Chapter Four

Knowledge Representation and Reasoning

AI agents that can form representations of a complex world, use a process of inference to derive new representations about the world, and use these new representations to deduce what to do.

Human beings are very good at acquiring new information by combining raw knowledge, experience with reasoning. And also, they are good at understanding, reasoning and interpreting knowledge. And using this knowledge, they are able to perform various actions in the real world. But how do machines perform the same?

Logic and Inference

In artificial intelligence, logic and inference are essential for knowledge representation and reasoning systems. Propositional logic, the simplest form of logic, deals with true or false statements, expressed using atomic and compound propositions. Logical connectives include negation, conjunction, disjunction, implication, and biconditional. Truth tables represent the possible truth values of propositions. The section also covers the precedence of connectives, logical equivalence, properties of operators, and the limitations of propositional logic.

Logic and inference are fundamental aspects of reasoning and decision-making processes. In various domains, including mathematics, philosophy, computer science, and artificial intelligence, logic provides a systematic framework for analyzing and evaluating arguments and statements.

At its core, logic deals with the principles of valid reasoning. It aims to establish rules and methods that allow us to draw sound conclusions based on given information or premises. Inference, on the other hand, refers to the process of deriving new information or conclusions from existing knowledge.

One of the primary tools of logic is formal language, which provides a precise and unambiguous means of expressing ideas and propositions. Through the use of symbols, variables, and logical operators such as conjunction, disjunction, implication, and negation, formal languages allow us to construct complex statements and reason about their truth or falsity.

In logic, we often employ deductive reasoning, which involves deriving specific conclusions from general principles or premises. Deductive reasoning follows a strict set of rules, such as modus ponens and modus tollens, to ensure the validity of the resulting conclusions. This processis crucial in mathematical proofs, where each step must be logically sound to establish the truthof a statement.

In addition to deductive reasoning, logic encompasses inductive reasoning, which involvesmaking generalizations or predictions based on observed patterns or evidence. Inductive reasoning allows us to draw probable conclusions that are likely to be true but not guaranteed. It plays a significant role in scientific reasoning and empirical investigations, where hypotheses aretested and revised based on observed data.

Furthermore, logic provides tools for analyzing the relationships between statements and determining their logical connections. This includes techniques such as truth tables, logical equivalences, and formal proof systems like natural deduction or axiomatic systems. These tools enable us to assess the validity of arguments, identify fallacies, and construct rigorous proofs.

In the field of artificial intelligence, logic and inference play a vital role in knowledge representation and reasoning systems. By encoding knowledge and rules in a logical formalism, AI systems can perform complex inference tasks, such as logical deduction, reasoning with uncertainty, and solving constraint satisfaction problems.

Overall, logic and inference are powerful tools for structuring and evaluating reasoning processes. They provide a systematic approach to analyze arguments, make inferences, and draw conclusions based on well-defined rules and principles. Whether in mathematics, philosophy, computer science, or artificial intelligence, logic continues to be a fundamental discipline that underlies rational thinking and decision-making.

Example:

Suppose we have the following premises:

Premise 1: All mammals are animals.

Premise 2: Elephants are mammals.

From these premises, we can use logical inference to draw a conclusion:

Conclusion: Therefore, elephants are animals.

In this example, we are using deductive reasoning and the logical principle of categorical syllogism. By applying the general statement "All mammals are animals" to the specific statement "Elephants are mammals," we can infer those elephants, being mammals, are also animals.

This example illustrates how logic and inference allow us to derive new information or conclusions based on existing knowledge and logical rules.

Propositional Logic

Propositional logic is the simplest form of logic where all the statements are made by propositions. A proposition is a declarative statement which is either true or false. It is a technique of knowledge representation in logical and mathematical form.

Example:

a) It is Sunday.
b) The Sun rises from West (False proposition)
c) 3+3= 7(False proposition)
d) 5 is a prime number.

Propositional logic is also called Boolean logic as it works on 0 and 1. In propositional logic, we use symbolic variables to represent the logic, and we can use any symbol for a representing a proposition, such A, B, C, P, Q, R, etc. Propositions can be either true or false, but it cannot be both. Propositional logic consists of an object, relations or function, and logical connectives. These connectives are also called logical operators. The propositions and connectives are the basic elements of the propositional logic. Connectives can be said as a logical operator which connects two sentences. A proposition formula which is always true is called ***tautology***, and it is also called a valid sentence. A proposition formula which is always false is called ***Contradiction***. A proposition formula which has both true and false values is called **Contingency**. Statements which are questions, commands, or opinions are not propositions such as "Where is Abebe", "How are you", "What is your name", are not propositions.

The syntax of propositional logic defines the allowable sentences for the knowledge representation. There are two types of Propositions:

- Atomic Propositions
- Compound propositions

Atomic propositions are the simple propositions. It consists of a single proposition symbol. These are the sentences which must be either true or false. **Example**: – a) 2+2 is 4, it is an atomic proposition as it is a true fact. – b) "The Sun is cold" is also a proposition as it is a false fact.

Compound proposition: Compound propositions are constructed by combining simpler or atomic propositions, using parenthesis and logical connectives. Example: – a) "It is raining today, and street is wet." – b) "Abebe is a doctor, and his clinic is in Adama."

Logical Connectives

Logical connectives are used to connect two simpler propositions or representing a sentence logically. We can create compound propositions with the help of logical connectives. There are mainly five connectives, which are given as follows:

- Negation: A sentence such as ¬ P is called negation of P. A literal can be either Positive literal or negative literal.

- Conjunction: A sentence which has ∧ connective such as, P ∧ Q is called a conjunction. Example Abebe is a student and hardworking. It can be written as, P= Abebe is a student, Q= Abebe is hardworking. it can be represented as P ∧ Q
- Disjunction: A sentence which has ∨ connective, such as P ∨ Q is called disjunction, where P and Q are the propositions. Example: "Abebe is a doctor or Engineer", Here P= Abebe is Doctor. Q= Abebe is Engineer, so we can write it as P ∨ Q.
- Implication: A sentence such as P→Q, is called an implication. Implications are also known as if-then rules. It can be represented as If it is raining, then the street is wet. Let P= It is raining, and Q= Street is wet, so it is represented as P → Q
- Biconditional: A sentence such as P⇔ Q is a Biconditional sentence. Example If I am breathing, then I am alive. P= I am breathing, Q= I am alive, it can be represented as P ⇔ Q.

Connective Symbol	**Word**	**Technical term**	**Example**
¬ or ~	Not	Negation	¬ A or ~A
∧	AND	Conjunction	A ∧ B
∨	OR	Disjunction	A ∨ B
→	Implies	Implication	A → B
⇔	If and only if	Biconditional	A ⇔ B

Truth Table

In propositional logic, we need to know the truth values of propositions in all possible scenarios. We can combine all the possible combination with logical connectives, and the representation of these combinations in a tabular format is called Truth table. Following are the truth table for all logical connectives:

Negation

P	**¬ P**
True	False
False	True

Conjunction

P	**Q**	**P ∧ Q**
True	True	True
True	False	False
False	True	False
False	False	False

Disjunction

P	**Q**	**P ∨ Q**
True	True	True
True	False	True
False	True	True
False	False	False

Implication

P	**Q**	**P → Q**
True	True	True
True	False	False
False	True	True
False	False	True

Biconditional

P	**Q**	**P → Q**
True	True	True
True	False	False
False	True	False
False	False	True

We can build a proposition composing three propositions P, Q, and R. This truth table is made-up of 8th tuples as we have taken three proposition symbols. P V Q **→ ¬R**

P	Q	R	¬R	Pv Q	PvQ→¬R
True	True	True	False	True	False
True	True	False	True	True	True
True	False	True	False	True	False
True	False	False	True	True	True
False	True	True	False	True	False
False	True	False	True	True	True
False	False	True	False	False	True
False	False	False	True	False	True

Precedence of connectives

Just like arithmetic operators, there is a precedence order for propositional connectors or logical operators. This order should be followed while evaluating a propositional problem. Following is the list of the precedence order for operators:

Precedence	Operators
First Precedence	Parenthesis
Second Precedence	Negation
Third Precedence	Conjunction(AND)
Fourth Precedence	Disjunction(OR)
Fifth Precedence	Implication
Six Precedence	Biconditional

Logical equivalence

Logical equivalence is one of the features of propositional logic. Two propositions are said to be logically equivalent if and only if the columns in the truth table are identical to each other. Let's take two propositions A and B, so for logical equivalence, we can write it as A⇔ B. In below truth table we can see that column for ¬A ∨ B and A →B, are identical hence A is Equivalent to B.

A	B	¬A	¬AV B	A→B
T	T	F	**T**	**T**
T	F	F	**F**	**F**
F	T	T	**T**	**T**
F	F	T	**T**	**T**

Properties of Operators

Commutativity: $P \wedge Q = Q \wedge P$, or $P \vee Q = Q \vee P$

Associativity: $(P \wedge Q) \wedge R = P \wedge (Q \wedge R)$, $(P \vee Q) \vee R = P \vee (Q \vee R)$

Identity element: $P \wedge True = P$, $P \vee True = True$.

Distributive: $P \wedge (Q \vee R) = (P \wedge Q) \vee (P \wedge R) P \vee (Q \wedge R) = (P \vee Q) \wedge (P \vee R)$

DE Morgan's Law: $\neg (P \wedge Q) = (\neg P) \vee (\neg Q)$ $\neg (P \vee Q) = (\neg P) \wedge (\neg Q)$ Double-negation elimination: $\neg (\neg P) = P$

Limitations of Propositional logic

- We cannot represent relations like ALL, some, or none with propositional logic. Example: – All the girls are intelligent. – Some apples are sweet.
- Propositional logic has limited expressive power.
- In propositional logic, we cannot describe statements in terms of their properties or logical relationships.

First Order Logic

. First-order logic allows for the representation of complex sentences and natural language statements that propositional logic cannot handle. It introduces the concepts of objects, relations, functions, and connectives as the basic elements of first-order logic. The syntax of first-order logic includes constants, variables, predicates, functions, connectives, equality, and quantifiers. The content explains the use of quantifiers, including universal quantifiers (for all) and existential quantifiers (for some), to express statements about the quantity of objects in the universe of discourse. It also discusses the distinction between free variables and bound variables in first-order logic formulas. Overall, the content highlights the expressive power of first-order logic in representing complex statements and the role it plays in knowledge representation in artificial intelligence.

First-order logic is another way of knowledge representation in artificial intelligence. It is an extension to propositional logic. It is sufficiently expressive to represent the natural language statements in a concise way. First-order logic is also known as Predicate logic or First-order predicate logic. First-order logic is a powerful language that develops information about the objects in an easier way and can also express the relationship between those objects.

First-order logic (like natural language) does not only assume that the world contains facts like propositional logic but also assumes the following things in the world:

- Objects: A, B, people, numbers, colors, wars, theories, squares, pits, wumpus, ...
- Relations: It can be unary relation such as: red, round, is adjacent, or n-any relation such as: the sister of, brother of, has color, comes between
- Function: Father of, best friend, third inning of, end of, ...

As a natural language, first-order logic also has two main parts: Syntax and Semantics

Following are the basic elements of First-order logic syntax:

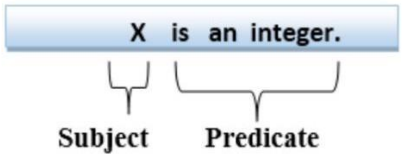
- Constant 1, 2, A, John, Mumbai, cat,....
- Variables x, y, z, a, b,....
- Predicates Brother, Father, >,....
- Function sqrt, LeftLegOf, ...
- Connectives $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$, $\neg$
- Equality ==
- Quantifier $\forall$, $\exists$

Atomic sentences are the most basic sentences of first-order logic. These sentences are formed from a predicate symbol followed by a parenthesis with a sequence of terms. We can represent atomic sentences as Predicate (term1, term2, ..., term n). Example: Abebe and Kebede are brothers: => Brothers (Abebe, Kebede). Dimet is a cat: => cat (Dimet).

Complex Sentences: Complex sentences are made by combining atomic sentences using connectives. First-order logic statements can be divided into two parts:

- Subject: is the main part of the statement.
- Predicate: A predicate can be defined as a relation, which binds two atoms together in a statement.

Consider the statement: "x is an integer.", it consists of two parts, the first part x is the subject of the statement and second part "is an integer," is known as a predicate.



Quantifiers in First Order Logic

A quantifier is a language element which generates quantification, and quantification specifies the quantity of specimen in the universe of discourse. These are the symbols that permit to determine or identify the range and scope of the variable in the logical expression.

There are two types of quantifiers:

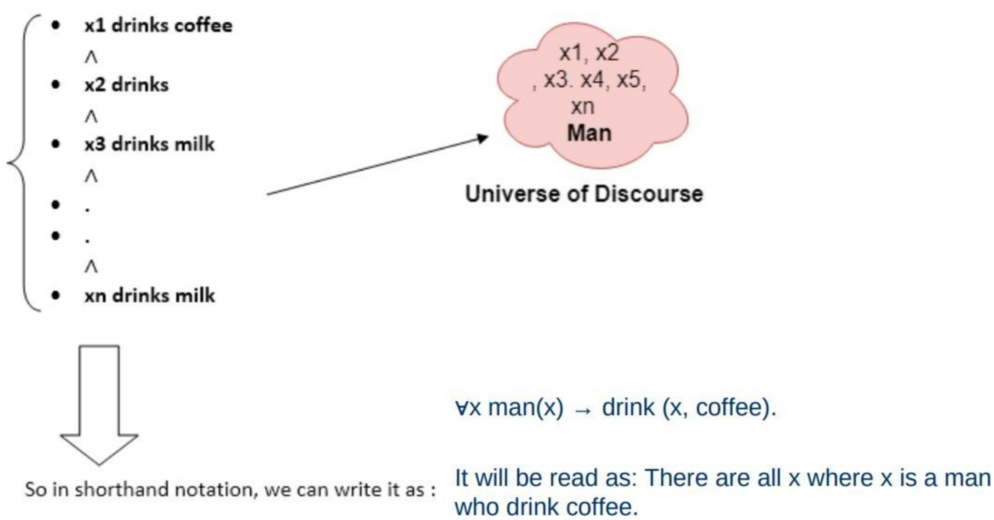
- Universal Quantifier, (for all, everyone, everything)
- Existential quantifier, (for some, at least one).

Universal quantifier: is a symbol of logical representation, which specifies that the statement within its range is true for everything or every instance of a particular thing. The Universal quantifier is represented by a symbol, ∀ which resembles an inverted A.

Note: In universal quantifier we use implication "→ ".

If x is a variable, then ∀x is read as: For all x or For each x or For every x.

Example: All man drink coffee. Let a variable x which refers to a cat so all x can be represented in Universe of Discourse (UOD) as below:



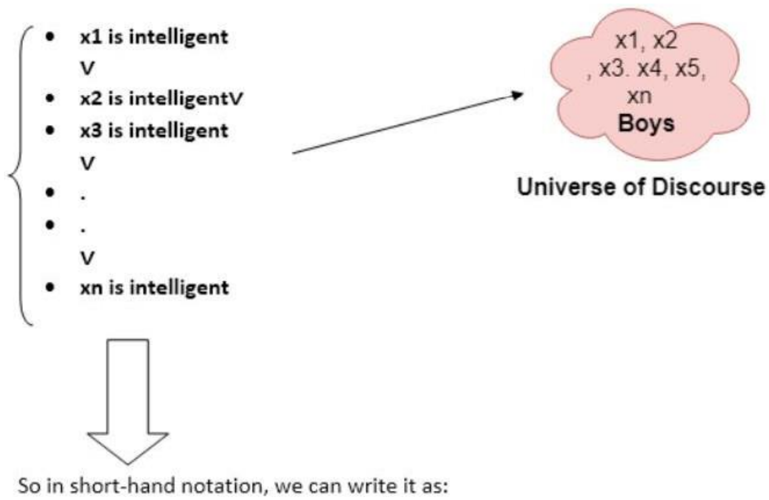
Existential Quantifier

Existential quantifiers are the type of quantifiers, which express that the statement within its scope is true for at least one instance of something. It is denoted by the logical operator ∃, which resembles as inverted E. When it is used with a predicate variable then it is called as an existential quantifier.

Note: In Existential quantifier we always use AND or Conjunction symbol (Ʌ).

– If x is a variable, then existential quantifier will be ∃x or ∃(x). And it will be read as: There exists a 'x.' or For some 'x.' or For at least one 'x.'

Example: Some boys are intelligent. ∃x: boys(x) Ʌ intelligent(x). It will be read as: There are some x where x is a boy who is intelligent.



Points to remember: The main connective for universal quantifier ∀ is implication →. The main connective for existential quantifier ∃ is and Ʌ.

Properties of Quantifiers:

– In universal quantifier, ∀x ∀y is similar to ∀y ∀x.
– In Existential quantifier, ∃x∃y is similar to ∃y∃ x.
– ∃x ∀y is not similar to ∀y ∃x.

Examples

1) All birds fly. In this question the predicate is "fly(bird)." And since there are all birds who fly so it will be represented as follows. ∀x bird(x) →fly(x).

2) Every man respects his parent. In this question, the predicate is "respect(x, y)," where x=man, and y= parent. Since there is every man so will use ∀, and it will be represented as follows: ∀x man(x) → respects (x, parent).
3) Some boys play football. In this question, the predicate is "play(x, y)," where x= boys, and y= game. Since there are some boys so we will use ∃, and it will be represented as: ∃x boys(x) →play(x, cricket).
4) Not all students like both Mathematics and Science. In this question, the predicate is "like(x, y)," where x= student, and y= subject. Since there are not all students, so we will use ∀ with negation, so representation for this: ¬ (x) [student(x) → like(x, Mathematics) Ʌ like(x, Science)].
5) Only one student failed in Mathematics. In this question, the predicate is "failed(x, y)," where x= student, and y= subject. Since there is only one student who failed in Mathematics, so we will use following representation for this: ∃(x) [student(x) →failed (x, Mathematics) (y) Ʌ ∀ [¬(x==y) Ʌ student(y) → ¬failed (x, Mathematics)].

Free and Bound Variables

The quantifiers interact with variables which appear in a suitable way. There are two types of variables in First order logic which are given below:

Free Variable: A variable is said to be a free variable in a formula if it occurs outside the scope of the quantifier. Example: ∀x (y) ∃ [P (x, y, z)], where z is a free variable.

Bound Variable: A variable is said to be a bound variable in a formula if it occurs within the scope of the quantifier. Example: ∀x [A (x) B(y)], here x and y are the bound variables.

Knowledge Representation

Knowledge Representation in AI is the study of expressing the beliefs, intentions, andjudgments of an intelligent agent in a suitable form for automated reasoning. It involves modeling intelligent behavior and solving complex problems using knowledge.

The different types of knowledge that need to be represented in AI include declarative knowledge, structural knowledge, procedural knowledge, meta-knowledge, and heuristic knowledge. Declarative knowledge includes concepts, facts, and objects, while structural knowledge describes the relationships between concepts and objects. Procedural knowledge focuses on knowing how to do something, and meta-knowledge pertains to knowledge about other types of knowledge. Heuristic knowledge represents expert knowledge in a specific field or subject.

The cycle of knowledge representation in AI consists of various components, including perception, learning, knowledge representation and reasoning, planning, and execution. The perception component retrieves information from the environment, while the learning component focuses on self-improvement and acquiring new knowledge. Knowledge representation and reasoning are essential components that enable machines to display human-like intelligence by understanding and utilizing knowledge. Planning and execution depend on the analysis of knowledge representation and reasoning, where planning involves determining a sequence of actions to achieve a specific goal.

Overall, knowledge representation in AI is vital for enabling machines to understand, learn from, and behave intelligently using different types of knowledge.

Knowledge Representation in AI describes the representation of knowledge. Basically, it is a study of how the beliefs, intentions, and judgments of an intelligent agent can be expressed suitably for automated reasoning. One of the primary purposes of Knowledge Representation includes modeling intelligent behavior for an agent.

Knowledge Representation and Reasoning (KR, KRR) represents information from the real world for a computer to understand and then utilize this knowledge to solve complex real-life problems like communicating with human beings in natural language. Knowledge representation in AI is not just about storing data in a database, it allows a machine to learn from that knowledge and behave intelligently like a human being.

The different kinds of knowledge that need to be represented in AI include: Objects, Events, Performance, Facts, Meta-Knowledge, and Knowledge-base.

Types of Knowledge

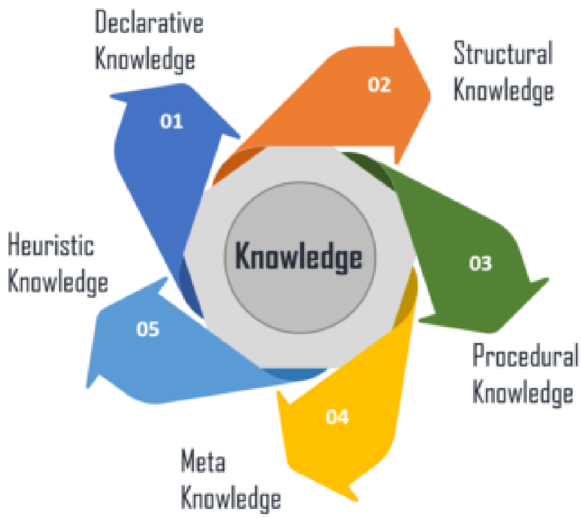
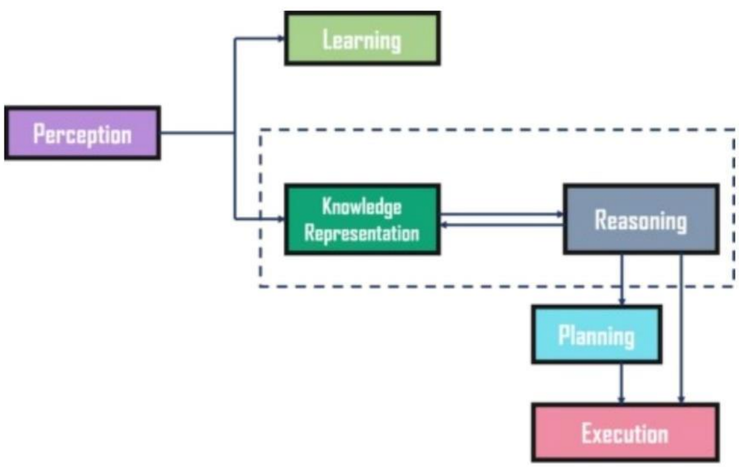


Figure 19: Types of Knowledge

- *Declarative Knowledge* – It includes concepts, facts, and objects and expressed in a declarative sentence.
- *Structural Knowledge* – It is a basic problem-solving knowledge that describes the relationship between concepts and objects.
- *Procedural Knowledge* – This is responsible for knowing how to do something and includes rules, strategies, procedures, etc.
- *Meta Knowledge* – Meta Knowledge defines knowledge about other types of Knowledge.
- *Heuristic Knowledge* – This represents some expert knowledge in the field or subject.

Cycle of Knowledge Representation in AI

Artificial Intelligent Systems usually consist of various components to display their intelligent behavior. Some of these components include: Perception, Learning, Knowledge Representation & Reasoning, Planning, and Execution.



The Perception component retrieves data or information from the environment. With the help of this component, you can retrieve data from the environment, find out the source of noises and check if the AI was damaged by anything.

Also, it defines how to respond when any sense has been detected. Then, there is the Learning Component that learns from the captured data by the perception component. The goal is to build computers that can be taught instead of programming them. Learning focuses on the process of self-improvement. In order to learn new things, the system requires knowledge acquisition, inference, acquisition of heuristics, faster searches, etc.

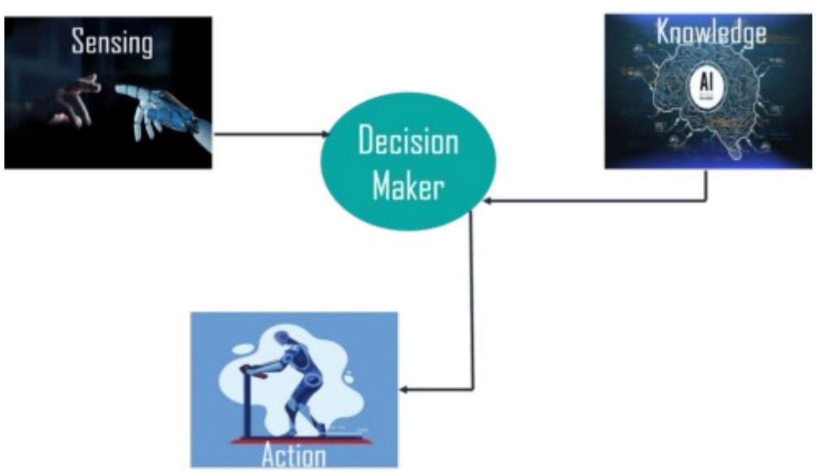
The main component in the cycle is Knowledge Representation and Reasoning which shows the humanlike intelligence in the machines. Knowledge representation is all about understanding intelligence. Instead of trying to understand or build brains from the bottom up, its goal is to understand and build intelligent behavior from the top-down and focus on what an agent needs to know in order to behave intelligently. Also, it defines how automated reasoning procedures can make this knowledge available as needed.

The Planning and Execution components depend on the analysis of knowledge representation and reasoning. Here, planning includes giving an initial state, finding their preconditions and effects, and a sequence of actions to achieve a state in which a particular goal holds. Now once the planning is completed, the final stage is the execution of the entire process.

Relation between Knowledge & Intelligence

In the real world, knowledge plays a vital role in intelligence as well as creating artificial intelligence. It demonstrates the intelligent behavior in AI agents or systems. It is possible for an agent or system to act accurately on some input only when it has the knowledge or experience about the input.

Example



In this example, there is one decision-maker whose actions are justified by sensing the environment and using knowledge. But, if we remove the knowledge part here, it will not be able to display any intelligent behavior. Now that you know the relationship between knowledge and intelligence, let‘s move on to the techniques of Knowledge Representation in AI.

Knowledge Representation in AI

In logical representation, a language with well-defined rules is used to represent propositions. It focuses on precise syntax and semantics, allowing for sound inference and logical reasoning. Logical representation is advantageous for performing logical reasoning and serves as the basis for programming languages. However, it has limitations and can be challenging to work with, and its naturalness and inference efficiency may be compromised.

On the other hand, semantic network representation provides an alternative to predicate logic. Knowledge is represented in the form of graphical networks consisting of nodes and arcs that describe relationships between objects. Semantic networks are a natural way of representing knowledge and convey meaning transparently. They are easy to understand and extend. However, they can be computationally expensive at runtime, lack equivalent quantifiers, and rely on the creator's knowledge rather than exhibiting inherent intelligence.

Fame representation and production rules as additional techniques for knowledge representation. Frame representation involves structuring

knowledge into records or frames containing attributes and values to describe entities. It simplifies programming by grouping related data but may have limitations in inference. Production rules use condition-action pairs to guide problem-solving steps. While expressed in natural language and highly modular, they lack learning capabilities and may lead to inefficiencies when many rules are active.

Additionally, the content mentions the requirements of a good knowledge representation system, such as representational accuracy, inferential adequacy, inferential efficiency, and acquisitional efficiency. It also discusses different approaches to knowledge representation, including simple relational knowledge, inheritable knowledge, and inferential knowledge.

The section concludes with a discussion on the challenges and considerations in knowledge representation, such as important attributes, relationships among attributes, existence in an isa hierarchy, reasoning about attribute values, single-valued attributes, granularity of representation, representation of sets of objects, finding the right structure for a situation, and accessing relevant parts of a knowledge database.

Logical representation: is a language with some definite rules which deal with propositions and has no ambiguity in representation. It represents a conclusion based on various conditions and lays down some important communication rules. Also, it consists of precisely defined syntax and semantics which supports the sound inference. Each sentence can be translated into logics using syntax and semantics.

Syntax	Semantics
• It decides how we can construct legal sentences in logic. • It determines which symbol we can use in knowledge representation. • Also, how to write those symbols	• Semantics are the rule by which we can interpret the sentence in the logic • It assigns a meaning to each sentence.

Advantages:

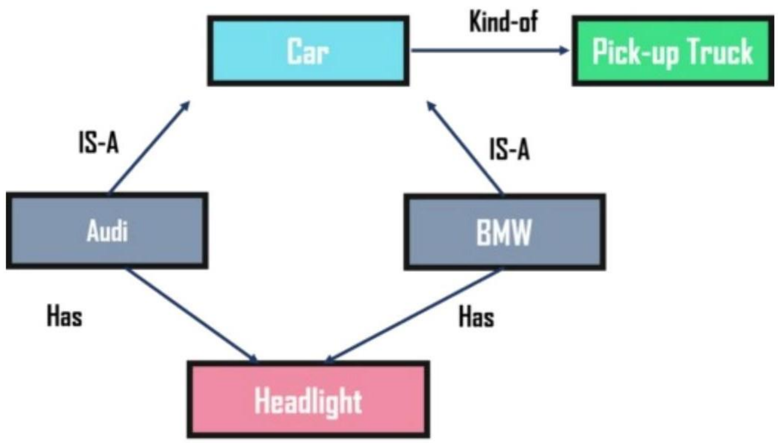
- Logical representation helps to perform logical reasoning.
- This representation is the basis for the programming languages.

Disadvantages:

- Logical representations have some restrictions and are challenging to work with.
- This technique may not be very natural, and inference may not be very efficient.

Semantic Network Representation

Semantic networks work as an alternative of predicate logic for knowledge representation. In Semantic networks, you can represent your knowledge in the form of graphical networks. This network consists of nodes representing objects and arcs which describe the relationship between those objects. Also, it categorizes the object in different forms and links those objects. This representation consists of two types of relations: IS-A relation (Inheritance) and Kind-of-relation



Advantages:

- Semantic networks are a natural representation of knowledge.
- Also, it conveys meaning in a transparent manner.
- These networks are simple and easy to understand.

Disadvantages:

- Semantic networks take more computational time at runtime.
- Also, these are inadequate as they do not have any equivalent quantifiers.
- These networks are not intelligent and depend on the creator of the system.

Frame Representation

A frame is a record like structure that consists of a collection of attributes and values to describe an entity in the world. These are the AI data structure that divides knowledge into substructures by representing stereotypes situations. Basically, it consists of a collection of slots and slot values of any type and size. Slots have names and values which are called facets.

Advantages:

- It makes the programming easier by grouping the related data.
- Frame representation is easy to understand and visualize.
- It is very easy to add slots for new attributes and relations.
- Also, it is easy to include default data and search for missing values.

Disadvantages:

- In frame system inference, the mechanism cannot be easily processed.

- The inference mechanism cannot be smoothly proceeded by frame representation.
- It has a very generalized approach.

Production Rules

In production rules, agent checks for the condition and if the condition exists then production rule fires and corresponding action is carried out. The condition part of the rule determines which rule may be applied to a problem. Whereas, the action part carries out the associated problem-solving steps. This complete process is called a recognize-act cycle. The production rules system consists of three main parts: The set of production rules, Working Memory, and The recognize-act-cycle.

Advantages:

- The production rules are expressed in natural language.
- The production rules are highly modular and can be easily removed or modified.

Disadvantages:

- It does not exhibit any learning capabilities and does not store the result of the problem for future uses.
- During the execution of the program, many rules may be active. Thus, rule-based production systems are inefficient.

Representation Requirements

A good knowledge representation system must have properties such as:

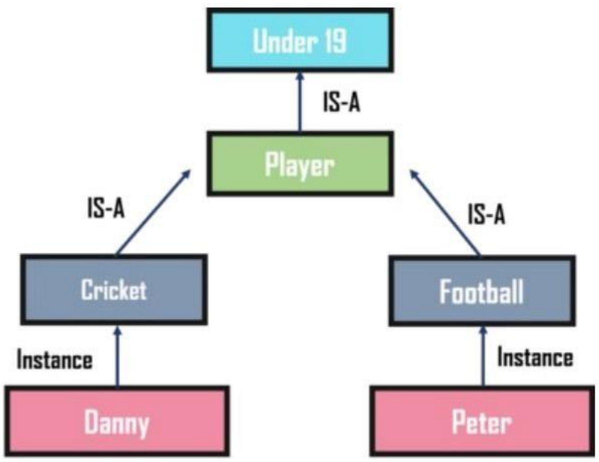
- Representational Accuracy: It should represent all kinds of required knowledge.
- Inferential Adequacy: It should be able to manipulate the representational structures to produce new knowledge corresponding to the existing structure.
- Inferential Efficiency: The ability to direct the inferential knowledge mechanism into the most productive directions by storing appropriate guides.
- Acquisitional efficiency: The ability to acquire new knowledge easily using automatic methods.

Approaches to Knowledge Representation in AI

Simple Relational Knowledge: It is the simplest way of storing facts which uses the relational method. Here, all the facts about a set of the object are set out systematically in columns. Also, this approach of knowledge representation is famous in database systems where the relationship between different entities is represented. Thus, there is little opportunity for inference.

Name	Age	Emp ID
John	25	100071
Amanda	23	100056
Sam	27	100042

Inheritable Knowledge: In the inheritable knowledge approach, all data must be stored into a hierarchy of classes and should be arranged in a generalized form or a hierarchal manner. Also, this approach contains inheritable knowledge which shows a relation between instance and class, and it is called instance relation. In this approach, objects and values are represented in Boxed nodes.



Inferential Knowledge: this approach represents knowledge in the form of formal logic. Thus, it can be used to derive more facts. Also, it guarantees correctness. Example: Statement 1: Abebe is a footballer. Statement 2: All footballers are athletes. Then it can be represented as;

- footballer (Abebe)
- ∀x = footballer(x) → Athlete (x)s

Issues in Knowledge Representation

The fundamental goal of knowledge Representation is to facilitate inference (conclusions) from knowledge. The issues that arise while using Knowledge Representation techniques are many. Some of these are explained below.

Important Attributed:

– Any attribute of objects so basic that they occur in almost every problem domain?

- There are two attributed –instance‖ and –isa‖, that are general significance. These attributes are important because they support property inheritance.

Relationship among attributes:

- Any important relationship that exists among object attributed?
- The attributes we use to describe objects are themselves entities that we represent.
- The relationship between the attributes of an object, independent of specific knowledge they encode, may hold properties like: Inverse. This is about consistency check, while a value is added to one attribute. The entities are related to each other in many different ways.

Existence in an isa hierarchy:

- This is about generalization-specification, like, classes of objects and specialized subsets of those classes, there are attributes and specialization of attributes.
- For example, the attribute height is a specialization of general attribute physical-size which is, in turn, a specialization of physical-attribute.
- These generalization-specialization relationships are important for attributes because they support inheritance.

Technique for reasoning about values

- This is about reasoning values of attributes not given explicitly.
- Several kinds of information are used in reasoning, like, height: must be in a unit of length, Age: of a person cannot be greater than the age of person‘s parents.
- The values are often specified when a knowledge base is created.

Single valued attributes

- This is about a specific attribute that is guaranteed to take a unique value.
- For example, a baseball player can at time have only a single height and be a member of only one team.
- Knowledge Representation systems take different approaches to provide support for single valued attributes.

Choosing Granularity:

- At what level of detail should the knowledge be represented?
- Regardless of the Knowledge Representation formalism, it is necessary to know: At what level should the knowledge be represented and what are the primitives? Should there be a small number or should there be a large number of low-level primitives or High-level facts.

High-level facts may not be adequate for inference while Low-level primitives may require a lot of storage.

Example of Granularity: Suppose we are interested in following facts: Abebe spotted Kebede. This could be represented as Spotted (agent (Abebe), object (Kebede)). Such a representation would make it easy to answer questions such are: Who spotted Kebede? Suppose we want to know: Did Abebe see Kebede? Given only one fact, we cannot discover that answer. We can addother facts, such as Spotted (x, y) → saw (x, y). We can now infer the answer to the question.

Set of objects: How should sets of objects be represented?

There are certain properties of objects that are true as member of a set but not as individual;

Example: Consider the assertion made in the sentences: ―there are more sheep than people in Australia‖, and ―English speakers can be found all over the world.‖ To describe these facts, the only way is to attach assertion to the sets representing people, sheep, and English.

The reason to represent sets of objects is:

- if a property is true for all or most elements of a set, then it is more efficient to associate it once with the set rather than to associate it explicitly with every element of the set.
- This is done, in logical representation through the use of universal quantifier, and in hierarchical structure where node represent sets and inheritance propagate set level assertion down to individual.

Finding Right structure:

- Given a large amount of knowledge stored in a database, how can relevant parts be accessed when they are needed?
- This is about access to right structure for describing a particular situation.
- This requires, selecting an initial structure and then revising the choice.

While doing so, it is necessary to solve following problems:

- How to perform an initial selection of the most appropriate structure.
- How to fill in appropriate details from the current situations.
- How to find a better structure if the one chosen initially turns out not to be appropriate.
- What to do if none of the available structures is appropriate.
- When to create and remember a new structure.

Semantic Network Representation

Semantic networks are graphical representations used as an alternative to predicate logic for knowledge representation. In a semantic network, knowledge is represented using nodes to represent objects and arcs to describe the relationships between those objects. The network structure allows for categorization of objects and the linking of related objects. Semantic networks are easily understandable and can be extended as needed. There are two main types of relations in semantic networks: the IS-A relation (Inheritance) and the Kind-of-relation.

Semantic networks are alternative of predicate logic for knowledge representation. In Semantic networks, we can represent our knowledge in the form of graphical networks. This network consists of nodes representing objects and arcs which describe the relationship between those objects. Semantic networks can categorize the object in different forms and can also link those objects. Semantic networks are easy to understand and can be easily extended. This representation consists of mainly two types of relations:

- IS-A relation (Inheritance)
- Kind-of-relation

Example: Following are some statements which we need to represent in the form of nodes and arcs.

Statements:

- Tom is a cat.
- Tom is a mammal
- Tom is owned by Abebe.
- Tom is brown colored.
- All Mammals are animal.

Advantages of Semantic representation

- Semantic networks are a natural representation of knowledge.
- Semantic networks convey meaning in a transparent manner.
- These networks are simple and easily understandable.

Drawbacks in Semantic representation

- Semantic networks take more computational time at runtime as we need to traverse the complete network tree to answer some questions. It might be possible in the worst-case scenario that after traversing the entire tree, we find that the solution does not exist in this network.
- Semantic networks try to model human-like memory (Which has 1015 neurons and links) to store the information, but in practice, it is not possible to build such a vast semantic network.
- These types of representations are inadequate as they do not have any equivalent quantifier, e.g., for all, for some, none, etc.
- Semantic networks do not have any standard definition for the link names.
- These networks are not intelligent and depend on the creator of the system.

Reasoning

Typesof reasoning, including deductive reasoning, inductive reasoning, abductive reasoning,common sense reasoning, monotonic reasoning, and non-monotonic reasoning. It explainsthe characteristics and examples of each type of reasoning.

The advantages and disadvantages of each reasoning type are discussed, highlighting their strengths and limitations. The content also explores the distinction between deductive and inductive reasoning and provides an overview of uncertainty in knowledge representation. It introduces probabilistic reasoning as a way to handle uncertain knowledge and explains concepts such as probability, conditional probability, prior probability, posterior probability, and sample space.

Overall, the section aims to provide a comprehensive understanding of reasoning in AI, covering different types of reasoning and their applications in solving problems with uncertain or incomplete information.

The reasoning is the mental process of deriving logical conclusion and making predictions from available knowledge, facts, and beliefs. Or we can say, "Reasoning is a way to infer facts from existing data." It is a general process of thinking rationally, to find valid conclusions. In artificial intelligence, the reasoning is essential so that the machine can also think rationally as a human brain, and can perform like a human.

In artificial intelligence, reasoning can be divided into the following categories:

- Deductive reasoning
- Inductive reasoning
- Abductive reasoning
- Common Sense Reasoning
- Monotonic Reasoning
- Non-monotonic Reasoning

Deductive reasoning

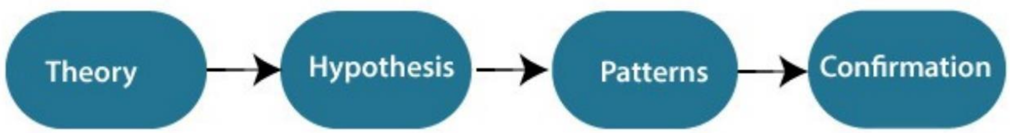
Deductive reasoning is deducing new information from logically related known information. It is the form of valid reasoning, which means the argument's conclusion must be true when the premises are true. Deductive reasoning is a type of propositional logic in AI, and it requires various rules and facts. It is sometimes referred to as top-down reasoning, and contradictory to inductive reasoning. In deductive reasoning, the truth of the premises guarantees the truth of the conclusion. Deductive reasoning mostly starts from the general premises to the specific conclusion, which can be explained as below example.

Premise-1: All the human eats veggies

Premise-2: Abebe is human.

Conclusion: Abebe eats veggies.

The general process of deductive reasoning is given below:

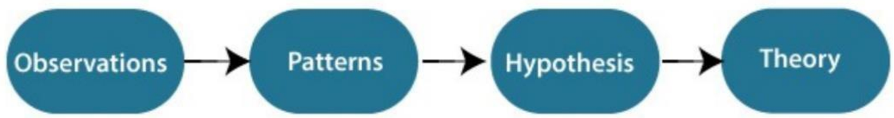


Inductive reasoning

Inductive reasoning is a form of reasoning to arrive at a conclusion using limited sets of facts by the process of generalization. It starts with the series of specific facts or data and reaches to a general statement or conclusion. Inductive reasoning is a type of propositional logic, which is also known as cause-effect reasoning or bottom-up reasoning. In inductive reasoning, we use historical data or various premises to generate a generic rule, for which premises support the conclusion. In inductive reasoning, premises provide probable supports to the conclusion, so the truth of premises does not guarantee the truth of the conclusion.

Example:

- Premise: All of the pigeons we have seen in the zoo are white.
- Conclusion: Therefore, we can expect all the pigeons to be white.



Abductive reasoning

Abductive reasoning is a form of logical reasoning which starts with single or multiple observations then seeks to find the most likely explanation or conclusion for the observation. Abductive reasoning is an extension of deductive reasoning, but in abductive reasoning, the premises do not guarantee the conclusion.

Example:

Implication: Football ground is wet if it is raining

Axiom: Football ground is wet.

Conclusion It is raining.

Common Sense Reasoning

Common sense reasoning is an informal form of reasoning, which can be gained through experiences. Common Sense reasoning simulates the human ability to make presumptions about events which occurs on every day. It relies on good judgment rather than exact logic and operates on heuristic knowledge and heuristic rules.

Example:

- One person can be at one place at a time.
- If I put my hand in a fire, then it will burn.

The above two statements are the examples of common-sense reasoning which a human mind can easily understand and assume.

Monotonic Reasoning

In monotonic reasoning, once the conclusion is taken, then it will remain the same even if we add some other information to existing information in our knowledge base. In monotonic reasoning, adding knowledge does not decrease the set of prepositions that can be derived. To solve monotonic problems, we can derive the valid conclusion from the available facts only, and it will not be affected by new facts. Monotonic reasoning is not useful for the real-time systems, as in real time, facts get changed, so we cannot use monotonic reasoning. Monotonic reasoning isused in conventional reasoning systems, and a logic-based system is monotonic. Any theorem proving is an example of monotonic reasoning.

Example:

- *Earth revolves around the Sun.* It is a true fact, and it cannot be changed even if we add another sentence in knowledge base like, "The moon revolves around the earth" Or "Earth is not round," etc.

Advantages of Monotonic Reasoning:

- In monotonic reasoning, each old proof will always remain valid.
- If we deduce some facts from available facts, then it will remain valid for always.

Disadvantages of Monotonic Reasoning:

- We cannot represent the real-world scenarios using Monotonic reasoning.
- Hypothesis knowledge cannot be expressed with monotonic reasoning, which means facts should be true.
- Since we can only derive conclusions from the old proofs, so new knowledge from the real world cannot be added.

Non-monotonic Reasoning

In Non-monotonic reasoning, some conclusions may be invalidated if we add some more information to our knowledge base. Logic will be said as non-monotonic if some conclusions can be invalidated by adding more knowledge into our knowledge base. Non-monotonic reasoning deals with incomplete and uncertain models. "Human perceptions for various things in daily life, "is a general example of non-monotonic reasoning.

Example:

Let suppose the knowledge base contains the following knowledge:

- Birds can fly
- Penguins cannot fly
- Pitty is a bird, So from the above sentences, we can conclude that Pitty can fly. However, if we add one another sentence into knowledge base "Pitty is a penguin", which concludes "Pitty cannot fly", so it invalidates the above conclusion.

Advantages of Non-monotonic reasoning:

- For real-world systems such as Robot navigation, we can use non-monotonic reasoning.
- In Non-monotonic reasoning, we can choose probabilistic facts or can make assumptions.

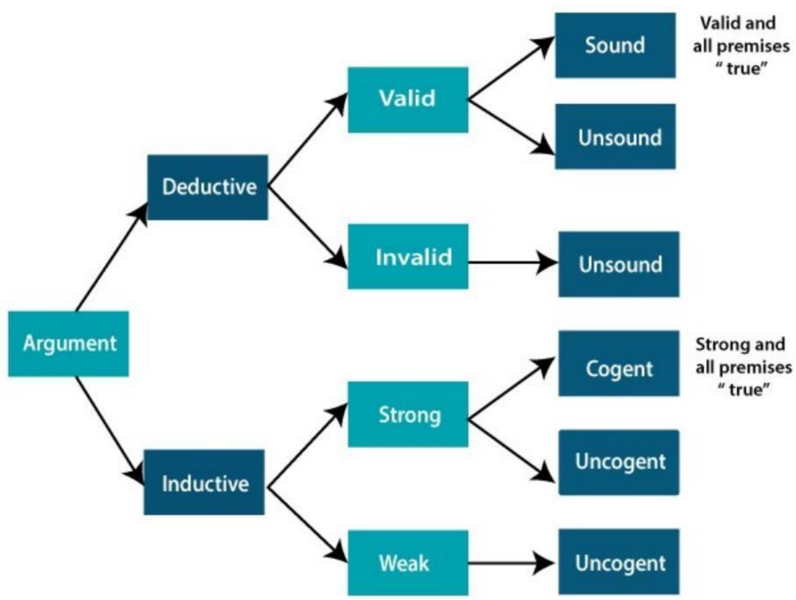
Disadvantages of Non-monotonic Reasoning:

- In non-monotonic reasoning, the old facts may be invalidated by adding new sentences.

- It cannot be used for theorem proving.

Inductive vs. Deductive reasoning

- Deductive reasoning uses available facts, information, or knowledge to deduce a valid conclusion, whereas inductive reasoning involves making a generalization from specific facts, and observations.
- Deductive reasoning uses a top-down approach, whereas inductive reasoning uses a bottom-up approach.
- Deductive reasoning moves from generalized statement to a valid conclusion, whereas Inductive reasoning moves from specific observation to a generalization.
- In deductive reasoning, the conclusions are certain, whereas, in Inductive reasoning, the conclusions are probabilistic.
- Deductive arguments can be valid or invalid, which means if premises are true, the conclusion must be true, whereas inductive argument can be strong or weak, which means conclusion may be false even if premises are true.



Uncertainty

Till now, 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 $A \rightarrow B$, 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.

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.

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.

Need of probabilistic reasoning in AI:

- When there are unpredictable outcomes.
- When specifications or possibilities of predicates becomes too large to handle.
- When an unknown error occurs during an experiment.

In probabilistic reasoning, there are two ways to solve problems with uncertain knowledge:

- Bayes' rule
- Bayesian Statistics

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

- 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.

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

P(¬A) = probability of a not happening event.

P(¬A) + P(A) = 1.

Event: Each possible outcome of a variable is called an event.

- Sample space: The collection of all possible events is called sample space.
- Random variables: Random variables are used to represent the events and objects in the real world.
- Prior probability: The prior probability of an event is probability computed before observing new information.
- Posterior Probability: The probability that is calculated after all evidence or information has taken into account. It is a combination of prior probability and new information.

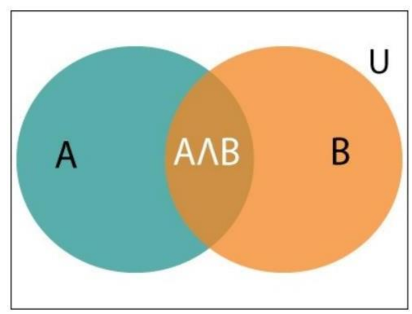
Conditional probability

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:

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

Where, P(A ∧ B)= Joint probability of A and B

P(B)= Marginal probability of B.



It can be explained by using the below Venn diagram, where B is occurred event, so sample space will be reduced to set B, and now we can only calculate event A when event B is already occurred by dividing the probability of P(A∧B) by P(B).

Example:

In a class, there are 70% of the students who like English and 40% of the students who likes English and mathematics, and then what is the percent of students those who like English also like mathematics?

Solution:

Let, A is an event that a student likes Mathematics

B is an event that a student likes English.

$$P(A|B) = \frac{P(A \wedge B)}{P(B)} = \frac{0.4}{0.7} = 57\%$$

Hence, 57% are the students who like English also like Mathematics.