The Knowledge Engineering Review, Vol 6:3, 1991, 195–222

A review of learning

S. KOCABAS

Marmara Scientific and Technological Research Centre, PK 21, Gebze, Kocaeli, Turkey

Abstract

Learning is one of the important research fields in artificial intelligence. This paper begins with an outline of the definitions of learning and intelligence, followed by a discussion of the aims of machine learning as an emerging science, and an historical outline of machine learning. The paper then examines the elements and various classifications of learning, and then introduces a new classification of learning based on the levels of representation and learning as knowledge-, symbol- and device-level learning. Similarity- and explanation-based generalization and conceptual clustering are described as knowledge level learning methods. Learning in classifiers, genetic algorithms and classifier systems are described as symbol level learning, and neural networks are described as device level systems. In accordance with this classification, methods of learning are described in terms of inputs, learning algorithms or devices, and outputs. Then there follows a discussion on the relationships between knowledge representation and learning, and a discussion on the limits of learning in knowledge systems. The paper concludes with a summary of the results drawn from this review.

1 Introduction

Learning can be defined in general terms as the processes in which intelligent systems increase their knowledge and improve their skills. Learning processes include the acquisition of new common-sense, descriptive, definitive, technical and theoretical knowledge, the development of skills, organization of new knowledge into general, effective representations, and the discovery of new facts and theories through observation and experimentation.

Simon (1983a) defines learning as follows: "Learning is any change in a system that allows it to do more efficiently and more effectively on repetition of the same task or on another task drawn from the same population. The change should be more or less irreversible so that learning doesn't disappear rapidly". He points to the relationship between learning and discovery, and states that there can be all kinds of learning without discovery, such as changing qualitative parameters. A large part of the research effort in the domain of machine learning is directed at machine discovery. There is also a connection between learning and understanding, which includes the whole natural language problem. Most of what we learn we get from other people, communicated to us in natural language.

Michalski (1986) defines learning as constructing or modifying representations of what is being experienced. This definition shifts the reference point of learning from Simon's "performance" to "representation of experience". Lenat (see Michalski et al., 1986), on the other hand, proposes "synergy" as an indication of learning, defining synergy as getting more out than what is put in a system in terms of its knowledge content.

Since learning, understanding and intelligence are closely related concepts, to introduce a little more scope into the definitions of learning, a few words about intelligence would be in order. Lenat & Feigenbaum (1987) define intelligence in terms of search as the power to rapidly find an adequate solution in what appears a priori (to observers) to be an immense search space. However,

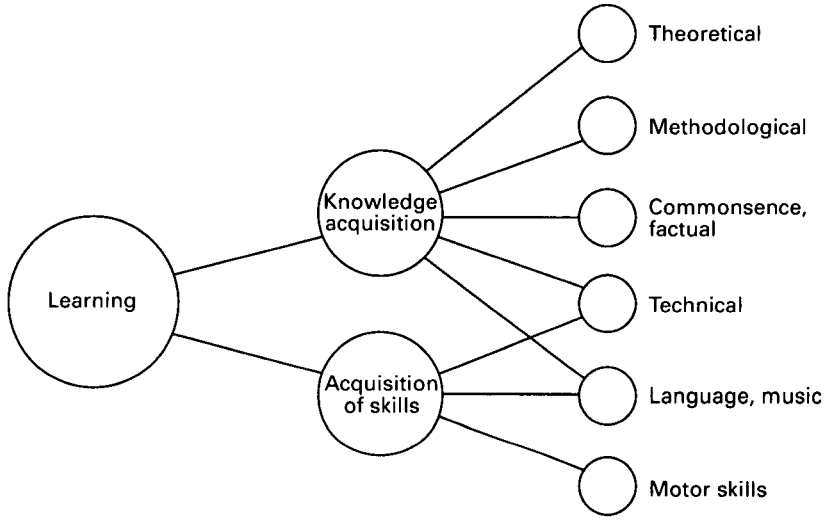


Figure 1 Basic forms of learning and types of knowledge or skill learned. The diagram suggests that different types of knowledge may require different learning methods

Feigenbaum (1989) has recently redefined intelligence as the ability to assemble the useful pieces of knowledge to perform a certain task, or "knowledge assembly", rather than search.

Fatmi & Young (1970) define intelligence in physical and biological terms as the ability to perceive order in a system previously believed to be disordered. On can also define intelligence in terms of basic computational and physical concepts as follows: "Intelligence is the ability to generate or understand a set of instructions which can bring order to a previously disordered system".

1.1 Machine learning

The study and computer modelling of learning processes constitutes the subject matter of machine learning. There are two fundamental reasons for studying learning: to understand the process itself, and to provide computers with the ability to learn. It has long been a goal of artificial intelligence to develop computer systems that could be taught rather than programmed. Understanding human learning well enough to reproduce aspects of that learning is in itself a worthy scientific goal. Computational modelling of learning processes can help in the understanding of human learning. Gaining insights into the principles underlying human learning abilities is also likely to lead to more effective educational techniques.

One basic scientific objective of machine learning is the exploration of alternative learning mechanisms, the scope and limitations of certain methods, the information which must be available to the learner, the issue of coping with imperfect training data, and the discovery of general techniques applicable in many task domains. By constructing and testing learning systems, one can determine the effectiveness and limitations of particular approaches to learning. Another aim of machine learning is to build systems which will function as intelligent research assistants or intelligent personal assistants (Buchanan & Feigenbaum, 1978; Michalski, 1986).

Learning can be viewed in two basic forms (Carbonell et al., 1983): knowledge acquisition and skill refinement. Knowledge acquisition is a conscious process, and involves the acquisition of commonsense, theoretical, methodological and technical knowledge. Knowledge is acquired in the forms of descriptions, models, schemas and rules (see Fig. 1).

Skill refinement involves a gradual improvement of motor and cognitive skills through practice, such as learning to ride a bicycle. In contrast to knowledge acquisition, it mostly takes place at a lower cognitive level. As illustrated in Fig. 1, there can be overlaps between knowledge and skill acquisition. Carbonell et al. (1983) consider that knowledge acquisition belongs in the field of

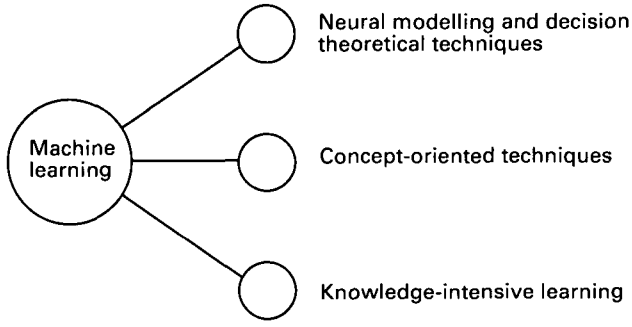


Figure 2 Historical classification of machine learning. The progress is in clockwise direction

artificial intelligence research, while skill refinement comes closer to sub-symbolic processes, such as studied in adaptive control systems.

1.2 Historical outline of machine learning

Research in machine learning has evolved in three stages (see Carbonell et al., 1983), each centred around a different model: (1) neural modelling and decision theoretic techniques: (2) symbolic learning; and (3) knowledge intensive learning. Fig. 2 summarizes this historical process.

According to the understanding in the first stage, learning systems start with little or no initial structure or knowledge. The major thrust based on the *tabula rasa* approach involved building neural nets or self organizing machines, with random or partially random initial structure. Learning in these systems consisted of incremental changes in the probabilities that threshold logic units would transmit a signal. The hope was that if a system were given a set of stimuli, a source of feedback and enough degrees of freedom to modify its own organization, it would adapt itself toward an optimum organization. Attempts were made, for example, to simulate evolution in the hope that intelligent programs would result from processes of random mutation and natural selection (Cohen & Feigenbaum, 1982, p. 325). Various analogues of neurons were developed and tested, such as the perceptron (Rosenblatt, 1958). Related research involved the development of classifier systems (Nilsson, 1965) and the mathematical theory of genetic algorithms (Holland, 1975).

Practical results sought by the earlier neural modelling and decision theoretic approaches met with limited success. Subsequent theoretical studies have revealed strong limitations of the "knowledge free" perceptron-type learning systems (Carbonell et al., 1983). However, the interest has been rekindled by the recent developments in neural networks (e.g., see Barrow, 1989).

In the second stage, concept oriented learning began to emerge in the early 1960s. This model used semantic nets or predicate logic rather than numerical or statistical methods. In concept acquisition, the system learns by constructing a symbolic representation of a given set of concepts through the analysis of examples and counterexamples of these concepts. Related work includes research on human concept acquisition (Hunt & Hovland, 1963; Feigenbaum, 1963; Simon & Lea, 1974). During this stage of research on learning, it began to become clear that learning is a complex process and that, consequently, a learning system cannot be expected to learn high-level concepts by starting without any knowledge at all. Winston's (1975) work was influential in this development.

The third stage represents the recent period of research starting in the mid 1970s. In this stage, researchers have widened their interest beyond learning isolated concepts from examples, and have begun to investigate a wide range of learning methods most of which are based on knowledge-rich systems. It was recognized that to acquire new knowledge a system must already possess a great deal of initial knowledge. In addition to the earlier work on learning from instructions, researchers began to investigate a wider variety of methods such as inductive learning, learning from observation and discovery and learning by analogy.

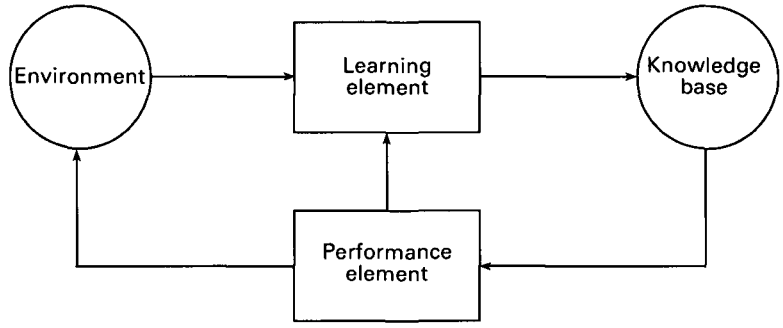


Figure 3 Elements of learning based on Simon's model

In contrast to previous efforts, a number of current systems incorporate methods to control their focus of attention by generating research tasks; to propose experiments to gather data; formulate new concepts and discover previously unknown regularities in data; draw interesting analogies; automatically learn problem-solving heuristics; develop generalized plans for achieving a goal; generate, test and verify hypotheses; resolve contradictions; and to develop useful theoretical concepts. Such systems include AM (Lenat, 1979), EURISKO (Lenat, 1983), BACON (Langley, 1981), IDS (Nordhausen & Langley, 1987), PI (Thagard & Holyoak, 1985), KEKADA (Kulkarni & Simon, 1988), and CYC (Lenat et al., 1986; Lenat & Guha, 1990).

2 Elements of learning

Simon (1983b) introduces a useful model of learning systems (see also Cohen & Feigenbaum, 1982), according to which the constituents of learning are: environment, the learning unit, knowledge base and the performance unit (see Fig. 3). The environment supplies some information to the learning unit which uses this information to make improvements in a knowledge base, and the performance unit uses this information to perform its task. The information gained during the activity can serve as feedback to the learning unit, and can change the environment. This simple model allows us to classify learning systems according to their relationships to these elements.

The degree of generality of abstraction and the quality of the information supplied by the environment is an important factor in designing learning systems. General or abstract information is relevant to a broad class of problems while specialized information is relevant to a single problem. The task of the learning unit can be viewed as the task of bridging the gap between the levels of generality at which the information is provided by the environment and the level at which the performance unit can use the information to carry out its function.

The learning unit must be able to form hypotheses to bridge the gap between the levels of generality, and it must receive some feedback that allows it to evaluate its hypotheses and revise them if necessary. Accordingly, if the learning system is given general or abstract advice about its performance task, it must fill in the details so that the performance unit can use the information in particular situations. Correspondingly, if the system is given very specific information, the learning unit must generalize this information into a hypothesis or rule that can be used by the performance unit in a broader class of situations.

Machine learning can be classified in several different ways regarding the underlying strategies, representation of knowledge, application domain, and according to the levels at which the outcomes and the methods of learning are describable.

The main criterion in the classification of learning by underlying strategy is inference. Some learning strategies do not use inference, while some others use a substantial amount of inference. As the amount of inference the learner is capable of performing increases, the burden placed on the teacher or external environment decreases. Fig. 4 shows general strategies of learning as: (1) rote

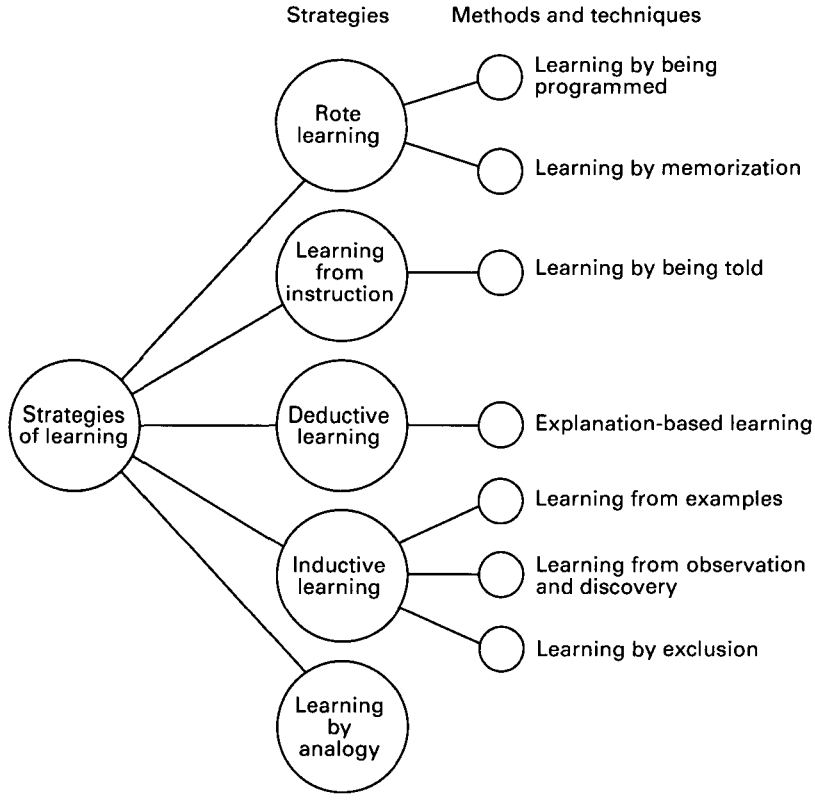


Figure 4 Classification of learning by underlying strategies and methods

learning; (2) learning form instructions: (3) deductive learning; (4) inductive learning; and (5) learning by analogy.

In rote learning, the environment provides information exactly at the level of the performance task, and thus no inference is needed. Rote learning can be viewed in two distinct forms: learning by being programmed, and learning by memorization. Samuel's checkers player (Cohen & Feigenbaum, 1982, p. 332) learns its skills by this method in addition to generalization (Akl, 1987).

In learning by being told, the information provided by the environment is general or abstract, and the learning unit must perform some inference to fill in the details. One of the tasks of machine learning is to build systems that can accept instruction or advice, and which can store and apply this knowledge effectively. The operation of learning systems which use this strategy can be divided into five steps:

(i) request advice from an expert,
(ii) assimilate advice into the internal representation;
(iii) convert the instruction into usable form;
(vi) integrate it into the knowledge base;
(v) evaluate the resulting actions of the performance unit.

Examples are the TEIRESIAS and FOO programs (see, e.g., Cohen & Feigenbaum, 1982, p. 333).

Deductive learning has been recognized as a distinct strategy more recently. This strategy includes knowledge reformulation, knowledge compilation, creation of macro operators, chunking and other truth preserving transformations, and explanation-based learning. Some computational systems that use deductive methods are SOAR (Laird et al., 1986; 1987), LEX (Mitchell et al., 1986), and PET (Porter & Kibler, 1986).

Inductive learning includes learning from examples, learning from observation and discovery. In learning from examples, given a set of examples and counterexamples of a concept, the learner induces a general concept description. This strategy includes learning methods such as similarity-

based learning (Mitchell, 1983; Michalski, 1983), and conceptual clustering (Michalski & Stepp, 1983). Learning with genetic algorithms (Holland, 1986) may also be included in this category. Systems such as ID3 (Quinlan, 1983), and CLUSTER/2 (Michalski & Stepp, 1983) can be regarded as inductive learning systems.

Learning from observation and discovery is also called unsupervised learning, and includes methods such as qualitative and quantitative discovery, theory formation, theory revision, and the creation of classification criteria to form taxonomic hierarchies. Learning from observation and discovery combines different inference methods such as induction, deduction, abstraction, and abduction (or retroduction—see Darden, 1987). This form of learning requires more inference than any other learning strategy discussed so far. Learning from observation can be divided into two subclasses according to the degree of interaction with the environment as passive observation and active experimentation. Examples of systems which learn by discovery are, AM (Lenat, 1979), EURISKO (Lenat, 1983), BACON (Langley, 1978), STAHL (Zytkow & Simon, 1987), STAHLp (Rose & Langley, 1986), GLAUBER (Langley, et al., 1987), IDS (Nordhausen & Langley, 1987), KEKADA (Kulkarni & Simon, 1988), BR-3 (Kocabas, 1990), AbE (O'Rorke et al., 1990) and COAST (Rajamoney, 1990).

In learning by analogy (e.g. see, McDermott, 1979; Falkenheiner, 1987; Carbonell, 1982; Greiner, 1988) the information provided by the environment is relevant only to a similar performance task. The learning system must discover the analogy and form analogous rules and hypotheses for its present performance task. Learning by analogy requires different kinds of inference on the part of the learner. It combines deductive and inductive methods. Finding the common substructure involves inductive inference, whereas preforming analogical mapping is a form of deduction.

The above classification of learning strategies helps one to compare various learning systems in terms of their strategies and methods and the types of the external information they use. It also helps to identify the gaps in our knowledge on learning strategies and methods, and to conduct research in those areas. In any act of human learning, a mixture of these strategies is usually involved. Most current research focuses on a single learning strategy, but machine learning research has to give increasing attention to multistrategy systems.

As to the classification of learning systems according to the types of knowledge they can acquire, these types can be listed as: parameters in algebraic expressions (perceptrons, neural networks, classifier systems), decision trees (simple expert systems), formal grammars, production rules (rule-based expert systems), predicate logic expressions, graphs and networks, frames, schemata and taxonomies (frame-based systems). Each type is more or less associated with a particular form of representation.

3 Methods of learning at different levels

Methods of learning can be loosely divided into three levels, in a slightly different way from Newell's (1982) classification[1] as follows: (1) knowledge-level learning; (2) symbol-level learning; (3) device-level learning (see Fig. 5). In this classification, methods of learning are viewed in accordance with the levels of representation and the levels at which the outcomes are described.

In knowledge-level learning, the method and the outcome of learning can be described without any reference to a specific symbolism or device. In symbol-level learning, the method and outcome of learning can be described in terms of a particular type of symbolism or data structure, without reference to any particular device. In device-level learning, on the other hand, the methods and outcomes of learning can only be described in reference to a particular device.

Learning methods such as abstraction, abduction, analogy, similarity-based generalization (Michalski, 1986), explanation-based generalization (Mitchell et al., 1986) and conceptual clustering (Michalski & Stepp, 1983) can be included in knowledge-level learning; classifiers (Nilsson,

[1] See also Dietterich (1986) for learning at the knowledge level.

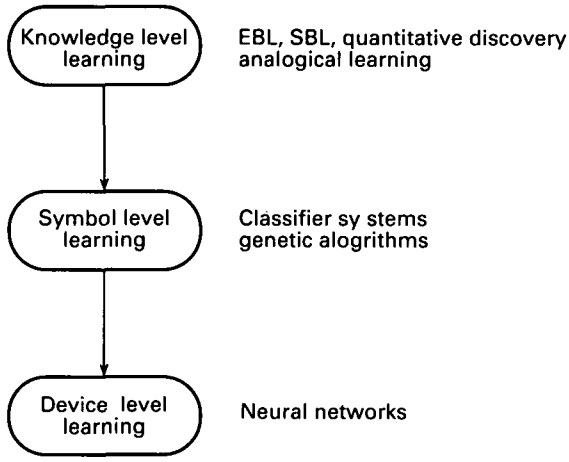


Figure 5 Learning in machines can be considered at different levels depending on the level the method and the outcome of learning can be expressed. Higher level learning methods can control the lower level ones, as shown by the arrows

1965) and genetic algorithms (Holland, 1975) in symbol-level learning; while learning in connectionist systems (Rosenblatt, 1958) and neural networks (Barrow, 1989) can be considered as device-level learning. This classification is not clear cut, as there may be overlaps between the stated levels concerning some methods. Nevertheless, it may be useful in clarifying some of the important differences between various methods of learning. Intelligent systems can be designed to successfully integrate these different levels of learning.

Carbonell and Langley (1987) describe various methods of machine learning in several task-based categories. These are: learning from examples, learning search heuristics, learning by analogy, grammar acquisition, and learning by discovery. Carbonell et al. (1983) provide three different classifications of machine learning methods according to the underlying learning strategy, the types of knowledge learned, and by domain of application. In the present work, however, learning methods are not examined in task-based categories, underlying strategies, the types of knowledge learned, or the domain of application, but according to the levels defined above as knowledge level, symbol level and device level learning methods. The purpose of looking at learning methods in this way is to explore the vertical integration of these methods.

3.1 Knowledge level learning methods

Learning methods at this level include abstraction, abduction, analogy and inductive and deductive methods. In this section only three knowledge level methods are described. These are similarity-based learning, explanation-based learning and conceptual clustering.

3.1.1 Similarity-based learning

Similarity-based generalization (see, e.g., Dieterich & Michalski, 1983; Michalski, 1986; Mitchell et al., 1986) searches for features shared among examples (facts or objects) in a class, and looks for common causes and explanations of why different examples belong to the same class. It generalizes over the differences between examples either by ignoring the differing features or by formulating concepts that encompass the differences.

This method employs some kind of inductive bias to guide the inductive leap for defining a concept from a subset of examples. This bias is typically built into the generalizer by only providing it with knowledge of those example features that are presumed relevant to describing the concept to be learned. Because this method is based on searching for features that are common to the training examples, it is called similarity-based generalization.

Similarity-based generalization can be summarized in symbolic terms as follows: let $a_1, \ldots, a_m$ be individuals (e.g., raven, pigeon, ostrich), C is the class (e.g., bird) to which these individuals

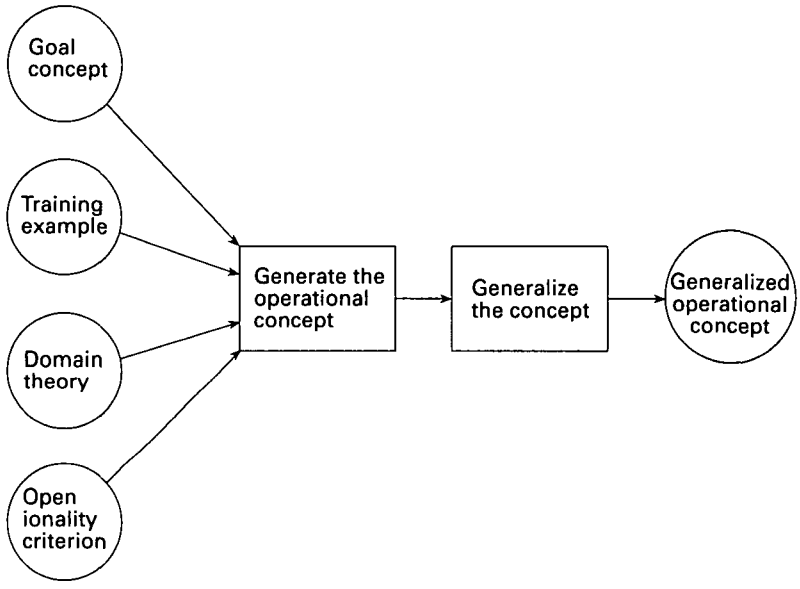


Figure 6 The processes of explanation-based generalization

belong, P is the set of features or properties (e.g., having feathers, wings, two legs) these individuals may have, and P_c is the set of common properties:

$P_c \, e \, P$
$C(a_1), \ldots, C(a_m)$
$P = [P_1, \ldots, P_n]$ (the set of the properties of the individuals)
$P_c = [P_i, \ldots, P_k]$ (the set of common properties)

then the generalized concept is defined as $C_g(x) = P_c(x)$.

Similarity-based learning has been used in learning blocks world concepts (see Dietterich & Michalski, 1983). Meta-DENDRAL (Buchanan & Feigenbaum, 1978) uses this method in discovering the rules of chemical analysis by mass spectrometry.

Similarity-based generalization is an empirical, data-intensive method that relies on large numbers of training examples to constrain the search for the correct generalization. A deductive learning method that overcomes the limitations of similarity-based generalization is described next.

3.1.2 Explanation-based learning

Explanation-based generalization (Mitchell et al., 1986), puts the emphasis on the role of explanatory knowledge. It applies a system's background knowledge to formulate a high-level conceptual explanation of a given fact or event.

Unlike similarity-based generalization, which relies on many training examples and an inductive bias to constrain the search for a correct generalization, explanation-based generalization constrains the search by relying on knowledge of the task domain and of the concept under study. After analyzing a single training example in terms of this knowledge, this method is able to produce a valid generalization of the example along with a deductive justification of the generalization in terms of the system's knowledge. More precisely, explanation-based generalization analyzes the training example by first constructing an explanation of how the example satisfies the definition of the concept under study. The features of the example identified by this explanation are then used as the basis for formulating the general concept definition. The justification for this concept definition follows from the explanation constructed for the training example.

Explanation-based generalization requires the following: (1) a definition of the concept under study (goal concept); (2) a training example; (3) a domain theory; and (4) a description of the form in which the learned concept must be expressed, i.e., the operationality criterion (see Fig. 6). The

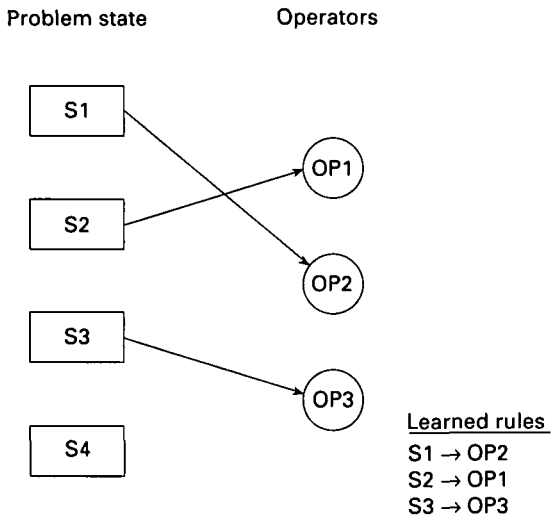


Figure 7 LEX learns search heuristics by associating problem states with operators

explanation-based generalization system must be able to explain to itself why the training example is an example of the concept under study. A concept definition describes the necessary and sufficient conditions for being an example of the concept.

Explanation-based generalization can be characterized in symbolic terms as a relation involving the generalization of an operational explanation and the generalized explanation as follows:

$$L(f(C,e,T,O),E), \quad \text{or}$$
$$E = gen(f(C,e,T,O)),$$

where C is the goal concept, e is the training example, T is the domain theory, O is the operationality criterion, and E is the explanation. The product of learning by explanation is a generalized concept description.

Upon the introduction of the training example e, the system finds a string of explanations S, which constitutes the proof that $C(e)$, i.e., e is an instance of the goal concept C. S is determined by a function f as

$$S = f(C,e,T,O).$$

Explanation-based generalization overcomes the main limitation of similarity-based methods, which is their inability to produce justified generalizations. This is accomplished by assuming that the learner has available knowledge of the domain, the goal concept and the operationality criterion, whereas similarity-based generalization does not rely on any of these inputs.

Explanation-based learning has also been applied to learning search heuristics, as in the LEX system (Mitchell et al., 1986, p. 61). The system learns search heuristics by associating problem states with operators, which are both given (Fig. 7). Learning of this kind is in fact learning how to associate a problem state with an operator, and is different from real rule discovery, such as exhibited by Lenat's (1983) EURISKO.

The LEX system has been applied to learning how to simplify integral expressions in mathematics. The program has a set of operators, each of which performs an operation on integral expressions. For example, its OP3 operator moves constants outside the integrand. The goal concept of LEX is to find a class of integrals to which a rule such as

$$\text{R3:} \quad \int rf(x)\,dx \rightarrow r \int f(x)\,dx$$

which transforms the integral expression by moving constants outside the integrand, can be applied. When the system is given an integral expression

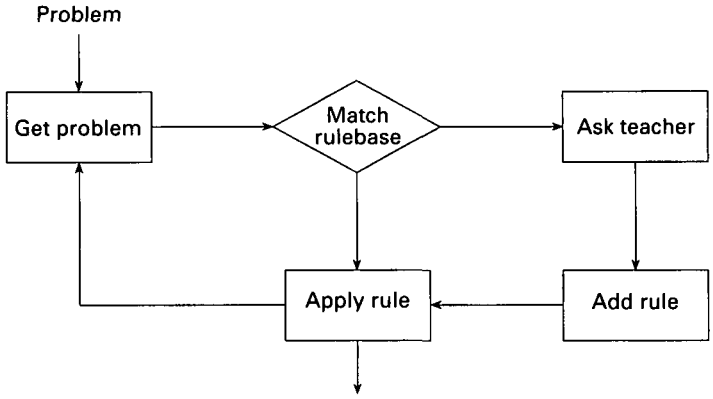


Figure 8 PET's control structure in episodic learning

$$\int 7x^2\,dx.$$

When OP3 is applied to this problem state, as the training example, it transforms it to

$$7\int x^2\,dx.$$

LEX must find an operational generalization of $7x^2\,dx$ which can be associated with OP3.

An important limitation of explanation-based learning is that the learning depends strongly on what the learner already knows. One consequence of this is that the degree of generalization produced for a particular training example will depend strongly on the generality with which the rules in the domain theory are expressed. Explanation-based learning is subject to a few problems: (1) the incomplete theory problem; (2) the intractable theory problem; (3) the inconsistent theory problem. Accordingly, if the domain theory is incomplete, intractable or inconsistent, explanation-based generalization cannot produce operational concept descriptions. An integration of similarity and explanation-based methods may overcome the weakness of each method, but this is an area that needs to be explored.

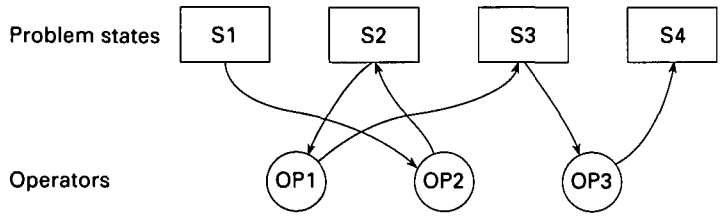
An improved version of explanation-based learning is episodic learning (Porter & Kibler, 1986). In this method the system learns how to associate not only problem states to operators, but also to a sequence of operations in which the operators are used. In this way, it constitutes an integration of learning and problem solving. Episodic learning constructs sequences of action-rules, where each rule is of the form

state-description → operator.

These rules are learned incrementally by explanation-based generalization and integrated into episodes. Learned episodes are then applied by the problem solver to guide the search for problem solutions. They are used by the problem solver to suggest solution paths. Episodic learning is particularly useful in learning planning sequences.

Porter & Kibler (1986) have applied episodic learning in a system called PET written in Prolog. When PET forms its operational rules, it assigns a score to its rules. Preceding rules in a sequence have higher scores. PET starts with a problem to solve, a goal to be achieved and a set of operators which can be applied to the problem. When presented with a problem, PET searches for an applicable rule. Conflicts are resolved in favour of the rule with the highest score. Fig. 8 shows PET's control structure.

PET has been applied to learning how to solve linear equations and integral expressions. In its application to simultaneous linear equations, it has several operators. Two of these operators combine the variable terms in an equation. Another one combines the constant terms in the equation. Another operator deletes the terms with zero coefficient in the equation. The remaining



Learned rule: S1 → OP2 → S2 → OP1 → S3 → OP3 → S4

Figure 9 PET's association of problem states to operators in episodes. The program assigns scores to the successful episodes. The scores are increased by the number of successful episodes in a sequence

operators carry out linear algebraic operations such as differencing, adding and multiplying equations with a constant. In this domain, PET's goal is to simplify the problem state by reducing the number of terms in the equations. This allows the problem solver to recognize progress before completely solving the problem. Initially, PET has an empty rule base of knowledge concerning when to apply each operator.

Each operator achieves a simplification, and the rules that are learned are added to the knowledge base. In this way, the system forms a network of rules for its operators which are used in solving problems in multiple linear equations. PET does not form a rule unless that rule can be integrated into an existing episode. The advantage of this restrictive policy is that only those rules with known utility are added to the rule base. Another advantage is that PET is not confused by bad advice from the teacher. In the domain of symbolic integration, PET uses 18 primitive operators to form its rules. Fig. 9 illustrates how PET associates problem states to operators in a sequence.

The major limitation of episodic learning is that the rules are overly specific. PET generalizes its rules by perturbation, which automates the generation of training instances by making small changes to a single training instance supplied by the teacher. Modifications performed by the perturbation operator of PET are of two types: replace a feature by a null feature, or replace a feature by its sibling in a class which represents a concept of the problem domain, and assume that the perturbed instances have the same outcome. In the field of multiple linear equations, these features are the coefficients in the equations.

3.1.3 Conceptual clustering

Constructing meaningful classifications of given objects or phenomena is an important problem in science. The process of constructing classifications is a form of "learning from observation". Conceptual clustering (Michalski & Stepp, 1983) is a method of classifying objects. It differs from the traditional techniques based on cluster analysis and numerical taxonomy in the way the clusters are formed. In conceptual clustering a configuration of objects form a class only if it is describable by a concept from a predefined set of concepts. In addition, conceptual clustering classifies objects into hierarchies to form conceptual networks. The problem of classification in conceptual clustering can be defined as follows:

> Given: (1) a set of objects; (2) a set of attributes to be used to characterize the objects; (3) the number of clusters desired; (4) knowledge of the problem constraints, properties of attributes and a criterion of evaluating the quality of constructed classifications.
>
> Find: a hierarchy of objects classes, in which each class is described by a complex concept (usually disjunctive), based on a set of attributes. Subclasses of the parent class and their subclasses should have a similar description and should optimize an assumed criterion.

Structuring objects into such hierarchies is called conjunctive conceptual clustering if the complex concepts which describe the clusters are conjunctive. In conventional data analysis, the similarity

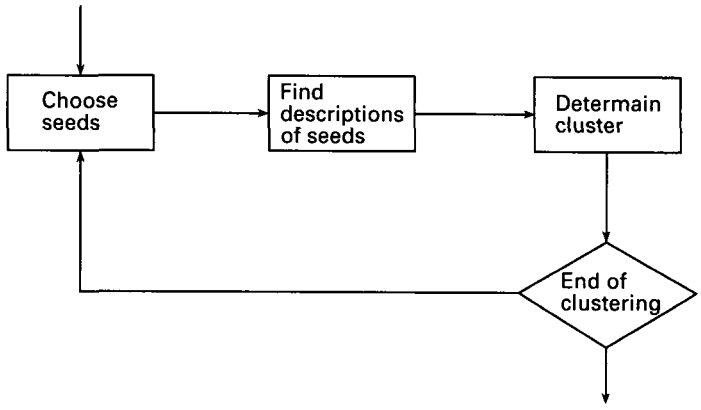


Figure 10 Controls of CLUSTER/2

between any two objects is characterized by the value of a similarity function applied to the symbolic descriptions of objects. In conceptual clustering, the similarity between events a and b is called conceptual cohesiveness, which can be defined as follows:

$$conceptual\text{-}cohesiveness(a,b) = f(a,b,E,C),$$

where E is the surrounding events and C is a set of concepts which are available for describing a and b together.

In CLUSTER/2 (Michalski & Stepp, 1983), the best-known conceptual clustering system, objects are represented by "events". An event is an object description in the form of a vector of values of a set of discrete variables (e.g., properties). CLUSTER/2's operation is as follows:

(1) From a given collection of events E, the initial seeds are selected randomly or according to a criterion.
(2) Find some description for each seed by employing a characterization method.
(3) Using each such description determine a cluster for the seed on which it is based.
(4) For each cluster select a new seed and repeat the process.

The system repeats this process for different sets of seeds giving alternative partitions of the object set each with associated descriptions. The descriptions are used in deciding among the competitors, and the best partition is used to create the first branches in the taxonomic hierarchy. CLUSTER/2 then applies the above process recursively to each of the resulting classes, finding partitions of each class and providing the partitions with descriptions. Each such partition leads to additional branches at lower levels of the tree, and this process of subdivision continues until further partitions cease to provide useful sub-classifications of the data. Fig. 10 summarizes the operation of CLUSTER/2. The system has been applied to the classification of Spanish songs. A simpler form of conceptual clustering has also been implemented in GLAUBER (Langley et al., 1987) for the classification of chemical substances into functional groups such as acids, alkalis and salts.

In knowledge level learning systems, deductive and inductive methods are directly applied to high level representations of knowledge, whereas symbol level systems cannot operate on such high level representations. They need input and output interfaces to translate them to lower level representations. Knowledge level systems can provide data directly acceptable to symbol level systems, and can use the output of the latter. In this way they can control the operations of the symbol level systems.

3.2 Symbol level learning methods

Symbol level systems differ from knowledge level systems basically in their representation of knowledge learning methods. They utilize a fixed alphabet, and fixed length vectors, matrices or

Qualities	Values of an unknown e	Weights e_1	e_2
q_i	v_i	w_{1i}	w_{2i}
Has wings	1	1	1
Has engine	0	0	1
Has feathers	1	1	0
S_i		2	1

Figure 11 A classifier system that decides whether an object e is a bird or a plane by the values v_i given to the qualities q_i. If the values are (1,0,1), the classifier instantiates e to "bird", for this gives the maximum $\Sigma\, v_i \cdot w_{ji}$. The training of this system is carried out by providing the values of a known e and increasing w_i by v_i on successful guesses

ordered lists in their knowledge representation, while knowledge level systems can use a much more flexible syntax and an indefinite set of symbols. In this subsection two symbol level learning methods are explained: classification and genetic algorithms.

3.2.1 Learning in classifiers

Classifiers (Nilsson, 1965; Hunt, 1975) are inductive learning systems. In these systems knowledge is represented as vectors or matrices of values to a set of parameters. Because they are low level learning systems, complex input and output interfaces may have to be incorporated with them for translating external knowledge into internal representation, and internal representation into externally readable form.

Classifiers are applicable to problem solving in areas with incomplete and/or inconsistent domain theory. They are particularly useful in conflict resolution problems. The problem state of a classifier can be described as a relationship.

$$P(v_i,w_{ij}), \quad i = 1, \ldots, n; \quad j = 1, \ldots, m,$$

where v_i and w_{ij} correspond to the values of the domain features q_i and domain events e_j. To define the goal state, let S_j be defined as

$$S_j = \sum_{i=1}^{n} v_i w_{ij}.$$

The goal state is to find an event e such that

$$f(max(S_j)) = e, \quad j=1, \ldots, m.$$

In such a classifier, learning consists of finding a vector of values $(v_1, \ldots, v_n)$ which satisfies the criterion given above for each training example. Fig. 11 illustrates how these ideas are applied in a classifier. A slight variation of the above relationship formulates the problem states as

$$P'(v_{ij},w_i) \quad i = 1, \ldots, n; \quad j = 1, \ldots, m.$$

$$S'_j = \sum_{j=1}^{m} v_{ij} w_i.$$

In this case, the goal is to find an event e such that $f'(max(S'_j)) = e$. Similarly, in this case learning consists of finding a matrix $((w_{11}, \ldots, w_{1n}), \ldots, (w_{m1}, \ldots, w_{mn}))$ which satisfies the criterion for the training examples. Fig. 12 illustrates how this version can be applied. Depending on the dimensions of their evaluation matrix, classifiers are divided into two broad categories as linear and

Qualities q_i	Values of known e's e_1 v_{i1}	e_2 v_{i2}	e_3 v_{i3}	e_4 v_{i4}	Weights w_i
Available	1	1	0	1	1
Cheap	0	1	1	1	1
Flammable	0	0	0	1	-2
Radioactive	0	0	1	0	-5
S_j	1	2	-4	0	

Figure 12 A classifier that finds for a chemical experiment, the best material e among (e_1,e_2,e_3,e_4) by the values of q_i and by the weights given to each q_i. In this case, the classifier instantiates e to e_2, for this corresponds to maximum $\Sigma\ v_{ij} \cdot w_i$. The training of this system is carried out by providing tuples of examples e_j,e_k and their values v_{ij},v_{ik}, and by increasing w_i by the values of the correct e

nonlinear classifiers. Examples of how classifiers can be integrated with knowledge level learning and discovery methods can be found in Kocabas (1989).

3.2.2 Learning in genetic algorithms

Genetic algorithms (Holland, 1975) are adaptive and general purpose search techniques based on the principles of population genetics. They consist of a set of genetic operators which act upon fixed-length strings of symbols. Basic genetic operators[2] are mutation, crossover and reproduction. Mutation is the random alteration of the contents of a string position; crossover simply exchanges a randomly selected segment (allele) between the pairs of strings; reproduction copies individual strings according to their values (fitnesses).

A simple example to the implementation of genetic algorithms in learning can be given as follows. Consider an ordered set of properties and an open set of objects, where each object possesses a subset of these properties, so that each object can be characterized by this subset. Genetic operators such as mutation and crossover can be applied in generating new and useful concepts starting from an existing object. To illustrate, consider that there are objects b, c and d and an ordered set of properties p, q, r, s, t, and u such that the objects b, c and d have the properties

b: $(q\ \ s)$
c: $(q\ \ r\ \ t)$
d: $(q\ \ s\ \ t)$.

The same objects can now be redefined in an equivalent way as

b: [0,1,0,1,0,0]
c: [0,1,1,0,1,0]
d: [0.1.0.1.1.0],

where the square brackets represent the ordered set of domain properties $[p,q,r,s,t,u]$ with the 1's indicating that the object has the property in the same position of the ordered set.

The mutation operator can be applied to b to obtain [0,1,1,1,0,0] by mutating the third position in its concept description. One can then carry out observations to see if there exists objects that match the new concept description. If there are such objects, then this may be a useful concept. Note that [0,1,1,1,0,0] is also a specialization of [0,1,0,1,0,0]. Mutations can also yield generalizations of concepts simply by turning a 1 into a 0 in a concept description.

Similarly, the crossover operator generates new concepts by combining some of the properties of two objects. So, for example, a crossover between b and c can produce d, which inherits only the

[2]Detailed descriptions of genetic algorithms can be found in Holland (1975)

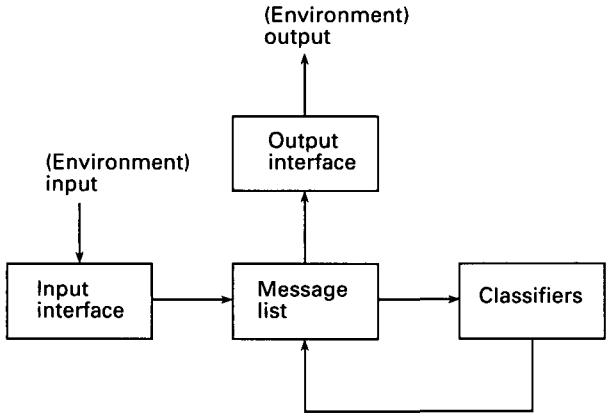


Figure 13 The elements and control of a genetic algorithm-based classifier system described by Holland (1986)

properties *q*, *s* and *t* from the former. The new concepts can be used in classifying such objects into a hierarchy based on the generality of their concept descriptions and the kind of properties they possess. De Jong (1988) explains how genetic algorithms can be utilized in learning systems.

One of the most difficult problems in rule-based systems development is the automatic generation of plausible new rules. Holland (1986) integrates rule-based systems with genetic algorithms to develop his theory of adaptive systems called "classifier systems".[3] According to his definition, a classifier system consists of a set of "classifiers", a message list, an input interface and an output interface (Fig. 13). In these systems knowledge is represented as messages in message lists.

The input interface detects features from an environment and transforms them into messages. In this way the current state of the environment is received in the form of messages. The messages are kept in a message list. The rules become active when a set of messages in the message list matches their condition parts. Their action parts produce new messages which are added to the message list.

Holland's classifiers differ from the rules of production systems in several ways: (1) classifiers can be active in parallel; (2) they can handle conflicts; (3) they use a message list; (4) they use and generate messages in fixed length based on a fixed alphabet; (5) they can easily be coupled with other classifiers; (6) their internal control and external communication have the same structure (the messages); (7) they do not need an interpreter. Holland (1986) proposes genetic algorithms as a solution to the problem of "brittleness" in learning systems.

The basic execution cycle of genetic algorithms in the context of classifier systems is described as follows: (1) select pairs from the set of classifiers (population) according to strength; (2) apply genetic operators to the pairs creating new classifiers (offspring); (3) replace the weakest classifiers with the offspring (see Fig. 14). Genetic operators produce new rules which are tested and evaluated against the environment. Holland (1986) argues that the integration of genetic algorithms with rule-based methods can provide the basis to develop systems that can be used in generating such new rules. He proposes as future work to study the possibilities of inducing a classifier system to construct models of its environment for purposes of planning and look-ahead, without disturbing the other activities of the system. In fact, genetic algorithms have been implemented in scheduling (Whitley et al., 1989), and in inducing control rules (Odetayo & McGregor, 1989). There are also studies on integrating genetic algorithms with neural networks (see Harp et al., 1989; Davis, 1989).

Despite their resemblances to rule-based systems, the classifier systems described by Holland fall into the category of symbol level systems in the present classification because of their low level

[3] Holland's classifier systems should not be confused with the classifiers described above. He uses the term "classifier" for a particular type of condition/action rules.

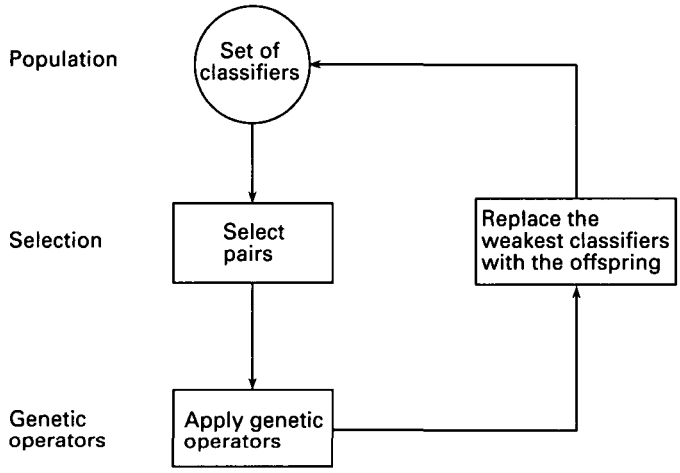


Figure 14 Main elements and the basic execution cycle of genetic algorithms

representation. Holland (1986) also implicitly states that classifiers, and in general symbol level systems, do not explicitly assign abstract symbols to problem states.

3.3 Device level learning

This subsection describes neural networks (Rumelhart & McClelland, 1986; Hopfield & Tank, 1986; Barrow, 1989) as device level learning systems. The main difference between the symbol level and device level learning systems is in the representation of knowledge. In the former, it is represented with labelled symbols, which allow symbol level functions or algorithms to be applied. In the latter, knowledge is represented directly and affects the processing without undergoing any interpretative process.

3.3.1 Learning in neural networks

Neural nets are composed of a large number of interconnected units divided into input, output and hidden nodes, as shown in Fig. 15. A single processing unit merely sums up the weighted activation on its input lines, transforms this sum according to a simple mathematical formula, and passes the resulting function to its output lines. Therefore, in general terms, information processing in neural networks consists of the units transforming their input into some output, which is then modulated by the weights of connections as input to other units. Learning in these systems is defined in terms of the total adjustments of the weights continuously so that the network's output tends toward the desired output without involving changes to the structure of the network.

Neural networks are trained by specifying constraints and providing training examples. When learning is complete (or at an adequate level), knowledge is said to be represented by the totality of the connections in the form of weights of connection between processing units in the network. In other words, knowledge is represented by the entire network. Neural networks can also be trained by genetic algorithms instead of random search (e.g., see Harp et al., 1989; Caudell & Dolan, 1989). Neural networks have been used in optimization (Naft, 1989) and expert system applications (e.g., see Kottai & Bahill, 1989).

From a functional point of view, there is little difference between classifiers and neural networks. Perhaps the only difference between the two types of systems is that neural nets use a simple continuous evaluation function, while the classifiers use a discrete function. Most of the learning tasks which can be carried out by classifier systems can equally be carried out by neural networks. On the other hand, neural networks support parallelism in a direct way, while classifiers need to be adapted to parallel processing hardware and software.

As classifiers, neural nets are also low level systems and lack representational power, and

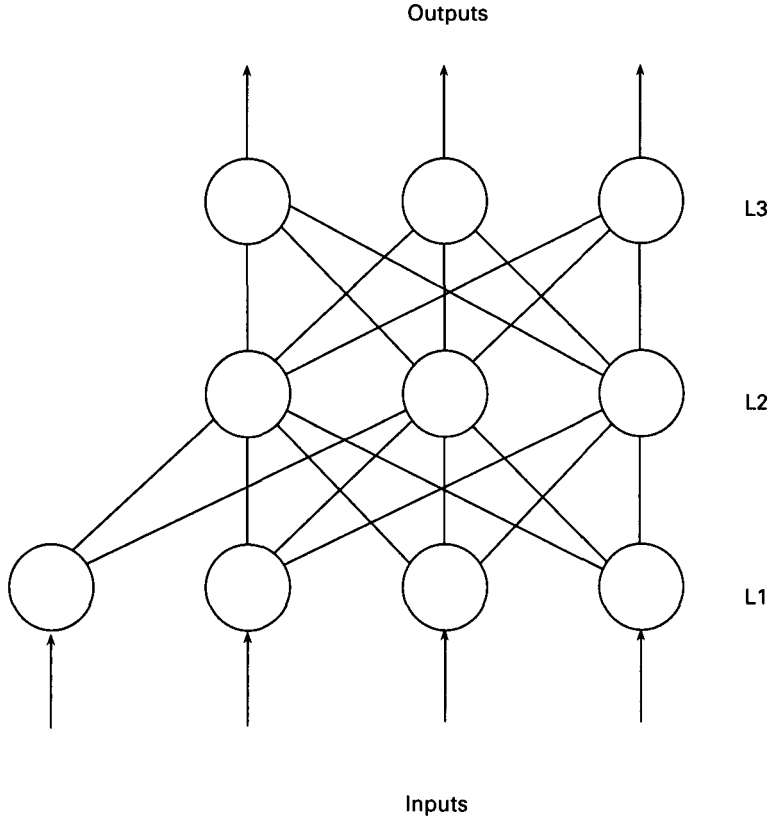


Figure 15 A three layered neural net. The input pattern is received by the bottom layer. The top layer contains output units whose output constitutes the response of the network to the input. The layer between performs the computations by means of a simple continuous function

therefore need elaborate input and output interfaces for higher level representations (such as representing complex propositions and frame-like concepts) and learning tasks (such as planning, symbolic, mathematical and theoretical reasoning). Recently, the theoretical work on combining the low level efficiency of neural networks and the representational power of knowledge level systems has gained momentum, but there is still a long way to go in this direction (see Hinton, 1990).

Another way of eliminating the gaps between the knowledge level systems and symbol and device level systems is their integration for efficient implementations. Device level and symbol level systems can constitute the elements of more abstract knowledge level systems. Symbol level systems such as genetic algorithms and classifier systems can define and control device level systems. Accordingly, symbol level systems can be designed such that they make effective use of device level systems. In a similar way, knowledge level systems can control symbol level systems.

4 Relationships between representation and learning

In existing knowledge systems the following general forms of knowledge representation or their combinations are used: rule-based, frame, predicate logic, semantic networks, classifiers and neural networks. Each representation has its advantages and disadvantages regarding the applicability of various methods of learning. Table 1 provides a comparison of several methods of representation in terms of learning and the levels of representation.

In rule-based systems (see Hayes-Roth, 1985; Frost, 1986), knowledge is represented as simple and compound propositions and condition-action rules. These systems can use both forward and backward chaining. They are essentially knowledge level systems, and allow the implementation of deductive and inductive learning methods such as specialization, generalization and abstraction.

Table 1 Knowledge representation methods and their comparison in terms of representation and learning

Methods	*Descriptive knowledge represented by*	*Prescriptive knowledge represented by*	*Level of representation and learning*	*Types of reasoning and learning facilitated*
Rule-based representation	facts, inference rules	production rules	knowledge level	deductive methods (specializations), inductive methods (generalizations), abstraction
Frame representation	frames, slots, schemata	procedures (demons, constrains, watchdogs)	knowledge level	Deductive methods, inheritance and taxonomic reasoning, non-monotonic reasoning, reasoning about events and processes, learning by analogy, theory revision
Predicate logic representation	simple and complex predicate statements, schemata	condition-action rules	knowledge level	deductive methods (EBL, conceptual clustering), inductive methods (SBL, other generalizations), abstraction, abduction, non-monotonic reasoning, theory revision
Semantic networks	arcs, nodes	—	symbol level	inheritance-based and taxonomic reasoning, specializations, generalizations
Classifiers and genetic algorithms	vectors, matrices, message lists	—	symbol level	inductive learning, conflict resolution, classification, learning control rules
Neural networks	neural nets	—	device level	inductive learning, conflict resolution, classification

By themselves, rule-based systems cannot represent structured knowledge such as can be represented by frames. Unlike predicate logic-based systems, they rely on simple inference mechanisms. Many rule-based systems combine frame or logic based features. The DENDRAL (Buchanan & Feigenbaum, 1978), Meta-DENDRAL (Buchanan & Feigenbaum, 1978), MOLGEN (Friedland, 1979), and BACON (Langley, 1978) systems can be considered as rule-based systems. Some large rule-based systems have the blackboard control architecture (e.g., see Englemore & Morgan, 1988; Brugge & Buchanan, 1989). In these systems, rules are organized into "knowledge sources", which send their output to a "blackboard". A scheduler controls the activities of the knowledge sources in accordance with the problem states indicated by the blackboard.

Frame representation (e.g., Fikes & Kehler, 1985; Frost, 1986), is based on the theory of frames developed by Minsky (1977). In these systems knowledge is represented in data structures called "frames" which can incorporate sets of attribute descriptions called "slots". A frame system provides constructs for organizing objects and classes in taxonomic structures. In this way, frames provide structured representations of objects or classes of objects, so that a frame can contain prototype descriptions of members of a class, as well as descriptions of the class as a whole, and these prototypes can be used in creating default descriptions of the objects.

Frame systems are particularly powerful in inheritance-based inference, because the taxonomic relationships among frames enable descriptive information to be shared among multiple frames by inheritance. Frames also facilitate reasoning about events and processes (see Forbus, 1984), and consequently the systematic implementation of theory revision methods (see Karp, 1990; Rajamo-

ney, 1990). As they allow the structured representation of descriptive knowledge, they also facilitate analogical reasoning.

Frame systems provide no direct methods for describing how the knowledge stored in frames is to be used. Association of domain-dependent behaviour with frames are made by attaching to them procedures written in the host programming language. There are basically three types of procedures which apply to the data in a frame: constraints, demons and watchdogs.

Constraints are activated when a slot is to be updated, to make sure that erroneous or irrelevant information is not placed into the slot. Demons are fired after an update of information in a slot takes place, and they execute a series of commands. Watchdogs are activated whenever the value in a slot they are attached to is accessed rather than updated. The effects of the constraints, demons and watchdogs depend on the application. Each procedure is attached to a specific slot related with the data in that slot. These procedures can be inherited by other frames. Lenat's (1979) AM system makes extensive use of this facility.

Frame systems generally incorporate rule-based or logic-based methods to provide the necessary, effective inference mechanisms. Lenat's CYC system (see Lenat & Guha, 1990) integrates frames with predicate logic for the efficient implementation and control of different inference mechanisms.

Frame systems can be regarded as knowledge level systems as far as the descriptive knowledge represented in frames are concerned. The procedures attached to slots can be regarded as knowledge level or symbol level, depending on whether they are condition-action rules or procedures.

Frame systems are used in successful knowledge engineering tools like KL-ONE, KRL, S1 and KEE. Among the systems modelling scientific discovery, the AM system (Lenat, 1979), EURISKO (Lenat, 1983) and PI (Thagard, 1988) can be considered basically as frame-based systems, though they incorporate rule-based features.

In logic-based systems, knowledge is represented as simple and complex predicate statements. Such systems are based on the first-order predicate calculus. This calculus provides expressive power and a well-developed formalism. Predicate logic had been used in representing high-level concepts like infinity, continuity and causality, and also developing various axiom systems in mathematics, physics and biology (see Carnap, 1958, pp. 177–225). Kowalski (1979) explains how predicate logic can be used in problem solving. Logic-based systems rely on resolution theorem proving for their inference mechanisms. In these systems, prescriptive knowledge is represented in the form of condition-action rules of procedures.

The adoption of Prolog as a logic programming language by the Japanese Fifth Generation Computing Systems project as its core language (e.g., see Kowalski, 1984) has contributed to the consolidation of this language in scientific and technical programming. Developments in metalogical programming (e.g., see Stirling, 1984) further increases the importance of Prolog.

Logic-based systems support a variety of deductive learning methods such as explanation-based learning and conceptual clustering, and inductive methods like similarity-based learning and other forms of generalizations. They also support abstraction, abduction, non-monotonic reasoning and theory revision. However, the predicate calculus itself does not have facilities for defining complex constructs such as can be expressed in frames. For this reason, purely logic-based systems are less suitable for analogical reasoning than frame systems. Similarly, taxonomic and inheritance-based reasoning can be implemented more efficiently in frame systems, even though they can also be implemented in logic-based systems by using metalogical constructs.

Descriptive knowledge represented in frames can be translated into predicate logic statements (Frost, 1986, pp. 479–481; Brewka, 1987). However, in these translations the external structure of the knowledge represented by a frame is usually lost. Frost (1986) describes how inheritance and taxonomic relationships, slot constraints and matching can be expressed in predicate logic notation. He also describes how default values can be accommodated by incorporating non-monotonic logic.

Semantic networks (e.g., see Charniak & McDermott, 1985; Sowa, 1987), also called associat-

ive networks, are another major form of knowledge representation. In such systems, knowledge is represented in the form of nodes and arcs, where nodes represent objects, classes and events, while arcs represent relationships that hold between the concept nodes. There is a strong structural parallelism between the semantic network and predicate logic representation. Knowledge that can be expressed by predicate statements can be represented in semantic networks, with arcs representing the predicates and nodes representing terms. Semantic networks provide a scheme of pointers and back pointers that facilitate accessing information easily.

In semantic networks, descriptive knowledge can be organized into concept hierarchies, which can be used to transfer inheritance properties and relations. Examples of such hierarchies are "is-a" and "instance-of" hierarchies: "instance-of" designates the membership of an individual to some class, while "is-a" says that one class is a subclass of another. However, in semantic networks links between the concepts do not allow for exceptions as in frame systems.

Classifiers (Nilsson, 1965; Hunt, 1975) and classifier systems based on genetic algorithms (Holland, 1986) are inductive learning systems. In these systems, knowledge is represented as vectors or matrices of values to a set of parameters or a set of message lists. These systems support, besides inductive learning, conflict resolution, classification and learning control rules.

Neural nets (Rumelhart & McClelland, 1986; Hopfield & Tank, 1986; Barrow 1989) are composed of a large number of interconnected processing units. In these systems, knowledge is represented by the totality of the connections. Neural networks support inductive learning, classification and conflict resolution.

5 Knowledge systems and the limits of machine learning

As more and more complex tasks are set for artificial intelligence systems, a larger amount of knowledge needs to be represented in them. The scope of knowledge must be widened to avoid a common problem within the current systems referred to as falling off the "knowledge cliff" (see Michalski, 1986) or "brittleness" (Holland, 1986). A system performs well within the scope of knowledge provided to it, but any slight move outside its scope causes the performance to deteriorate rapidly.

Amarel (see Michalski et al., 1986, p. 32) stresses the importance of developing large theory formation systems like Meta-DENDRAL in the fields of biology and physics, for the development of such systems provides empirical insight into the problems of learning and discovery. While recognizing the importance of isolated research on learning, he emphasizes the need to have systems performing various learning activities in a specific domain in an integrated way. Lenat (see Michalski et al., 1986) also emphasizes the necessity of integrated learning systems, arguing that such systems are necessary in building more realistic models of human learning.

On the other hand, as machine learning research discovers or invents new methods and clarifies the existing ones, these methods are integrated in large systems. So, one can argue that without clear methodologies or formalisms it would be difficult to integrate these methods in an efficient way. This means that the research in finding new methods and clarifying the existing ones, and the research into integrated uses of these methods, must go hand in hand,.

The development and the integrated use of machine learning methods are also important in knowledge acquisition in building large knowledge-based systems. General knowledge systems such as CYC (Lenat et al., 1986; Lenat & Guha, 1990) should incorporate different methods of learning in order to cope with the problems of acquiring commonsense knowledge. Such systems should be able to learn the missing facts with direct interaction rather than editing their knowledge base on each case. This means that at some point, the development of these systems is closely dependent on the development and the clarification of machine learning methods.

Simon (1983a), pointing out that human learning is a slow process, questions the idea of making computers learn in the same tedious way. It can be argued that, among other things, the slow pace of human learning precesses may be due to extensive knowledge organization during and after knowledge acquisition. If this is true, we may not be able to improve the speed of learning in

machines with large knowledge bases. Simon also draws attention to the fact that, unlike computers, there is no copy process in human learning. Knowledge stored in one human brain cannot be transferred into another's brain directly. Again, it can be argued that this may never be possible if extensive knowledge organization is taking place during human learning, for people may have different ways of organizing knowledge. However, the knowledge organization of two persons expert in the same field may be closer together. This could explain why two experts can learn faster from one another than a person who is not an expert in the same field.

In an interesting paper, Lenat and Feigenbaum (1987) explore the relationships between a system's knowledge and its learning performance. They introduce a principle which is summarized by the dictum in the context of learning as "knowledge is power". They formulate a principle which they call "the knowledge principle" as follows: A system exhibits intelligent understanding and action at a high level of competence primarily because of the specific knowledge that it can bring to bear: the concepts, facts, representations, methods, models, metaphors and heuristics about its domain of endeavour. Lenat and Feigenbaum (1987) highlight the importance of knowledge by pointing out that many searches are costly, but preserving knowledge for future use is not.

Lenat and Feigenbaum (1987) also introduce a principle to emphasize clarity in knowledge, "the explicit knowledge principle", which states that much of the knowledge in an intelligent system needs to be represented explicitly. However, it can be argued that the effectiveness of this principle depends on the organization of knowledge in a system. Some leading researchers point to the scarcity of research on knowledge acquisition, but this should not be considered surprising, because this problem cannot be tackled effectively before more serious research on knowledge organization is carried out. (For more detailed discussion on knowledge organization, see Kocabas, 1989).

Simon (1983a) proposes the priorities of research on learning as:

(1) research aimed at the simulation and understanding of human learning;
(2) research aimed at understanding why human learning is so slow and inefficient, and correspondingly, at examining the possibility that machine learning schemes can be devised that will avoid some of the tediousness of learning;
(3) research on natural language interfaces for knowledge-based systems;
(4) research on programming from incomplete instructions (automatic programming);
(5) research on discovery programs.

He stresses the necessity of choosing clear targets for useful research in machine learning.

Simon (1983a), like Lenat and Feigenbaum (1987), states that for large knowledge systems, learning will turn out to be more efficient than programming after some stage, however inefficient such learning may be. Gaining a deeper understanding of human learning will continue to provide important clues about what to imitate and what to avoid in machine learning programs. If he is correct about the role of human learning, then it follows that among the most significant kinds of learning research to carry out in artificial intelligence are those that are oriented toward understanding human learning.

Lenat and Feigenbaum (1987) propose a principle to guide empirical research in artificial intelligence, which they call "the empirical inquiry hypothesis". According to this principle, the most profitable way to investigate artificial intelligence is to embody our hypotheses in programs and gather data by running the programs. They propose a three stage research programme for the coming three decades of artificial intelligence research as follows:

(1) Slowly hand-code a large, broad knowledge base.
(2) When enough knowledge is present, it will be faster to acquire more through reading assimilating databases, etc.
(3) To go beyond the frontier of human knowledge,the system will have to rely on learning by discovery.

Not surprisingly, Lenat and Feigenbaum (1987) emphasize that the artificial intelligence researcher

must make a major time commitment to the domain(s) in which his/her programs are to be competent.They cite that Stefik and Friedland spent two years learning about molecular biology before developing MOLGEN, and a decade-long time span has been assigned for CYC (Lenat et al., 1986; Lenat & Guha, 1990). In fact, the design strategy of the latter project is entirely based on the three principles stated above.

Lenat and Feigenbaum (1987) estimate that an average human being may have a knowledge base of the equivalent of under a million frame. Lenat's CYC project aims at including 30,000 articles of a one-volume encyclopedia in 30 frames per article corresponding to 900,000 concepts. (The EDR project in Japan (e.g., see Feigenbaum, 1989) aims to produce a network of 750,000 concepts.) Other independent estimates propose about the same number of concepts for human knowledge (see Lenat & Feigenbaum, 1987, p. 1178). This may seem to be less than what we store in our brains, but a million concepts can produce a trillion one-step inferences involving pairs of such concepts. An interesting conclusion of this analysis is that the search performed by human thinking may not have to be deeper than a few steps at once.

In its current first stage, CYC's growth relies on the efficiency of its knowledge representation rather than its integration of learning methods. It acquires new knowledge by hand-coding and editing (see Lenat & Guha, 1990). Only in or after the second stage is it projected to acquire new knowledge by learning and discovery in any significant sense. At that stage, the system's performance will be dependent on its integration of various learning methods in an efficient way. It is clear that CYC's design philosophy does not follow human cognitive development, which heavily relies on language acquisition, initially. However, as yet there is no alternative to the design philosophy of CYC in the same scale (the Japanese EDR project follows the same lines).

6 Summary

Machine learning methods had been classified and reviewed by various researchers from different perspectives. Carbonell & Langley (1987) describe machine learning methods in task-based categories, while Carbonell et al. (1983) provide alternative classifications of machine learning methods according to the underlying strategy, types of knowledge learned, and by the domain of application. This paper presented a review from an entirely different standpoint, namely from the standpoint of the levels of knowledge representation and learning.

To provide a conceptual and historical background, various definitions of learning were first given together with the objectives and a historical outline of machine learning as an emerging field of science. Then, to complete the background, the elements and the strategies of learning were briefly examined. The primary reason for looking at the machine learning methods form the angle of the levels of representation and learning was to examine how different learning methods were defined and operated at each level, and how they could be integrated in a vertical manner.

Machine learning methods and the systems that use them were classified in three groups as knowledge-, symbol- and device-level methods and systems. Knowledge level systems are those whose inputs, operations and outputs are directly expressible in a high level representation without specific reference to a particular symbolism. Symbol level methods on the other hand, can only be defined in a certain symbolism, while device level methods can only be described with reference to a certain device.

Logical and extralogical methods of inference such as deduction, analogy, abstraction and abduction are more easily applicable to knowledge level systems. Low level inductive methods such as classification and conflict resolution are easier to implement in symbol level and device level systems. Therefore, a vertical integration of learning at different levels can be very useful in the design of complex learning systems.

Knowledge level systems can accommodate symbol level methods such as classification and conflict resolution by directly providing inputs to, and by receiving the outputs from symbol level systems for further processing. Similarly, symbol level methods can be used for controlling device level systems. Such applications have been described in the recent research in artificial intelligence.

The paper also examined the relationships between representation and learning. Each knowledge representation method has its advantages and disadvantages. Complex knowledge systems must be capable of using different representations in combination as necessary. Low level systems such as classifiers and neural networks require complex input and output interfaces for representing and processing knowledge at higher levels. As the level of representation decreases, the complexity of the input and output units increases. In low level systems the problem states are expressed in lists, vectors, matrices or in the strength of connections. In such systems all problems are solved at one level and outputs are translated by the output unit. These systems have found applications in areas where problems are reducible to classification and recognition. Classifiers use discrete functions in calculating the weights of their evaluation matrix. This makes them easier to combine with genetic algorithms.

The recent knowledge intensive approach to learning highlights the importance of initial knowledge in learning and intelligent behaviour. This approach inevitably brings to the top of the agenda the question of how to organize and represent a large amount of knowledge in an efficient way. Efficiency in learning depends on the representation used. The choice and the design of representation in turn, is effected by the strategies and methods of organizing knowledge. Therefore, knowledge organization, representation and learning are closely linked issues in designing complex systems.

A complex knowledge system should be capable of solving problems at different levels efficiently. Communication and control between these levels should be direct, i.e., without using elaborate input and output interfaces. This can be achieved by integrating a series of representations each efficient at a different level. A methodology for organizing complex systems in layers needs to be developed, so that knowledge at different levels can be represented and processed effectively. Classifiers and neural networks can be expected to find wider application only after such a methodology is developed.

Acknowledgements

The author wishes to express his gratitude to Mark Sandler for his comments on an earlier draft, and to John Fox for his detailed comments and suggestions to improve the structure of the paper. Thanks are also due to the anonymous reviewers of the *Knowledge Engineering Review* for their comments.

References

References are divided into groups according to subjects and are annotated.

History of machine learning, learning theory, and methods of learning

Barrow, HG, 1989. "AI, neural networks and early vision". *AISB Newsletter* **89** 6–25.
This is the text of an invited talk at the AISB-89 conference. It aprovides a good summary of the history of the ideas and methods that led to the development of neural networks.

Carbonell, JG, 1982. "Experiential learning in analogical problem solving". *AAAI* 168–173.
This paper describes how analogical reasoning can be used to generate exemplary solutions to related problems, and more general plans can be induced from these solutions.

Carbonell, JG and Langley, P, 1987. "Machine learning". In: SC Shapiro and D Eckroth (eds.), *Encyclopedia of Artificial Intelligence.* Wiley Interscience.
The authors provide a survey of the history of machine learning, and describe the methods of machine learning task based categories.

Carbonell, JG, Michalski, RS and Mitchell, T, 1983. "An overview of machine learning". In: RS Michalski, JG Carbonell and T Mitchell (eds.), *Machine Learning: An artificial intelligence approach.* Morgan Kaufmann.
The authors explain the objectives of machine learning, describe its evolution and various learning methods according to the underlying strategy.

Caudell, TP and Dolan, CP, 1989. "Parametric connectivity: training of constrained networks using genetic algorithms". In: *Proceedings of the Third International Conference on Genetic Algorithms,* pp 370–374. San Mateo, CA: Morgan Kaufmann.
The paper describes how genetic algorithms are used in training neural networks.

Cohen, PR and Feigenbaum, EA (eds.), 1982. *The Handbook of Artificial Intelligence, Vol. III.* Pitman.
This is the first comprehensive handbook of artificial intelligence, covering a wide range of topics. However, I do not know if it has a new edition.

Darden, L, 1987. "Viewing the history of science as compiled hindsight". *The AI Magazine* **8** (2) 33–42.
The author views the history of science as an important source for developing and testing the computational models of scientific discovery. She also provides informal definitions of logical and extralogical methods of inference such as deduction, induction, abstraction, abduction and analogy.

Davis, L, 1989. "Mapping neural networks into classifier systems". In: *Proceedings of the Third International Conference on Genetic Algorithms,* pp 375–378. San Mateo, CA: Morgan Kaufmann.
The paper describes the functional similarities between classifier systems and neural networks, and argues that any neural network can be transformed into a classifier system that is functionally isomorphic to the former.

De Jong, K, 1988. "Learning with genetic algorithsm: an overview". *Machine Learning* **3** 121–138.
This paper describes how genetic algorithms can be applied to machine learning problems.

Dietterich, T, 1986. "Learning at the knowledge level". *Machine Learning* **1** 287–316.
The author introduces a theory of knowledge and symbol level learning, and analyzes the methods of some of the learning systems. The author examines the describability of the behaviour of these systems at different levels.

Dietterich, T and Michalski, RS, 1983. "A comparative review of selected methods for learning from examples". In: RS Michalski, JG Carbonell, and TM Mitchell (eds.), *Machine Learning: An artificial intelligence approach,* Morgan Kaufmann.
The authors examine various learning systems from the standpoint of the adequacy of their methods of representation, their rules of generalization, computational efficiency, and flexibility and extensibility.

Falkenheiner, B, 1987. "Scientific theory formation through analogical inference". In: *Proceedings of the Fourth International Workshop on Machine Learning,* pp 218–229. Los Altos, CA: Morgan Kaufmann.
This paper describes how analogy may be used to discover and refine qualitative models of the physical world.

Fatmi, HA and Young, RW, 1970. *Nature* **228** (93).
The authors define "intelligence" in computational terms, and discuss the implications of their definition.

Feigenbaum, EA, 1963. "The simulation of verbal behaviour". In: EA Feigenbaum and J Feldman (eds.), *Computers and Thought,* pp 297–309. McGraw-Hill.

Greiner, R, 1988. "Learning by understanding analogies". *Artificial Intelligence* **35** 81–125.
This paper defines analogical inference, and describes how it can be used in problem solving.

Harp, SA, Samad, T and Guha, A, 1989. "Towards the genetic synthesis of neural networks". In: *Proceedings of the Third International Conference on Genetic Algorithms,* pp 360–369. San Mateo, CA: Morgan Kaufmann.
This paper describes how genetic algorithms can be used in training neural networks. The authors evaluate the efficiency of their methods.

Hinton, GE, 1990. "Preface to the special issue on connectionist symbol processing". *Artificial Intelligence* **46** 1–4.
The author draws attention to the representational gulf between neural networks and higher level learning systems, and introduces the reader to the latest attempts to bridge this gulf.

Holland, JH, 1975. *Adaptation in Natural and Artificial Systems.* University of Michigan Press.
In this important book the author develops his theory of genetic algorithms.

Holland, JH, 1986. "Escaping brittleness: the possibilities of general-purpose learning algorithms applied to parallel rule-based systems". In: RS Michalski, JG Carbonell and TM Mitchell (eds.), *Machine Learning: An artificial intelligence approach.* Morgan Kaufmann.
This paper describes the basic features of genetic algorithms, and explains how they can be combined with rule-based methods to provide general purpose inductive learning systems (or classifier systems) with an example.

Hopfield, JJ and Tank, DW, 1986. "Computing with neural circuits". *Science* **233** 625–633.

Hunt, EB and Hovland, CI, 1963. "Programming a model of human concept formation". In: EA Feigenbaum and J Feldman (eds.), *Computers and Thought,* pp 310–325. McGraw-Hill.

Hunt, EB, 1975. *Artificial Intelligence.* Academic Press.
In this book the author gives a formal description of linear and nonlinear classifiers.

McDermott, J, 1979. "Learning to use analogies". In: *Proceedings of the Sixth International Conference on Artificial Intelligence,* pp 568–576.

This paper is one of the early works on analogy. It examines the constituents of analogical reasoning and describes a rule-based system capable of analogical reasoning.

Michalski, RS, 1983. "A theory and methodology of inductive learning". In: RS Michalski, JG Carbonell and TM Mitchell (eds.), *Machine Learning: An artificial intelligence approach*. Morgan Kaufmann.

In this work, the author views inductive learning as a heuristic search through a space of symbolic descriptions. The descriptions are generated by the application of inductive and deductive inference methods. The author classifies inductive learning into several different types and provides analytical descriptions of each type.

Michalski, RS, 1986. "Understanding the nature of learning: issues and research directions". In: RS Michalski, JG Carbonell and TM Mitchell (eds.), *Machine Learning*. Morgan Kaufmann.

In this paper, the author discusses various definitions of learning and formulates the goals and directions in machine learning research. He also examines the current research paradigms in machine learning, and reviews learning strategies.

Michalski, RS and Stepp, RE, 1983. "Learning from observation: conceptual clustering". In: RS Michalski, JG Carbonell and TM Mitchell (eds.), *Machine Learning*. Morgan Kaufmann.

The authors introduce a detailed description of one of the major methods of learning from observation called "conceptual clustering".

Michalski, RS, Amarel, S, Lenat, DB, Michie, D and Winston, PH, 1986. "Machine learning: challenges of the Eighties". In: RS Michalski, JG Carbonell and TM Mitchell (eds.), *Machine Learning*. Morgan Kaufmann.

The authors discuss the important tasks for machine learning research and the role of machine learning in artificial intelligence. They also discuss the strategies that should be followed in machine learning research in the near future.

Mitchell, T, 1983. "Learning and problem solving". In: *Proceedings of the Eighth International Joint Conference on Artificial Intelligence,* pp 1139–1151. Karlsruhe, Germany: Morgan Kaufmann.

Mitchell, T, Keller, RM and Kedar-Cabelli, ST, 1986. "Explanation based generalization: A unifying view". *Machine Learning* **1** (1) 47–80.

The authors describe one of the most productive methods of machine learning: explanation-based generalization, and explain how the method has been used in various learning tasks.

Naft, J, 1989. "A modified Hopfield net approach to multi-objective design optimization for printed circuit board component placement". *Int. J. Neural Networks Research and Applications* **1** 78–85.

This paper describes an application of neural networks to optimization problems.

Newell, A, 1982. "The knowledge level". *Artificial Intelligence* **18** (1) 87–127.

This is one of the influential papers in artificial intelligence. The author investigates the representational issues of learning and problem solving at several different levels.

Nilsson, NJ, 1965. *Learning Machines: Foundations of trainable pattern-classifying systems*. McGraw-Hill.

In this book, the author provides a symbolic description of classifiers.

Odetayo, MO and McGregor, DR, 1989. "Genetic algorithms for inducing control rules for a dynamic system". *Proceedings of the Third International Conference on Genetic Algorithms,* pp 177–182. San Mateo, CA: Morgan Kaufmann.

This paper describes how genetic algorithms are used for automatically inducing control rules for a dynamic physical system.

Porter, BW and Kibler, DF, 1986. "Experimental goal regression: A method for learning problem-solving heuristics". *Machine Learning* **1** (3) 249–286.

In this paper, the authors describe episodic learning, an enhanced version of explanation-based generalization. They explain with examples how the method has been implemented in a program.

Quinlan, JR, 1983. "Learning efficient classification procedures and their application to chess end games". In: RS Michalski, JG Carbonell and TM Mitchell (eds.), *Machine Learning: An artificial intelligence approach*. Morgan Kaufmann.

Rosenblatt, F, 1958. "The perceptron: A probabilistic model for information storage and organization in the brain". *Psychological Review* **65** 386–407.

This paper in one of the earliest works in connectionist systems.

Rumelhart, DE and McClelland, JL, 1986. *Parallel Distributed Processing: Explorations in the microstructures of cognition*. The MIT Press.

This book constitutes an important contribution to cognitive science.

Simon, HA, 1983a. "Why should machines learn?" In: RS Michalski, JG Carbonell and TM Mitchell (eds.), *Machine Learning*. Morgan Kaufmann.

In this paper the author describes the objectives of machine learning.

Simon, HA, 1983b. "Search and reasoning in problem solving". *Artificial Intelligence* **21** (1–2) 7–30.

In this paper, the author discusses the role of reasoning in problem representation and problem solving.

Simon, HA and Lea, G, 1974. "Problem solving and rule induction: A unified view". In: L, Gregg (ed.), *Knowledge and Cognition.* Erlbaum.
In this paper, the authors introduce a general model for learning, and describe its constituents.
Sowa, J, 1987. "Semantic networks". In: SC Shapiro (ed.), *Encyclopedia of Artificial Intelligence.* Wiley Interscience.
The author provides a review of semantic networks, including a historical survey, early work, and discusses the relationships between logic, frame representation and semantic networks.
Stirling, L, 1984. "Logical levels of problem solving". *J. Logic Programming* **2** 151–163.
This paper identifies three levels of knowledge necessary for intelligent problem solving: The levels of domain knowledge, method and strategies, and planning. The author relates these levels to the distinction between object and meta languages.
Whitley, D, Starkweather, T and Fuquay, D, 1989. "Scheduling problems and traveling salesmen: The genetic edge recombination operator". *Proceedings of the Third International Conference on Genetic Algorithms,* pp 133–140. San Mateo, CA: Morgan Kaufmann.
This paper describes an application of genetic algorithms to one of the classic problems in artificial intelligence.
Winston, PH, (ed.), 1975. *The Psychology of Computer Vision,* pp 157–209. McGraw-Hill.

Learning and discovery systems
Akl, S, 1987. "Checkers-playing programs". In: SC Shapiro (ed.), *Encyclopedia of Artificial Intelligence.* Wiley Interscience.
Various checkers playing programs and their learning methods are described.
Friedland, P, 1979. "Knowledge-based experiment design in molecular genetics". In: *Proceedings of the Sixth International Joint Conference on Artificial Intelligence,* pp 285–287.
The paper describes MOLGEN, a rule-based system that designs experiments in molecular genetics.
Kocabas, S, 1990. "Conflict resolution as discovery in particle physics". *Machine Learning* (in press).
This paper describes an impasse resolution system that models certain discoveries in particle physics.
Kottai, RM and Bahill, T, 1989. "Expert systems made with neural networks". *Int. J. Neural Networks Research and Applications* **1** 211–226.
This paper examines the similarities and differences between expert systems built on neural networks and conventional shells.
Kulkarni, D and Simon, HA, 1988. "The processes of scientific discovery: The strategy of experimentation". *Cognitive Science* **12** 139–175.
This paper describes a theory driven system that models empirical discovery in biochemistry.
Laird, JE, Newell, A and Rosenbloom, PS, 1986. "Chunking in SOAR". *Machine Learning* **1** 11–46.
The paper describes a complex general system SOAR, which incorporates different learning methods.
Laird, JE, Rosenbloon, PS and Newell, A, 1987. "SOAR: An architecture for general intelligence". *Artificial Intelligence* **33** 1–64.
This paper describes SOAR's control structure and its capabilities.
Langley, P, 1978. "BACON1: A general discovery system". In: *Proceedings of the Second National Conference of the Canadian Society for Computational Studies.*
The paper describes the earliest version of BACON, a well known quantitative discovery program.
Langley, P, 1981. "Data-driven discovery of physical laws". *Cognitive Science* **5** 31–54.
In this paper, the author describes the general heuristics that are used in quantitative discovery.
Langley, P, Simon, HA, Bradshaw, GL and Zutkow, JM, 1987. *Scientific Discovery: Computational explorations of the creative processes.* The MIT Press.
This book is an excellent introduction to the current research into the computational modelling of scientific discovery. It describes several discovery systems such as GLAUBER, STAHL and BACON.
Lenat, DB, 1979. "On automated scientific theory formation: a case study using the AM program". In: J Hayes, D Michie and LI Mikulich (eds.), *Machine Intelligence 9,* pp 251–283. Halstead.
The author describes one of the most important discovery programs, AM, which rediscovers many arithmetical concepts starting with the basic concepts of set theory. The paper explains how AM's heuristics work.
Lenat, DB, 1983. "EURISKO: A program that learns new heuristics and domain concepts". *Artificial Intelligence* **21** (1–2) 61–98.
In this paper, the author describes EURISKO, a theory driven discovery system developed as a successor to the AM program. EURISKO has the additional capability of discovering new problem solving heuristics.
Nordhausen, B and Langley, P, 1987. "Towards an integrated discovery system". In: *Proceedings of the Tenth International Joint Conference on Artificial Intelligence,* pp 198–200.
In this interesting paper, the authors describe IDS, a discovery system that integrates qualitative and

quantitative methods. IDS is the first fully implemented discovery system with the capability of sensing and changing its environment.

O'Rorke, P, Morris, S and Schulenburg, D, 1990. "Theory formation by abduction". In: J Shrager and P Langley (eds.), *Computational Models of Scientific Discovery and Theory Formation.* Morgan Kaufmann.
In this paper, the authors view abduction as a general method for theory revision. They describe a discovery program, AbE, and its application to the paradigm shift in the eighteenth century chemistry.

Rajamoney, SA, 1990. "A computational approach to theory revision". In: J Shrager and P Langley (eds.), *Computational Models of Scientific Discovery and Theory Formation.* Morgan Kaufmann.
This paper describes a qualitative theory revision program with the capability of designing experiments to resolve contradictions in its explanations.

Rose, D and Langley, P, 1986. "Chemical discovery as belief revision". *Machine Learning* **1** 423–452.
The paper describes one of the early examples of theory revision systems which has been applied to eighteenth century chemistry.

Thagard, P, 1988. *Computational Philosophy of Science.* The MIT Press.
This is an important book that relates machine learning and discovery to philosophy of science. It also describes a program that models theoretical discovery.

Thagard, P and Holyoak, K, 1985. "Discovering the wave theory of sound: Inductive inference in the context of problem solving". In: *Proceedings of the Ninth International Joint Conference on Artificial Intelligence,* pp 610–612.
In this paper the authors describe a program that models the discovery of theory of sound.

Zytkow, JM and Simon, HA, 1986. "A theory of historical discovery: The construction of componential models". *Machine Learning* **1** 107–137.
The authors describe a data driven system that simulates the discoveries of the componential models of substances in the seventeenth century chemistry.

Knowledge representation

Brewka, G, 1987. "The logic of inheritance in frame systems". In: *Proceedings of the Tenth International Joint Conference on Artificial Intelligence,* pp 438–488.
This paper presents a formal description of inheritance in frame systems, together with a formalism for transforming knowledge represented in a frame into predicate logic statements.

Carnap, R. 1958. *Introduction to Symbolic Logic and its Applications.* Dover Publications.
This book is interesting in its being one of the earliest and best examples of the linguistic philosophical approach to logic.

Charniak, E and McDermott, J, 1985. *Introduction to Artificial Intelligence.* Addison Wesley.
This book is one of the well known textbooks in artificial intelligence.

Engelmore, R and Morgan, T (eds.), 1988. *Blackboard Sytems.* Addison Wesley.
This book provides a detailed description of the methods used blackboard systems, and describes a number of applications.

Fikes, R and Kehler, T, 1985. "The role of frame-based representation in reasoning". *Commun. ACM* **28** (9) 904–920.
The authors describe the elements and the methods of the frame-based representation with examples. The paper does not include the new techniques used in frame systems.

Forbus, KD, 1984. "Qualitative process theory". *Artificial Intelligence* **24** 85–168.
The author describes the principles of qualitative representation of physical processes.

Frost, R, 1986. *Introduction to Knowledge Base Systems.* Collins Professional and Technical Books.
This is one of the earliest and most detailed books on knowledge-based systems.

Hayes-Ruth, F, 1985. "Rule-based systems". *Commun ACM* **26** (9) 921–932.
This paper describes rule-based representation and its methods with examples.

Kocabas, S, 1989. *Functional Categorization of Knowledge: Applications in modeling scientific research and discovery.* PhD thesis, Department of Electronic and Electrical Engineering, King's College London.
This work introduces a methodology for organizing knowledge into functional categories, so that different types of inference can be implemented more efficiently.

Kowalski, R, 1979. *Logic for Problem Solving.* North Holland.
The book introduces first order predicate logic as a language for general problem solving.

Kowalski, R, 1984. "Logic programming in the fifth generation". *The Knowledge Engineering Review* **1** (1) 26–38.
The paper describes how Prolog is used in knowledge representation and problem solving.

Minsky, M, 1977. "Frame system theory. thinking: readings in cognitive science". In: PN Johnson-Laird and PC Wason (eds.), *Open University Set Book. Cambridge University Press,* pp 355–376.
This is one of the most influential papers in the development of knowledge-based systems.

Large knowledge-based systems

Brugge, JA and Buchanan, BG, 1989. "Evolution of a knowledge based system for determining structural components of proteins". *Expert Systems* **6** (3) 144–156.
The authors describe a knowledge-based system, ABC, which determines the structural components of proteins. The system is based on the blackboard control architecture.

Buchanan, BG and Feigenbaum, EA, 1978. "Dendral and Meta-Dendral: Their application dimension". *Artificial Intelligence* **11** 5–24.
The paper describes DENDRAL, which constitutes one of the earliest successful implementations of the rule-based methods in scientific research, and Meta-DENDRAL, a system that discovers the heuristics to be used by the former.

Feigenbaum, EA, 1989. "An interview". *Expert Systems* **6** (2) 112–115.
The author discussed the current developments in artificial intelligence, and makes projections into the future of the discipline is a leading researcher.

Lenat, DB, Prakesh, M and Shepherd, M, 1986. "CYC: using common sense knowledge to overcome brittleness and knowledge acquisition bottlenecks". *The AI Magazine* **7** (4) 65–85.
This paper describes some of the design ideas of one of the largest research projects in artificial intelligence. The project is called CYC, and aims at building a general knowledge base including a large amount of commonsense knowledge.

Leant, DB, and Feigenbaum, EA, 1987. "On the threshold of knowledge". In: *Proceedings of the Tenth International Joint Conference on Artificial Intelligence,* pp 1173–1182.
In this paper, the authors discuss the problems of building complex intelligent systems. Even though it contains some controversial views, the paper is an important contribution to artificial intelligence and should not be ignored.

Lenat, DB and Guha RV, 1991. *Building large Knowledge Based Systems: Representation and inference in the CYC Project.* Addison Wesley.
This is a new book in which the authors expand on Lenat et al.'s (1986) paper, and describe the knowledge representation and inference methods of CYC. Should not be missed by anyone who is interested in building large knowledge bases.