BIT 2319 ARTIFICIAL INTELLIGENCE REVIEW QUESTIONS WITH ANSWERS

.....

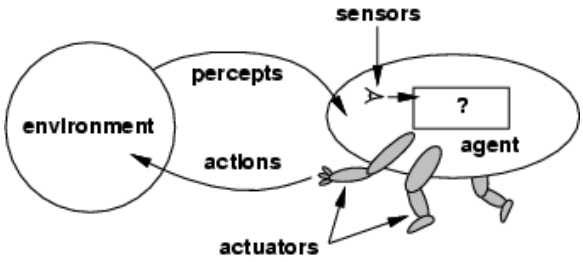
.....

1) *What is AI?*

Systems that think like humans	*Systems that think rationally*
Systems that act like humans	*Systems that act rationally*

2) ***Define an agent.***

*An **agent** is anything that can be viewed as **perceiving** its **environment** through **sensors** and **acting upon** that environment through **actuators.***



3) ***What is an agent function?***

*An agent's behavior is described by the **agent function** that maps any given **percept sequence** to an **action.***

4) ***Differentiate an agent function and an agent program.***

Agent Function	*Agent Program*
An abstract mathematical description	*A concrete implementation,running on the agent Architecture.*

5) ***What can Ai do today?***

- Autonomous Planning and Scheduling
  - Spacecraft control
  - Goal-directed planning, detection, diagnosis, problem recovery
- Game Planing
  - IBM Deep Blue
  - World chess champion
- Autonomous Control
  - CMU NAVLAB
  - Computer-controlled mini-van
  - Crossed the US without human control over 98% of the time
- Diagnosis
  - Medical diagnosis in several areas of medicine (e.g., pathology)
  - Explanation, justification for decisions
- Logistics Planning
  - DOD's Dynamic Analysis and Replanning Tool
  - Logistics planning of 50,000 vehicles, cargo, people
  - Embarkation, destination, route, conflict resolution
  - Paid back all of DARPA's 30-year investment in AI
- Robotics
  - Robotic surgical assistants
  - Cooperating autonomous robots in reconnaissance
  - Exploration of the Solar System

6) What is a task environment? How it is specified?

Task environments *are essentially the "problems" to which rational agents are the "solutions."*

A Task environment is specified using PEAS (Performance, Environment, Actuators, Sensors) description.

7) Give an example of PEAS description for an automated taxi.

Agent Type	Performance Measure	Environment	Actuators	Sensors
Taxi driver	Safe: fast, legal, comfortable trip, maximize profits	Roads, other traffic, pedestrians, customers	Steering, accelerator, brake, signal, horn, display	Cameras, sonar, speedometer, GPS, odometer, accelerometer, engine sensors, keyboard

Figure 2.4 PEAS description of the task environment for an automated taxi.

8) Give PEAS description for different agent types.

Agent Type	Performance Measure	Environment	Actuators	Sensors
Medical diagnosis system	Healthy patient, minimize costs, lawsuits	Patient, hospital, staff	Display questions, tests, diagnoses, treatments, referrals	Keyboard entry of symptoms, findings, patient's answers
Satellite image analysis system	Correct image categorization	Downlink from orbiting satellite	Display categorization of scene	Color pixel arrays
Part-picking robot	Percentage of parts in correct bins	Conveyor belt with parts; bins	Jointed arm and hand	Camera, joint angle sensors
Refinery controller	Maximize purity, yield, safety	Refinery, operators	Valves, pumps, heaters, displays	Temperature, pressure, chemical sensors
Interactive English tutor	Maximize student's score on test	Set of students, testing agency	Display exercises, suggestions, corrections	Keyboard entry

Figure 2.5 Examples of agent types and their PEAS descriptions.

9) List the properties of task environments.

- *Fully observable vs. partially observable.*
- *Deterministic vs. stochastic.*
- *Episodic vs. sequential*
- *Static vs, dynamic.*
- *Discrete vs. continuous.*
- *Single agent vs. multiagent.*

10)Write a function for the table driven agent.

```
function TABLE-DRIVEN-AGENT(percept) returns an action
  static: percepts, a sequence, initially empty
          table, a table of actions, indexed by percept sequences, initially fully specified

  append percept to the end of percepts
  action ← LOOKUP(percepts, table)
  return action
```

Figure 2.7 The TABLE-DRIVEN-AGENT program is invoked for each new percept and returns an action each time. It keeps track of the percept sequence using its own private data structure.

11) What are the four different kinds of agent programs?

Simple reflex agents;
Model-based reflex agents;
Goal-based agents; and
Utility-based agents.

12) Explain a simple reflex agent with a diagram.

Simple reflex agents

The simplest kind of agent is the **simple reflex agent.** These agents select actions on the basis AGENT
of the *current* percept, ignoring the rest of the percept history.

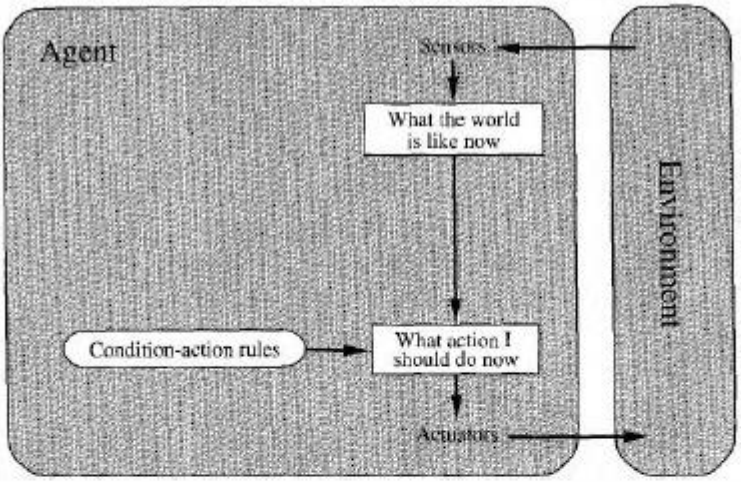


Figure 2.9 Schematic diagram of a simple reflex agent.

```
function SIMPLE-REFLEX-AGENT(percept) returns an action
  static: rules, a set of condition–action rules

  state ← INTERPRET-INPUT(percept)
  rule ← RULE-MATCH(state, rules)
  action ← RULE-ACTION[rule]
  return action
```

Figure 2.10 A simple reflex agent. It acts according to a rule whose condition matches the current state, as defined by the percept.

13) Explain with a diagram the model based reflex agent.

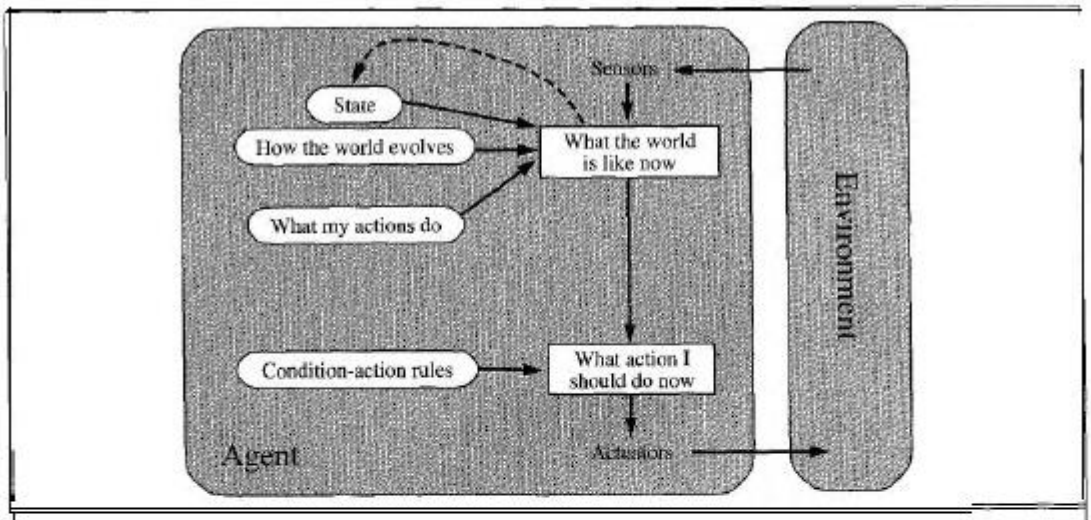


Figure 2.11 *A* model-based reflex agent.

```
function REFLEX-AGENT-WITH-STATE(percept) returns an action
  static: state, a description of the current world state
          rules, a set of condition–action rules
          action, the most recent action, initially none

  state ← UPDATE-STATE(state, action, percept)
  rule ← RULE-MATCH(state, rules)
  action ← RULE-ACTION[rule]
  return action
```

Figure 2.12 A model-based reflex agent. It keeps track of the current state of the world using an internal model. It then chooses an action in the same way as the reflex agent.

13a) Explain with a diagram the goal based reflex agent.

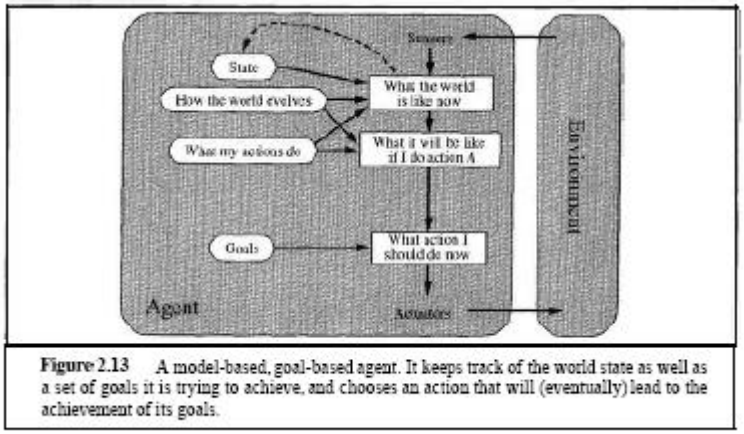


Figure 2.13 A model-based, goal-based agent. It keeps track of the world state as well as a set of goals it is trying to achieve, and chooses an action that will (eventually) lead to the achievement of its goals.

Knowing about the current state of the environment is not always enough to decide what
to do. For example, at a road junction, the taxi can turn left, turn right, or go straight on.

The correct decision depends on where the taxi is trying to get to. In other words, as well as a current state description, the agent needs some sort of **goal** information that describes situations that are desirable-for example, being at the passenger's destination.

13b) What are utility basedagents?

Goals alone are not really enough to generate high-quality behavior in most environments. For example, there are many action sequences that will get the taxi to its destination (thereby achieving the goal) but some are quicker, safer, more reliable, or cheaper than others. A **utility function** maps a state (or a sequence of states) onto a real number, which describes the associated degree of happiness.

13c) What are learning agents?

A learning agent can be divided into four conceptual components, as shown in Fig- 2.15. The most important distinction is between the learning element, which is re-
ELEMENT sponsible for making improvements, and the performance element, which is responsible for selecting external actions. The performance element is what we have previously considered to be the entire agent: it takes in percepts and decides on actions. The learning element uses
CRITIC feedback from the critic on how the agent is doing and determines how the performance element should be modified to do better in the future.

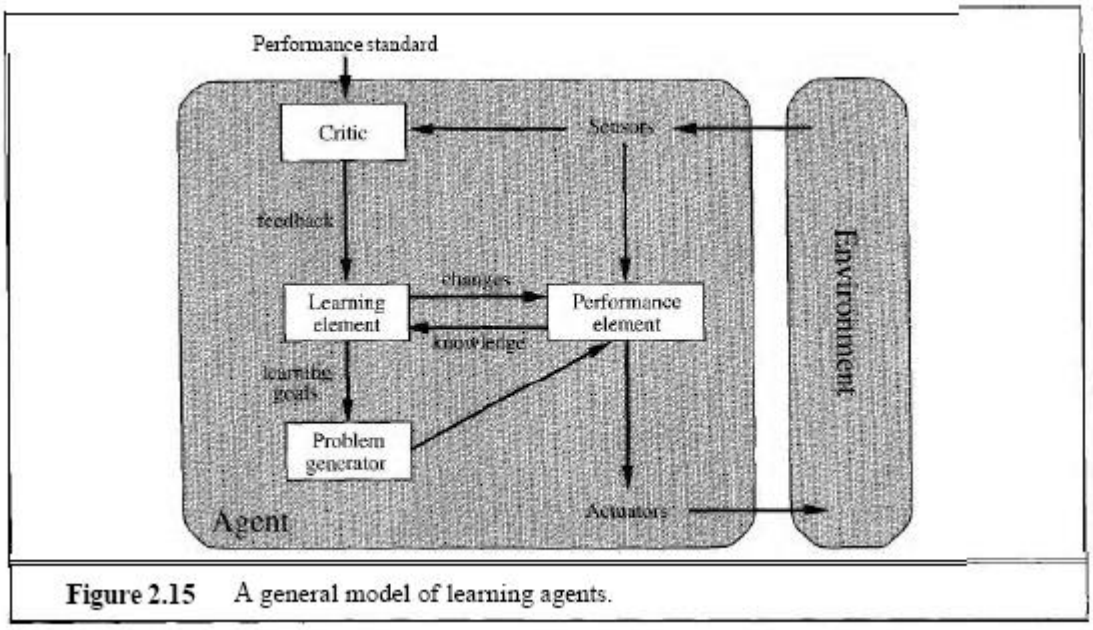


Figure 2.15 A general model of learning agents.

Searching Techniques

14)Define the problem solving agent.

A Problem solving agent is a ***goal-based*** *agent . It decide what to do by finding sequence of actions that lead to desirable states. The agent can adopt a goal and aim at satisfying it.*
Goal formulation is the first step in problem solving.

15) Define the terms goal formulation and problem formulation.

Goal formulation,*based on the current situation and the agent's performance measure,is the first step in problem solving.*

The agent's task is to find out which sequence of actions will get to a goal state.

Problem formulation *is the process of deciding what actions and states to consider given a goal*

16)List the steps involved in simple problem solving agent.

(i) Goal formulation
(ii) Problem formulation
(iii) Search
(iv) Search Algorithm
(v) Execution phase

17) Define search and search algorithm.

The process of looking for sequences actions from the current state to reach the goal state is called ***search.***
The ***search algorithm*** *takes a* ***problem*** *as* ***input*** *and returns a* ***solution*** *in the form of* ***action sequence.*** *Once a solution is found,the* ***execution phase*** *consists of carrying out the recommended action..*

18)What are the components of well-defined problems?

- *The* ***initial state*** *that the agent starts in . The initial state for our agent of example problem is described by In(Arad)*
- *A* ***Successor Function*** *returns the possible* ***actions*** *available to the agent. Given a state x,SUCCESSOR-FN(x) returns a set of {action,successor} ordered pairs where each action is one of the legal actions in state x,and each successor is a state that can be reached from x by applying the action.*

For example,from the state In(Arad),the successor function for the Romania problem would return

{ [Go(Sibiu),In(Sibiu)],[Go(Timisoara),In(Timisoara)],[Go(Zerind),In(Zerind)] }

- *Thr* ***goal test*** *determines whether the given state is a goal state.*

- *A* ***path cost*** *function assigns numeric cost to each action. For the Romania problem the cost of path might be its length in kilometers.*

19)Differentiate toy problems and real world problems.

TOY PROBLEMS	*REAL WORLD PROBLEMS*
A ***toy problem*** *is intended to illustrate various problem solving methods. It can be easily used by different researchers to compare the performance of algorithms.*	*A* ***real world problem*** *is one whose solutions people actually care about.*

20) Give examples of real world problems.

(i) Touring problems
(ii) Travelling Salesperson Problem(TSP)
(iii) VLSI layout

(iv) Robot navigation
(v) Automatic assembly sequencing
(vi) Internet searching

21) List the criteria to measure the performance of different search strategies.

- **Completeness** : Is the algorithm guaranteed to find a solution when there is one?
- **Optimality** : Does the strategy find the optimal solution?
- **Time complexity** : How long does it take to find a solution?
- **Space complexity** : How much memory is needed to perform the search?

22) Differentiate Uninformed Search(Blind search) and Informed Search(Heuristic Search) strategies.

Uninformed or Blind Search	Informed or Heuristic Search
o No additional information beyond that provided in the problem definition o Not effective o No information about number of steps or path cost	o More effective o Uses problem-specific knowledge beyond the definition of the problem itself.

23) Define Best-first-search.

Best-first search is an instance of the general TREE-SEARCH or GRAPH-SEARCH algorithm in which a node is selected for expansion based on the evaluation function f(n). Traditionally,the node with the *lowest* evaluation function is selected for expansion.

---

24) Differentiate Uninformed Search(Blind search) and Informed Search(Heuristic Search) strategies.

Uninformed or Blind Search	Informed or Heuristic Search
o No additional information beyond that provided in the problem definition	o Uses problem-specific knowledge beyond the definition of the problem itself.

○ Not effective ○ No information about number of steps or path cost	○ More effective

25)Define Best-first-search.

Best-first search is an instance of the general TREE-SEARCH or GRAPH-SEARCH algorithm in which a node is selected for expansion based on the evaluation function f(n). Traditionally,the node with the *lowest* evaluation function is selected for expansion.

26) What is a heuristic function?

A **heuristic function** or simply a heuristic is a function that ranks alternatives in various search algorithms at each branching step basing on an available information in order to make a decision which branch is to be followed during a search.
For example, for shortest path problems, a *heuristic* is a function, *h(n)* defined on the nodes of a search tree, which serves as an estimate of the cost of the cheapest path from that node to the goal node. Heuristics are used by informed search algorithms such as Greedy best-first search and A* to choose the best node to explore.

27)What is admissible heuristic?

Admissible Heuristic is a heuristic h(n) that never overestimates the cost from node n to
the goal node

28) What are relaxed problems?

- A problem with fewer restrictions on the actions is called a *relaxed problem*
- The cost of an optimal solution to a relaxed problem is an admissible heuristic for the original problem
- If the rules of the 8-puzzle are relaxed so that a tile can move *anywhere*, then *hoop(n)* gives the shortest solution
- If the rules are relaxed so that a tile can move to *any adjacent square*, then *hmd(n)* gives the shortest solution

29) What is greedy best-first-search?

Greedy best-first-search tries to expand the node that is closest to the goal,on the grounds that that is likely to lead to a solution quickly. For example,it evaluates nodes by using just the heuristic function : $f(n) = h(n)$

30) What is A search?*

A* search *is the most widely-known form of best-first search. It evaluates the nodes by combining g(n),the cost to reach the node,and h(n),the cost to get from the node to the goal:*

$$f(n) = g(n) + h(n)$$

Where f(n) = estimated cost of the cheapest solution through n.

g(n) is the path cost from the start node to node n.

h(n) = heuristic function

A search is both complete and optimal.*

31) What is Recursive best-first search?

Recursive best-first search *is a simple recursive algorithm that attempts to mimic the operation of standard best-first search,but using only linear space.*

32) What are local search algorithms?

Local search *algorithms operate using a single current state(rather than multiple paths) and generally move only to neighbors of that state. The local search algorithms are not systematic. The key two advantages are (i) they use very little memory – usually a constant amount,and (ii) they can often find reasonable solutions in large or infinite(continuous) state spaces for which systematic algorithms are unsuitable.*

33) What are the advantages of local search?

- *Use very little memory – usually a constant amount*
- *Can often find reasonable solutions in large or infinite state spaces (e.g., continuous)*
  - *Unsuitable for systematic search*
- *Useful for pure optimatization problems*
  - *Find the best state according to an objective function*
  - *Traveling salesman*

34) What are optimization problems?

In optimization problems *,the aim is to find the best state according to an* ***objective function*** *The optimization problem is then: Find values of the variables that minimize or maximize the objective function while satisfying the constraints.*

35) What is Hill-climbing search?

The ***Hill-climbing*** *algorithm is simply a loop that continually moves in the direction of increasing value –that is uphill. It*

terminates when it reaches a "peak" where no neighbor has a higher value. The algorithm does not maintain a search tree,so the current node data structure need only record the state and its objective function value. Hill-climbing does not look ahead beyond the immediate neighbors of the current state.

36) What are the problem faced by hill-climbing search?

Hill-climbing often get stuck for the following reasons :

- ***Local maxima*** *– A local maxima is a peak that is higher than each of its neighboring states,but lower than the local maximum. Hill climbing algorithm that reach the vicinity of a local maximum will be drawn upwards towards the peak,but will then be stuck with nowhere else to go.*
- ***Ridges*** *– Ridges result in a sequence of local maxima that is very difficult for greedy algorithms to navigate.*
- ***Plateaux*** *: a plateau is an area of state space landscape where the evaluation function is flat. A hill-climbing search might be unable to find its way off the plateau.*

37) What is local beam search?

The ***local beam search*** *algorithm keeps track of k states rather than just one. It begins with k randomly generated states. At each step,all the successors of all k states are generated. If any one is a goal,the algorithm halts. Otherwise,it selects the k best successors from the complete list and repeats.*

38)What are the variants of hill-climbing?

- ***Stochastic hill-climbing***
  - *Random selection among the uphill moves.*
  - *The selection probability can vary with the steepness of the uphill move.*
- ***First-choice hill-climbing***
  - *cfr. stochastic hill climbing by generating successors randomly until a better one is found.*
- ***Random-restart hill-climbing***
  - *Tries to avoid getting stuck in local maxima.*

39) Explain briefly simulated annealing search.

Simulated annealing *is an algorithm that combines* ***hill climbing*** *with* ***random walk*** *in some way that yields both* ***efficiency*** *and* ***completeness.***

40) What is genetic algorithm?

A **Genetic Algorithm(or GA)** is a variant of stochastic beam search in which successor states are generated by combining two parent states,rather than by specifying a single state

(1) Define communication.

Communication is the intentional exchange of information brought about by the production and perception of **signs** drawn from a shared system of conventional signs. Most animals use signs to represent important messages: food here, predator nearby etc. In a partially observable world, communication can help agents be successful because they can learn information that is observed or inferred by others.

(2) What is speech act?

What sets humans apart from other animals is the complex system of structured messages known as **language** that enables us to communicate most of what we know about the world. This is known as speech act.

Speaker, hearer, and **utterance** are generic terms referring to any mode of communication.

The term **word** is used to refer to any kind of conventional communicative sign.

(3) **What are the capabilities gained by an agent from speech act?.**

- **Query** other agents about particular aspects of the world. This is typically done by asking questions: Have you smelled the wumpus anywhere?
- **Inform** each other about the world. This is done by making representative statements: There's a breeze here in 3 4. Answering a question is another Pund of informing.
- **Request** other agents to perform actions: Please help me carry the gold. Sometimes **indirect speech act** (a request in the form of a statement or question) is considered more polite: I could use some help carrying this. An agent with authority can give commands (Alpha go right; Bravo and Charlie go lejl), and an agent with power can make a threat (Give me the gold, or else). Together, these kinds of speech acts are called **directives.**
- **Acknowledge** requests: **OK.**
- **Promise** or commit to a plan: I'll shoot the wumpus; you grab the gold.

(4)Define formal language.

A formal language is defined as a (possibly infinite) set of' ***strings.*** Each string is a concatenation of ***terminal symbols,*** sometimes called words. For example, in the language of first-order logic, the terminal symbols include A and P, and a typical string is "P A Q." . Formal languages such as first-order logic
and Java have strict mathematical definitions. This is in contrast to ***natural languages,*** such
as Chinese, Danish, and English, that have no strict definition but are used by a community.

(5) Define a grammar.

A ***grammar*** is a finite set of rules that specifies a language. Formal languages always
have an official grammar, specified in manuals or books. Natural languages have no official
grammar, but linguists strive to discover properties of the language by a process of scientific
inquiry and then to codify their discoveries in a grammar.

(6) What are the component steps of communication? Explain with an example.

The component steps of communication

A typical communication episode, in which speaker S wants to inform hearer H about proposition
P using words W, is composed of seven processes:

1) Intention. Somehow, speaker S decides that there is some proposition P that is worth
saying to hearer H. For our example, the speaker has the intention of having the hearer know
that the wumpus is no longer alive.

2) Generation. The speaker plans how to turn the proposition P into an utterance that
makes it likely that the hearer, upon perceiving the utterance in the current situation, can infer
the meaning P (or something close to it). Assume that the speaker is able to come up with
the words "The wumpus is dead," and call this W.

3) Synthesis. The speaker produces the physical realization W'of the words W. This
can be via ink on paper, vibrations in air, or some other medium. In Figure 22.1, we show

the agent synthesizing a string of sounds W'written in the phonetic alphabet defined on page 569: "[thaxwahmpaxsihzdehd]." The words are run together; this is typical of quickly spoken speech.

4) Perception. *H perceives the physical realization W' as W1 and decodes it as the words W2. When the medium is speech, the perception step is called* ***speech recognition;*** *when it is printing, it is called* ***optical character recognition.***

5) Analysis. *H infers that W2 has possible meanings P1, . . . , Pn. We divide analysis into three main parts:*

a) ***syntactic interpretation (or parsing****),*

b) ***Semantic interpretation,*** *and*

*c)****Pragmatic interpretation****.*

Parsing *is the process of building a* ***parse tree*** *for an input string, as shown in Figure 22.1. The interior nodes of the parse tree represent phrases and the leaf nodes represent words.*

Semantic interpretation *is the process of extracting the meaning of an utterance as an expression in some representation language. Figure 22.1 shows two possible semantic interpretations: that the wumpus is not alive and that it is tired (a colloquial meaning of dead). Utterances with several possible interpretations are said to be* ***ambiguous.***

Pragmatic interpretation *takes into account the fact that the same words can have different meanings in different situations.*

6) Disambiguation. H *infers that S intended to convey P, (where ideally P, = P). Most speakers are not intentionally ambiguous, but most utterances have several feasible interpretations.. Analysis generates possible interpretations;if more than one interpretation is found, then disambiguation chooses the one that is best.*

7) ***Incorporation. H*** *decides to believe P, (or not). A totally naive agent might believe everything it hears, but a sophisticated agent treats the speech act as evidence for P,, not confirmation of it.*

(7) What is machine learning?

Machine learning

- *Like human learning from past experiences,a computer does not have "experiences".*
- *A computer system learns from data, which represent some "past experiences" of an application domain.*
- *Objective of machine learning : learn a target function that can be used to predict the values of a discrete class attribute, e.g., approve or not-approved, and high-risk or low risk.*
- *The task is commonly called:* ***Supervised learning, classification, or inductive learning***

(8) Define Supervised learning .

Supervised learning *is a machine learning technique for learning a function from training data. The training data consist of pairs of input objects (typically vectors), and desired outputs. The output of the function can be a continuous value (called regression), or can predict a class label of the input object (called classification). The task of the supervised learner is to predict the value of the function for any valid input object after having seen a number of training examples (i.e. pairs of input and target output). To achieve this, the learner has to generalize from the presented data to unseen situations in a "reasonable" way.*

Another term for supervised learning is ***classification****. Classifier performance depend greatly on the characteristics of the data to be classified. There is no single classifier that works best on all given problems. Determining a suitable classifier for a given problem is however still more an art than a science. The most widely used classifiers are the* ***Neural Network*** *(Multi-layer Perceptron),* ***Support Vector Machines, k-Nearest Neighbors, Gaussian Mixture Model, Gaussian, Naive Bayes, Decision Tree and RBF classifiers.***

(9) ***Compare Supervised learning and unsupervised learning..***

Supervised vs. unsupervised Learning

- ***Supervised learning****:*
  classification *is seen as supervised learning from examples.*
  - ***Supervision****: The data (observations, measurements, etc.) are labeled with pre-defined classes. It is like that a "teacher" gives the classes (supervision).*
  - ***Test data*** *are classified into these classes too.*
- ***Unsupervised learning*** *(clustering)*
  - *Class labels of the data are unknown*
  - *Given a set of data, the task is to establish the existence of classes or clusters*

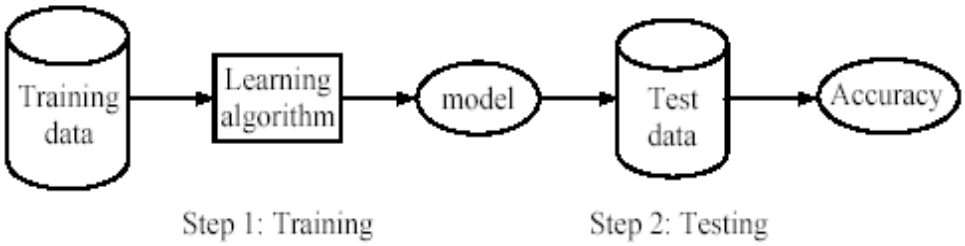
- *in the data*

(10) Explain the steps in Supervised learning process.

Supervised learning process: two steps

- ***Learning*** *(training): Learn a model using the training data*
- ***Testing****: Test the model using unseen test data to assess the model accuracy*

$$Accuracy = \frac{\text{Number of correct classifications}}{\text{Total number of test cases}},$$



(11) What is a decision tree?

A decision tree takes as input an object or situation described by a set of attributes and returns a "decision" – the predicted output value for the input.

A decision tree reaches its decision by performing a sequence of tests.

Example : "HOW TO" manuals (for car repair)

(12)Explain the Decision Tree learning with an example.

A decision tree reaches its decision by performing a sequence of tests. Each internal node in the tree corresponds to a test of the value of oine of the properties, and the branches from the node are labeled with the possible values of the test. Each leaf node in the tree specifies the value to be returned if that leaf is reached. The decision tree representation seems to be very natural for humans; indeed, many "How To" manuals (e.g., for car repair) are written entirely as a single decision tree stretching over hundreds of pages.

A somewhat simpler example is provided by the problem of whether to wait for a table at a restaurant. The aim here is to learn a definition for the ***goal predicate*** *Will Wait. In setting this up as a learning problem, we first have to state what attributes are available to describe examples in the domain.*

we will see how to automate this task; for

now, let's suppose we decide on the following list of attributes:

1. *Alternate: whether there is a suitable alternative restaurant nearby.*
2. *Bar: whether the restaurant has a comfortable bar area to wait in.*
3. *Fri/Sat: true on Fridays and Saturdays.*
4. *Hungry: whether we are hungry.*
5. *Patrons: how many people are in the restaurant (values are None, Some, and Full).*
6. *Price: the restaurant's price range ($, $$, $$$).*
7. *Raining: whether it is raining outside.*
8. *Reservation: whether we made a reservation.*
9. *Type: the kind of restaurant (French, Italian, Thai, or burger).*
10. *WaitEstimate: the wait estimated by the host (0-10 minutes, 10-30, 30-60, >60).*

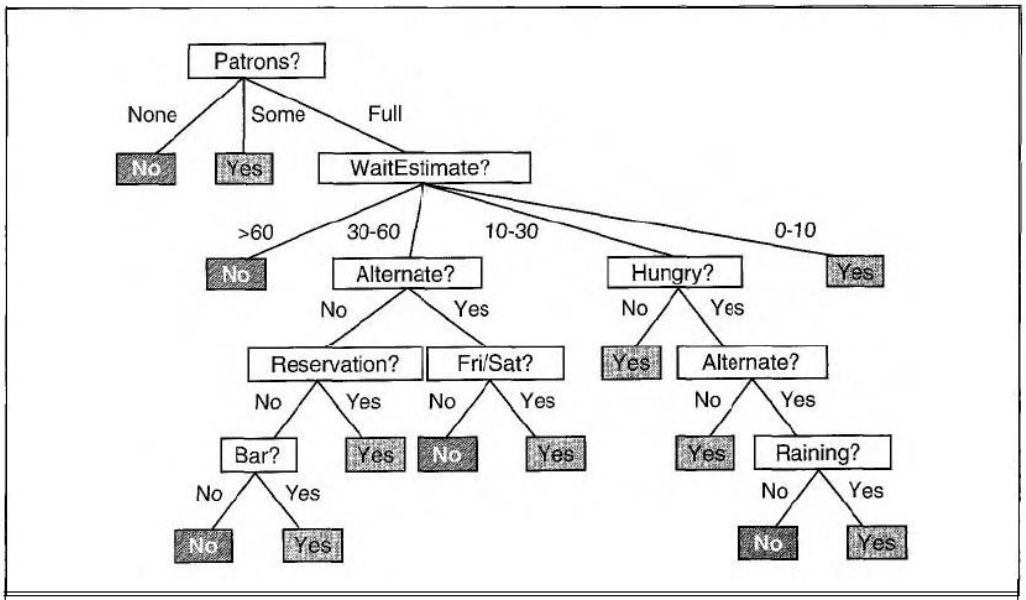


Figure 18.2 A decision tree for deciding whether to wait for a table.

(13) Give an example of decision tree induction from examples.

An example for a Boolean decision tree consists of a vector of' input attributes, X, and a single Boolean output value y. A set of examples (X1,Y1) . . . , (X2, y2) is shown in Figure 18.3. The positive examples are the ones in which the goal Will Wait is true (XI, X3, . . .); the negative examples are the ones in which it is false (X2, X5, . . .). The complete set of examples is called the ***training set.***

Example	Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	*Yes*
X_2	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–40*	*No*
X_3	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	*Yes*
X_4	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	*Yes*
X_5	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	*No*
X_6	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	*Yes*
X_7	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	*No*
X_8	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	*Yes*
X_9	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	*No*
X_{10}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	*No*
X_{11}	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	*No*
X_{12}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	*Yes*

Figure 18.3 Examples for the restaurant domain.

(14) Explain the Decision Tree Algorithm for an example problem.

The basic idea behind the Decision-Tree-Learning-Algorithm is to test the most important attribute first. By "most important," we mean the one that makes the most difference to the classification of an example. That way, we hope to get to the correct classification with a small number of tests, meaning that all paths in the tree will be short and the tree as a whole will be small.

function DECISION-TREE-LEARNING(*examples*, *attribs*, *default*) **returns** a decision tree
  inputs: *examples*, set of examples
    attrzbs, set of attributes
    default, default value for the goal predicate

```
if examples is empty then return default
else if all examples have the same classification then return the classification
else if attrzbs is empty then return MAJORITY-VALUE(examples)
else
    best ← CHOOSE-ATTRIBUTE(attribs, examples)
    tree ← a new decision tree with root test best
    m ← MAJORITY-VALUE(examples)
    for each value v_i of best do
        examples_i ← {elements of examples with best = v_i}
        subtree ← DECISION-TREE-LEARNING(examples_i, attribs − best, m)
        add a branch to tree with label v_i and subtree subtree
    return tree
```

Figure 18.5 The decision tree learning algorithm.

(15) How the performance of learning algorithm is assessed?

Assessing the performance of the learning algorithm

A learning algorithm is good if it produces hypotheses that do a good job of predicting the
classifications of unseen examples.

Obviously, a prediction is good if it turns out to be true, so we can assess the quality of a hypothesis by checking its predictions against the correct classification once we know it. We do this on a set of examples known as the ***test set.*** *If we train on all our available examples, then we will have to go out and get some more to test on, so often it is more convenient to adopt the following methodology:*

1. Collect a large set of examples.
2. Divide it into two disjoint sets: the ***training set*** *and the* ***test set.***
3. *Apply the learning algorithm to the training set, generating a hypothesis h.*
4. Measure the percentage of examples in the test set that are correctly classified by h.
5. Repeat steps 2 to 4 for different sizes of training sets and different randomly selected training sets of each size.

The result of this procedure is a set of data that can be processed to give the average prediction quality as a function of the size of the training set. This function can be plotted on a graph, giving what is called the ***learning curve*** *for the algorithm on the particular domain.*

(16) What are learning curves?

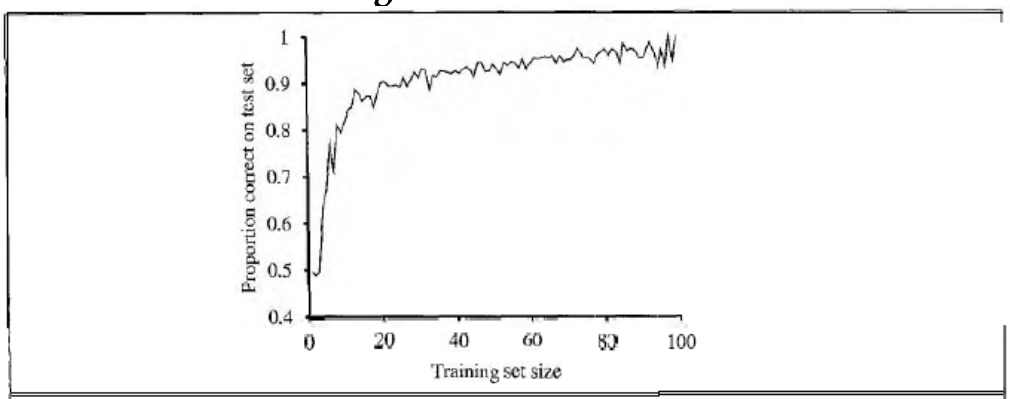


Figure 18.7 A learning curve for the decision tree algorithm on 100 randomly generated examples in the restaurant domain. The graph summarizes 20 trials.

.The average predicting quality of an algorithm can be plotted as a function of the size of the training set as shown in Fig 18.7. This is called a learning curve.
It can be noticed that, as the training set grows, the prediction quality increases. (For this reason, such curves are also called ***happy graphs.)*** *This is a good sign that there is indeed some pattern in the data and the learning algorithm is picking it up.*

(17) What is Ensemble Learning?

Ensemble *means a group producing a single effect.*

Ensemble learning *is the process by which multiple models, such as classifiers or experts, are strategically generated and combined to solve a particular computational intelligence problem. Ensemble learning is primarily used to improve the (classification, prediction, function approximation, etc.) performance of a model, or reduce the likelihood of an unfortunate selection of a poor one.*
Recently in the area of machine learning the concept of combining classifiers is proposed as a new direction for the improvement of the performance of individual classifiers. These classifiers could be based on a variety of classification methodologies, and could achieve different rate of correctly classified individuals. The goal of classification result integration algorithms is to generate more certain, precise and accurate system results.
Ensemble Classification
Aggregation of predictions of multiple classifiers with the goal of improving accuracy.

(18) What is Explanation based Learning?

Explanation based learning is a *method for extracting* ***general rules*** *from individual observations. The basic idea behind EBL is first construct an explanation of the observation using prior knowledge.*
EXAMPLE – A caveman toasting a lizard on the end of a pointed stick:
In the case of lizard toasting,the cavemen generalize by explaining the success of the pointed stick : It supports the lizard while keeping the hands away from the fire. From this explanation,they infer a general rule : that any long,rigid,sharp object can be used to toast small,soft-bodies edibles.
The general rules follows logically from the background knowledge possessed by the cavemen.

(19) What is Relevance Based Learning?

Relevance-based learning (RBL) *uses prior knowledge in the form of determinations to identify the relevant attributes, thereby generating a reduced hypothesis space and speeding up learning.*
RBL also allows deductive generalizations from single examples.
EXAMPLE
Consider the case of the traveler to Brazil meeting her first Brazilian. On hearing him speak Portugese,she immediately conclude that Brazilians speak Portugese. The inferen ce was based on her Background Knowledge,namely,that people in a given country speak the same language. The same is expressed in First Order Predicate Logic as follows :

$$Nationality(x, n) \wedge Nationality(y, n) \wedge Language(x, l) \Rightarrow Language(y, l) \quad (A)$$

"If x and y have the same nationality n and x speaks language l, then y also speaks it."

Sentences such as (A) express a strict form of relevance: given nationality, language is fully determined. (Put another way: language is a function of nationality.) These sentences are called ***functional dependencies or determinations.*** *They occur so commonly in certain kind of applications (e.g., defining database designs)*

(20) *Explain with a diagram the Cumulative Learning Process.*

The use of prior knowledge in learning leads to a picture of ***cumulative learning,*** *in which learning agents improve their learning ability as they acquire more knowledge.*

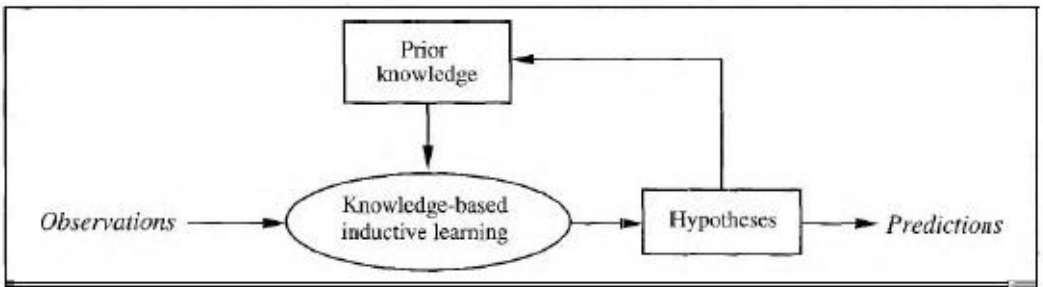


Figure 19.6 A cumulative learning process uses, and adds to, its stock of background knowledge over time.

The modem approach is to design agents that already know something and are trying to learn some more The general idea is shown schematically in Figure 19.6.

(21) *What is inductive logic programming?*

Inductive Logic Programming(ILP) combines inductive methods with the power of First Order Representation. The object of Inductive learning program is to come up with a set of sentences for the Hypotheses such that entailment constraint is satisfied.

EXAMPLE

The family tree example:

The descriptions

Father(Philips,Charles)

Father(Philips,Anne)

Parent(x,y) ⇔ [Mother(x,y) ∨ Father(x,y)]
Grandparent(x,y) ⇔ [∃z Parent(x,z) ∧ Parent(z,y)]

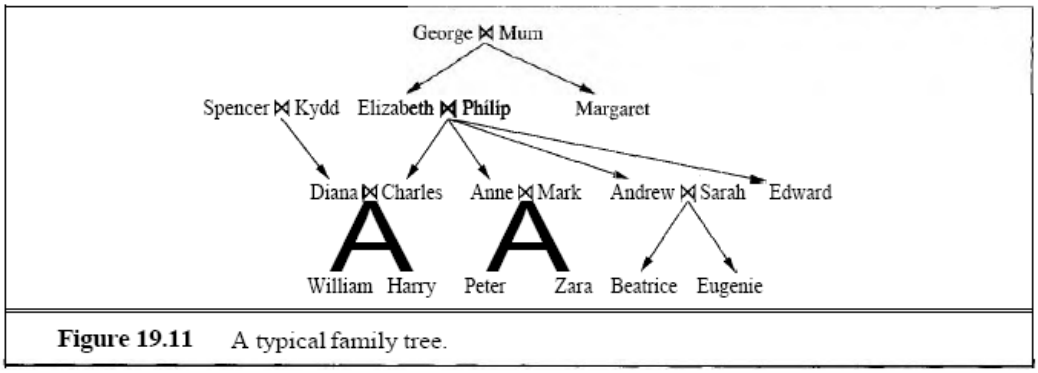


Figure 19.11 A typical family tree.

(22) What is Bayesian Learning?

Bayesian learning simply calculates the probability of each hypothesis,given the data,and makes predictions on that basis. That is,the predictions are made by using all the hypotheses,weighted by their probabilities,rather than using just ahigh "best" hypothesis. In this way learning is reduced to probabilistic inference.

(23) What are neural networks?

Traditionally, the term ***neural network*** *had been used to refer to a network or circuit of biological neurons[citation needed]. The modern usage of the term often refers to artificial neural networks, which are composed of artificial neurons or nodes.*

Artificial neural networks are made up of interconnecting artificial neurons (programming constructs that mimic the properties of biological neurons). Artificial neural networks may either be used to gain an understanding of biological neural networks, or for solving artificial intelligence problems.

In the artificial intelligence field, artificial neural networks have been applied successfully to speech recognition, image analysis and adaptive control, in order to construct software agents (in computer and video games) or autonomous robots. Most of the currently employed artificial neural networks for artificial intelligence are based on statistical estimation, optimization and control theory.

A ***Neural network*** *is an artificial system (made of artificial neuron cells). It is modeled after the way the human brain works. Several computing cells work in parallel to produce a result. This is usually seen as one of the possible ways Artificial intelligence can work. Most neural network can tolerate if one or more of the processing cells fail.*

What is important in the idea of neural networks is that they are able to learn by themselves, an ability which makes them remarkably distinctive in comparison to normal computers, which cannot do anything for which they are not programmed.

(24) Explain the units in the neural network.

Units in neural networks

Neural networks are composed of nodes or ***units*** *(see Figure 2:0.15) connected by directed* ***links.*** *A link from unit j to unit i serves to propagate the* ***activation*** *aj from* ***j*** *to i. Each link also has a numeric* ***weight*** *Wj,i ssociated with it, which determine the strength and sign of the connection. Each unit i first computes a weighted sum of its inputs:*

Then it applies an **activation function** g to this sum to derive the output:

$$a_i = g(in_i) = g\left(\sum_{j=0}^{n} W_{j,i} a_j\right) . \qquad (20.10)$$

Notice that we have included a **bias weight** $W_{0,i}$ connected to a fixed input $a_0 = -1$. We will explain its role in a moment.

The activation function g is designed to meet two desiderata. First, we want the unit to be "active" (near +1) when the "right" inputs are given, and "inactive" (near 0) when the "wrong" inputs are given. Second, the activation needs to be nonlinear, otherwise the entire neural network collapses into a simple linear function (see Exercise 20.17). Two choices for g

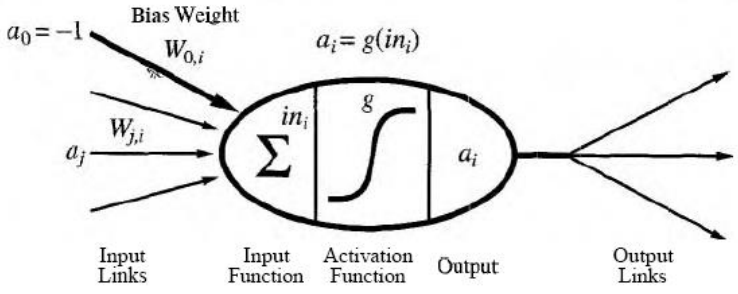


Figure 20.15 A simple mathematical model for a neuron. The unit's output activation is $a_i = g(\sum_{j=0}^{n} W_{j,i} a_j)$, where a_j is the output activation of unit j and $W_{j,i}$ is the weight on the link from unit j to this unit.

are shown in Figure 20.16: the **threshold** function and the **sigmoid function** (also known as the **logistic function).** The sigmoid function has the advantage of being differentiable, which we will see later is important for the weight-learning algorithm. Notice that both functions have a threshold (either hard or soft) at zero; the bias weight $W_{0,i}$ sets the *actual* threshold for the unit, in the sense that the unit is activated when the weighted sum of "real" inputs $\sum_{j=1}^{n} W_{j,i} a_j$ (i.e., excluding the bias input) exceeds $W_{0,i}$.

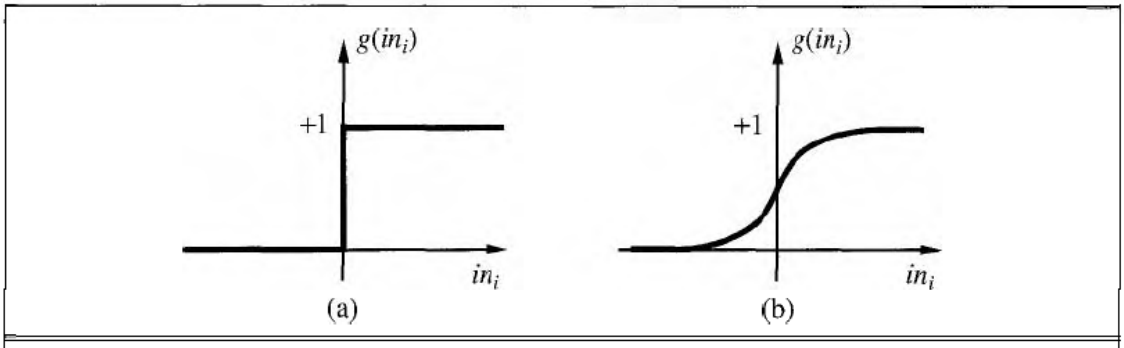


Figure 20.16 (a) The **threshold** activation function, which outputs 1 when the input is positive and 0 otherwise. (Sometimes the **sign** function is used instead, which outputs ± 1 depending on the sign of the input.) (b) The **sigmoid** function $1/(1 + e^{-x})$.

(25) What is reinforcement learning?

Reinforcement learning refers to a class of problems in machine learning which postulate an agent exploring an environment in which the agent perceives its current state and takes actions. The environment, in return, provide a a reward (which can be positive or negative). Reinforcement learning algorithms attempt to find a policy for maximising cumulative reward for the agent over the curse of the problem (Wikipedia).

Examples

Playing chess:

Reward comes at end of game

Ping-pong:

Reward on each point scored

Animals:

Hunger and pain - negative reward
food intake – positive reward

(26) What is passive reinforcement learning?

Passive Reinforcement Learning

- *Assume fully observable environment.*

- *Passive learning:*
  - *Policy is fixed (behavior does not change).*
  - *The agent learns how good each state is.*
- *Similar to policy evaluation, but:*
  - *Transition function and reward function or unknown.*
- *Why is it useful?*
  - *For future policy revisions.*

(27) What is active reinforcement learning?

Active Reinforcement Learning

- *Using passive reinforcement learning, utilities of states and transition probabilities are learned.*
- *Those utilities and transitions can be plugged into Bellman equations.*
- *Problem?*
  - *Bellman equations give optimal solutions given correct utility and transition functions.*
  - *Passive reinforcement learning produces approximate estimates of those functions.*