An Overview of Some Recent Machine Learning Research and Applications

Brent A. Metfessel and Paul O'Rorke

September 27, 1990

Technical Report 90-31

This work was supported in part by grant IRI-8813048 from the National Science Foundation, a contract with Douglas Aircraft Company, and by a Microelectronics Innovation and Computer Research Opportunities (MICRO) grant from the University of California.

An Overview of Some Recent Machine Learning Research and Applications

Brent A. Metfessel and Paul O'Rorke

September 27, 1990

1 Introduction

A number of definitions of learning exist, reflecting the fact that the term "learning" is ambiguous. We learn when we memorize an address, acquire tennis playing skills, or master a new language but these are very different forms of learning. Definitions of learning range from the trite statement that "learning means never making the same mistake twice" to several pages of philosophical treatise. The founders of the field of Artificial Intelligence define learning pragmatically in terms of improved performance. Marvin Minsky defines learning as "making useful changes in our minds." Herb Simon defines it in terms of changes in a system which enable it to do the same task more effectively the next time (Simon, 1983).

Machine Learning (ML) is the subfield of AI exploring computational approaches to learning. ML researchers adopt a practical point of view and consider learning to refer to a change of state of a system such that the system in question can perform a task in a manner closer to the optimum performance (e.g., faster or more efficiently). A number of machine learning paradigms are presently used or being actively researched. In this brief survey we focus on some recent work on symbolic approaches to inductive learning from examples and explanation-based learning (EBL) but we also discuss other machine learning research paradigms such as connectionist networks and genetic algorithms.

2 Symbolic Approaches to Learning

The most prominent class of methods for learning from examples is sometimes called "inductive," "empirical," or "similarity-based learning" (SBL). We will use the convenient label SBL even though differences are as important to these methods as similarities. SBL is the best understood of the machine learning methods, and commercial applications exist. The primary application of SBL is in the development of expert systems.

2.1 General Considerations

The "knowledge engineering bottleneck" is a well-known phenomenon in expert system development (Feigenbaum, 1979), where domain knowledge to be encoded in the expert system's database is obtained by interviews of domain experts by computer specialists called "knowledge engineers." Since they often do not speak the same language, the interviews can go on for months. In addition, much of the experts' knowledge is tacit knowledge; that is, the expert is not able to articulate or make explicit all that the many years of experience in the domain area has provided him. SBL ameliorates this difficulty by using actual examples from the domain area that the machine learns to classify. SBL algorithms typically require a collection of *positive* examples of a concept or category (e.g., a collection of lung cancer cases) and a collection of *negative* cases (e.g., a collection of patient cases without lung cancer but with abnormal chest x-rays). The positive examples are required for generalizing the categories or rules that the machine learns, whereas the negative cases are important for category specialization —narrowing the categories down so as to not include the negatives while still including the positive cases in the relevant category. The hope is then that future instances given as input will be properly categorized.

Naturally, an inductive learning system needs to have the input and the categories/concepts that it needs to learn given in a machine-readable representation language. There is a tradeoff between flexibility and tractability— more flexible and complex representation languages typically require more computer space and time (Clark, 1990). A relatively simple and efficient way of representing examples is in terms of "feature vectors." A medical diagnosis example of clinical depression might be represented as a feature vector:

Exam: mental status
Rate of Speech: slow
Mood: depressed
Affect: blunted
Hallucinations: None
Thought content: Moderate suicidal ideation
Category: +

Probably the best-known example of inductive learning is the ID3 algorithm developed by J. R. Quinlan (Quinlan, 1986b). This algorithm is based on earlier work on CLS, a "Concept Learning System" (Hunt, Marin, & Stone, 1966). From feature vectors given to the system, a decision tree consistent with the positive and negative examples is constructed. The object of ID3 is to obtain the simplest tree that accomplishes this goal, using an information-theoretic measure of uncertainty called entropy. The feature that gives the minimum average uncertainty about the class is the one that provides the most information about the class, so it is selected for testing at the next tree node.

The AQ algorithm, developed by R. S. Michalski in the early '70s (Michalski, 1975), builds conceptual descriptions which expand outwards from positive examples while being careful not to cover cells in the feature space which contain negative examples. AQ learns *rules* from input examples instead of building decision trees.

2.2 Evaluations of SBL Methods

Several studies and evaluations have been done of aspects of individual symbolic concept learning algorithms such as ID3. In these studies variants of a single ML method are tested on the same data. Quinlan has studied the effects of noise on concept learning (Quinlan, 1986a), methods for simplifying decision trees (Quinlan, 1987b), and methods for generating production rules from trees (Quinlan, 1987a). For a comparison of different tree pruning strategies, see Mingers (1989).

Comparisons of alternative SBL strategies also exist. A comparison of AQ and ID3 (O'Rorke, 1982) found that AQ tended to be more expensive computationally but it produced more comprehensible and more accurate concept descriptions than ID3. A study by Shavlik and Mooney found that AQ's tendency to fit the data more closely than ID3 can be a liability when the training examples include noise (Mooney, Shavlik, Towell, & Gove, 1989).

A relatively new inductive algorithm called CN2 aims to "combine the efficiency and ability to cope with noisy data of ID3 with the if-then rule form and flexible search strategy of the AQ family" (Clark & Niblett, 1989). CN2 outputs a decision list, which is an ordered set of if-then rules. In experiments on various domains, CN2 was found to have performance comparable (in terms of efficiency and ability to deal with noisy data) to ASSISTANT. Developed by I. Kononenko, I. Bratko, and E. Roskar in 1984, ASSISTANT is a descendent of ID3 that outputs a decision tree but also has a tree pruning mechanism for processing noisy data. [1]

2.3 Applications of SBL

Of all the machine learning techniques presently available, SBL is the most fully utilized in practical applications. Perhaps the earliest application of ML methods to a knowledge engineering task was a comparison by Michalski and Chilausky of rules produced by AQ against a hand-coded expert system in the domain of soybean disease diagnosis. Seventeen categories of disease were possible. The AQ system was correct 98 percent of the time, whereas the expert system was correct only 72 percent of the time (Michalski & Chilausky, 1980).

The ACLS (Advanced Concept Learning System) based on CLS and ID3 was developed in 1981 by Intelligent Terminals Limited. ACLS was a UCSD-PASCAL program that produced a Pascal conditional expression to be used as a classification rule (Paterson & Niblett, 1982). ACLS later led to the development of a number of commercially available inductive learning systems including Expert-Ease (1983), EX-TRAN (1984), and RuleMaster (1984). Expert-Ease, for example, ran on an IBM PC or compatible with at least 128K RAM and a double-sided/double-density disk drive (Milman, 1984). According to a flyer distributed by Expert Systems, Inc. in 1984, the system cost approximately $2000 and had been purchased by at least 300 major corporations including Exxon, General Motors, Miami VA Medical Center and General Electric.

Kinney, Cortada, Seinfeld, and Keck (1984) described a study done in Miami concerning medical patients admitted to a cardiac care unit (CCU) with suspected MI (myocardial infarction or "heart attack"). Using six attributes (quality of chest pain, results of a chest

[1] Interestingly, a Bayesian classifier, using the independence assumption, did not perform significantly less accurately than the other algorithms tested.

x-ray, etc.) used for diagnosis of MI, Expert-Ease was given 25 training examples and 50 test examples. The patient cases were classified into one of three groups: acute MI, angina, or non-cardiac causes for the chest pain. About 60% of patients admitted to the CCU at this center with suspected MI actually had an MI. The decision rule constructed by Expert-Ease correctly classified patients 94 percent of the time (47 out of 50 correct). This compared quite favorably with human physicians, who are often overly cautious (Kinney et al., 1984). [2]

Carter and Catlett (1987) investigated ID3 as applied to assessing whether credit card applications should be accepted or rejected. The decision was based on certain risk parameters, such as "amount of money in the bank", "length of time at the present address", etc. This is a difficult domain even for humans, and the researchers note that depending on experience and other factors, assessments are often inconsistent between assessors as well as time-consuming. In the investigation, there were 690 credit applications available, 340 of which were used as ID3's training set. After approximately 30 decision trees were built, 150 examples were provided for each decision tree in a testing paradigm. As a result of the testing, attributes, selection criteria, and other parameters were chosen for the performance run (200 samples). During the performance run, both the existing method and ID3 were compared and it was noted from a one sided t-test that the differences were not significant (both categorized the applications with about an 80 percent success rate). However, the C4 algorithm, an extension of ID3 with the addition of a pruning algorithm that collapses subtrees into leaves, performed with about an 85 percent success rate (noted to be statistically significant when compared to the existing method). Other remarkable aspects of this study included the high number of examples needed for SBL to properly operate, as well as the difficulty that was experienced with cutoff points for attributes with numeric values.

Carter and Catlett reviewed other commercial uses of inductive learning packages, including fault diagnosis for printed circuit boards (ITT Europe) and assessment of engine performance of NASA's space shuttle. Westinghouse Electric's nuclear fuel division used inductive learning techniques to improve a process control application, resulting in at least 10 million dollars per year in additional revenue (Carter et al., 1987).

3 Explanation-Based Learning

3.1 General Considerations

Mitchell et al. give a concise summary of EBL when they state that "the characteristic common to these methods is that their ability to *generalize* from a single example follows from their ability to *explain* why the training example is a member of the concept being learned" (Mitchell, Keller, & Kedar-Cabelli, 1986). The way EBL generates an explanation of an example from its domain theory is through unification, computing the most general unifier that allows the domain knowledge to be connected together in a general way. The explanations are usually deductive consequences of inference rules provided prior to learning

[2] Despite such favorable results it should be noted that SBL and expert systems are not widely used in medicine with the exception of a few teaching hospitals. This is likely due to the conservatism of the medical community as a whole and the brittleness of expert systems in general (i.e., expert systems suffer abrupt deterioration in accuracy when taken outside their relatively narrow domain of expertise).

(DeJong & Mooney, 1986). DeJong (1988) gives an excellent introduction to EBL. Overviews of some current research issues in explanation-based learning may be found in Minton (1990) and in Holder (1990).

3.2 Variants of EBL

A number of versions of EBL exist. Some important variants are described below.

3.2.1 Learning Apprentice Systems

One type of EBL-based system is the learning apprentice. The knowledge base here is interactive, and the machine learns through observation and analysis of the problem-solving methods that the users engage in during their *normal* utilization of the machine. An early study of EBL in an apprenticeship context was Paul O'Rorke's work on the "Mathematician's Apprentice" project (O'Rorke, 1986b). O'Rorke's system, MA, used explanation-based acquisition of schemata (knowledge collections such as frames or scripts). MA starts with very limited problem-solving ability, observes the teacher's behavior, explains why the behavior is successful, and learns new schemata as well as rules governing their use.

A more applications oriented system known as LEAP (Mitchell, Mahadevan, & Steinberg, 1986) was implemented as an augmentation to VEXED, a knowledge-based assistant in VLSI design. From training examples as well as knowledge previously encoded in the knowledge base, LEAP used explanation-based generalization to come up with general rules. Training examples provided by the user are tailored to focus mainly on refining knowledge discovered missing from the knowledge base.

The PRODIGY system (Carbonell, Knoblock, & Minton, 1989), developed at Carnegie Mellon University, is an integrated learning apprentice, problem-solver and planner. The system has been tested in a number of domains including blocks world, matrix multiplication, robotic planning, and machine-shop scheduling. The EBL module of PRODIGY develops control rules (rules controlling search) from a problem-solving trace, which consists of the entire search tree for the solution, including failed attempts. PRODIGY is able to learn both from a successful search path and from unsuccessful ones (the latter paths leading to rejection rules, used to filter nonviable solution paths). After a control rule is developed from an example, empirical utility tests are run during subsequent problem-solving to determine if the rule should be kept active. In other words, a rule is kept only if the cumulative savings in search time significantly outweighs the cumulative cost (in time) that it takes to match the rule. In addition, a user-interface allows dialog with a teacher (expert user), allowing the teacher to evaluate and offer guidance to the system's learning and problem-solving. If the system has difficulty with a problem, it is able to point out where the difficulty occurs, thus allowing the teacher to communicate with the system. This may involve communicating the necessary domain knowledge for the machine to come up with a solution, or even providing the solution directly (i.e., providing an operator sequence that solves the problem or subproblem). PRODIGY may also ask the teacher for reasons why the method s/he provided is more efficient or applicable than other solutions which might seem to apply.

3.2.2 Learning by Understanding Explanations

An example of the LBUE paradigm was developed by Martin and Redmond for causal modeling. Their system, EDSEL, is an expert diagnostic system for automobiles developed on a causal model. They develop a scenario where a car has stalled and attempts are made to diagnose the problem. Assume the system is missing some causal relationships and backward chaining is attempted:

(not(run engine)) ←
  (not(spin crankshaft)) ←
    (not(down-stroke cylinder)) ←
      (not(combustion cylinder))

The machine could not reason beyond this level due to an incomplete causal model. Then the authors had a fictitious mechanic hypothesize that a stuck butterfly valve is the cause. Forward chaining is attempted:

(not(movable butterfly-valve)) →
  (low(flow air carburetor))

Once again, the reasoning goes no farther and the forward and backward chains do not meet. However, the mechanic then puts in the explanation that low carburetor air flow leads to a low air/gas mixture which causes difficulty with cylinder combustion. The forward and backward chains meet and the following explanation results:

(not(movable butterfly-valve)) →
  (low(flow air carburetor)) →
    (low(mix air gas)) →
      (not(combustion cylinder)) →
        (not(down-stroke cylinder)) →
          (not(spin crankshaft)) →
            (not(run engine))

When collapsed, this causal relationship becomes:

(not(movable butterfly-valve)) → (not(run engine))

This new explanation is then added to the domain knowledge of the machine (Martin & Redmond, 1989).

3.2.3 Generalization to N

One of the goals of explanation-based learning is to filter out extraneous details of examples so that learning can focus on causally relevant essentials. Unfortunately, the original techniques (implemented in a domain-independent fashion in systems such as EGGS (Mooney & Ben-

nett, 1986), PROLOG-EBG (Kedar-Cabelli & McCarty, 1987), and PROLEARN (Prieditis & Mostow, 1987)) fail to filter out all the irrelevant details when applied to examples involving recursive or iterative concepts. Since repetition and recursion occur in important domains such as programming and circuit design, researchers have begun to study extensions of EBL designed to deal with recursion.

3.3 Evaluations of Explanation-Based Learning

Explanation-based learning techniques for improving a number of different performance elements have been evaluated. For example, O'Rorke (1986a) described EBL experiments on logical problem solvers. In his award winning paper at AAAI-87, Keller (1987) described an EBL system for learning in the context of symbolic integration problem solving. Minton (1988) described results of empirical tests of EBL methods for learning search control rules in the context of GPS-based planning and scheduling. Mooney (1987) described a general EBL system and its application to text comprehension and planning. Shavlik (1988) studied EBL in the context of physical reasoning and planning tasks. Segre (1988) described EBL for robotic manipulators.

Experimental analyses of learning problem solvers have demonstrated that the manner of integration is important. In experiments comparing rote learning and EBL, O'Rorke found that simply plugging a learning method into a performance system in the obvious way may lead to anomalous results. An experimental analysis found that an initial problem solver had control features that hampered both rote and explanation-based learning (O'Rorke, 1989). Since there can be subtle, unanticipated interactions between performance and learning elements, it is important to integrate them carefully so that learning maximizes improvements in performance.

Early research on EBL made the implicit assumption that EBL was always a good idea. However, experiments have shown that EBL sometimes hurts performance rather than improving it (Minton, 1988). The "utility problem" revolves around the question of what to learn and when to use explanation-based learning so that overall performance improves. Mooney et al. (1989) attempted to make generalizations reconciling earlier (seemingly contradictory) results.

A recent analysis of PRODIGY/EBL (Etzioni, 1990) indicated that standard EBL can cause performance to deteriorate in domains where explanations of both success and failure are recursive but a static evaluation technique can be used to learn search control rules avoiding the recursion. Alternatively, Shavlik (1990) described a method for doing EBL in recursive domains and reported on encouraging experimental results of applying a system called BAGGER2 to blocks-world planning problems. Letovsky described an alternative approach for learning recursive concepts using structural induction (Letovsky, 1990). Subramanian analyzed a cost model to determine when learning recursive concepts is likely to "pay off" (Subramanian & Feldman, 1990). Bhatnagar and Mostow described an adaptive planner called FS2 that performed well in recursive domains (Bhatnagar & Mostow, 1990).

3.4 Applications of EBL

Most studies of EBL have been performed using unrealistic domain theories, e.g., "micro-worlds" like blocks-world or STRIPS-world. EBL has not yet progressed to the level of practical applicability attained by SBL. While still controversial (Davis, 1990; Keller, 1990), initial results of attempts to apply variants of EBL to relatively realistic domains such as model-based diagnosis (El Fattah & O'Rorke, 1990; Resnick, 1989), design (Blumenthal, 1990; Kellogg et al., 1989), and scheduling (Eskey & Zweben, 1990) are encouraging.

4 Connectionist Approaches to Learning

Connectionist methods, "parallel distributed processing" (PDP) (McClelland & Rumelhart, 1988), and neural network methods involve highly parallelized networks of computational units somewhat analogous to a biological network of neurons, with each "neuron" being able to excite (through positive activation) or inhibit (through negative activation) other units to which it is connected. The neurons, or units, within the network are usually arranged in a number of layers, including an "input" layer and an "output" layer. In between these two layers are "hidden layers". Generally, the input layer is activated first by a stimulus, with resulting activation spreading to other network units until the output layer becomes activated. It usually takes many cycles of spreading activation for the network to reach a stable state, with each unit having a numeric weight (activation level). This weight can be positive (representing excitatory activation) or negative (representing inhibitory activation) (Ford, 1989).

To control the spread of activation, algorithms such as backpropagation and Boltzman learning are used. Although backpropagation is the faster of the two algorithms and used more commonly, the required continuous error feedback in backpropagation makes it too slow for many practical applications, and thus new research areas such as "unsupervised backpropagation" and "generative backpropagation" have emerged (Fayyad, Laird, & Irani, 1989).

A neural network system does not need predetermined rules to exhibit learning phenomena. Data (generally in the form of examples and their classifications) is provided to the input layer during a training session and activation spreads throughout the system. During the activation cycles, the hidden layer units act as feature detectors; that is, each neuron unit will look for a certain feature (or collection of features) and give a strong response when that pattern is found. The output layer then constructs its output pattern based on the feature patterns that the hidden layer has detected. The purpose of the network training session is to generate a set of feature detectors in the hidden layer which results in a consistent and appropriate output layer response. Although neural networks can learn without any previous domain knowledge, better and faster training can often be achieved when a preset bias can be set up within the network, the bias being based on the domain features that an expert thinks are more important (Caudill, 1989). Once the training is complete, the network can then process novel examples.

5 Genetic Algorithms

Genetic algorithms are modeled on biological evolutionary systems. The environment is represented as strings of symbols, most commonly binary strings of 0 and 1. Each point in the problem space is represented by a unique string of these symbols. These strings are the "genetic material" (chromosomes) with each 0 or 1 symbol being an allele. The system maintains a certain population of such strings, and uses as feedback a fitness measure (or process performance measure). These strings compete against each other, recombine with each other, and have a finite chance of having the value of an allele change (mutate). Those chromosomes with the highest fitness factor have the most chance of creating offspring (time is also measured in intervals known as generations). The basic cycle is as follows: first, the candidates from the existing solutions are partitioned, with the highest fitness candidates selected out for reproduction; next, pairs of chromosomes are randomly chosen for crossover mating (exchanging genes with each other); then the mutation operator is applied, and the weakest performers are eliminated using the fitness function. The process is then iterated (Austin, 1990).

6 Comparison

Overall, it appears that SBL methods are noise tolerant and work well in domains where not much theory is known. However, they typically require large numbers of examples to ensure good performance.

One advantage of EBL is that the explanation, often in the form of a proof tree, justifies the generalization made on the example in terms of the domain theory that the machine uses. Thus, unlike SBL, EBL does not need the hefty amount of input data that SBL often must have. The domain theory and the ability to explain examples provided to EBL systems can substitute for large numbers of examples. However, EBL does not enable the machine to learn anything fundamentally new. There are no "inductive leaps" involved in EBL. Rather, EBL is used to learn to solve problems *faster*.

EBL is sensitive to imperfections in the domain theory. If the domain theory is incorrect, the justifications provided by explanations may be false. If the domain theory is incomplete, the EBL system may be unable to provide an explanation and unable to learn from some experiences. A third difficulty is the intractability problem, where computer space or time may be inadequate to develop an explanation (which might occur in very large or very complex domain theory spaces). Such imperfections are common in complex, real world domains such as medicine. EBL is most useful when the domain can be well-formalized. Since many domains can be partly formalized but there are not enough examples to ensure good results with SBL, many researchers are working on hybrid SBL/EBL algorithms. Previously encoded domain knowledge could enable SBL algorithms to efficiently generate decision rules less subject to inductive errors, whereas the additional examples provided by SBL could make up for some incompleteness in the domain theory thereby enhancing the performance of EBL.

The advantages of neural networks include massive parallelism and the ability to perform well in the presence of incomplete, partly incorrect, or conflicting data. These networks are

also interesting in that they offer models of biological neural systems, and thus neural network designs may be able to take advantage of new findings in neuroscience. Neural nets have been most successful in perceptual problems such as character-recognition, as well as in other tasks where the preexisting domain knowledge is sparse. On the other hand, neural nets can take a long time to reach stability. See Fahlman (1988) for an empirical study of the convergence speed of backpropagation learning. Several researchers feel that connectionist methods are in general much slower than other well-established symbolic learning methods such as ID3. Furthermore, once a neural system has reached stability (i.e., is "tuned"), it becomes very difficult to track down the reasons for its success on some cases and its failure on others. As for the algorithms that run the network, a hill-climbing method is often used, which can lead to premature halting at local maxima or minima (Buchanan, 1989). See Dietterich, Hild, and Bakiri (1990) and Mooney et al. (1989) for comparisons of connectionist and symbolic approaches to learning.

Genetic algorithms are most useful when the search space is poorly understood, large and complex. 500 to 1000 samples may be needed in order to properly evaluate the genetic structures for fitness, and thus are not appropriate for domains where large samples cannot be found. In this respect, genetic algorithms have requirements similar to those of SBL. The purpose of the fitness measure is then to select the best-performing structures, which means that the domain must not be so poorly understood that a fitness evaluation function cannot be constructed (e.g., desired goal states must be clearly known). It may be possible to utilize a hybrid system of SBL and genetic algorithms; that is, learning a fitness evaluation scheme through SBL, then utilizing the scheme via a genetic algorithm. Genetic algorithms can learn quite quickly—significant improvements in fitness have been seen after 10 generations. Often the systems employing genetic algorithms are separated into two components—a genetic algorithm-based learning component and a task component with which to effect a behavior change. Genetic algorithms have been used to change parameters, data structures, and even executable code (DeJong, 1988).

TABLE OF MACHINE LEARNING METHODS

ML Method	*Advantages*	*Disadvantages*
Similarity-Based Learning	-Little or no knowledge of domain theory needed -Can handle noisy domains -Has shown practical value	-Often needs much data in the form of examples -Currently suited more for classification than for problem-solving
Explanation-Based Learning	-Uses knowledge effectively -Needs few examples (often only one) -Can make a slow problem solver run faster	-Sensitive to imperfections in domain theory (e.g., incorrectness) -Does not learn "new information" -Not demonstrated to be practically useful as yet
Genetic Algorithms	-Work well in large, complex and ill-defined search spaces.	-Need large populations -Computationally expensive
Neural Networks	-Can handle noisy and conflicting data -Highly parallel processing -Does not need domain knowledge	-Slow to reach stability -Hill climbing subject to errors -Difficult to understand reasons for success and failure

7 Conclusion

This report is an executive summary of recent research on similarity-based learning, explanation-based learning, connectionist approaches to learning, and genetic algorithms. No attempt was made to make this a comprehensive survey. Instead, we have attempted to give pointers to more complete treatments of specific topics. We have attempted to say something about the conditions of applicability of various machine learning methods and we have attempted to characterize current knowledge about the advantages and disadvantages of competing approaches. It should be noted, however, that research on identifying strengths and weaknesses of ML methods is still in its infancy; new results are appearing all the time and occasionally these lead to changes in the interpretation of the results of previous experimental comparisons.

Progress is being made on "hybrid" algorithms; that is, combinations of two or three machine learning techniques in one system. It is hoped that hybrid systems will be able to exploit the advantages of complementary ML methods while avoiding their disadvantages. An example of a hybrid system integrating EBL and connectionist learning is described in Towell, Shavlik, and Noordewier (1990). For a recent example of an integration of EBL and SBL, see Vilain, Koton, and Chase (1990). Bergadano et al. describe an integration of EBL and SBL applied to real world vibration analysis problems in Bergadano, Giordana, Saitta, Marchi, and Brancadori (1990).

References

Austin, S. (March 1990). An introduction to genetic algorithms. *AI Expert*, 49–53.

Bergadano, F., Giordana, A., Saitta, L., Marchi, D. D., & Brancadori, F. (1990). Integrated learning in a real domain. *Proceedings of the Seventh International Conference on Machine Learning* (pp. 322–329). Austin, TX: Morgan Kaufmann.

Bhatnagar, N., & Mostow, J. (1990). Adaptive search by explanation-based learning of heuristic censors. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 895–901). Boston, MA: AAAI Press/ The MIT Press.

Blumenthal, B. (1990). Empirical comparisons of some design replay algorithms. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 902-907). Boston, MA: AAAI Press/ The MIT Press.

Buchanan, B. G. (1989). Can machine learning offer anything to expert systems? *Machine Learning, 4*, 251–254.

Carbonell, J. G., Knoblock, C. A., & Minton, S. (1989). *PRODIGY: An integrated architecture for planning and learning.* (Technical Report CMU- CS-89-189). Pittsburgh, PA: Carnegie Mellon University, School of Computer Science.

Carter, C., & Catlett, J. (Fall 1987). Assessing credit card applications using machine learning. *IEEE Expert*, 71–79.

Caudill, M. (December 1989). Using neural nets: Representing knowledge (Part 1). *AI Expert*, 34–41.

Clark, P. (1990). *Machine learning: Techniques and recent developments* (Technical Report TIRM-90-041). Glasgow, Scotland: The Turing Institute.

Clark, P., & Niblett, T. (1989). The CN2 induction algorithm. *Machine Learning*, *3*, 261–283.

Davis, R. (1990). Form and content in model based reasoning. *Proceedings of the AAAI-89 Workshop on Model Based Reasoning* (pp. 11–27). Detroit, MI: Boeing Computer Services.

DeJong, G. F. (1988). An introduction to explanation-based learning. In H. Shrobe (Ed.), *Exploring artificial intelligence.* San Mateo, CA: Morgan Kaufmann.

DeJong, G. F., & Mooney, R. (1986). Explanation-based learning: An alternative view. *Machine Learning*, *1*, 145–176.

DeJong, K. (1988). Learning with genetic algorithms: An overview. *Machine Learning*, *3*, 121–138.

Dietterich, T. G., Hild, H., & Bakiri, G. (1990). A comparative study of ID3 and backpropagation for English text-to-speech mapping. *Proceedings of the Seventh International Conference on Machine Learning* (pp. 24–31). Austin, TX: Morgan Kaufmann.

El Fattah, Y., & O'Rorke, P. (1990). *Learning multiple fault diagnosis* (draft submitted to CAIA-91: The IEEE Conference on AI Applications). Department of Information and Computer Science, University of California, Irvine.

Eskey, M., & Zweben, M. (1990). Learning search control for constraint-based scheduling. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 908–915). Boston, MA: AAAI Press/ The MIT Press.

Etzioni, O. (1990). Why PRODIGY/EBL works. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 916–922). Boston, MA: AAAI Press/ The MIT Press.

Fahlman, S. (1988). *An empirical study of learning speed in back-propagation networks* (Technical Report CMU-CS-88-162). Pittsburgh, PA: Carnegie Mellon University, School of Computer Science.

Fayyad, U. M., Laird, J. E., & Irani, K. B. (1989). Conference Report: The Fifth International Conference on Machine Learning. *AI Magazine*, 79–84.

Feigenbaum, E. A. (1979). Themes and case studies of knowledge engineering. In D. Michie (Ed.), *Expert systems in the micro electronic age.* Edinburgh: Edinburgh University Press.

Ford, N. (1989). From information to knowledge management: The role of rule induction and neural net machine learning techniques in knowledge generation. *Journal of Information Science, 15*, 299–304.

Holder, L. B. (1990). The general utility problem in machine learning. *The Seventh International Conference on Machine Learning* (pp. 402–410). Austin, TX: Morgan Kaufmann.

Hunt, E. B., Marin, J., & Stone, P. (1966). *Experiments in induction.* New York: Academic Press.

Kedar-Cabelli, S. T., & McCarty, L. T. (1987). Explanation-based generalization as resolution theorem proving. *Proceedings of the Fourth International Workshop on Machine Learning* (pp. 383–389). Irvine, CA: Morgan Kaufmann.

Keller, R. M. (1987). Defining operationality for explanation-based learning. *The Sixth National Conference on Artificial Intelligence* (pp. 482–487). Seattle, WA: Morgan Kaufmann.

Keller, R. M. (1990). In defense of compilation: A response to Davis' "Form and content in model based reasoning". *Proceedings of the AAAI-89 Workshop on Model Based Reasoning* (pp. 22–31). Boston, MA: Boeing Computer Services.

Kellogg, C., R. A. Gargan, J., Mark, W., McGuire, J. G., Pontecorvo, M., Schlossberg, J. L., & Sullivan, J. W. (April 1989). The acquisition, verification, and explanation of design knowledge. *SIGART Newsletter*, 163–165.

Kinney, E. L., Cortada, X., Seinfeld, J., & Keck, D. (1984). A prototype automated artificial intelligence for classifying chest pain. *Circulation, 70.*

Letovsky, S. (1990). Operationality criteria for recursive predicates. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 936–941). Boston, MA: AAAI Press/ The MIT Press.

Martin, J. D., & Redmond, M. (April 1989). Acquiring knowledge by explaining observed problem solving. *SIGART Newsletter*, 77–83.

McClelland, J. L., & Rumelhart, D. E. (1988). *Explorations in parallel distributed processing.* Cambridge, MA: MIT Press.

Michalski, R. S. (1975) . Synthesis of optimal and quasi-optimal variable valued logic formulas. *Proceedings of the 1975 International Symposium on Multiple Valued Logic* (pp. 76–87). Indiana University, Bloomington.

Michalski, R. S., & Chilausky, R. L. (1980). Learning by being told and learning from examples: An experimental comparison of the two methods of knowledge acquisition in the context of developing an expert system for soybean disease diagnosis. *International Journal of Policy Analysis and Information Systems, 4*, 125–161.

Milman, J. O. (1984). Do it yourself expert systems with artificial intelligence on a microcomputer. *IEEE Compcon'84 Fall Conference on the Small Computer (R)evolution* (pp. 20–26). Arlington, VA: IEEE Press.

Mingers, J. (1989). An empirical comparison of pruning methods for decision tree induction. *Machine Learning, 4*, 227–243.

Minton, S. (1988). Quantitative results concerning the utility of explanation-based learning. *Proceedings of the Seventh National Conference on Artificial Intelligence* (pp. 564–569). St. Paul, MN: Morgan Kaufmann.

Minton, S. (1990). Issues in the design of operator composition systems. *Proceedings of the Seventh International Conference on Machine Learning* (pp. 304–312). Austin, TX: Morgan Kaufmann.

Mitchell, T. M., Keller, R. M., & Kedar-Cabelli, S. T. (1986). Explanation-based generalization: A unifying view. *Machine Learning, 1*, 47–80.

Mitchell, T. M., Mahadevan, S., & Steinberg, L. I. (1986). A learning apprentice system for VLSI design. In T. M. Mitchell, J. G. Carbonell, & R. S. Michalski (Eds.), *Machine learning: A guide to current research.* Boston, MA: Kluwer.

Mooney, R., & Bennett, S. (1986). A domain independent explanation-based generalizer. *Proceedings of the National Conference on Artificial Intelligence* (pp. 551–555). Philadelphia, PA: Morgan Kaufmann.

Mooney, R., Shavlik, J., Towell, G., & Gove, A. (1989). An experimental comparison of symbolic and connectionist learning algorithms. *Eleventh International Joint Conference on Artificial Intelligence* (pp. 775–780). Detroit, MI: Morgan Kaufmann.

Mooney, R. J. (1987). *A general explanation-based learning mechanism and its application to narrative understanding.* Doctoral dissertation, Coordinated Science Laboratory, University of Illinois at Urbana-Champaign.

O'Rorke, P. (1982). *A comparative study of inductive learning systems AQ11-P and ID-3 using a chess endgame test problem* (Technical Report F-82-899 (ISG82-2)). University of Illinois at Urbana-Champaign, Department of Computer Science.

O'Rorke, P. (1986a). *Explanation-based learning via constraint posting and propagation.* Doctoral dissertation, Department of Computer Science, University of Illinois at Urbana-Champaign.

O'Rorke, P. (1986b). Recent progress on the mathematician's apprentice project. In T. M. Mitchell, J. G. Carbonell, & R. S. Michalski (Eds.), *Machine learning: A guide to current research.* Boston, MA: Kluwer.

O'Rorke, P. (1989). LT revisited: Explanation-based learning and the logic of principia mathematica. *Machine Learning, 4*, 117–159.

Paterson, A., & Niblett, T. (1982). *ACLS User's Manual.* Glasgow, Scotland: Intelligent Terminals Limited.

Prieditis, A. E., & Mostow, J. (1987). PROLEARN: Towards a Prolog interpreter that learns. *Proceedings of the National Conference on Artificial Intelligence* (pp. 494–498). Seattle, WA: Morgan Kaufmann.

Quinlan, J. R. (1986a). The effect of noise on concept learning. In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine learning.* San Mateo, CA: Morgan Kaufmann.

Quinlan, J. R. (1986b). Induction of decision trees. *Machine Learning, 1,* 81–106.

Quinlan, J. R. (1987a). Generating production rules from decision trees. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence* (pp. 304–307). Milan, Italy: Morgan Kaufmann.

Quinlan, J. R. (1987b). Simplifying decision trees. *International Journal of Man-Machine Studies, 27,* 221–234.

Resnick, P. (1989). *Generalizing on multiple grounds: Performance learning in model-based troubleshooting* (M.S. Thesis Technical Report AI-TR 1052). Cambridge, MA: Massachusetts Institute of Technology, MIT Artificial Intelligence Laboratory.

Segre, A. M. (1988). *Machine learning of robot assembly plans.* Boston, MA: Kluwer.

Shavlik, J. W. (1988). *Generalizing the structure of explanations in explanation-based learning.* Doctoral dissertation, Coordinated Science Laboratory, University of Illinois at Urbana-Champaign.

Shavlik, J. W. (1990). Acquiring recursive and iterative concepts with explanation-based learning. *Machine Learning, 5,* 39–70.

Simon, H. A. (1983). Why should machines learn? In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell, (Eds.), *Machine learning: An artificial intelligence approach.* San Mateo, CA: Morgan Kaufmann.

Subramanian, D., & Feldman, R. (1990). The utility of EBL in recursive domain theories. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 942–949). Boston, MA: AAAI Press/ The MIT Press.

Towell, G. G., Shavlik, J. W., & Noordewier, M. O. (1990). Refinement of approximate domain theories by knowledge based neural networks. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 861–866). Boston, MA: AAAI Press/ The MIT Press.

Vilain, M., Koton, P., & Chase, M. P. (1990). On analytical and similarity-based classification. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 867–874). Boston, MA: AAAI Press/ The MIT Press.