# AI UNIT - 3 CONCISE

## UNIT 3: Learning, Game Playing, and NLP

### Part 1: Learning in Artificial Intelligence `[PYQ]`

#### 1. What is Learning?

- **Core Definition:** Learning denotes changes in the system that enable it to do the same task more efficiently next time.
- **Alternative Definitions:**
  - Constructing or modifying representations of what is being experienced.
  - Making useful changes in minds (or system's knowledge/parameters).
- **Goals/Benefits:**
  - Improves understanding and efficiency.
  - Enables discovery of new, previously unknown things/structures (e.g., data mining).
  - Allows filling in incomplete observations/specifications about a domain (expands expertise, lessens brittleness).
  - Facilitates building adaptive software agents.
  - Reproduces an important aspect of intelligent behavior.

#### 2. What Characterizes a Learning System?

- **Iterative Process:**
  1. Produce a result.
  2. Evaluate against an expected result (if available).
  3. Tweak the system based on evaluation.
- **Discovery:** Can discover patterns without prior expected results (unsupervised learning).
- **Transparency Types:**
  - **Open Box:** System changes (e.g., in KB) are clearly visible and interpretable by humans.
  - **Black Box:** System changes are not readily visible or understandable.

#### 3. What is the Architecture of a Learning Agent/System? `[PYQ]` `[FIG]`

- **(See Unit 1, Learning Agent for diagram: Performance Element, Critic, Learning Element, Problem Generator, interacting with Environment via Sensors/Actuators, and a Knowledge Base/Generalizer).**
- **Main Components:**
  - **Knowledge Base (KB):** Stores what is being learned, domain representation, problem space.

- **Performer:** Does something with the KB to produce results.
- **Critic:** Evaluates results against expected results/performance standards.
- **Learner:** Takes feedback from critic; modifies KB or performer.

- **Optional Component:**
  - **Problem Generator:** May generate test cases to evaluate performance.

### 4. Examples of Learning Systems `[FIG]` (Table format)

- **Animal Guessing Game:** Binary decision tree (Representation); Walk tree, ask questions (Performer); Human feedback (Critic); Elicit & add question (Learner).
- **Playing Chess:** Board layout, rules, moves (Representation); Chain rules, identify move (Performer); Who won (credit assignment) (Critic); Adjust rule weights (Learner).
- **Categorizing Documents:** Vector of word frequencies (Representation); Apply functions to categorize (Performer); Human-categorized documents (Critic); Modify function weights (Learner).
- **Fixing Computers:** Frequency matrix of causes/symptoms (Representation); Use symptoms to ID causes (Performer); Human input on symptoms/cause (Critic); Update frequency matrix (Learner).
- **Identifying Digits (OCR):** Probability of digits, pixel matrix (Representation); Input features, output probability (Performer); Human-categorized training set (Critic); Modify association weights (Learner).

### 5. Different Learning Paradigms `[PYQ]`

- **Rote learning:** Direct entry of rules/facts; Memorization.
- **Learning by taking advice:** Human/system interaction; Advice operationalization. `[PYQ]`
- **Learning in problem solving:** Parameter adjustments, learning macro-operators, chunking.
- **Learning from examples (Induction):** Using specific examples to reach general conclusions. `[PYQ]`
- **Explanation-based learning (EBL):** Learn from one example, then generalize; knowledge-intensive. `[PYQ]`
- **Learning through discovery:** Unsupervised; specific goal not given; finding patterns.
- **Learning through analogy:** Determining correspondence between different representations; case-based reasoning. `[PYQ]`
- **Formal learning theory:** Mathematical model of learning (e.g., PAC learning).
- **Neural net learning & genetic learning:** Evolutionary search, biologically inspired network learning. `[PYQ]`

### 6. How does Rote Learning work?

- **Definition:** Basic learning activity; memorization.
- **Mechanism:** Knowledge copied into KB without modification. Direct entry of rules/facts.
- **Application:** Developing ontologies; Data caching for performance.
- **Benefit:** Saves re-computation time by storing computed values.

- **Key Capabilities:**
  - **Organized storage:** For fast retrieval.
  - **Generalization:** To keep stored objects manageable for large state spaces.

### 7. How does Learning by Taking Advice work? `[PYQ]`

- **Simplicity:** Easiest way of learning.
- **Process:**
  1. Expert/programmer writes instructions/advice.
  2. System integrates/learns advice.
  3. System can do new things based on advice.
- **Inference Requirement:** More inference than rote learning.
- **Operationalization:** `[PYQ]`
  - Stored KB knowledge transformed into an operational form.
  - Program turns advice into usable expressions (concepts, actions). Critical ability.
- **Consideration:** Reliability of knowledge source.

### 8. How does Learning Occur in Problem Solving?

- **Context:** Program learns by generalizing from its *own experiences*.
- **Types Discussed:**
  - **a. Learning by Parameter Adjustment:** `[PYQ]`
    - **Scenario:** System uses an evaluation procedure combining features into a score (e.g., polynomial: Score = $\Sigma$ ci * ti).
    - **Challenge:** Knowing a priori weights (ci) for features (ti).
    - **Process:** Start with estimates, modify weights based on experience. Increase weights for good predictors, decrease for poor.
    - **Key Questions:** When/how much to change coefficients? **Credit Assignment Problem** (assigning responsibility for outcome). `[PYQ]`
    - **Method:** Hill-climbing search.
  - **b. Learning with Macro-Operators (MACROPs):** `[PYQ]`
    - **Definition:** Sequence of actions (operators) treated as a whole.
    - **Process:** Solve problem, store computed plan (sequence of actions) as a single MACROP with preconditions (initial conditions) and postconditions (goal achieved).
    - **Benefit:** Efficiently uses past experience. Critical for non-serializable subgoals. Allows domain-specific learning.
    - **Generalization:** Replacing constants with variables allows MACROPs for similar problems.
    - **Example:** STRIPS planning.

- **c. Learning by Chunking:** `[PYQ]`
  - **Similarity:** Similar to macro-operators. Used in production systems.
  - **Chunking Process:** When a useful sequence of rule firings occurs (solves sub-problem/impasse), store it as a single new rule ("chunk").
  - **Benefit:** Captures search control knowledge. Reduces problem-solving steps.
  - **Example:** SOAR architecture learns via chunking when impasses are resolved.
- **d. The Utility Problem in Learning:** `[PYQ]`
  - **Definition:** Knowledge learned to *improve* performance *degrades* it instead.
  - **Context:** Common in speedup learning systems (learning control rules). Systems slow down if they learn too much unrestrainedly.
  - **Paradox:** Individual rules may be positive, but collectively negative.
  - **Solutions:**
    - **Hardware:** Parallel memory systems ("active memories").
    - **Utility Measurement (e.g., PRODIGY):** Maintain utility for each rule (savings, frequency, match cost). Discard negative/low utility rules.

### 9. How does Learning by Analogy work? `[PYQ]`

- **Definition:** Acquiring new knowledge about an input entity by transferring it from a known similar entity.
- **Central Intuition:** If two entities are similar in some respects, they could be similar in others.
- **Types:**
  - **Transformational Analogy:** `[FIG]` (Slide 23)
    - Find similar *solution*. Copy it, make suitable substitutions.
    - Focuses on final solution, not derivation steps.
  - **Derivational Analogy:** `[FIG]` (Slides 24-25)
    - Finds problems sharing aspects based on similarity metric.
    - Retrieves *derivation (steps)* of previous solution.
    - Perturbs old derivation incrementally for new problem. Considers *how* problem was solved.

### 10. What is Explanation-Based Learning (EBL)? `[PYQ]`

- **Definition:** Learning from a single example by:
  1. **Explaining:** Using existing domain knowledge (theory) to explain *why* the training example is an instance of the target concept.
  2. **Generalizing:** Turning the explanation into a more general rule/concept definition.
- **Approach:** Analytical, knowledge-intensive (requires good domain theory).
- **(Slides 26-31 likely show EBL process: explanation proof tree, generalization, new rule formation).** `[FIG]`

### 11. How does Learning by Discovery work? (Unsupervised)

- **Definition:** Acquiring knowledge without a teacher.
- **Types Discussed:**
    - **a. Theory-Driven Discovery (e.g., AM program - 1976):**
        - **Goal:** Discovers concepts in elementary math/set theory.
        - **How it Works:** Uses frame-based representation, heuristic search (~250 heuristics), Hypothesis & Test, agenda control.
        - **Discoveries:** Integers, Addition, Multiplication, Prime Numbers, Goldbach's Conjecture.
    - **b. Data-Driven Discovery (e.g., BACON program - 1981):**
        - **Context:** Makes sense of empirical data.
        - **How it Works:** Starts with variables, inputs experimental data, holds some constant, notices trends, infers mathematical laws.
        - **Discoveries:** Ideal gas law, Kepler's 3rd law, Ohm's law.
    - **c. Clustering:** `[PYQ]`
        - **Definition:** Grouping data into new classes/clusters. Similar objects in same cluster, dissimilar in others.
        - **Goal:** Construct meaningful partitioning; maximize intra-class similarity, minimize inter-class.
        - **Process:** Given feature vectors, find partition into 'c' subsets. Discover labels automatically.
        - **AutoClass (Example):** Bayesian approach for optimal classes. Class membership is probabilistic.

### 12. What is Formal Learning Theory?

- **Focus:** Provides a formal mathematical model of learning; analyzes learnability.
- **Example: Theory of the Learnable (Valiant - leading to PAC Learning):** `[PYQ]`
    - **Probably Approximately Correct (PAC) Learning:**
        - A device learns a concept if, given positive/negative examples, it produces a hypothesis `h` that classifies future examples correctly with probability `1 - ε` (error tolerance).
    - **Complexity Factors:** Error tolerance (ε), number of features (t), complexity/size of hypothesis (f).
    - **Trainability:** Concept class is efficiently learnable if training examples needed is polynomial in (1/ε, t, f).
- **Goal:** Quantify knowledge use in learning mathematically.

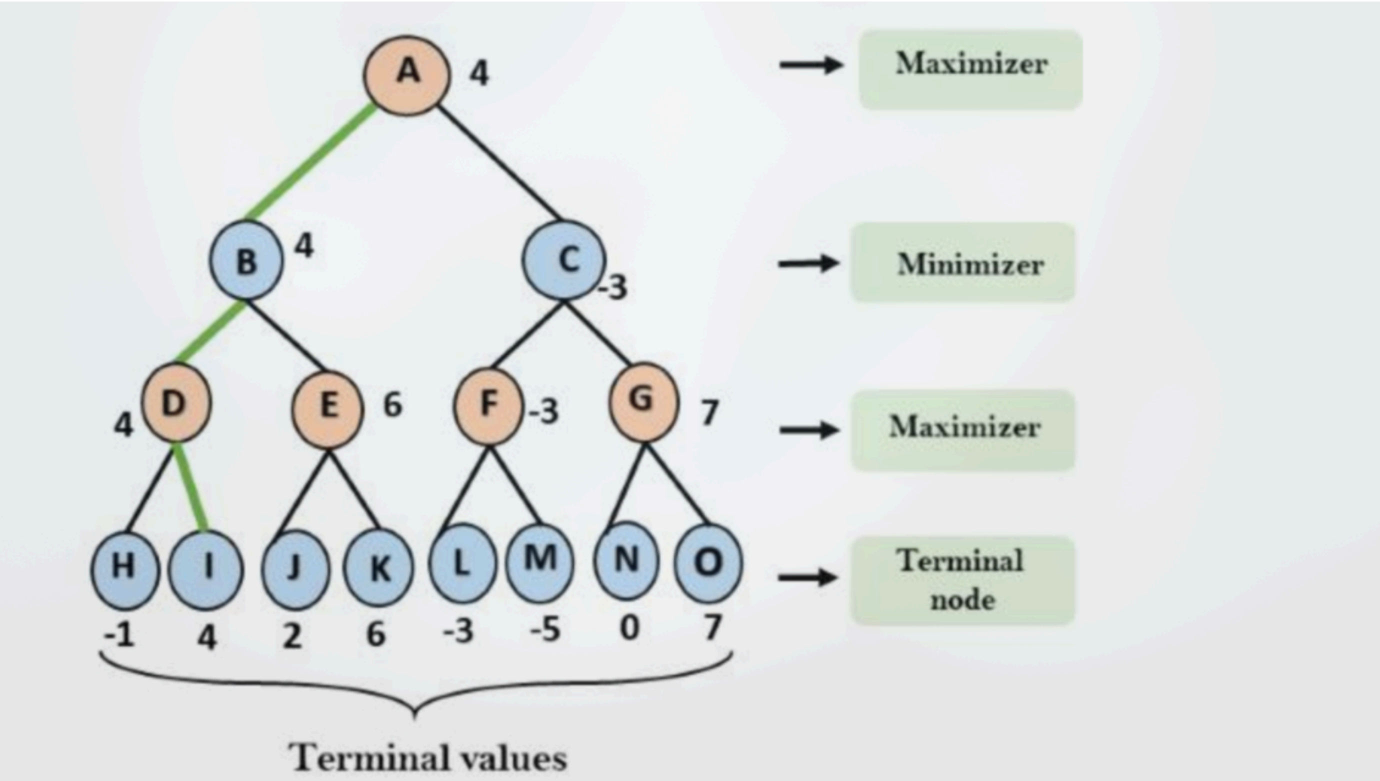
### 13. Neural Net Learning and Genetic Learning `[PYQ]`

- **A. Neural Net Learning (Artificial Neural Networks - ANNs):** `[PYQ]` `[FIG]`
    - Biologically inspired by brain structure.

- Networks of interconnected processing units (artificial neurons).
  - Learning by adjusting connection strengths (weights) based on training data and error (e.g., backpropagation).
  - Architectures: MLPs, CNNs, RNNs, Transformers.
- **B. Genetic Learning (Genetic Algorithms - GAs):** `[PYQ]`
  - Inspired by biological evolution ("survival of the fittest").
  - Evolutionary search technique.
  - Maintains a population of candidate solutions.
  - Uses operators: fitness evaluation, selection, crossover (recombination), mutation to evolve better solutions.

## Part 2: Game Playing in AI `[PYQ]`

### 14. What is the Minimax Algorithm? `[PYQ]` `[FIG]`




- **Type:** Backtracking algorithm for decision making in game theory/AI.
- **Application:** Two-player, zero-sum, perfect information games (Tic-Tac-Toe, Chess).
- **Goal:** Find optimal move for MAX player, assuming MIN opponent also plays optimally.
- **Players:**
  - **Maximizer (MAX):** Tries to get highest score (max benefit).
  - **Minimizer (MIN):** Tries to get lowest score (min benefit for MAX).
- **Evaluation:** Each game state (node) assigned an evaluation score. Positive favors MAX, negative favors MIN.

### 15. How does the Minimax Algorithm Work? `[PYQ]` `[FIG]`

1. **Generate Game Tree:** Possible moves and resulting states to a certain depth or terminal states.
2. **Apply Utility Function:** Assign scores (utility values) to terminal (leaf) nodes.
3. **Backtrack and Propagate Values:** (DFS traversal)
   - At **Terminal Nodes:** Use utility value.
   - At **MAX Nodes:** Choose MAXIMUM value from children.
   - At **MIN Nodes:** Choose MINIMUM value from children.
   - Continue to root. Root value is best MAX score; move leading to it is optimal.

- **(Example workflow from notes p.10, visualized in p.26-28, and handwritten notes p.35-38).**

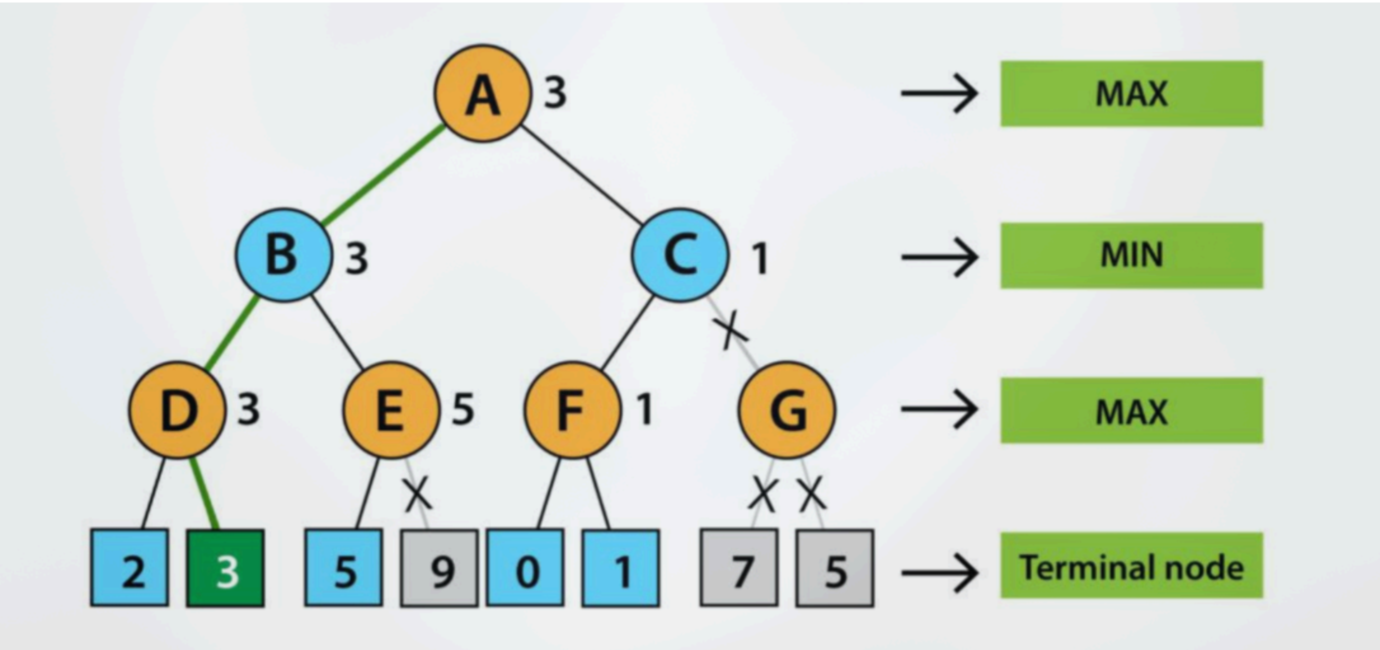
### 16. Properties of Minimax `[PYQ]`

- **Completeness:** Yes, if game tree is finite.
- **Optimality:** Yes, if both players play optimally.
- **Time Complexity:** O(b^m) (b=branching factor, m=max depth). Exponential.
- **Space Complexity:** O(b*m) (for DFS, stores path).

### 17. Limitation of the Minimax Algorithm

- **Slow Performance:** For complex games (Chess, Go) with large b and m, due to exponential time. Explores entire tree to depth 'm'.

### 18. What is Alpha-Beta Pruning? `[PYQ]` `[FIG]`




- **Definition:** Modified Minimax; an optimization technique.
- **Goal:** Reduce nodes examined by Minimax, returning same optimal move. Prunes branches that can't influence final decision.
- **Mechanism:** Uses two threshold parameters:

- **Alpha (α):** Best (highest-value) choice found *so far* for MAX. Initial: -∞.
- **Beta (β):** Best (lowest-value) choice found *so far* for MIN. Initial: +∞.

- **Pruning Condition:** Pruning occurs when **α ≥ β**. `[PYQ]`
- **Scope:** Applicable at any depth; can prune single leaves or entire sub-trees.

### 19. Key Points and Working of Alpha-Beta Pruning `[PYQ]` `[FIG]`

- **Key Points:**
  - MAX player only updates α.
  - MIN player only updates β.
  - Node's *computed value* (min/max of children) passed upwards, not α/β values.
  - α and β values passed *down* to child nodes during exploration.
- **(Working Example from notes p.12, visualized in p.30-34).**
  1. Start at Root (A): α = -∞, β = +∞.
  2. Explore D (MAX children): α updated (e.g., to 3). D returns 3.
  3. At B (MIN parent of D): β updated to min(+∞, 3) = 3. Now B has α=-∞, β=3.
  4. Explore E (MAX sibling of D, under B): α passed as -∞, β as 3.
  5. At E: if first child's value makes E's α (say 5) ≥ B's β (3) -> **PRUNE** remaining children of E.
  6. And so on...

### 20. How does Move Ordering Affect Alpha-Beta Pruning? `[PYQ]`

- **High Dependence:** Pruning effectiveness highly depends on node examination order.
- **Worst Ordering:** Best moves examined last. Minimal/no pruning. Time ≈ Minimax O(b^m).
- **Ideal Ordering:** Best move always examined first. Maximum pruning. Time ≈ O(b^(m/2)).
- **Rules for Good Ordering:**
  - Explore best move from shallowest node first.
  - Use heuristics to order nodes (potentially best ones first).
  - Domain knowledge (e.g., Chess: captures > threats > forward moves).
  - Iterative deepening, transposition tables.

## Part 3: Natural Language Processing (NLP) `[PYQ]`

### 21. What is Natural Language Processing (NLP)? `[PYQ]`

- **Definition:** Branch of CS/AI focused on enabling computers to communicate with people using everyday, natural language.
- **Related Field:** Computational Linguistics.
- **Two Goals:**
  - **Science Goal:** Understand how language operates.

    - **Engineering Goal:** Build systems to analyze/generate language; reduce man-machine gap.
- **Two Views:** Classical (Symbolic, Rule-based) vs. Statistical/ML (Data-driven).

### 22. Core Areas/Levels of NLP (NLP Trinity/Stages) `[PYQ]` `[FIG]`

- **(Diagram likely shows increasing complexity from Morphology to Discourse).**
- **Morphological Analysis:** Word structure (morphemes).
- **Part-of-Speech (POS) Tagging:** Assigning word categories (noun, verb). `[PYQ]`
- **Chunking/Shallow Parsing:** Identifying basic phrases (Noun Phrases, Verb Phrases).
- **Parsing/Syntactic Analysis:** Determining full sentence structure (grammar). `[PYQ]`
- **Semantics:** Extracting meaning (word sense, sentence meaning, semantic roles). `[PYQ]`
- **Discourse & Coreference:** Analyzing relationships between sentences, resolving pronouns.
- **Algorithms/Models:** HMM, MEMM, CRF, RNN.

### 23. Why is NLP/Language Technology Relevant in the Indian Context?

- **Linguistic Diversity:** Highly multilingual (22 official, 122 major languages).
- **Digital Divide:** Only ~20% understand English; 80% potentially excluded if content not in local languages.
- **Internet/Social Media Growth:** Necessitates NLP for processing Indian language content.
- **Goal:** Effective communication, inclusive digital society, citizen flexibility.

### 24. Indian Government Initiatives in NLP (e.g., TDIL)

- **TDIL Programme (Technology Development for Indian Languages - MeiTY):**
    - **Objective:** Develop tools/techniques for human-machine interaction without language barriers; create/access multilingual resources.
- **Major Initiatives:**
    - **Machine Translation (MT):** Anuvadaksh (English to Indian Langs), Angla-Bharti (English to Indian Langs), Sampark (Indian Lang to Indian Lang).
    - **Cross-Lingual Information Access (CLIA):** For major Indian languages.
    - **OCR (Robust Document Analysis & Recognition):** For 14 languages.
    - **Text-to-Speech (TTS):** In Indian Languages.
    - **Automatic Speech Recognition (ASR):** In Indian Languages.
    - **Domain-Specific MT:** E.g., Hindi to English (Judicial Domain).

### 25. NLP in Governance (Case Study: MyGov.in Portal)

- **MyGov.in Portal:** Citizen-centric platform for participatory governance.
- **NLP Challenge:** Manually mining relevant info from huge user post volume is infeasible.
- **Code-Mixing Issue:** Users mix languages (e.g., Hindi + English).

- **Need for NLP/ML:** To analyze feedback, understand public opinion (Sentiment Analysis), handle code-mixed data.

### 26. Why Analyse Public Opinion using NLP?

- **Importance:** Public opinions aid betterment of human lives.
- **Opportunity:** User-generated content offers new ways to understand social behavior.
- **Benefit for Governance:** Helps anticipate social changes, adapt to population needs.
- **Relevant Field:** Opinion Mining / Sentiment Analysis. `[PYQ]`

### 27. Projected Growth and Evolution of NLP

- **Growth:** Exponential; e.g., $16 billion market by 2021 (~16% CAGR).
- **Reasons for Growth:** Rise of Chatbots, need for customer insights, automation (messaging, translation, speech tasks).
- **Major Players:** Amazon, Google, Microsoft, Facebook, IBM.
- **Evolution:** Human-computer interaction → human-computer *conversation*.
- **Critical Advancements:** Biometrics, Humanoid Robotics.

### 28. How can NLP be used specifically in Governance?

- **Goal:** Improve service delivery, decrease citizen-government interaction gap.
- **Uses on Government Websites:**
  - Making e-governance info available in multiple languages (MT).
  - Natural Language Generation (NLG) for reports, summaries.
  - Chatbots for information access/service requests (multilingual support).

### 29. NLP in Business and Healthcare

- **Business:**
  - **Sentiment Analysis:** Public/customer opinion on products/services.
  - **Email Filters:** Spam filtering, categorization.
  - **Voice Recognition:** Smart voice-driven interfaces/services.
  - **Information Extraction:** Key data from documents.
- **Healthcare:**
  - **Improved EHR Experience:** Analyzing EHRs, clinical notes. Voice-support, predictive analytics.
  - **Reduced Communication Gap:** Patient interaction in own language (portals).
  - **Improved Quality of Care:** Calculating inpatient care measures, monitoring guidelines.
  - **Patient Identification:** Identifying patients needing improved care coordination.

### 30. NLP in Finance and Other Domains

- **Finance:**

- **Credit Scoring:** ML/NLP to assess creditworthiness.
- **Document Search/Analysis:** Automating document processing.
- **Fraud Detection:** Analyzing transactions/communications for fraud.
- **Stock Market Prediction:** Sentiment analysis on news/social media.

- **Other Domains:**
  - **National Security:** Sentiment analysis in cross-border languages, hate speech/radicalization detection.
  - **Recruitment:** Searching/screening applications/resumes.

### 31. Perspectives and Allied Disciplines of NLP `[FIG]`

- **AI Inter-dependencies:** NLP interacts with Search, Logic, ML, Vision, KR, Planning, Robotics, Expert Systems.
- **Allied Disciplines:** Philosophy, Linguistics, Probability & Statistics, Cognitive Science, Psychology, Brain Science, Physics (Info Theory), CS & Engg.

### 32. What is the Turing Test? (NLP Context) `[PYQ]` `[FIG]`

- (Reiteration from Unit 1) Test of machine's ability to exhibit human-equivalent intelligent behavior via natural language conversation.

### 33. Natural Languages vs. Computer Languages

- **Ambiguity:** Primary difference. Natural languages are inherently ambiguous at multiple levels.
- **Formal Languages (Programming):** Designed to be unambiguous. Defined by grammar for unique parse. Efficient, deterministic parsing.

### 34. Stages of NLP Processing and Ambiguity Examples `[PYQ]` `[FIG]`

- **(NLP architecture involves stages, each with ambiguity).**
- **Phonetics & Phonology (Speech Processing):**
  - Challenges: Homophones (bank/bank), word boundary segmentation (I got [ua]plate), disfluencies.
- **Morphological Analysis (Word Structure):** `[PYQ]`
  - Definition: Study of internal word structure. Morpheme = smallest meaningful unit.
  - Task: Segmenting words (carried → carry + ed).
  - Challenge: Ambiguity (unlockable → un+lockable OR unlock+able?).
- **Lexical Analysis (Dictionary Lookup):** `[PYQ]`
  - Task: Access dictionary, get word properties (POS, semantic features).
  - Challenge: Lexical Disambiguation (POS: dog noun/verb; WSD: dog animal/person). Neologisms.
- **Syntactic Analysis (Sentence Structure / Parsing):** `[PYQ]` `[FIG]`

- Task: Detect structure (e.g., S → NP VP).
    - Challenge: Structural Ambiguity (Scope: "old men and women"; PP Attachment: "I saw the boy with a telescope"). Garden Path Sentences.
- **Semantic Analysis (Meaning Representation):** `[PYQ]`
    - Task: Represent meaning (Predicate Calculus, Semantic Nets, Frames). Identify semantic roles.
    - Challenge: Semantic Role Labeling (SRL) Ambiguity ("Visiting aunts can be a nuisance"). WSD ("strong interest" vs "pay interest").
- **Pragmatics (Contextual Meaning / User Intent):** `[PYQ]`
    - Task: Model user intention, understand meaning beyond literal. Requires world knowledge.
    - Challenge: Very hard. (Tourist asking about sandals implies wanting them).
- **Discourse (Multi-Sentence Analysis):**
    - Task: Processing sentence sequences, understanding inter-sentence relationships.
    - Challenge: Anaphora / Co-reference Resolution ("John put carrot on plate and ate *it*.").

### 35. Specific NLP Tasks `[PYQ]`

- **Word Segmentation:** Breaking character strings into words (e.g., for Chinese).
- **Part-of-Speech (POS) Tagging:** Annotating words with POS tags. `[PYQ]`
- **Phrase Chunking:** Finding non-recursive NPs, VPs.
- **Parsing:** Full syntactic analysis, generating parse tree. `[PYQ]`
- **Word Sense Disambiguation (WSD):** Determining correct meaning of ambiguous words. `[PYQ]`
- **Semantic Role Labeling (SRL):** Identifying semantic roles of phrases relative to verbs.
- **Textual Entailment:** Determining if one sentence logically implies another.
- **Anaphora/Co-reference Resolution:** Identifying phrases referring to the same entity.
- **Information Extraction (IE):** `[PYQ]`
    - **Named Entity Recognition (NER):** Identifying names (people, places, orgs). `[PYQ]`
    - **Relation Extraction:** Identifying relations between entities.
- **Question Answering (QA):** Answering natural language questions from text. `[PYQ]`
- **Text Summarization:** Producing short summaries. `[PYQ]`
- **Sentiment Analysis:** Extracting subjective info, polarity. `[PYQ]`
- **Machine Translation (MT):** Translating between languages. `[PYQ]`

### 36. Why is Ambiguity Resolution Hard and What Knowledge is Needed? `[PYQ]`

- **Requirement:** Correct interpretation needs combining knowledge from multiple levels:
    - **Syntax:** Grammatical structure.
    - **Semantics:** Word meanings.
    - **Pragmatics:** Contextual understanding.

    - **World Knowledge:** Common sense facts.

### 37. Approach to Acquiring Knowledge for NLP: Traditional vs. Modern

- **Traditional / Rationalist Approach:**
    - Manual knowledge acquisition. Human specialists specified rules (lexicons, grammars).
    - Problems: Difficult, time-consuming, error-prone, brittle, expensive.
- **Modern / Empirical / Statistical / Learning Approach:** `[FIG]`
    - Uses ML to *automatically acquire* knowledge from data (annotated text corpora).
    - Process: Labeled data → ML Algorithm → NLP System.
    - Benefits: More robust, adaptable, handles variability better. Dominant since 1990s.

### 38. Key Milestones in NLP History `[PYQ]`

- **1950s:** Shannon (probabilistic models), Chomsky (formal grammars), first parsers, Bayesian OCR.
- **1960s:** MIT AI Lab (BASEBALL, ELIZA), semantic nets, Brown Corpus.
- **1970s:** Deeper understanding systems (SHRDLU), Prolog for parsing, early HMMs for speech.
- **1980s:** More complex grammars (unification, TAG), symbolic discourse, initial statistical POS tagging.
- **1990s: "Statistical Revolution":** Rise of empirical methods, annotated corpora (Penn Treebank), statistical MT, robust parsers, IE systems.
- **2000s:** Diverse ML methods (SVMs, MaxEnt, CRFs), shared tasks/corpora, unsupervised/semi-supervised focus, shift to semantics (WSD, SRL).
- **2010s-Present: Deep Learning dominates:** Word2vec, GloVe, CNNs, RNNs, Transformers (BERT), end-to-end learning, large pre-trained models, XAI. **Turing Award 2018** for Deep Learning (Bengio, Hinton, LeCun).

### 39. Types of Machine Learning Relevant to NLP `[PYQ]`

- **Machine Learning:** Acquiring models from data/experience. Learning parameters, structure, hidden concepts.
- **Types:**
    - **Unsupervised Learning:** No teacher feedback; detect patterns (e.g., clustering). `[PYQ]`
    - **Reinforcement Learning:** Feedback via rewards/punishments. `[PYQ]`
    - **Supervised Learning:** Given examples of correct input-output pairs (labeled data). `[PYQ]`
        - **Classification:** Discrete outputs (e.g., spam/ham, topic, sentiment). `[PYQ]`
        - **Regression:** Continuous outputs.

### 40. How does Supervised Learning Work (e.g., Classification)?

- **Goal:** Given training set (x1, y1)...(xn, yn) where yi=f(xi) for unknown f, discover hypothesis h ≈ f.
- **Process:**

1. **Data:** Collect labeled instances. Split: Training, Validation (Held-Out), Test sets.
2. **Features:** Define attributes characterizing input `x`.
3. **Experimentation Cycle:**
    - Learn model parameters on Training set.
    - Tune hyperparameters on Validation set.
    - Evaluate final performance on Test set.
    - **Crucial:** Never "peek" at Test set during training/tuning.

- **Evaluation:** Accuracy, Precision, Recall, F1-score.
- **Challenge: Overfitting:** Fitting training data too closely, poor generalization.

### 41. How is Text Classification used in NLP? `[PYQ]`

- **Task:** Assigning predefined labels/categories to documents.
- **Examples:** Topic Labeling, Genre Classification, Opinion/Sentiment Analysis, Spam Detection.
- **Methods History:** Manual Classification, Hand-coded Rules, **Supervised Machine Learning** (k-NN, Naive Bayes, SVMs, Deep Learning - widely used).

### 42. Why has Deep Learning become prominent in NLP? `[PYQ]`

- Transforms how machines understand/interact with complex data. Mimics brain's neural networks.
- **Limitations of Shallow ML:** Required hand-crafted features, suffered from curse of dimensionality.
- **Deep Learning (DL) Advantages:**
    - Learns representations (features) *automatically* from data.
    - Effective at complex patterns via multiple layers.
    - Flexible, (almost) universal framework. Can learn supervised/unsupervised.
    - Effective end-to-end learning. Can leverage large training data.
    - Outperformed other ML ~2010 (Speech, Vision, then NLP).
- **Key DL Developments in NLP:** Distributed Representations (Word2vec, ELMo, BERT), Neural Architectures (CNNs, RNNs/LSTMs, Transformers), RL apps, Attention.
- **Impact:** Led to state-of-the-art results in many NLP tasks.

### 43. What is Stemming? `[PYQ]`

- NLP process to reduce inflected/derived words to their word stem, base, or root form.
- Stem isn't necessarily a linguistically correct root; often a truncated version.
- **Examples:** "running" → "run"; "studies" → "studi"; "beautiful" → "beauti".

### Other Related NLP Preprocessing Processes: `[PYQ]`

- **1. Tokenization:** Breaking text into smaller units (tokens - words, punctuation). `[PYQ]`
- **2. Lowercasing:** Converting all text to lowercase.
- **3. Stop Word Removal:** Removing common, low-meaning words ("a", "the", "is"). `[PYQ]`

- **4. Punctuation Removal/Handling:** Removing/replacing punctuation.
- **5. Lemmatization:** `[PYQ]`
  - Reduces words to base/dictionary form (lemma). Unlike stemming, considers meaning & POS.
  - More accurate than stemming; produces actual dictionary words.
  - Slower, more computationally intensive than stemming.
  - Examples: "running" → "run"; "studies" → "study"; "better" → "good".
- **6. Part-of-Speech (POS) Tagging:** (Covered earlier) Assigning grammatical category. `[PYQ]`
- **7. Named Entity Recognition (NER):** (Covered earlier) Identifying/categorizing named entities. `[PYQ]`
- **8. Sentence Segmentation (Sentence Boundary Disambiguation):** Dividing text into sentences.