

# AI Viva Answers

This is a copy of a conversation between ChatGPT & Anonymous.



I would be providing you with the units of chaps in ai , I want you to answer me in chat viva appropriate but still some deep as in not one line one para concise answer for each topic got it?

Got it! Send me the units or topics from your AI syllabus, and I'll give you viva-appropriate answers—concise but with enough depth to show understanding (one paragraph per topic, not just one-liners). Ready when you are.



## UNIT 2

Heuristic Search Techniques: Generate-and-Test; Hill Climbing; Properties of A\* algorithm, Best-first Search; Problem Reduction.

Constraint Satisfaction problem: Interference in CSPs; Backtracking search for CSPs; Local Search for CSPs; structure of CSP Problem.

Beyond Classical Search: Local search algorithms and optimization problem, local search in continuous

spaces, searching with nondeterministic action and partial observation, online search agent and unknown environments.

## UNIT 3

Knowledge Representation: Introduction to Knowledge Representation, Knowledge agent, Predicate logic, WFF, Inference rule & theorem proving: forward chaining, backward chaining, resolution; Propositional knowledge, Boolean circuit agents. Rule Based Systems, Knowledge Reasoning: Forward reasoning: Conflict resolution, backward reasoning: Use of backtracking,

Structured Knowledge Reasoning: Semantic Net - slots, inheritance, Frames- exceptions and defaults attached predicates, Conceptual Dependency formalism,

Reasoning Under Uncertainty: Source of Uncertainty, Probabilistic Reasoning and Uncertainty; Probability

theory; Bayes Theorem and Bayesian networks, Certainty Factor, Dempster-Shafer theory, Non Monotonic

Reasoning, Truth maintenance Systems, Overview of Fuzzy Logic

## UNIT 4

Introduction: What is NLP, Steps in Natural Language Processing,

Syntactic Analysis(Parsing): Grammars and Parsers, Augmented Transition Networks, Unification grammars

Semantic Analysis: Semantic grammar, Case grammars, Conceptual parsing, Approximately

Compositional Semantic Interpretation.

Discourse and Pragmatic Processing: Using focus in Understanding, Modeling Beliefs, Using Goals and

Plans for Understanding, Speech Acts, Conversational Postulates  
Text classification (Spell Checking), Probabilistic Language Models, Implementation aspects of Syntactic Analysis(Parsing)

## UNIT 5

Game Playing: Overview and Examples.

Domain: Overview, MiniMax, Alpha-Beta Cut-off, Refinements, Iterative deepening, The Blocks World,

Components of A Planning System, Goal Stack Planning, Nonlinear Planning Using Constraint Posting, Hierarchical Planning, Reactive Systems.

## UNIT 6

Deep Learning: Introduction, Why to go deep? Architecture of Deep Network, Restricted Boltzmann Machines, Deep belief Network, Tensor Flow, Deep Learning libraries, Deep Learning platform, The no,

Caffe, Deep Learning Use Cases.

Applications: Overview of Artificial Intelligence Domains, AI-Robotics, AI-Neural Networks, AI-IOT, Computer Vision in AI

Case Studies: Automatic Bird Identification using Deep Learning, Tumkur monitoring using Computer

Vision, Text to Speech Conversion using APIs

Perfect. Let's begin with **UNIT 2**. Here are viva-appropriate, concise yet detailed answers:

## UNIT 2 – Heuristic Search Techniques & CSP

### 1. Generate-and-Test:

Generate-and-Test is a simple problem-solving approach where potential solutions are generated and tested against a goal or constraint. It does not use any intelligence in choosing the next state and is often inefficient for large spaces. It's best suited when the solution space is small or when combined with heuristics to reduce blind search.

### 2. Hill Climbing:

Hill Climbing is a local search algorithm that continuously moves towards the direction of increasing value (or decreasing cost) to find the peak of the solution space. It's simple and memory-efficient but can get stuck in local maxima, plateaus, or ridges, which limits its effectiveness in complex problems.

### 3. Properties of A\* Algorithm:

A\* is a best-first search algorithm that uses a cost function  $f(n) = g(n) + h(n)$  where  $g(n)$  is the path cost from start to node  $n$  and  $h(n)$  is the heuristic estimate from  $n$  to the goal. It is complete, optimal (if  $h$  is admissible and consistent), and efficient, making it widely used in AI pathfinding and graph traversal.

### 4. Best-First Search:

Best-First Search expands the node that appears to be closest to the goal using a heuristic function. It combines features of both BFS and DFS and can be faster than A\*, but it is not guaranteed to be optimal or complete unless specific conditions (like admissible heuristics) are met.

## 5. Problem Reduction:

Problem Reduction solves complex problems by breaking them down into smaller sub-problems (AND/OR tree). It's used in goal-based planning and rule-based reasoning, where solving a large problem involves recursively solving simpler components, allowing more manageable search.

## 6. Constraint Satisfaction Problem (CSP):

A CSP involves variables, domains, and constraints, and the goal is to assign values to variables such that constraints are satisfied. It abstracts many AI problems like scheduling or puzzle solving, where constraints limit the valid combinations of variable assignments.

## 7. Interference in CSPs:

Interference occurs when assigning a value to one variable reduces or eliminates valid options for others due to constraints. This can lead to dead ends, making constraint propagation techniques like forward checking or arc consistency critical for efficient solving.

## 8. Backtracking Search for CSPs:

Backtracking is a depth-first search method where variables are assigned values one at a time. If a constraint is violated, the algorithm backtracks to the previous variable and tries a different value. It's simple but can be optimized with heuristics and constraint propagation.

## 9. Local Search for CSPs:

Local search like Min-Conflicts starts with a complete assignment and iteratively makes changes to reduce conflicts. It's useful for large CSPs where complete backtracking is inefficient. It doesn't guarantee completeness but often finds good enough solutions quickly.

## 10. Structure of CSP Problem:

The structure includes variables, domains, and constraints. Representing CSPs as constraint graphs allows identification of independent sub-problems or tree-structured CSPs, enabling efficient algorithms like tree decomposition and arc-consistency techniques.

---

Let me know when you're ready for **UNIT 3**, or if you want any topic from Unit 2 explained more.



NEXT

---

Great! Here's **UNIT 3 – Knowledge Representation and Reasoning** in viva-style answers:

## UNIT 3 – Knowledge Representation & Reasoning

### 1. Introduction to Knowledge Representation:

Knowledge Representation (KR) is the method of encoding knowledge about the world in a form that an AI system can utilize to solve complex tasks. It includes structured formats like rules, logic, frames, and semantic networks that help an intelligent agent make informed decisions.

### 2. Knowledge Agent:

A knowledge agent is an intelligent system capable of perceiving its environment and making decisions based on its internal knowledge base. It uses reasoning mechanisms over stored knowledge to draw conclusions, learn, and interact meaningfully with its surroundings.

### **3. Predicate Logic, WFF, Inference Rule & Theorem Proving:**

Predicate logic extends propositional logic with quantifiers and predicates, allowing the expression of statements about objects and their relationships. A Well-Formed Formula (WFF) is a syntactically correct logical expression. Theorem proving involves applying inference rules like Modus Ponens to deduce new facts, with strategies like forward and backward chaining.

### **4. Forward & Backward Chaining, Resolution:**

Forward chaining is a data-driven method that starts with known facts and applies inference rules to extract more data until a goal is reached. Backward chaining works in reverse, starting from the goal and breaking it into sub-goals. Resolution is a powerful inference rule used in logic for proving contradictions and validating logical entailments, particularly in automated theorem proving.

### **5. Propositional Knowledge & Boolean Circuit Agents:**

Propositional knowledge deals with facts represented in Boolean variables and logic connectives. Boolean circuit agents use this logic to make decisions based on fixed truth values, useful in basic reasoning and hardware-based logical inference systems.

### **6. Rule-Based Systems:**

Rule-based systems use "IF-THEN" rules to represent knowledge and make decisions. They consist of a knowledge base, inference engine, and working memory. These systems are common in expert systems where human expertise is codified into rules for problem-solving.

### **7. Forward Reasoning: Conflict Resolution:**

In rule-based systems, when multiple rules are triggered, conflict resolution strategies like priority, recency, or specificity determine which rule to apply first. This ensures consistent and goal-directed reasoning.

### **8. Backward Reasoning & Use of Backtracking:**

Backward reasoning attempts to prove a goal by recursively checking if known rules can derive it. If a path fails, backtracking helps explore alternative rules or branches. This method is efficient in goal-driven systems and diagnostic applications.

### **9. Semantic Networks & Frames:**

Semantic networks represent knowledge as a graph of nodes (concepts) and links (relations), using inheritance for shared attributes. Frames are data structures that hold stereotypical information about objects, allowing inheritance, default values, and exceptions. Both support structured and reusable knowledge representation.

### **10. Conceptual Dependency Formalism:**

This formalism represents the meaning of natural language statements using a fixed set of conceptual primitives. It focuses on the underlying meaning rather than syntactic structure, enabling better understanding and inference in NLP-based AI systems.

### **11. Reasoning Under Uncertainty:**

Many real-world problems involve incomplete or ambiguous data. AI handles this with probabilistic models and non-deterministic logic to make reasonable inferences despite uncertainty.

### **12. Probabilistic Reasoning, Bayes Theorem, Bayesian Networks:**

Bayes Theorem updates probabilities based on new evidence, and Bayesian networks graphically represent probabilistic relationships among variables. These tools allow reasoning under uncertainty, especially in diagnostic systems and decision-making.

### 13. Certainty Factor & Dempster-Shafer Theory:

Certainty Factor models belief in expert systems using heuristic values between 0 and 1. Dempster-Shafer Theory is a generalization of probability theory that allows combining evidence from different sources to calculate belief functions and manage uncertainty.

### 14. Non-Monotonic Reasoning & Truth Maintenance Systems:

Non-monotonic reasoning allows AI to withdraw conclusions when new information contradicts previous knowledge, mimicking human reasoning. Truth Maintenance Systems (TMS) help maintain consistency in the knowledge base when beliefs change, by tracking dependencies and justifications.

### 15. Overview of Fuzzy Logic:

Fuzzy Logic deals with reasoning that is approximate rather than fixed and exact. It enables systems to handle partial truth values between 0 and 1, making it ideal for applications like control systems and decision-making in uncertain environments.

---

Ready for **UNIT 4 – NLP**, or want to revise anything from Unit 3?



NEXT

---

Here's **UNIT 4 – Natural Language Processing (NLP)** explained in viva-appropriate format:

## UNIT 4 – Natural Language Processing

### 1. What is NLP & Steps in NLP:

Natural Language Processing (NLP) enables machines to understand, interpret, and generate human language. The main steps include lexical analysis (breaking text into tokens), syntactic analysis (grammar structure), semantic analysis (meaning extraction), discourse integration (context linking), and pragmatic analysis (real-world interpretation).

### 2. Syntactic Analysis – Grammars and Parsers:

Syntactic analysis, or parsing, determines whether a sentence is grammatically correct using formal grammars like Context-Free Grammar (CFG). Parsers like top-down and bottom-up analyze sentence structure to produce parse trees, which represent syntactic hierarchy.

### 3. Augmented Transition Networks (ATN):

ATNs are finite-state machines used for parsing natural language by incorporating procedures and memory into transitions. They are more powerful than regular transition networks, allowing recursive and conditional transitions to handle complex language constructs.

### 4. Unification Grammars:

Unification grammars use feature structures and unification operations to parse sentences. Unlike traditional grammars, they incorporate linguistic attributes (like number, tense) and ensure consistency during parsing, making them useful in computational linguistics.

### 5. Semantic Grammar, Case Grammar, Conceptual Parsing:

Semantic grammar uses rules based on semantic roles rather than syntax, while case grammar assigns roles like agent, action, and object to sentence parts. Conceptual parsing interprets sentences based on underlying concepts and actions, focusing on meaning rather than structure.

## 6. Approximately Compositional Semantic Interpretation:

This approach handles situations where the meaning of a sentence can't be fully derived from its parts due to idioms or ambiguity. It uses statistical or heuristic techniques to approximate overall sentence meaning while considering word combinations and context.

## 7. Discourse and Pragmatic Processing – Using Focus, Beliefs, Goals, Speech Acts:

Discourse processing goes beyond individual sentences to understand connected text. Focus tracking helps resolve pronouns and context, while modeling beliefs and goals aids in understanding intention. Speech acts like requests or assertions add pragmatic meaning based on the speaker's goal.

## 8. Conversational Postulates:

Conversational postulates are assumptions that guide interactions, such as expecting relevant or truthful responses. They help agents interpret indirect language, infer speaker intentions, and maintain coherent dialogues in NLP systems.

## 9. Text Classification (Spell Checking):

Text classification categorizes text into predefined labels using techniques like Naive Bayes or SVM. Spell checking involves identifying and correcting spelling errors using dictionary matching, edit distances, and probabilistic models.

## 10. Probabilistic Language Models:

These models estimate the probability of a word sequence to predict or generate text. N-gram models and advanced ones like Hidden Markov Models (HMM) or neural LMs (e.g., GPT) are used for speech recognition, autocomplete, and translation.

## 11. Implementation Aspects of Syntactic Analysis (Parsing):

Implementation involves designing grammars, choosing parsing strategies (top-down/bottom-up), managing ambiguity, and handling performance. Efficient data structures and algorithms are essential for building fast and robust NLP parsers.

Shall we proceed to **UNIT 5 – Game Playing and Planning**?



YUPP

Here's **UNIT 5 – Game Playing and Planning Systems** in viva-style format:

# UNIT 5 – Game Playing and Planning Systems

## 1. Game Playing – Overview and Examples:

Game playing in AI involves building agents that can make decisions in competitive environments like chess or tic-tac-toe. These problems are modeled as search problems where the AI evaluates possible moves, predicts the opponent's response, and aims to maximize its own chances of winning.

## 2. MiniMax Algorithm:

The MiniMax algorithm is used in two-player zero-sum games where one player's gain is another's loss. It simulates all possible moves, assuming both players play optimally, and chooses the move that minimizes the maximum loss (or maximizes the minimum gain). It's the foundation of decision-making in turn-based games.

### 3. Alpha-Beta Pruning:

Alpha-Beta Pruning enhances MiniMax by cutting off branches in the game tree that don't need to be explored because they cannot affect the final decision. This reduces computation time without affecting the result, making deeper searches more efficient.

### 4. Refinements and Iterative Deepening:

Refinements include techniques like move ordering, heuristic evaluation, and transposition tables to speed up game tree exploration. Iterative Deepening performs MiniMax repeatedly with increasing depth limits, combining depth-first search efficiency with the completeness of breadth-first search.

### 5. The Blocks World:

The Blocks World is a classic AI planning domain where an agent manipulates blocks on a table to achieve a goal configuration. It illustrates the need for action sequencing, handling preconditions, and planning under constraints, often using STRIPS-like representations.

### 6. Components of a Planning System:

A planning system typically consists of:

- A goal (desired final state),
- An initial state,
- A set of actions/operators (with preconditions and effects),
- A planner that generates a sequence of actions (plan) to reach the goal from the initial state.

### 7. Goal Stack Planning:

Goal Stack Planning breaks a complex goal into sub-goals and uses a stack to keep track of what needs to be achieved. It works backward from the goal, pushing and popping tasks while satisfying their preconditions recursively.

### 8. Nonlinear Planning using Constraint Posting:

Unlike linear planning where actions are strictly ordered, nonlinear planning allows partially ordered plans. Constraint posting adds temporal or causal constraints during planning, and plans are refined iteratively, allowing more flexible and efficient solutions.

### 9. Hierarchical Planning:

Hierarchical planning decomposes high-level goals into smaller sub-tasks or operators. It uses domain-specific knowledge in the form of methods or schemas to simplify complex planning problems and supports abstraction in planning.

### 10. Reactive Systems:

Reactive systems respond to changes in real-time environments without precomputed plans. They rely on condition-action rules and are used in scenarios where the environment is dynamic and unpredictable, like robotic navigation or real-time strategy games.

---

Ready for the **final UNIT 6 – Deep Learning & Applications?**



YOSS

---

Here's **UNIT 6 – Deep Learning and Applications** explained in a viva-ready, concise yet detailed format:



## UNIT 6 – Deep Learning and Applications

### 1. Introduction to Deep Learning & Why Go Deep:

Deep Learning is a subset of machine learning that uses neural networks with many layers (deep architectures) to automatically learn features from raw data. Going deeper enables the model to capture more abstract and hierarchical representations, making it powerful for tasks like image and speech recognition.

### 2. Architecture of Deep Network:

A deep neural network consists of an input layer, multiple hidden layers, and an output layer. Each layer applies transformations using weights and activation functions like ReLU or sigmoid. The depth (number of layers) allows for complex feature extraction and better generalization if trained properly.

### 3. Restricted Boltzmann Machines (RBM):

RBMs are unsupervised, energy-based neural networks with two layers – a visible layer and a hidden layer. They learn probability distributions over inputs and are often used for feature extraction or as building blocks in deep belief networks.

### 4. Deep Belief Network (DBN):

A DBN is a stack of RBMs where the output of one is the input for the next. They are pre-trained layer by layer in an unsupervised fashion, then fine-tuned with supervised learning. DBNs are effective for dimensionality reduction and generative modeling.

### 5. TensorFlow & Deep Learning Libraries:

TensorFlow is a popular open-source framework for building and training deep learning models. It supports GPU acceleration and automatic differentiation. Other libraries include PyTorch, Keras (user-friendly front-end for TensorFlow), Caffe (mainly for vision), and Theano (now discontinued but historically important).

### 6. Deep Learning Platforms – Theano, Caffe, TensorFlow:

- **Theano:** Early symbolic computation library; now outdated.
- **Caffe:** Known for speed and efficiency in image-based models.
- **TensorFlow:** Widely adopted, supports large-scale deployments, and flexible model building. Each platform provides tools for model training, deployment, and optimization.

### 7. Deep Learning Use Cases:

Applications span multiple fields:

- **Computer Vision:** Image recognition, face detection, object tracking.
  - **Speech and Text:** Voice assistants, translation, sentiment analysis.
  - **Healthcare:** Disease prediction, medical imaging.
  - **Finance & Business:** Fraud detection, demand forecasting.
- It is a backbone for AI innovations across industries.

### 8. Overview of AI Domains – Robotics, Neural Networks, AI-IoT, Computer Vision:

- **AI-Robotics:** Combines AI with mechanical systems for autonomous agents.
- **AI-Neural Networks:** Core of deep learning for recognizing patterns.



- **AI-IoT:** Smart devices that sense and act using AI logic (e.g., smart homes).
- **Computer Vision:** Enables machines to interpret and understand visual information.

### 9. Case Study – Automatic Bird Identification using Deep Learning:

A CNN model is trained on labeled bird images to recognize species. Features like color, beak shape, and wing patterns are automatically learned by the network, enabling accurate classification of bird species from new images.

### 10. Case Study – Tumkur Monitoring using Computer Vision:

AI-powered cameras and computer vision techniques are used to monitor Tumkur city traffic or public spaces. Models detect violations, track vehicle movements, and provide analytics to support urban planning and law enforcement.

### 11. Case Study – Text to Speech Conversion using APIs:

Text-to-speech (TTS) systems convert written text into natural-sounding speech. APIs like Google Text-to-Speech or Amazon Polly use deep learning for pronunciation, intonation, and language modeling, enabling realistic voice assistants and accessibility tools.

That wraps up **Unit 6** and your entire AI viva guide!

Would you like a combined PDF of these viva-style answers for all units?



YOSS

Your viva-ready PDF for **AI Units 2 to 6** is ready. You can download it using the link below:



Let me know if you want a version for Units 1 to 3 or any formatting changes!



Give me pdf again

Sure! You can download the PDF for **AI Viva Units 2 to 6** from the link below:

Let me know if you need it split by unit or formatted differently.