

Probability & Bayesian Networks

Explain the Bayes Theorem.

Bayes Theorem is a fundamental principle in probability theory that helps us to update what we believe after seeing new evidence. It shows how to update what we believe based on what we already know and what we just found out.

The basic form of Bayes' Theorem is:

$$P(A|B) = P(B|A) \times P(A) / P(B)$$

Where:

- $P(A|B)$ = The probability of A is true given that B has happened
- $P(B|A)$ = The probability of B would happen if A is true
- $P(A)$ = The initial probability of A
- $P(B)$ = The total probability of B

It helps us answer the question: “Now that we’ve seen this happen, what’s the chance that what we believed is still true?” We start with what we already know, then we combine it with the new information we’ve just seen, and the theorem gives us an updated belief. We use it in many real-life situations—like in medicine to guess a disease, in spam filters to catch junk emails, or in machine learning to make better predictions.

How a Bayes net encodes a joint distribution?

A Bayesian Network encodes a joint probability distribution in a simple and smart way. Each variable is a node, and edges between them show how they're connected or depend on each other. Instead of listing every possible case, it breaks things down into smaller pieces—showing the chance of each variable based on the ones it depends on. Then, it multiplies all these smaller chances to get the full picture.

For a group of variables X_1, X_2, \dots, X_n , a Bayes net shows their joint probability by,

$$P(X_1, X_2, \dots, X_n) = \prod_i P(X_i \mid \text{Parents}(X_i))$$

A survey was conducted at a juice bar, and the following data were collected:
 8 male students, 5 female students, 3 male teachers and 2 female teachers
 bought orange juice 6 male students, 7 female students, 2 male teachers and 4
 female teachers bought mango juice 4 male students, 5 female students, 2
 male teachers and 2 female teachers bought lemon juice Answer the
 following question: C= teacher | student i. Create a Joint probability
 distribution table for the given data. ii. $P(C)=?$ iii. $P(C | \text{Female})=?$ iv. $P(C |$
 $\text{Female, Student})=?$ Calculate the probability using inference by enumeration.

A well-known symptom of measles is a rash (the event of having which we denote R). Assume that the probability of having a rash if one has measles is $P(R|M) = 0.95$. However, occasionally children with flu also develop rash, and the probability of having a rash if one has flu is $P(R|F) = 0.08$. Upon examining the child, the doctor finds a rash. What is the probability that the child has measles?

We are given:

$P(R | M) = 0.95$ (Probability of
 rash if the child has measles)

$P(R | F) = 0.08$ (Probability of
 rash if the child has flu)

The child has a rash — we want to
 find:

$P(M | R) = ?$ (Probability the child
 has measles given rash)

To use Bayes' Theorem, we apply:

$P(M | R) = [P(R | M) \times P(M)] / P(R)$

We already have:

$P(R | M) = 0.95$

$P(R | F) = 0.08$

We also need the priors: Lets

Assume

$P(M) = 0.1$ (10% chance child has
 measles)

$P(F) = 0.9$ (90% chance child has
 flu)

**Step 1: Find $P(R)$ using Total
 Probability Rule**

$P(R) = P(R | M) \times P(M) + P(R | F)$
 $\times P(F)$

$= 0.95 \times 0.1 + 0.08 \times 0.9$

$= 0.095 + 0.072$

$= 0.167$

Step 2: Apply Bayes' Theorem

$P(M | R) = (0.95 \times 0.1) / 0.167$

$= 0.095 / 0.167$

≈ 0.568

So, there's about a 56.8% chance the
 child has measles, given that they
 have a rash.

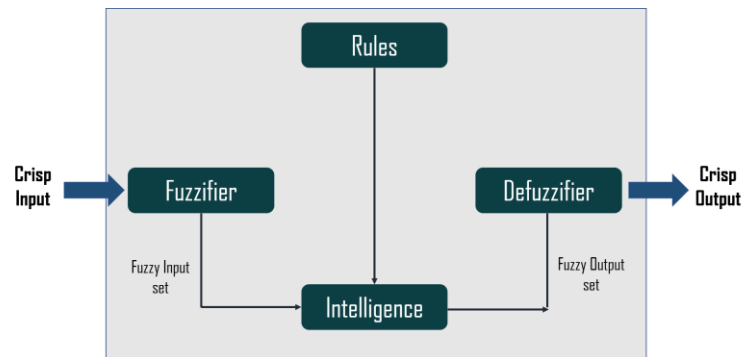
What is the role of membership function in Fuzzy Logic? Describe the fuzzy logic process diagram in brief.

Role of Membership Function in Fuzzy Logic:

1. It tells us how much something belongs to a fuzzy set by giving it a value between 0 and 1.
2. It shows how input values are connected to their level of belonging.
3. We can use different shapes for it, like triangles, trapezoids, or bell curves, based on what the system needs.

Fuzzy Logic Process Diagram: It typically follows four main steps,

1. **Fuzzification:** Converts crisp input values (like numbers) into fuzzy values using membership functions.
2. **Rule Evaluation (Inference Engine):** Applies IF-THEN fuzzy rules to the fuzzified inputs.
3. **Aggregation:** Combines the outputs from all active rules to form a single fuzzy output set.
4. **Defuzzification:** Converts the fuzzy output back into a crisp value (like a number) using methods like Centroid or Max Membership.



What do you mean by uncertainty? What are the different sources of uncertainty?

Uncertainty means we're not fully sure about something or don't have complete information. In AI, many real-life situations are unclear or unpredictable, so it's hard for the system to always make perfect choices.

Sources of Uncertainty:

1. **Incomplete Knowledge:** We may not have all the necessary information about the environment or situation.
2. **Imprecise or Vague Information:** Information may be vague or not clearly defined.

3. **Noisy Data or Sensor Errors:** Sensors or data sources can provide **inaccurate or faulty** readings.
4. **Randomness in the Environment:** Some things just happen by chance and can't be predicted.
5. **Ambiguity:** One thing might mean different things in different cases.
6. **Lack of Training Data or Examples:** The system might not have seen enough examples to be sure.

How would you model the speed of a vehicle (65km/h) in both fuzzy and crisp sets?

1. **Crisp Set:** In a crisp set, 65 km/h either fits the group or it doesn't. Like if "Fast" means anything over 60 km/h, then 65 is Fast (value = 1). If it's not, then it's just not Fast (value = 0). There's no in-between.
2. **Fuzzy Set:** In a fuzzy set, 65 km/h can belong to more than one group at the same time, but with different levels. Like it could be 60% Moderate and 40% Fast. It lets things change more smoothly between categories.

"Fuzzy set is the generalized version of crisp set"- Do you agree with the statement? Justify your answer.

Yes, I agree with the statement: "Fuzzy set is the generalized version of crisp set."

Justification:

Crisp Set: In normal (classical) sets, something either belongs to the set or it doesn't. It's either 0 or 1.

Fuzzy Set: A fuzzy set allows in-between values. Instead of saying "yes" or "no," it says "how much" something belongs to a set. This is useful when things aren't completely clear or exact.

Fuzzy sets include crisp sets as a special case. Fuzzy sets can also handle all the values in between 0 and 1. That's why fuzzy sets are considered a more general or flexible version of crisp sets.

Example:

Let's say we're talking about "tall people."

- In a crisp set, anyone above 6 feet is tall (1), and anyone below is not (0).
- In a fuzzy set, someone who's 5'11" might be 0.9 tall, someone who's 5'9" might be 0.7 tall, and so on.

Since fuzzy sets can handle both crisp (0 or 1) and partial (between 0 and 1) membership, they are rightly called a generalization of crisp sets.

Natural Language Processing (NLP)

What is NLP? Write down the steps through which NLP is conducted. Write down the types of ambiguity in NLP.

Natural Language Processing (NLP) is a field of artificial intelligence that helps computers understand and work with human language. It allows computers to read, listen, speak, or write in ways that make sense to people.

Steps of NLP with Examples:

1. Lexical Analysis: It breaks the text into smaller parts like words, sentences, or paragraphs. It's the first step that prepares the text for deeper analysis.

Example:

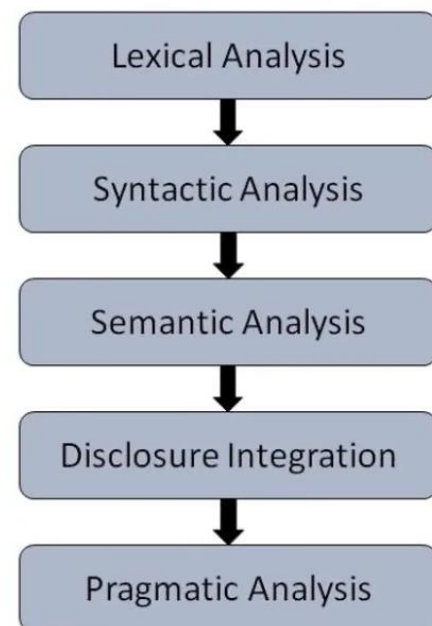
- Input: "The cats are playing."
- Output: ['The', 'cats', 'are', 'playing']

2. Syntactic Analysis (Parsing): It checks if the sentence follows grammar rules. It builds a structure showing how words relate to each other.

Example:

- Input: "She reading is book."
- Output: This is incorrect.
- Correct form: "She is reading a book."

3. Semantic Analysis: It figures out what the sentence means and checks if the words make sense together.



Example:

- Input: "Hot ice cream melted slowly."
- Output: The phrase "hot ice cream" doesn't make much sense.

4. Discourse Integration: It connects sentences together to keep the meaning clear and consistent.

Example:

- Sentence 1: "Sarah went to the library."
- Sentence 2: "She borrowed a book."
- Output: "She" is understood to mean "Sarah."

5. Pragmatic Analysis: It looks at the real meaning behind a sentence by using real-world knowledge and context.

Example:

- Input: "Can you pass the salt?"
- Output: It's not really a question—it's a polite way of asking someone to pass the salt.

Types of Ambiguity in NLP

1. **Lexical Ambiguity:** This happens when a word has more than one meaning.

Example: The word "board" can mean a flat wooden piece (noun) or the act of getting onto something, like a bus (verb).

2. **Syntactic Ambiguity:** This is when a sentence can be understood in more than one way because of its structure.

Example: "He lifted the beetle with the red cap." Did he use a red cap to lift the beetle, or did he lift a beetle that had a red cap?

3. **Referential Ambiguity:** This happens when it's unclear who or what a pronoun refers to.

Example: "Rima went to Gauri. She said, 'I am tired.'" Here, it's not clear whether Rima or Gauri is tired.

Compare top down and bottom up parsing approach in parsing sentence tree.

Aspect	Top-Down Parsing	Bottom-Up Parsing
1. Definition	Starts from the whole sentence and breaks it down step by step.	Starts from the words and builds up to the full sentence.
2. Approach	Predicts what should come next using grammar rules and checks if it fits.	Combines words into phrases, then phrases into bigger parts.
3. Efficiency	Can be slower because it might guess wrong and have to try again.	Usually faster because it works with what's actually there.
4. Error Handling	Has trouble with confusing or wrong sentences and may get stuck.	Handles small confusions better by focusing on real input.
5. Use in AI	Used when the system relies on strict grammar rules.	Used more in smart, data-based systems like AI and neural nets.

Knowledge-Based Systems & Logic

How can the Wumpus World be described in terms of its features and environment?

- 1. Environment Features:** The Wumpus World is like a small grid map where each square can have a pit, some gold, or a monster called the Wumpus. The agent doesn't see everything clearly but gets hints.
- 2. Agent Capabilities and Goals:** The agent can walk, turn, grab the gold, shoot an arrow, or climb out of the cave. Its goal is to find the gold and get out safely without falling into a pit or getting eaten by the Wumpus. It has to use clues and think logically to make smart moves.

The figure below shows a Wumpus World game, where the agent starts from location [1,1]. Briefly discuss with diagrams how the agent generates knowledge through perceiving the environment and choosing his next move.

Exploring the Wumpus World

OK			
OK A	OK		

4	SSSSS Stench		Breeze	PIT
3	Wumpus	Breeze SSSSS Stench Gold	PIT	Breeze
2	SSSSS Stench		Breeze	
1	START	Breeze	PIT	Breeze
	1	2	3	4


No breeze and stench at (1, 1)

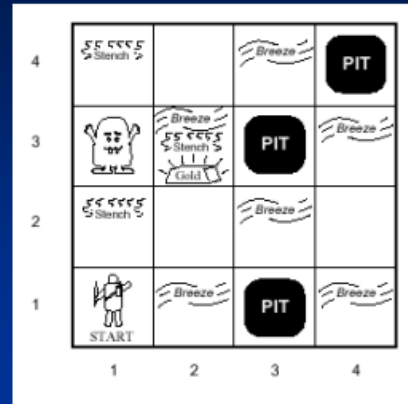
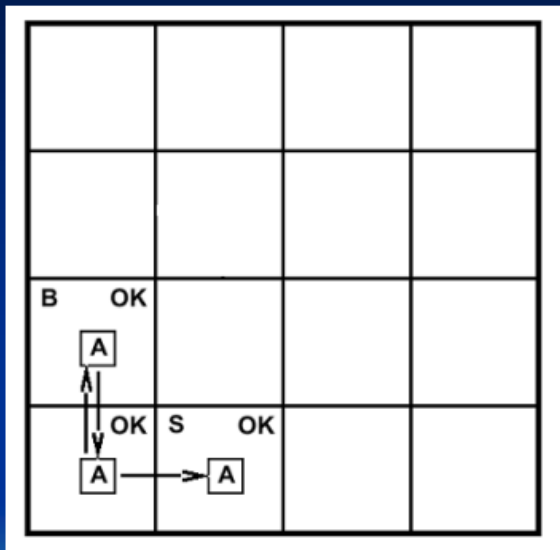
➡ (1, 2), (2, 1) are OK

B	OK		
A			
OK	OK		

4	SSSSS Stench		Breeze	PIT
3	Wumpus	Breeze SSSSS Stench Gold	PIT	Breeze
2	SSSSS Stench		Breeze	
1	START	Breeze	PIT	Breeze
	1	2	3	4

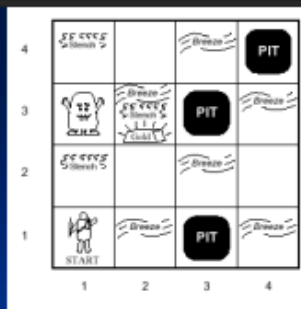
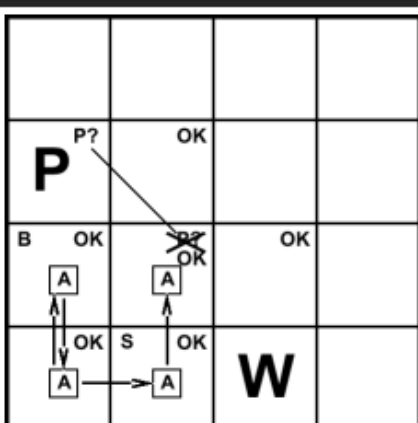
Stench in (2, 1) (B)

➡  in (3, 1) or (2, 2)



Breeze in (1, 2) (B)

PIT in (1, 3) or (2, 2)



in (3, 1) or (2, 2)

PIT in (1, 3) or (2, 2)

PIT & not in same room

(2, 2) is safe !

in (3, 1) or (2, 2)

PIT in (1, 3) or (2, 2)

PIT & not in same square

(2, 2) is safe !

Knowledge base

Inference

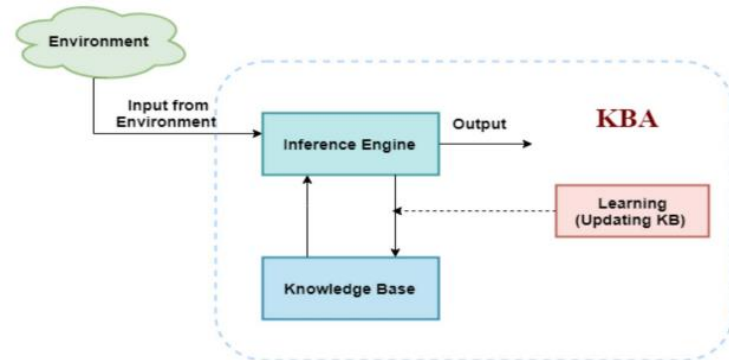
New Knowledge

What is knowledge-base? Draw and discuss the architecture of a knowledge-based agent. Write the tasks of TELL and ASK function in a knowledge based system.

A knowledge base is a collection of helpful facts and rules that an intelligent system uses to understand things, solve problems, and make smart decisions about the world around it.

A knowledge-based agent has two main parts:

1. **Knowledge Base (KB)** – where it keeps facts and rules
2. **Inference Engine (IE)** – the part that thinks and figures things out using the KB



It works like this:

1. It gets information from its surroundings (called percepts)
 2. It uses what it knows and the inference engine to decide what to do
 3. It takes action or sometimes just adds new facts to what it knows
-
- **TELL:** Adds new facts or info to what the system already knows.
 - **ASK:** Looks up or figures out answers from the known info.
 - **PERFORM:** Does something based on what the system decides.

State the ontological and epistemological commitment of the propositional, first-order, probability theory, and fuzzy logic.

Ontologically, Propositional and first-order logic see the world as made up of clear facts and things that are either true or false. But probability and fuzzy logic understand that sometimes things are uncertain or not so clear. Epistemologically, propositional and first-order logic think of it as exact and sure, while probability shows knowledge as chances or likelihoods, and fuzzy logic shows it as partly true or partly false.

Differentiate between forward chaining and backward chaining. In what aspects forward chaining should be used?

Point	Forward Chaining	Backward Chaining
1. How It Works	Starts with what you already know and keeps applying rules to find new information.	Starts with what you want to prove and checks if what you know supports it.
2. Main Idea	Driven by data — it moves forward from facts.	Driven by the goal — it moves backward from the target.
3. When to Use	When you have lots of facts and want to see what conclusions can be made.	When you have a specific question or goal to check or prove.
4. Speed & Focus	Can be slower if there are too many rules firing without a clear goal.	Usually faster for one goal but not great if you need all possible conclusions.
5. Where It's Used	Common in expert systems, automation, and monitoring tools.	Common in medical diagnosis, decision-making systems, and logic-based tools.

✓ When to Use Forward Chaining

- When you already **have a lot of facts** or data in hand.
- When you want to **find all the possible results** from what you know.
- When the system needs to **respond automatically** to conditions (like in smart devices or monitoring).
- When your rules are built to **flow naturally from facts to conclusions**.
- In places like **fact-based systems, sensors, or control systems**, where data is coming in all the time.

Explain with example why universal quantification elimination is easy while existential quantification elimination is not that much easy.

Universal quantifier elimination is easy because we can just plug in any example. For instance, if we know “**All birds can fly**” (written as $\forall x (\text{Bird}(x) \rightarrow \text{CanFly}(x))$), then we can easily say “**If a sparrow is a bird, then it can fly**” — we just picked “sparrow” as an example. This works because the rule applies to *every* bird.

But existential quantifier elimination is trickier. Suppose we have “**There exists a bird that cannot fly**” (written as $\exists x (\text{Bird}(x) \wedge \text{CannotFly}(x))$). This tells us *some* bird can’t fly — but we don’t know *which* one. We can’t just say “**Penguin can’t fly**” unless we’re sure it’s the penguin the statement is talking about. So, while universal elimination is like picking any example freely, existential elimination needs more care because we don’t know exactly *who* the example is.

Briefly explain why FOL is considered as the generalization of PL.

First-Order Logic (FOL) is called a generalization of Propositional Logic (PL) because it can do everything PL does—and more. PL talks about whole statements being true or false, but FOL adds extra tools like **quantifiers** (such as "for all" and "there exists"), **variables**, and **predicates**. This lets FOL talk about objects, their properties, and how they relate to each other, making it much more powerful and flexible.

The following knowledge base R1, R2, R3, R4, R5 has given below.

R1: $\neg P_{1,1}$

R2: $B_{1,1} \leftrightarrow (P_{2,1} \vee P_{1,2})$

R3: $B_{2,1} \leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

R4: $\neg B_{1,1}$

R5: $B_{2,1}$

Show that $\neg P_{1,2}$ is true

► From R2 and R4:

We use:

- $B_{1,1} \leftrightarrow (P_{2,1} \vee P_{1,2})$ — i.e., $B_{1,1}$ is true iff $(P_{2,1} \vee P_{1,2})$ is true.
- But from R4: $\neg B_{1,1}$, i.e., $B_{1,1}$ is false

So:

$P_{2,1} \vee P_{1,2}$ must also be false

This implies:

$\neg P_{2,1} \wedge \neg P_{1,2}$

✓ So we conclude:

- $\neg P_{2,1}$
- $\neg P_{1,2} \leftarrow$ this is what we needed to prove ✓

What is Conjunctive Normal Form (CNF)? Convert the following sentence to CNF form: $B_{1,1} \supset (P_{2,1} \vee P_{1,2})$

CNF (Conjunctive Normal Form) is a conjunction (\wedge) of disjunctions (\vee) of literals.

1. Replace biconditional (\leftrightarrow):

$$B_{1,1} \leftrightarrow (P_{2,1} \vee P_{1,2}) \rightarrow (B_{1,1} \rightarrow (P_{2,1} \vee P_{1,2})) \wedge ((P_{2,1} \vee P_{1,2}) \rightarrow B_{1,1})$$

2. Replace implications (\rightarrow):

$$\rightarrow \neg B_{1,1} \vee P_{2,1} \vee P_{1,2} \text{ and } \neg(P_{2,1} \vee P_{1,2}) \vee B_{1,1}$$

3. Apply De Morgan's Law:

$$\neg(P_{2,1} \vee P_{1,2}) \rightarrow \neg P_{2,1} \wedge \neg P_{1,2} \rightarrow \text{distribute with } \vee B_{1,1}$$

4. Final CNF form:

$$\neg B_{1,1} \vee P_{2,1} \vee P_{1,2}$$

$$\neg P_{2,1} \vee B_{1,1}$$

$$\neg P_{1,2} \vee B_{1,1}$$

✓ CNF:

$$(\neg B_{1,1} \vee P_{2,1} \vee P_{1,2}) \wedge (\neg P_{2,1} \vee B_{1,1}) \wedge (\neg P_{1,2} \vee B_{1,1})$$

Neural Networks & Machine Learning

What is learning? What are supervised learning, unsupervised learning, and reinforcement learning?

Learning means a machine gets better at doing something by using past experience or data.

- ❖ **Supervised learning** is when the machine learns from examples that already have the right answers.
- ❖ **Unsupervised learning** is when the machine looks at data without answers and tries to find patterns or groups.
- ❖ **Reinforcement learning** is when the machine learns by trying things out and getting rewards or punishments based on what it does.

How do neural networks differ from conventional computers? Briefly explain.

Neural networks work in a very different way compared to regular computers. A normal computer follows fixed steps or rules to solve a problem. But a neural network learns from examples. It changes the strength

of the connections between its artificial “neurons” based on what it sees. This helps it get better over time. That’s why neural networks are great at things like recognizing faces or understanding speech—things that are hard to explain with clear rules.

How do single-layer recurrent neural networks differ from multilayer feedforward neural networks?

1. **Memory:** A single-layer RNN can remember what it saw before; a feedforward network can’t.
2. **Loop vs. Straight:** RNNs have loops that send info back; feedforward networks just go straight forward.
3. **Best Use:** RNNs work well for things that come in order, like text or time; feedforward networks are better for things like pictures.

What is an Artificial Neural Network? Compare artificial and biological networks. What aspects of biological networks are not mimicked by artificial ones? What aspects are similar?

An **Artificial Neural Network (ANN)** is a computational model inspired by the structure and functioning of the human brain. It consists of **interconnected layers of nodes (neurons)** that process data by assigning weights and applying activation functions.

Aspect	Artificial Neural Network (ANN)	Biological Neural Network
1. Structure	Consists of nodes (neurons) arranged in layers	Consists of real neurons connected by synapses
2. Signal Transmission	Uses numerical values and weights	Uses electrochemical signals
3. Speed	Fast in performing specific computations	Slower, but capable of massively parallel processing
4. Energy Efficiency	Consumes more power, especially in large models	Extremely energy-efficient
5. Adaptability	Needs retraining for new tasks	Highly adaptable and capable of transfer learning

Aspects Not Mimicked by Artificial Networks

1. Biological neurons can rewire themselves over time; ANNs usually have fixed architectures.
2. The brain can perform billions of parallel operations in real time, far beyond ANN capabilities.
3. Biological networks can recover from damage and grow new connections.
4. Real neurons use complex chemical processes and feedback loops, which ANNs do not replicate.
5. Humans learn from context, emotions, and experience. ANNs rely solely on input-output data.

Similar Aspects

1. Both have units (neurons) connected to others (via synapses or weights).
2. Both learn patterns from experience (data or environment).
3. Both process information in multiple stages (layers).
4. Like neurons firing based on stimulus, ANN nodes activate based on input thresholds.
5. Both can recognize and classify patterns like faces, voices, etc.

Explain the process of knowledge acquisition in expert systems. Why is it considered a bottleneck in expert system development?

Knowledge acquisition is the process of collecting, organizing, and encoding knowledge from human experts or other sources into a form that can be used by an expert system.

Steps in Getting Knowledge for Expert Systems:

1. **Find the Expert:** Choose someone who really knows the topic well.
2. **Collect Information:** Ask questions, observe, or use surveys to gather what they know.
3. **Sort and Organize:** Arrange the information into rules, patterns, or key ideas.
4. **Write It Formally:** Turn the information into a form that a computer can understand.

5. **Check and Improve:** Test it with real examples and make changes if needed.

Why This is Often a Problem (Bottleneck):

1. **Hard to Explain Gut Feelings:** Experts may know things but struggle to explain them clearly.
2. **Takes a Lot of Time:** It needs many meetings and careful note-taking.
3. **Experts May Be Busy or Unwilling:** Sometimes, they don't have time or don't want to help much.
4. **Topics Can Be Too Complicated:** Some fields are so complex that it's hard to capture everything.
5. **Risk of Getting It Wrong:** The person writing it down might not fully understand the expert.

What are expert system shells? How do they help in building expert systems?

An expert system shell is like a ready-made toolkit for building expert systems. It has the main parts needed—like the knowledge base, the reasoning engine, and the user interface—but it doesn't have any actual expert knowledge inside

How they help build expert systems:

1. Save time by providing ready tools like rule editors and reasoning engines.
2. Let you focus on adding knowledge without programming the system logic.
3. Offer easy-to-use graphical interfaces for adding and testing rules.
4. Can be reused for different systems by changing the knowledge base.
5. Include built-in reasoning methods to draw conclusions automatically.

Describe the concept of inductive learning with an example.

Inductive learning is when you figure out a rule by looking at many examples instead of being told the rule directly. You notice patterns in what you see and then make a general idea from that.

Example:

If you see these:

- The sun comes up every morning.
- The sun came up yesterday.
- The sun came up the day before yesterday.

You learn from these that the sun comes up every morning. You made a general rule from these specific times.

In machine learning, a computer might look at lots of pictures of cats and dogs and learn what makes a cat different from a dog. Then it can tell if a new picture is a cat or a dog based on what it learned.

Explain the following inference rules of propositional logic with proper examples:

- a. Hypothetical Syllogism**
- b. Modus Ponens**
- c. Modus Tollens**

a. Hypothetical Syllogism

What it means:

If “P leads to Q” and “Q leads to R” are true, then you can say “P leads to R.”

In short:

If $P \rightarrow Q$ and $Q \rightarrow R$, then $P \rightarrow R$.

Example:

- If it rains (P), the ground gets wet (Q).
- If the ground is wet (Q), the game will be canceled (R).
- So, if it rains (P), the game will be canceled (R).

b. Modus Ponens

What it means:

If “P leads to Q” is true and P actually happens, then Q must happen.

In short:

If $P \rightarrow Q$ and P , then Q .

Example:

- If it is sunny (P), I will go for a walk (Q).
 - It is sunny (P).
 - So, I will go for a walk (Q).
-

c. Modus Tollens**What it means:**

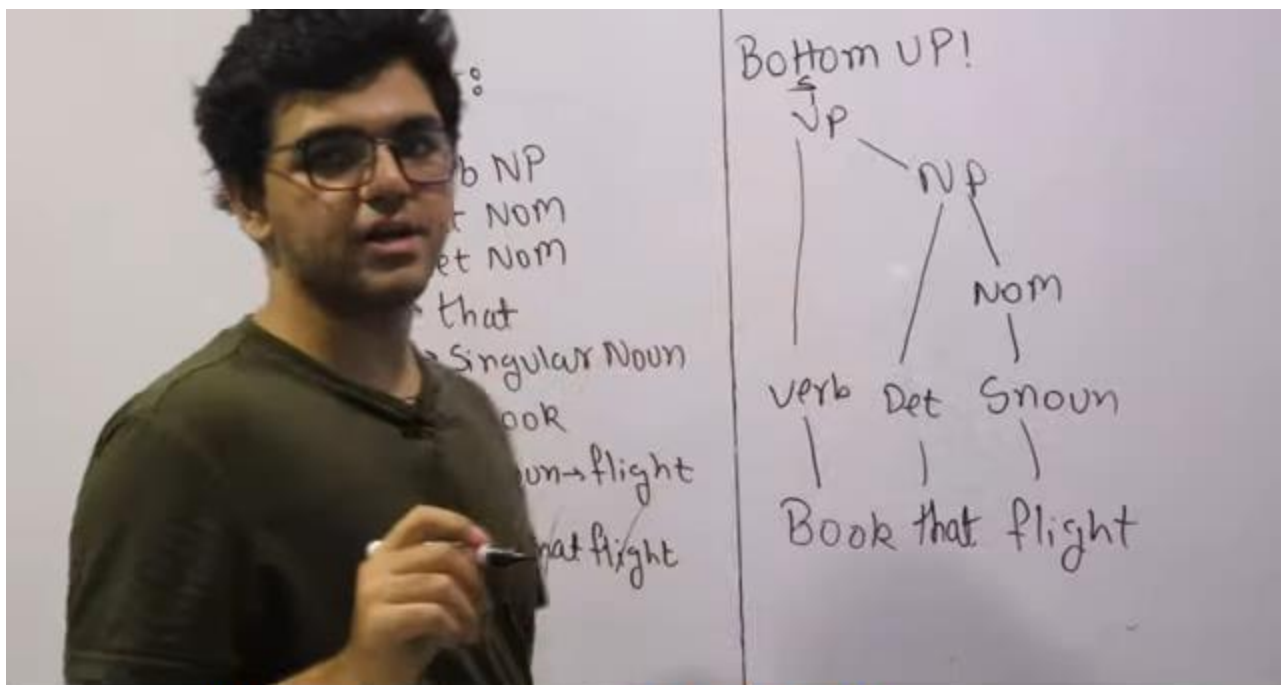
If “ P leads to Q ” is true but Q didn’t happen, then P didn’t happen either.

In short:

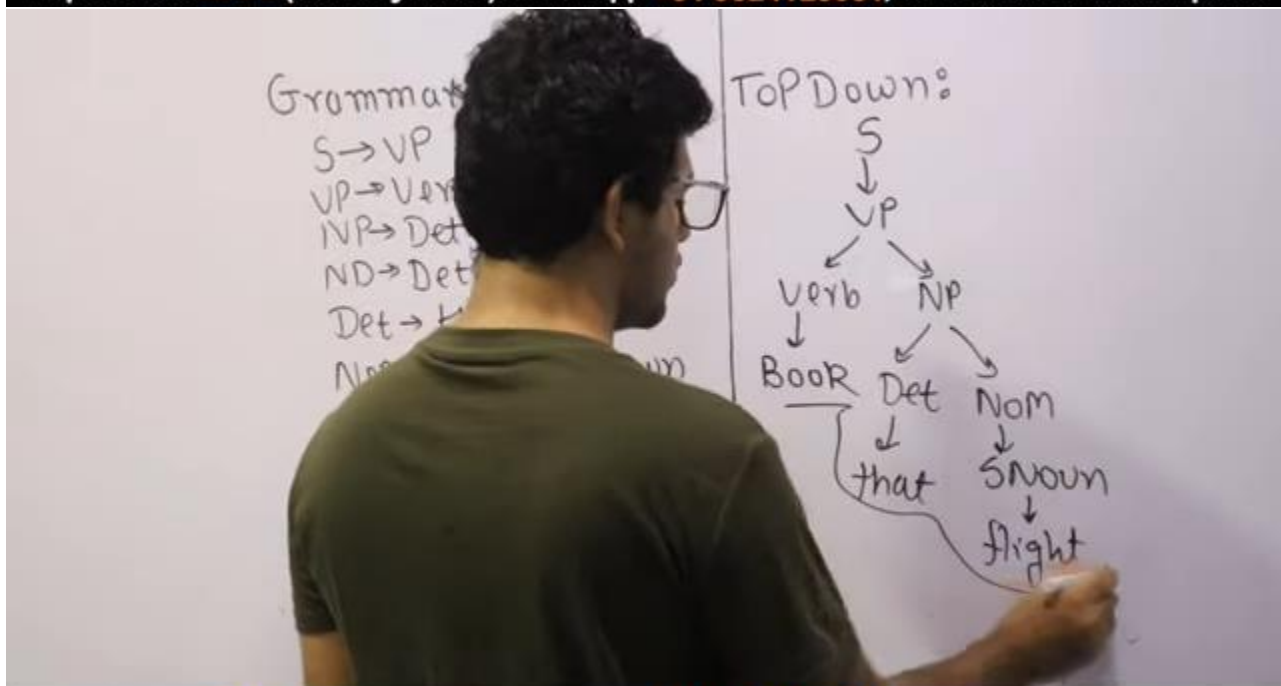
If $P \rightarrow Q$ and not Q , then not P .

Example:

- If the alarm is set (P), it will ring in the morning (Q).
- The alarm didn’t ring (not Q).
- So, the alarm wasn’t set (not P).



To purchase notes (latest syllabus) WhatsApp: +91 9324125031, more details in description.



To purchase notes (latest syllabus) WhatsApp: +91 9324125031, more details in description.