CS Exit Ethiopia

# Complete Study Notes

Computer Science Exit Examination

1 selected course | 18 November 2026

**Selected courses**

01. Introduction to Artificial Intelligence

https://csexitethiopia.com

abelasnake2003@gmail.com | Telegram: @Abechu7

# Course contents

Each course begins on a new page. PDF bookmarks provide question, study-note section and revision-topic navigation.

Introduction to Artificial Intelligence 6

# How to use these study notes

## START01 A practical study cycle

Start with a small set of questions before reading.

Mark the exact thing that stopped you: an unknown term, a rule, a calculation, a trace or a choice between alternatives.

Use the question lookup to open the relevant section.

Read one section, close it, and explain the rule in your own words.

Solve the question again and say why each other choice fails.

Then change one assumption or input and solve the variation.

This checks understanding beyond recognition.

A useful 60-minute session: 10 minutes of cold questions, 20 minutes of focused reading, 20 minutes of closed-book solving and 10 minutes updating an error log.

Adjust the timing to your needs.

Revisit mistakes after a gap, for example the next day, several days later and the following week.

These intervals are a practical starting plan, not a fixed learning law.

Record the section ID, your mistaken rule, the corrected rule and one boundary case.

Keep the log short enough to review.

Mix related courses during revision: programming with data structures; databases with security; networking with administration.

Explain how the same word changes meaning across contexts.

## START02 What to emphasize

Emphasize definitions, key differences, algorithm costs, SDLC models, databases, networking and security.

Use short oral questions to check understanding and correct common misconceptions.

Combine those habits with the tracing, calculation, design and diagnosis demanded by the Ministry learning outcomes.

Allocate revision by both exam share and your error log.

A course with fewer items can still deserve more time if your foundations are weak.

The blueprint is a planning baseline, not a prediction of exact questions.

Definitions: distinguish nearby concepts, not just their names.

Calculations: write units, formulas, assumptions and intermediate steps.

Code and SQL: check validity before tracing the text; state any changes before analyzing a corrected version.

Algorithms and protocols: know the input, decision rule, state change, termination and failure cases.

Design choices: state the criterion.

A choice can change when cost, reliability, consistency or latency matters more.

| Course group | Items in the 100-item blueprint | Revision implication |
|---|---|---|
| Web Programming | 9 | Reserve extra practice for HTML, CSS, JavaScript, PHP, client/server behavior and safe data handling. |
| Data Structures and Algorithms | 7 | Trace operations and compare their assumptions and costs. |
| Each of the other fourteen courses | 6 each | Study all sixteen; do not abandon a whole course based on a short note or an alleged repeated-question list. |

## START03 How to handle exam traps

Underline NOT, FALSE, BEST and EXCEPT before comparing choices.

Check whether the stem supplies enough facts to select one answer.

A long scenario is not automatically an analysis question, and a familiar answer letter is not evidence.

When code is invalid, a diagram is absent, or two choices express the same rule, identify the clarification needed before answering.

Learn the correct rule and the smallest clarification needed.

Do not memorize an arbitrary key as a universal fact.

Distinguish worst case from average case, and an upper bound from a tight bound.

Check language version, SQL dialect, addressing assumptions, graph weights and tie-breaking rules.

For subnetting and scheduling, calculate rather than guess from a familiar number.

Use timed mixed papers only after you can explain your errors.

Revisit the reading sections behind recurring mistakes.

# Introduction to Artificial Intelligence

Complete Study Notes

## AI01 | AI capabilities and scope

Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language.

Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence.

A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Narrow or weak AI addresses bounded tasks, such as face recognition, spam filtering or a voice assistant.

General AI denotes broad competence transferable across domains; strong AI also has a philosophical meaning involving actual understanding.

Do not equate a humanoid body, autonomous vehicle, or many features with general intelligence.

### Exam tip

Exam trap

- Data analysis overlaps AI when it uses learning or inference. It is not categorically excluded from AI.

Introduction to Artificial Intelligence module

## AI02 | Four approaches and the Turing test

Thinking humanly models human mental processes; thinking rationally derives conclusions by valid logic.

Acting humanly seeks humanlike behavior; acting rationally selects actions expected to achieve objectives.

Thought processes and observed behavior are different evaluation targets.

The conversational Turing test asks whether a judge can reliably distinguish a machine from a human through their dialogue.

It requires language use, knowledge, reasoning and often learning.

It tests an operational behavior criterion, not proof of consciousness.

Cognitive models can be tested against response times, errors and psychological experiments.

### Example

Choose the approach

- Finding a best action sequence to a goal fits acting rationally. Explaining human errors with a computational model fits thinking humanly.

Introduction to Artificial Intelligence module

## AI03 | Agents, percepts and action

An agent perceives an environment through sensors and acts through actuators or effectors.

Cameras sense; motors move.

Software sensors may be incoming messages and software actions may be API requests.

Humans are agents but are not artificial agents.

A percept is one observation; a percept sequence is the entire observation history.

An agent function maps percept sequences to actions: f:P*→A. An agent program implements that mapping on an architecture; agent = architecture + program.

Environment means the setting in which it senses and acts, not merely the search space.

Autonomy means relying substantially on experience rather than only a designer's prior instructions.

Introduction to Artificial Intelligence module

## AI04 | Rationality and performance

A performance measure defines success externally: safety, useful deliveries, elapsed time, or cost.

Rationality depends on that measure, prior knowledge, percept history, and available actions.

A rational agent selects the action with the best expected performance given its information.

Omniscience would mean knowing actual action outcomes in advance.

Rationality does not require omniscience or guaranteed success.

Even complete outcome knowledge cannot create a successful action when every available action fails.

Information gathering and exploration may therefore be rational actions.

### Example

Uncertain route

- A route with expected travel time 20 minutes can be the rational choice even if an unforeseen closure makes it take 40 minutes.

Introduction to Artificial Intelligence module

## AI05 | Specify the task with PEAS

PEAS means Performance measure, Environment, Actuators, Sensors.

Specify all four before designing an agent.

For a taxi: performance includes safety and travel time; environment includes roads, traffic and passengers; actuators include steering, brakes and displays; sensors include cameras, positioning and speed readings.

A sensor is not an actuator just because it influences a decision.

The performance measure is not the percept sequence: one defines success, the other supplies evidence.

Separate what the agent can observe from what a designer knows about the world.

Introduction to Artificial Intelligence module

## AI06 | Simple and model-based reflex agents

A simple reflex agent matches condition-action rules against the current percept only: if dirty, clean.

It has no persistent world-state memory.

This is adequate when the current observation contains everything needed for its rule, but partial observability can make the same percept require different actions.

A model-based reflex agent maintains internal state using the previous state, previous action, current percept, and models of world change and action effects.

It then applies rules to that state.

Both kinds can use condition-action rules; memory and a world model distinguish them.

### Example

Remember a hidden room

- Remembering that a room was cleaned after it leaves sensor range requires internal state. Merely detecting current dirt does not.

Introduction to Artificial Intelligence module

## AI07 | Goals, utility and model updates

A goal-based agent considers future action sequences that reach a desired state.

A utility-based agent compares degrees of desirability and trade-offs, choosing maximum expected utility: $EU(a)=\Sigma s\ P(s|a,evidence)U(s)$.

Reaching the destination is a goal; balancing arrival time, safety and cost requires preferences among outcomes.

Updating an internal state after selecting an action predicts the action's effect; it does not itself modify the external environment.

A stored variable named goal is insufficient to prove a goal-based policy if BEST_ACTION never uses it.

Model-based, goal-based, utility-based and learning capabilities can coexist.

Introduction to Artificial Intelligence module

## AI08 | Learning-agent components

The performance element selects actions.

The learning element changes the agent to improve future performance.

The critic evaluates behavior against a performance standard and supplies feedback.

The problem generator suggests exploratory actions likely to yield useful experience.

Learning is improvement on task T, measured by P, through experience E. For spam filtering, T is classifying email, E is past labeled messages, and P might be recall at a required precision.

Exploration trades immediate reward for information; exploitation uses the best action currently known.

Learning can improve reflex, goal-based or utility-based agents.

Introduction to Artificial Intelligence module

## AI09 | Environment dimensions

Fully observable means sensors reveal relevant state; partially observable means relevant information is hidden or noisy.

Deterministic means state and action determine the next state; stochastic means outcomes involve randomness.

Strategic uncertainty about an opponent is distinct from dice-like randomness.

Episodic decisions are independent; sequential decisions affect later situations.

Static worlds do not change while deliberating; dynamic ones do.

A clock affecting score makes an otherwise static game semidynamic.

Discrete quantities have distinct alternatives; continuous quantities vary along real-valued ranges.

Known versus unknown concerns knowledge of the transition model, not observability.

### Example

Classify examples

- Chess is sequential, discrete and deterministic given both players' chosen moves; untimed chess is static, timed chess semidynamic. A fixed crossword is static. Driving is partially observable, sequential, dynamic and partly continuous.

Introduction to Artificial Intelligence module

## AI10 | Single-state, multiple-state and contingency problems

With a known deterministic model and full observation, a single current state can be used.

Without observation but with known deterministic transitions, a sensorless or multiple-state problem tracks a belief state: a set of possible states.

Nondeterminism or partial observation may require a contingency plan with branches based on later percepts.

Exploration or online search is required when the state space or transition model is unknown; observation alone does not identify this category.

These classifications can overlap.

A pairing based only on "partially observable, deterministic" cannot distinguish a known sensorless problem from an unknown exploration problem.

Russell and Norvig, problem-solving slides

## AI11 | Formulate before searching

First formulate the goal; then formulate the problem by specifying initial state, legal actions, transition model, goal test, and action costs.

Search selects a sequence; execution applies it.

A state abstracts away irrelevant world details.

In route finding, states are cities, actions are roads, successors are neighboring cities, and path cost is accumulated distance or time.

A solution is a legal path ending at a goal; an optimal solution has minimum cost.

Automatic programming or program synthesis searches for a program satisfying a specification; conditional plans and loops require richer representations than a fixed action list.

Introduction to Artificial Intelligence module

## AI12 | Frontiers and evaluation criteria

A search node records a state plus parent, action, depth and path cost.

Expanding a node generates its successors.

OPEN or frontier stores discovered candidates awaiting expansion; CLOSED records expanded states.

Reaching the same state through different paths does not imply the same node or cost.

Completeness means finding a solution if one exists under stated assumptions; optimality means returning the best solution.

Time counts work, often generated nodes; space counts simultaneous storage.

Let b be branching factor, d shallowest goal depth, m maximum search depth, and l a depth limit.

Graph-search bounds in vertices V and edges E differ from implicit-tree bounds.

Introduction to Artificial Intelligence module

## AI13 | Breadth-first search

BFS removes the oldest frontier entry using a FIFO queue.

Start with the root, remove the head, test it, append unseen successors at the tail, and repeat.

It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal.

It minimizes number of edges, so minimum cost requires equal positive step costs.

The tree contains 1+b+...+b^d nodes through depth d, giving O(b^d) under generation-time goal testing.

Expanding depth-d nodes before testing can generate depth d+1 and yields O(b^(d+1)); specify the convention.

Explicit finite graph traversal with visited-state tracking takes O(V+E) time and O(V) space.

### Example

Trace a queue

- If S has children A,B and A has C,D, BFS removal order starts S,A,B,C,D, assuming that child order.

Introduction to Artificial Intelligence module

## AI14 | Depth-first search

DFS follows the deepest pending branch using a LIFO stack or recursion.

FILO means the same order as LIFO.

With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Implicit-tree worst-case time is O(b^m), and space is O(bm) when pending siblings are stored, or O(m) for path-only recursive generation.

DFS is not generally optimal and can fail on infinite paths or cycles.

On a finite graph, visited-state tracking makes traversal complete but needs O(V) storage.

Do not confuse the O(b^m) worst-case time bound with the DFS space bounds above.

Introduction to Artificial Intelligence module

## AI15 | Uniform-cost search and Dijkstra

UCS uses a priority queue ordered by actual accumulated path cost g(n).

Remove a minimum-g node; accept a goal when removed, not when first generated.

Relax each edge: if a cheaper path is found, update its frontier cost and parent.

With finite branching and each step at least ε>0, UCS is complete and optimal.

Finite graphs permit nonnegative edges with proper duplicate handling.

Dijkstra uses the same minimum-distance principle, often computing all reachable distances; UCS commonly stops at a goal and generates states on demand.

Queue initialization is an implementation choice, not a fundamental distinction.

Negative edges invalidate the usual finalized-distance argument.

Introduction to Artificial Intelligence module

## AI16 | Depth limits, iterative deepening and two directions

DLS runs DFS to depth l and reports cutoff separately from failure.

It misses goals deeper than l.

Iterative deepening runs DLS with limits 0,1,2,...; for finite b it finds the shallowest solution with O(b^d) time and O(bd) frontier space, optimal for equal step costs.

Repeated shallow work is small compared with the deepest level.

Bidirectional search grows from start and goal until frontiers connect.

With suitable reverse actions, meeting tests and equal-cost BFS, time and space may fall to O(b^(d/2)).

Weighted variants require a sound stopping bound; the first intersection need not be cheapest.

IDA* deepens an f=g+h threshold rather than only a depth limit.

Introduction to Artificial Intelligence module

## AI17 | Heuristics and greedy best-first search

A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge.

It can reduce explored work, but computing it also costs time.

Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Best-first is a family that orders OPEN by an evaluation function.

Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid.

It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch.

On a finite graph with full duplicate handling it is complete, but not generally optimal.

Its implicit finite-tree worst case can be O(b^m).

Introduction to Artificial Intelligence module

## AI18 | Admissibility, consistency and A*

A* orders by f(n)=g(n)+h(n).

Admissibility means 0≤h(n)≤h*(n), the true cheapest remaining cost.

Consistency means h(n)≤c(n,n′)+h(n′) on every edge, with h(goal)=0; it implies admissibility on states with goal paths.

Consistency makes f nondecreasing along a path.

Admissibility suffices for optimal A* tree search; graph search that never reopens CLOSED states normally requires consistency.

With an admissible inconsistent heuristic, reopen states when g improves.

Test a goal when removed at minimum f. Completeness also needs finite branching and positive lower-bounded costs in infinite spaces.

Worst-case time and memory can be exponential; no heuristic universally guarantees speed.

Introduction to Artificial Intelligence module

UC Berkeley CS188, Search Note 1

## AI19 | Work a heuristic-search example

For S→A cost 2, S→B cost 1, A→G cost 2, B→G cost 10, set h(A)=2, h(B)=1, h(G)=0.

After expanding S, greedy chooses B because 1<2 and can then choose G, returning cost 11.

A* compares A:f=2+2=4 and B:f=1+1=2; after B, its G entry costs 11.

It next expands A and replaces G with cost 4, which is removed and accepted.

For a sliding puzzle with unit moves, misplaced-tile count and summed Manhattan distances excluding the blank are lower bounds.

Straight-line distance is a lower bound on road distance; it is not automatically a lower bound on travel time without conversion using a valid speed bound.

Introduction to Artificial Intelligence module

## AI20 | Local optimization and evolutionary search

Local search keeps a complete candidate configuration and improves its objective; the path may be irrelevant.

Hill climbing chooses better neighbors.

First-improvement accepts the first improvement; steepest-ascent examines all neighbors; stochastic variants choose among improvements.

Local maxima, plateaus and ridges can stop progress.

Random restarts or limited sideways steps can help without guaranteeing success.

A genetic algorithm maintains a population, evaluates fitness, selects parents, applies crossover and mutation, and repeats replacement.

It searches without guaranteeing global optimality.

Means-ends analysis selects a difference between current and goal states, chooses an operator reducing it, and recursively solves its preconditions.

Neither method is simply A* running backward.

Introduction to Artificial Intelligence module

## AI21 | Constraint satisfaction

A CSP specifies variables X, domains D, and constraints C. A solution assigns every variable a value and satisfies every constraint.

Complete means all variables assigned; consistent means no violated constraint.

A complete assignment can still be inconsistent.

Backtracking assigns a variable, tries a legal value, and undoes the choice on failure.

Choose the variable with fewest remaining values; break ties by degree; try values that rule out fewest neighbor choices.

Forward checking removes incompatible neighbor values.

Arc consistency removes a value when a binary constraint offers it no support in the other domain.

CSP is a problem framework using search and propagation, not a universal fourth uninformed traversal.

Introduction to Artificial Intelligence module

## AI22 | Games and minimax

Adversarial search models agents with competing objectives.

A game specifies states, player to move, legal actions, successor function, terminal test and utility.

Perfect information means previous moves and current state are visible; it does not reveal future choices or imply absence of chance.

Zero-sum utility makes one player's gain the other's loss.

Minimax assigns terminal utilities, takes minima at MIN nodes and maxima at MAX nodes, then chooses a maximum-valued root action.

It gives the best guaranteed outcome against optimal opposition within the modeled tree.

For stochastic outcomes add chance nodes weighted by probabilities.

### Example

Back up leaves

- If two MIN children have leaves [3,5] and [2,9], their values are 3 and 2. MAX chooses the first branch with value 3.

Introduction to Artificial Intelligence module

## AI23 | Alpha-beta and game evaluation

Alpha is MAX's best lower bound; beta is MIN's best upper bound.

Pass bounds down, update alpha at MAX and beta at MIN, and prune when alpha≥beta.

Pruning preserves the minimax value for the same searched tree.

Good move ordering increases pruning: worst-case O(b^m), ideal ordering about O(b^(m/2)).

At a depth cutoff use an evaluation function estimating position desirability.

Positive favors the designated MAX player, negative MIN; the scale is a convention, with +1/0/−1 often win/draw/loss.

Material counts weighted pieces; mobility counts legal options; territory scores controlled regions.

These are different features, often combined.

Heuristics improve limited computation, not guarantee victory.

Introduction to Artificial Intelligence module

## AI24 | Represent knowledge for inference

Knowledge representation encodes facts, objects, relations, events and rules so a machine can use them.

Syntax defines legal expressions; semantics defines their meaning.

A knowledge base stores assertions; an inference system derives consequences.

Declarative knowledge says what is true; procedural knowledge specifies how to act.

A semantic network connects concept nodes by labeled relations such as is-a. Frames describe stereotyped entities through slots, values and defaults.

Production systems use IF conditions THEN conclusions or actions, matching rules against working memory and selecting among applicable rules.

A graph suits relationships such as frequently bought together; frames suit attributes.

No representation is always best for recommendation without task requirements.

Introduction to Artificial Intelligence module

## AI25 | Propositional logic and models

Propositional atoms represent whole true-or-false claims. ¬P negates; P∧Q requires both; P∨Q requires at least one; P→Q is false only when P is true and Q false; P↔Q requires matching truth values.

Thus P→Q ≡ ¬P∨Q. De Morgan gives ¬(P∧Q) ≡ ¬P∨¬Q.

A model is an interpretation making a sentence true.

Valid means true in every interpretation; satisfiable means true in at least one; unsatisfiable means true in none.

Invalid means not valid, and may still be satisfiable.

P∨¬P is valid; P is satisfiable but not valid; P∧¬P is unsatisfiable.

Entailment KB⊨α means every model of KB also satisfies α.

Introduction to Artificial Intelligence module

## AI26 | First-order logic and quantifier scope

First-order logic adds objects, predicates, functions and quantified object variables.

Human(a) is a property; Likes(a,b) a relation; father(a) a function term.

Propositional logic has no object quantifiers, but its truth still depends on the assigned atom values.

Use ∀x(Human(x)→Mortal(x)) for all humans, and ∃x(Student(x)∧Tall(x)) for at least one tall student.

These are useful translation patterns, not rules forbidding other connectives. ∀x∃y Likes(x,y) allows a different y for each x; ∃y∀x Likes(x,y) requires one shared y. Negation swaps quantifiers: ¬∀x P(x) ≡ ∃x¬P(x).

Only occurrences within the correct quantifier's scope are bound.

### Example

Someone at the door

- ∃x(StandingAt(x,door)∧Tall(x)) requires one individual satisfying both predicates. Using implication instead allows an unrelated individual to satisfy it vacuously.

Introduction to Artificial Intelligence module

## AI27 | Inference, unification and chaining

Modus ponens derives Q from P and P→Q; modus tollens derives ¬P from P→Q and ¬Q. Affirming Q does not prove P. Universal instantiation substitutes an object for a universally quantified variable.

Unification finds a consistent substitution: Knows(x,Ada) and Knows(Ben,y) unify with x=Ben,y=Ada.

Distinct constants cannot be unified by renaming them.

Resolution combines clauses P∨Q and ¬P∨R into Q∨R. To prove α, add ¬α to KB in clause form and derive contradiction.

Forward chaining starts with facts and fires rules; backward chaining starts with a query and proves required premises.

Sound inference preserves truth; completeness means finding all entailed conclusions in its specified setting.

Introduction to Artificial Intelligence module

## AI28 | Deduction, induction and abduction

Deduction applies truth-preserving rules: all people are mortal; Ada is a person; therefore Ada is mortal.

Induction generalizes observations to a broader rule and can be overturned by new examples.

Abduction proposes a plausible explanation: wet ground suggests rain, although sprinklers could explain it.

Analogy transfers a pattern between similar situations.

Classical logical entailment is monotonic: adding premises preserves previous consequences.

Default reasoning can be nonmonotonic: infer a bird usually flies, then withdraw this when told it is a penguin.

Common-sense defaults are useful with incomplete knowledge, but are not guaranteed truths.

Blind search is a search strategy, not a definition of reasoning.

Introduction to Artificial Intelligence module

## AI29 | Probability and uncertain models

Probability describes uncertainty about events.

$P(\neg A)=1-P(A)$; $P(A|B)=P(A\cap B)/P(B)$, for $P(B)>0$.

Bayes gives $P(H|E)=P(E|H)P(H)/P(E)$: posterior equals likelihood times prior divided by evidence probability.

For mutually exclusive exhaustive hypotheses, $P(E)=\Sigma_i P(E|Hi)P(Hi)$.

A Bayesian network is a directed acyclic graph with conditional probability tables; its joint distribution factors as the product of each node's probability given its parents.

A Markov model assumes the next state depends on the current state rather than the entire past.

These represent uncertainty; fuzzy membership represents a different issue.

### Example

Update a hypothesis

- If $P(H)=0.1$, $P(E|H)=0.8$ and $P(E|\neg H)=0.2$, then $P(E)=0.26$ and $P(H|E)=0.08/0.26\approx 0.308$. Likelihood 0.8 is not the posterior.

Introduction to Artificial Intelligence module

## AI30 | Fuzzy logic and vague categories

Fuzzy sets assign graded membership $\mu(x)$ between 0 and 1 to concepts such as warm or tall.

A temperature can belong to warm with degree 0.7.

This is not a 70% probability that an otherwise crisp warm proposition is true.

Boolean logic uses only true and false; probability describes uncertainty about events.

A fuzzy controller fuzzifies measurements, applies graded IF-THEN rules, combines outputs and often defuzzifies to a numerical action.

Fuzzy control suits linguistic rules about degrees; reinforcement learning suits reward-based adaptation.

Merely saying an environment is complex or uncertain does not uniquely select either method.

James Brule, Fuzzy Systems tutorial, CMU archive

## AI31 | Supervised learning and prediction

Supervised learning fits a mapping from labeled input-target examples.

Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices.

Predictive analytics uses historical patterns to forecast outcomes and can use either.

Specify the target before choosing a method.

A decision tree tests an attribute at each internal node, follows the corresponding branch and predicts at a leaf.

ID3 chooses high information gain: $H(S)=-\Sigma p_i \log_2 p_i$; gain is parent entropy minus weighted child entropy.

Gini impurity is $1-\Sigma p_i^2$, zero for a pure class.

Pruning limits overfitting.

Neural networks, support-vector machines and trees can all be supervised.

Introduction to Artificial Intelligence module

## AI32 | Unsupervised, semi-supervised and active learning

Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features.

Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance.

K-means alternates assignment to nearest centroids and centroid updates.

Semi-supervised learning combines some labels with additional unlabeled data.

Active learning chooses informative cases for an oracle to label; it still uses labels. “Exploratory learning” is not an exclusive standard synonym: it can mean unsupervised exploration or active data acquisition, so define the feedback.

A task named summarization or time-series analysis is not automatically supervised; its training targets determine that.

Introduction to Artificial Intelligence module

Burr Settles, From Theories to Queries

## AI33 | Reinforcement learning

An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward.

Rewards evaluate outcomes rather than supply the correct action label.

Rewards may be delayed; not every correct-looking action receives immediate reward.

An illustrative discounted return is $G_t=r_{t+1}+\gamma r_{t+2}+\gamma^2 r_{t+3}+\ldots$, with $0\leq\gamma<1$.

The policy maps states or histories to action choices.

Exploration gathers information; exploitation pursues currently estimated reward.

RL is part of machine learning, which is part of AI. “No labels” alone fits both unsupervised learning and RL; reward-driven interaction distinguishes RL.

Introduction to Artificial Intelligence module

## AI34 | Neural networks and deep learning

A neuron computes $z=\Sigma i w_i x_i+b$ and output $a=\varphi(z)$.

For $x=(2,1)$, $w=(0.5,-1)$, $b=0.2$, $z=0.2$.

Linear activation returns z; sigmoid returns $1/(1+e^{-z})$; tanh is bipolar; ReLU returns $\max(0,z)$.

Nonlinearities let multiple layers represent nonlinear relationships.

Feedforward networks pass signals toward outputs without recurrent state loops; recurrent networks carry state across sequence steps.

Training adjusts weights, often through gradients and backpropagation.

Backpropagation is not the same as a recurrent connection.

Deep learning uses multiple representation layers and can learn features from images, sound and text.

CNNs share filters across image locations.

ANN is a brain-inspired abstraction, not a literal brain simulation; $DL \subset ML \subset AI$.

Introduction to Artificial Intelligence module

## AI35 | Generalization and validation

Overfitting means fitting training-specific details so performance drops on unseen data; underfitting means the model is too limited or poorly fitted even on training data.

A learning curve plots performance against training size or training progress.

Complexity control, regularization, pruning and more representative data can help.

Train data fits parameters; validation data selects models or settings; a held-out test set estimates final performance.

In k-fold validation, train k models, each on $k-1$ folds and evaluate on the remaining fold, then average scores.

Stratification approximately preserves label proportions.

Fit preprocessing within each training fold.

Time-ordered data needs splits respecting time; repeated tuning on the final test set leaks information.

Introduction to Artificial Intelligence module

scikit-learn, Cross-validation

## AI36 | Measure the right errors

With a fixed positive class, TP means true positives, FP false positive predictions, FN missed positives, TN true negatives.

Accuracy=(TP+TN)/N; precision=TP/(TP+FP); recall=TP/(TP+FN); F1=2PR/(P+R).

State the averaging and zero-denominator convention when relevant.

For TP=30, FP=10, FN=20, TN=40, accuracy=70%, precision=75%, recall=60%, and F1≈66.7%.

Accuracy can hide failure on a rare class.

Choose recall when missed positives matter; choose precision when false alarms are costly.

Regression errors include MAE=mean|y−ŷ| and MSE=mean(y−ŷ)²; squared error penalizes large misses more strongly.

Introduction to Artificial Intelligence module

## AI37 | Language processing tasks

NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation.

Speech recognition converts audio to text; speech synthesis generates speech.

They overlap language processing but solve different input/output tasks.

Tokenization divides text into processing units; tokens may be words, punctuation or subwords.

POS tagging assigns grammatical roles; named-entity recognition identifies people, organizations and locations.

Parsing identifies grammatical structure; semantics concerns meaning; discourse links sentences; pragmatics uses context and speaker intent.

Sentiment analysis predicts evaluative tone.

Ambiguity can occur at every level: “bank” has multiple senses, and “I saw the man with a telescope” has multiple attachments.

Introduction to Artificial Intelligence module

## AI38 | Morphology, sounds and prosody

Morphology studies meaningful word parts: un-happi-ness contains a prefix, stem and suffix; cats contains cat and plural -s. It differs from tokenization, which identifies processing boundaries.

Phonetics studies physical speech sounds; phonology studies a language’s sound patterns and contrasts.

Prosody concerns stress, rhythm, timing, prominence and intonation across syllables and phrases.

A pitch rise can help signal a question; shifting stress can change emphasis.

These cues assist dialogue and speech processing.

Pragmatics concerns intended meaning in context, not the internal components of words.

Introduction to Artificial Intelligence module

Jurafsky and Martin, Speech and Language Processing, phonetics chapter

## AI39 | Perception, vision and robotics

Computer vision interprets images or video: classification names image content, detection locates objects, segmentation labels pixels, and recognition identifies objects or people.

Occlusion, lighting change, noise, unfamiliar conditions and incomplete context make open-world perception difficult even when a face detector works well.

Robotics combines mechanical, electrical, control and computing systems to sense, plan and act physically.

Components include sensors, controller, power supply, actuators and an end effector.

Locomotion may be legged, wheeled or tracked, with trade-offs in terrain, stability and efficiency.

Software can act autonomously without being a robot.

Humanoid shape is not evidence of general intelligence.

Introduction to Artificial Intelligence module

## AI40 | Expert systems and choosing methods

An expert system applies specialized knowledge to a domain such as equipment diagnosis.

The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts.

The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Rules offer inspectable domain reasoning but can be brittle and costly to maintain.

Learned models handle data-rich patterns but depend on representative data and validation.

Search suits explicit action alternatives; probabilistic models suit uncertain evidence.

Bias in data or evaluation can create unfair outcomes.

Evaluate errors across relevant groups, robustness, privacy and human oversight rather than assuming automation is always beneficial.

Introduction to Artificial Intelligence module

## AI41 | Clarify search and logic questions

A* is forward best-first search combining actual past cost with estimated future cost; it does not inherently run a backward frontier.

Both A* and IDA* can have exponential worst cases and optimality under their conditions. “Minimum heuristic” and “estimated closest” describe the same greedy choice.

LIFO and FILO are synonyms.

Using information from the current state alone does not establish that a search uses a heuristic.

Ask whether remaining cost is estimated.

Both “invalid iff unsatisfiable” and “satisfiable implies valid” are false in general; distinguish validity from satisfiability.

Evaluate each validity claim separately, or identify all false statements when several are offered.

Introduction to Artificial Intelligence module

UC Berkeley CS188, Search Note 1

## AI42 | Distinguish agent categories and components

The common teaching list contains five agent classes when learning agents are included, or four decision architectures before learning is added.

State the convention when counting agent types. “Target-based” can mean goal-based; “unity-based” is not utility-based.

A generic rule does not distinguish simple from model-based reflex without specifying memory.

A goal declaration unused by the action policy does not establish goal-based behavior. “Sensors and effectors” describes interaction, not omniscience or intelligence by itself.

Define the mechanical distinction when comparing effectors and actuators as acting components.

Introduction to Artificial Intelligence module

## AI43 | Clarify overlapping learning and AI labels

ANNs, genetic search and reinforcement learning all improve candidate behavior or performance through experience in different ways.

Deep learning is an architecture family usable with supervised, unsupervised or reinforcement objectives.

Decision trees and deep models may both be supervised.

To distinguish learning regimes, specify labels, reward or unlabeled structure.

Face recognition is both an applied task and a narrow capability.

Broad machine-intelligence goals can include emulating human intelligence and building systems that learn.

Humanlike problem-solving alone does not uniquely select expert systems.

Python is widely used, but C++ and Java also implement AI; “commonly used” does not isolate one language.

McCarthy is associated with naming AI, not its sole invention.

Introduction to Artificial Intelligence module

## AI44 | Interpret quantified statements and search assumptions

The formula ∀x∃y∃t¬F(x,y,t) means that for each person there is a person and time at which fooling fails.

By quantifier negation this is ¬∃x∀y∀t F(x,y,t), so nobody fools everybody all the time.

Forbidding every successful instance is a much stronger claim than this formula.

Preserve the quantifiers and negation when translating between a formula and a verbal statement.

Compare Dijkstra and uniform-cost search under stated implementations rather than treating queue initialization as an intrinsic difference.

For chess, specify the unit of decision: moves within a match are sequential; independently reset matches can be episodes.

Select a recommendation design from explicit requirements; graph relationships can support a choice without excluding other designs.

Introduction to Artificial Intelligence module

## AI-S01 | Programmed computation and learned patterns

A conventional computer manipulates digital information according to a program or sequence of instructions. Programs can implement storage, retrieval and learning operations. Neural networks can learn pattern recognition from examples, so executing instructions and learning patterns are not mutually exclusive descriptions. This provides no universal claim that every machine is better or worse than every human at completing missing information.

AI algorithms are not exclusive to one programming language. Rumale is a Ruby machine-learning library with classification and clustering implementations, demonstrating that Ruby can implement learning algorithms as well as the general-purpose languages already discussed. Language choice and learning regime are different decisions.

NIST glossary - Computer

Introduction to Artificial Intelligence module · page 71

Rumale - official project README

## AI-S02 | Ordered lookup versus state-space traversal

Binary search exploits the order of a sorted sequence: a comparison narrows the remaining candidate interval instead of exploring arbitrary successor states. Array bisection therefore differs from breadth-first or depth-first traversal of a general state-space graph. A storage container such as an array does not by itself specify FIFO or LIFO frontier behavior.

Python documentation - Array bisection and searching sorted lists

Introduction to Artificial Intelligence module · page 18

## AI-S03 | Agent task models and practical planning

An AI task is modeled as an agent interacting with its environment: observations enter through sensors and selected actions affect the environment through actuators. This task-level description differs from an agent implementation, which combines a program with a computing architecture.

In ordinary usage, rationality means reasonable, sensible judgment. For an AI agent, the operational criterion is the action with the best expected performance given its percept history and available knowledge, rather than guaranteed success or knowledge of the actual future.

Planning constructs actions to reach a goal. Scheduling additionally assigns activities to times and resources while respecting constraints such as resource conflicts and precedence. It is an AI application when search or constraint reasoning chooses feasible or useful assignments; automated telescope experiment scheduling is an example.

Introduction to Artificial Intelligence module · page 11

Introduction to Artificial Intelligence module · page 7

## AI-S04 | Attributing the name of artificial intelligence

John McCarthy is credited with coining the term artificial intelligence in the 1955 proposal for the Dartmouth summer research project. The Dartmouth workshop took place in 1956 and helped establish the field. Naming the field does not mean that one person invented all its ideas or methods.

Stanford HAI - What is Artificial Intelligence?

Dartmouth - Artificial Intelligence Coined at Dartmouth

## AI-S05 | Related search and game-evaluation methods

Simulated annealing follows a current candidate and randomly samples a neighboring move. It accepts improvements and sometimes accepts worse moves according to a temperature that decreases over time. Unlike genetic search, its defining mechanism is not a population undergoing parent selection, crossover and mutation.

For two admissible heuristics, $h_2$ dominates $h_1$ when $h_2(n) \geq h_1(n)$ for every node. Dominance compares estimates; it is not itself a condition ensuring that an arbitrary heuristic never overestimates. Both estimates must separately satisfy the relevant admissibility assumptions.

A Monte Carlo game simulation samples possible continuations and uses their outcomes to estimate position value. An evaluation function is the scoring mapping itself; sampling is one way of obtaining an estimate, while minimax backs values up and alpha-beta removes branches that cannot affect that backed-up decision.

Russell and Norvig - Informed Search and Exploration

Silver et al. - Mastering the game of Go with deep neural networks and tree search

## AI-S06 | Prediction, transfer and explicit representations

Transfer learning reuses representations or a model learned on an earlier task for a new task, for example adapting pretrained image features to new image classes. This is a reuse strategy, not a specification that the new task has no labels or rewards.

Support-vector machines include supervised classification and regression methods. They use support vectors in their decision function rather than being defined by a next-state conditional-dependence assumption. In estimation, variance measures how predictions change across different training samples; overfitting instead names poor generalization caused by fitting training-specific details.

An ontology describes entities or individuals, classes and properties and may state logical relationships and restrictions. It organizes explicit conceptual knowledge. Conditional probability tables and a probabilistic factorization are additional modeling commitments, not inherent in the word ontology.

An application programming interface (API) exposes defined operations through which software requests functionality. It can be an interface used by an agent, but the interface alone is not the general entity that perceives an environment and selects actions.

TensorFlow - Transfer learning and fine-tuning

scikit-learn - Support vector machines

scikit-learn - Bias-variance decomposition

W3C - OWL 2 Primer

NIST glossary - Application Programming Interface

## AI-S07 | Behavior tests, understanding and software tests

Searle’s Chinese-room argument challenges whether executing formal symbol-manipulation rules is sufficient for understanding. It is a philosophical argument about understanding, not the conversational judge-and-hidden-participant procedure of the Turing test.

Black-box software testing examines externally visible functionality without inspecting implementation internals. A specification may concern any software behavior; this testing method does not by itself require conversational indistinguishability from a human.

John Searle - Minds, Brains, and Programs (author draft)

NIST glossary - Black box testing