Section 1.1: Artificial Intelligence

Artificial Intelligence (AI) as a branch of computer science is defined as the study of mental faculties through the use of computational models [Charniak & McDermott 1985]. The world of intelligence can be viewed as three segments:

1. A knowledge complete domain. This is a domain wherein the basic knowledge and all related knowledge is known completely and definitely. This allows human beings to devise exact methods or algorithms to solve any problem that may arise in this domain (example, Sorting a set of numbers).

2. A knowledge incomplete domain. In this domain a significant amount of the concerned knowledge is known but this collection is still incomplete and ambiguous. In other words patterns of this domain are not known completely and exactly. The knowledge available does allow a particular amount of processing to be done, which means that problems in this domain are still solvable, though they are more difficult to solve than problems of the knowledge complete domain. The ill-formed nature and
incompleteness of patterns in this domain require a higher degree of capability to be handled. It is this capability that one recognises as a mark of intelligence (example, Diagnosing a disease).

(3) A knowledge unknown domain. The third domain is one in which the knowledge is so little known, that no processing of situations in this domain is possible.

Domain (1), provides algorithms and has formed the initial area of application of computers. AI techniques have made possible the use of computers in domain 2 as well.

The incomplete and ill-formed nature of patterns in domain (2), results in the generation of a large unconstrained search space, in which to find a solution for a given problem. The size of this search space leads to a non-polynomial computation time for algorithmic methods employing traditional search techniques, like depth-first search. To overcome this time complexity, AI provides two approaches:

1) Weak methods : employing heuristic search techniques.
2) Strong methods: employing knowledge about the domain to constrain the search space.

This work deals with two major areas of AI: intelligent tutoring systems and machine learning systems.

Section 1.2: Intelligent Tutoring Systems (ITS)

Section 1.2.1: Computer assisted learning (CAL)

CAL, the spearhead of computer-based-education conceives of the computer as a learning resource and a catalyst for the imagination of the student. The main fields of CAL are [Gunzenhauser 1984]:

1) Interactive programming: application of dialogue languages like LOGO in computer science courses.

2) Design and application of simulation models in the areas of science, management and technology.

3) On-line help systems to aid users of software packages and technical devices.

4) Method bank educational systems which advise newcomers on appropriate process methods.
The advantages of a CAL based learning approach are [Hudson 1984]:

1) The computer can always be well prepared for instruction.

2) It insists that a given concept must be fully understood before a student can move on.

3) It always provides a smooth, flowing lecture and gives hints and clues, helpful in getting the correct idea.

4) The learner is reinforced by engaging in learning activity.

5) The flexible rescheduling on the system, between students, is advantageous.

6) Instruction can be given to a number of locations at different places.

7) Students can learn at their own pace.

8) Quality of instruction is standardised.

9) Material can be presented again and again, with patience.

10) Relieves school personnel for research work.

11) Students forced to respond and concentrate on their studies.
CAL, also overcomes, the problems caused by the paucity of qualified teachers.

CAL programs are divided into two streams: linear and branching. CAL had its origination, from Programmed Learning, in 1958, with a program that taught binary arithmetic [Hudson 1984]. In the seventies, two major CAL packages were developed in America: PLATO (Programmed Logic for Automatic Teaching Operations) and TICCIT (Time shared Interactive Computer Controlled Information Television) [Rosenberg 1986]. The different varieties of CAL programs are a) Multichoice tests b) Drill and test c) Educational games d) Simulations e) Experimental aids and f) Idiosyncratic programs.

Though, CAL programs have had a decent run, they are limited to subjects that can be clearly analysed into unambiguous YES / NO terms; a feature, lacking in subjects like, sociology, politics, religion and literature [Hudson 1984]. Also, CAL programs have, very limited powers of reasoning [Bonnet 1985]. The above deficiencies led to Intelligent Tutoring Systems (ITS), which incorporated Artificial Intelligence techniques in the teaching process.
Section 1.2.2 : AI in CAL

Intelligent Tutoring Systems (ITS) starts from the principle that teaching programs should themselves be experts in that particular field [Bonnet 1985], while traditional CAL programs are statically organised and reflect only, superficially, the pedagogical and domain knowledge of expert human teachers [Wegner 1987]. The explicit representation of knowledge to be communicated categorises ITS with knowledge communication systems.

The knowledge base in an ITS, must be capable of meeting the following requirements [Arun kumar and Sarukesi 1989]:

- the system must be able to understand what it is doing and what it is teaching.
- the system needs different levels of help and hint functions to assist the student in finding an adequate path to the solution.
- the system must be able to analyse former performance of the learner by a Student Model.
- the system must be capable of deciding when to give a particular hint. It should recognise fundamental difficulties of the student, which occur during the problem solving process. In
special cases, the system must even produce counter examples.

The diagram of an ITS is presented in Figure 1. In addition to the subject knowledge base (SKB) which represents the subject matter to be taught and an interface, three major models are incorporated in the system.
Figure 1. An ideal, Intelligent Tutoring System
(1) Model of an expert (EM), which represents the processes used by a human expert in solving problems in the domain. This is primarily used for diagnostic purposes.

(2) Model of a tutor (TM), which represents the pedagogical knowledge. This is used for help and hint modes.

(3) Model of a student (SM), which represents the different types of student behaviours that could be encountered in the domain. The SM describes the student's knowledge about what is being taught [Clancey 1986].

From the viewpoint of epistemic fidelity, (defined as the degree of completeness to which the physical realisation of a representation, brings a rendition of the expertise as defined at the epistemic level [Wegner 1987] ), frame-based CAL maintains all the knowledge that is to be conveyed in an external representation: the interface; while in the ideal ITS the interface is strictly an external representation of the expertise represented internally. In current ITS, there is a degree of overlap between the two representations. Figure 2, depicts this comparative evaluation of the educational systems.
Figure 2. Comparative evaluation of educational systems [Wegner 1987]
The emphasis on the same granularity level of epistemic fidelity for both the internal and the external representation is primarily to allow the system to fully control the meaning of and monitor, each communication step in the teaching process [Wegner 1987].

From the perspective of the components of an educational system an useful distinction arises between CAL and ITS: An ITS represents at least one component in the form of a qualitative model, in contrast to the quantitative representations in CAL systems [Clancey 1986] (A qualitative model is defined as a description of objects and processes in terms of spatial, temporal and causal relations. They are not numeric or physical analogs [Clancey 1986].).

The field of ITS was launched by SCHOLAR [Carbonell 1970]. The subject matter was the geography of South America. The subject knowledge was represented in the form of a semantic network. SCHOLAR separated the tutorial knowledge from the domain knowledge and expressed dialogue management tasks in the terms of general representational abstractions. WHY was developed as a tutoring system for rainfall processes [Stevens & Collins 1977]. This followed the Socratic tutoring method, wherein the tutor does not teach a subject by direct
presentation, but leads the student by successive questions to formulate general principles on the basis of individual cases, to examine the validity of her own hypotheses, to discover contradictions and finally extract correct inferences from the known facts.

SOPHIE (SOPHisticated Instructional Environment) was designed with the pedagogical purpose of providing a reactive learning environment in which the student can test her ideas and receive advice from a critique [Brown et.al. 1975]. The domain of SOPHIE was the troubleshooting of faulty electronic circuits. SOPHIE-I [Brown & Burton 1975] used multiple representations of its domain knowledge:

1) Simulation based with a mathematical model of the circuit.
2) Procedural with a set of intelligent specialists using the model.
3) Declarative with a semantic net of facts.

SOPHIE-I used quantitative simulation for making inferences. This has a major failing in its inability to give complete causal reasons for inferences. In SOPHIE-II [Brown et.al. 1976] a troubleshooting expert was added. This module can demonstrate troubleshooting strategies. Reasoning, in contrast to SOPHIE-I, is done qualitatively making meaningful measurements.
and explaining strategic decisions. SOPHIE-III [Brown et al. 1982] is designed to be the core of an environment centering on the learning needs of a student. Its expertise is divided into an electronics expert and a troubleshooting expert with the latter working on top of the electronics expert.

STEAMER [Hollan, Hutchins & Weitzman 1984] is an instructional tool for training engineers who will operate large ships. It employs an interactive, inspectable simulation based on computer graphics, to support development of an accurate model of the system, by the student. In an effort to render a conceptual rather than a physical view, the simulation of the propulsion plant presented, reflects more a mental model (a mental model is an internal model that a human has of the domain), as used by an expert than an exact physical model.

QUEST (Qualitative Understanding of Electrical System Troubleshooting) [White & Frederiksen 1986] investigated existence of multiple mental models for a domain. The domain knowledge about electrical circuits is represented internally as a qualitative model.

Many ITS have been developed for the domain of programming: MALT, was used in the domain of machine code programs. It maintained a tree of sub-problems, together with their breakdown into primitive tasks. Probabilites attached to
these sub-problem branches, represented the student's expertise. The traversal of the tree determined which problem was to be presented and also, constructed an ideal solution [du.Boulay 1988]. GREATERP, developed to teach LISP, uses the process of model tracing to follow the student's behaviour. Mistakes, by the student are flagged, as soon as they are detected [Anderson & Skwarecki 1986]. PROUST, demonstrated, impressive capabilities to diagnose a wide variety of logical errors in student's PASCAL programs. The system is provided with, the 'goals' of a particular programming exercise and a repertoire of 'plans' to achieve each of the goals. A sophisticated matcher, finds the best fit between the student's program and the expected structure of goals and plans. The 'violations' by the student are then reported [du.Boulay 1988].

SAMPLE [Micarelli et.al.1992] was developed to teach high school and college students how to analyse steady state electrical circuits. The fundamental building blocks for knowledge representation such as frames, production rules and procedural attachments are supported by a context (worlds) system. The tutoring module has a control sub-system which monitors the interaction with the student and a remedial sub-system which attempts to provide the right remediation to the student.
SHERLOCK [Lajoie & Lesgold 1989], a computer-based coached practice environment was used to teach Air Force trainees, troubleshooting of the electronic equipment they use to make diagnosis of faulty devices of an F-15 aircraft. SHERLOCK provides coaching on specific skills when trainees ask for help and demonstrate through their performance that assistance is needed for strengthening their skills. Pedagogically SHERLOCK approximates an apprenticeship's environment. The system has much in common with ITS, though it is not driven by a dynamic student model.

Section 1.3: Machine learning systems

Machine learning denotes changes in the system that are adaptive in the sense that they enable the system to do the same tasks or tasks drawn from the same population, more effectively and more efficiently, the next time [Simon 1983]. Learning covers a wide range of phenomena, from skill refinement to knowledge acquisition. Knowledge acquisition itself includes different activities ranging from rote learning or simple storing of information through learning by discovery to learning by analogy.
The fundamental machine learning strategies are [Michalski 1984]:

1) **Rote learning**: Also called direct implanting of knowledge; this strategy requires no inference or input processing on the part of the learner. This strategy includes learning by being constructed, programmed or modified by an external agent. Samuel's checker's program [Samuel 1963] used rote learning to remember values of situations encountered previously (This program also uses the learning strategy of parameter adjustment).

2) **Learning by being told**: In this strategy the learner transforms the knowledge from the input language to an internally usable representation. The learner also integrates it with prior knowledge to facilitate effective retrieval and use (example, TEIRESIAS [Davis & Lenat 1982]).

3) **Learning by deductive inference**: This strategy conducts deductive inference on the knowledge translated from the input language (example, Automatic theorem prover).

4) **Learning by Analogy**: This learning strategy transforms or extends existing knowledge applicable in one domain to perform a similar task in another domain. This involves retrieving relevant knowledge and then appropriately transforming it to be applicable in a new situation (example, [Carbonell 1983], [Winston 1984], [Burstein 1984]).
5) **Learning from examples**: In this, the learner induces a general concept description, given a set of examples, or a set of, examples and counter examples. The system has no prior knowledge from which it can derive the desired knowledge. This strategy is also called concept acquisition. The implementation can be a one-step, batch process or a multi-step, incremental process. The batch process presents all examples at once. In the incremental process examples are introduced one by one with the learner forming tentative hypotheses based on data at a given step. The hypotheses are refined after consideration of new examples. The examples may be given by a human teacher (example, [Winston 1975], AQ11 [Michalski & Larson 1978]).

6) **Learning from experimentation**: This forms a type of learning from examples. Here the examples are generated by the system through search or other active effort (example, [Mitchell, Utgoff & Banerji 1983]).

7) **Learning by observation and discovery**: In this learning strategy there is no teacher present. The strategy includes processes like, generating classifications (example, [Michalski & Stepp 1983b]), or discovering new relationships and laws (example, [Langley, Simon & Bradshaw 1983]). Two sub-classifications of this strategy are made according to the degree
of interaction between the learning system and the external environment:

a) Passive Observation: Here the learner builds a description of a given set of observations (example, [Michalski & Stepp 1983a]).

b) Active experimentation: Here the learner makes changes in the environment and observes the results of these changes (example, [Lenat 1976]).

The learning strategies 1 to 7 have been ordered in increasing amount of effort on the part of the learning system and corresponding decrease in effort by the teacher.

Learning by induction is the underlying theme in strategies 5, 6 and 7. A general paradigm of inductive inference is [Michalski 1984]:

**Given** (a) Observational statements (Facts) (b) A tentative inductive assertion (which may be null) (c) background knowledge that defines the goal of inference, the preference criterion (the inductive bias) for choosing among plausible hypotheses,

**Find** An inductive assertion (hypothesis) that tautoligically implies the observational statements and satisfies the background knowledge.
The process of induction involves operations of generalising, transforming, correcting and refining knowledge representations in order to accommodate new facts.

In the area of concept learning, Winston introduces the idea of a Near-Miss example. A Near-Miss is an object that is a counter example of the concept to be learned, but is very similar to positive instances. This allows the induction process to be more focussed [Winston 1975]. Mitchell introduced another approach called Version-Spaces. This works by maintaining a set of possible descriptions (a Version-space) of the concept to be learned and evolving that set as new examples are presented [Mitchell 1978]. The algorithm for narrowing the version-space is called the candidate elimination algorithm.

A third approach was introduced by Quinlan, called induction of decision trees. The program called ID3, uses an iterative method to build up a decision tree, which represents concepts [Quinlan 1986].

Learning by induction has also introduced an analytical knowledge-intensive approach, contrasting the empirical data-intensive approach discussed so far (The empirical approach is called Similarity Based Learning (SBL)). This approach called Explanation Based Learning (EBL) attempts to learn from a single example 'y' by first explaining why 'y' is an
example of the target concept and then generalising this explanation [De Jong & Mooney 1986]. EBL programs are given:

(a) A training example (b) a goal concept (c) operationality criterion (description of which concepts are usable) and (d) domain theory. They are required to find a generalisation of the training instance that is sufficient to describe the goal concept and also satisfies the operationality criterion.

The majority of work on concept finding has used techniques for generalisation. Generalisation works well for conjunctive concepts but faces difficulties with disjunctive concepts as well as with errorful or dynamic environments. An alternative technique, discrimination learning works by looking for differences between positive and negative examples, rather than for features held in common between all positive examples. This strategy though slower in learning than generalisation can handle disjunctive concepts, noise and dynamic environments [Langley 1985].

Learning by analogy revolves around noting the significant aspects of a situation which would serve to match a new problem with existing knowledge. The focus in learning by analogy is therefore on [Carbonell 1986]:

a) What it means for problems or environments to share
a) What it means for problems or environments to share significant aspects.

b) What knowledge is transferred from past experience to the new knowledge.

c) precisely how the knowledge transfer occurs.

d) How analogically related experiences are selected from a potentially vast long term memory of past problem solving episodes.

Transformational analogy uses the idea of transforming a solution to a previous problem into a solution for a current problem. Carbonell (1983), describes one method for this transformation. Whole solutions are viewed as states in a problem space, called T-space. T-operators prescribe the methods of transforming solutions (states) into other solutions. Transformational analogy does not look at how the previous problem was solved. The methods used in the course of a problem solving episode, called its derivation, are often very pertinent to the conversion of the old problem solving episode to the new one. Analogical reasoning that takes the derivation into account is called Derivational analogy [Carbonell 1986].

Other learning strategies which have been employed successfully are:

1) Learning by parameter adjustment
2) Learning with macro operators

3) Learning by chunking

Learning by parameter adjustment works by increasing the value of co-efficients of features that appear to lead to overall success and decreasing values of co-efficients that lead to failures [Samuel 1963].

In the learning with macro operators strategy sequences of actions that can be treated as a whole set are clubbed together and designated, in future, by a single macro operator [Korf 1985].

Learning by chunking is a process similar to learning with macro operators. The difference lies in its general applicability towards any goal state. Also chunking emphasises how learning can occur during problem solving. Learning by chunking is exploited by SOAR [Laird et.al. 1986].

Two environments that provide opportunity for application of learning strategies are:

1) Failure-driven learning: Observation of how a current mode of operation fails, helps revise the concept definition or produces a new concept [Sussman 1975]. Hall [Hall 1988], utilised this technique in situations where an EBL system fails to explain a given positive example as an instance of the goal concept. The system employs precedent analysis, partial explanations of a precedent (or rule) to isolate the new technique it embodies and rule reanalysis. A new rule is thus conjectured.

2) Learning with genetic algorithms: Genetic algorithms are a collection of adaptive search procedures based on models
containing three basic elements: (1) a Darwanian notion of "fitness" which governs the extent to which an individual can influence future generations. (2) a "mating operator" which produces offspring for the next generation and (3) "genetic operators" which determine the genetic make-up of offspring from the genetic material of the parents [De Jong 1988]. The learning strategy involves maintaining a population of tested structures and using genetic algorithms to generate new structures with better performance expectations.

Machine learning systems have also been built around the theory of Rough sets [Kononenko & Zorc 1994]. A rough set is defined as an approximation of set of objects Y, using two unions of equivalence classes of objects, P-lower approximation of Y and P-upper approximation of Y. (Y is a set belonging to the finite set of objects and P is a subset of attributes of the finite set of attributes.) The learning algorithm works in three steps:

1) Reduction of those attributes from set Q that do not change the Q-equivalence classes.
2) Elimination of attributes from reduct A until the lowest permitted predefined value of Gamma is reached.
3) Generation of rules for each class in turn using only attributes from P.

Application of Rough set theory (RST) to machine learning has the following drawbacks [Kononenko & Zorc 1994]:

1) Trivial notions are formalised complicatedly in RST. The definition of the boundary region only represents a part of the instance space where one does not have enough attributes to
discriminate between classes. But there is no information about the distribution of instances inside the boundary region.

2) The derived rules use only a fixed subset of attributes, discarding other, probably useful ones. Applicability of the theory therefore to noisy, incomplete data sets is doubtful.

3) RST uses ad-hoc heuristics and ad-hoc definitions instead of using results from probability theory and from information theory.

Experiments by [Kononenko & Zorc 1994] have shown that results of RST when compared to ASSISTANT 86 (an inductive system) and a naive Bayesian classifier are poor, with respect to classification accuracy. This in spite of their having used multiple sets of rules in their implementation of RST.