Introduction to Expert Systems

1. Introduction

As pointed out by the critics of artificial intelligence (e.g., Hubert Dreyfus [1979] in his book What Computers Can't Do?), the prediction of early AI researchers are far from coming true. For example Herbert Simon in 1957 predicted:

1. That within ten years a digital computer will be the world's chess champion, unless the rules bar it from competition.

2. That within ten years a digital computer will discover and prove an important new mathematical theorem.

3. That within ten years most theories in psychology will take the form of computer programs, or of qualitative statements about the characteristics of computer programs.

The first prediction is partially true in the sense that computers can play good chess, although they are not the best players. The second prediction is realized in the sense that new theorems have been proven on the computer. Their level of "importance" depends on your perspective. The third prediction reveals more about the level of human arrogance possible in the field of computer science.

One of the few hard results to come out of the first 20 years of AI research is that intelligence requires knowledge. Search techniques based on non-knowledge-based search or computational logic, were only successful on small well-structured problems, such as games, small games. However, real world complex problems often have characteristics that cause their search space to expand exponentially with the number of parameters involved. For these problems, the "traditional" techniques have generally proved to be inadequate and a new approach was needed. Edward Feigenbaum in a series of lectures in the early 1970's at Carnegie-Mellon is cited as saying:

*You people are working on toy problems. Chess and logic are toy problems. If you solve them you'll have solved a toy problem. And that's all you'll have done. Get out into the real world and solve real world problems.*

In the 1970's a shift from a search for broad general laws of thinking toward an appreciation for specific knowledge occurred. Facts, experiential and intuitive knowledge, and how to use that knowledge became the focus of new research. This new approach emphasized knowledge and problem representation in specific application *domains* and led to the fields of *knowledge*...
2. Definitions

The term "Knowledge Engineering" was coined by Edward Feigenbaum (or perhaps Penny Nii), a pioneer(s) in expert systems and founder of several expert systems companies. Feigenbaum was a pioneer in the field and, in fact, arrived at Stanford when it first opened its computer science department. Knowledge engineering is a discipline devoted to integrating human knowledge in computer systems. The distinctive characteristic of any knowledge-based system is that its processes are state-driven rather than hard-coded. They are characterized by user-invisible inference procedures, resulting in a great degree of data independence. Not only can users represent data in a high-level, human-oriented manner, but they are also spared the effort of describing the operations used to retrieve those data.

The definition of an expert or knowledge-based system has evolved over the years. Rauch [1984] defines knowledge-based systems as:

A class of computer programs intended to serve as consultants for decision making. These programs use a collection of facts, rules of thumb, and other knowledge about a limited field to help make inferences in the field. They differ substantially from conventional computer programs in their goals may have no algorithmic solution, and they must make inferences based on incomplete or uncertain information. They are called expert systems because they address problems normally thought to require human specialists for solution, and knowledge-based because researchers have found that amassing a large amount of knowledge, rather than sophisticated reasoning techniques, is responsible for the success of the approach.

Expert systems are thus software systems that mimic the deductive or inductive reasoning of a human expert. This should not mean that the computer is processing information like the human expert nor is the silicon brain physically organized like the expert system. Expert knowledge in a particular domain not only includes published information and theories, but more importantly includes private knowledge that is not readily available in the literature. This private knowledge consists largely of rules of thumb that have come to be called heuristics. Representing and encoding such knowledge is the central task in knowledge engineering.

2.1 Exercise

List the forms and kinds of knowledge that human experts possess. Answers might include: statistical data, scientific facts and theories, engineering judgment, heuristics, strategic knowledge, holistic knowledge.

3. Motivation

Knowledge is a scarce resource whose effective utilization can create wealth and improvements to the human condition. The transmission of knowledge from human expert to trainee usually requires some combination of education and internship. An expert system that can extract knowledge and make it available to a wider audience can have its economic and social advantages. So say the commercial expert systems consulting and software firms:

- Knowledge is a scarce resource
- Knowledge is power and thus has its price
- Training, internship, and experience takes time and is expensive
Expert systems could reduce the cost of training, synthesis, design, control, diagnostics, ..

- Expert systems make knowledge available to a wider audience and improve performance.
- Expert systems can improve safety of human workers by assisting in environments that may be hazardous to humans.
- It is possible that an expert system that synthesizes the knowledge of many experts could perform in ways that no single expert can.
- The expertise can be codified in permanent form, unlike human experts who may retire, quit, die, etc.

4. Structure of an Expert System

Rule-based expert systems should contain, at the very least, the three components of an AI production system:

4.1 Components of an AI Production System

1. Knowledge Base - Domain-specific facts and heuristics associated with the problem. In a rule-based expert system the knowledge base is composed of a set of production rules. Many existing rule-based systems contain thousands of rules, usually obtained by interviewing experts for weeks or months.

2. General Data Base - Relevant common knowledge, historical information, statistical data engineering coefficients, etc. For example a general data base might include a list of design components or a table of material properties.

3. Inference Engine and Controller: - A reasoning system that acts upon the domain-specific knowledge, general data base and problem-specific input from the user. Inference engines will vary with the form of the knowledge base, including architectures based on the following:

- Backward/forward chaining in systems based on classical logic
• Inheritance in frame/object systems
• Nonmonotonic reasoning and truth maintenance systems
• Probabilistic and Bayesian inference
• Fuzzy logic and fuzzy inference (e.g. minimax rules)
• Dempster-Shafer theory
• Modal logic

4.2 Knowledge Representations

(a) PRODUCTION RULES:

Production rules often take the form of "IF-THEN" rules that read: "if A is true, then B is true with a certainty factor, CF(B,A)". With Boolean or classical logic this certainty factor is not used, all statements are either True or False (nil).

Example:

If Jim's weight (A) is "light" then his height is "short" with a certainty factor of CF(B,A).

(b) SUBJECTIVE PROBABILITY:

The knowledge base might show a probabilistic rather than a deterministic influence in which the states are precisely defined but the inference mapping from one state (or set of states) to another may be uncertain.

If A then B with a specific probability distribution. Influence diagrams/Bayes' belief networks provide an intuitive graphical means for representing probabilistic dependencies and independencies.

Example:
If Jim's weight (A) is "light (² 100 lbs.) then his height (B) is:

³ 7 ft. with a probability of 0.03
³ 6 ft. but < 7 ft. with a probability of 0.07
³ 5 ft. but < 6 ft. with a probability of 0.5
² 4 ft. with a probability of 0.4

(c) FUZZY INFERENCEx:

Or the knowledge might be based on a form of fuzzy logic, in which both the states and the inferences are not precisely defined.

If x has a certain membership in A, then a certain membership in B is implied.

Example:

Jim is "short" with varying degrees of membership depending on height:

³ 7 ft. with a membership of 0
³ 6 ft. but < 7 ft. with a membership of 0.1
³ 5 ft. but < 6 ft. with a membership of 0.6
² 4 ft. with a membership of 1

4.2 Unique Characteristics

An expert system differs from conventional computer programs in that there is a clear separation of general domain knowledge (e.g., rules forming a knowledge base) from information about the current problem (input data) and inference rules to be applied to the knowledge and data bases. In other words, one should be able to view the knowledge base separate from the control or inference strategy of the computer program. The knowledge is in a declarative format for immediate review and evaluation and not embedded in the control logic as in sequential programming. Ideally the knowledge base should be flexible enough so that new knowledge can be added or subtracted from the problem without having to rewrite the program.

4.3 Additional Components of Expert Systems
• **Knowledge Acquisition System:** natural language, graphics, interactive features
• **Input/Output System:** problem-specific input data/advice, explanations

5. General Uses of Expert Systems

- (CATS-1 or DELTA: diesel engine repair, PROSPECTOR)
- Data analysis and interpretation (DENDRAL, PROSPECTOR)
- Monitoring and control (VM-IBM, IDES)
- Planning and scheduling (Intelligent PERT systems, ISIS)
- Consult (Expert Consultant)
- Symbolic manipulation (MACSYMA)
- Design (RI or XCON, HI-RISE, ADIS, PRIDE, SACON, PLASHTRAN)
- Conceptualize (AM, IstPRINCE)
- Crises Decision Making (templates, military applications)
- Instruction, learning, explanation (META-DENDRAL, Guidon)

6. Examples

6.1 DENDRAL

The DENDRAL project was initiated in 1965 shortly after Feigenbaum moved to Stanford [Lindsay, Buchanan, Feigenbaum, and Lederberg: 1980]. It started very much in the spirit of his "Let's stop playing games and start solving real problems" criticism of the AI community during his Carnegie Melon lectures. At Stanford, Feigenbaum met and worked with Nobel laureate, Joshua Lederberg, professor of genetics who shared Feigenbaum's fascination with the possible uses of computers to model scientific thinking. Together they began to write the reasoning programs that could infer molecular hypotheses from chemical data, i.e., to infer the chemical structure of a compound given data available to physical chemists: weight, mass spectrographic data, nuclear magnetic resonance, etc.

A molecule can be considered as an undirected connected graph: nodes represent atoms and edges the chemical bonds between them. Fragmentation corresponds to the breaking of edges in the graph which stands for chemical bonds in the molecule. Atom migrations correspond to the detachment of nodes from one subgraph and their attachment to another subgraph.

![Molecule Graph Representation](image)
A mass spectrometer is a piece of laboratory equipment that works by bombarding a chemical sample with a beam of electrons, which causes the compound to fragment and its components to be rearranged. An analysis of the rearranged fragments gives some indication of the arrangement of the initial molecule. The problem is that any complex molecule can fragment in more than one way under the effect of an external force, say an electron bombardment. Depending on the circumstances and under some random elements, different bonds can break and differing patterns of migration can result. In other words, the effect of electron bombardment is not deterministic. Thus the theory of mass spectrometry is either stochastic in nature or our knowledge is incomplete or both. So how can one use the results?

If you had some program that could systematically enumerate all possible arrangements consistent with the data, you could generate one at a time and eliminate those that failed other theoretical constraints. Unfortunately chemical synthesis is in a class of problems that are exponential in nature (like chess) and becomes unmanageably large very quickly.

DENDRAL employs an efficient variant of generate-and-test in its problem solving strategy and a form of forward chaining in its search technique. The DENDRAL generator can enumerate every possible organic structure that satisfies the constraints apparent in the data by systematically generating partial molecular structures consistent with the data and then elaborating them in all possible ways. By rapidly eliminating implausible substructures (constraints), it avoids an otherwise exponential search. Although knowledge in DENDRAL is represented as a procedural code DENDRAL was interesting as it was one of the first knowledge intensive system developed.

6.2 META-DENDRAL

adds analysis knowledge to DENDRAL by proposing and selecting fragmentation rules for organic structures. It generates and tests possible fragmentations by examining experimental data, retaining those hypothetical rules that prove sufficiently valuable. In other words, META-DENDRAL proposes rules based on specific cases in a training set and generalizes them for application to the general data set. A rule is retained if it will apply frequently but predict incorrect fragmentations very rarely. DENDRAL surpasses all humans at its task and, as a consequence, has caused a redefinition of the roles of humans and machines in chemical analysis.

6.3 MACSYMA

MACSYMA started life as a program called SAINT (1961) and lives now as MACSYMA [Martin & Fateman, 1971]. Richard Fateman has implemented it onto the Berkeley Unix VAX's under the name Vaxima. Like DENDRAL it surpasses most human experts. It performs differential and integral calculus symbolically and excels at simplifying symbolic expressions. It incorporates hundreds of rules garnered from experts in applied mathematics. Originally based on heuristics, symbolic computation programs like MACSYMA employ intricate mathematical algorithms to perform symbolic computation at an impressive level of technical efficiency.

6.4 MYCIN

Mycin addresses the problem of diagnosing and treating infectious blood diseases. Its knowledge comprises over 400 rules relating possible conditions to associated interpretations. In problem-solving, MYCIN tests a rule's conditions against available data or requests data from the physician. If appropriate, it tries to infer the truth or falsity of a condition from other rules. In other words, it checks for a possible inconsistency. With 450 rules, when a panel of experts judged MYCIN's performance of several different agents (including interns, doctors, specialists), its performance was judged as good as or superior to that of all others. In other words, the experts themselves found it difficult to distinguish the MYCIN response from that of a colleague.
MYCIN technology is based on a knowledge base of independent rules of a simple IF-THEN form. It uses backward chaining control strategy in order to produce a parallel explanatory capability in the manner of a doctor. In other words, it takes symptoms as input and reasons backward to predict the cause of the disease.

At Stanford, a domain-independent version of MYCIN was produced, called EMYCIN, that contains the general structure of MYCIN but is domain independent; it does not contain the knowledge base of infectious blood diseases. It was used as the "shell" for many expert systems that followed, including the SACON expert system described in the next section.

**MYCIN SAMPLE RULE:**

**IF**

the stain of the organism is gramneg, and the morphology of the organism is rod, and the aerobicity of the organism is aerobic

**THEN**

there is strongly suggestive evidence (0.8) that the class of the organism is enterobacteriaceae.

One criticism of MYCIN (as indicated by this example) is that you must almost be an expert to use it or understand it. Another major criticism of the MYCIN application is that it calculates the event of disease of maximum likelihood. The user is provided with the relative probability of having one disease compared to another. As we will cover in class, one usually needs absolute probabilistic information in order to make the best medical decision. For example,

Suppose MYCIN tells you that it is likely that you have an infectious blood disease, BAD NEWS, with relative probability of 90% compared to all other diseases considered by MYCIN. The suggested treatment is a blood transfusion every few days for a month. However, there is a 1/10,000 chance of contracting AIDS with this transfusion treatment. If contracted, BAD NEWS is extremely debilitating for a few months but not fatal.

Would the value of "p" make a difference in your decision? Would you follow the suggested treatment if the absolute probability is p=0.99? If p=0.1?
6.5 DELTA

DELTA or CATS-1 or DELTA was developed by the General Electric Company to help railroad maintenance personnel to diagnose maintenance problems and prescribe appropriate maintenance action for GE's diesel-electric locomotives (Bonissone [85]). Prior to the development of this expert system, if a difficult field maintenance problem occurred either GE would fly a maintenance expert to the field or the locomotive would be transported to the maintenance yard where the expertise was available. David Smith, Senior Field Service Engineer was in high demand as a maintenance expert with over 40 years experience solving these problems. This was an ideal application for the current expert system technology in the early 1980's. The experts in this area were in scarce supply, the tasks they performed were significant to the maintenance of an important product and the knowledge was codifiable in rule form.

The DELTA system captured trouble-shooting rules to isolate faults and generate inquiries for the user to investigate an answer. The system was also designed with substantial help facilities, including information on location and identification of components, replacement parts and a description of standard repair procedures. DELTA was interfaced with a videodisk player to display components and assemblies in detail. If requested, it could actually display a training film that showed the steps required for the repair.

6.6 PROSPECTOR

PROSPECTOR uses a knowledge representation similar to MYCIN to model mineral deposit relationships and now includes about a dozen knowledge bases for different kinds of deposits. Like MYCIN, it determines the most plausible "diagnosis" (deposit) given the
"symptoms" (geological data). Components of the production rules are represented as relations between entities and their properties. Examples include AGE, COMPOSITION, FORM, LOCATION, SIZE, and TEXTURE. Examples of unknown entities are kinds and quantities of MINERALS or ROCKS. Again the probabilistic information given is relative, not absolute (note: PROSPECTOR has gone through many iterations at SRI International and some variations use more sophisticated probabilistic techniques). The PROSPECTOR system is not in everyday use. However, SRI has reported that it was able to identify a mineral site of potential commercial significance that had been overlooked by a mining company.

6.7 R1 (or XCON)

R1 or XCON™ is an expert system which configures VAX systems for DEC Digital Equipment Corporation. It is unquestionably a commercial success story and has reportedly save DEC millions of dollars. Its input is a customer's order and its output a set of diagrams depicting the spatial relationships between components, together with components that the customer did not order but are necessary to complete the system. A typical VAX system includes many components from 50 to 150 in the following categories:

CPU

memory control units

peripherals: tape drives, floppy disks, hard disks, size, printers, ...

drivers for the peripherals

cabinets, cables

There are many constraints because most permutations of components are not feasible. Only certain components can be attached to one another, and this limits the possible combinations that can be considered. Knowledge about the interconnections is represented by thousands of production rules. The control strategy is hierarchical and subproblems are solved in order of the importance of their associated goal. XCON™ uses forward reasoning and is written in the OPS5 language. XCON™ is reported to have been used continuously in a production mode since January 1980 [Soloway et al.:1987], processing over 80,000 orders [Jackson:1986,126-141]. It contains over 6,000 rules with knowledge of over 20,000 different components. Despite its size the average consultation takes less than 2 minutes running on a VAX-11/780 and that a significant subset of rules are used (around 50%). About 10% of XCON's solutions exhibit a problem and must be reevaluated by a human expert. However, most of the work can still be utilized even if an error occurs.

XCON Sample Rule:

IF

the most current active context is distributing massbus devices

and there is a single port disk that has not been assigned to a massbus;

and there are no unassigned dual port disk drives

and the number of devices that each massbus should support is known

and there is a massbus that has been assigned at least one disk drive
and that should support additional disk drives

and the type of cable needed to connect the disk drive to the previous device on the disk drive is known

THEN

assign the disk drive to the massbus

LESSONS LEARNED:

McDermott (1982) has a number of interesting observations concerning the knowledge acquisition involved in developing XCON.

- Human experts have a clear idea of the regular decomposition of the main tasks into subtasks, and the temporal relationships between these subtasks.
- Within subtasks, however, their behavior is more driven by exceptions rather than regularities, e.g., when performing subtask x, do y unless z.
- Humans are not very good at recalling these exceptional circumstances on demand, i.e., they are driven by results.

Jackson [86:140] reports that the speed performance of XCON using a Lisp pattern-matching algorithm outperforms PROLOG even if parallel processing is considered.

6.8 DART

DART diagnoses faults in computer hardware systems by using first principle information embedded in the model of the design of the device being diagnosed. The system has been applied to simple computer circuits. The system uses resolution theorem proving to attempt to generate a proof related to the cause of the device’s malfunction. Developed at Stanford University.

6.9 STEAMER

STEAMER is an intelligent Computer-Aided-Instruction (CAI) system designed to instruct Navy personnel in the operation and maintenance of propulsion plant for a 1078-class frigate. A qualitative simulation of the propulsion plant is linked to a graphical interface program that displays animated color diagrams of the plant subsystems, including simulated components like valves, switches, and pumps. The system can monitor the students performance on the system, acknowledging appropriate student actions and correcting inappropriate ones. The qualitative simulation allows the student to observe the impact of changing some of the plant parameters, such as temperature and pressure (Turban [88:375]). More details on STEAMER will be provided in the Chapter titled "Qualitative Reasoning". STEAMER was developed by the U.S. Navy in cooperation with Bolt, Beranek, and Newman (Cambridge, MA).

6.10 Campbell Soup

Campbell Soup worked with Texas Instruments (TI) Corp. to develop an expert system to diagnose malfunctions that can occur in cooker systems, called "hydrostatic sterilizers." Although campbell plant operators and maintenance personnel were skilled at handling the day-to-day problems with the sterilizers, there were occasional failures that required the assistance of experienced experts that were also knowledgeable about the design, installation and installation of the sterilizers. An
evolutionary approach was used in the development of the expert system. The initial prototype contained 32 rules, growing to 151 rules in its implementation plus start-up and shut-down procedures. An example of an English translation of a rule (Turban [88:616-617]) is given below:

\[
\begin{align*}
\text{IF} & \quad \text{the cooker's symptom is TEMPERATURE-DEVIATION, and the problem temperature is T30-INTERMEDIATE-COOLING-SPRAY, and the input and output air signals for TIC-30 are correct, and the valve on TCB-30 is not open,} \\
& \quad \text{THEN the problem with the cooker is that TCV-30 is not working properly. Check the instrumentation and the air signal.}
\end{align*}
\]

**QUESTION:**

XCON™ was originally named R1. How did R1 get its name?

7. Exercise

Question (a) Is Kermit a living organism?

Answer this question by applying rules 1-6 below.

(1) Plants reproduce themselves.

(2) All entities that reproduce themselves are living organisms.

(3) A frog is an animal.

(4) All animals reproduce themselves.

(5) Viruses reproduce themselves.

(6) Kermit is a frog.

Question (b) Is a virus a living organism?

Exercise Write a Fortran program that will answer questions (a) and (b).

8. References:


Lindsay, Buchanan, Feigenbaum, and Lederberg: 1980


