A MODEL FOR LEARNING SYSTEMS

Reid G. Smith, Tom M. Mitchell, Richard A. Chestek, and Bruce G. Buchanan

Departments of Computer Science and Electrical Engineering,
Stanford University.
Stanford, California, 9405.

Abstract*

A model for learning systems is presented, and representative AI, pattern recognition, and control systems are discussed in terms of its framework. The model details the functional components felt to be essential for any learning system, independent of the techniques used for its construction, and the specific environment in which it operates. These components are performance element, instance selector, critic, learning element, blackboard, and world model. Consideration of learning system design leads naturally to the concept of a layered system, each layer operating at a different level of abstraction.

Descriptive Terms: adaptation, learning, concept-formation, induction, performance element, instance selector, critic, learning element, blackboard, world model, multi-layered systems.

1 Introduction

Learning systems have been the subject of wide research interest for a number of years. The terms adaptation, learning, concept-formation, induction, self-organization, and self-repair have all been used in the context of this study. Learning system (LS) research has been conducted within many different scientific communities, however, and these terms have come to have a variety of meanings* It is therefore often difficult for members of these communities to recognize that problems which appear unrelated as a result of variations in terminology may in fact be identical. Learning system models as well are often tuned to the requirements of a particular discipline and are not suitable for application in related disciplines. We have therefore synthesized a new LS model which provides a common language for unified characterization of systems constructed from a number of different perspectives. This model encourages examination of the strengths and weaknesses of the individual functional components necessary for any learning system. Because the model enables a designer to isolate these functional components and specify the information which must be available to them", it is particularly useful as a paradigm for new learning systems.

In the context of this paper, a learning system is considered to be any system which uses information obtained during one interaction with its environment to improve its performance during future interactions. This definition is intentionally broad and may include man/machine systems in which humans take on active roles as required functional components•

11 This work was supported by the Advanced Research Projects Agency under contract DAHC 15-73-C-0UJ5. the National Institutes of Health under grant RR 00612-07. the Naval Air Systems Command under contract N0019-76-C-0250, and the National Science Foundation under contract GK-M1972. Reid Smith is supported by the Research and Development Branch of the Department of National Defence of Canada. C. Richard Johnson, Jr. provided very helpful comments on adaptive control systems. We also received many valuable suggestions from members of the Heuristic Programming Project at Stanford*

In the following sections we will summarize two different approaches to the construction of systems that can be said to learn. The first approach centers on the concept of an adaptive system and is primarily associated with research in pattern recognition and control theory: the second is that of artificial intelligence (Al).

2 Adaptive System Approach to Learning

In the control literature, learning is generally assumed to be synonymous with adaptation, and is often viewed as estimation or successive approximat ion of the unknown parameters of a mathematical structure which is chosen by the LS designer to represent- the system under study [6] [10]. Once this has been done, control techniques known to be suitable for the particular chosen structure can be applied. Thus the emphasis has been on parameter learning, and the achievement of stable, reliable performance [25]. Problems are commonly formulated in stochastic terms, and the use of statistical procedures to achieve optimal performance with respect to some performance criterion such as the probability of correct pattern classification, or mean square error, is standard [33].

There are many overlapping and sometimes contradictory definitions of the terms related to adaptive systems. The following set, formulated by Glorioso 111] serves to illustrate the main features. An adaptive system is defined as a system which responds acceptably with respect to some performance criterion in the face of changes in the environment or its own internal structure, A learning system is an adaptive system that responds acceptably within some time interval following a change in its environment, and a self-repairing system is one that responds acceptably within some time interval following a change in its internal structure. Finally, a self-organizing system is an adaptive or learning systeminwhich the initial state is unknown, random, or unimportant.

Other terms often used to describe learning systems in the pattern recognition and control literature are "supervised" and "unsupervised" learning [5l [10J. Supervised learning, or "learning with teacher", assumes the existence of an external entity (usually a human) which presents the system with a set of training instances, evaluates the performance of the system for those instances, and provides the correct responses. Unsupervised learning, or "learning without teacher", assumes that the environment provides all instances, but does not provide the correct responses. Performance is to be evaluated by the system itself. Tsypkin [28] has pointed out that unsupervised learning is somewhat of an illusion in the sense that a teacher/designer defines the standards which determine the quality of operation of the LS at the outset, whether or not he is present during the actual operation of the system.

3 Artificial Intelligence Approach to Learning

Although early AI research was closely tied to pattern recognition and the adaptive systems approach, (see, for example [23] and [29];, the two fields diverged in the 1960's, and are now quite distinct. Whereas the pattern recognition and control research emphasizes adjustment of parameters, AI research emphasizes construction of symbolic structures, based on conceptual relations. For example, Feigenbaum's EPAM program [7] used a discrimination net (i.e., a tree of tests and branches) to store the relations required to recall nonsense syllables in a rote learning experiment (see [91, 126], and [32] for further examples).

In AI, it is commonly believed that a learning system should have sufficient internal structure to develop a "strong theory" of its environment [8] [16]. Much emphasis has therefore been placed on building "knowledge-based" or "expert" systems that not only have the capacity for high performance, but can also explain their performance in symbolic terms [U],

Winston [32] describes various levels of sophistication in learning systems: learning by being programmed, learning by being told, learning from a series of examples, and finally learning by discovery. We see in this categorization a gradual shift in responsibility from the designer/teacher to the learning system/student. At the highest level, the system is able to find its own examples, and carry on autonomously.

14 Effects of the Environment

The environment in which an LS operates may have a profound effect upon its design, and therefore it is of interest to consider a few major environment classes. LS environments can be divided into two major categories: those that ?rovide the correct response for each training ?nstance (supervised learning) and those that do not (unsupervised learning). Supervised learning systems operate within a stimulus-response environment in which the desired LS output is supplied with^ each training instance. Examples include Samuel's "book move" checkers program [21] *[22]*, and grammatical inference programs [12].

Unsupervised LS's operate within an environment of instances for which the correct response is not directly available. The version of Samuel's program which learns by playing checkers against an opponent falls into this category [21]. Learning systems operating within this type of environment must themselves infer the correct response to each training instance by observation of system performance for a series of instances. As a result, assignment of credit or blame for overall performance to individual responses can be a problem for these systems [18].

Environments can be further categorized as "noise-free" or "noisy". Noise-free environments, such as that of Winston's structural description learning program [32] provide instances paired with correct responses in which the data are assumed to be perfectly reliable. Noisy environments, on the other hand, do not provide such perfect information, as is usually the case when real data are involved (pattern recognition and control systems frequently operate within noisy environments [1] [5] [6]).

5 The Proposed LS Model

We are concerned with the functional description of LS's and their interaction with the environments in which they operate. Many of the functional components of an LS are essential to intelligent systems in general, as noted also by Simon and Lea [2*0.

5-1 The Model - Overview

The proposed LS model is shown in Figure 1. The PERFORMANCE ELEMENT is responsible for generating an output In* response to a training instance. The INSTANCE SELECTOR selects suitable training instances from the environment. The CRITIC analyzes the output of the performance element in terms of some standard of performance. The LEARNING ELEMENT makes specific changes to the system in response to the analysis of the critic. Communication among the functional components is shown via a BLACKBOARD to ensure that each functional component has access to all required system information, such as the emerging knowledge base. Finally, the LS operates within the constraints of a WORLD MODEL which contains the general, assumptions and methods that define the domain of activity of the system.

The components of the model are conceptual entities which specify functions that must be performed to effect learning. They simplify the characterization of existing oysters, and will assist designers in the construction of new systems. Although the functional decomposition suggested by *he model is not necessarily reflected in the physical decomposition of many existing systems, we do advocate such a correspondence in future learning system designs.

In the following sections, we present detailed discussions of the LS model components shown in Figure 1. In addition, Appendix I contains detailed characterizations of representative AI, pattern recognition, and control systems in terms of the model. The reader may find it helpful to refer occasionally to this appendix while reading the following sections.

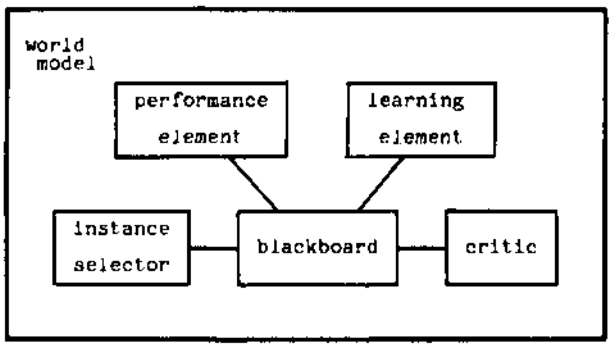

Figure 1. The Components of a Learning System

5.2 Performance Element

The performance element uses the learned information to perform the stated task. It has been included in the LS model because of the intimate relationship between what information is to be learned and how this learned information is to be used.

Performance elements are usually tailored more to the requirements of the task domain than to the architecture of the LS. In general, the performance element can be run in a stand-alone mode without learning, independent of the rest of the LS (e.g., Samuel s checker playing program [21] *[22])*. In any LS, however, the ability to improve performance presupposes a method of communicating learned information to the performance element. Since its architecture must allow learned information to affect its decisions, additional constraints are placed on the performance element within an LS. The performance element should be constructed so that information about its internal machinations is readily available to the other system components. This information can be used to make possible detailed criticism of performance, and intelligent selection of further instances to be examined by the system.

The performance elements of existing systems also vary In the ways in which they may be altered by learning. For example, systems whose operation is determined by a set of production rules [30] [31] have the potential to exhibit richer variations than systems whose operation is keyed only to the adjustment of parameter values [14 [171.

5.3 Insta,nce Selector

The instance selector selects training instances from the environment that are to be used by the LS. It is a functional component, not clearly isolated in earlier adaptive system models.

In reviewing existing LS's we have found that methods for instance selection vary mainly along the dimensions of responsibility and sophistication. The responsibility for instance selection varies between the extremes of completely external ("passive") selection, and completely internal ("active") selection. Instance selection in Samuel's book move checkers program [21] *[22]* is externally controlled, whereas Popplestone s program [20]. which learns the features that characterize a winning position in tic-tac-toe, generates its own training instances. It forms alternate hypotheses, and then generates instances to choose among them (relying upon an external critic to evaluate these instances). In the adaptive systems literature, Tse and Bar-Shalom [27] use a form of active instance selection known as "dual-control". They adjust the input to a system in such a way as to simultaneously control its output and obtain information about its internal structure.

The degree of sophistication used for LS instance selection is also an important consideration. In order to qualify as sophisticated, an instance selector must be sensitive to the current abilities and deficiencies of the performance element and must construct or select instances which are designed to improve performance. Winston [32] has shown the advantages to be accrued through presenting carefully constructed examples and "near-misses" of the concepts to be acquired by an LS. In general. careful instance selection can improve the reliability and efficiency of an LS. We must note, however, that this may not always be permitted by the environment in which the LS operates, as is generally the case for adaptive control systems [0].

54 Critic

The critic analyses the current, abilities of the performance element. It may play three roles; EVALUATOR. DIAGNOSTICIAN, and THERAPIST. The critic always operates as an evaluator In that it embodies a standard by which to assess the behaviour of the performance element. This is the role that has been emphasized in earlier adaptive system models [10] [11] [25].

The critic may also operate as a diagnostician, and localize the reasons for poor performance. This type of behavior is essential for resolution of the credit assignment problem described by Minsky [18]. In Its role as diagnostician, the critic is exemplified by the bug classifier and summarizer in Sussman's HACKER [26].

Finally the critic may operate as a therapist, and make specific recommendations for improvement or suggestions about future instances. In Waterman's poker player [30], the critic as therapist suggests the bet that should have been made by the performance element for a particular training instance.

Not all systems exhibit sufficiently complex behavior to warrant critics that fulfill all three functions. The critic as therapist in particular is not often seen in simple systems.

The dividing line between critic and learning element is difficult to distinguish, and it is certainly possible to view therapy as a function of the learning element, rather than one of the critic. However, in mapping existing LS's into our model^ we have adopted the convention that the critic s recommendations to the learning element are at an abstract level removed from the implementation considerations such as data representation. This clearly separates the two different functions of deciding what kind of change is needed and deciding how to implement that change.

In some LS's the functions of the critic have been left to the human who uses the system. For example. MYCIN/TEIRESIAS [H] uses a human critic, acting as evaluator, diagnostician, and therapist to suggest alterations to its rule base.

5.5 Learning Element

The learning element is an interface between the critic ana the performance element, responsible for translating the abstract recommendations of the critic into specific changes in the rules or parameters used by the performance element.

Representations for learned information exhibit great variety. They include, for example production rules [30], parameterized polynomials T21], executable procedures [26], signature tables *[22],* stored facts [7], and graphs [32]. The method of incorporating new learned information is dependent upon this representation, and even among systems which use similar representations, competing methods are found (contrast, for example, [3] and [30]).

The extent to which the learned information is altered in response to each training instance is an important LS design consideration. In some systems [32]. the learning element incorporates exactly the Information supplied by the critic. Were the same training instance to occur later, the response of the performance element would be exactly as the critic advised for the first occurrence. This type of learning is well suited to environments which provide perfect data and to systems with reliable critics. Under these conditions the LS will converge rapidly to the desired behavior. If such a system were provided with an incorrect classification by the environment or less than reliable advice by the critic, however, it might commit itself to incorrect assumptions from which it is difficult to recover. Systems which make less drastic chances to the learned knowledge on the basis of a single training instance are less vulnerable to imperfect information, but consequently require more training instances to converge to the desired behavior. Many statistical LS's fall into this category [19]. Other systems consider several training instances at a time in order to minimize the effect of occasional noisy instances [3].

5.6 Blackboard

The blackboard of our model is a global data base which also functions as a system communications mechanism. It is similar to the concept introduced in the HEARSAY system [15]. The blackboard holds two types of information: the information usually associated with the "knowledge base" in AI programs, and the temporary Information used by the LS components. The knowledge base often contains the set of rules, parameter values, symbolic structures, and so on, currently being used by the performance element. Such information can be used as an aid to sophisticated instance selection if it is readily available. The temporary, system-oriented information includes, for example, the intermediate decisions made by the performance element in selecting a particular response. Detailed criticism by the critic is dependent upon the availability of this information.

In many existing systems this information is not so clearly separated or defined. The communication links between functional components, especially, are often programmed directly. Because the same information is required by many of the individual functional components of any LS, however, a blackboard is a more appropriate communications mechanism.

5.7 Worses Model

Whereas the blackboard contains information that can be altered by the LS components, the world model contains the fixed conceptual framework within which the system operates. The contents of the world model include definitions of objects and relations in the task domain, the syntax and semantics of the information LO be learned, and the methods to be used by the LS. Among task domain definitions are, for example, the rules of a game and the representation or inputs and outputs for the performance element. This part of the worjd model simply defines the task of the performance element, and the standard of performance (the evaluation function) to be applied by the critic. Domain specific heuristics are also commonly added to the world model of AI systems to guide inferences made by the $_x$LS (e.g., the blocks world heuristics of Winston's program *132]).* Definitions of the syntax and semantics of information to be learned define the mode of communication between the learning and performance elements.

The assumptions and constraints from which the world model is composed are of critical importance in the design and characterization of LS's. Although many of these assumptions are often hidden in the various functional components, the LS designer and user must both be aware of each of them. We believe that, where possible, world model constraints should be made explicit in order to allow for their modification during the design process.

6 Multi-Layer Learning Systems

Although the world model cannot be altered by the LS that uses it. the designer can alter its contents in order to improve LS performance. He often changes parameters and procedures of the basic LS after observing and criticizing its behavior for some carefully chosen training set. These alterations result in a new version of the LS, which is then tested on some training set, and so on. The designer views the whole LS as a system whose performance needs improvement, and he selects instances, criticizes performance, and makes changes accordingly. In other words, the designer's activities can be modeled by a system whose components are just those of Figure $_e$\. This leads us to the concept of layered LS's, each higher layer able to change the world model (vocabulary, assumptions, etc.) of the next lower layer on the basis of criticizing its performance on a chosen set of instances. Thus, adjustments can be made to the world modeJ of some learning system LS1 by another learning system, LS2, which has its own functional components (critic, world model, etc.). In turn, it is conceivable that a third system, LS3, could adjust the world model of LS2, and so on. The designer constitutes the final critic, of course, operating above of the "top-level" LS. Each lower layer constitutes the performance element of the next higher layer, and inter-layer communication is effected through the blackboards of the various layers. The use of a blackboard in the single layer LS model was partly motivated by its attractiveness in the multi-layer context.

This multi-layer architecture can be differentiated from a hierarchical architecture. The latter involves only bottom-to-top propagation of information, whereas the former involves bidirectional information passing; that is, the effects of adjustments made in a layer may propagate both to lower and higher level layers.

One existing LS which may be viewed as a layered system is the version of Samuel 's program *[22]* which learns a polynomial evaluation function for selecting checkers moves (see Appendix 1 for details). The lower layer (LS1) in this system adjusts the coefficients of a given set of game board features in order to improve performance of the move selection program. The second layer system (LS2) adjusts the set of board features used in the evaluation function in order to improve the performance of LS1. Since LS1 is contained in LS2 as the performance element, all the assumptions necessary for its operation also belong to the LS2 world model. In addition, the LS2 world model contains assumptions about the set of allowable game board features and the standard for evaluating LS1 performance.

A single layer LS, then, can never move outside its world model to make radical revisions to its way of viewing the task to achieve a "paradigm shift", as discussed by Kuhn [13]. However, a shift in the conceptual framework of LS1 could be made by a properly programmed LS2 [2]. We believe thai a layered approach such as that described above provides a useful system organization for learning at various levels of abstraction in complex domains. Although there are examples of this kind of layering in the literature [21] [29], no one has carried it as far as our model suggests, and it appears that we are just now reaching the point of understanding single layer learning systems well enough to consider developing more sophisticated systems.

7 Summary

The proposed LS model provides a common language for characterization and comparison of different types of learning systems which operate in a variety of task domains. We believe the model is a useful conceptual guide for LS design, because it isolates the essential functional components, and the information that must be available to these components.

We have alluded to a number of desirable features for future learning system designs. First, the design should be modular and individual modules should correspond to the functional components shown in the model. The knowledge used by the system should be made explicit and collected, as much as efficiency considerations permit, in a world model component. The parts of the LS that are to be adjustable especially must be explicitly exposed. We have emphasized the importance of intelligent criticism, and suggested that active instance selection be further examined. Finally, we have suggested a multi-layer architecture for learning at different levels of abstraction.

Appendix i

Characterization of Existing Systems

In this appendix several existing LS's are characterized using the framework provided by the model described in Section 5. The systems selected are representative of several approaches to machine learning. Because the blackboard contains information in a state of flux, its contents are not specified explicitly for the systems characterized below.

Meta-DENDRAL, Buchanan, et al. [3]

Purpose: Learn to predict data points in the mass spectra of molecules.
Environment: Set of all known molecule/data-point pairs.
Performance Element: Predicts peaks (data points) in mass-spectra of molecules using learned production rules. Employs a model of mass spectrometry for translating between mass-spectral processes (predicted by the rules) and data points in the spectrum.
Instance Selector: Accepts a set of known molecule/spectrum pairs from the user.
Critic: Evaluation - determines the suitability of the set of predictions generated by a rule. Diagnosis - states whether the rule is acceptable, too specific, or too general. Therapy - recommends adding or deleting features to the left-hand sides of rules.
Learning Element: Conducts a heuristic search through the space of plausible rules using a predefined rule generator. At each step in the search the potential rule's performance is reviewed by the critic.
World Model: Representation of molecules as graphs, production rule model of mass spectrometry, vocabulary of rules used to represent learned information; heuristics used by the critic in directing the rule search.

Learning Structural Descriptions from Examples. Winston {32}

Purpose: Learn to identify blocks world structures (such as arches and towers).
Environment: Set of possible line drawing/structure-classification pairs.
Performance Element: Decides class of structures to which the input structure belongs. Uses a model of the structure class supplied by the learning element.
Instance Selector: Accepts training instances supplied individually by the user.
Critic: Evaluation - compares the classification made by the Performance Element against the correct classification as supplied with each training instance. Diagnosis - generates a comparison description pointing out differences between the model and the structure description.
Learning Element: Constructs a model of the class of structures under consideration. Examines the comparison description supplied by the critic, and modifies the model to strengthen or weaken the correspondence between the model and the training instance.
World Model: Representation of scenes as line drawings, method of translating line drawings to graphical descriptions, grammar for representing the learned information, domain-specific heuristics for resolving among possible changes to each structure class model.

Checker Plaver. Samuel [21] *[22]*

Purpose: Learn to play good game of checkers (here we discuss only the version of the program which learns a linear polynomial evaluation function by examination of moves suggested by experts ("book moves").
Environment: Set of all legal game boards.

LS1 (lowest layer):

Purpose: Learn a good set of coefficients for combining board features in a linear polynomial evaluation function.
Performance Element: Uses the learned evaluation function to rank plausible moves for a given board position.
Instance Selector: Reads instances from a list of pre-defined game-board/recommended-move pairs.
Critic: Evaluation - examines the ranking given to the book move by the performance element. Diagnosis - suggests that the book move should be ranked above all other moves.
Learning Element: Adjusts weights of linear polynomial to make move selection correspond to the critic's recommendation.
World Model: Syntax of game board, form and features of linear polynomial evaluation function, method for adjusting evaluation function, and rules of checkers.

LS2:

Purpose: Improve the performance of LS1 by selection of a good set of board features.
Performance Element: LS1.
Instance Selector: The entire set of possible training instances is simply passed to LS1 (via the blackboard).
Critic: Evaluation - analyses the learning ability of LS1 (i.e., the LS2 performance element) with the current set of evaluation function features. Diagnosis - singles out features which are not useful. Therapy - selects new features from a predefined list to replace useless features.
Learning Element: Redefines the current set of features as recommended by the critic.
World Model: The LS1 world model plus the set of featureswhich may be considered, and the performance standard employed by the LS2 critic.

Poker Plaver. Waterman [30]

Purpose: Learn a good strategy for making bets in draw poker.
Environment: Set of all legal poker game states.
Performance Element: Applies the learned production rules to generate actions in a poker game, e.g.. bets.
Instance Selector: Selects each game state derived by play against an opponent as a training instance.
Critic: Two versions of the program use two different critics. In both cases the critic performs the following functions: Evaluation - decides whether the poker bet made by the Performance Element was acceptable. Diagnosis - gives important state variables for deciding the correct bet. Therapy - provides the bet which the Performance Element should have made. In "explicit" learning the critic is an expert poker player , either human or programmed. In "implicit" learning, the evaluation and therapy are deduced from the next action of the opponent and a set of predefined axioms, while diagnosis is read from a predefined "decision matrix".
Learning Element: Modifies and adds production rules to the system. Mistakes are corrected by adding a new rule in front of the rule responsible for the incorrect response.
World Model: Rules of poker, features used to describe the game state, the language of production rules, heuristics for updating the rule base, the model of an opponent.

Model Reference Adaptive Control. Landau [14]

Purpose: Construct a "controller" which preprocesses inputs to an existing system (called the "plant"). The behavior of the combined controller-plant system is to mimic the behavior of a third system (called the "reference model") on the training data.
Environment: The plant to be controlled, and the set of possible inputs (including disturbances).
Performance Element: The controller - a system whose output is used as input to the plant. Its behavior is a function of the input signal, past I/O behavior of the plant, and a set of adjustable parameters.
Instance Selector: Accepts data sequence (as input to the controller) from the environment.
Critic: Evaluation - applies a measure of performance which is some function of the arithmetic difference between the plant and reference model outputs. In some cases the reference model is mathematically defined, and can therefore be considered part of the critic.

In other cases the reference model is an actual system, and is considered part of the environment.

Learning Element: Modifies the parameters of the performanceelement (controller), depending on the performance measure supplied by the critic.

World Model: Control theory assumptions (time invariance, linearity, etc.) and techniques, and the standard of performance embodied in the critic.

References

1. H. G. Barrow and R. J. Popplestone, "Relational Descriptions in Picture Processing," in Machine Intelligence 2, B. Meltzer and D. Michie" eds., American Elsevier, N. Y., 1972, pp. 377-396.

2. B. G. Buchanan, "Scientific Theory Formation by Computer," Proceedings of the NATO Advanced Study Institute on Computer Oriented Learning Processes, Bonas, France, 197*4.

3. B. G. Buchanan and T. M. Mitchell, "Model-Directed Learning of Production Rules," presented at the Workshop on Pattern-Directed Inference Systems, Honolulu, Hawaii, May 1977.

4. R. Davis, "Applications of Meta-Level Knowledge to the Construction, Maintenance, and Use of Large Knowledge Bases " STAN-CS-76-552, Stanford University, July 1976.

5. R. 0. Duda and P. E. Hart, Pattern Classification and Scene Analysis. Wiley, N.

6. D. D. Donalson and F. H. Kishi, "Review of Adaptive Control Theories and Techniques," in C. T. Leondes, ed., Modern Control Systems Theory. McGraw-Hill, N. Y., 1965, pp.

7. E. A. Feigenbaum, "The Simulation of Verbal Learning Behaviour," in E. A. Feigenbaum and J. Feldman, eds., Computers and Thought, McGraw-Hill, N. Y., 19637 PP. 297-W.

8. E. A. Feigenbaum, B. G. Buchanan, and J. Lederberg, "On Generality and Problem Solving: A Case Study Using The DENDRAL Program," in Machine Intelligence 6, B. Meltzer and D. Michie, eds., American Elsevier, N. Y., 1971, pp. 165-190.

9. R. Fikes, P. Hart and N. J. Nilsson, "Learning and Executing Generalized Robot Plans," Artificial Intelligence 1972, 3, pp.251-288.

10. K. S. Fu. "Learning Control Systems - Review and Outlook," IEEE Trans. on Automatic Control, Vol. 15, No. 2, April 1970, pp. 210-221.

11. R. M. Glorioso, Engineering Cybernetics. Prentice-Hall, Englewood Cliffs, N. J., 1975.

12. E. B. Hunt Artificial Intelligence. Academic Press, N. Y., 1975.

13. T. S. Kuhn, TUfi. Structure of Scientific Revolutions, 2nd edition, Univ. Chicago Press,

14. I. D. Landau, "A Survey of Model Reference Adaptive Techniques - Theory and Applications," Automatica, Vol. 10, No. 4, July 1974, pp. 353-379.

15. V. R. Lesser, R. D. Fennell, L. D. Erman. and D. R. Reddy, "Organization of the HEARSAY II Speech Understanding System " IEEE Trans, on Acoustics, Speech, and Signal Processing, Vol. ASSP-23 NO 1 February 1975 nn. 11-23.

16. J. McCarthy, "Programs with Common Sense," in M. Minsky ed., Semantic Information Processing. MIT Press"Cambridge,Mass., 1968; pp? 1403-418.

17. D.Michie,on Machine Intelligence, Wiley, N.

18. M. Minsky, "Steps Toward Artificial Intelligence," in E. A. Feigenbaum and J. Feldman, eds., Computers and Thought. McGraw-Hill, N. Y., 19637 PP 406-450

19. N. J. Nilsson, Learning Machines. McGraw-Hill, N. Y., 1965.

20. R. J. Popplestone, "An Experiment in Automatic Induction," in Machine Intelligence 5, B. Meltzer and Michie, eds., Edinburgh University Press, 1969.

21. A. L. Samuel, "Some Studies in Machine Learning Using the Game of Checkers," in E. A. Feigenbaum and J. Feldman, eds.. Computers and Thought. McGraw-Hill, N. Y., 1963, pp7 7 1 - 1 *

22. A. I.. Samuel, "Some Studies in Machine Learning Using the Game of Checkers II Recent Progress," IBM Journal of Research and Development, Vol. 11, No. 6, November 1967, pp. 601-617.

23. 0. G. Selfridge and U. Neisser, "Pattern Recognition by Machine," in E. A. Feigenbaum and J. Feldman, Computers and Thought. McGraw-Hill, N. Y., 1963; pp. 237-250.

24. H. A. Simon and G. Lea, "Problem Solving and Rule Induction: A Unified View." in L. W. Gregg, ed., Knowledge and Cognition. Lawrence Erlbaum Associates, PotomacMaryland, 1974, pp. 105-127.

25. J. Sklansky "Adaptation, Learning, Self-Repair, and Feedback," IEEE Spectrum, Vol. 1, No. 5, May 1964, pp. 172-174.

26. G. J. Sussman, "A Computational Model of Skill Acquisition," MIT AI-TR-297, August 1973.

27. E. Tse and Y. Bar-Shalom, "Actively Adaptive Control for Nonlinear Stochastic Systems," IEEE Proceedings, Vol. 64, No. 7, August 1976, pp. 1172-1181.

28. Ya. Z. Tsypkin, "Self Learning - What Is It? " IEEE Trans, on Automatic Control. Vol. AC-13, No. 6, December 1968, pp. 608-612.

29. U Uhr and C. Vossler, "A Pattern-Recognition Program That Generates, Evaluates, and Adjusts Its Own Operators," in E. A. Feigenbaum and J. Feldman, eds., Computers and Thought. McGraw-Hill, N. Y., 1963, pp. 25T268.

30. D. A. Waterman, "Generalization Learning Techniques for Automating the Learning or Heuristics," Artificial Intelligence, Vol. 1, Nos. 1 and 2, 1970, pp. 121-170.

31. D. A. Waterman, "Adaptive Production Systems," IJCAI4 Proceedings, Tbilisi, USSR, September 1975, PP. 296-303*

32. P. H. Winston, ed.. The Psychology of Computer Vision. McGraw-HilT9757

33. B. Wittenmark, "Stochastic adaptive control methods: a survey," Int. J. Control, Vol. 21, No. 5, May 1975, pp. 705-730.