University of Wisconsin Computer Sciences Technical Report 859 (June 1989)

Combining Explanation-Based and Neural Learning: An Algorithm and Empirical Results

Jude W. Shavlik

Geoffrey G. Towell

Computer Sciences Department
University of Wisconsin
1210 West Dayton Street
Madison, WI 53706

shavlik@cs.wisc.edu
(608) 262-7784

Keywords: explanation-based learning, neural networks, connectionism, symbolic systems, imperfect domain theories, neural network topologies, hybrid machine learning systems

Abstract

Machine learning is an area where both symbolic and neural approaches have been heavily investigated. However, there has been little research into the synergies achievable by combining these two learning paradigms. A hybrid approach that combines the symbolically-oriented explanation-based learning paradigm with the neural back-propagation algorithm is described. Most realistic problems can never be formalized exactly. However, there is much to be gained by utilizing the capability to reason nearly correctly. In the presented EBL-ANN algorithm, a “roughly-correct” explanatory capability leads to the acquisition of a classification rule that is almost correct. The rule is mapped into a neural network, where subsequent refinement improves it. This approach overcomes problems that arise when using imperfect domain theories to build explanations and addresses the problem of choosing a good initial neural network configuration. Empirical results show that the hybrid system more accurately learns concepts than an explanation-based system by itself and that the hybrid also learns them much faster than a neural learning system by itself.

Combining Explanation-Based and Neural Learning: An Algorithm and Empirical Results

1. INTRODUCTION

Explanation-based learning (EBL) and artificial neural networks (ANNs) are two actively pursued approaches to machine learning. These two styles approach the learning problem from very different viewpoints. EBL falls within the traditional symbolically-oriented approach to artificial intelligence. It has shown promise in high-level tasks such as planning [Minton88], natural language processing [Mooney89a], and diagnosis [Hammond88]. Neural (or connectionist) networks are largely numerically-oriented. Large numbers of processing units, each with minimal capability, operate without global control and together produce intelligent behavior. ANNs have proven successful on low-level tasks, such as perception [McClelland81] and signal processing [Ahalt89], as well as on classification and diagnostic tasks [Mooney89b]. Each approach has its strengths and weaknesses. An approach that combines the two styles, building on each's strengths and overcoming each's weaknesses, is described and a successful implementation reported. The approach is not intended to be plausible at the level of neurophysiology. Rather, it investigates whether symbolic and neural learning techniques can be profitably combined to produce more powerful, general-purpose machine learning algorithms.

Explanation-based learning is a recently developed and actively pursued approach to machine learning [DeJong86, Mitchell86]. In this type of learning, the solution to a sample problem is generalized into a form that can be later used to solve conceptually similar problems. The generalization process is driven by the explanation of why the solution worked. Knowledge about the domain allows the explanation to be developed, and then generalized, thereby producing a general concept from the solution to a specific problem. EBL systems can be used to produce general classification rules by analyzing the classification of a specific object (e.g., a rule for recognizing cups [Mitchell86]) or to produce general plans by observing the achievement of a goal in a specific situation (e.g., a plan for building towers [Shavlik90]).

One advantage of EBL systems is that they only require a small number of examples to learn a concept, unlike empirically-based learning methods (e.g., [Michalski83, Mitchell82, Quinlan86, Rumelhart86a]) which require large numbers of examples. Only a few examples are needed because EBL systems possess the ability to explain what is relevant in these examples. Other approaches, including connectionist ones, rely on statistics to determine relevance and hence need many examples. They may be mislead by spurious correlations unless someone initially selects a small set of relevant features and uses them to describe the training examples.

The strength of EBL arises largely from it use of prior knowledge to guide its learning. However, EBL's dependence on prior knowledge is the major source of EBL's weaknesses. The basic algorithms (e.g., [Hirsh87, Kedar-Cabelli87, Mooney86, Shavlik89a]) assume a complete and correct "domain theory" [Mitchell86, Rajamoney87]. This domain theory, which is usually a collection of inference rules describing a given domain, is used to build the explanations that are then generalized. If the domain theory is not complete, some examples cannot be explained and, hence, nothing is learned. Incorrect domain theories lead to incorrectly learned results. A final problem is that, even if correct and complete, a domain theory may not be usable because its complexity makes it intractable.

In any real-world domain, a computer system's model can only approximate reality. Needed inference rules may be missing, while many of those possessed will lead to incorrect results in some situations. Furthermore, the complexity of problem solving prohibits any semblance of completeness and tractability. Finally, some domains are inherently uncertain. The actions of other agents, both competitors and cooperators, cannot be fully determined, nor can the physical world be exactly predicted. In these circumstances a domain theory is necessarily approximate [Bennett88]. To be truly useful, EBL systems must be able to work in uncertain, intractable, incompletely-specified, and changing environments, something they currently are incapable of doing.

Artificial neural networks have also generated much interest recently. ANNs consist of a collection of simple, neuron-like processing units. These units are connected by weighted links, where the weights indicate the degree of influence one unit has on another. The activation of a unit is determined by a weighted function of the activations of all connecting units. Computation proceeds by activating *input* units and waiting until the activation of the *output* units stabilizes. During this process, activation may traverse through intermediary, or *hidden*, units.

One area in which ANNs have appeal is that of inductively acquiring concepts from examples. A set of examples, each labelled as belonging to a particular class, is provided to the ANN. Its task is to determine a procedure for classifying future examples. Impressive performance has been achieved on problems such as converting text to speech [Sejnowski87], evaluating moves in backgammon [Tesauro89], and diagnosing diseases [Mooney89b].

Several algorithms have been developed for training an ANN to classify new examples (e.g., [Hinton86, Rosenblatt62, Rumelhart86b]). Training is accomplished by analyzing a set of provided, pre-classified examples to determine how to appropriately modify the connection weights so that these training examples are more accurately classified.

While the performance of ANNs that adjust their connection weights has been impressive, there are some problems with this form of machine learning:

(1) *Training times are lengthy.* Recent experiments [Fisher89, Mooney89b, Shavlik89b, Weiss89] indicate that while ANNs classify new examples slightly better than do symbolic learning algorithms (such as ID3 [Quinlan86]), they require 100 to 1000 times as much training time.

(2) *The initial weights can greatly effect how well the concept is learned.* Because training ANNs usually involves a hill-climbing algorithm, there is the problem of local minima. Occasionally the learning algorithms cannot find any way to adjust weights to locally improve performance. Classification correctness on new examples can depend significantly on the initial set of weights of the ANN. For example, in some experiments [Shavlik89b] involving the conversion of text to speech, correctness on a large dictionary of examples ranged from 1% to 31%. The only difference between the runs was the randomly-chosen initial set of weights. Others have also reported that performance can be highly dependent on the initial set of weights [Ahmad88]. One way to overcome this problem is to train several networks, keeping the one that performs best on the training examples. For example, in [Ahmad88] usually 20 runs are performed. However, this substantially increases the training time.

(3) *There is no known problem-independent way to choose a good network topology.* The performance of an ANN can be greatly effected by the layout and number of hidden units, as well as the specification of which units are connected to which other units. Again, several runs with different topologies can be performed, keeping the best.

(4) *Many training examples are needed.* As is standard in learning systems that inductively acquire concepts from examples, ANNs, unlike EBL systems, need many training examples to acquire a concept [Baum89].

(5) *There is no way to focus attention on significant features.* ANNs have traditionally been trained in environments where the set of features to be considered is carefully selected by the experimenter. Without this pre-selection, the ANN would be forced to consider thousands of features normally dismissed as irrelevant by the human user of the system.

(6) *The learned network is difficult to interpret.* While the meanings of the ANN's input and output units are easily understood because they are fixed by the environment, the meaning of individual hidden units frequently cannot be determined. Hence, it is usually hard to ascertain reasons for the answers provided by the ANN.

The next section describes the hybrid approach that overcomes the weaknesses of both EBL and ANNs. Following that, the algorithm is presented in detail and applied to two sample problems. Related research and some open issues are then described. Common Lisp code that implements the algorithm appears in the appendix.

2. AN INTEGRATED APPROACH

EBL systems can take advantage of existing knowledge and, hence, need few examples to learn a new concept. However, they require a well-formed base with which to build explanations. Incorrect concepts may be learned if a faulty domain theory is used. Conversely, ANNs provide a way to incrementally adjust the representation of a concept on the basis of additional examples. Their weakness is that they require a large number of examples and many training epochs to converge on an acceptable concept, especially if they start from a randomly chosen initial state.

An approach that integrates these two styles of learning may accrue the benefits of each while avoiding their shortcomings. The hybrid EBL-ANN system described below has been developed with this goal in mind. Basically, the EBL system produces categorization rules rapidly from a small number of examples. These rules are then refined by an ANN, using some additional examples. In this hybrid system, a complete and correct domain theory is not necessary, as the ANN refines learned concepts. Also, fewer training examples are needed than in a pure ANN, as the EBL portion of the system produces an initial network that is close to the correct concept.

In the hybrid EBL-ANN approach, an approximate domain theory may be used. With this domain theory, the EBL portion of the system can build only roughly-correct explanations. These explanations need not be proofs, but only reasonable arguments as to why the current item belongs to a specific category or why the current plan meets the given goal. That is, the domain theories can be incorrect. For domain theories that are intractable or incomplete, simplifying or default assumptions can be made in order to produce these roughly-correct explanations (Chien, 1989; Mitchell *et al.*, 1986).

Many EBL systems have domain theories of the desired form already, in particular those involving natural language (e.g, (Mooney & DeJong, 1985)). However, they treat their domain theories as being as logically correct as theories representing mathematical calculations (e.g., (Shavlik, 1985)). The hard part is not writing roughly-correct domain theories. The hard part is developing a learning system that properly treats the domain theory as roughly correct.

In the hybrid EBL-ANN system, the EBL portion of the system produces a rough explanation and then generalizes it using a standard explanation-based generalization algorithm. The final, generalized result is then converted into a neural network. The antecedents of each classification rule determine the weights running from the input units to a hidden unit, while the consequents determine the weights from a hidden unit to the output unit. Features that are believed to be important, according to the result produced by the EBL portion of the system, are highly weighted. Features that the rule says should be absent receive large negative weights. Finally, features that appear to be irrelevant receive weights near zero. These apparently irrelevant features must appear in the ANN because the roughly-correct EBL system may have mistakenly thought they were unimportant.

Once the ANN is produced, additional training examples are provided to refine the network, using standard neural network learning techniques. The final result can be some function of the inputs that is not expressible using the symbols and vocabulary in the domain theory. Hence, the hybrid system can increase the expressiveness of the basic EBL system, thereby increasing accuracy. Also, since the approach is naturally incremental, if the concept being learned is slowly varying over time, the ANN can capture these changes if it periodically receives feedback. Using only an EBL system would require rewriting portions of the domain theory whenever a concept changed. Finally, ANNs can store exceptions along with general concepts (McClelland & Rumelhart, 1986), another way to correct an EBL domain theory.

The major assumption underlying the EBL-ANN approach is that the space of possible concepts is reasonably “smooth.” When this assumption holds, it is likely that starting at a nearly-correct point in the space will facilitate reaching the correct concept. The validity of this assumption in real-world domains must be tested empirically. The assumption about the smoothness of concept space is also made by *case-based reasoning* researchers (Koton, 1989). In this paradigm, previous solutions are slightly modified (“tweaked”) so that they apply to the current problem (Schank & Leake, 1989).

Ahmad (1988) reports on an experiment that supports the hypothesis underlying the EBL-ANN approach. In this experiment, a neural network is configured with the “correct” set of weights. Next, the weights are perturbed. After some retraining, the network performs substantially better than if it had started off with random weights, even when the perturbations are large. This indicates the value of starting off with a set of weights that is close to a correct solution. Sejnowski & Rosenberg (1987) report similar experimental results.

3. THE EBL-ANN ALGORITHM

Table I lists the hybrid EBL-ANN learning algorithm. A roughly-correct domain theory involving binary features[1] is assumed, as are a collection of training examples and criteria for termination of learning.

****** Insert Table I near here ******

In the hybrid algorithm, the EBL portion first uses the domain theory to classify one of the positive examples.[2] An explanation can be viewed as a deductive proof that the training example is a member of the concept being learned. The resulting explanation is then generalized using the EGGS algorithm (Mooney & Bennett, 1986). Currently, the algorithm uses the explanation generated for the first example encountered which can be explained. (An open research issue is how to best integrate the explanations of several examples or several different explanations of the same example into a single ANN.) The EGGS algorithm assumes that, in the course of solving a problem, a collection of pieces of general knowledge (e.g., inference rules, rewrite rules, or plan schemata) are interconnected, using unification to insure compatibility. The resulting explanation is generalized by determining the most general unifier that allows the general pieces of knowledge to be connected in the same way. This involves replacing the constants in the specific explanation with constrained variables. The result is a new composite knowledge structure that contains the unifications that must hold in order for the knowledge pieces to be combined in the given way.

To visualize the EBL process, assume a tree-structured explanation such as the one in Figure 1. In this figure, all conclusions are conjunctive in that all antecedents must be satisfied in order for a node[3] to be true. If the leaf nodes can be consistently satisfied, the root node, which concludes the example is a member of the concept, will also be satisfied. Internal nodes can be ignored. There is no need to again reason about combining the pieces of knowledge together to make the classification. Since a substantial amount of work can be expended constructing the original solution, the new knowledge structure can lead more rapidly to future solutions. A generalized version of the solution is saved and can be efficiently applied to future

---

[1] Being binary is not a restriction, in that multiple-valued features can easily be converted into binary features. For example, assume *color* is a feature that can take on the values *red, blue,* or *yellow*. This can be converted to the following binary features: *color-is-red; color-is-blue; color-is-yellow*. Note, though, that this conversion leads to some loss of information; namely, that *color* can take on only one value. In the binary representation, *color-is-red* and *color-is-blue* are no more related than are *color-is-red* and *size-is-small*.

[2] If the domain theory can also explain why something is *not* a member of the concept being learned, two separate ANNs can be produced. Alternately, a single ANN with two output nodes could be used. However, the integration of multiple explanations into a single ANN is an open research issue.

problems. Also notice that the explanation selects which features are relevant to classification.

****** Insert Figure 1 near here. ******

Notice, though, that in this standard style of EBL, no *knowledge level learning* (Dietterich, 1986) occurs. That is, only concepts in the deductive closure of a system's existing knowledge can be learned. Basic EBL systems speed up problem solvers but do not change what a system could solve if given enough time. The ANN portion of the hybrid system addresses the issue of knowledge level learning by refining the rules encapsulated in the ANN.

Since dealing with variables in neural networks is an open research area, the EBL-ANN algorithm currently assumes that domain theories are purely propositional (i.e., they do not contain any variables). This means that finding the most general unifier is trivial. In this case, the primary role of the EBL system is to focus attention on the features most likely relevant to classifying an example. The relevant features are those appearing in the leaf nodes of the explanation.

Following the application of the EGGS algorithm, the generalized explanation is converted into an ANN, as explained below. From this point, training continues following a standard neural network training procedure called *back-propagation* (Rumelhart, Hinton & Williams, 1986). Back-propagation adjusts inter-unit weights to reduce the difference between the ANN's actual and desired output. Training by repeated presentation of the set of pre-classified examples in random order[4] continues until enough training examples are properly classified or until a training limit is reached. An example is considered correctly classified if the ANN's output is within 0.25 of the correct output. (The setting of this parameter is discussed further later.)

Translation of the classification rule learned by the EBL component into an ANN network is guided by the structure of the explanation. The core layout of the ANN is isomorphic to the explanation structure; Table II lists the correspondences. Each conclusion, final or intermediate, in the explanation structure has a corresponding unit in the ANN. The final conclusion becomes the output unit for the network, while intermediate conclusions are the hidden units. Similarly, features used as inputs to the explanation correspond to units in the input layer of the network. Highly weighted links are added to connect antecedents units which the explanation structure

[3] For clarity, *node* will always be used to refer to components of explanations while *unit* will always be used to refer to processing elements in neural networks.

[4] A single exposure to each of the training examples constitutes a *training epoch*.

indicates are necessary to their consequent unit. Conversely, links with large negative weights are added to the network to connect *prohibitory* antecedent units (i.e., antecedents which must be false) to their consequent unit. One important question is what values the weights and biases on the ANN's nodes should initially have. These are set initially such that the network will only respond when the generalized explanation would also do so. The bias[5] on each unit is set so that the unit will only be significantly active when: 1) all the units corresponding to antecedents that are necessary are active; 2) none of the units corresponding to antecedents that are prohibitory are active.

****** Insert Table II near here. ******

To understand how weight and bias setting works, consider a unit with $n+m$ links, n from necessary antecedents, and m from prohibitory antecedents. The weights on the n links from the necessary antecedents are set to ω while the weights on the m links from the prohibitory antecedents are set to $-\omega$. The bias for the unit is set to $n\omega - \frac{\omega}{2}$. When the bias is set to this level, all of the necessary antecedents must be present and none of the prohibitory antecedents can be present for the unit to have a significant activation. To prevent symmetries that are hard to break (Rumelhart, Hinton & Williams, 1986), all weights and biases are adjusted by a random amount between $-\varepsilon$ and $+\varepsilon$. Currently, $\omega = 5$ and $\varepsilon = 0.1$ are used.

Recall that the generalized explanation may be incorrect. The neural learning algorithm will refine the learned concept, but it must have the nodes and links that allow it to do so. At a minimum, this means additional input units must be added and connected to the network for every possible binary feature not mentioned in the explanation. Additional connections are made by first grouping the units according to the maximum length path from some input unit to each unit. New links, with a randomly-chosen weight within ε of zero, are made between units in successive groups. New connections are not made if there already is a link between a pair units or a loop would be formed. The addition of these low-weight connections allows the insertion into the ANN network of features which were thought to be irrelevant by the roughly-

[5] The activation of a unit i is defined to be $\dfrac{1}{1 + e^{-(\sum_j w_{ij} \times activation_j \; - \; \theta_i)}}$, where j ranges over the units that connect to unit i. θ_i is often called the *bias* of unit i. Roughly speaking, unless the sum of weighted activations on connected units exceeds this bias, unit i is not active. That is, the bias indicates how much weighted activation a unit needs to receive from its neighbors before it "fires."

correct domain theory.

After the ANN is initialized in a nearly correct configuration, training using back-propagation and the complete set of pre-classified examples completes the learning process.

A Sample Translation

Consider the artificial domain theory presented in Table III which defines when an object is a member of category A. Figure 2a represents the only explanation possible given the rules in Table III. The solid and dotted lines in Figure 2a respectively represent necessary and prohibitory dependencies.

****** Insert Table III near here. ******

Figure 2b represents the network which results from the translation into a neural network of the explanation in Figure 2a. The thick lines in Figure 2b represent the links in the neural network which correspond to dependencies in the explanation. Thus, with ω=5, the weight on the thick solid lines is 5 while the weight on the thick dotted line is -5. The lighter solid and dotted lines represent the links added to the network to allow refinement of the domain theory; all these lines have a randomly-chosen weight within ε of zero. The number in parentheses next to each hidden and output unit is the unperturbed bias of that unit (i.e., its bias before adding a random number within ε of zero).

****** Insert Figure 2 near here. ******

4. TWO EXPERIMENTS

The implementation of the EBL-ANN hybrid algorithm has been tested using two simple domain theories. The first involves recognizing cups, and the second involves recognizing chairs. Using the same parameter settings, the hybrid system EBL-ANN outperforms the EBL-ALONE (i.e., stand-alone EBL) and ANN-ALONE (i.e., stand-alone back-propagation) systems in both domains.

Learning to Recognize Cups

The first experimental domain is an adaptation of the traditional cup recognition problem (Mitchell *et al.*, 1986). Table IV lists a set of Prolog rules which define a roughly-correct theory for the recognition of cups. The first rule can be read *"X is a cup if it is stable, liftable, and an open vessel."* Using these rules something analogous to a coffee mug can be recognized as a cup. However, cups without handles (e.g. Styrofoam cups) cannot be recognized. Also, a pail, where drinking is prevented by the handle across the mouth of the cavity (DeJong & Mooney, 1986) is falsely classified as a cup.

****** Insert Table IV near here. ******

The hybrid algorithm is run using the training set of five cups and five non-cups listed in Table V. Notice that Styrofoam cups, which cannot be explained by the domain theory, are included (Cups *c2* and *c3*), as are pails (Non-cup *n2*). Also notice that irrelevant features (e.g., *expensive* and *fragile*) are included.

Figure 3 depicts the ANN as initially constructed from the explanation of the first positive training example, Cup *c1*.[6] For clarity, units are labeled by the name of the concept they represent. While only the links and units mentioned in the explanation are shown, but recall that additional low-weighted links are also in the ANN. Figure 4 presents EBL-ANN's performance on the ten training examples averaged over 50 trials, where each trial is initialized using a different random perturbation of the weights.

****** Insert Figure 3 near here. ******

****** Insert Figure 4 near here. ******

For comparison, the performances of EBL-ALONE and ANN-ALONE also averaged over 50 trials, are included. As EBL-ALONE can use only the rules appearing in Table 4, and those rules will correctly classify seven of the ten examples, it can never do better or worse than 70% correct. Thus, in Figure 3 the line representing the performance of EBL-ALONE is horizontal at 70% correct. For ANN-ALONE, the conventions used in (Mooney *et al.*, 1989) are followed: there is one-layer of hidden units; the number of hidden units is set at 10% of the sum of the number of input and output units (with a minimum of two hidden units); each input unit is connected to each hidden unit; each hidden unit is connected to the output unit; there are no direct connections between the input and output units. Having one layer of hidden units and full connectivity between layers is a common topology in neural network experiments.

The performance of the hybrid algorithm is initially the same as EBL-ALONE because the procedure which translates explanations into networks generates networks which emulate the EBL rules. After a brief period of floundering, the hybrid quickly achieves 100% correctness on

[6] As only one explanation is possible given the cup domain theory of Table IV, the network of Figure 3 could also have been based upon explanations of Cups *c4* or *c5*.

the training examples. Averaged over the 50 trials, the hybrid system takes 16 epochs and has a standard deviation of 2.6 epochs. ANN-alone takes an average of 37 epochs and has a standard deviation of 4.8 epochs.

Figure 4 indicates that the hybrid algorithm achieves the goals which motivated its creation. First, it is able to correctly classify the training set using one-half the number of training epochs required by stand-alone back-propagation. Second, 100% correctness is achieved, even though an imperfect domain theory is used.

Furthermore, the results listed in Table VI indicate that EBL-ANN is better able to generalize its learning than either system alone. The numbers in this table represent the average performance after the 50 learning trials used in Figure 4. For each trial, a different collection of 50 positive and 50 negative cups was used to test each system's ability to generalize to novel examples. The *test set* collections were generated by randomly choosing values for the binary features to form an object, then testing the created object with an oracle domain theory[7] to determine if it should be considered a cup or not. (This procedure allows duplicates and members of the training set in the testing set. However, the space is large enough that this is not a significant issue.) On these test set collections of cups and non-cups, EBL-ANN performed better than either ANN-ALONE or EBL-ALONE. An analysis of variance[8] on these results suggests that the difference in correctness between EBL-ANN and ANN-ALONE is significant with 99.9% confidence, as is the difference in correctness between EBL-ANN and EBL-ALONE .

****** Insert Table VI near here. ******

The results for EBL-ALONE in Table VI also show that EBL-ANN has to both generalize and specialize the initial cup domain theory. EBL-ANN generalized the domain theory at a small cost of incorrectness on the non-cups, with the result that cups and non-cups were correctly classified at about the same rate. By contrast, ANN-ALONE learned an overly general definition of cups; it was correct on 99% of the cups but only 68% of the non-cups.

---

[7] The oracle for cups expands the definition contained in Table IV to include cups without handles so long as they are made of Styrofoam. Also, the obstacle classifies objects with handles on the top as non-cups.

[8] Analysis of variance is a standard statistical test. It compares the variance within a group to the variance between groups, in order to determine if the population means of two or more groups are different. The results of an analysis of variance are expressed in terms of two quantities, F and p. F is the ratio of the variations explained by the grouping to variations left unexplained by the grouping. So, large values of F indicate that the groups are likely to be meaningful, while p is based upon the F statistic, the number of groups, and the total number of observations. It is the probability that the population means of the groups are the same. Therefore, $p<.001$ means that it is possible to conclude with 99.9% certainty that the population means of the groups being considered are different.

Figure 5 presents the average of 50 ANNs following learning by the EBL-ANN system. Although it is not explicit in the figure, the weight on *made of styrofoam* is such that either *made of styrofoam* or *has handle* is sufficient to activate *graspable*. Strongly negative links connect *handle on top* to both *stable* and *open vessel*. Thus, the presence of *handle on top* precludes activation of either of these units. Also, a strongly negative link connects *made of paper* to *graspable*. Its presence is due to spurious correlations in the training data and is one of the principle reasons that EBL-ANN correctly classified only 88% of the randomly generated cups. The thin solid and dotted lines respectively represent negative and positively weighted links with a weight which is significantly different from zero but not sufficient to play a major role in the calculations of the network. The exact weights and biases, averaged over 50 trials, are reported in the appendix.

The result of neural network learning in EBL-ANN is that both false negatives and false positives in the training set, with respect to the original domain theory, are now correctly classified. This correction resulted from the network learning the importance of two features, *made of styrofoam* and *handle on top*, which had previously been ignored. The network also regarded the feature *made of paper* as significant due to correlations in the training data. A method to eliminate the class of problems exemplified by the connection from *made of paper* to *graspable*, such as using using EBL failure to direct back propagation, is an open research issue (see Section 6).

The hybrid system needed about half as many training epochs to learn the training set. Speedup achieved by using the hybrid heavily depends on the setting of the parameter that determines how close the output must be in order to be considered correct. If this parameter is set too small (e.g., 0.05), many epochs are needed to perform the final refinements. This reduces the relative advantage of starting with a roughly-correct ANN. The trade-off is that if this parameter is set too large (e.g., 0.5), the final ANN is less likely to appropriately correct the domain theory. Spurious changes to inter-unit connections can do a good enough job to properly classify the training examples, thereby prematurely terminating training. (With more training examples, a higher setting may be appropriate. A good setting for this parameter is an open research issue.)

While this study, as well as the next one involving chairs, indicates the promise of the EBL-ANN approach, it should be remembered that these are two rather artificial domains with very small training sets. Before firm conclusions are drawn, similar experiments must be run on "real-world" domains, using large training and test sets containing actual examples.

Learning to Recognize Chairs

The second example, which replicates the methodology of the first example, involves learning how to recognize chairs. A roughly-correct set of rules appear in Table VII. Among other things, this theory requires that chairs have four legs, a padded seat, and back support.

****** Insert Table VII near here. ******

The EBL-ANN algorithm is run using the training set of five chairs and five non-chairs listed in Table VIII. Only the first chair is recognized by the domain theory, so the explanation of Chair *c1* is always used to establish the network for the EBL-ANN system. Chairs *c2*, *c3* and *c4* are not recognized as chairs because they are three-legged while Chairs *c2*, *c4* and *c5* lack back support. Conversely, Non-chairs *n2* and *n5*, which cannot support weight, are classified as chairs by this domain theory. Hence, the domain theory properly classifies only four of the sample chairs.

****** Insert Table VIII near here. ******

Figure 6 contains the ANN as initially constructed from the explanation of the first training example. As before, units are labeled by the name of the concept they represent, and only the links and units mentioned in the explanation are shown. The performance of the three systems appears in Figure 7. The curves are the averages over 50 trials, where each trial involves different random perturbations of the initial weights. The average number of epochs needed by the EBL-ANN hybrid is 20 with a standard deviation of 5.7, while for ANN-ALONE it is 34 with a standard deviation of 4.4. As happened when learning to recognize cups, the hybrid system is more accurate than EBL-ALONE and learns faster than ANN-ALONE.

****** Insert Figure 6 near here. ******

****** Insert Figure 7 near here. *****

Testing the classification accuracy of the three systems on novel examples was performed.[9] The results listed in Table IX show that EBL-ANN achieves a better representation of chairs than either ANN alone or the initial domain theory, outperforming both on sets of randomly chosen

---

[9] The oracle for chairs constrains Table VII's domain theory by requiring that chains be able to support a load. This oracle also classifies as chairs objects that the original domain theory rejects for having three legs (instead of four) or for lacking back support.

test examples. An analysis of variance indicates that the superiority of EBL-ANN is significant at the 99.9% confidence level both by comparison to EBL-ALONE and by comparison to ANN-ALONE. As also seen in the cup example, ANN-ALONE over generalizes, with the result that it correctly classifies all the chairs but incorrectly classifies almost 70% of the non-chairs. EBL-ANN also learns an overly general definition of chairs; however, it only misclassifies about 50% of the non-chairs.

****** Insert Table IX near here. ******

Figure 8 shows the average final network for recognizing chairs. (The exact weights and biases are reported in the appendix.) Comparing Figure 8 with Figure 5, it is clear that results of learning by the EBL-ANN system are less obvious for chairs than for cups. However, close scrutiny of Figure 8 reveals that back-propagation training corrects the results of the EBL system in several ways. First, the requirement for back support is dropped allowing Chairs *c2*, *c4* and *c5*, which do not have back support, to be accepted as chairs. Recall that in the cup domain theory, links are added but not dropped. The dropping of the requirement for back support shows that EBL-ANN has the ability to generalize rules by dropping antecedents in addition to its already demonstrated ability to specialize. Second, three-legged chairs are recognized (Chairs *c2*, *c3*, and *c4*). The weights are such that the *stable* is active when a chair has either three or four legs, so long as they are equal length. Third, to be a chair, an object must be able to support a sizable load. This feature, whose inclusion is required to correctly classify *n2* and *n5* as non-chairs, plays a significant role in the calculation of every hidden unit to which it is attached. This significant linkage of *supports load* to several hidden units again points to the need for a tighter integration of the EBL and ANN components, although it is possible that when using real world domains and large training sets, the frequency of “undesirable” connections will be reduced.

****** Insert Figure 8 near here ******

5. RELATED WORK

Three research areas bear directly on this work. One, there is a growing body of work on combining explanation-based and empirical approaches to learning. Two, there is additional research on using symbolic rules to guide neural network learning. Three, work on representing symbolic data structures in neural networks is relevant to the issue of mapping from the EBL subsystem into ANNs.

Others have investigated the combination of explanation-based and empirical learning methods (Lebowitz, 1986; Pazzani, 1988; Dietterich & Flann, 1988; Bergadano & Giordana, 1988; Danyluk, 1987). Some systems use empirical learning to provide input to an EBL system. For example, the UNIMEM (Lebowitz, 1986) system analyzes a collection of examples and finds similarities. These similarities are then explained using EBL. Approaches more closely related to the EBL-ANN approach are those that use EBL to produce training examples for an empirical learning system. For instance, the IOE system (Dietterich & Flann, 1988) uses a conventional inductive learning algorithm to empirically analyze a collection of explanations, thereby refining the domain theory. However, the bulk of this research on hybrid systems has focused on combining systems that perform conventional, “symbol-based” computation.

Two other projects which explore the possible synergies between explanation-based learning and neural networks were reported at the 1989 Machine Learning Workshop. Katz (1989) describes an approach which, like EBL-ANN, maps explanations into neural networks. However, the focus of his work is on improving the time required to classify an object rather than on revising the rules used to create the explanation. As a result, the initial resemblance of a network to an explanation structures is rapidly lost during learning. Thus, although the network may correct errors in an explanation structure, it is impossible to interpret those corrections as symbolic rules which could be used in the construction of future networks. Work by Diederich (1989) also integrates knowledge into a neural network. But the knowledge in Diederich’s networks consists of partial ISA hierarchies and the information learned by the network is the presented training instances and more complete ISA hierarchies. While this approach may be applied to classification tasks, it does not address the problems of incorrect or incomplete domain theories. In addition, Oliver and Schneider (1988) are investigating using rules to decompose problems into subproblems. Their results show that decomposing problems significantly reduces the learning time of neural networks. Unlike the EBL-ANN approach, they do not address the problem of dealing with incorrect rules.

Many researchers have investigated the representation of symbolic data structures in ANNs (Rumelhart, Smolensky, McClelland & Hinton, 1986; Shastri, 1988; Thrift, 1989; Whitehead & Ballard, 1988). This work is relevant to the further development of techniques for mapping an EBL system’s symbolic data structures into neural networks. Rumelhart, Smolensky *et al.* (1986) discuss the representation of schemata in neural networks. The results of EBL systems, which learn useful compositions of other concepts, can be viewed as schemata. Mappings from rules to neural networks have been proposed (Touretsky & Hinton, 1988; Gallant, 1988). These approaches, though, do not deal with mapping useful combinations of rules into ANNs, nor do they address the issue of correcting rules. In case-based reasoning systems, concrete episodes are stored; a neural approach for representing and selecting cases is described in (Thrift, 1989). EBL systems produce generalized cases, and the issue of selecting the appropriate generalization to apply to a new problem is important (Minton, 1988), but is currently ignored in the EBL-ANN system.

6. OPEN RESEARCH ISSUES

While the current implementation of the EBL-ANN system has produced promising results, several aspects of it require refinement and extension. In addition, the approach must be tested in several real-world domains, in order to get a better estimate of its value.

Adding Additional Hidden Units

The current algorithm adds input units in addition to those involved in the explanation, but does not add any hidden units. Incorporating more hidden units may increase the power of the algorithm. Rules of thumb for the amount and connectivity of additional hidden units are needed.

Allowing Variables

Another extension of the initial algorithm involves the use of variables. The current algorithm assumes explanations are *propositional* and do not involve variables. A major strength of EBL algorithms is that they determine the most general constraints on the objects in a problem, a task usually accomplished by reasoning about which variables have to be equated with constants or other variables. Techniques proposed in (Touretsky & Hinton, 1988; Smolensky, 1987) for variable binding in ANNs are being investigated. These techniques will be extended so that variables initially believed to be identical by the EBL subsystem can become separate variables following further training. Conversely, initially distinct variables may need to become unified.

Allowing Real-Valued and Hierarchical Features

A major shortcoming of the initial implementation is that all features must be *nominal*. That is, the possible values of a feature are described in an unordered set. The algorithm will be extended so that features that are *linear* (e.g., volume as a real number) or *hierarchical* (e.g., shapes where squares are polygons) can be handled. These three are the standard types of features used in inductive learning (Michalski, 1983).

One way of using real-valued features that nicely falls within the EBL-ANN approach is to have the EBL subsystem reason using qualitative values, while having the ANN subsystem use quantitative values. For example, in order to simplify the building of explanations, the

EBL system could use *small*, *medium*, and *large* as possible sizes. After the initial ANN is constructed, actual measurements could be used.

Providing Feedback from the ANN Subsystem to the EBL Subsystem

The research undertaken so far investigates having an EBL system provide initial guidance to an ANN. It is also desirable to have the results produced by the ANN affect the EBL portion of the system. Two approaches for doing so are under investigation. The first is to map the refined ANN back into the vocabulary of the EBL system so that the acquired concept can be symbolically analyzed and used by the EBL subsystem to acquire additional concepts. The second is to analyze what has changed in the ANN in order to locate the most uncertain portions of the concept, which can then be focussed upon. These two approaches are elaborated below.

After the ANN refines a rule acquired by the EBL portion of the system, the rule cannot be used by the EBL subsystem to produced additional explanations because it is not in the vocabulary of the EBL subsystem. Intuitively, the hybrid approach nicely matches the phenomenon where after extensive experience with a task, one can no longer articulate how decisions are made. However, techniques are being developed for mapping the ANN back into the EBL subsystem's vocabulary. The re-represented rules can then be used to build new explanations. If additional hidden units appear in the ANN, some undefined symbols may appear in the rules, thereby extending the vocabulary of the EBL subsystem. The techniques could also be used to make the knowledge encoded in an ANN human-comprehensible and to refine the basic rules of the EBL system. For example, after training the domain theory could be re-expressed and presented to the system's user.

By having a symbolic system analyze which weights in the ANN are fluctuating the most and by considering the magnitude of the back-propagated error signals, the most uncertain aspects can be localized. This information can be used in several different ways. For example, it can help produce focused questions of a human advisor, e. g., "Is feature *x* important?". Also, if the system designs its own experiments (Carbonell & Gil, 1987; Rajamoney, 1989), this information can suggest which experiments will prove informative.

Using EBL Failure to Direct Back-Propagation

An issue previously mentioned is that the EBL-ANN system does not always add new links in the most appropriate place. For example, the hybrid system learned that a strong positive weight from *supports load* to any of the four intermediate conclusions to which it is connected produces the correct classifications of the chairs and non-chairs in the training set. This problem is reduced by having a low setting for the parameter that determines when the output is close enough to being correct, because low settings lead to additional back-propagation training but at the cost of reducing the training time advantage of the hybrid. Methods of localizing back-propagation by examining explanation failure are being investigated in order to reduce the chance of inappropriate connections.

This method would only operate when back-propagation produced a false negative response. Rather than simply propagating error backward through the entire network, the EBL subsystem would be called, and an attempt to explain how the example fits the concept would be made. If the EBL subsystem is unable to complete the explanation, the areas of the explanation which are incomplete could be identified. Those identified areas would be the only areas of the ANN to receive back-propagated error. For example, in attempting to explain that a Styrofoam cup is a cup, both the stable and open-vessel consequents can be fully explained by the provided rules. However, liftable cannot be explained because graspable cannot be explained. Thus, the EBL component would indicate that back-propagation should focus its efforts on determining some other way to suggest that an object is graspable.

Applying EBL-ANN to Planning

Classification tasks provide an attractive test bed in which to initially develop the EBL-ANN approach. Inductive learning systems, such as ANNs, have long done well on this type of task. Planning is a more challenging task with which to further develop the approach. There has been much less progress on using inductive learning to produce plans. Largely, this is due to the interdependencies among the steps in a plan. A learner must be able to reason about the causal relationships among the entities in the plan in order to decide what is and what is not relevant. EBL systems are well-suited to learning about plans, as they are capable of explaining the interdependencies among the steps in a plan (Shavlik, in press-a). However, as previously mentioned, EBL systems are limited by the accuracy and coverage of their domain theories. Applying the EBL-ANN approach to planning problems promises to eliminate this limitation.

Techniques for representing and executing stored plans in neural networks are reported in (Whitehead & Ballard, 1988). The applicability of these techniques to the EBL-ANN approach are being investigated.

Acquiring Recursive Concepts

Many, if not most, concepts and plans have a recursive or iterative nature. That is, a concept can involve any number of repeated components and some actions can be performed as many times as necessary in order to achieve the goal of a plan. For example, a planner may need to make a number of adjustments in order to complete a task. Sometimes four adjustments may be needed, sometimes six, other times none. The exact number will depend on the details of the current situation — presence of other agents, weather conditions, etc. Nevertheless, underlying this variability is essentially a single plan. It would be foolish for a performance system to have a different plan for each possible number of adjustments.

The recently-developed BAGGER2 algorithm (Shavlik, in press-b) recognizes the presence of recursive processes in the explanations of specific problems. When any such traces are encountered, a recursive concept or plan is produced. When recursive processes are not detected, the result is the same as that produced by EGGS. The BAGGER2 algorithm greatly increases the generality of acquired concepts and plans, thereby reducing training time and improving the effectiveness of the system that uses the acquired results. This ability to acquire a recursive concept from a single training example is especially important in domains where only a small number of training scenarios are available.

Within neural network research, *recurrent* networks are used to represent recursive processes (Pineda, 1987). EBL-ANN is being extended so that recursive concepts can be acquired. This is being accomplished by using the BAGGER2 algorithm to produce roughly-correct concepts, which are then mapped into recurrent ANNs and refined.

Integrating Multiple Explanations into a Single ANN

A major issue in EBL is that a single training example may not suffice to fully learn the complete concept. For example, a general plan may involve several different ways to accomplish a subgoal while the training example may only use one of these. One way this problem can be addressed is by merging the results of multiple explanations into a single learned concept. Techniques are being developed for combining multiple explanations into a single ANN. A complicated concept, possibly containing several special cases, will be acquired by generalizing the explanations of several training examples. This means that the individual training examples can be simpler and, thus, more easily explained.

7. CONCLUSION

Using an explanation-based learning system to guide the construction of the initial neural network is an approach that overcomes the shortcomings of both explanation-based and neural learning. In the presented EBL-ANN approach, the initial neural network configuration is determined by the generalized explanation of the solution to a specific classification task. Since the construction of the initial network is motivated, fewer training examples are needed than if the initial network configuration is randomly chosen. The presence of irrelevant features is also less of a problem because the EBL subsystem focuses attention on the features most likely to be relevant. In addition, this hybrid system overcomes a major shortcoming of explanation-based learning, namely, the problem of incorrect domain theories.

In the EBL-ANN system, a “roughly-correct” domain theory leads to the acquisition of a classification rule that is nearly correct. Subsequent refinement by the neural network improves it. Preliminary empirical results indicate that the hybrid system more accurately learns concepts than just the explanation-based system by itself, that the hybrid learns concepts much faster than the neural learning system by itself, and that the results of its learning generalize to unseen examples better than either the neural learning system by itself or the explanation-based system by itself.

Most realistic problems can never be formalized exactly. However, there is much to be gained by utilizing the capability to reason approximately correctly. EBL provides a way to profitably use causal models of the world, while ANNs provide a way to refine roughly-correct concepts. By combining symbolic and neural learning algorithms, the presented research promises to broaden the applicability of machine learning techniques, producing learning algorithms that are not brittle and which can produce concepts whose accuracy improves through experience.

ACKNOWLEDGEMENTS

We would like to thank James Hendler and the reviewers for their detailed and valuable comments and suggestions. This research was partially supported by the University of Wisconsin Graduate School.

Table A-1. Average weights and biases of EBL-ANN for cups.

Feature	Open Vessel	Graspable	Stable	Liftable	Cup
BIAS	7.63	1.97	2.57	7.19	11.52
liftable					4.02
open vessel				0.45	6.58
graspable				4.71	
stable				0.15	5.58
flat bottom	-0.28	0.52	5.11		
made of styrofoam	1.21	3.01	1.68		
made of ceramic	0.21	0.05	0.12		
made of paper	-1.74	-2.55	-1.79		
has handle	0.24	4.86	0.15		
handle on top	-2.24	-0.47	-2.25		
handle on side	2.46	0.34	2.41		
light	0.29	0.57	0.15	5.20	
expensive	-1.42	-0.69	-1.55		
fragile	1.26	1.22	1.69		
has concavity	4.80	0.64	0.08		
concavity points up	5.36	0.71	0.11		

Table A-2. Average weights and biases of EBL-ANN for chairs.

Feature	Can Support	Stable	Support Bottom	Support Back	Comfortable	Chair
BIAS	7.87	5.65	13.44	3.21	8.70	9.16
can support						4.05
stable	-1.57					5.05
support bottom	5.23					
support back	3.36					
comfortable	0.74					9.47
equal length legs		7.43	-0.24	-0.22	0.42	
four legs		4.22	-1.68	-1.81	-1.64	
three legs		2.45	0.97	1.20	0.25	
has arms		0.87	1.15	1.28	1.02	
horizontal seat		1.73	4.31	-0.58	0.10	
med. size seat		1.72	4.30	-0.59	0.10	
med. height seat		1.68	4.30	-0.60	-1.40	
supports load		3.88	2.85	2.77	4.30	
vertical back		-0.83	-1.69	3.43	-2.39	
has wheels		0.48	-0.15	-0.25	0.18	
smooth		2.51	-0.22	-0.19	6.94	
padded		1.68	-0.68	-0.57	3.60	

References

[Ahalt89] S. C. Ahalt, F. D. Garber, I. Jouny and A. K. Krishnamurthy, "Performance of Synthetic Neural Network Classification of Noisy Radar Signals," in *Advances in Neural Information Processing Systems, Volume 1*, D. S. Touretsky (ed.), Morgan-Kaufmann, Los Altos, CA, 1989, pp. 281-288.

[Ahmad88] S. Ahmad, "A Study of Scaling and Generalization in Neural Networks," Technical Report CCSR-88-13, Center for Complex Systems Research, University of Illinois, Urbana, IL, November 1988.

[Baum89] E. B. Baum and D. Haussler, "What Size Net Gives Valid Generalization?," *Neural Computation 1*, (1989), pp. 151-160.

[Bennett88] S. Bennett, "Real World EBL: Learning Error Tolerant Plans in the Robotics Domain," *Proceedings of the AAAI Explanation-Based Learning Symposium*, Stanford, CA, March 1988, pp. 122-126.

[DeJong86] G. F. DeJong and R. J. Mooney, "Explanation-Based Learning: An Alternative View," *Machine Learning 1*, 2 (1986), pp. 145-176.

[Fisher89] D. H. Fisher and K. B. McKusick, "An Empirical Comparison of ID3 and Back-propagation," *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Detroit, MI, August 1989.

[Hammond88] K. J. Hammond and N. Hurwitz, "Extracting Diagnostic Features from Explanations," *Proceedings of the AAAI Explanation-Based Learning Symposium*, Stanford, CA, March 1988, pp. 31-35.

[Hinton86] G. E. Hinton and T. J. Sejnowski, "Learning and Relearning in Boltzmann Machines," in *Parallel Distributed Processing, Vol. I*, D. E. Rumelhart and J. L. McClelland (eds.), MIT Press, Cambridge, MA, 1986, pp. 282-317.

[Hirsh87] H. Hirsh, "Explanation-Based Generalization in a Logic-Programming Environment," *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, Milan, Italy, August 1987, pp. 221-227.

[Kedar-Cabelli87] S. T. Kedar-Cabelli and L. T. McCarty, "Explanation-Based Generalization as Resolution Theorem Proving," *Proceedings of the Fourth International Workshop on Machine Learning*, Irvine, CA, June 1987, pp. 383-389.

[McClelland81] J. L. McClelland and D. E. Rumelhart, "An Interactive Activation Model of the Effect of Context in Perception: Part 1. An Account of Basic Findings," *Psychological Review 88*, (1981), pp. 375-407.

[Michalski83] R. S. Michalski, "A Theory and Methodology of Inductive Learning," in *Machine Learning: An Artificial Intelligence Approach*, R. S. Michalski, J. G. Carbonell and T. M. Mitchell (eds.), Tioga Publishing Company, Palo Alto, CA, 1983, pp. 83-134.

[Minton88] S. N. Minton, *Learning Search Control Knowledge: An Explanation-Based Approach*, Kluwer Academic Publishers, Hingham, MA, 1988.

[Mitchell82] T. M. Mitchell, "Generalization as Search," *Artificial Intelligence 18*, 2 (1982), pp. 203-226.

[Mitchell86] T. M. Mitchell, R. M. Keller and S. Kedar-Cabelli, "Explanation-Based Generalization: A Unifying View," *Machine Learning 1*, 1 (1986), pp. 47-80.

[Mooney86] R. J. Mooney and S. W. Bennett, "A Domain Independent Explanation-Based Generalizer," *Proceedings of the Fifth National Conference on Artificial Intelligence*, Philadelphia, PA, August 1986, pp. 551-555.

[Mooney89a] R. J. Mooney, *A General Explanation-Based Learning Mechanism and its Application to Narrative Understanding*, Pitman, London, 1989.

[Mooney89b] R. J. Mooney, J. W. Shavlik, G. Towell and A. Gove, "An Experimental Comparison of Symbolic and Connectionist Learning Algorithms," *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Detroit, MI, August 1989, pp. 775-780.

[Quinlan86] J. R. Quinlan, "Induction of Decision Trees," *Machine Learning 1*, 1 (1986), pp. 81-106.

[Rajamoney87] S. Rajamoney and G. F. DeJong, "The Classification, Detection and Handling of Imperfect Theory Problems," *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, Milan, Italy, August 1987, pp. 205-207.

[Rosenblatt62] F. Rosenblatt, *Principles of Neurodynamics*, Spartan, New York, 1962.

[Rumelhart86a] D. E. Rumelhart, G. E. Hinton and J. L. McClelland, "A General Framework for Parallel Distributed Processing," in *Parallel Distributed Processing: Explorations in the Micro-Structure of Cognition*, D. E. Rumelhart and J. L. McClelland (eds.), MIT Press, Cambridge, MA, 1986, pp. 46-73.

[Rumelhart86b] D. E. Rumelhart, G. E. Hinton and J. R. Williams, "Learning Internal Representations by Error Propagation," in *Parallel Distributed Processing, Vol. I*, D. E. Rumelhart and J. L. McClelland (eds.), MIT Press, Cambridge, MA, 1986, pp. 318-362.

[Sejnowski87] T. J. Sejnowski and C. R. Rosenberg, "Parallel Networks that Learn to Pronounce English Text," *Complex Systems 1*, (1987), pp. 145-168.

[Shavlik89a] J. W. Shavlik, "Acquiring Recursive Concepts with Explanation-Based Learning," *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Detroit, MI, August 1989, pp. 688-693.

[Shavlik89b] J. W. Shavlik, R. J. Mooney and G. G. Towell, "Symbolic and Neural Net Learning Algorithms: An Experimental Comparison," Technical Report 857, Department of Computer Science, University of Wisconsin, Madison, WI, 1989.

[Shavlik90] J. W. Shavlik, *Extending Explanation-Based Learning by Generalizing the Structure of Explanations,* Pitman, London, 1990.

[Tesauro89] G. Tesauro and T. J. Sejnowski, "A Parallel Network that Learns to Play Backgammon," *Artificial Intelligence 39,* 3 (1989), pp. 357-390.

[Weiss89] S. M. Weiss and I. Kapouleas, "An Empirical Comparison of Pattern Recognition, Neural Nets, and Machine Learning Classification Methods," *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Detroit, MI, August 1989.