Applications of Machine Learning and Rule Induction

Pat Langley[◇]

Robotics Laboratory, Computer Science Dept.
Stanford University, Stanford, CA 94305

Herbert A. Simon

Department of Psychology
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

An important area of application for machine learning is in automating the acquisition of knowledge bases required for expert systems. In this paper, we review the major paradigms for machine learning, including neural networks, instance-based methods, genetic learning, rule induction, and analytic approaches. We consider rule induction in greater detail and review some of its recent applications, in each case stating the problem, how rule induction was used, and the status of the resulting expert system. In closing, we identify the main stages in fielding an applied learning system and draw some lessons from successful applications.

Introduction

Machine learning is the study of computational methods for improving performance by mechanizing the acquisition of knowledge from experience. Expert performance requires much domain-specific knowledge, and knowledge engineering has produced hundreds of AI expert systems that are now used regularly in industry. Machine learning aims to provide increasing levels of automation in the knowledge engineering process, replacing much time-consuming human activity with automatic techniques that improve accuracy or efficiency by discovering and exploiting regularities in training data. The ultimate test of machine learning is its ability to produce systems that are used regularly in industry, education, and elsewhere.

In this paper we examine some recent successes in applying machine learning to real-world problems. First, we review five basic learning paradigms, then focus on one of these: methods for inducing logical rules from experience. We describe eight fielded applications of these methods at some length, then discuss a number of other applications efforts in less detail. We conclude by discussing some of the lessons suggested by these projects.

◇ Also affiliated with the Institute for the Study of Learning and Expertise, 2164 Staunton Court, Palo Alto, CA 94306.

Five Paradigms for Machine Learning

Machine learning is a diverse field, held together by common goals and similar evaluation methods. The general aim is to improve performance on some task, and the general approach involves finding and exploiting regularities in training data. Most evaluation is experimental in nature, aimed at showing that the learning method leads to performance on a separate test set, in one or more realistic domains, that is better than performance on that test set without learning. Despite these similarities, researchers in machine learning tend to associate themselves with one or another of five main paradigms, each of which shares basic assumptions about representation, performance methods, and learning algorithms.

One major paradigm, associated with the area of *neural networks*, represents knowledge as a multilayer network of units that spreads activation from input nodes through internal units to output nodes. Weights on the links determine how much activation is passed on. The activations of output nodes can be translated into numeric predictions or discrete decisions about the class of the input. Approaches to learning within the neural net framework typically improve classification or prediction accuracy by modifying the weights on the links. One common learning algorithm, among the many that have been explored, carries out gradient descent search through the space of weights, modifying them in an attempt to minimize the errors that the network makes on training data. Widrow, Rumelhart, and Lehr (CACM, March, 1994) summarize recent research on neural networks and describe some applications of this approach.

A second framework, known as *instance-based* or *case-based* learning, represents knowledge in terms of specific cases or experiences and relies on flexible matching methods to retrieve these cases and apply them to new situations. One common scheme, known as *nearest neighbor*, simply finds the stored case nearest (according to some distance metric) to the current situation, then uses it for classification or prediction. Case-based learning typically stores training instances in memory; generalization occurs at retrieval time, with much of the power residing in the indexing scheme and the similarity metric used to identify relevant cases, though more sophisticated variants may adapt a retrieved case to the new situation. Allen (CACM, March, 1994) describes the case-based approach, along with some recent applications.

Genetic algorithms, a third paradigm within machine learning, typically represents knowledge by Boolean or binary features, sometimes used as the conditions and actions of rules. The most common interpreter for this knowledge employs an all-or-none matching process, using strengths associated with rules to resolve conflicts. In some cases, a production-system architecture lets rules apply in sequence, producing multi-step behavior. The standard learning operators in genetic algorithms, called *crossover* and *mutation* in analogy to biological genetic mechanisms, generate new candidate rules from parents that have high strengths, where strength or "fitness" reflects some measure of performance on training cases. In effect, genetic methods carry out parallel hill climbing, retaining a set of competing and sometimes complementary descriptions in memory. Goldberg (CACM, March, 1994) reviews genetic approaches to both machine learning and optimization problems.

A fourth paradigm, which we will call *rule induction*, employs condition-action rules, decision trees, or similar knowledge structures. Here the performance element sorts instances down the branches of the decision tree or finds the first rule whose conditions match the instance, typically using an all-or-none match process. Information about classes or predictions are stored in the action sides of the rules or the leaves of the tree. Learning algorithms in the rule-induction framework usually carry out a greedy search through the space of decision trees or rule sets, typically using a statistical evaluation function to select attributes for incorporation into the knowledge structure. Most methods partition the training data recursively into disjoint sets, attempting to summarize each set as a conjunction of logical conditions. Quinlan [20] describes one such rule-induction algorithm in some detail.

A final approach, sometimes termed *analytic* learning, represents knowledge as rules in logical form, and typically employs a performance system that solves multi-step problems using some search process. A common technique is to represent knowledge as Horn clauses (as in the Prolog language), then to phrase problems as "theorems" and to search for proofs. Learning mechanisms in this framework use background knowledge to construct proofs or "explanations" of experience, then compile the proofs into more complex rules that can solve similar problems either with less search (using local "search-control rules") or in a single step (using "macro-operators"). Most work on analytic learning has focused on improving the efficiency of search, but some has dealt with improving accuracy on classification tasks.

The reasons for the distinct identities of these paradigms are more historical than scientific. The different communities had their origins in different traditions, and they rely on different basic metaphors. For instance, proponents of neural networks emphasize analogies to neurobiology, case-based researchers to human memory, students of genetic algorithms to evolution, specialists in rule induction to heuristic search, and backers of analytic methods to reasoning in formal logic. One can question whether this division benefits the field, as differences of notation and rhetoric often obscure important underlying similarities.

However, recent experimental comparisons between different learning methods have helped break down these boundaries, as has the increasing tendency to describe the results of learning in terms of geometric decision boundaries. Many researchers (including the current authors but certainly not everyone) now hold that neural networks are no more "subsymbolic" than logical rules (though they may produce quite different decision boundaries), that analytic methods are not guaranteed to learn from fewer instances than rule-induction methods (though they do in some cases), and that logical rules are not necessarily more easily understood by domain experts than other representations (though they are in some domains). Claims are increasingly backed by careful experimental studies rather than rhetorical statements.

The research literature reveals a number of healthy trends along these lines. Hybrid methods that cross paradigm boundaries are increasingly common. These include algorithms for inducing decision trees that contain linear threshold units and techniques for transforming rules into neural networks and back again. Research on theory revision combines analytic methods' emphasis on background knowledge with rule induction's emphasis on heuristic search. And recent work on inductive logic programming, reviewed by Bratko and Muggleton (this issue), adapts algorithms for rule induction to such logical representations as those used in languages like PROLOG. These convergences are the signs of a balanced and maturing field.

We should note that most research on machine learning, with the exception of work in the analytic paradigm, has focused on simple classification or prediction tasks, and the most robust learning methods are designed for such problems. The restriction to classification is not really very severe, since one can usually decompose a complex process such as design, control, or planning into a sequence of individual steps, each of which involves simple classification or prediction. We will see that many efforts have taken exactly this approach.

In the remainder of this paper, we review some applications of rule induction (among the most mature of the approaches) and, in one case, analytic learning. We focus on these paradigms not because they are more central to machine learning or more robust than the others, but because current surveys of neural networks, case-based learning, and genetic algorithms have recently appeared in this journal (March, 1994). Our goal is to complement those articles, thus providing readers with a more complete view of recent advances in machine learning.[1] Readers can find additional references to the various approaches to machine learning in `http://robotics.stanford.edu/people/langley/mlrefs.ps`.

Fielded Applications of Rule Induction

To clarify the potential for rule induction in real-world problems, in this section we consider some fielded applications of this approach. In each case we describe the problem, its reformulation in terms of machine learning, and the status of the resulting knowledge base. However, this sample far from exhausts the fielded applications, and in closing we mention briefly some other recent uses of the rule-induction approach.

INCREASING YIELD IN CHEMICAL PROCESS CONTROL

Fuel for nuclear power plants is commonly generated by transforming Uranium hexafloride gas into pellets of Uranium dioxide powder. These pellets must be of high quality, but experts cannot predict when a batch of pellets will be good or bad. Researchers at Westinghouse used statistical methods to predict pellet quality with partial success, but interactions among the predictive attributes limited the effectiveness of this approach.

Leech [14] followed a different path in which decision-tree induction played a central role. He collected samples of pellet batches of high and low quality, along with their manufacturing control settings (e.g., pelleting parameters and powder characteristics), some numeric and others symbolic. He ran these training data through a decision-tree algorithm, then transformed the resulting tree into rules that predicted pellet quality. He repeated this process to find rules for predicting qualita-

1. Unfortunately, we do not have space to review the growing literature on learning with probabilistic representations, including trees of probabilistic concepts (e.g., [7]) and Bayesian influence networks (e.g., [3]). But this learning paradigm is still young and, to our knowledge, has yet to produce any fielded applications.

tive powder attributes, which were then used in the top-level rules, giving a structured knowledge base.

After careful evaluation, Leech presented these rules to experienced process engineers, who found them acceptable, and plant technicians began using them to control the pelleting process. As new data became available, he repeated the induction process to produce more accurate rules. The fielded expert system led to increased throughput, higher pellet yield, and reduced inventory, increasing Westinghouse's business (in 1984) by more than ten million dollars per year.

Making Credit Decisions

Loan companies regularly use questionnaires to collect information about people applying for credit, which they then use in deciding whether to make loans. This process has long been partially automated. For example, American Express UK used a statistical decision process based on discriminant analysis to reject applicants falling below a certain threshold and to accept those exceeding another. The remaining 10 to 15 percent of the applicants fell into a "borderline" region and were referred to loan officers for a decision. However, records showed that the loan officers were no more than 50% accurate in predicting whether these borderline applicants would default on their loans.

These observations motivated American Express UK to try methods from machine learning to improve the decision process. Starting with 1014 training cases and 18 descriptive attributes (such as age and years with an employer), Michie [16] and his colleagues used an induction method to produce a decision tree, containing around 20 nodes and ten of the original features, that made correct predictions on 70% of the borderline applicants. In addition to achieving improved accuracy, the company found the rules attractive because they could be used to explain the reasons for decisions to applicants. Although this project was intended as exploratory and took under a week's effort by the development team, American Express UK was so impressed that they put the resulting knowledge base into use without further development.

Diagnosis of Mechanical Devices

Electric motor pumps play an important role in the chemical industry, and preventive maintenance has become a common strategy for reducing interruptions. At Enichem, a chemical branch of a large Italian oil company, diagnosticians regularly check each pump and measure vibrations at various points to determine whether it needs repairs. The machinery includes a motor and a pump, whose shafts are connected by an elastic joint; both motor and pump are anchored to the ground by elastic supports containing bearings. Typical faults include an unbalanced pump, faulty bearings, and distortion of the base. Domain experts at Sogesta rely on Fourier analysis of the vibrations to aid them in their diagnostic decisions.

Giordana, Neri, and Saitta [8] believed that this task would benefit from the use of machine learning. Previously, they had worked with an expert at Enichem to produce an expert system for the diagnosis of motor pumps, representing knowledge in terms of rules, using traditional interviewing techniques to infer the knowledge and coding the information manually to construct the rule base. During this process, the researchers found that the expert measured vibrations at different places on the pump, then used the resulting mathematical analyses in his diagnosis.

After collecting 209 examples of pump measurements and getting the domain expert to label instances as examples of various faults, Giordana et al. ran these data through an induction algorithm to produce a new set of diagnostic rules. Their method used causal knowledge, also gleaned from the expert, to constrain the rule-induction process and increase the likelihood that he would accept the results. Experiments indicated that the learned knowledge base was more accurate than the hand-crafted one, and the induced rules have now replaced the original ones in the diagnostic system. Since their installation, there has been a noticeable reduction in idle times due to improper halting of machines; moreover, the learned rules have greatly aided the less experienced person who replaced the human expert upon his retirement.

Automatic Classification of Celestial Objects

The second Palomar Observatory Sky Survey has produced about three terabytes of image data, containing nearly two billion sky objects. In the past, astronomers have classified and catalogued the objects in photographic plates manually. However, here the aim was to handle stars and nebulae considerably fainter than either visual inspection or existing computer methods could support, and attempts to handcraft expert systems for the task had not produced reliable advances.

In response, Fayyad, Smyth, Weir, and Djorgovski [6]) adapted a machine learning approach to the problem. First they used image-processing techniques to describe each object in a set of images in terms of standard numerical attributes, such as object magnitude, area, ellipticity, and statistical moments of object and core brightness. After astronomers assigned a label to each described object (star, galaxy, etc.), the researchers ran these training data through a decision-tree algorithm that produced a tree for classifying new objects. Initial results were discouraging, yielding low accuracies on novel test objects. However, Fayyad et al. worked with the astronomers to devise additional predictive attributes, defined in terms of the others, which increased the accuracy of the induced knowledge base to 94% – above the level specified by astronomers as necessary for scientific data analysis.

The researchers embedded the resulting classifier in a database management system that supports a variety of uses by astronomers, such as statistical analyses of stellar and galactic distributions. The system is currently being used to classify all objects in the Sky Survey image automatically, a task that would be impractical for humans. The system classifies objects ranging down to some that are one magnitude fainter than any cataloged in large-scale surveys to date, producing a catalog at least three times the size achievable without machine learning.

Monitoring Quality of Rolling Emulsions

The Sendzimir mill, commonly used to roll cold steel, is cooled and lubricated by an emulsion of water and oil, on whose properties the quality of the steel depends critically. For this reason, the Steel Works Jesenice (located in Jesenice, Slovenia) continuously monitors such properties as the oil concentration, the concentration of iron, and the presence of bacteria. Based on these measurements, the factory staff determine the quality of the emulsion and any necessary treatments, such as increasing the magnetic filtering or replacing the emulsion. In complex situations, the staff would consult an expert chemist, but as he was not always available, they sought to manually elicit his expertise through dialogue with him.

When this approach did not succeed, the developers collaborated with local university researchers on an inductive approach, using examples of the expert's decisions as training data [11]. The induced decision tree was installed in the steel works, but later, after a change in the emulsion and its supplier, the knowledge ceased to perform satisfactorily. When attempts at manual adaptation did not work, the developers collected new examples of the expert's decisions and used the same induction method to obtain a revised decision tree. However, they were successful only after formulating a new set of attributes in collaboration with the expert. The resulting knowledge base has been in regular use at the factory since 1989.

Reducing Banding in Rotogravure Printing

Rotogravure printing involves pressing a continuous supply of paper against a chrome-plated, engraved copper cylinder that has been bathed in ink. Unfortunately, grooves or bands sometimes develop on the cylinder during the printing process and appear on the printed pages. The print run must then be halted and in some cases the cylinder replaced, at a substantial cost. The reasons for banding are largely unknown, and experts cannot reliably predict when it will occur.

Evans and Fisher [5] decided that decision-tree induction might be useful in reducing banding, which had become a significant problem at a plant of R. R. Donnelley, a large U.S. printing company. Working with technicians at the plant, they collected positive and negative cases of banding, along with environmental factors (suggested as potentially relevant by the technicians) present in each case. Evans and Fisher ran these data through a rule-induction algorithm, which constructed a decision tree to predict the probability of banding in various classes of situations.

The researchers translated the induced decision tree into a small set of rules, which they posted on one Donnelley plant floor for use by printing teams. Technicians now use these rules to set ink viscosity and other factors under their control, and this new procedure has greatly reduced the frequency of banding effects. For example, banding incidents dropped from 384 in 1990 to 135 in 1991, and went down still further to 66 in the following year, as printing teams came to accept the value of the rules.

Improving Separation of Gas from Oil

When crude oil is extracted from the ground, the oil is usually admixed with natural gas, and before a refinery can begin to process the oil, it must first be separated from the gas. However, one can configure in different ways the size, weight, geometry, and components of the separation vessel. British Petroleum used decision-tree induction to determine the best settings for these parameters as a function of the relative amounts of gas, oil, and water, the pressure, viscosity, and temperature of the mixture, and similar factors.

The complexity of the configuration task led the developers to use an approach called *structured induction*. This scheme incorporates the decisions made by some trees as tests on branches in higher-level trees, but decomposes the learning task by inducing each decision tree separately. Guilfoyle [9] reports that the British Petroleum developers collected 1600 training instances, providing a knowledge base of some 2500 rules organized into 25 sets which the company subsequently translated into 14000 lines of Fortran code. By 1987, the software was in regular use at four different sites, dealing with a task in ten minutes that had previously required human experts over a day.

Preventing Breakdowns in Electrical Transformers

Utility companies often use large, oil-filled electrical transformers to distribute power. However, deteriorating insulation, overheating, joint failure, and other problems can cause very costly breakdowns. Experts can predict failures accurately from gas chromatographs that reveal chemical traces in the transformer oil. To reduce these experts' work loads, Hartford Steam Boiler, an insurer of industrial equipment, funded development of an expert system for this task using rule induction. The resulting system, described by Riese [22], contains 27 sets of rules that check the validity of data, identify the presence of symptoms, infer faults from symptoms, and suggest corrective actions. Experimental evaluation on 859 test cases showed the induced rules agreed with the expert's diagnosis in all but four cases. In 1990, the system was in regular use, automatically producing reports for clients of the insurance company.

Additional Fielded Applications of Rule Induction

The above examples constitute only a fraction of the fielded applications of decision-tree and rule induction, though few results are published in the scientific literature. For instance, Donald Michie [15] has reported four induced knowledge bases for diagnosing faults in circuit boards that are in routine use in a European electronics laboratory and that save millions of dollars a year. Jean Hayes-Michie (personal communication, 1994) has described another expert system, again developed through decision-tree induction, regularly used by Siemens to configure fire-detection equipment for buildings. Gill Mowforth (personal communication, 1993) mentions yet another system, developed partly with decision-tree methods, now used by a South African bank to evaluate applications for credit cards. And David Stirling (personal communication, 1994) has used a similar approach to develop rules for predicting effects in a rolling steel mill, now used by BHP Stainless in Australia.

A few software companies actually specialize in the application of decision-tree and rule induction. For example, David Isherwood (personal communication, 1994) of Attar Software reports a system that provides advice on share trading, currently used by over 20 security dealers in six European countries; a system that predicts which overdue mortgages are likely to be paid, used by the Leeds Permanent Building Society; a fault-diagnosis system for public pay phones that reduces visits by engineers and speeds repairs; a system that predicts the likelihood of retaining good salespeople for an insurance company; and a system that profiles average claims for different medical treatments, used by a health insurance company to monitor excessive claims from both clients and providers.

In a similar vein, Rudolph Sillen (personal communication, 1995) of Novacast describes a support tool for advising administrators on value-added taxes, in use at several Swedish sites since 1992; a thermal analysis system that controls the treatment of iron alloys, used by a Swedish foundry since 1994 and saving $50 per ton by minimizing scrap, increasing yield, and reducing energy and additives; an advisory system for selecting paints for metals and other coating processes, in commercial use in Sweden since 1993; a system for evaluating the capabilities of military units that saves the Swedish Defence Material Administration ten million crowns a year; and a system that predicts whether breast cancer patients will develop new tumors within five years after an operation, used since 1993 by doctors at the Central Hospital in Karlstad, Sweden. Thamir Hassan (personal communication, 1994) at Infolink Decision Services reports that his company has also fielded a number of systems developed through similar methods.

Other Applied Work on Machine Learning

In addition to the fielded applications described in the previous section, we should mention a number of other efforts that have a strong applied flavor. Although these systems are not currently in regular use, the range of tasks covered gives additional evidence of the robustness and flexibility of rule-induction methods.

Automated Completion of Repetitive Forms

Completing forms is a tedious activity that continues to occupy enormous time in both business and government. Even partial automation of the process would produce substantial savings, but the cost of writing a separate expert system for each form often forestalls this approach. Hermens and Schlimmer [10] have developed a form-filling advisory system that learns to predict its users' preferences through observation. They used an incremental version of decision-tree induction to find rules for predicting the default entry for each field in terms of other fields already specified. The user can always override the predicted value and revise the learned rules. Experiments showed that the form-filling apprentice saved up to 87% in keystroke effort and correctly predicted nearly 90% of the entries on the form. The system was used by administrative staff in Hermens and Schlimmer's university department for eight months, until changes in hardware ended the project.

Supporting Maintenance of Knowledge Bases

One of the earliest sets of expert systems (for the automatic design of motors, generators, and transformers in operation at the Westinghouse Corporation in 1956) went out of use after a few years because of the recurring cost of revising them manually to incorporate new design knowledge. As the technology of expert systems has matured, it has become clear that approximately half of their lifetime cost is incurred in maintaining the knowledge base. Regular maintenance is needed not only because of errors introduced at coding time, but also because the problem itself changes over time, as devices and users evolve.

For instance, Langley, Drastal, Rao, and Greiner [13] describe a diagnostic system for computerized tomography scanners that is used on a regular basis by technicians at a Siemens operating company, but in which errors in the knowledge base have started to emerge. Langley et al. considered using existing induction algorithms for theory revision to handle this problem, but the available theory-revision methods were designed for knowledge represented either as Horn clauses or decision trees, whereas the existing diagnostic system uses a fault hierarchy. However, the researchers borrowed a search framework from existing methods, while replacing the learning operators with ones appropriate to fault hierarchies. This method has not yet been tested in the field, but preliminary evaluations with synthetic but realistic data have been encouraging.

Increasing the Speed of a Natural-Language Interface

Natural-language interfaces have become increasingly common, but as their flexibility and coverage grows, the need for efficient parsing algorithms is growing as well. An interface that is slow to respond to improvements in parsers can lose users. Samuelson and Rayner [23] applied an analytic learning method to this problem. They noted that, because the linguistic knowledge in their natural-language system was given in a definite clause grammar, it could be easily transformed into the Horn-clause representation often used in analytic learning techniques.

Their approach compiles a successful parse tree for a sentence into a macro-operator that can handle analogous sentences or phrase structures in a single step. The system also constructs a decision tree to index the resulting rules by the lexical categories of their constituents. Using this approach, Samuelson and Rayner reduced by a factor of three the time taken to parse sentences from a large corpus based on users' actual queries.

Testing Engines for the Space Shuttle

The main engines for the space shuttle require extensive testing before they become operational. Each test firing produces over 100 megabytes of data from pressure, temperature, velocity, strain, and acceleration sensors located throughout the engine. Teams of engineers examine these data to determine whether enough tests have been run and whether the engine's performance meets stringent criteria. They must decide whether another test firing is needed, whether to replace engine components, and so forth.

Because this evaluation process itself is expensive, Rocketdyne used structured-induction methods (similar to those used in the British Petroleum effort) to construct recursively structured decision trees for the task. Modesitt [18] describes one of the resulting systems, designed to handle data from static-fire tests, which contained over 1500 rules organized into 48 rule sets. Another knowledge base, constructed to analyze dynamic data such as frequencies and vibrations, was induced in a similar fashion. Both were embedded in a larger software system for supporting the testing process.

Field tests of the various modules were encouraging, but also suggested extensions to the overall system.

Forecasting Severe Thunderstorms

Although numerical models can predict large-scale weather patterns a day in advance, local forecasting still relies on the expertise of human meteorologists. For example, to determine the chance of severe thunderstorms they use factors like the amount of low-level moisture and the destabilization potential at low and high levels, which they in turn analyze using such data as the dew point, advection variables, and stability indices. Zubrick and Riese [25] describe an expert system for this task developed, using decision-tree induction, by a meteorologist at the National Severe Storms Forecast Center. The system's hierarchical structure supports explanation of its predictions, and in tests during a one-week period in which five severe thunderstorms occurred, it made more accurate predictions than the traditional methods.

Predicting the Structure of Proteins

One largely unsolved problem in molecular biology involves predicting the secondary structure (folding) of proteins from information about their primary amino acid sequences. Some hand-crafted theories exist, but their predictive abilities are disappointing. Muggleton, King, and Sternberg [19] attacked this problem using inductive logic programming, which they felt was appropriate for such a relational domain. Taking 16 proteins that contained only α helices, they treated each position in these proteins as a training instance. They also included background facts about the residues at each position and about the physical and chemical properties of those residues. The initial rules generated by the induction algorithm were moderately accurate but, after adding these rules' predictions as background facts and repeating the induction process, the second rule set produced better results. Another repetition of this strategy gave predictive rules that were 81% accurate on four separate test proteins, considerably higher than other results in the protein literature.

Automation of Scheduling in a Steel Mill

Materials scheduling in steel mills is a complex task that experts divide into three major components: receiving incoming materials into stockpiles; transferring materials from stockpiles to plants for crushing, blending, or blasting; and routing iron ore through screening or crushing plants. For example, depending on the size of ore lumps in a batch, one may crush them, blend them with other material, or send them directly to the blast furnace. Michie [17] describes an effort by Pohang Iron and Steel Company, in South Korea, to construct an expert system for this process using structured decision-tree induction. The applications team interviewed experts to determine potentially relevant attributes for each component task, then ran training data through a decision-tree algorithm to produce a structured knowledge base. The resulting scheduling system, which includes 40 rule sets, performed comparably to domain experts during operational tests.

Learning Strategies for Flight Control

Flying and landing even a small airplane employs complex sensori-motor skills that experts have difficulty communicating to others, but a knowledge-based system for these tasks would be useful both as a pilot aid and in training novices. Sammut, Hurst, Kedizer, and Michie [24] collected traces of expert behavior on a flight simulator, storing the pilot's actions and the associated sensor readings at each time step. They treated the description for each step as a training case, which they passed to a decision-tree algorithm after partitioning the data into distinct tasks, such as taking off, turning, and landing. The resulting rules, which propose the actions for a given task and sensor readings, control the simulated aircraft as accurately as the expert they imitate, and recent studies suggest that adding turbulence to the simulator leads to robust flying behavior across a range of situations.

Additional Applications and Related Approaches

Nor does above list exhaust the examples of machine learning applications. Researchers have explored a broad range of tasks, though diagnosis has been an especially popular problem area. For example, El Attar and Hamery [4]) have applied rule-induction methods to the diagnosis and repair of helicopter blades. The literature abounds with examples of machine induction for medical diagnosis of humans (e.g., [21]), and many of the online data sets fall into this area. Despite repeated demonstrations that the induced knowledge bases can be more accurate than physicians, few of these efforts have led to fielded systems. But the problems do not lie in the use of induction to generate the knowledge base, for doctors have been reluctant to adopt hand-crafted knowledge bases as well.

We have focused here on techniques that come from the machine learning community, but independent developments in statistics have produced similar methods. Breiman, Friedman, Olshen, and Stone [2] describe a set of methods for inducing decision trees, which they tested on a variety of applied problems, such as predicting the survival of recent heart-attack patients. A related line of statistical work, known as *automated interaction detection* [1], has been widely used in the analysis of survey data. Similar techniques are now included in SPSS, a widely available statistical package, making the technology of rule induction accessible to a wide audience.

Some Strategies and Lessons

Efforts to apply rule induction and other machine learning methods follow a standard pattern, but one that has seldom been made explicit in the literature. In this section we attempt to characterize the main stages of the process, while noting some lessons from the examples presented earlier. In closing, we draw some tentative conclusions about the sources of power in successful applications.

Formulating the Problem

The first step in using machine learning to solve any real-world problem is to reformulate the problem in terms that can be handled by some induction method. Process control, diagnosis, and scheduling are complex tasks, yet one can identify components that involve simple classification, a task for which there exist robust induction algorithms. Repeatedly we see developers transforming an apparently difficult problem into a one-step classification task. In the applications we examined, only the work by Langley et al. and Samuelson et al. employed learning methods that dealt directly with more complex performance elements. But neither project has as yet produced a fielded knowledge base, whereas many of the simpler approaches have.

A number of developers have relied on a technique known as *structured induction*, which involves dividing a complex task into subproblems, then providing training data for each one separately. Zubrick and Reese [25], Leech [14], and Modesitt [18] all took this approach, producing performance systems that carry out multi-step inference, but, by factoring it, avoid this complexity during the induction process. Muggleton et al.'s [19] scheme, which added predictions produced by learned rules as background knowledge for later rounds of induction, provides an alternative way of decomposing the learning task.

The best formulation of the problem may not always be the one most intuitive to a machine-learning researcher. In process-control domains, it seems natural to search for rules or trees that directly predict the values of process variables, such as ink viscosity in printing, from environmental ones like humidity. However, on two of the control tasks we examined ([4], [14]), developers instead used induction to find rules to predict directly the effects of both process and environmental variables, apparently because users were more familiar with this formulation. On the other hand, similar work reported by Sammut et al. [24] and Michie [17] took the more 'natural' approach, so no general conclusions can be drawn.

Determining the Representation

The second step in applying machine learning techniques is to settle on an effective representation for both training data and the knowledge to be learned. We are not referring here to the representational formalism, such as decision trees or neural networks, but to the attributes or features used to describe examples and to characterize the result of learning.

Representation engineering – finding an effective representation of the phenomena – was central to most of the projects we examined. In some cases, this involved little more than talking with domain experts and getting their advice on attributes that were likely to have predictive value. In other cases (e.g., Fayyad et al.'s work [6]), it involved a painstaking search of the feature space, looking for descriptors that could provide the discriminating power the more obvious features lacked.

In some cases the "primitive" features may be computed by already established methods. Fayyad et al. relied heavily on established techniques for image processing to transform their digital images into attribute-value descriptions that could be handled by decision trees. Zubrick and Reese

[25] incorporated traditional statistical measures in their work on forecasting thunderstorms, and Giordana et al. [8] used the output of Fourier analysis as primitive attribute values.

Collecting the Training Data

After settling on a task and a representation, one can collect the training data needed for the induction process. In some domains, this process is straightforward and can even be automated, but in others it can pose a significant challenge. In Evans and Fisher's [5] work on banding in rotogravure printing, the researchers asked the printing technicians to record periodically the values of the process variables and the outcome, but the technicians were reluctant to waste time collecting data on a machine that was working well. Only after considerable effort were they persuaded to record values when the machine was working properly as well as when it failed. Most application domains fall somewhere between these two extremes, with some help from the experts being needed to classify training data or to generate them.

The availability of data depends heavily on the instrumentation of the systems that are being studied. In the ideal situation, the expert system can be tied directly into the flow of data from the operating system's instruments. As expert systems become more common, instrumentation for them will increasingly be designed into the machines they are guiding; however, for the foreseeable future, accessing the available data streams and generating data where they have been lacking will be an important part of applied work in machine learning.

Evaluating the Learned Knowledge

Rules induced from training data are not necessarily of high quality. The performance of knowledge acquired in this way is an empirical question that must be answered before that knowledge can be used on a regular basis. One standard approach to evaluation involves dividing the data into two sets, training on the first set, and testing the induced knowledge on the second. One can repeat this process a number of times with different splits, then average the results to estimate the rules' performance on completely new problems. Kibler and Langley [12] experimental methods of this sort for a broad class of learning algorithms.

However, human experts are available in many domains, and it would be foolhardy to ignore their opinions, even when they cannot articulate their knowledge fully. Thus, an important part of the evaluation process is experts' examination of the learned knowledge. If significant problems emerge at this stage, they may suggest revisions to the problem formulation or representation. Evans and Fisher [5] encourage such an iterative process in developing a fielded application, and other work we have seen took similar approaches.

Fielding the Knowledge Base

The final stage in applications is fielding the learned knowledge base. We intend this term in the broadest possible sense. In some cases, the knowledge acquired can be used without even embedding it in a computer system. In Evans and Fisher's [5] work, a simple rule set written on paper was enough for humans to use in making decisions that alleviated their banding problem. In other cases, as in Fayyad et al.'s [6] and Modesitt's [18] domains, users expected not only computer implementation of the learned knowledge, but also considerable software support that had nothing to do with machine learning.

The important consideration is that the learned knowledge be *used*. Graphical interfaces may increase the chances of use in some domains but hurt them in others. Explanation capabilities may be welcomed by some users but not by others. In some cases (such as Giordana et al.'s work [8]), the existence of a fielded hand-crafted expert system has been useful in fielding the learned knowledge base. Users who are already convinced that a knowledge-based system is beneficial are unlikely to object to having an improved knowledge base, although the fact that machine learning generated this knowledge may have had little meaning to them. For this reason, it is easier to introduce machine learning systems as extensions of expert systems that are already in place than to introduce both the expert system and its learning component at the same time.

We have made a number of comments on the role of users and experts both in designing the learning system and in securing its actual use. Everything that has been written and said about the importance of motivating users and domain experts, the need for their participation in the design and application processes, and the need to introduce computer interfaces that are usable and convenient for them applies in spades to the design and application of machine learning to industrial and other real-life situations.

Sources of Power in Applied Machine Learning

In this paper we examined a number of applications of rule induction, some in regular use and others moving toward that goal. Most of these application efforts have used well-understood, established induction algorithms that operate on supervised, attribute-value data, and do not employ the more sophisticated techniques that dominate the research literature. Developers need not be ashamed of this fact; it is quite appropriate that applications draw on methods that have proved their power, reliability, and versatility in other applications or in laboratory tests, and if simple methods are available that have these properties, so much the better.

In fact, close inspection of these projects suggests that much of the power comes not from the specific induction method, but from proper formulation of the problems and from crafting the representation to make learning tractable. In these cases, machine learning has not completely automated the knowledge engineering process, but it *has* replaced knowledge engineering with two simpler tasks: characterizing the problem and and designing a good representation. Developers need not play down this fact; reducing the time and effort needed to develop knowledge-based systems, however short this may fall of complete automation, can produce systems of great practical value, as we have seen.

Although we have concentrated on rule-induction methods, one might question – given our comments about sources of power – whether equivalent results would not emerge if one replaced the rule-induction algorithms with neural network, genetic, or case-based learning techniques. Recent comparatives studies in the literature, which show roughly equivalent performance across many domains, are consistent with this prediction. Consequently, given equivalent tools, each person may well want to use the ones with which they are most comfortable and familiar.

It is probably not an accident that quite different procedures produce similar results in application. Similar phenomena have been noticed in applying diverse management science tools to problems like scheduling. Where this occurs, it may result from the nature of the problem space. If global optima are easy to find or if local optima are nearly as good as the global one, then many methods may produce comparable performance. Engineers, accustomed to working in complex situations that do not admit analytic solutions, have long been aware of these facts. Rivers can be spanned with suspension bridges, trusses, cantilevers, and other radically different designs, and often there is no conclusive reason for choosing one over another.

Machine learning may never entirely replace knowledge engineering as a framework for constructing knowledge-based systems, but our examples show that significant progress toward automation has already been made, and we anticipate that rule induction and other learning methods will become increasingly prevalent as their benefits become better known.

Acknowledgements

We thank Peter Clark, Donald Michie, and Steve Muggleton for pointing us toward many of the application efforts we have reported; Donald Michie deserves a large share of credit for fostering many of these projects and advising their developers. Comments from Ivan Bratko, Donald Michie, and two anonymous reviews improved an earlier draft of this article. This work was funded in part by grants to the first author from the Office of Naval Research (N00014-94-1-0505 and N00014-94-1-0746) and the Advanced Research Projects Agency, through the Institute for the Study of Learning and Expertise.

References

[1] Biggs, D., de Ville, B., & Suen, E. (1991). A method of choosing multiway partitions for classification and decision trees. *Journal of Applied Statistics*, *18*, 49–62.

[2] Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees.* Belmont: Wadsworth.

[3] Cooper, G. F., & Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, *9*, 309–347.

[4] El Attar, M., & Hamery, X. (1994). Industrial expert system acquired by machine learning. *Applied Artificial Intelligence*, *8*, 497–542.

[5] Evans, B., & Fisher, D. (1994). Overcoming process delays with decision-tree induction. *IEEE Expert*, *9*, 60–66.

[6] Fayyad, U. M., Smyth, P., Weir, N., & Djorgovski, S. (1995). Automated analysis and exploration of image databases: Results, progress, and challenges. *Journal of Intelligent Information Systems*, *4*, 1–19.

[7] Gennari, J. H., Langley, P., & Fisher, D. H. (1989). Models of incremental concept formation. *Artificial Intelligence*, *40*, 11–61.

[8] Giordana, A., Neri, F., & Saitta, L. (in press). Automated learning for industrial diagnosis. In P. Langley & Y. Kodratoff (Eds.), *Fielded applications of machine learning*. San Francisco: Morgan Kaufmann.

[9] Guilfoyle, C. (1986). Ten minutes to lay the foundations. *Expert Systems User*, August, 16–19.

[10] Hermens, L. A., & Schlimmer, J. C. (1994). A machine-learning apprentice for the completion of repetitive forms. *IEEE Expert*, *9*, 28–33.

[11] Karba, N., & Drole, R. (1989). Expert system for the cold rolling mill of the Steel Works Jesenice. *Proceedings of the Thirteenth Symposium on Information Technologies*. Sarajevo, Yugoslavia.

[12] Kibler, D., & Langley, P. (1988). Machine learning as an experimental science. *Proceedings of the Third European Working Session on Learning* (pp. 81–92). Glasgow: Pittman.

[13] Langley, P., Drastal, G., Rao, B., & Greiner, R. (1994). Theory revision in fault hierarchies. *Proceedings of the Fifth International Workshop on Principals of Diagnosis*. New Paltz, NY.

[14] Leech, W. J. (1986). A rule-based process control method with feedback. *Advances in Instrumentation*, *41*, 169–175.

[15] Michie, D. (1987). Current developments in expert systems. In J. R. Quinlan (Ed.), *Applications of expert systems*. Wokingham, UK: Addison-Wesley.

[16] Michie, D. (1989). Problems of computer-aided concept formation. In J. R. Quinlan (Ed.), *Applications of expert systems* (Vol. 2). Wokingham, UK: Addison-Wesley.

[17] Michie, D. (1992). Directions in machine intelligence. *Computer Bulletin*, September/October, 9–11.

[18] Modesitt, K. L. (1990). Inductive knowledge acquisition: A case study of Scotty. In K. L. McGraw & C. R. Westphal (Eds.), *Readings in knowledge acquisition: Current practices and trends*. Chichester, UK: Ellis Horwood.

[19] Muggleton, S., King, R. D., & Sternberg, M. J. E. (1992). Protein secondary structure prediction using logic-based machine learning. *Protein Engineering*, *5*, 647–657.

[20] Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. San Francisco: Morgan Kaufmann.

[21] Quinlan, J. R., Compton, P. J., Horn, K. A., & Lazarus, L. (1987). Inductive knowledge acquisition: A case study. In J. R. Quinlan (Ed.), *Applications of expert systems*. Wokingham, UK: Addison-Wesley.

[22] Riese, C. (1984). *Transformer fault detection and diagnosis using RuleMaster by Radian* (Technical Report). Austin, TX: Radian Corporation.

[23] Samuelson, C., & Rayner, M. (1991). Quantitative evaluation of explanation-based learning as an optimization tool for a large-scale natural language system. *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence* (pp. 609–615). Sydney: Morgan Kaufmann.

[24] Sammut, C., Hurst, S., Kedizer, D., & Michie, D. (1992). Learning to fly. *Proceedings of the Ninth International Conference on Machine Learning* (pp. 385–393). Aberdeen, Scotland: Morgan Kaufmann.

[25] Zubrick, S. M., & Riese, C. E. (1985) An expert system to aid in severe thunderstorm forecasting. *Proceedings of the Fourteenth Conference on Severe Local Storms*. Indianapolis.