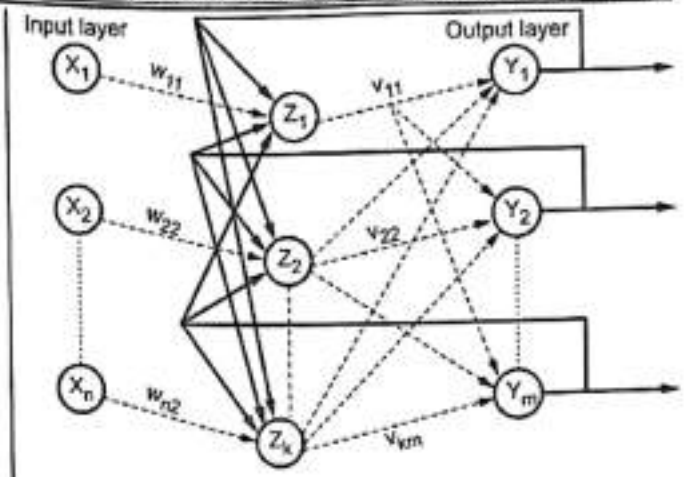
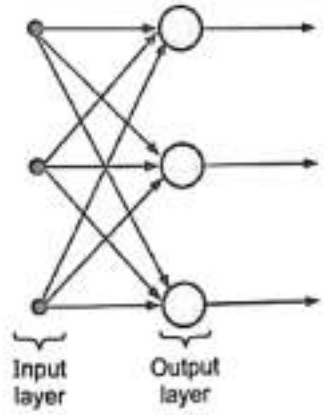
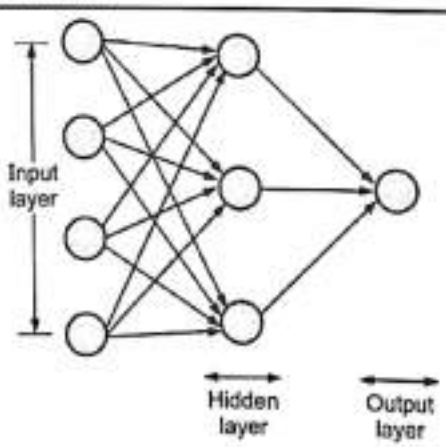


Fig. 4.5.5 : Multiplayer recurrent network

☞ **Difference between Multi-Layer Feed Forward Neural Network and Signal Layer Feed Forward Neural Network**

Sr. No.	Single layer feed forward Neural Network	Multi-layer feed forward Neural Network
1.	Formation of layer is done by processing and combining elements with other processing elements.	Multi-layer is formed by interconnection of several layers.
2.	There is linking between input and output layer.	Between input and output layers, there are multiplelayers, which are known as hidden layers.
3.	Inputs are connected to the processing nodes with various weights resulting series of output one for node.	Input layers receive input and stores input signal and output layer generates output.
4.	There is nohidden layer.	There are several hidden layers in a network.
5.	In certain applications, it is not efficient.	More hidden layers create complexity of network, but the output is efficient.
		

Limitations of Neural Networks

1. The NN needs training to operate.
2. The architecture of a neural network is different from the architecture of microprocessors. Hence, emulation is necessary.
3. Requires high processing time for large neural networks.

4.6 CONVOLUTIONAL NEURAL NETWORK

A **Convolutional Neural Network (ConvNet/CNN)** is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other.

Convolution is a **mathematical operation**. In mathematics, especially in Fourier, Laplace, Z-transforms, convolution operation paves the way to find the inverse transforms. The concept of convolution operates on two functions f and g that produces a third function (f * g), in the transformed domain.

The term convolution refers to both the **result function and the method of evaluating** it. It is **defined** as the **integral of product of two functions**, in which one function is shifted. The integral is evaluated for all values of shift, and gives the convolution function.

Convolutional network is a **special class** of **multilayer perceptrons,** designed to recognise two-dimensional shapes with a high degree of accuracy to skewing, scaling and to translation and other forms of distortion. This task is achieved by a supervised manner which includes the following forms of constraints :

(1) Each neuron extracts local feature from the previous layer. Convolution layers apply a convolution operation to the input passing the result to the next layer. A convolution converts all the pixels in its receptive field into a single value. The final output of the convolutional layer is a vector.

- Convolutional layers are the major building blocks used in convolution neural networks.
- In convolution layer, a neuron is only connected to a local area of input.
- A convolution layer contains units whose receptive fields cover a patch of receptive layer.
- The convolution layer is the core building block of a convolution network.

(2) Each computational layer is composed of feature maps in the form of a plane in which each individual neuron shares the same synaptic weights. It has the following beneficial effects :

(i) Shift invariance through the use of **convolution** with a **kernel** of small size (and using sigmoid functions).

(ii) Reduction in the number of free parameters.

(3) **Subsampling** : Each convolutional layer performs local averaging and subsampling. It reduces the sensitivity of the feature maps distortion.

- In convolutional network, all weights in all layers are learned through training. The network learns to extract its own features automatically.
- In deep learning, a convolutional neural network (CNN) is an artificial neural network, applied to analyze visual imagery. They are also called as shift invariant or space invariant artificial neural networks (SIANN). Convolutional neural networks are equivariant and not invariant to translation.
- They have applications in image and video recognition, image segmentation, image classification, medical image analysis, image and video recognition, natural language processing brain-computer interfaces, and financial time series.
- CNNs are regularized versions of multilayer perceptron which are fully connected layers, i.e. each neuron in one layer is connected to all neurons in the next layer.

These days, deep learning is the threshold of many advanced technological applications, from self-driving cars to art and music. Deep learning enables the computer to build complex concepts from simpler concepts. Very soon we shall see that deep learning will be extended to applications that enable machines to thinks on their own.

- A Convolution Neural Network (CNN) is a **feed-forward neural network (FNN)**. So far CNNs have established extraordinary performance in image search services, voice recognition and natural language processing (NLP).
- We have already seen that a regular multilayer perceptronon gives good result for small images. But it breaks down for larger images.
- For example, if the first layer has 1000 neurons, then there will be 10 million connections.
- CNNs solve this problem using partially connected layers. CNN has fewer parameters than a deep neural network (DNN), hence it requires less training data.
- In 'convolution neural network' convolution is employed in the network. In simple terms, two images which can be represented as matrices are multiplied to give an output that is used to extract features from the image.
- In addition, CNN can detect any particular feature anywhere on the image, but a Deep Neural Network(DNN) can detect it only in that particular location.
- Since images have generally repetitive features, CNNs can generalize much better than DNNs for image processing tasks such as classification.
- A CNNs architecture has prior knowledge of how pixels are organized.
- Lower layers identify features in small areas of the images, while higher layers combine features of lower-layer into larger features. In case of DNNs, this doesn't work.
- Thus, in short, CNN is a class of neural networks that specializes in processing data that has a grid-like topology, such as an image.
- Each neuron works in its own receptive field and is connected to other neurons in a way that they cover entire visual field.
- A CNN design begins with feature extraction and finishes with classification.
- Deep Neural Networks that can recognize and classify particular features from images and are widely used for analyzing visual images.

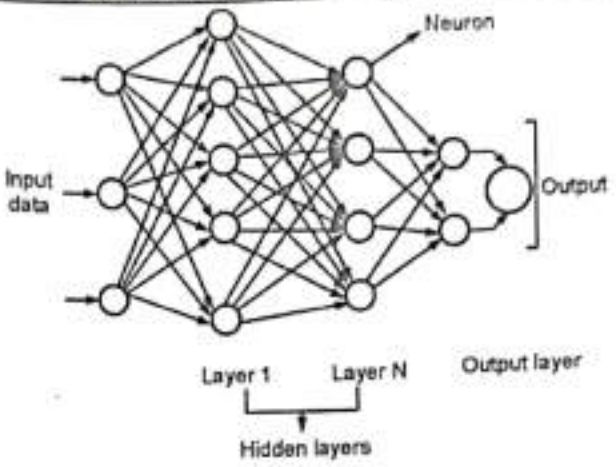


Fig. 4.6.1 : DNN

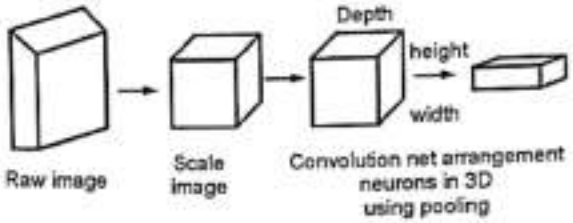


Fig. 4.6.2 : CNN

In Fig. 4.6.1, there is DNN neural network. On the right of Fig. 4.6.2, a convolution net arranges its neurons in 3-dimensions, as seen in one of the layers.

4.6.1 Architecture of CNN

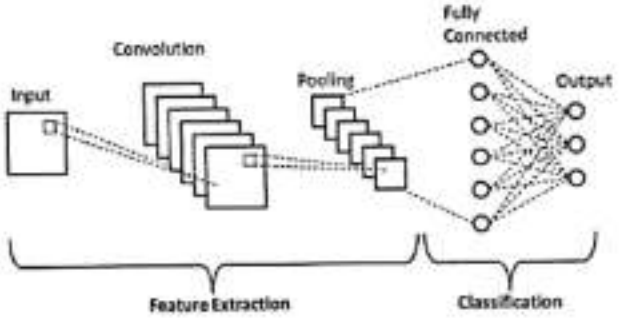


Fig 4.6.3 : Basic Architecture of CNN

A conventional neural network consists of an input layer, hidden layers and output layer (as shown in the Fig. 4.6.3). The middle layers are called as hidden layers because their inputs and outputs are governed by activation function and convolution.

In CNN, the input is a tensor with a shape:

(Number of inputs) × (input height) × (input width) × (input channels).

After passing through a convolutional layer, the image becomes a features map, also called an activation map, with shape:

(number of inputs) × (feature map height)

× (feature map width) × (feature map channels).

The input is convolved in convolutional layers and the result is forwarded to the next layer. Each convolutional neuron processes data only for its receptive field.

Fully connected feed forward neural networks architecture is impractical for larger inputs. It requires a very high number of neurons. For example, a fully connected layer of size 100 × 100 has 10,000 weights for each neuron in the second layer. Instead using convolution, a 5 × 5 filing regions, only 25 learnable parameters are required. Also, convolutional neural networks are ideal for data with a grid-like topology since separate features are taken into account during convolution.

Layers in CNN

- **Input Layer :** The input layer passes the data directly to the first hidden layer. Here the data is multiplied by the first hidden layer's weights. Then the input layer passes the data through the activation function then it passes on.
- **Convolutional Layer :** The convolution layer is the core building block of the CNN and it performs convolution. It carries the main portion of the network's computational load.

  This layer performs a dot product between two matrices, where one matrix is the set of learnable parameters otherwise known as a kernel, and the other matrix is the restricted portion of the receptive field. The kernel is spatially smaller than an image but is more in-depth. This means that, if the image is composed of three (RG(B) channels, the kernel height and width will be spatially small, but the depth extends up to all three channels.

  During the forward pass, the kernel slides across the height and width of the image-producing the image representation of that receptive region. This produces a two-dimensional representation of the image known as an activation map that gives the response of the kernel at each spatial position of the image. The sliding size of the kernel is called a **stride**.

- The following example illustrates how convolution layer works :

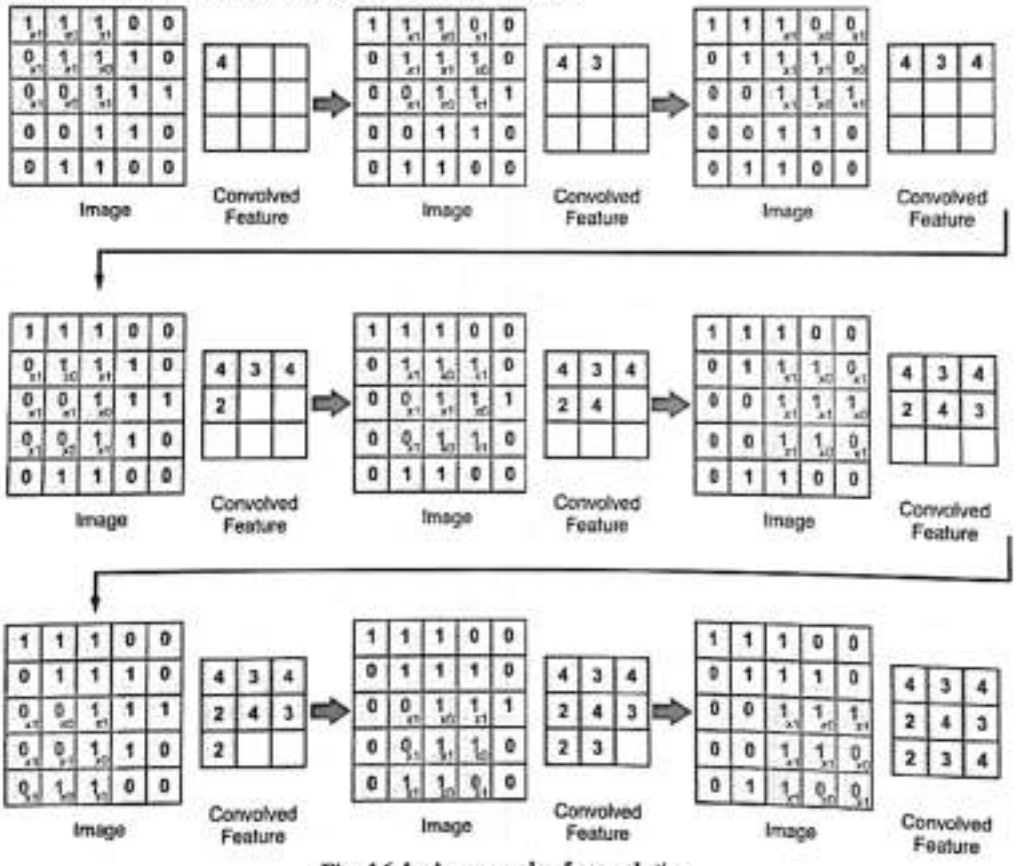


Fig. 4.6.4 : An example of convolution

- In the above example, we have taken 5x5x1 input image, I.
- The element involved in carrying out the convolution operation in the first part of a Convolutional Layer is called the **Kernel/Filter, K**. We have selected K as a 3x3x1 matrix.

$$K = \begin{matrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{matrix}$$

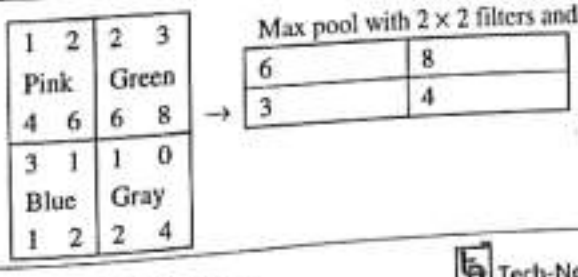
- The Kernel shifts 9 times because of Stride Length = 1 (Non-Strided), every time performing a matrix multiplication operation between K and the portion P of the image over which the kernel is hovering.
- The filter moves to the right with a certain Stride Value till it parses the complete width.
- Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.
- Thus, the objective of the Convolution Operation is to extract the high-level features such as edges, from the input image.
- There is no need to limit ConvNets to just one Convolutional Layer.
- The first ConvLayer is traditionally responsible for capturing Low-Level information such as edges, colour, gradient direction, and so on.
- With the addition of layers, the architecture adjusts to the High-Level characteristics as well, giving us a network that understands the photos in the dataset in the same way that we do.
- The operation produces two types of results: one in which the dimensionality of the convolved feature is lowered when compared to the input, and the other in which the dimensionality is either increased or unchanged.
- This is accomplished by using Valid Padding in the first case and Same Padding in the second.

Pooling Layer

- In most cases, a Convolutional Layer is followed by a Pooling Layer. Pooling is basically 'downscaling' the image obtained from the previous layers.
- It can be compared to shrinking an image to reduce the pixel density.
- The pooling layer summarizes the features present in a region of the feature map generated by a convolutional layer.
- The primary aim of this layer is to decrease the size of the convolved feature map to reduce the computational costs.
- It reduces the number of parameters to learn and the amount of computation performed in the network.
- This is performed by decreasing the connections between layers and independently operates on each feature map. Depending upon method used, there are several types of Pooling operations.
- There are two types of widely used pooling in CNN layer.
  1. Max pooling
  2. Average Pooling

1. Max-pooling

- The popular kind of pooling is **Max-pooling**. Suppose we want to pool by a ratio of 2. It implies that the height and width of your image will be half of its original value.
- So we have to compress every 4 pixels (a 2 × 2 gri(d) and map it to a new single pixel with **losing** the **important** data from the **missing** pixels.
- Max pooling is done by taking the largest value of these 4 pixels. Thus one new pixel represents 4 old pixels by using the largest value of these 4 pixels. This is repeated for every group of 4 pixels throughout the whole image.



Max pool with 2 × 2 filters and stride 2.

6	8
3	4

- Here the largest pixels are 6,8,3 and 4. Here the quality of an image is reduced to avoid computational load on the system. By reducing the quality of the image, we can increase the 'depth' of the layer, to have more features in the reduced image.
- Reducing the image size helps the convolution layer after the pooling layer to look for 'higher level features'. It means that the convolution layer looks at the picture as a whole.

2. Average pooling

- **Average pooling** is different from **Max Pooling**. Average pooling retains 'less important' information about the elements of a block, or pool.
- But Max pooling chooses the maximum value and discards less important values.
- But 'less important' is also sometimes useful in a variety of situations.

4	3	1	5
1	3	4	8
4	5	4	3
6	5	9	4

Av [4, 3, 1, 3] = 2.75

4	3	1	5		
1	3	4	8	→	2.8 4.5
4	5	4	3		5.3 5.0
6	5	9	4		

Fig 4.6.3 : Average Pooling

Fully Connected Layer

- The FC layer helps to map the representation between the input and the output.
- The Fully Connected (F(C) layer consists of the weights and biases along with the neurons and is used to connect the neurons between two different layers.
- These layers are usually placed before the output layer and form the last few layers of a CNN Architecture.
- In this, the input image from the previous layers is flattened and fed to the FC layer.
- The flattened vector then undergoes few more FC layers where the mathematical functions operations usually take place. In this stage, the classification process begins to take place.
- Output Layer :
- The output layer performs final stage of convolution.
- Each neuron is assigned a receptive field and is assigned the possible characters.

Padding

- In Convolutional Neural Networks (CNN), padding is a term that refers to the a number of pixels added to an image when it is being processed by the kernel of a CNN.
- For example, if padding in CNN is kept zero, then every pixel value that is added will be of value zero.
- Padding is simply a process of adding layers of zeros to our input images so as to avoid the problems mentioned above.
- CNNs are used extensively for tackling problems occurring in image processing and predictive modelling or classification tasks. The main application of CNN is to analyze image data.
- Now, every image in any dataset is a matrix of its pixel values. When working with simple CNN for an image, we get output reduced in size and that is loss of data. And that hinders to obtain a proper result according to our requirements.
- When we do not want the shape or our outputs to reduce in size, the addition of more layers in the data can help to obtain a proper result and that addition can be done by padding.

Types of Padding

There are three types of padding:

(1) Same padding
(2) Valid Padding
(3) Causal padding

(1) Same padding

In this type of padding, the padding layers have zero values in the outer frame of the images or data so the filter we are using can cover the edge of the matrix and it carries the required inference.

(2) Valid padding

- This type of padding is called as no padding. Here we don't apply any padding, but the input gets, fully covered by the filter and so every pixel of the image is valid.
- Valid padding is to be used with Max-pooling layers.
- Here we use every pixel or point value while learning the model as valid pixel. It works on the validation of pixel and not on the size of the input.

(3) Causal padding

- This type of padding works with one-dimensional convolution layers; we can use them majorly in time series analysis.
- Since a time, series is sequential data, it helps in adding zeros at the start of data, and it helps in predicting the values of previous time steps.
- We consider an example of a simplified image to understand the idea of edge detection.
- Here, a 6 × 6 matrix convolved with a 3 × 3 matrix, output is 4 × 4 matrix. recall that if n × n image convolved with f × f kernel or filter then output image is of size.

$(n - f + 1) \times (n - f + 1)$

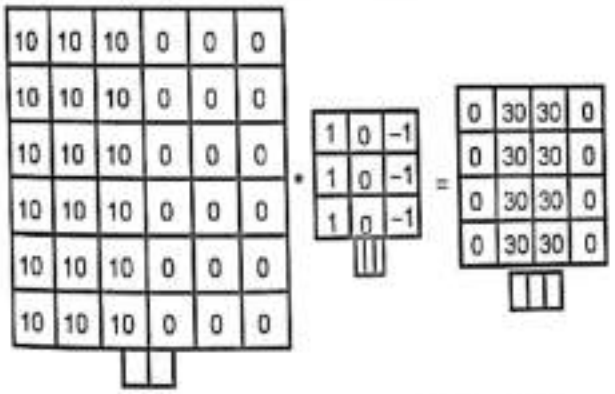


Fig. 4.6.5 : A 6 × 6 image convolved with 3 × 3 filter

As we have already seen, there are two problems with convolution :

1. After multiple convolution operation, our original image becomes small which we don't want to happen.
2. Corner features of any image are not used much in the output

Here padding preserves the size of the original image :

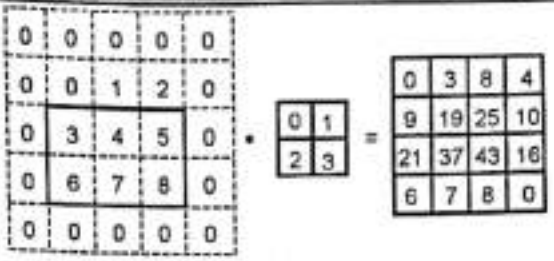


Fig. 4.6.6 : Padded image convolved with 2 * 2 kernel

Hence, if a (n × n) matrix convolved with an (f × f) matrix with padding P then the size of the output image becomes,

$(n + 2P - f + 1) \times (n + 2P - f + 1)$; here P = 1

Remark

- Zero padding is introduced to make the shapes match as required, equally on every side of the input map.
- Valid means no padding.

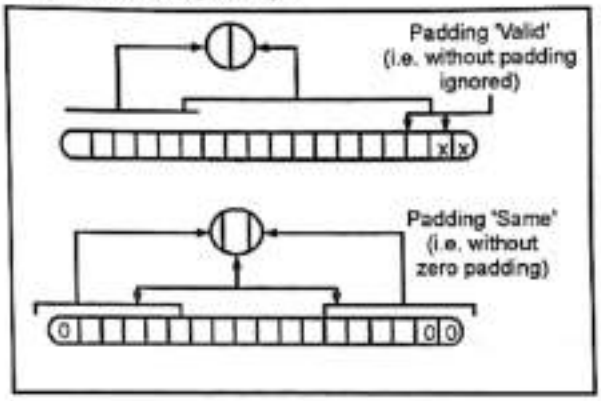


Fig. 4.6.7 : Same versus valid padding with CNN

Problem with Simple Convolution Layer

- A simple CNN of size (n × n) with (f × f) filter/kernel size gives result for output image as

  $(n - f + 1) \times (n - f + 1)$
- For example in any convolution operation with a (8 × 8) image and (3 × 3) filter the output image size will be (6 × 6).
- Thus the output of the layers is shrunk in comparison to the input. Again, the filters we are using may not focus on the corners every time when it moves on the pixels e.g.

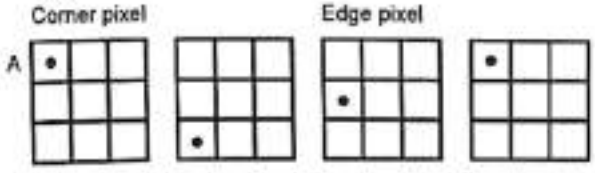


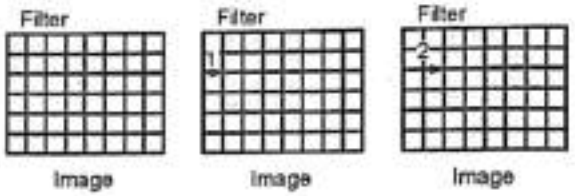
Fig 4.6.8

- The above image is an example of the movement of a filter of size (3 × 3) on an image of size (6 × 6), corner pixel A is coming under the filter in only one movement. This shown that pixel A is misinterpreted.
- This causes loss of information available in the corners and also the output from the layers is reduced and this reduced information may create confusion for next layer. This problem of model can be solved by padding layer.
- The convolution layers reduce the size of the output. So when we want to save the information presented in the corners, we can use padding layers where padding helps by adding extra rows and columns on the outer dimension of the images. So the size of the **input data** will remain similar to the output data.
- Padding basically extends the area of an image in which convolutional neural network processes. The kernel/filter which moves across the image scans each pixels and converts the image into a smaller image.
- Padding is added to the outer frame of the image to allow for more space for the filter to cover in the image. This creates a more accurate analysis of images.

Strided Convolution

- Stride means number of pixel over shifts the input matrix.
- A strided convolution is another type of building block of C, that is used in convolutional neural network.
- Suppose we want to convolve the 7 times 7 image with 3 time 3 filters using a stride of 2.
- Thus stride is the distance between random location where the convolution kernel is applied.
- The whole concept of stride convolution is that we can stride the window over the input vector, matrix or tensor.
- The stride parameter indicates the length of step in the stride. In any framework it is always 1. Of course we can increase the stride-step length in order to save space or cut calculation time.
- We may forgo some information while doing so. We cannot have a stride of 0, this would mean not sliding at all.
- Applying convolution means sliding a kernel over an input signal outputting a weighted sum where the weights are the values inside the kernel.
- If we use a convolution with a (2 × 2) stride, the step is 2 in both x and y direction, followed by non-strided convolution, stride 1, step 1.
- We observe that the output of a convolution with stride 2 halves the width and height of the input, whereas the output of a convolution with stride 1 has width = input ... width – 2 and height = height – 2, since the kernel is 3 × 3.

Stride



Left image : Stride = 0
Middle image : Stride = 1
Right image : Stride = 2

Fig. 4.6.9

Rectified Linear Unit

- In artificial neural networks, the rectifier activation function is an activation function defined as the positive part of it's argument.

$$f(x) = x^{+} = \max(0, x)$$

where x is the input to a neuron. This is also known as a ramp function and is analogous to half wave rectification in electrical engineering.

- A node or unit that implements this activation function is referred to as a rectifier linear activation unit or ReLu.
- In a neural network, the activation function is responsible for transforming the summed weighted input from the node into activation of the node or output for that input.

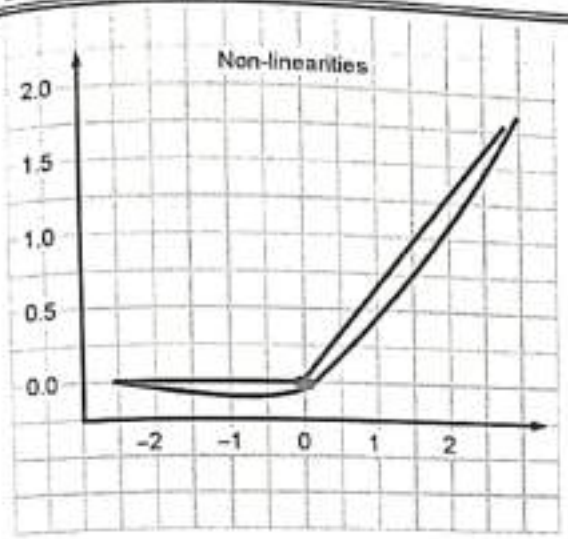


Fig. 4.6.10 : Plot of ReLu rectifier near x = 0 (and GELu) function (Gaussian Error Linear unit)

- The rectified linear artificial function or ReLu is a piecewise linear function that will output the input directly if it is positive, otherwise the output is zero.
- It has become default activation function for many types of neural networks because a model that uses it is easier to train and often achieves better performance.

Here we discussed the rectified linear activation function (ReLu) for deep learning neural networks :

(1) The sigmoid and hyperbolic tangent activation functions cannot be used in networks with many layers because of gradient problems and gradient vanishes. But the rectified linear activation function overcomes the vanishing gradient problems allowing models to learn faster and to perform better.

(2) The rectified linear activation function is default activation when developing multilayer perception and convolutional neural networks.

(3) For a given node, the inputs are multiplied by the weights in a node and summed together. This value is referred to as the summed activation of the node. The summed activation is then transformed via an activation function and defines the specific output or "activation" of the node.

(4) The linear activation functions are the simplex activation function. A network consisting of linear activation functions cannot learn complex mapping functions but it is very easy to train. But linear activation functions predict regression problems, hence they are used in output layer for networks.

(5) On the other hand nonlinear activation functions learn more complex structures in data. The sigmoid and hyperbolic tangent activation functions are used as nonlinear activation functions.

(6) The sigmoid activation function is also called as logistic function in neural networks. The input to the function is transformed into a value between 0.0 and 1.0. Inputs which are larger than 1.0 are regarded as 1.0 similarly values smaller than 0.0 are taken as 0.0. The shape of the function is an S-shape from zero up through 0.5 to 1.0.

(7) Hyperbolic tangent function is similar shaped nonlinear activation function, and it's output values lie between – 1.0 and 1.0. This function has better predictive performance and was easier to train. Thus hyperbolic tangent function's performance is better than logistic sigmoid function.

(8) The sigmoid and tanh functions saturate. Tanh function saturates to 1.0 for larger values and sigmoid function saturate to 0 to – 1.0 for small values.

☞ Advantages of Rectified Linear Unit

(1) Here gradient propagation is better : Very few vanishing gradient problems, compared to sigmoidal activation function. Moreover sigmoidal activation function saturate in both direction. In a randomly initialised network, only bout 50% of hidden units have a non-zero output.

(2) Computation is fast and easy. It involves only addition, multiplication and comparison.

(3) It is scale invariant, i.e.

$$\max(0, ax) = a \max(0, x) \text{ for } a \geq 0$$

(4) Rectifying activation functions were trained to separate specific excitation and unspecific inhibition in the neural networks.

(5) The use of rectifier as a non-linearity enables deep supervised training without requiring unsupervised pre-training. Compared to sigmoid function or other similar activation functions, rectified linear units allow effective training of deep neural architectures on complex database.

Disadvantages of Rectified Linear Unit

(1) The function is differentiable except at zero, and the value of the derivative at 0 is arbitrarily chosen as 0 or 1.

(2) It is unbounded

(3) It is not zero-centered.

(4) Rectified linear unit neurons become sometimes inactive for essentially all inputs. In this state, there is no gradients flow backward, through the neuron, and so the neuron remains in an inactive state and 'dies'. This is called as **'Vanishing gradient'** problem.

(5) In some cases, model capacity gests decreased because large number of neurons in a network get struck in dead states.

This problems happens especially when the learning rate is too high. By using leaky ReLus, the performance is reduced.

Variants : Linear Variants

(I) Leaky ReLu

When the unit is not active, leaky ReLus allow a small, positive gradient :

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ 0.01x, & \text{otherwise} \end{cases}$$

(II) Parametric ReLu

This is an advanced variant that Leaky ReLu. Here the coefficient of leakage is made into a parameter and that is learned along with the other neural network parameters. Here :

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ ax, & \text{otherwise} \end{cases}$$

> **Note :** Note that for $a \leq 1$, this is equivalent to $f(x) = \max(x, ax)$ and has a relation of "maxout" networks.

Non-Linear Variants

(I) Gaussian Error Linear Unit (GELu)

It is a smooth approximation to the rectifier. It has a bump when $x < 0$, and is non-monotonic. It serves as a default, activation for some standard models.

It is defined as : $f(x) = x \cdot \phi(x)$

Where $\phi(x)$ is the cumulative distribution function of the standard normal distribution.

(II) Softplus

A smooth approximation to the rectifiers is the analytic function :

$$f(x) = \log_e (1 + e^x),$$

and it is called as **'Softplus'** or **'SmoothReLu'** function .

If we include sharpness parameters 'K'

$$\text{then } f(x) = \frac{\log (1 + e^{Kx})}{K}$$

$$\text{and } f'(x) = \frac{1}{K}\left[\frac{1}{(1 + e^{Kx})} \cdot Ke^{Kx}\right] = \frac{e^{Kx}}{1 + e^{Kx}}$$

Dividing N and D by e^{Kx}

$$f'(x) = \frac{1}{1 + e^{-Kx}}$$

Thus the logistic sigmoid function is a smooth approximation of the derivative of the rectifier.

The multivariable generalisation of single-variable softplus is the 'Log Sum Exp' function with the first argument kept to zero.

$$LSE_0^+ (x_1, x_2, \ldots, x_n) = LSE (0, x_1, \ldots, x_n)$$
$$= \log (1 + e^{x1} + e^{x2} + \ldots + e^{xn})$$

The 'Log Sum Exp' function is

$$LSE (x_1, x_2, \ldots, x_n) = \log (e^{x1} + e^{x2} + \ldots + e^{xn})$$

It's gradient is the softmax

(III) ELU

ELU is exponential linear unit. Using this mean activations are made closer to zero. And it speeds up learning. And we can have higher classification accuracy than ReLus.

It is defined as : $$f(x) = \begin{cases} x, & \text{if } x > 0 \\ a(e^x - 1), & \text{otherwise} \end{cases}$$

Where $a \geq 0$ and is a hyper-parameter.

The ELU is a smoothed version of ReLu, which has the form

$$f(x) = \max \cdot (-a, x), \quad \text{where } a \geq 0.$$

Regularisation

To prevent overfitting in the training phase, there are different ways of controlling training of CNNs. In particular they are L2/L1 regularisation and max-norm constraints :

(1) **L2 regularisation :** This regularisation is implemented by penalising directly the squared magnitude of all parameters in the objectives. Using the gradient descent parameter, every weight is decayed linearly towards zero by L2 regularisation method.

(2) **L1 regularisation :** Here in this method, we add the term $\lambda|\omega|$ for each weight ω to the objective.

It is also possible to combine the L1 regularisation with L2 regularisation $\lambda_1 |\omega_1| + \lambda_2 |\omega_2|$ and is known as Elastic-net regularisation.

(3) **Max-Norm constraint :** Here we enforce an absolute upper bound on the magnitude of the weight vector for every neuron and use projected gradient descent to enforce the constraint. This is altogether a different form of regularisation.

4.6.2 Types/Architecures of CNNs

There are various architectures of CNNs available which have been key in building algorithms which power and shall power AI as a whole in the foreseeable future. Some of them have been listed below:

1. LeNet	2. AlexNet
3. VGGNet	4. GoogLeNet
5. ResNet	6. ZFNet

1. LeNet

- ann LeCun, Leon Bottou, Yosuha Bengio and Patrick Haffner proposed a neural network architecture for handwritten and machine-printed character recognition in 1990's which they called **LeNet-5**.

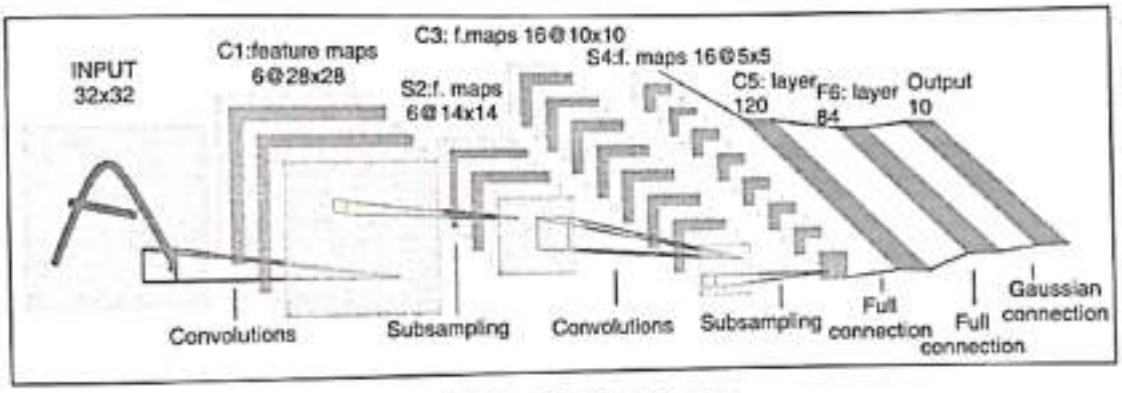


Fig. 4.6.11 : LeNet-5 architecture

- The LeNet-5 architecture consists of two sets of convolutional and average pooling layers, followed by a flattening convolutional layer, then two fully-connected layers and finally a softmax classifier.

(1) **First Layer :** The input for LeNet-5 is a 32×32 grayscale image which passes through the first convolutional layer with 6 feature maps or filters having size 5×5 and a stride of one. The image dimensions changes from $32 \times 32 \times 1$ to $28 \times 28 \times 6$.

(2) **Second Layer :** Then the LeNet-5 applies average pooling layer or sub-sampling layer with a filter size 2×2 and a stride of two. The resulting image dimensions will be reduced to $14 \times 14 \times 6$.

(3) **Third Layer :** Next, there is a second convolutional layer with 16 feature maps having size 5×5 and a stride of 1. In this layer, only 10 out of 16 feature maps are connected to 6 feature maps of the previous layer.

(4) **Fourth Layer :** The fourth layer (S4) is again an average pooling layer with filter size 2×2 and a stride of 2. This layer is the same as the second layer (S2) except it has 16 feature maps so the output will be reduced to $5 \times 5 \times 16$.

(5) **Fifth Layer :** The fifth layer (C5) is a fully connected convolutional layer with 120 feature maps each of size 1×1. Each of the 120 units in C5 is connected to all the 400 nodes (5x5x16) in the fourth layer S4.

(6) **Sixth Layer :** The sixth layer is a fully connected layer (F6) with 84 units.

(7) **Output Layer :** Finally, there is a fully connected softmax output layer $\hat{y}$ with 10 possible values corresponding to the digits from 0 to 9.

Layer		Feature map	Size	Kernel size	Stride	Activation
Input	Image	1	32 × 32	–	–	–
1	Convolution	6	28 × 28	5 × 5	1	tanh
2	Average pooling	6	14 × 14	2 × 2	2	tanh
3	Convolution	16	10 × 10	5 × 5	1	tanh
4	Average Pooling	16	5 × 5	2 × 2	2	tanh
5	Convolution	120	1 × 1	5 × 5	1	tanh
6	FC	–	84	–	–	tanh
Output	FC	–	10	–	–	softmax

Fig. 4.6.12 : Summary of LeNet-5 architecture

2. AlexNet

- **AlexNet** was the first convolutional network which used GPU to boost performance. the authors introduced padding to prevent the size of the feature maps from reducing drastically.
- AlexNet won the 2012 ImageNet competition with a top-5 error rate of 15.3% compared to second place top-5 error rate of 26.2%.
- The input to this model is the images of size 227 * 227 * 3.
- AlexNet architecture consists of 5 convolutional layers, 3 max-pooling layers, 2 normalization layers, 2 fully connected layers, and 1 softmax layer.
- Each convolutional layer consists of convolutional filters and a nonlinear activation function ReLU.
- The pooling layers are used to perform max pooling.
- Input size is fixed due to the presence of fully connected layers.
- The input size is mentioned at most of the places as 224 × 224 × 3 but due to some padding which happens it works out to be 227 × 227 × 3
- AlexNet overall has 60 million parameters.
- AlexNet was trained on a GTX 580 GPU with only 3 GB of memory which couldn't fit the entire network. So the network was split across 2 GPUs, with half of the neurons(feature maps) on each GPU.

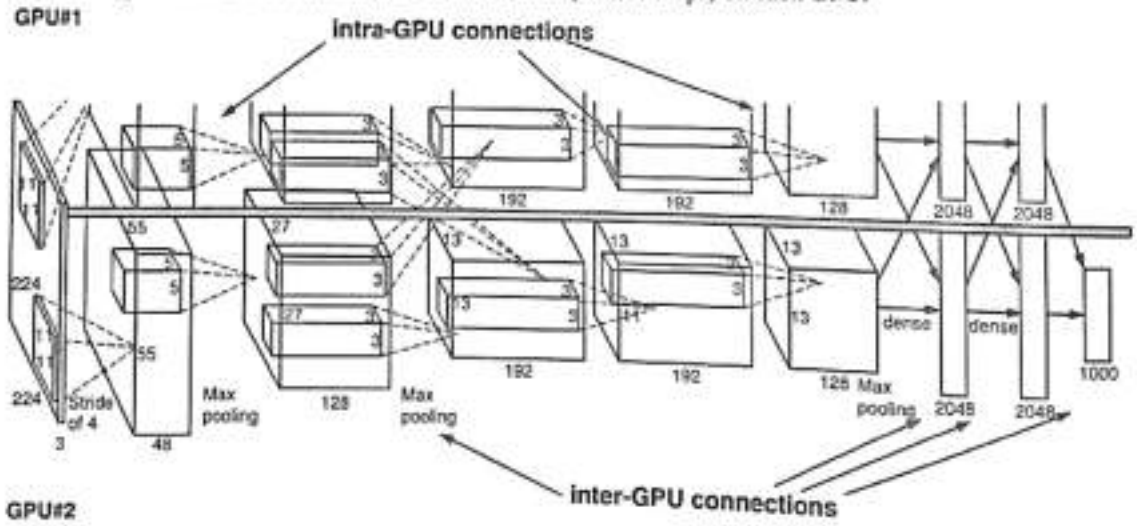


Fig. 4.6.13 : Architecture of AlexNet

Convolution and Maxpooling Layers

- We apply the first convolution layer with 96 filters of size 11 × 11 with stride 4. The activation function used in this layer is relu. The output feature map is 55 × 55 × 96.
- The output size of a convolution layer is calculated as:

  Output = ((Input-filter size)/ stride) + 1
- Also, the number of filters becomes the channel in the output feature map.
- Next, we have the first Maxpooling layer, of size 3*3 and stride 2. Then we get the resulting feature map with the size 27 × 27 × 96.
- After this, we apply the second convolution operation. This time the filter size is reduced to 5 × 5 and we have 256 such filters. The stride is 1 and padding 2. The activation function used is again relu. Now the output size we get is 27 × 27 × 256.
- Again, we applied a max-pooling layer of size 3 × 3 with stride 2. The resulting feature map is of shape 13 × 13 × 256.
- Now we apply the third convolution operation with 384 filters of size 3 × 3 stride 1 and also padding 1. Again the activation function used is relu. The output feature map is of shape 13 × 13 × 384.
- Then we have the fourth convolution operation with 384 filters of size 3 × 3. The stride along with the padding is 1. On top of that activation function used is relu. Now the output size remains unchanged i.e 13 × 13 × 384.
- After this, we have the final convolution layer of size 3 × 3 with 256 such filters. The stride and padding are set to one also the activation function is relu. The resulting feature map is of shape 13 × 13 × 256.

Fully Connected and Dropout Layers

Layer	8 filters /Filter size neurons	Stride	Padding	Size of feature rump	Activation function
Dropout 1	rate = 0.5	–	–	6 × 6 × 256	–
Fully Connected 1	–	–	–	4096	ReLU
Dropout 2	rate = 0.5	–	–	4096	–
Fully Connected 2	–	–	–	4096	ReLU
Fully Connected 3	–	–	–	1004	Soltmax

Fig. 4.6.14 : Summary of Fully Connected and Dropout layers in AlexNet

- After this, we have our first dropout layer. The drop-out rate is set to be 0.5.
- Then we have the first fully connected layer with a relu activation function. The size of the output is 4096. Next comes another dropout layer with the dropout rate fixed at 0.5.
- This followed by a second fully connected layer with 4096 neurons and relu activation.
- Finally, we have the last fully connected layer or output layer with 1000 neurons as we have 10000 classes in the data set.
- The activation function used at this layer is Softmax.

3. VGGNet

- VGGNet is a Convolutional Neural Network architecture proposed by Karen Simonyan and Andrew Zisserman from the University of Oxford in 2014.
- This paper mainly focuses on the effect of the convolutional neural network depth on its accuracy.
- The input to VGG based convNet is a 224 × 224 RGB image.
- Preprocessing layer takes the RGB image with pixel values in the range of 0 – 255 and subtracts the mean image values which is calculated over the entire ImageNet training set.

- The input images after preprocessing are passed through these weight layers.
- The training images are passed through a stack of convolution layers.
- There are total of 13 convolutional layers and 3 fully connected layers in VGG16 architecture.
- VGG has smaller filters (3 × 3) with more depth instead of having large filters.
- It has ended up having the same effective receptive field as if you only have one 7 × 7 convolutional layers.
- Another variation of VGGNet has 19 weight layers consisting of 16 convolutional layers with 3 fully connected layers and same 5 pooling layers.
- In both variation of VGGNet there consists of two Fully Connected layers with 4096 channels each which is followed by another fully connected layer with 1000 channels to predict 1000 labels.
- Last fully connected layer uses softmax layer for classification purpose.

Architecture

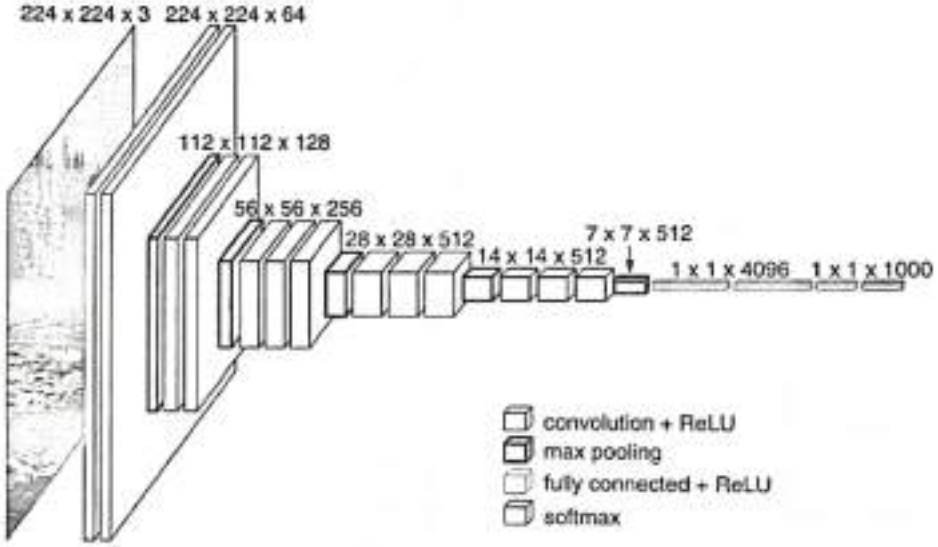


Fig. 4.6.15 : Architecture of VGGNet

- The first two layers are convolutional layers with 3 × 3 filters, and first two layers use 64 filters that results in 224 × 224 × 64 volume as same convolutions are used.
- The filters are always 3 × 3 with stride of 1.
- After this, pooling layer was used with max-pool of 2 × 2 size and stride 2 which reduces height and width of a volume from 224 × 224 × 64 to 112 × 112 × 64.
- This is followed by 2 more convolution layers with 128 filters.
- This results in the new dimension of 112 × 112 × 128.
- After pooling layer is used, volume is reduced to 56 × 56 × 128.
- Two more convolution layers are added with 256 filters each followed by down sampling layer that reduces the size to 28 × 28 × 256.
- Two more stack each with 3 convolution layer is separated by a max-pool layer.
- After the final pooling layer, 7 × 7 × 512 volume is flattened into Fully Connected (F(C) layer with 4096 channels and softmax output of 1000 classes.

4. GoogLeNet

- **GoogleNet (or Inception V1)** was proposed by research at Google (with the collaboration of various universities) in 2014 in the research paper titled "Going Deeper with Convolutions".
- This architecture was the winner at the ILSVRC 2014 image classification challenge. It has provided a significant decrease in error rate as compared to previous winners AlexNet (Winner of ILSVRC 2012) and ZF-Net (Winner of ILSVRC 2013) and significantly less error rate than VGG (2014 runner up).
- This architecture uses techniques such as 1 × 1 convolutions in the middle of the architecture and global average pooling.

Features of GoogleNet

The GoogLeNet architecture is very different from previous state-of-the-art architectures such as AlexNet and ZF-Net. It uses many different kinds of methods such as 1 × 1 convolution and global average pooling that enables it to create deeper architecture. In the architecture, we will discuss some of these methods :

(1) 1 × 1 convolution

- The inception architecture uses 1 × 1 convolution in its architecture.
- These convolutions used to decrease the number of parameters (weights and biases) of the architecture. By reducing the parameters, we also increase the depth of the architecture.

(2) Global Average Pooling

- In the previous architectures such as AlexNet, the fully connected layers are used at the end of the network.
- These fully connected layers contain the majority of parameters of many architectures that causes an increase in computation cost.
- In GoogLeNet architecture, there is a method called global average pooling is used at the end of the network.
- This layer takes a feature map of 7 × 7 and averages it to 1 × 1. This also decreases the number of trainable parameters to 0 and improves the top-1 accuracy by 0.6%

(3) Inception Module

- The inception module is different from previous architectures such as AlexNet, ZF-Net. In this architecture, there is a fixed convolution size for each layer.
- In the Inception module 1 × 1, 3 × 3, 5 × 5 convolution and 3 × 3 max pooling performed in a parallel way at the input and the output of these are stacked together to generated final output.
- The idea behind that convolution filters of different sizes will handle objects at multiple scale better.

(4) Auxiliary Classifier for Training

- Inception architecture used some intermediate classifier branches in the middle of the architecture, these branches are used during training only.
- These branches consist of a 5 × 5 average pooling layer with a stride of 3, a 1 × 1 convolutions with 128 filters, two fully connected layers of 1024 outputs and 1000 outputs and a softmax classification layer.
- The generated loss of these layers added to total loss with a weight of 0.3.
- These layers help in combating gradient vanishing problem and also provide regularization.

Architecture

The overall architecture is 22 layers deep. The architecture was designed to keep computational efficiency in mind. The idea behind that the architecture can be run on individual devices even with low computational resources. The architecture also contains two auxiliary classifier layer which is connected to the output of Inception (4(a) and Inception (4(d) layers.

The architectural details of auxiliary classifiers as follows :

(1) An average pooling layer of filter size 5×5 and stride 3.

(2) A 1 × 1 convolution with 128 filters for dimension reduction and ReLU activation.

(3) A fully connected layer with 1025 outputs and ReLU activation

(4) Dropout Regularization with dropout ratio = 0.7

(5) A softmax classifier with 1000 classes output similar to the main softmax classifier.

This architecture takes image of size 224 x 224 with RGB color channels. All the convolutions inside this architecture uses Rectified Linear Units (ReLU) as their activation functions.

type	patch size/stride	output size	depth	#1 × 1	#3 × 3 reduce	#3 × 3	#5 × 5 reduce	#5 × 5	Pool proj	params	ops
convolution	7 × 7/2	112 × 112 × 64	1	–	–	–	–	–	–	2.7K	34M
max pool	3 × 3/2	56 × 56 × 64	0	–	–		–	–	–	–	–
convolution	3 × 3/1	56 × 56 × 192	2	–	64	192	–	–	–	112K	360M
max pool	3 × 3/2	28 × 28 × 192	0	–	–	–	–	–	–	–	–
inception (3a)	–	28 × 28 × 256	2	64	96	128	16	32	32	159K	128M
inception (3b)	–	28 × 28 × 480	2	128	128	192	32	96	64	380K	304M
max pool	3 × 3/2	14 × 14 × 480	0		–	–	–	–	–	–	–
inception (4a)	–	14 × 14 × 512	2	192	96	2184	16	48	64	364K	73M
inception (4b)	–	14 × 14 × 512	2	160	112	224	24	64	64	437K	88M
inception (4c)	–	14 × 14 × 512	2	128	128	256	24	64	64	463K	100M
inception (4d)	–	14 × 14 × 528	2	112	144	288	32	64	64	580K	119M
inception (4e)	–	14 × 14 × 832	2	256	160	320	32	128	128	840K	170M
max pool	3 × 3/2	7 × 7 × 832	0	–	–	–	–	–	–	–	–
inception (5a)	–	7 × 7 × 832	2	256	160	320	32	128	128	1072K	54M
inception (5b)	–	7 × 7 × 1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7 × 7/1	1 × 1 × 1024	0	–	–	–	–	–	–	–	–
dropout (40%)	–	1 × 1 × 1024	0	–	–	–	–	–	–	–	–
linear	–	1 × 1 × 1000	1	–	–	–	–	–	–	1000K	1M
softmax	–	1 × 1 × 1000	0	–	–	–	–	–	–	–	–

Fig. 4.6.16 : Architectural details of GoogLeNet

5. ZFNet

- **ZFNet** is a modified version of AlexNet which gives a better accuracy.In ImageNet Large Scale Visual Recognition Challenge (ILSVR(C) 2013, ZFNet came to the limelight having significant improvement over AlexNet.
- One major difference in the approaches was that ZF Net used 7x7 sized filters whereas AlexNet used 11x11 filters. The intuition behind this model is that by using bigger filters we were losing a lot of pixel information, which we can retain by having smaller filter sizes in the earlier conv layers.
- The number of filters increase as we go deeper. This network also used ReLUs for their activation and trained using batch stochastic gradient descent.

Architecture

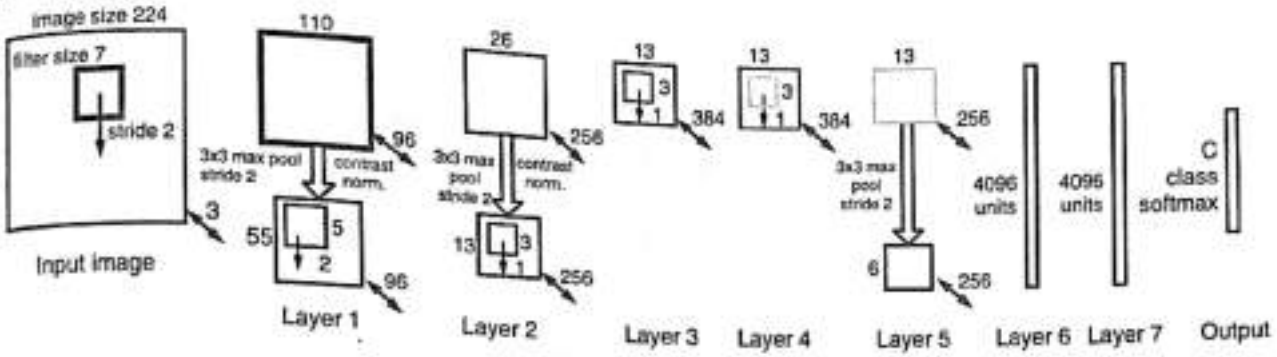


Fig. 4.6.17: Architecture of ZFNet

- Our input is 224x224x3 images.
- Next, 96 convolutions of 7x7 with a stride of 2 are performed, followed by ReLU activation, 3x3 max pooling with stride 2 and local contrast normalization.
- Followed by it are 256 filters of 3x3 each which are then again local contrast normalized and pooled.
- The third and fourth layers are identical with 384 kernels of 3x3 each.
- The fifth layer has 256 filters of 3x3, followed by 3x3 max pooling with stride 2 and local contrast normalization.
- The sixth and seventhlayers house 4096 dense units each.
- Finally, we feed into a Dense layer of 1000 neurons i.e., the number of classes in ImageNet.

6. Resnet

- **Residual Network (ResNet)** is one of the famous deep learning models that was introduced by Shaoqing Ren, Kaiming He, Jian Sun, and Xiangyu Zhang in their paper in 2015. The ResNet model is one of the popular and most successful deep learning models so far.

Residual Blocks

- The problem of training very deep networks has been relieved with the introduction of these Residual blocks and the ResNet model is made up of these blocks.
- The problem of training very deep networks has been relieved with the introduction of these Residual blocks and the ResNet model is made up of these blocks.

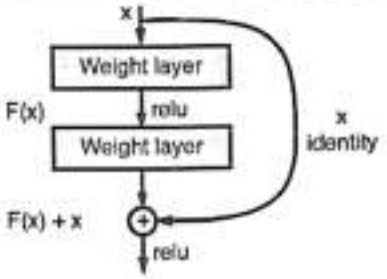


Fig. 4.6.18 : Residual block

- In the Fig. 4.6.18, the very first thing we can notice is that there is a direct connection that skips some layers of the model.
- This connection is called **'skip connection'** and is the heart of residual blocks.
- The output is not the same due to this skip connection. Without the skip connection, input 'X gets multiplied by the weights of the layer followed by adding a bias term.
- Then comes the activation function, f() and we get the output as H(x).

$$H(x) = f(wx + b) \text{ or } H(x) = f(x)$$

- Now with the introduction of a new skip connection technique, the output is H(x) is changed to

$$H(x) = f(x) + x$$

- But the dimension of the input may be varying from that of the output which might happen with a convolutional layer or pooling layers. Hence, this problem can be handled with these two approaches:
  - Zero is padded with the skip connection to increase its dimensions.
  - 1×1 convolutional layers are added to the input to match the dimensions. In such a case, the output is:

$$H(x) = f(x) + w1.x$$

- Here an additional parameter w1 is added whereas no additional parameter is added when using the first approach.
- These skip connections technique in ResNet solves the problem of vanishing gradient in deep CNNs by allowing alternate shortcut path for the gradient to flow through.
- Also, the skip connection helps if any layer hurts the performance of architecture, then it will be skipped by regularization.

Architecture

- There is a 34-layer plain network in the architecture that is inspired by VGG-19 in which the shortcut connection or the skip connections are added.
- These skip connections or the residual blocks then convert the architecture into the residual network.

4.6.3 Applications of CNN

1. **Object detection** : With CNN, we now have sophisticated models like R-CNN, Fast R-CNN, and Faster R-CNN that are the predominant pipeline for many object detection models deployed in autonomous vehicles, facial detection, and more.
2. **Semantic segmentation** : In 2015, a group of researchers from Hong Kong developed a CNN-based Deep Parsing Network to incorporate rich information into an image segmentation model. Researchers from UC Berkeley also built fully convolutional networks that improved upon state-of-the-art semantic segmentation.
3. **Image captioning** : CNNs are used with recurrent neural networks to write captions for images and videos. This can be used for many applications such as activity recognition or describing videos and images for the visually impaired. It has been heavily deployed by YouTube to make sense to the huge number of videos uploaded to the platform on a regular basis.
4. **Face Recognition** : CNNs are used to identifying every face in the picture, focusing on each face despite external factors, such as light, angle, pose, etc., identifying unique features and comparing all the collected data with already existing data in the database to match a face with a name.
5. **Analysing Documents** : Convolutional neural networks can also be used for document analysis. This is not just useful for handwriting analysis, but also has a major stake in recognizers.
6. **Historic and Environmental Collections** : CNNs are also used for more complex purposes such as natural history collections. These collections act as key players in documenting major parts of history such as biodiversity, evolution, habitat loss, biological invasion, and climate change.
7. **Understanding Climate** : CNNs can be used to play a major role in the fight against climate change, especially in understanding the reasons why we see such drastic changes and how we could experiment in curbing the effect.
8. **Advertising** : CNNs have already brought in a world of difference to advertising with the introduction of programmatic buying and data-driven personalized advertising.

4.7 RECURRENT NEURAL NETWORK

Recurrent Neural Network (RNN) is a type of Neural Network where the outputs from previous step are fed as input to the current step in order to predict the output of the layer.

Need of RNN

- In traditional neural networks, all the inputs and outputs are independent of each other.
- But the situations like predicting the next word of a sentence, the previous words are required and hence there is a need to memorize the previous words.
- RNN solved this issue with the help of a Hidden Layer. The main and most important feature of RNN is Hidden state, which remembers some information about a sequence.
- Feed-forward neural networks cannot handle sequential data, considers only the current input and cannot memorize previous inputs.
- The solution to these issues is the RNN. An RNN can handle sequential data, accepting the current input data, and previously received inputs. RNNs can memorize previous inputs due to their internal memory.

Simple RNN

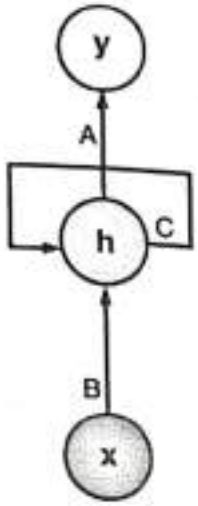


Fig. 4.7.1 : Simple RNN

- As indicated in fig. 4.7.1, the nodes in different layers of the neural network are compressed to form a single layer of recurrent neural networks.
- Here, "x" is the input layer, "h" is the hidden layer, and "y" is the output layer. A, B, and C are the network parameters used to improve the output of the model.
- RNN uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output.
- This reduces the complexity of parameters, unlike other neural networks.
- At any given time t, the current input is a combination of input at x(t) and x(t-1).
- The output at any given time is fetched back to the network to improve on the output.

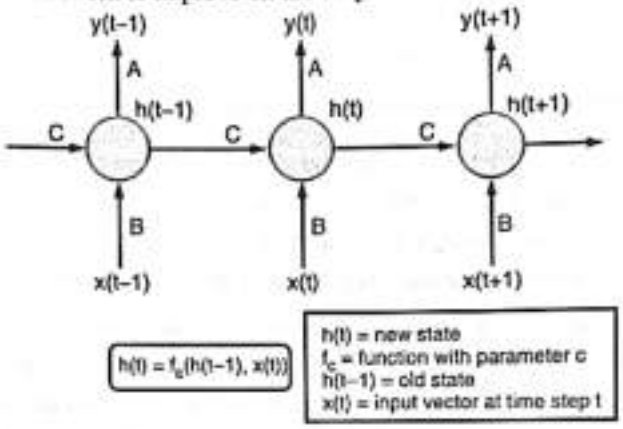


Fig. 4.7.2 : Fully connected RNN

Working of RNN

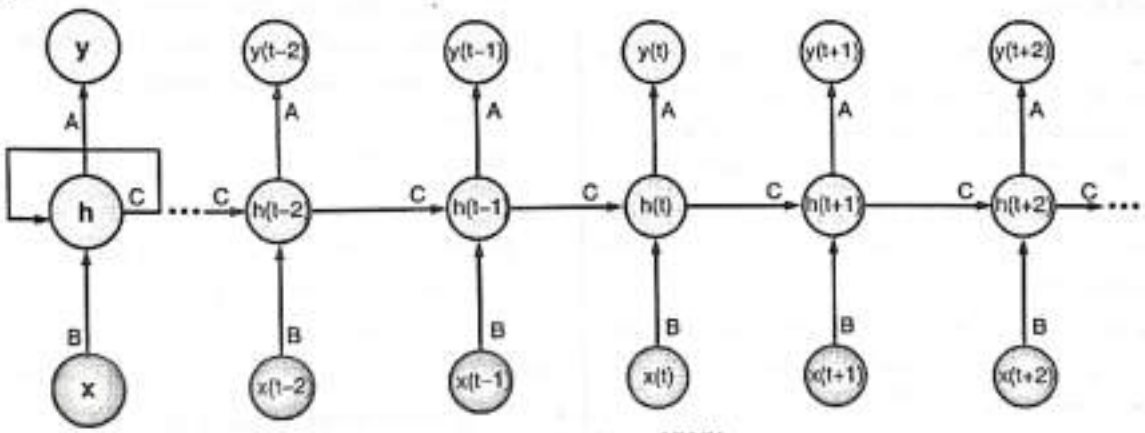


Fig. 4.7.3 : Working of RNN

- The input layer 'x' takes in the input to the neural network and processes it and passes it onto the middle layer.
- The middle layer 'h' can consist of multiple hidden layers, each with its own activation functions and weights and biases.
- The Recurrent Neural Network will standardize the different activation functions and weights and biases so that each hidden layer has the same parameters.
- Then, instead of creating multiple hidden layers, it will create one and loop over it as many times as required.
- Based on number of inputs and outputs RNN can be classified into One to One, One to Many, Many to One and Many to Many RNN.
- **One to One RNN** is also known as Vanilla Neural Network. It is used for general machine learning problems, which has a single input and a single output.

- **One to Many RNN** has a single input and multiple outputs. An example of this is the image caption.
- **Many to One RNN** takes a sequence of inputs and generates a single output. Sentiment analysis is a good example of this kind of network where a given sentence can be classified as expressing positive, negative or neutral sentiments.
- **Many to Many RNN** takes a sequence of inputs and generates a sequence of outputs. Machine translation is one of the examples.

RNN Vs Feed-Forward Neural networks

- A feed-forward neural network allows information to flow only in the forward direction, from the input nodes, through the hidden layers, and to the output nodes. There are no cycles or loops in the network.
- In a feed-forward neural network, the decisions are based on the current input. It doesn't memorize the past data, and there's no future scope. Feed-forward neural networks are used in general regression and classification problems.

☞ Applications

- Natural Language Processing-Text mining, Language Modelling, Text Summarization, Sequence Prediction and Sentiment analysis can be carried out using an RNN for Natural Language Processing (NLP).
- Machine Translation-Given an input in one language, RNNs can be used to translate the input into different languages as output.
- Time Series Prediction-Any time series problem, like predicting the prices of stocks in a particular month, can be solved using an RNN.
- Image Captioning, Generating Image Descriptions and Video Tagging -RNN are used to caption an image by analyzing the activities present.
- Speech Recognition-RNN are used for predicting phonetic segments considering sound waves from a medium as an input sourceto recognize speech.
- Call Center Analysis - This can be considered as one of the major applications of RNNs in the field of audio processing.
- Face detection, OCR Applications as Image Recognition.
- Other applications like Music composition, Handwriting Recognition, Stock market prediction.

☞ Advantages

1. Ans RNN remembers each and every information through time, because of the feature to remember previous inputs, it is useful in time series prediction. This is called Long Short Term Memory.
2. Recurrent neural network are even used with convolutional layers to extend the effective pixel neighborhood.

☞ Disadvantages

1. Gradient vanishing problem - RNNs suffer from the problem of vanishing gradients. The gradients carry information used in the RNN, and when the gradient becomes too small, the parameter updates become insignificant. This makes the learning of long data sequences difficult.
2. Gradient exploding problem - While training a neural network, if the slope tends to grow exponentially instead of decaying, this is called an Exploding Gradient. This problem arises when large error gradients accumulate, resulting in very large updates to the neural network model weights during the training process.
3. Training an RNN is a very difficult task.
4. Due to Gradient problems, long training time, poor performance, and bad accuracy are the major issues.
5. It cannot process very long sequences if using tanh or relu as an activation function.

Long Short-Term Memory (LSTM) Networks

- **Long Short-Term Memory (LSTM)** networks are a type of recurrent neural network capable of learning order dependence for long periods in sequence prediction problems. This is a behavior required in complex problem domains like machine translation, speech recognition, and more.
- As we know now that RNN consists of multiple layers similar to a Feed-Forward Neural Network: the input layer, hidden layer(s), and output layer.

- However, RNN contains recurrent units in its hidden layer, which allows the algorithm to process sequence data.
- It does it by recurrently passing a hidden state from a previous timestep and combining it with an input of the current one.

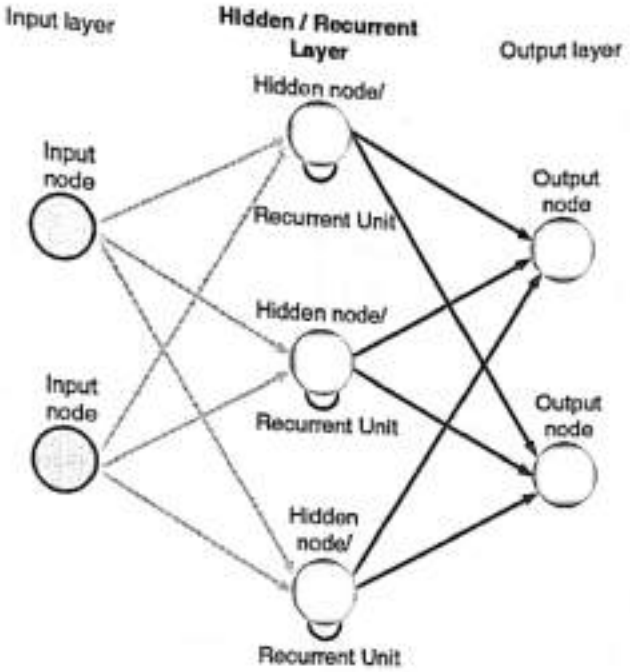


Fig. 4.7.4 : A standard RNN architecture

- Timestep is nothing but single processing of the inputs through the recurrent unit (chain of repeating modules).
- The number of timesteps is equal to the length of the sequence. So, RNNs utilize recurrent units to learn from the sequence data.

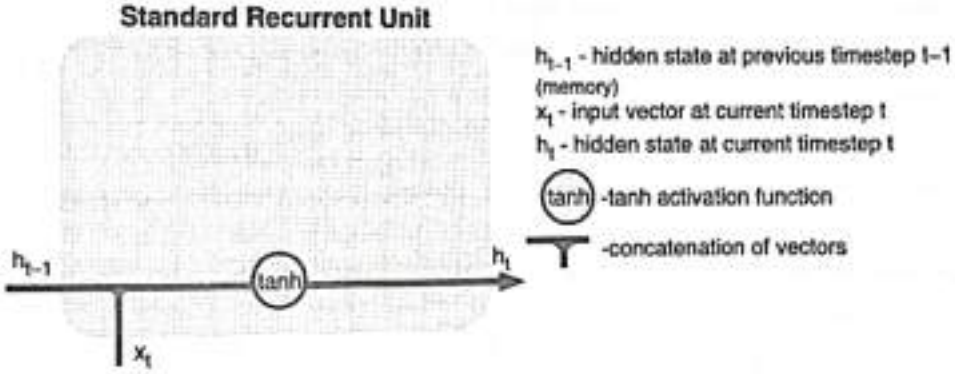


Fig. 4.7.5 : A Standard Recurrent Unit

- If we look inside the simplified recurrent unit diagram of a standard RNN (weights and biases not shown here), we can see that there are only two major operations: combining the previous hidden state with the new input and passing it through the activation function. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.
- After the hidden state is calculated at timestep t, it is passed back to the recurrent unit and combined with the input at timestep t+1 to calculate the new hidden state at timestep t+1.
- This process repeats for t+2, t+3, ..., t+n until the predefined number (n) of timesteps is reached.
- Meanwhile, LSTM employs various gates to decide what information to keep or discard. Also, it adds a cell state, which is like a long-term memory of LSTM.

Working of LSTM

LSTM recurrent unit is much more complex than that of RNN, which improves learning but requires more computational resources.

LSTM recurrent unit processes information in the following way :

(1) **Hidden state and new inputs :** hidden state from a previous timestep (h_{t-1}) and the input at a current timestep (x_t) are combined before passing copies of it through various gates.

(2) **Forget gate :** this gate controls what information should be forgotten. Since the sigmoid function ranges between 0 and 1, it sets which values in the cell state should be discarded (multiplied by 0), remembered (multiplied by 1), or partially remembered (multiplied by some value between 0 and 1).

(3) **Input gate :** helps to identify important elements that need to be added to the cell state. Note that the results of the input gate get multiplied by the cell state candidate, with only the information deemed important by the input gate being added to the cell state.

(4) **Update cell state :** first, the previous cell state (c_{t-1}) gets multiplied by the results of the forget gate. Then we add new information from (input gate × cell state candidate) to get the latest cell state (c_t).

(5) **Update hidden state :** the last part is to update the hidden state. The latest cell state (c_t) is passed through the tanh activation function and multiplied by the results of the output gate.

(6) Finally, the latest cell state (c_t) and the hidden state (h_t) go back into the recurrent unit, and the process repeats at timestep t+1. The loop continues until we reach the end of the sequence.

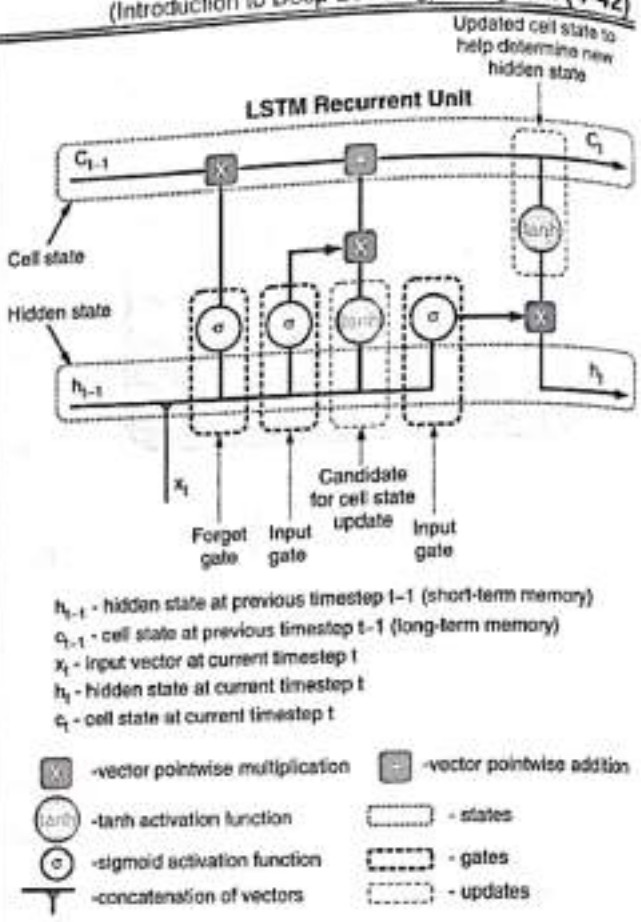


Fig. 4.7.6 : A Recurrent Unit of LSTM

Applications

- Long Short-Term Memory (LSTM) can solve numerous tasks not solvable by previous learning algorithms for recurrent neural networks (RNNs).
- This includes handwriting recognition and generation, language modeling and translation, acoustic modeling of speech, speech synthesis, protein secondary structure prediction, analysis of audio, and video data among others.

Deep RNN

- An RNN is deep with respect to time. But what if it's deep with respect to space as well, as in a feed-forward network?
- This is the fundamental notion that has inspired researchers to explore Deep Recurrent Neural Networks, or Deep RNNs.
- In a typical deep RNN, the looping operation is expanded to multiple hidden units.

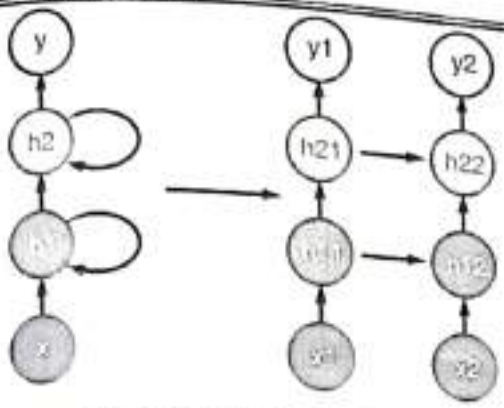


Fig. 4.7.7 : Two-layer deep RNN

- We can also introduce depth to a hidden unit to make RNN deep.

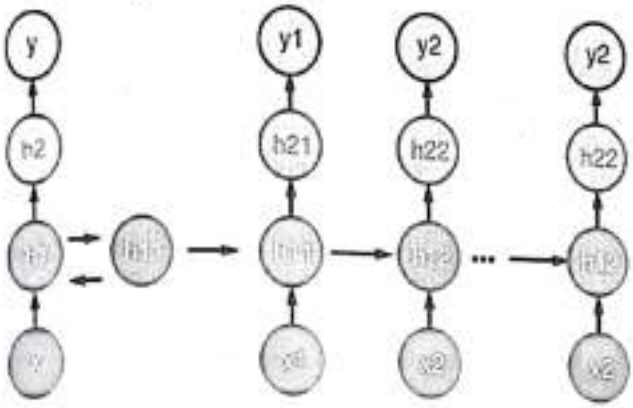


Fig. 4.7.8 : Multi-layer deep RNN

- This model increases the distance traversed by a variable from time t to time t+1.
- It can have simple RNNs, GRUs, or LSTMs as its hidden units.
- It helps in modeling varying data representations by capturing the data from all ends, and enables passing multiple/single hidden state(s) to the multiple hidden states of the subsequent layers.
- In deep RNNs, the hidden state information is passed to the next time step of the current layer and the current time step of the next layer.
- There exist many different flavors of deep RNNs, such as LSTMs, GRUs, or vanilla RNNs. Conveniently these models are all available as parts of the high-level APIs of deep learning frameworks.

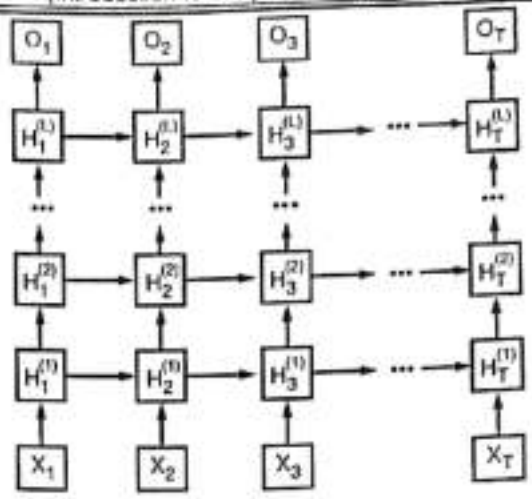


Fig. 4.7.9 : Architecture of deep RNN

4.8 AUTOENCODER

Introduction

- **Autoencoders** are a specific type of feedforward neural networks where the input is the same as the output.
- The aim of an autoencoder is to learn a lower-dimensional representation (encoding) for a higher-dimensional data, typically for dimensionality reduction, by training the network to capture the most important parts of the input image.
- The code(lower-dimensional representation) is a compact "summary" or "compression" of the input, also called the **latent-space representation.**
- Today data denoising and dimensionality reduction for data visualization are considered as two main interesting practical applications of autoencoders.
- With appropriate dimensionality and sparsity constraints, autoencoders can learn data projections that are more interesting than PCA or other basic techniques.

Architecture

- An autoencoder consists of 3 components: encoder, code and decoder.
- The encoder compresses the input and produces the code, the decoder then reconstructs the input only using this code.

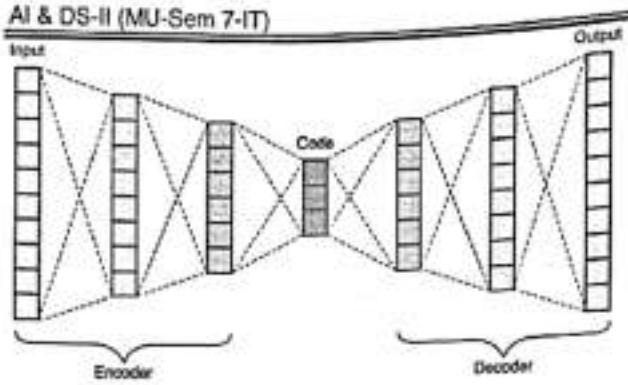


Fig. 4.8.1 : Architecture of Autoencoder

1. **Encoder :** A module that compresses the train-validate-test set input data into an encoded representation that is typically several orders of magnitude smaller than the input data.
2. **Bottleneck/Code :** A module that contains the compressed knowledge representations and is therefore the most important part of the network.
3. **Decoder :** A module that helps the network "decompress" the knowledge representations and reconstructs the data back from its encoded form. The output is then compared with a ground truth.

Both the encoder and decoder are fully-connected feedforward neural networks, essentially the ANNs.

Code is a single layer of an ANN with the dimensionality of our choice. The number of nodes in the code layer (code size) is a hyperparameter that we set before training the autoencoder.

As seen in the Fig. 4.8.1, first the input passes through the encoder, which is a fully-connected ANN, to produce the code. The decoder, which has the similar ANN structure, then produces the output only using the code. The goal is to get an output identical with the input. Note that the decoder architecture is the mirror image of the encoder. This is not a requirement but it's typically the case. The only requirement is the dimensionality of the input and output needs to be the same.

Training Autoencoder

There are 4 hyperparameters that we need to set before training an autoencoder :

(1) **Code size :** The code size or the size of the bottleneck is the most important hyperparameter used to tune the autoencoder. It is the number of nodes in the middle layer. The bottleneck size decides how much the data has to be compressed. This can also act as a regularization term. Smaller size results in more compression.

(2) **Number of layers :** Like all neural networks, an important hyperparameter to tune autoencoders is the depth of the encoder and the decoder. The autoencoder can be as deep as we like. While a higher depth increases model complexity, a lower depth is faster to process.

(3) **Number of nodes per layer :** The number of nodes per layer defines the weights we use per layer. Typically, the number of nodes decreases with each subsequent layer in the autoencoder as the input to each of these layers becomes smaller across the layers.

(4) **Loss function :** The loss function we use to train the autoencoder is highly dependent on the type of input and output we want the autoencoder to adapt to. If we are working with image data, the most popular loss functions for reconstruction are MSE Loss and L1 Loss. In case the inputs and outputs are within the range [0,1], as in MNIST, we can also make use of Binary Cross Entropy as the reconstruction loss.

Types of Autoencoders

There are basically 7 types of autoencoders:

1. Denoising autoencoder
2. Sparse Autoencoder
3. Deep Autoencoder
4. Contractive Autoencoder
5. Undercomplete Autoencoder
6. Convolutional Autoencoder
7. Variational Autoencoder

1. Denoising Autoencoder

- Denoising autoencoders create a corrupted copy of the input by introducing some noise.
- This helps to avoid the autoencoders to copy the input to the output without learning features about the data.
- These autoencoders take a partially corrupted input while training to recover the original undistorted input.

- The model learns a vector field for mapping the input data towards a lower dimensional manifold which describes the natural data to cancel out the added noise.

Advantages

1. It was introduced to achieve good representation. Such a representation is one that can be obtained robustly from a corrupted input and that will be useful for recovering the corresponding clean input.
2. Corruption of the input can be done randomly by making some of the input as zero. Remaining nodes copy the input to the noised input.
3. Minimizes the loss function between the output node and the corrupted input.
4. Setting up a single-thread denoising autoencoder is easy.

Drawbacks

1. To train an autoencoder to denoise data, it is necessary to perform preliminary stochastic mapping in order to corrupt the data and use as input.
2. This model isn't able to develop a mapping which memorizes the training data because our input and target output are no longer the same.

2. Sparse Autoencoder

- Sparse autoencoders have hidden nodes greater than input nodes. They can still discover important features from the data.
- A generic sparse autoencoder is visualized where the obscurity of a node corresponds with the level of activation.
- Sparsity constraint is introduced on the hidden layer. This is to prevent output layer copy input data.
- Sparsity may be obtained by additional terms in the loss function during the training process, either by comparing the probability distribution of the hidden unit activations with some low desired valueor by manually zeroing all but the strongest hidden unit activations.
- Some of the most powerful AIs in the 2010s involved sparse autoencoders stacked inside of deep neural networks.

Advantages

1. Sparse autoencoders have a sparsity penalty, a value close to zero but not exactly zero. Sparsity penalty is applied on the hidden layer in addition to the reconstruction error. This prevents overfitting.
2. They take the highest activation values in the hidden layer and zero out the rest of the hidden nodes. This prevents autoencoders to use all of the hidden nodes at a time and forcing only a reduced number of hidden nodes to be used.

Drawbacks

1. For it to be working, it's essential that the individual nodes of a trained model which activate are data dependent, and that different inputs will result in activations of different nodes through the network.

3. Deep Autoencoder

- Deep Autoencoders consist of two identical deep belief networks, one network for encoding and another for decoding.
- Typically, deep autoencoders have 4 to 5 layers for encoding and the next 4 to 5 layers for decoding. We use unsupervised layer by layer pre-training for this model.
- The layers are **Restricted Boltzmann Machines** which are the building blocks of deep-belief networks. Processing the benchmark dataset MNIST, a deep autoencoder would use binary transformations after each RBM.
- Deep autoencoders are useful in topic modeling, or statistically modeling abstract topics that are distributed across a collection of documents. They are also capable of compressing images into 30 number vectors.

Advantages

1. Deep autoencoders can be used for other types of datasets with real-valued data, on which you would use Gaussian rectified transformations for the RBMs instead.
2. Final encoding layer is compact and fast.

Drawbacks

1. Chances of overfitting to occur since there are more parameters than input data.
2. Training the data maybe a nuance since at the stage of the decoder's backpropagation, the learning rate should be lowered or made slower depending on whether binary or continuous data is being handled.

4. Contractive Autoencoder

- The objective of a contractive autoencoder is to have a robust learned representation which is less sensitive to small variation in the data.
- Robustness of the representation for the data is done by applying a penalty term to the loss function.
- Contractive autoencoder is another regularization technique just like sparse and denoising autoencoders.
- However, this regularizer corresponds to the Frobenius norm of the Jacobian matrix of the encoder activations with respect to the input.
- Frobenius norm of the Jacobian matrix for the hidden layer is calculated with respect to input and it is basically the sum of square of all elements.

Advantages

1. Contractive autoencoder is a better choice than denoising autoencoder to learn useful feature extraction.
2. This model learns an encoding in which similar inputs have similar encodings. Hence, we're forcing the model to learn how to contract a neighborhood of inputs into a smaller neighborhood of outputs.

5. Undercomplete Autoencoder

- The objective of undercomplete autoencoder is to capture the most important features present in the data.
- Undercomplete autoencoders have a smaller dimension for hidden layer compared to the input layer. This helps to obtain important features from the data.
- It minimizes the loss function by penalizing the g(f(x)) for being different from the input x.

Advantages

1. Undercomplete autoencoders do not need any regularization as they maximize the probability of data rather than copying the input to the output.

Drawbacks

1. Using an overparameterized model due to lack of sufficient training data can create overfitting.

6. Convolutional Autoencoder

- Autoencoders in their traditional formulation does not take into account the fact that a signal can be seen as a sum of other signals.
- Convolutional Autoencoders use the convolution operator to exploit this observation. They learn to encode the input in a set of simple signals and then try to reconstruct the input from them, modify the geometry or the reflectance of the image. They are the state-of-art tools for unsupervised learning of convolutional filters. Once these filters have been learned, they can be applied to any input in order to extract features.
- These features, then, can be used to do any task that requires a compact representation of the input, like classification.

Advantages

1. Due to their convolutional nature, they scale well to realistic-sized high dimensional images.
2. Can remove noise from picture or reconstruct missing parts.

Drawbacks

1. The reconstruction of the input image is often blurry and of lower quality due to compression during which information is lost.

7. Variational Autoencoder

- Variational autoencoder models make strong assumptions concerning the distribution of latent variables.
- They use a variational approach for latent representation learning, which results in an additional loss component and a specific estimator for the training algorithm called the Stochastic

- Gradient Variational Bayes estimator. It assumes that the data is generated by a directed graphical model and that the encoder is learning an approximation to the posterior distribution where Φ and θ denote the parameters of the encoder (recognition model) and decoder (generative model) respectively.
- The probability distribution of the latent vector of a variational autoencoder typically matches that of the training data much closer than a standard autoencoder.

☞ Advantages

1. It gives significant control over how we want to model our latent distribution unlike the other models.
2. After training you can just sample from the distribution followed by decoding and generating new data.

☞ Drawbacks

1. When training the model, there is a need to calculate the relationship of each parameter in the network with respect to the final output loss using a technique known as backpropagation. Hence, the sampling process requires some extra attention.

☞ Applications of autoencoders

1. **Dimensionality reduction:** Undercomplete autoencoders are those that are used for dimensionality reduction. These can be used as a pre-processing step for dimensionality reduction as they can perform fast and accurate dimensionality reductions without losing much information.

   Furthermore, while dimensionality reduction procedures like PCA can only perform linear dimensionality reductions, undercomplete autoencoders can perform large-scale non-linear dimensionality reductions.

2. Image denoising: Autoencoders like the denoising autoencoder can be used for performing efficient and highly accurate image denoising.

   Unlike traditional methods of denoising, autoencoders do not search for noise, they extract the image from the noisy data that has been fed to them via learning a representation of it. The representation is then decompressed to form a noise-free image.

   Denoising autoencoders thus can denoise complex images that cannot be denoised via traditional methods.

3. Generation of image and time series data: Variational Autoencoders can be used to generate both image and time series data.

   The parameterized distribution at the bottleneck of the autoencoder can be randomly sampled to generate discrete values for latent attributes, which can then be forwarded to the decoder, leading to generation of image data. Variational autoencoders can also be used to model time series data like music.

4. Anomaly Detection: Undercomplete autoencoders can also be used for anomaly detection.

⏭ 4.9 RESTRICTED BOLTZMANN MACHINE (RBM)

- **Boltzmann Machine** is a network of neurons in which all the neurons are connected to each other. In this machine, there are two layers named visible layer or input layer and hidden layer.
- The visible layer is denoted as v and the hidden layer is denoted as the h.
- In Boltzmann machine, there is no output layer. Boltzmann machines are random and generative neural networks capable of learning internal representations and are able to represent and (given enough time) solve tough combinatoric problems.
- A **restricted** term refers to that we are not allowed to connect the same type layer to each other.
- In other words, the two neurons of the input layer or hidden layer can't connect to each other.
- Although the hidden layer and visible layer can be connected to each other.
- As in this machine, there is no output layer so the question arises how we are going to identify, adjust the weights and how to measure the that our prediction is accurate or not.
- All the questions have one answer, that is **Restricted Boltzmann Machine.**
- The RBM algorithm was proposed by Geoffrey Hinton (2007), which learns probability distribution over its sample training data inputs.

- It has seen wide applications in different areas of supervised/unsupervised machine learning such as feature learning, dimensionality reduction, classification, collaborative filtering, and topic modeling.
- If we consider the example movie rating, movies like Avengers, Avatar, and Interstellar have strong associations with the latest fantasy and science fiction factor.
- Based on the user rating RBM will discover latent factors that can explain the activation of movie choices.
- In short, RBM describes variability among correlated variables of input dataset in terms of a potentially lower number of unobserved variables.
- The energy function is given by

$$E(V, h) = -a^T v - b^T h - v^T W h$$

☞ Working of RBM

In RBM there are two phases through which the entire RBM works:

▶ Phase 1 :

- In this phase, we take the input layer and using the concept of weights and biased we are going to activate the hidden layer.
- This process is said to be Feed Forward Pass. In Feed Forward Pass we are identifying the positive association and negative association.

Feed Forward Equation

(1) **Positive Association :** When the association between the visible unit and the hidden unit is positive.

(2) **Negative Association :** When the association between the visible unit and the hidden unit is negative.

▶ Phase 2 :

- As we don't have any output layer. Instead of calculating the output layer, we are reconstructing the input layer through the activated hidden state.
- This process is said to be Feed Backward Pass. We are just backtracking the input layer through the activated hidden neurons.
- After performing this we have reconstructed Input through the activated hidden state. So, we can calculate the error and adjust weight in this way:

Feed Backward Equation

Error = Reconstructed Input Layer-Actual Input layer

Adjust Weight = Input*error*learning rate (0.1)

After doing all the steps we get the pattern that is responsible to activate the hidden neurons.

☞ Advantages

1. Expressive enough to encode any distribution and computationally efficient.
2. Faster than traditional Boltzmann Machine due to the restrictions in terms of connections between nodes.
3. Activations of the hidden layer can be used as input to other models as useful features to improve performance

☞ Disadvantages

1. Training is more difficult as it is difficult to calculate the Energy gradient function.
2. CD-k algorithm used in RBMs is not as familiar as the back propagation algorithm.
3. Weight Adjustment

☞ Applications of RBM

(1) **Hand Written Digit Recognition** : is a very common problem these days and is used in a variety of applications like criminal evidence, office computerization, check verification, and data entry applications.

(2) **Real time intra pulse recognition of radar** : Intra-pulse features extraction of radar is of high significance and it suffers from challenges like data representation capability and the robustness to cope with noise. In order to achieve better recognition performance, ambiguity function (AF) of radar signals is taken as intra-pulse characteristics. So, here the restricted Boltzmann machine (RBM) is adopted, a stochastic neural network, to extract features effectively.

4.10 APPLICATIONS OF DEEP LEARNING

- Deep learning applications are used in industries from automated driving to medical devices.
- Automotive researchers are using deep learning to automatically detect objects such as 'stop signs and 'traffic lights'.
- In addition, deep learning is used to detect pedestrians, which help to decrease accidents. Deep learning is driving advances in AI that are changing our world.
- Some applications of Deep Learning are listed below:
  (1) Industries
  (2) Self Driving Cars
  (3) News Aggregation and Fraud News Detection
  (4) Natural Language Processing
  (5) Virtual Assistants
  (6) Entertainment
  (7) Visual Recognition
  (8) Fraud Detection
  (9) Healthcare
  (10) Personalisations
  (11) Detecting Developmental Delay in Children
  (12) Colourisation of Black and White images
  (13) Adding sounds to silent movies
  (14) Automatic Machine Translation
  (15) Automatic Handwriting Generation
  (16) Automatic Game Playing
  (17) Language Translations
  (18) Pixel Restoration
  (19) Photo Descriptions
  (20) Demographic and Election Predictions
  (21) Deep Dreaming

4.11 MULTIPLE CHOICE QUESTIONS

Q. 4.1 Why do we need biological neural networks?
(a) to solve tasks like machine vision & natural language processing
(b) to apply heuristic search methods to find solutions of problem
(c) to make smart human interactive & user-friendly system
(d) all of the mentioned
✓**Ans. : (d)**

Q. 4.2 What is plasticity in neural networks?
(a) input pattern keeps on changing
(b) input pattern has become static
(c) output pattern keeps on changing
(d) output is static
✓**Ans. : (a)**

Q. 4.3 What are the issues on which biological networks proves to be superior than AI networks?
(a) robustness & fault tolerance
(b) flexibility
(c) collective computation
(d) all of the mentioned
✓**Ans. : (d)**

Q. 4.4 The fundamental unit of network is
(a) brain (b) nucleus
(c) neuron (d) axon
✓**Ans. : (c)**

Q. 4.5 What are dendrites?
(a) fibers of nerves
(b) nuclear projections
(c) other name for nucleus
(d) none of the mentioned
✓**Ans. : (a)**

Q. 4.6 When the cell is said to be fired?
(a) if potential of body reaches a steady threshold values
(b) if there is impulse reaction
(c) during upbeat of heart
(d) none of the mentioned
✓**Ans. : (a)**

Q. 4.7 Function of dendrites is?
(a) receptors
(b) transmitter
(c) both receptor & transmitter
(d) none of the mentioned
✓**Ans. : (a)**

Q. 4.8 What is purpose of Axon?
(a) receptors (b) transmitter
(c) transmission (d) none of the mentioned
✓**Ans. : (c)**

Q. 4.9 The cell body of neuron can be analogous to what mathematical operation?
(a) summing (b) differentiator
(c) integrator (d) none of the mentioned
✓**Ans. : (a)**

Q. 4.10 Why can't we design a perfect neural network?
(a) full operation is still not known of biological neurons
(b) number of neurons is itself not precisely known
(c) number of interconnections is very large & is very complex
(d) all of the mentioned
✓**Ans. : (d)**

Q. 4.11 What is the feature of ANNs due to which they can deal with noisy, fuzzy, inconsistent data?
(a) associative nature of networks
(b) distributive nature of networks
(c) both associative & distributive
(d) none of the mentioned
✓**Ans. : (c)**

Q. 4.12 What was the name of the first model which can perform wieghted sum of inputs?
(a) McCulloch-pitts neuron model
(b) Marvin Minsky neuron model
(c) Hopfield model of neuron
(d) none of the mentioned
✓**Ans. : (a)**

Q. 4.13 What is an activation value?
(a) weighted sum of inputs
(b) threshold value
(c) main input to neuron
(d) none of the mentioned
✓**Ans. : (a)**

Q. 4.14 Positive sign of weight indicates?
(a) excitatory input
(b) inhibitory input
(c) can be either excitatory or inhibitory as such
(d) none of the mentioned
✓**Ans. : (a)**

Q. 4.15 Negative sign of weight indicates?
(a) excitatory input (b) inhibitory input
(c) excitatory output (d) inhibitory output
✓**Ans. : (b)**

Q. 4.16 The amount of output of one unit received by another unit depends on what?
(a) output unit (b) input unit
(c) activation value (d) weight
✓**Ans. : (d)**

Q. 4.17 The process of adjusting the weight is known as?
(a) activation (b) synchronisation
(c) learning (d) none of the mentioned
✓**Ans. : (c)**

Q. 4.18 The procedure to incrementally update each of weights in neural is referred to as?
(a) synchronization (b) learning law
(c) learning algorithm
(d) both learning algorithm & law
✓**Ans. : (d)**

Q. 4.19 In what ways can output be determined from activation value?
(a) deterministically (b) stochastically
(c) both deterministically & stochastically
(d) none of the mentioned
✓**Ans. : (c)**

Q. 4.20 How can output be updated in neural network?
(a) synchronously (b) asynchronously
(c) both synchronously & asynchronously
(d) none of the mentioned
✓**Ans. : (c)**

Q. 4.21 When both inputs are 1, what will be the output of the pitts model nand gate ?
(a) 0 (b) 1
(c) either 0 or 1 (d) z
✓**Ans. : (a)**

Q. 4.22 What is the name of the model in figure below?

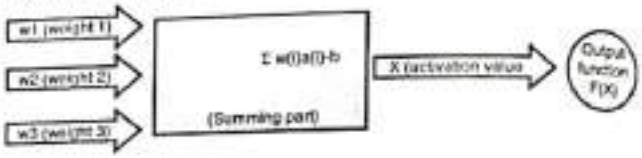


Fig. Q. 4.22

(a) Rosenblatt perceptron model
(b) McCulloch-pitts model
(c) Widrow's Adaline model
(d) None of the mentioned
✓**Ans. : (b)**

Q. 4.23 What is nature of function F(x) in the figure given in Q. 4.22?
(a) linear (b) non-linear
(c) can be either linear or non-linear
(d) none of the mentioned
✓**Ans. : (b)**

Q. 4.24 What does the character 'b' represents in the diagram given in Q. 4.22?
(a) bias (b) any constant value
(c) a variable value (d) none of the mentioned
✓Ans. : (a)

Q. 4.25 If 'b' in the figure below is the bias, then what logic circuit does it represents?

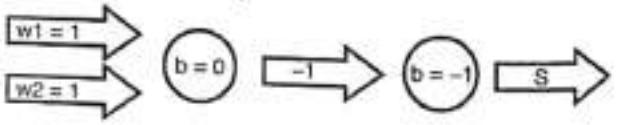


Fig. 4.25

(a) or gate (b) and gate
(c) nor gate (d) nand gate
✓**Ans. : (c)**

Q. 4.26 When both inputs are 1, what will be the output of the figure given in Q.4.25?
(a) 0 (b) 1
(c) either 0 or 1 (d) z
✓**Ans. : (a)**

Q. 4.27 On what parameters can change in weight vector depend?
(a) learning parameters (b) input vector
(c) learning signal (d) all of the mentioned
✓**Ans. : (d)**

Q. 4.28 Feedforward networks are used for?
(a) pattern mapping (b) pattern association
(c) pattern classification (d) all of the mentioned
✓**Ans. : (d)**

Q. 4.29 _____ in which we give input to our model.
(a) Input layer (b) Output layer
(c) Hidden layer (d) None of these ✓**Ans. : (a)**

Q. 4.30 The input from Input layer is then feed into the _____.
(a) Input layer (b) Output layer
(c) Hidden layer (d) None of these ✓**Ans. : (c)**

Q. 4.31 The data is then fed into the model and output from each layer is obtained this step is called _____.
(a) Feedforward (b) Feed backward
(c) input layer (d) Output layer ✓**Ans. : (a)**

Q. 4.32 In _____ holds the raw input of image used to build ConvNets.
(a) Input Layer
(b) Convolution Layer
(c) Activation Function Layer
(d) Pool Layer ✓**Ans. : (a)**

Q. 4.33 _____ computes the output volume by computing dot product between all filters and image patch.
(a) Input Layer (b) Convolution Layer
(c) Activation Function Layer
(d) Pool Layer ✓**Ans. : (b)**

Q. 4.34 _____ will apply element wise activation function to the output of convolution layer.
(a) Input Layer
(b) Convolution Layer
(c) Activation Function Layer
(d) Pool Layer ✓**Ans. : (c)**

Q. 4.35 _____ common types of pooling layers.
(d) 2 (b) 3
(c) 4 (d) 5 ✓**Ans. : (a)**

Q. 4.36 _____is a pooling operation that selects the maximum element from the region of the feature map covered by the filter.
(a) Max Pooling (b) Average Pooling
(c) Global pooling (d) None of these ✓**Ans. : (a)**

Q. 4.37 _____ computes the average of the elements present in the region of feature map covered by the filter.
(a) Max Pooling (b) Average Pooling
(c) Global pooling (d) None of these ✓**Ans. : (b)**

Q. 4.38 _____reduces each channel in the feature map to a single value.
(a) Max Pooling (b) Average Pooling
(c) Global pooling (d) None of these ✓**Ans. : (c)**

Q. 4.39 Which of the following is/are Limitations of deep learning?
(a) Data labeling (b) Obtain huge training datasets
(c) both 1 and 2 (d) None of the above ✓**Ans. : (c)**

Q. 4.40 RNNs stands for?
(a) Receives neural networks
(b) Receives neural networks
(c) Recording neural networks
(d) Recurrent neural networks ✓**Ans. : (d)**

Q. 4.41 In which neural net architecture, does weight sharing occur?
(a) Convolutional neural Network
(b) Recurrent Neural Network
(c) Fully Connected Neural Network
(d) Both 1 and 2 ✓**Ans. : (d)**

Q. 4.42 What consist of Boltzmann machine?
(a) fully connected network with both hidden and visible units
(b) asynchronous operation
(c) stochastic update
(d) all of the mentioned ✓**Ans. : (d)**

Q. 4.43 Which, if any, of the following propositions is true about fully-connected neural networks (FCNN)?
(a) In a FCNN, there are connections between neurons of a same layer.
(b) In a FCNN, the most common weight initialization scheme is the zero initialization, because it leads to faster and more robust training.
(c) A FCNN with only linear activations is a linear network.
(d) None of the above ✓**Ans. : (c)**

Q. 4.44 What is deep in deep learning?
(a) The deep in deep learning is a reference to any kind of deeper understanding achieved by the approach
(b) It stands for the idea of successive layers of representations in deep learning
(c) Answers A & B
(d) None of the above ✓**Ans. : (b)**

Q. 4.45 What are deep neural networks in deep learning?

(a) In deep learning, layered representations are (almost always) learned via modelscalled deep neural networks, structured in literal layers stacked on top of each other

(b) These networks are the brain neurons studying in neurobiology

(c) These are models of human brain

(d) Neural network is a cell in brain ✓**Ans. : (a)**

Q. 4.46 Which statement is true?

(a) Deep learning is a mathematical framework for learning representations from data

(b) Deep learning is a biological framework for learning representations from brain

data

(c) Deep learning is an analogue framework for learning representations from data

(d) Deep learning is a digital framework for learning representations from data ✓**Ans. : (a)**

Q. 4.47 What are layers in deep learning?

(a) Deep neural networks do this input-to-target mapping via a deep sequence of simple data transformations called layers and that these data transformations are learned by exposure to examples

(b) It is a sort of membrane in brain

(c) None of the above

(d) All of the above ✓**Ans. : (a)**

Q. 4.48 What is loss function in deep learning?

(a) To control the output of a neural network, you need to be able to measure how far this output is from what you expected

(b) To calculate loss in banks

(c) These are true targets of data

(d) These are the predicted values only ✓**Ans. : (a)**

Descriptive Questions

Q. 1 What is Deep Learning? Explain working of Deep learning networks.

Q. 2 Differentiate between Machine Learning and Deep Learning.

Q. 3 Explain the operations of Dendrite, soma and axon in the biological neuron.

Q. 4 Differentiate biological neural network and artificial neural network.

Q. 5 Draw and explain McCulloch Pitts neuron architecture.

Q. 6 Explain common activation functions used in neural network.

Q. 7 What are the important properties of activation function used in neural networks?

Q. 8 List the different activation functions used in neural network.

Q. 9 With mathematical equations list four different activation functions used in neurons.

Q. 10 Explain different types of Deep learning networks.

Q. 11 Differentiate between Multi-Layer Feed Forward Neural Network and Signal Layer Feed Forward Neural Network.

Q. 12 Explain convolutional networks in detail.

Q. 13 Explain CNN architecture.

Q. 14 Explain function of Hidden layers in CNN.

Q. 15 Explain Pooling layer in CNN and different types of pooling.

Q. 16 Write short note on Padding.

Q. 17 What are types of Padding?

Q. 18 What is strided convolution?

Q. 19 Discuss rectified linear unit and its advantages.

Q. 20 Explain different types of Variants.

Q. 21 What are Autoencoders? Discuss types of Autoencoders.

Q. 22 Write a short note on: LSTM networks.

Q. 23 What is Deep Learning? Write various applications of Deep Learning.

Q. 24 How is recurrent unit in LSTM different from RNN?

Q. 25 Explain RNN architecture and list applications of RNN.

Q. 26 Write a short note: Deep RNN.

Q. 27 Implement XOR function using Mc-Culloch-Pitts neuron (consider binary data).

Q. 28 Implement OR function using Mc-Culloch-Pitts neuron.

Chapter Ends...

□□□

MODULE 5

CHAPTER 5 Advanced ML Classification Techniques

University Prescribed Syllabus w.e.f Academic Year 2022-2023

Ensemble Classifiers : Introduction to Ensemble Methods, Bagging, Boosting, Random forests, Improving classification accuracy of Class Imbalanced Data. Metrics for Evaluating Classifier Performance, Holdout Method and Random Subsampling, Cross-Validation, Bootstrap, Model Selection Using Statistical Tests of Significance, Comparing Classifiers Based on Cost–Benefit and ROC Curves.

Self-learning Topics : Introduction to ML (Revision), Introduction to Reinforcement Learning.

5.1 CLASSIFICATION

We have studied classification and different classifiers earlier. Let's recall some key concepts of classification.

(1) **Classification** is a form of data analysis that extracts models describing data classes. It is a Supervised learning technique that is used to identify the category of new observations on the basis of training data. A classifier, or classification model, predicts categorical labels (classes). Numeric prediction models continuous-valued functions. Classification and numeric prediction are the two major types of prediction problems.

(2) **Decision tree induction** is a top-down recursive tree induction algorithm, which uses an attribute selection measure to select the attribute tested for each non-leaf node in the tree. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome. ID3, C4.5, and CART are examples of such algorithms using different attribute selection measures. Tree pruning algorithms attempt to improve accuracy by removing tree branches reflecting noise in the data. Early decision tree algorithms typically assume that the data are memory resident.

(3) **Naive Bayesian classification** is based on Bayes' theorem of posterior probability. It assumes class-conditional independence that the effect of an attribute value on a given class is independent of the values of the other attributes. It is a probabilistic classifier, which means it predicts on the basis of the probability of an object. Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

(4) **A rule-based classifier** uses a set of IF-THEN rules for classification. Rules can be extracted from a decision tree. These rules are easily interpretable and thus these classifiers are generally used to generate descriptive models. The condition used with "if" is called the antecedent and the predicted class of each rule is called the consequent. Rules may also be generated directly from training data using sequential covering algorithms.

5.2 MODEL EVALUATION AND SELECTION

- Once our classification model is ready, we would like an estimate of how accurately the classifier can predict/classify the output class.
- Based on this, we will come to know whether training done is sufficient or not. We can even think of building more than one classifier and then compare their accuracy.
- Let's see now, what is accuracy? How can we estimate it? Are some measures of a classifier's accuracy more appropriate than others? How can we obtain a reliable accuracy estimate?

5.2.1 Metrics for Evaluating Classifier Performance

- The following list depicts various metrics/measures of evaluating how "accurate" your classifier is at predicting the class label of tuples :
  (1) Accuracy (2) Error rate
  (3) Precision (4) Sensitivity (Recall)
  (5) Specificity (6) F1 score
  (7) AOC-ROC (8) Log Loss
- Before discussing these measures, we need to understand with certain terminologies related to Confusion matrix.
- It is the easiest way to measure the performance of a classification problem where the output can be of two or more type of classes.
- **A confusion matrix** is nothing but a table with two dimensions used for analyzing how well your classifier can recognize tuples of different classes viz. "Actual" and "Predicted" and furthermore, both the dimensions have "True Positives (TP)", "True Negatives (TN)", "False Positives (FP)", "False Negatives (FN)" as shown below:

		Actual Class 1	Actual Class 0	Total
Predicted Class	1	Ture Positives (TP)	False Positives (FP)	P
	0	False Negatives (FN)	True Negatives (TN)	N
		P'	N'	P + N

- **True Positives (TP)** : These refers to the positive tuples that were correctly labeled by the classifier. It is the case when both actual class and predicted class of data point is 1.
- **True Negatives (TN)** : These are the negative tuples that were correctly labeled by the classifier. It is the case when both actual class and predicted class of data point is 0.
- **False Positives (FP)** : These are the negative tuples that were incorrectly labeled as positive. It is the case when actual class of data point is 0 and predicted class of data point is 1.
- **False Negatives (FN)** : These are the positive tuples that were mislabeled as negative. It is the case when actual class of data point is 1 and predicted class of data point is 0.
- Here, TP and TN tell us when the classifier is getting things right, while FP and FN tell us when the classifier is getting things wrong. Now let's understand various evaluation measures.

1. Accuracy (Recognition rate)

- The accuracy of a classifier on a given test set is the percentage of test set tuples that are correctly classified by the classifier.
- In other words, Accuracy is the ratio of the number of correct predictions and the total number of predictions. The formula for Accuracy is:

$$\text{Accuracy} = \frac{TP + TN}{P + N}$$

- Accuracy is useful when the target class is well balanced but is not a good choice for the unbalanced classes.
- Whereas other measures, such as sensitivity (or recall), specificity, precision, F, and Fβ, are better suited to the class imbalance problem, where the main class of interest is rare.
- In reality, data is always imbalanced for example Spam email, credit card fraud, and medical diagnosis.
- Hence, if we want to do a better model evaluation and have a full picture of the model evaluation, other metrics such as recall and precision should also be considered.

2. Error/Misclassification rate

- Error rate for a classifier, M, is simply 1 – accuracy(M), where accuracy(M) is the accuracy of the model M.
- We can also write this as :

$$\text{Error rate} = \frac{FP + FN}{P + N}$$

3. Recall/Sensitivity

- Recall is a measure of completeness i.e., what percentage of positive tuples are labeled as such.
- Sensitivity refers to True Positive (recognition) rate which is the proportion of positive tuples that are correctly identified.
- Recall and Sensitivity are similar

$$\text{Recall/Sensitivity} = \frac{TP}{TP + FN} = \frac{TP}{P}$$

4. Specificity

Specificity is the true negative rate which is the proportion of negative tuples that are correctly identified.

$$\text{Specificity} = \frac{TN}{N}$$

5. Precision

Precision is a measure of exactness i.e., what percentage of tuples labeled as positive are actually such.

$$\text{Precision} = \frac{TP}{TP + FP}$$

6. F-Score

- Precision and Recall are combined into a single measure to form F measures (also known as the F1 score or F-score).
- F1 Score is the harmonic mean of precision and recall. It is maximum when Precision is equal to Recall.

$$F = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$F\beta = \frac{(1 + \beta 2) * \text{Precision} * \text{Recall}}{\beta 2\ \text{Precision} + \text{Recall}}$$

7. AUC-ROC

- The Area Under the Curve (AUC) is the measure of the ability of a classifier to distinguish between classes.

- And, The Receiver Operator Characteristic (ROC) is a probability curve that plots the TPR(True Positive Rate) against the FPR(False Positive Rate) at various threshold values and separates the 'signal' from the 'noise'.
- In a ROC curve, the X-axis value shows False Positive Rate/Recall and Y-axis shows True Positive Rate/Sensitivity. Higher the value of X means higher the number of False Positives (FP) than True Negatives (TN), while a higher Y-axis value indicates a higher number of TP than FN. So, the choice of the threshold depends on the ability to balance between FP and FN.
- From the graph shown in Fig. 5.2.1, the greater the AUC, the better is the performance of the model at different threshold points between positive and negative classes.
- This simply means that when AUC is equal to 1, the classifier is able to perfectly distinguish between all Positive and Negative class points.
- When AUC is equal to 0, the classifier would be predicting all Negatives as Positives and vice versa. When AUC is 0.5, the classifier is not able to distinguish between the Positive and Negative classes.

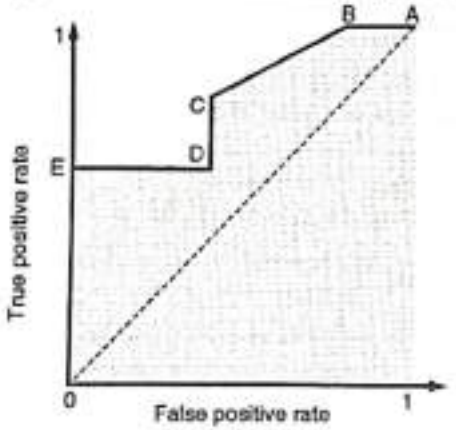


Fig. 5.2.1 : ROC Curve

8. Logarithmic Loss (Log Loss)

- It is also called Logistic regression loss or cross-entropy loss.
- It basically defined on probability estimates and measures the performance of a classification model where the input is a probability value between 0 and 1.
- It can be understood more clearly by differentiating it with accuracy.
- As we know that accuracy is the count of predictions (predicted value = actual value) in our model whereas Log Loss is the amount of uncertainty of our prediction based on how much it varies from the actual label.
- With the help of Log Loss value, we can have more accurate view of the performance of our model.
- Let's calculate these metrics for some reasonable real-world numbers. If we have 100,000 patients, of which 200 (20%) actually have cancer, we might see the following test results:

	Test Positive	Test Negative	Total
Patient Diseased	160	40	200
Patient Healthy	29940	69860	99800
Total	30100	69900	100000

For this data :

Sensitivity $= TP/(TP + FN) = 160 / (160 + 40) = 80.0\%$

Specificity $= TN/(TN + FP) = 69,860 / (69,860 + 29,940)$

$= 70.0\%$

- In other words, our test will correctly identify 80% of people with the disease, but 30% of healthy people will incorrectly test positive.
- By only considering the sensitivity (or accuracy) of the test, potentially important information is lost.
- However, classifiers can also be compared with respect to the following additional aspects, in addition to accuracy-based measures :

(1) **Speed :** Speed refers to the computational costs involved in creating and using the given classifier.

(2) **Robustness :** Robustness is the ability of the classifier to make correct predictions when the dataset contains noisy data or data with missing values. Robustness is typically assessed with a series of synthetic data sets representing increasing degrees of noise and missing values.

(3) **Scalability :** Scalability refers to the ability to construct the classifier efficiently given large amounts of data. Scalability is typically assessed with a series of data sets of increasing size.

(4) **Interpretability** : Interpretability refers to the level of understanding and insight that is provided by the classifier. Interpretability is subjective and therefore it is difficult to assess.

5.2.2 Holdout Method and Random Subsampling

- This is the simplest method to evaluate the classifier. In the **holdout method**, the given data set is randomly partitioned into two independent sets, a training set and a test set.
- Typically, we allocate two-thirds of the data to the training set, and the remaining one-third is allocated to the test set. For example, if there are 2000 data items present, 1200 are placed in training set and remaining 800 are placed in test set.
- The training set is used to train the model. The model is then validated with the test set to get accuracy, error rate and error estimate of model/classifier as shown in Fig. 5.2.2.
- The estimate of accuracy is pessimistic here because only a portion of the initial data is used to derive the model.
- If maximum possible data are allocated in training set for construction of model/classifier, then the classifier's error rates and estimates would be very low and accuracy would be high.
- The Hold-out method is not well suited for sparse data-set. Sparse data set is the data set in which classes are not equally distributed.

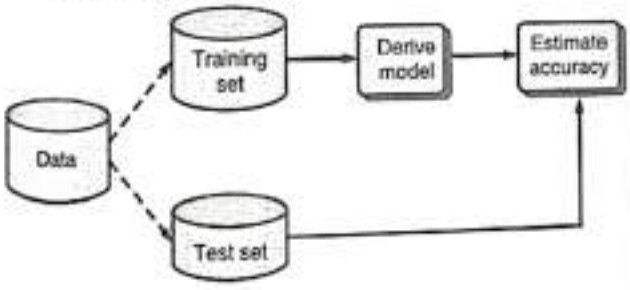


Fig. 5.2.2 : Holdout Method

- **Random subsampling** is a variation of the holdout method in which the holdout method is repeated k times. In this method, we form 'k' replica of given data. For each iteration (for each replica) a fix number of observation is chosen and it is kept aside as test set.
- The model is fitted to training set from each iteration, and an estimate of prediction error is obtained from each test set. The overall accuracy estimate is taken as the average of the accuracies obtained from each iteration.
- It is better approach than Holdout method for sparse dataset. But there is a chance of selecting same record in test set for other iteration.

5.2.3 Cross-Validation

- In **k-fold cross-validation**, the basic idea is to split the data into k chunks. Train with (k – 1) chunks of data objects, test it on $(k-1)^{th}$ chunk. Next time, include the one that you excluded in the last iteration to train your model and test it on this excluded chunk. Repeat. This should take you K iterations i.e., training and testing is performed k times.
- So, the data set is randomly partitioned into k mutually exclusive subsets or folds, D_1, D_2,..., D_k, each of approximately equal size. In iteration i, partition D_i is reserved as the test set, and the remaining partitions are collectively used to train the model.
- That is, in the first iteration, subsets D_2,..., D_k collectively serve as the training set to obtain a first model, which is tested on D_1; the second iteration is trained on subsets D_1, D_3,..., D_k and tested on D_2; and so on.
- Unlike the holdout and random subsampling methods, here each sample is used the same number of times for training and once for testing. For classification, the accuracy estimate is the overall number of correct classifications from the k iterations, divided by the total number of tuples in the initial data. This method is good for huge data.
- One can be a victim of skewed target values with Random subsampling and K-fold which can be fixed by stratification. **Stratification** makes sure that you get similar target distribution in each of your folds (chunks) of your data.
- **Leave-one-out** is a special case of k-fold cross-validation where k is set to the number of initial tuples.

- That is, only one sample is "left out" at a time for the test set. In stratified cross-validation, the folds are stratified so that the class distribution of the tuples in each fold is approximately the same as that in the initial data.
- For example, if K = 1000. Train your model with 999 objects, test it on 1000^{th} object. Repeat this with leaving out one object each time.
- This should take you 1000 iterations. In our example, this is 1000 times slower than random subsampling.
- This method is good for less data, unbalanced dataset and target values.
- In general, stratified 10-fold cross-validation is recommended for estimating accuracy (even if computation power allows using more folds) due to its relatively low bias and variance.

5.2.4 Bootstrap

- **Bootstrapping method** is a resampling statistical technique that evaluates statistics of a given population by testing a dataset by replacing the sample.
- This technique involves repeatedly sampling a dataset with random replacement.
- That is, each time a tuple is selected, it is equally likely to be selected again and re-added to the training set.
- For instance, imagine a machine that randomly selects tuples for the training set. In sampling with replacement, the machine is allowed to select the same tuple more than once.
- Steps that are involved in the Bootstrapping method:
  1. Randomly choose a sample size.
  2. Pick an observation from the training dataset in random order.
  3. Combine this observation with the sample chosen earlier.
- For the samples that are chosen in the representative sample size, they are referred to as the 'Bootstrapped samples' or the bootstrap sample size.
- On the other hand, the samples that are not chosen are referred to as the 'Out-of-the-bag' samples that serve as the testing dataset.
- Unlike other sampling distribution methods, the Bootstrapping method uses the samples procured from a study over and over again in order to use the replacement technique and ensure that the stimulated samples lead to an accurate evaluation.
- Other than ensuring the sampling of the accuracy of a given dataset, bootstrapping in statistics also allows one to estimate the confidence intervals of a given dataset.
- Bootstrapping method can be either parametric or non-parametric.
- In parametric bootstrap method, the distribution parameter must be known.
- This means that the assumption of the kind of distribution the sample has must be provided beforehand. Unlike the parametric bootstrap method, non-parametric bootstrap does not require the parameter of distribution to be known beforehand.
- Therefore, this type of bootstrap method works without assuming the nature of the sample distribution.

5.2.5 Model Selection Using Statistical Tests of Significance

- Comparing machine learning classification methods and selecting a final model is a common operation in applied machine learning.
- Suppose that we have generated two classification models, M_1 and M_2, from our data. Models are commonly evaluated using resampling methods like k-fold cross-validation from which mean skill scores are calculated and compared directly.
- Suppose that we have performed 10-fold cross-validation to obtain a mean error rate for each model. How can we determine which model is best?
- We may wish to select the model with the lowest error rate; however, the mean error rates are just estimates of error on the true population of future data cases.
- There can be considerable variance between error rates within any given 10-fold cross-validation experiment. Although the mean error rates obtained for M_1 and M_2 may appear different, that difference may not be statistically significant.

- Though this approach is simple, this can be misleading as it is hard to know whether the difference between mean skill scores is real or the result of a statistical chance.
- Statistical significance tests are designed to address this problem and quantify the likelihood of the samples of skill scores being observed given the assumption that they were drawn from the same distribution.
- If this assumption, or null hypothesis, is rejected, it suggests that the difference in skill scores is statistically significant.
- Generally, a statistical hypothesis test for comparing samples calculates how likely it is to observe two data samples given the assumption that the samples have the same distribution.
- The assumption of a statistical test is called the **null hypothesis** and we can calculate statistical measures and interpret them in order to decide whether or not to accept or reject the null hypothesis.
- What do we need to perform the statistical test? Suppose that for each model, we did 10-fold cross-validation, say, 10 times, each time using a different 10-fold data partitioning.
- We can average the 10 error rates obtained each for M_1 and M_2, respectively, to obtain the mean error rate for each model. For a given model, the individual error rates calculated in the cross-validations may be considered as different, independent samples from a probability distribution.
- In general, they follow a t-distribution with $k - 1$ degrees of freedom, here, $k = 10$.
- This allows us to do hypothesis testing where the significance test used is the **t-test**, or **Student's t-test**.
- Our hypothesis is that the two models are the same, or in other words, that the difference in mean error rate between the two is zero.
- If we can reject this null hypothesis, then we can conclude that the difference between the two models is statistically significant, and we can select the model with the lower error rate.
- Comparing machine learning models via statistical significance tests imposes some expectations that in turn will impact the types of statistical tests that can be used; for example:
  (1) **Skill Estimate :** A specific measure of model skill must be chosen. This could be classification accuracy (a proportion) or mean absolute error (summary statistic) which will limit the type of tests that can be used.
  (2) **Repeated Estimates :** A sample of skill scores is required in order to calculate statistics. The repeated training and testing of a given model on the same or different data will impact the type of test that can be used.
  (3) **Distribution of Estimates :** The sample of skill score estimates will have a distribution, perhaps Gaussian or perhaps not. This will determine whether parametric or nonparametric tests can be used.
  (4) **Central Tendency :** Model skill will often be described and compared using a summary statistic such as a mean or median, depending on the distribution of skill scores. The test may or may not take this directly into account.
- To determine whether M_1 and M_2 are significantly different, we compute t and select a significance level, $sig(\alpha)$.
- In practice, a significance level (α) of 5% or 1% is typically used. We then consult a table for the t-distribution. This distribution is available in standard textbooks on statistics.
- However, because the t-distribution is symmetric, typically only the upper percentage points of the distribution are shown.
- Therefore, we look up the table value for $z = \alpha/2$, which in this case is 0.025, where z is also referred to as a **confidence limit**.
- If $t > z$ or $t < -z$, then our value of t lies in the rejection region, within the distribution's tails.
- This means that we can reject the null hypothesis that the means of M_1 and M_2 are the same and conclude that there is a statistically significant difference between the two models.

- Otherwise, if we cannot reject the null hypothesis, we conclude that any difference between M_1 and M_2 can be attributed to chance.
- The t-statistic for **pairwise comparison** is computed as:

$$t = \frac{\overline{err}(M_1) - \overline{err}(M_2)}{\sqrt{var(M_1 - M_2)/k}}$$

Where,

$$var(M_1 - M_2) = \frac{1}{k}\sum_{i=1}^{k}[err(M_1)_i - err(M_2)_i - [\overline{err}(M_1) - \overline{err}(M_2)]]^2$$

- If two test sets are available instead of a single test set, then a **nonpaired version** of the t-test is used, where the variance between the means of the two models can be computed as:

$$var(M_1 - M_2) = \sqrt{\frac{var(M_1)}{k_1} + \frac{var(M_2)}{k_2}}$$

and k_1 and k_2 are the number of cross-validation samples used for M_1 and M_2, respectively. This is also known as the two-sample t-test. While consulting the table of t-distribution, in such case, the number of degrees of freedom used is taken as the minimum number of degrees of the two models.

5.2.6 Comparing Classifiers Based on Cost–Benefit and ROC Curves

- For assessing the costs (risks) and benefits (gains) associated with a classification model, the true positives, true negatives, false positives, and false negatives are useful.
- The cost associated with a false negative (such as incorrectly predicting that a diabetic patient is not diabetic) is far greater than those of a false positive (incorrectly yet conservatively labeling a nondiabetic patient as diabetic).
- In such cases, we can overshadow one type of error over another by assigning a different cost to each.
- These costs may consider the danger to the patient, financial costs of resulting therapies, and other hospital costs.
- Similarly, the benefits associated with a true positive decision may be different than those of a true negative.
- Till now, to compute classifier accuracy, we have assumed equal costs and essentially divided the sum of true positives and true negatives by the total number of test tuples.
- However, we can integrate costs and benefits by instead computing the average cost (or benefit) per decision.
- Other applications involving cost-benefit analysis include loan application decisions and target marketing mailouts.
- **ROC (Receiver operating characteristic) curves** are a useful visual tool for comparing two classification models.
- We have studied previously, an ROC curve for a given model shows the trade-off between the true positive rate (TPR) and the false positive rate (FPR).
- The area under the ROC curve is a measure of the accuracy of the model.
- Any increase in TPR occurs at the cost of an increase in FPR. For a two-class problem, an ROC curve allows us to visualize the trade-off between the rate at which the model can accurately recognize positive cases versus the rate at which it mistakenly identifies negative cases as positive for different portions of the test set.
- It is immediately apparent that a ROC curve can be used to select a threshold for a classifier which maximizes the true positives, while minimizing the false positives.
- However, different types of problems have different optimal classifier thresholds. For a cancer screening test, for example, we may be prepared to put up with a relatively high false positive rate in order to get a high true positive, it is most important to identify possible cancer sufferers.
- For a follow-up test after treatment, however, a different threshold might be more desirable, since we want to minimize false negatives, we don't want to tell a patient they're clear if this is not actually the case.
- The **AUC** can be used to compare the performance of two or more classifiers.
- A single threshold can be selected and the classifiers' performance at that point compared, or the overall performance can be compared by considering the AUC.

- Most published reports compare AUCs in absolute terms: "Classifier 1 has an AUC of 0.85, and classifier 2 has an AUC of 0.79, so classifier 1 is clearly better".
- It is, however, possible to calculate whether differences in AUC are statistically significant.

5.3 TECHNIQUES TO IMPROVE CLASSIFICATION ACCURACY

- In machine learning, no matter if we are facing a classification or a regression problem, the choice of the model is extremely important to have any chance to obtain good results.
- This choice can depend on many variables of the problem: quantity of data, dimensionality of the space, distribution hypothesis, etc.
- In ensemble learning theory, we call weak learners (or base models) models that can be used as building blocks for designing more complex models by combining several of them.
- The idea of ensemble methods is to try reducing bias and/or variance of such weak learners by combining several of them together in order to create a strong learner (or ensemble model) that achieves better performances.
- To outline the definition and practicality of Ensemble Methods, here we have used example of Decision tree classifier. However, it is important to note that Ensemble Methods do not only pertain to Decision Trees.
- A decision tree determines the predictive value based on series of questions and conditions. For instance, the simple Decision Tree shown in Fig. 5.3.1 determines on whether an individual should play outside or not.
- The tree takes several weather factors into account, and given each factor either makes a decision or asks another question. In this example, every time it is overcast, we will play outside.
- However, if it is raining, we must ask if it is windy or not? If windy, we will not play.
- But given no wind, tie those shoelaces tight because were going outside to play.

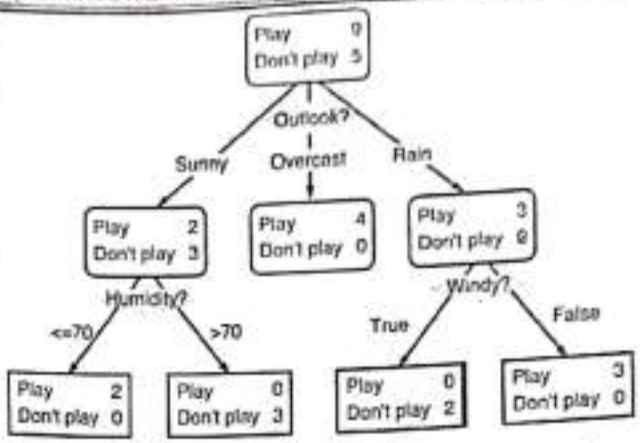


Fig. 5.3.1 : A decision tree to determine whether to play outside or not

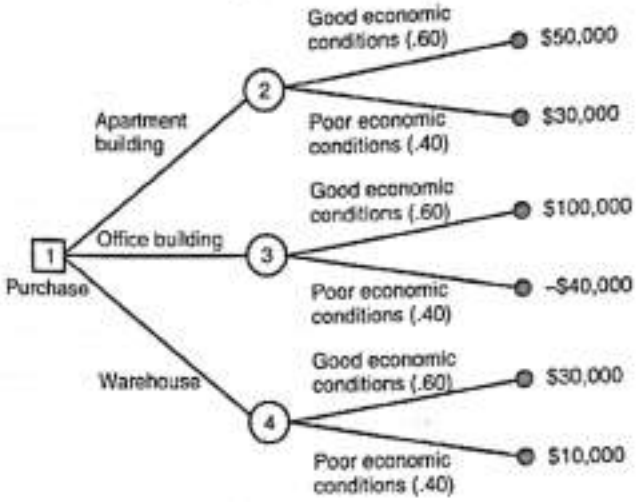


Fig. 5.3.2 : A decision tree to determine whether or not to invest in real estate

- Decision Trees can also solve quantitative problems as well with the same format. In the Tree to the left, we want to know whether or not to invest in a commercial real estate property. Is it an office building?
- A Warehouse? An Apartment building? Good economic conditions? Poor Economic Conditions? How much will an investment return? These questions are answered and solved using this decision tree.
- When making Decision Trees, there are several factors we must take into consideration: On what features do we make our decisions on? What is the threshold for classifying each question into a yes or no answer? In the first Decision Tree, what if we wanted to ask ourselves if we had friends to play with or not.

- If we have friends, we will play every time. If not, we might continue to ask ourselves questions about the weather. By adding an additional question, we hope to greater define the Yes and No classes.
- This is where Ensemble Methods come into picture! Rather than just relying on one Decision Tree and hoping we made the right decision at each split, Ensemble Methods allow us to take a sample of Decision Trees into account, calculate which features to use or questions to ask at eac h split, and make a final predictor based on the aggregated results of the sampled Decision Trees.

5.3.1 Introducing Ensemble Methods

- **Ensemble methods** is a machine learning technique that combines several base models in order to produce one optimal predictive model which helps to improve machine learning results.
- This approach allows the production of better predictive performance compared to a single model. Basic idea is to learn a set of classifiers and to allow them to vote. Ensembles tend to be more accurate than their component classifiers.

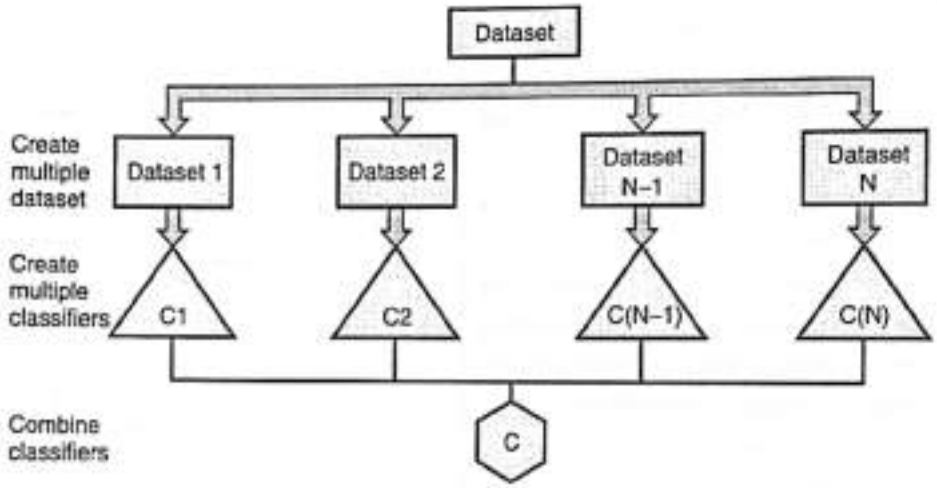


Fig. 5.3.3 : An overview of Ensemble methods/learning

- Different types of ensemble classifiers are:
  1. Bagging
  2. Boosting and AdaBoost
  3. Random Forests

5.3.2 Bagging (Bootstrap Aggregating)

- This approach combines **B**ootstrapping and **Agg**regation to form one ensemble model, that's why the name is **Bagging**.
- Consider yourself as a patient and you would like to have a diagnosis made based on the symptoms. Instead of asking one doctor, you may choose to ask several.
- If a certain diagnosis occurs more than any other, you may choose this as the final or best diagnosis.
- That is, the final diagnosis is made based on a majority vote, where each doctor gets an equal vote.
- If we replace each doctor by a classifier, and that's the basic idea behind bagging.
- Naturally, a majority vote made by a large group of doctors may be more reliable than a majority vote made by a small group.

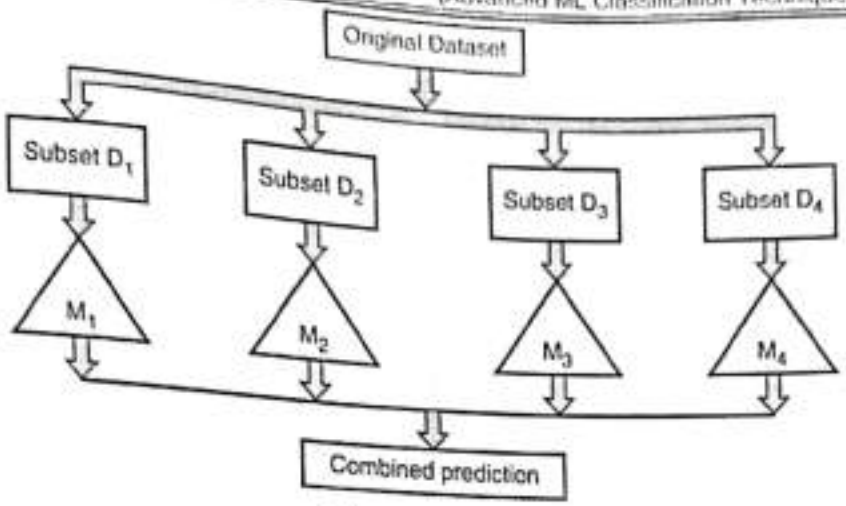


Fig. 5.3.4 : Bagging

- Given a sample of data, multiple bootstrapped subsamples are pulled.
- A Decision Tree is formed on each of the bootstrapped subsamples.
- Each training set is a bootstrap sample. After each subsample Decision Tree has been formed, an algorithm is used to aggregate over the Decision Trees to form the most efficient predictor.
- To classify an unknown tuple, X, each classifier, M_i, returns its class prediction, which counts as one vote. The bagged classifier, M*, counts the votes and assigns the class with the most votes to X.
- Bagging often considers homogeneous weak learners, learns them independently from each other in parallel and combines them following some kind of deterministic averaging process Bagging can be applied to the prediction of continuous values by taking the average value of each prediction for a given test tuple.

5.3.3 Boosting and AdaBoost

- Consider the same example that was taken in previous section, you as a patient, you have certain symptoms.
- Now, instead of consulting one doctor, you choose to consult several.
- Suppose you assign weights to the value or worth of each doctor's diagnosis, based on the accuracies of previous diagnoses they have made.
- The final diagnosis is then a combination of the weighted diagnoses. This is the basic idea behind **boosting.**
- Boosting often considers homogeneous weak learners, learns them sequentially in a very adaptative way (a base model depends on the previous ones) and combines them following a deterministic strategy.
- In boosting, weights are also assigned to each training tuple.
- A series of k classifiers is iteratively learned. After a classifier, M_i, is learned, the weights are updated to allow the subsequent classifier, M_{i+1}, to "pay more attention" to the training tuples that were misclassified by M_i.
- The final boosted classifier, M*, combines the votes of each individual classifier, where the weight of each classifier's vote is a function of its accuracy.
- In **adaptative boosting** (often called "adaboost"), we try to define our ensemble model as a weighted sum of L weak learners. It's a popular boosting algorithm.
- The basic idea is that when we build a classifier, we want it to focus more on the misclassified tuples of the previous round.
- Some classifiers may be better at classifying some "difficult" tuples than others. In this way, we build a series of classifiers that complement each other.
- We are given D, a data set of d class-labeled tuples, $(X_1, y_1),(X_2, y_2),...,(X_d, y_d)$, where y_i is the class label of tuple Xi.
- Initially, AdaBoost assigns each training tuple an equal weight of 1/d. Generating k classifiers for the ensemble requires k rounds through the rest of the algorithm.

- In round i, the tuples from D are sampled to form a training set, D_i, of size d. Sampling with replacement is used. This indicates the same tuple may be selected more than once.
- Each tuple's chance of being selected is based on its weight. A classifier model, M_i, is derived from the training tuples of D_i.
- Its error is then calculated using D_i as a test set. The weights of the training tuples are then adjusted according to how they were classified.
- If a tuple was incorrectly classified, its weight is increased. If a tuple was correctly classified, its weight is decreased.
- A tuple's weight reflects how difficult it is to classify. The higher the weight, the more often it has been misclassified.
- These weights will be used to generate the training samples for the classifier of the next round. This is how, a series of classifiers that complement each other are built.
- To compute the error rate of model M_i, we sum the weights of each of the tuples in D_i that M_i misclassified.

$$\text{error}(M_i) = \sum_{j=1}^{d} w_j \times \text{err}(X_j)$$

where $\text{err}(X_j)$ is the misclassification error of tuple X_j: If the tuple was misclassified, then $\text{err}(X_j)$ is 1; otherwise, it is 0. If the performance of classifier Mi is so poor that its error exceeds 0.5, then we abandon it. Instead, we try again by generating a new D_i training set, from which we derive a new M_i.

- The error rate of M_i affects how the weights of the training tuples are updated.
- If a tuple in round i was correctly classified, its weight is multiplied by $\text{error}(M_i)/(1 - \text{error}(M_i))$.
- Once the weights of all the correctly classified tuples are updated, the weights for all tuples (including the misclassified ones) are normalized so that their sum remains the same as it was before.
- To normalize a weight, we multiply it by the sum of the old weights, divided by the sum of the new weights.
- As a result, the weights of misclassified tuples are increased and the weights of correctly classified tuples are decreased.
- To predict a class label for a tuple X, boosting assigns a weight to each classifier's vote, based on how well the classifier performed.
- The lower a classifier's error rate, the more accurate it is, and therefore, the higher its weight for voting should be. The weight of the classifier's vote is calculated as :

$$\log \frac{1 - \text{error}(M_i)}{\text{error}(M_i)}$$

- For each class, c, we sum the weights of each classifier that assigned class c to X.
- The class with the highest sum is the "winner" and is returned as the class prediction for tuple X.
- Bagging is less susceptible to model overfitting. While bagging and boosting, both can significantly improve accuracy in comparison to a single model, boosting tends to achieve greater accuracy.

5.3.4 Random Forests

- **Random Forest** Models can be thought of as extension over bagging, as it is bagging with a slight twist. Each classifier in the ensemble is a decision tree classifier so that the collection of classifiers is a "forest".
- Classifier is generated using a random selection of attributes at each node to determine the split. During classification, each tree votes and the most popular class is returned.
- When deciding where to split and how to make decisions, bagged Decision Trees have the full disposal of features to choose from.
- Therefore, although the bootstrapped samples may be slightly different, the data is largely going to break off at the same features throughout each model.
- In contrary, Random Forest models decide where to split based on a random selection of features.
- Rather than splitting at similar features at each node throughout, Random Forest models implement a level of differentiation because each tree will split based on different features.
- This level of differentiation provides a greater ensemble to aggregate over, producing a more accurate predictor.

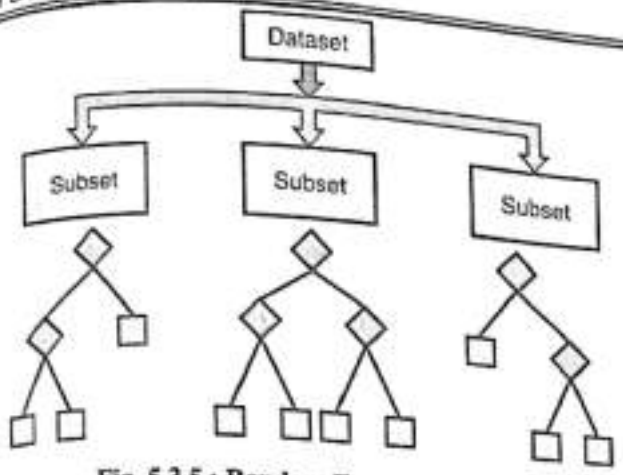


Fig. 5.3.5 : Random Forest Classifier

- Steps for implementing Random Forest Classifier :
  1. Multiple subsets are created from the original data set, selecting observations with replacement.
  2. A subset of features is selected randomly and whichever feature gives the best split is used to split the node iteratively.
  3. The tree is grown to the largest.
  4. Repeat the above steps and prediction is given based on the aggregation of predictions from n number of trees.

5.3.5 Improving Classification Accuracy of Class-Imbalanced Data

- When observation in one class is higher than the observation in other classes then there exists a class imbalance.
- Class Imbalance is a common problem in machine learning, especially in classification problems. Imbalance data can hamper our model accuracy big time. If the data set is imbalanced then in such cases, you get a pretty high accuracy just by predicting the majority class, but you fail to capture the minority class, which is most often the point of creating the model in the first place.
- Having an imbalanced dataset (imbalanced target variables) in a problem statement is always frustrating, and having a perfectly balanced dataset is always a myth.
- Mostly in Medical Science/ Healthcare Machine Learning problems, the data set is mostly biased.
- So, predicting outcome in such cases becomes very difficult as data becomes biased towards one particular class of outcome.
- Class Imbalance appear in many domains such as:
  (1) Fraud detection
  (2) Spam filtering
  (3) Disease screening
  (4) SaaS subscription churn
  (5) Advertising click-throughs
- If we consider a healthcare problem, our main goal is to reduce false Positive outcomes, as you cannot afford to let patients go away with a disease because of a biased algorithm.
- Since there are more 'Negatives' in a dataset, the Machine Learning Model becomes biased toward Negative Class.
- So, in some cases, it might predict 'Negative' for a 'Positive' Class.
- There are many ways to reduce the Class Imbalance/Bias Problem and improving the classification accuracy of class-imbalanced data:
  1. **Improve Data Collection and Preprocessing Techniques:** Collect more data and give much more time to preprocessing by detecting outliers and segment the data according to balanced class.
  2. **Resampling (Up Sampling and Down Sampling):** A widely adopted technique for dealing with highly unbalanced datasets is called resampling. It consists of removing samples from the majority class (under-sampling) and/or adding more examples from the minority class (over-sampling). If you have less data, then this technique is quite useful.

     Up(Over) Sampling is increasing the number of classes which is less in number by considering the data points closer to that of the original class. The simplest implementation of over-sampling is to duplicate random records from the minority class, which can cause overfishing.

     Down Sampling is the inverse of oversampling, i.e., reducing the number of classes having a higher number of data points. In under-sampling, the simplest technique involves removing random records from the majority class, which can cause loss of information.

3. **The threshold-moving :** This approach to the class imbalance problem does not involve any sampling. It applies to classifiers that, given an input tuple, return a continuous output value. That is, for an input tuple, X, such a classifier returns as output a mapping, f (X) → [0,1]. Rather than manipulating the training tuples, this method returns a classification decision based on the output values. In the simplest approach, tuples for which f (X) ≥ t, for some threshold, t, are considered positive, while all other tuples are considered negative. Other approaches may involve manipulating the outputs by weighting.

   In general, threshold moving moves the threshold, t, so that the rare class tuples are easier to classify and hence, there is less chance of costly false negative errors. Examples of such classifiers include naive Bayesian classifiers and neural network classifiers like backpropagation.

4. **Use Specific Algorithm Properties:** This is helpful for some Machine Learning Algorithms where you can give weights to a particular class, which may decrease bias. For example, you have 70% A Class and 30% B Class, where you can give more weight on A class because Algorithm may tend to be more biased towards Class B.

5. **Ensemble methods** discussed in section have also been applied to the class imbalance problem. The individual classifiers making up the ensemble may include versions of the approaches described here such as oversampling and threshold moving.

5.4 REINFORCEMENT LEARNING (RL)

- **Reinforcement learning** is an area of Machine Learning in which an agent learns to behave in an environment by performing the actions and seeing the results of actions. It is a feedback-based Machine learning technique. It is about taking suitable action to maximize reward in a particular situation. For each good action, the agent gets positive feedback, and for each bad action, the agent gets negative feedback or penalty.
- In Reinforcement Learning, the agent learns automatically using feedbacks without any labeled data, unlike supervised learning. Since there is no labeled data, so the agent is bound to learn by its experience only.

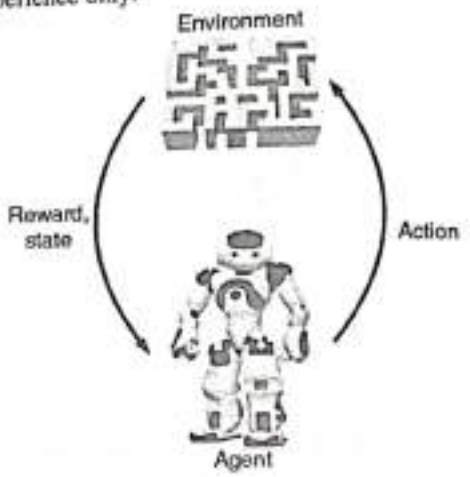


Fig. 5.4.1 : Reinforcement Learning

- RL solves a specific type of problem where decision making is sequential, and the goal is long-term, such as game-playing, robotics, etc.
- The agent interacts with the environment and explores it by itself. The primary goal of an agent in reinforcement learning is to improve the performance by getting the maximum positive rewards.
- The agent learns with the process of hit and trial, and based on the experience, it learns to perform the task in a better way. Hence, we can say that "Reinforcement learning is a type of machine learning method where an intelligent agent (computer program) interacts with the environment and learns to act within that." How a Robotic dog learns the movement of his arms is an example of Reinforcement learning.
- It is a core part of Artificial intelligence, and all AI agent works on the concept of reinforcement learning.
- Here we do not need to pre-program the agent, as it learns from its own experience without any human intervention.
- **Example :** Suppose there is an AI agent present within a maze environment, and his goal is to find the diamond. The agent interacts with the environment by performing some actions, and based on those actions, the state of the agent gets changed, and it also receives a reward or penalty as feedback.

- The agent continues doing these three things (take action, change state/remain in the same state, and get feedback), and by doing these actions, he learns and explores the environment.
- The agent learns that what actions lead to positive feedback or rewards and what actions lead to negative feedback penalty. As a positive reward, the agent gets a positive point, and as a penalty, it gets a negative point.
- It is employed by various software and machines to find the best possible behavior or path it should take in a specific situation.
- Reinforcement learning differs from supervised learning in a way that in supervised learning the training data has the answer key with it so the model is trained with the correct answer itself whereas in reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task. In the absence of a training dataset, it is bound to learn from its experience.

5.5 MULTIPLE CHOICE QUESTIONS

Q. 5.1 Which of the following algorithm is not an example of an ensemble method?

(a) Extra Tree Regressor
(b) Random Forest
(c) Gradient Boosting
(d) Decision Tree

✓**Ans. : (d)**

Q. 5.2 What is true about an ensembled classifier?

1. Classifiers that are surer can vote with more conviction
2. Classifiers can be surer about a particular part of the space
3. Most of the times, it performs better than a single classifier

(a) 1 and 2 (b) 1 and 3
(c) 2 and 3 (d) All of the above

✓**Ans. : (d)**

Q. 5.3 Which of the following option is / are correct regarding benefits of ensemble model?

1. Better performance
2. Generalized models
3. Better interpretability

(a) 1 and 3 (b) 2 and 3
(c) 1 and 2 (d) 1, 2 and 3

✓**Ans. : (c)**

Q. 5.4 Which of the following can be true for selecting base learners for an ensemble?

1. Different learners can come from same algorithm with different hyper parameters
2. Different learners can come from different algorithms
3. Different learners can come from different training spaces

(a) 1 (b) 2
(c) 1 and 3 (d) 1, 2 and 3

✓**Ans. : (d)**

Q. 5.5 True or False: Ensemble learning can only be applied to supervised learning methods.

(a) True (b) False

✓**Ans. : (b)**

Q. 5.6 True or False: Ensembles will yield bad results when there is significant diversity among the models.

Note: All individual models have meaningful and good predictions.

(a) True (b) False

✓**Ans. : (b)**

Q. 5.7 Which of the following is / are true about weak learners used in ensemble model?

1. They have low variance and they don't usually overfit
2. They have high bias, so they can not solve hard learning problems
3. They have high variance and they don't usually overfit

(a) 1 and 2 (b) 1 and 3
(c) 2 and 3 (d) None of these

✓**Ans. : (a)**

Q. 5.8 True or False: Ensemble of classifiers may or may not be more accurate than any of its individual model.

(a) True (b) False

✓**Ans. : (a)**

Q. 5.9 If you use an ensemble of different base models, is it necessary to tune the hyper parameters of all base models to improve the ensemble performance?

(a) Yes (b) No
(c) can't say

✓**Ans. : (b)**

Q. 5.10 Generally, an ensemble method works better, if the individual base models have _____?

Note: Suppose each individual base models have accuracy greater than 50%.

(a) Less correlation among predictions
(b) High correlation among predictions
(c) Correlation does not have any impact on ensemble output
(d) None of the above

✓**Ans. : (a)**

Q. 5.11 In an election, N candidates are competing against each other and people are voting for either of the candidates. Voters don't communicate with each other while casting their votes. Which of the following ensemble method works similar to above-discussed election procedure?

Hint: Persons are like base models of ensemble method.

(a) Bagging (b) Boosting

(c) A Or B (d) None of these ✓**Ans. : (a)**

Q. 5.12 Suppose you are given 'n' predictions on test data by 'n' different models (M_1, M_2, ... M_n) respectively. Which of the following method(s) can be used to combine the predictions of these models?

Note: We are working on a regression problem

1. Median
2. Product
3. Average
4. Weighted sum
5. Minimum and Maximum
6. Generalized mean rule

(a) 1, 3 and 4 (b) 1,3 and 6

(c) 1,3, 4 and 6 (d) All of above ✓**Ans. : (d)**

Q. 5.13 How can we assign the weights to output of different models in an ensemble?

1. Use an algorithm to return the optimal weights
2. Choose the weights using cross validation
3. Give high weights to more accurate models

(a) 1 and 2 (b) 1 and 3

(c) 2 and 3 (d) All of above ✓**Ans. : (d)**

Q. 5.14 Which of the following is true about averaging ensemble?

(a) It can only be used in classification problem

(b) It can only be used in regression problem

(c) It can be used in both classification as well as regression

(d) None of these ✓**Ans. : (c)**

Q. 5.15 Suppose there are 25 base classifiers. Each classifier has error rates of e = 0.35. Suppose you are using averaging as ensemble technique. What will be the probabilities that ensemble of above 25 classifiers will make a wrong prediction?

Note : All classifiers are independent of each other

(a) 0.05 (b) 0.06

(c) 0.07 (d) 0.09 ✓**Ans. : (b)**

Q. 5.16 Which of the following parameters can be tuned for finding good ensemble model in bagging based algorithms?

1. Max number of samples
2. Max features
3. Bootstrapping of samples
4. Bootstrapping of features

(a) 1 and 3 (b) 2 and 3

(c) 1 and 2 (d) all of above ✓**Ans. : (d)**

Q. 5.17 For the below confusion matrix, what is the recall?

	Not 5	5
Not 5	53272	1307
5	1077	4344

(a) 0.7 (b) 0.8

(c) 0.9 (d) 9.95 ✓**Ans. : (b)**

Q. 5.18 Which among the following evaluation metrics would you NOT use to measure the performance of a classification model?

(a) Precision (b) Recall

(c) Mean Squared Error (d) F1 score ✓**Ans. : (c)**

Q. 5.19 Which of the following is correct use of cross validation?

(a) Selecting variables to include in a model

(b) Comparing predictors

(c) Selecting parameters in prediction function

(d) All of the mentioned ✓**Ans. : (d)**

Q. 5.20 Point out the wrong combination.

(a) True negative = correctly rejected

(b) False negative = correctly rejected

(c) False positive = correctly identified

(d) All of the mentioned ✓**Ans. : (c)**

Q. 5.21 Which of the following is a common error measure?

(a) Sensitivity

(b) Median absolute deviation

(c) Specificity

(d) All of the mentioned ✓**Ans. : (d)**

Q. 5.22 Which of the following cross validation versions may not be suitable for very large datasets with hundreds of thousands of samples?

(a) k-fold cross-validation

(b) Leave-one-out cross-validation

(c) Holdout method

(d) All of the above ✓**Ans. : (b)**

Q. 5.23 Which of the following is a disadvantage of k-fold cross-validation method?

(a) The variance of the resulting estimate is reduced as k is increased.

(b) This usually does not take longer time to compute

(c) Reduced bias

(d) The training algorithm has to rerun from scratch k times

✓**Ans. : (d)**

Q. 5.24 Reinforcement learning is-

(a) Unsupervised learning

(b) Supervised learning

(c) Award based learning

(d) None

✓**Ans. : (c)**

Q. 5.25 Which of the following is an application of reinforcement learning?

(a) Topic modeling

(b) Recommendation system

(c) Pattern recognition

(d) Image classification

✓**Ans. : (b)**

Q. 5.26 Which of the following is true about reinforcement learning?

(a) The agent gets rewards or penalty according to the action

(b) It's an online learning

(c) The target of an agent is to maximize the rewards

(d) All of the above

✓**Ans. : (d)**

Q. 5.27 If TP=9 FP=6 FN=26 TN=70 then Error rate will be

(a) 45 percentage

(b) 99 percentage

(c) 28 percentage

(d) 20 percentage

✓**Ans. : (c)**

Descriptive Questions

Q. 1 What is Reinforcement Learning? Explain the significance of Award and action in RL.

Q. 2 Which are the various metrics for evaluating the classifier performance?

Q. 3 What is confusion matrix? What are the contents of it?

Q. 4 Compare and contrast different metrics for evaluating the classifier performance.

Q. 5 How can we compare classifiers based on cost-benefit and ROC curves?

Q. 6 Write a short note on: Holdout method

Q. 7 How Cross validation method can be used to evaluate classifiers?

Q. 8 Write a note on: Bootstrap.

Q. 9 What is meant by ensemble learning? What are the different types of ensemble classifiers?

Q. 10 Explain how bagging method works.

Q. 11 How is boosting method different than bagging?

Q. 12 Write a note on: Adaboost.

Q. 13 Explain Random Forest model in detail. Can we consider this as a extension over bagging method?

Q. 14 What is class imbalance problem? Explain it with an example.

Q. 15 How can we improve classification accuracy of Class-imbalanced data?

Chapter Ends...

❑❑❑

MODULE 6

CHAPTER 6

Trends and Applications in Data Science

University Prescribed Syllabus w.e.f Academic Year 2022-2023

Data Science : applications and case studies, Data science for text, image, video, audio. Data science for Multimodal applications.

Self-learning Topics: Image Net Large Scale Visual Recognition Challenge (ILSVRC).

6.1 DATA SCIENCE

- We are surrounded by data and in abundance. What do we do with all of this data?
- How do we make it useful to us? What are its real-world applications? These questions are the domain of data science.
- **Data science** is the domain of study that deals with vast volumes of data using modern tools and techniques to find unseen patterns, derive meaningful information, and make business decisions.
- It involves techniques for identifying, collecting, and exploring the data using colorful plots and graphs.
- Data science uses complex machine learning algorithms to build predictive models. The data used for analysis can come from many different sources and presented in various formats.
- Data visualization, Data mining and Model building are some of the important phases in Data Science.
- To understand Data Science, Consider this example. Jayesh loves books to read, but every time when he wants to buy some book, he is always confused that which book he should buy as there are plenty of choices in front of him. Now here, Data Science techniques will be useful.
- When he opens Amazon he will get product recommendations on the basis of his past purchases.
- When he chooses one of them, he also gets a recommendation to buy these books with this one as this set is mostly bought.
- So, all recommendations of products and showing set of books purchased collectively is one of the examples of Data Science.
- A **data scientist** analyzes business data to extract meaningful insights. In other words, a data scientist solves business problems through a series of steps, including :
  (1) Before tackling the data collection and analysis, the data scientist determines the problem by asking the right questions and gaining understanding.
  (2) The data scientist then determines the correct set of variables and data sets.
  (3) The data scientist gathers structured and unstructured data from many disparate sources such as enterprise data, public data, etc.
  (4) Once the data is collected, the data scientist processes the raw data and converts it into a format suitable for analysis. This involves cleaning and validating the data to guarantee uniformity, completeness, and accuracy.
  (5) After the data has been rendered into a usable form, it's fed into the analytic system which can be a machine learning algorithm or a statistical model. This is where the data scientists analyze and identify patterns and trends.
  (6) When the data has been completely rendered, the data scientist interprets the data to find opportunities and solutions.
  (7) The data scientists finish the task by preparing the results and insights to share with the appropriate stakeholders and communicating the results.
- Data scientists also rely heavily on artificial intelligence, especially its subfields of machine learning and deep learning, to create models and make predictions using algorithms and other techniques.

6.2 APPLICATIONS OF DATA SCIENCE

The role of Data Science Applications hasn't evolved overnight. Thanks to faster computing and cheaper storage, we can now predict outcomes in minutes, which could take several human hours to process. Data science has found its applications in almost every industry.

1. Healthcare	2. Self-Driving Cars
3. Gaming	4. Image Recognition
5. Recommendation Systems	6. Logistics
7. Fraud Detection	8. Internet Search
9. Targeted Advertising	10. Speech Recognition
11. Airline Route Planning	12. Augmented Reality
13. Autocomplete	14. Finance
15. Cybersecurity	

1. Healthcare

- Data science has led to a number of breakthroughs in the healthcare industry.
- With a vast network of data now available via everything from EMRs to clinical databases to personal fitness trackers, medical professionals are finding new ways to understand disease, practice preventive medicine, diagnose diseases faster and explore new treatment options.
- Healthcare companies are using data science to build sophisticated medical instruments to detect and cure diseases.

2. Self-Driving Cars

- Tesla, Ford and Volkswagen are all implementing predictive analytics in their new wave of autonomous vehicles.
- These cars use thousands of tiny cameras and sensors to relay information in real-time.
- Using machine learning, predictive analytics and data science, self-driving cars can adjust to speed limits, avoid dangerous lane changes and even take passengers on the quickest route.

3. Gaming

Video and computer games are now being created with the help of data science and that has taken the gaming experience to the next level.

4. Image Recognition

Identifying patterns in images and detecting objects in an image is one of the most popular data science applications.

5. Recommendation Systems

Netflix and Amazon give movie and product recommendations based on what you like to watch, purchase, or browse on their platforms.

6. Logistics

- Data Science is used by logistics companies to optimize routes to ensure faster delivery of products and increase operational efficiency.
- Logistics UPS turns to data science to maximize efficiency, both internally and along its delivery routes.
- The company's On-road Integrated Optimization and Navigation (ORION) tool uses data science-backed statistical modeling and algorithms that create optimal routes for delivery drivers based on weather, traffic, construction, etc.

7. Fraud Detection

Banking and financial institutions use data science and related algorithms to detect fraudulent transactions.

8. Internet Search

- There are many other search engines like Yahoo, Bing, Ask, AOL, Google and so on.
- All these search engines make use of data science algorithms to deliver the best result for our searched query in a fraction of seconds.
- Considering the fact that, Google processes more than 20 peta bytes of data every day. Had there been no data science, Google wouldn't have been the 'Google' we know today.

9. Targeted Advertising

- Starting from the display banners on various websites to the digital billboards at the airports, almost all of them are decided by using data science algorithms.
- This is the reason why digital ads have been able to get a lot higher CTR (Call-Through Rate) than traditional advertisements. They can be targeted based on a user's past behavior.

10. Speech Recognition

- Some of the best examples of speech recognition products are Google Voice, Siri, Cortana etc.
- Using the speech-recognition feature, even if you aren't in a position to type a message, your life wouldn't stop.
- Simply speak out the message and it will be converted to text.

11. Airline Route Planning

- Airlines companies started using data science to identify the strategic areas of improvements.

- Now using data science, the airline companies can predict flight delay, decide which class of airplanes to buy, whether to directly land at the destination or take a halt in between, effectively drive customer loyalty programs.
- Southwest Airlines, Alaska Airlines are among the top companies who've embraced data science to bring changes in their way of working.

12. Augmented Reality

- Data Science and Virtual Reality do have a relationship, considering a VR headset contains computing knowledge, algorithms and data to provide you with the best viewing experience.
- A very small step towards this is the high-trending game of Pokemon GO.
- The ability to walk around things and look at Pokemon on walls, streets, things that aren't really there.

13. Autocomplete

- AutoComplete feature is an important part of Data Science where the user will get the facility to just type a few letters or words, and he will get the feature of auto-completing the line.
- In Google Mail, when we are writing formal mail to someone so at that time data science concept of Auto complete feature is used where he/she is an efficient choice to auto-complete the whole line.
- Also, in Search Engines in social media, in various apps, AutoComplete feature is widely used.

14. Finance

- Machine learning and data science have saved the financial industry millions of dollars, and unquantifiable amounts of time. For example, JP Morgan's Contract Intelligence (COiN) platform uses Natural Language Processing (NLP) to process and extract vital data from about 12,000 commercial credit agreements a year.
- Thanks to data science, what would take around 360,000 manual labor hours to complete is now finished in a few hours.

15. Cybersecurity

- Data science is useful in every industry, but it may be the most important in cybersecurity.
- International cybersecurity firm Kaspersky is using data science and machine learning to detect over 360,000 new samples of malware on a daily basis.
- Being able to instantaneously detect and learn new methods of cybercrime, through data science, is essential to our safety and security in the future.

6.3 DATA SCIENCE CASE STUDIES

- Data science is a highly evolving domain with many practical applications and a huge open community.
- Solving a Data Science case study means analyzing and solving a problem statement intensively.
- A case study in data science requires a systematic and organized approach for solving the problem.
- Generally, four main steps are needed to tackle every data science case study:
  (1) Defining the problem statement and strategy to solve it.
  (2) Gather and pre-process the data by making relevant assumptions.
  (3) Select tool and appropriate algorithms to build machine learning /deep learning models.
  (4) Make predictions, accept the solutions based on evaluation metrics, and improve the model if necessary.

Let's see some case studies in Data Science.

Case Study 1 : Data Science in Hotel Recommendation System

A hotel recommendation system typically works on collaborative filtering that makes recommendations based on ratings given by other customers in the same category as the user looking for a product.

Use Case :

- We all plan trips and the first thing to do when planning a trip is finding a hotel.
- There are so many websites recommending the best hotel for our trip.

- A hotel recommendation system aims to predict which hotel a user is most likely to choose from among all hotels.
- So, to build this type of system which will help the user to book the best hotel out of all the other hotels. We can do this using customer reviews.
- For example, suppose you want to go on a business trip, so the hotel recommendation system should show you the hotels which other customers have rated best for business travel.
- It is therefore also our approach to build a recommendation system based on customer reviews and ratings.
- So, use the ratings and reviews given by customers who belong to the same category as the user and build a hotel recommendation system.

Case Study 2 : Data Science in Entertainment Industry

- Due to the Pandemic, demand for OTT (Over-the-top) media platforms has grown significantly.
- People prefer watching movies and web series or listening to the music of their choice at leisure in the convenience of their homes.
- This sudden growth in demand has given rise to stiff competition.
- Every platform now uses data analytics in different capacities to provide better-personalized recommendations to its subscribers and improve user experience.

Use Case

- How Netflix uses data science to personalize the content and improve recommendations?
- Netflix is an extremely popular internet television platform with stream able content offered in several languages and caters to various audiences.
- In 2006, when Netflix entered this media streaming market, they were interested in increasing the efficiency of their existing "Cinematch" platform by 10% and hence, offered a prize of $1 million to the winning team.
- This approach was successful as they found a solution developed by the BellKor team at the end of the competition that increased prediction accuracy by 10.06%. Over 200 work hours and an ensemble of 107 algorithms provided this result.
- These winning algorithms are now a part of the Netflix recommendation system.
- Netflix also employs Ranking Algorithms to generate personalized recommendations of movies and TV Shows appealing to its users.

Use Case

- How Spotify uses big data to deliver a rich user experience for online music streaming?
- Personalized online music streaming is another area where data science is being used. Spotify is a well-known on-demand music service provider launched in 2008, which effectively leveraged big data to create personalized experiences for each user.
- It is a huge platform with more than 24 million subscribers and hosts a database of nearly 20million songs; they use the big data to offer a rich experience to its users.
- Spotify uses this big data and various algorithms to train machine learning models to provide personalized content.
- Spotify offers a "Discover Weekly" feature that generates a personalized playlist of fresh unheard songs matching the user's taste every week.
- Using the Spotify "Wrapped" feature, users get an overview of their most favorite or frequently listened songs during the entire year in December.
- Spotify also leverages the data to run targeted ads to grow its business. Thus, Spotify utilizes the user data, which is big data and some external data, to deliver a high-quality user experience.

Case Study 3 : Data Science in Banking and Finance

- Data science is extremely valuable in the Banking and Finance industry.
- Several high priority aspects of Banking and Finance like credit risk modeling (possibility of repayment of a loan), fraud detection (detection of malicious or irregularities in transactional patterns using machine learning), identifying customer lifetime value (prediction of bank performance based on existing and potential customers), customer segmentation (customer profiling based on behavior and characteristics for personalization of offers and services).

- Finally, data science is also used in real-time predictive analytics (computational techniques to predict future events).

Use Case

- How HDFC utilizes Big Data Analytics to increase revenues and enhance the banking experience?
- One of the major private banks in India, HDFC Bank, was an early adopter of AI.
- It started with Big Data analytics in 2004, intending to grow its revenue and understand its customers and markets better than its competitors.
- Back then, they were trendsetters by setting up an enterprise data warehouse in the bank to be able to track the differentiation to be given to customers based on their relationship value with HDFC Bank.
- Data science and analytics have been crucial in helping HDFC bank segregate its customers and offer customized personal or commercial banking services.
- The analytics engine and SaaS use have been assisting the HDFC bank in cross-selling relevant offers to its customers.
- Apart from the regular fraud prevention, it assists in keeping track of customer credit histories and has also been the reason for the speedy loan approvals offered by the bank.

Case Study 4 : Data Science for Customer Personality Analysis

The analysis of customers is one of the most important roles that a data scientist has to do who is working at a product-based company.

Use Case

- Customer Personality Analysis is a detailed analysis of a company's ideal customers.
- It helps a business to better understand its customers and makes it easier for them to modify products according to the specific needs, behaviors and concerns of different types of customers.
- We have to do an analysis that should help a business to modify its product based on its target customers from different types of customer segments.
- For example, instead of spending money to market a new product to every customer in the company's database, a company can analyze which customer segment is most likely to buy the product and then market the product only on that particular segment.

Case Study 5 : Data Science for Text Emotions Detection

The idea here is to train a machine learning model to generate emojis based on input text. Then this machine learning model can be used in training Artificial Intelligent Chatbots.

Use Case

- A human can express his emotions in any form, such as the face, gestures, speech and text.
- The detection of text emotions is a content-based classification problem.
- Detecting a person's emotions is a difficult task, but detecting the emotions using text written by a person is even more difficult as a human can express his emotions in any form.
- Recognizing this type of emotion from a text written by a person plays an important role in applications such as chatbots, customer support forum, customer reviews etc. So, you have to train a machine learning model that can identify the emotion of a text by presenting the most relevant emoji according to the input text.

Case Study 6 : Covid 19 and Data Science

- In the past two years of the Pandemic, the power of data science has been more evident than ever.
- Different pharmaceutical companies across the globe could synthesize Covid 19 vaccines by analyzing the data to understand the trends and patterns of the outbreak.
- Data science made it possible to track the virus in real-time, predict patterns, devise effective strategies to fight the Pandemic, and many more.

Use Case

- How Johnson and Johnson used data science to fight the Pandemic?
- The data science team at Johnson and Johnson leverages real-time data to track the spread of the virus.

- They built a global surveillance dashboard (granulated to county level) that helps them track the Pandemic's progress, predict potential hotspots of the virus, and narrow down the likely place where they should test its investigational COVID-19 vaccine candidate.
- The team works with in-country experts to determine whether official numbers are accurate and find the most valid information about case numbers, hospitalizations, mortality and testing rates, social compliance, and local policies to populate this dashboard.
- The team also studies the data to build models that help the company identify groups of individuals at risk of getting affected by the virus and explore effective treatments to improve patient outcomes.

Case Study 7 : Data Science in Ecommerce

In the e-commerce sector, big data analytics can assist in customer analysis, reduce operational costs, forecast trends for better sales, provide personalized shopping experiences to customers, and many more.

Use Case

- How does Amazon use data science to personalize shopping experiences and improve customer satisfaction?
- Amazon is a globally leading eCommerce platform that offers a wide range of online shopping services.
- Due to this, Amazon generates a massive amount of data that can be leveraged to understand consumer behavior and generate insights on competitors' strategies.
- Amazon uses its data to provide recommendations to its users on different products and services.
- With this approach, Amazon is able to persuade its consumers into buying and making additional sales.
- This approach works well for Amazon as it earns 35% of the revenue yearly with this technique. Additionally, Amazon collects consumer data for faster order tracking and better deliveries.
- Similarly, Amazon's virtual assistant, Alexa, can converse in different languages; uses speakers and a camera to interact with the users.
- Amazon utilizes the audio commands from users to improve Alexa and deliver a better user experience.

Case Study 8 : Data Science in Supply Chain Management

- Predictive analytics and big data are driving innovation in the Supply chain domain.
- They offer greater visibility into the company operations, reduce costs and overheads, forecasting demands, predictive maintenance, product pricing, minimize supply chain interruptions, route optimization, fleet management, drive better performance, and more.

Use Case

- Optimizing supply chain with big data analytics : UPS
- UPS is a renowned package delivery and supply chain management company.
- With thousands of packages being delivered every day, on average, a UPS driver makes about 100 deliveries each business day.
- On-time and safe package delivery are crucial to UPS's success. Hence, UPS offers an optimized navigation tool "ORION" (On-Road Integrated Optimization and Navigation), which uses highly advanced big data processing algorithms.
- This tool for UPS drivers provides route optimization concerning fuel, distance, and time. UPS utilizes supply chain data analysis in all aspects of its shipping process.
- Data about packages and deliveries are captured through radars and sensors. The deliveries and routes are optimized using big data systems.
- Overall, this approach has helped UPS save 1.6 million gallons of gasoline in transportation every year, significantly reducing delivery costs.

6.4 DATA SCIENCE FOR MULTIMODAL APPLICATIONS

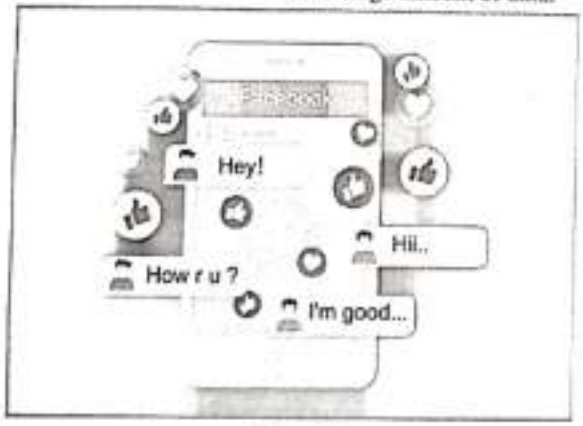
Multimodal Data

- Our experience of the world is multimodal: we see objects, hear sounds, feel the texture, smell odors, and taste flavors.

- Modality refers to the way in which something happens or is experienced and a research problem is characterized as multimodal when it includes multiple such modalities.
- In order for Artificial Intelligence to make progress in understanding the world around us, it needs to be able to interpret such multimodal signals together.
- Real-world applications usually involve data with various modalities, each containing valuable information.
- In order to enhance the performance of these applications, it is essential to adequately analyze all of the information extracted from the different data modalities.
- For example, images are usually associated with tags and text explanations; texts contain images to more clearly express the main idea of the article.
- Different modalities are characterized by very different statistical properties.

Facebook

- In today's era of digitalization, people spend hours on various social networking sites like Facebook, Instagram, WhatsApp, etc. Around 1.2 million people all over the globe upload 136,000 photos and update their status 293,000 times per minute.
- Our various activities whether it be commenting, tweeting, uploading something or anything else generates a large amount of data.
- In 2012, Facebook stated that it generates more than 500 terabytes of data every day. Data Science and many other Big Data technologies are helping Facebook to deal with such a large amount of data.



(IF1)Fig. 6.4.1

- Some of the examples that show how Facebook is using user data for developing smarter ideas and better products for bringing the world closer are:

(A) The Flashback

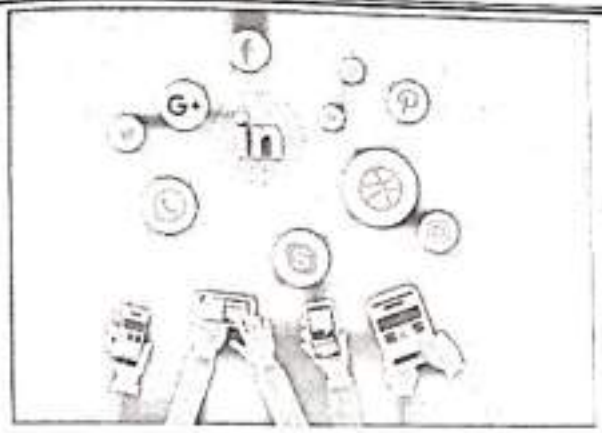
- You might have observed that on some special occasions like on your birthday, Facebook offers you to share a video that consists of some of the photos from your previous history.
- This is called "Flashback" which is a video collection of some of your previous uploads that received themost number of likes and comments.

(B) Celebrate Pride

- Facebook introduced a feature called "Celebrate Pride" for supporting the supreme court's judgment empowering same-sex marriage.
- This feature enables people to decorate their photos with the seven colors of the rainbow. The success of this step taken by Facebook was unbelievable.
- Millions of users changed their profile pictures in just a few hours.
- The Data Science team of Facebook analyzed the user data and observed that a large number of people were showing their support for the judgment. The success of "Celebrate Pride" was the result of this analysis.

LinkedIn

- LinkedIn is one of the most successful social media platforms that is connecting professionals across the globe.
- It also uses customer data for providing better services and customized user experience.
- LinkedIn stores a large amount of user data including several details like their contact information, previous history, interests, activities on different social networking sites, etc. in its data warehouse for being aware of the trends and patterns.



(IF2)Fig. 6.4.2

- Using the insights gained from the user data, LinkedIn connects individual users with their friends and people related to their areas of interest. It also helps them to make some decisions regarding the business.
- According to the different trends, LinkedIn provides various articles and other services that might match user interests. LinkedIn also enables users to promote their business to the right people by making use of targeting.
- Also, while using the customers' data, LinkedIn makes sure that the data is secure and no scrapping of data takes place from their site.

Uber

- Uber is one of the fastest-growing companies and has established its roots across 449 cities in 66 countries.
- It is completely ruling over the market with millions of users and around 1 billion Uber rides.
- The key behind this mesmerizing success story is the use of Big Data and Data Science for extracting insights and making some smart game-changing business decisions.

(IF3)Fig. 6.4.3

- Data Science has helped Uber to deal with issues like pricing policies, better cars, fake user accounts, fake rides, ranking and much more.
- Uber uses technologies like Hadoop and Spark for collecting data of each and every ride taken on Uber. Data Scientists at Uber use this data to understand the customers' point of view for solving their problems efficiently.
- The Data Science team at Uber performs a detailed analysis of data for various purposes like predicting the demand of rides, deciding the fares, identifying the cities with poor transportation services, etc.
- For maximizing their profit and the number of rides, Uber uses the idea of surge pricing.
- With the help of real-time analysis of data, they always provide rides at times when you are running late but they charge more than around two or three times the usual fares. This is done by the use of surge pricing algorithms.
- They are now moving towards the use of Machine Learning algorithms for implementing surge pricing to predict the areas of higher demand.
- This will help them to divert more and more drivers there. There are many other use cases that show the significance of Data Science at Uber.

Spotify

- Music plays an important role in the lives of people of almost all age groups.
- We frequently listen to our favorite songs in our daily routine such as while traveling, in leisure time, etc. to release our stress and relax. Today, there are many music playing applications in the market.
- You all might have heard the name "Spotify" at least once and most probably, you might have even used it.
- So, you must have observed that as soon as we start using it on a regular basis, it starts giving us personalized music recommendations and options to create customized playlists.
- This is what people like about it.

Fig. 6.4.4

- At the core of these personalized services lies a large amount of user data that Spotify, actually not only Spotify but most of the music playing applications are using.
- Spotify is using this data for optimizing their algorithms, improving user music experience, providing targeted ads, and making some good business strategies.
- In the present scenario, Spotify has around 108 million subscribers and around 124 million free users.
- The main goal of Spotify is to provide such a great experience to every user that will make them continue listening for hours.
- To achieve this, they are using many advanced Data Science and Machine Learning techniques to extract insights from the user data for matching with the music taste of their individual customer.
- Some of the features provided by Spotify that shows the use of Data Science are :

(A) Discover Weekly

- This is one of the most loved features of Spotify.
- It provides customized playlists to the users based on their previous activities by using a Machine Learning Algorithm.
- This algorithm examines the previous songs played by the user and creates new playlists similar to those songs.
- Along with providing customized playlist, Spotify also analyzes the users' reactions to individual songs.
- They observe whether the user played any song on repeat or changed it after a few seconds which helps them to develop an overall idea of the taste of different users.

(B) Daily Mixes

- Daily mixes are those playlists that Spotify generates by itself.
- These playlists include the songs that are either saved by the users or are of the artists followed by them or maybe something new that they may like.

6.5 IMAGENET LARGE SCALE VISUAL RECOGNITION CHALLENGE (ILSVRC)

- **The ImageNet Large Scale Visual Recognition Challenge (ILSVRC)** has been running annuallyfor five years (since 2010) and has become the standard benchmark for large-scale object recognition.
- ILSVRC follows in the footsteps of the PASCAL VOC challenge established in 2005,which set the precedent for standardized evaluation of recognition algorithms in the form of yearly competitions.
- As in PASCAL VOC, ILSVRC consists of two components :
  (1) A publicly available dataset, and
  (2) An annual competition and corresponding workshop. The dataset allows for the development and comparison of categorical object recognition algorithms, and the competition and workshop provide a way to track the progress and discuss the lessons learned from the most successful and innovative entries each year.
- The publicly released dataset contains a set of manually annotated training images.
- A set of test images is also released, with the manual annotations withheld.
- Participants train their algorithms using the training images and then automatically annotate the test images.
- These predicted annotations are submitted to the evaluation server.

- Results of the evaluation are revealed at the end of the competition period and authors are invited to share insights at the workshop held at the International Conference on Computer Vision (ICCV)or European Conference on Computer Vision (ECCV)in alternate years.
- ILSVRC annotations fall into one of two categories :
  (1) Image-level annotation of a binary label for the presence or absence of an object class in the image, e.g.," there are cars in this image" but "there are no tigers," and
  (2) Object-level annotation of a tight bounding box and class label around an object instance in the image, e.g., "there is a screwdriver centered at position (20,25)with width of 50 pixels and height of 30 pixels".
- The Image Net dataset is the backbone of ILSVRC. Image Net is an image dataset organized according to the WordNet hierarchy.
- Each concept in WordNet, possibly described by multiple words or word phrases, is called a "synonym set" or "synset".
- ImageNet populates 21,841 synsets of WordNet with an average of 650 manually verified and full resolution images.
- As a result, ImageNet contains14,197,122 annotated images organized by the semantichierarchy of WordNet (as of August 2014).
- ImageNet is larger in scale and diversity than the other image classification datasets.
- ILSVRC uses a subset of ImageNet images for training the algorithms and some of ImageNet's image collection protocols for annotating additional images for testing the algorithms.

6.6 MULTIPLE CHOICE QUESTIONS

Q. 6.1 Which of the following is performed by Data Scientist?
(a) Define the question
(b) Create reproducible code
(c) Challenge results
(d) All of the mentioned
✓**Ans. : (d)**

Q. 6.2 Data science is the process of diverse set of data through_____.
(a) Organizing data
(b) Processing data
(c) Analysing data
(d) All of the above
✓**Ans. : (d)**

Q. 6.3 The applications of Data Science are _____.
(a) Healthcare
(b) Fraud and Risk Detection
(c) Airline Route Planning
(d) All of the above
✓**Ans. : (d)**

Q. 6.4 The data science applications in healthcare are _____.
(a) Data Science for Medical Imaging
(b) Data Science for Genomics
(c) Drug Discovery with Data Science
(d) All of the above
✓**Ans. : (d)**

Q. 6.5 Which of the following is correct skills for a Data Scientist?
(a) Probability and Statistics
(b) Machine Learning / Deep Learning
(c) Data Wrangling
(d) All of the above
✓**Ans. : (d)**

Descriptive Questions

Q. 1 What is Data Science?

Q. 2 What are different applications of Data Science?

Q. 3 Explain applications of Data Science in Healthcare.

Q. 4 Explain applications of Data Science in Supply Chain Management.

Q. 5 Explain applications of Data Science in Banking and Finance.

Q. 6 Explain applications of Data Science in Entertainment industry.

Q. 7 What are roles of Data Scientist?

Q. 8 Using a case study, explain use of data science for multimodal applications.

Chapter Ends...

□□□