COMPARING LEARNING PARIDIGMS VIA DIAGRAMMATIC VISUALIZATION: A CASE STUDY IN CONCEPT LEARNING USING SYMBOLIC, NEURAL NET, AND GENETIC ALGORITHM METHODS

by

J. Wnek
J. Sarma
A. Wahab
R. S. Michalski

Proceedings of the 5th International Symposium on Methodologies for Intelligent Systems - ISMIS'90, Knoxville, TN. pp. 428-437, Oct. 1990.

COMPARING LEARNING PARADIGMS VIA DIAGRAMMATIC VISUALIZATION:

A Case Study in Single Concept Learning Using Symbolic, Neural Net and Genetic Algorithm Methods

Janusz Wnek, Jayshree Sarma
Ashraf A. Wahab and Ryszard S. Michalski

MLI 90-[illegible] 30

Shorter version of this paper will appear in Proceedings of the 5th International Symposium on Methodologies for Intelligent Systems, Knoxville, TN, October 1990.

COMPARING LEARNING PARADIGMS VIA DIAGRAMMATIC VISUALIZATION: A Case Study in Single Concept Learning Using Symbolic, Neural Net and Genetic Algorithm Methods

Janusz Wnek, Jayshree Sarma, Ashraf A. Wahab[1] and Ryszard S. Michalski

Center for Artificial Intelligence
Computer Science Department
George Mason University
4400 University Drive
Fairfax, VA 22030

ABSTRACT

Four different learning methods are experimentally compared by applying them to a series of simple, single concept learning problems. The methods compared include a rule-learning program, AQ15, a decision tree learning program, C4.5 (a successor of ID3), a program simulating a neural net trained by a backpropagation algorithm, BpNet, and a classifier system based on a genetic algorithm, CFS. The comparison employs a novel diagrammatic visualization technique that graphically represents training examples, the target and learned concepts, as well as the *exact error image*. The study described here is concerned with learning concepts generated by human subjects, thus, it is biased toward learning human-style descriptions (the next phase of research will consider other types of concepts). The results of experiments were that the ranking of the methods (from the lowest to the highest) on the basis of the average accuracy of the descriptions learned was the same as ranking on the basis of the simplicity of the descriptions, namely, the CFS method, BpNet, C4.5 and AQ15.

ACKNOWLEDGEMENTS

The authors thank Hugo de Garis who motivated us to perform these experiments, Kenneth De Jong for suggestions regarding the choice of criteria for comparison, and Jianping Zhang for help in performing the experiments. They also thank Janet Holmes for proof-reading of the article.

This research was supported in part by the Office of Naval Research under grant No. N00014-88-K-0397, in part by the Office of Naval Research under grant No. N00014-88-K-0226, and in part by the Defense Advanced Research Projects Agency under grant, administered by the Office of Naval Research, No. N00014-87-K-0874.

---

[1] *On leave from National Research Center, Giza, Egypt*

1. INTRODUCTION

In view of the great proliferation of learning paradigms and methods, there is significant interest in understanding the relationships among them, and in defining application areas for which they perform best. Recently, there have been several efforts to compare various learning systems, such as ID3, AQ15, neural nets using the backpropagation (BP) algorithm, and some statistical approaches.

For example, Fisher and McKusick (1989) compared ID3 and a neural net using BP algorithm on the problems of learning diagnostic rules for thyroid diseases and soybean plant diseases, and a few artificial problems. The comparison was based on the performance accuracy of testing examples and the training time. Their conclusion was that neural net gave better performance, but required a significantly longer training time and more training examples than ID3. Mooney et al (1989) compared ID3 with perceptron and backpropagation using the domain of soybean diseases, chess-end games, audiological disorders, and the Nettalk data set. Their conclusion was that the accuracy in classifying new examples was about the same for all the three systems, but the neural net performed better than ID3 when there was noise in the data. Weiss and Kapouleas (1989) compared ID3, predictive value maximization, neural net using BP, and a few statistical methods. They found that the statistical classifiers performed consistently better in terms of accuracy in classifying testing examples. Recently, Dietterich, Hild and Bakiri (1990) compared ID3 with a neural net using BP on the task of text-to-speech mapping. Their major conclusion was that neural net consistently outperformed ID3 in terms of the performance accuracy, and attributed this result to better capturing of statistical information by the neural net. In another recent study, Bergadano et al. (1990) compared POSEIDON (an extended version of AQ15 using a two-tiered concept representation) with exemplar-based type and decision tree type (ASSISTANT) learning programs. In their study involving two real world problems (labor contracts and congressional voting), descriptions learned by POSEIDON outperformed those produced by the other methods, both in terms of performance accuracy on new examples and in terms of the description's simplicity.

This study differs from the above in that it experimentally analyzes four different methods, compares them in terms of the *exact error rate* (rather than a statistical estimate), and also in terms of the complexity of the descriptions learned. The target and learned concepts are represented graphically by a novel technique of a *diagrammatic visualization* (Michalski, 1990). This technique permits one to display an *error image* that locates all errors precisely.

An important difference between learning approaches is in the cognitive aspect of the representation they use. Knowledge represented by logic-based rules and decision trees (especially when they are small) is relatively easy to comprehend, which is not the case with

knowledge represented by classifier systems or neural networks. Therefore, to evaluate the applicability of the methods to different problem areas, three problem types are distinguished: 1) learning cognitively-oriented concepts (the concepts have been generated by human subjects/experts, and the issue of the understandability of the learned descriptions is important), 2) learning concepts that represent some technical problem for which the understandability of concept descriptions is considered irrelevant, and 3) learning concepts generated randomly. This classification is useful because in some applications, e.g., in expert systems for human disease diagnosis, the comprehensibility of the learned knowledge is a crucial condition; while in some other applications this issue is irrelevant. The random concepts are used to test the ability of a method to learn any type of concepts.

In the first phase of our study, presented here, we compared various methods on problems of the first type, specifically, on learning concepts that are generated by human subjects, and thus naturally cognitively oriented. The current study therefore can be viewed as favoring methods employing symbolic representations, because such representations are more cognitively-oriented. The second phase is concerned with learning problems of the second and the third type, e.g., a 6- and 11-multiplexer; text-to-speech mapping (Sejnowski and Rosenberg, 1987), and several randomly generated problems; and will be published in a separate paper.

The methods compared include a neural net trained by a backpropagation algorithm (BpNet; Rumelhart et al., 1988), a classifier system employing a genetic algorithm (CFS; Riolo, 1988), the rule learning program AQ15 (Michalski et al, 1986), and the decision tree learning program C4.5 (a successor of ID3; Quinlan, 1986). These methods represent three basic learning paradigms: symbolic (rule base and decision tree learning), subsymbolic (neural net) and heterogeneous (a classifier system utilizing a genetic algorithm). In the process of learning, symbolic methods change knowledge structure, subsymbolic change parameters, while heterogeneous change both, the knowledge structure and parameters.

The comparison of the methods was done both in terms of the accuracy of the descriptions learned, and in terms of their complexity. To have some way to approximate the "cognitive" complexity of representations learned by these diverse methods, the concept of *disjunctive complexity* or, briefly, *D-complexity* of a representation was introduced. A precise measure of the D- complexity of a representation is defined by the number of conjunctive statements (rules) in the minimal DNF expression that is logically equivalent to the given representation. Because

finding such a minimal DNF expression for any given representation may be difficult (it is generally an NP-hard task), we use an estimate of the D-complexity.

For a method learning a rule-based representation, the number of rules generated by the method is simply taken as an estimate of D-complexity. For a decision tree learning method, the D-complexity is estimated by the number of leaves in the tree (since each leaf corresponds to a rule). For learning in neural nets and classifiers, the D-complexity is estimated by determining the number of conjunctive statements needed to re express the learned concept as a DNF formula. Such a formula can be obtained by determining a "cover" of the *image* of the learned concept in its *diagrammatic representation*. This representation employs the *General Logic Diagram* (GLD), which transforms a multidimensional space into a plane (Michalski, 1973, 1990; see next section).

The above methods were compared by applying them to the same group of concept learning problems. In our experiments, five target concepts were generated by different human subjects. For each target concept, increasing sets of training examples were generated, in order to determine the convergence of the learned concepts to the target concepts. The learned descriptions were compared in terms of the *exact error rate* , and in terms of a representation-dependent complexity, and an estimate of the representation-independent *D-complexity*.

2. EXPERIMENTAL METHODOLOGY

The experimental domain for testing the methods was the world of robots in the EMERALD[2] system, a large-scale system integrating several machine learning programs. For simplicity, the robots are described by just six multiple-valued attributes (Figure 1). The event space (the space of all possible robot descriptions) is 432, and the space of possible concepts is 2^{432} -1.

Attributes	Values of attributes
Head_shape	Round, Square, Octagon
Body_shape	Round, Square, Octagon
Is_Smiling	Yes, No
Holding	Sword, Balloon, Flag
Jacket_color	Red, Yellow, Green, Blue
Has_Tie	Yes, No

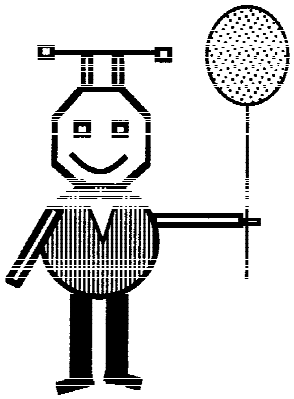


Figure 1. An example of a robot figure and descriptive attributes used.

In the experiment, human subjects were asked to create five different concept descriptions characterizing various selections of robots from a predefined set. Each such concept thus represents a partitioning of the set into those robots that belong to the concept and those that do not. The concepts descriptions were:

C_1: *Head is round and jacket is red or head is square and is holding a balloon*

C_2: *Smiling and is holding balloon or head is round*

C_3: *Smiling and not holding sword*

C_4: *Jacket is red and is wearing no tie or head is round and is smiling*

C_5: *Smiling and holding balloon or sword*

These descriptions were the basis for generating positive and negative training examples. For each concept C_i, five pairs of sets of training examples were generated: Set1a and Set1b (two different sets of 6% of the complete set of positive examples, and 3% of the complete set of negative examples), Set2a and Set2b (10% positive and 10% negative), Set3a and Set3b (15% positive and 10% negative), Set4a and Set4b (25% positive and 10% negative), Set5a and Set5b (100% positive and 10% negative). With the exception of the first pair, which was generated by

[2] EMERALD was developed at the Center for Artificial Intelligence at George Mason University; an earlier version of it was developed at the University of Illinois at Urbana (Kaufman, Michalski and Schultz, 1989).

the subjects who created the concepts, all other pairs were generated randomly.

The performance of each system was tested for all robot descriptions (432 different descriptions) and was evaluated in terms of error image (Figure 3), exact error rate and D-complexity. For each learned description, the exact error rate was determined (the cardinality of the set-difference between the learned concept and the target concept). The error rate was measured as a function of the number of training examples. This allowed to see the convergence of the learned concept to the target concept with the number of training examples. The D-complexity for descriptions generated by CFS and BpNet was estimated on the basis of the concept images represented through diagrammatic visualization (see section 4).

3. LEARNING SYSTEMS INVOLVED IN THE STUDY

As mentioned earlier, the symbolic paradigm was represented by the AQ15 rule learning program, and the C4.5 decision tree learning program. Of the various neural net algorithms developed, BpNet, the back- propagation has been the most popular. It is able to learn concepts that are not linearly separable by training the hidden layer nodes. The shell for the classifier system was developed by R. Riolo (1988). CFS-C package of subroutines and data structures is domain independent and provides routines to perform the *major cycle* of a classifier system. The implementation of the CFS classifier system for classification purposes was done in C (Wnek, 1990). AQ15 program is written in Pascal, C4.5 and BpNet are written in C.

NEURAL NET - BpNet

Each of the five backpropagation nets was trained using the corresponding positive and negative examples coded as binary strings. The learning parameters in BpNet system were set to: eta= 0.25, alpha= 0.50; the number of hidden units was set to 10% of the total number of input and output units (this was established after some experiments with BpNet was done seeking the best performance for classification issue.) Networks have been trained until they reached r.m.s. (root mean square) error below 0.0007.

CLASSIFIER SYSTEM - CFS

The program performs the following steps of the *major cycle* of classifier system:

- compares messages with classifiers and record all matches
- calculates bids, runs a competition, generates new messages by activating the strongest

classifiers; matches and activates effectors

- redistributes the payoff between classifiers by applying the bucket-brigade algorithm (BBA)

- applies genetic algorithms operators: crossover and mutation to generate new classifiers.

The classification was done by CLASS system which implements domain-dependent parts of the classifier system, e.g., emulation of detectors and effectors, payoff function. The system is working in the stimulus-response mode (no internal messages are posted.) The population of 60 classifiers is trained as the examples are fed into the system. The training process takes about 50 classifier system cycles per example. The reward for correct/incorrect answer is 6/-1, respectively (a full reward is paid to all active classifiers). The CLASS system has two effectors. Each of them consists of two basic parts: a condition and an action which is a classification.

DECISION TREE LEARNING PROGRAM - C4.5

The C4.5 program is a derivative of the ID3 program (Quinlan, 1986). ID3 builds the decision tree as the representation for the concept. Each interior node of the tree is associated with an attribute value while the leaf node represents the concept class, which is a conjunction of the attribute values. The arc out of the interior node represents a value of the attribute. Each path in the decision tree can then be considered as a distinct production rule, which is mutually exclusive. The algorithm starts with a training set of events belonging to the different classes. It selects a random subset of events (window) and compares the information measures of each attribute for the set. The attribute having the highest score is selected as the root of the tree. From this it generates a decision tree, adds misclassified objects and continues until the trial decision tree correctly classifies all objects not in the window. The algorithm iterates until each node has only events that belong to one class. The entire process is repeated by default 10 times.

RULE LEARNING PROGRAM - AQ15

AQ15 generates a decision rule by performing a heuristic search through a space of logical expressions and determines the rule that satisfies all the positive examples (E+) and none of the negative examples (E-) (Michalski et al, 1986). The goal of AQ is to find the most preferred rule according to the preference criterion specified. The final rules produced are in the Variable valued Logic VL1 language (Michalski and Larson, 1983). To do so, the program finds a cover (E+|E-), and a cover is defined as a disjunction of complexes which describes all the positive examples and none of the negative examples of the concept.

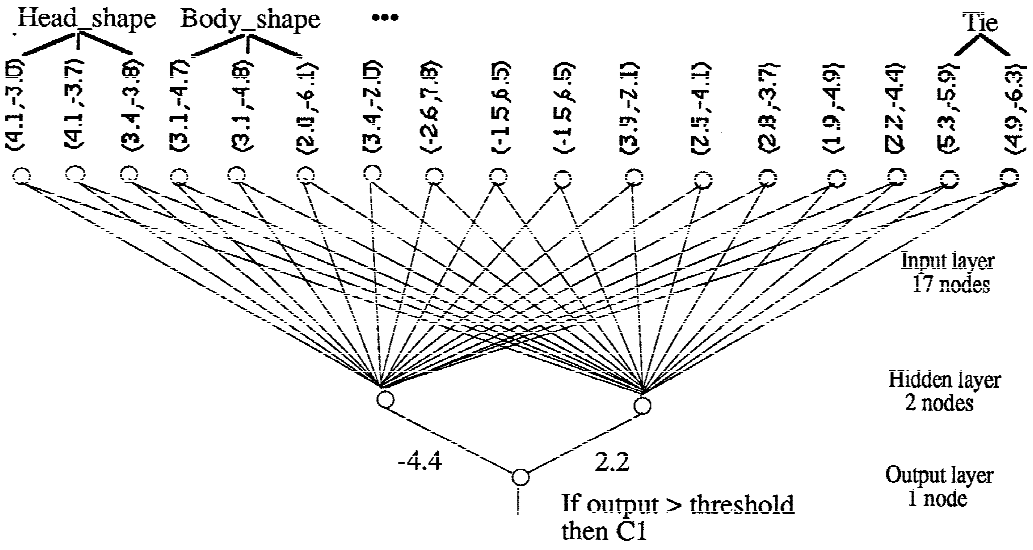
For all our experiments we used the default settings for AQ15 (no truncation, strict matching, intersecting covers). The intersecting covers mode was used which produces rules that may

intersect over areas of the event space where there are no learning events. Also the program tries to minimize the number of rules and the number of conditions.

4. REPRESENTATIONS LEARNED FROM EXAMPLES

For illustration and a comparison of representations, below is an example of the representation learned by each method. The representations shown below were learned from 6% positive and 3% negative examples of concept *C1* (*"Head is round and jacket is red or head is square and is holding a balloon"*).

A neural net generated by BpNet
(a 20-node net trained by backpropagation algorithm):



In each pair (a, b) a is the weight of the link to the left hidden node, and b is the weight to the link to the right node.

Classifiers generated by CFS (cycle-step 901):

Id	Classifier	Strength	BidRatio
1227	m000000 01 0# 1 10 #1 1, m00#00#0100110#1# /PASS/ 101101100#0000 0#	222	0.81
1236	m000000 01 0# 1 10 #1 1, m00#00#0100110#1# /PASS/ 101101100#0000 0#	219	0.81
1217	m0#0000 00 10 1 00 0# 0, m##0000#010100#0# /PASS/ 100#1101101100 11	208	0.78
0017	m000000 01 01 1 10 01 1, m00000#0100110111 /PASS/ 101101100#0000 0#	34	0.25

Each line above represents one classifier. The total population for representing the concept consists of 60 classifiers. The number of classifiers, as well as about 20 other parameters were determined experimentally; the remaining parameters from the total of about 150 took default values). For details see (Wnek, 1990).

A decision tree generated by C4.5

jacket_color = yellow: Unfriendly (2.0)
jacket_color = blue: Unfriendly (3.0)
jacket_color = red:
    head_shape = round: Friendly (3.0)
    head_shape = square: Friendly (1.0)
    head_shape = octagon: Unfriendly (1.0)
jacket_color = green:
    head_shape = round: Unfriendly (1.0)
    head_shape = square: Friendly (1.0)
    head_shape = octagon: Unfriendly (3.0)

A decision rule generated by AQ15

C1 <= (head = round or square) & (holding = sword or balloon) & (jacket = red or green)

To illustrate learned concept descriptions and locate their errors, as well as to determine their D-complexity, a *diagrammatic visualization* technique was employed (Michalski, 1990). This technique uses a planar diagram for representing a multidimensional space spanned over multi-valued attributes, and permits one to display the *error image*, which is an exact characterization of the errors (the set-difference between the target and learned concepts). In such a diagram, each combination of the values of the attributes (an instance of a concept) is represented as a small cell. For example, the upper left corner cell in each of the four diagrams in Figure 2 represents an example described by: Head_shape = round, Body_shape = round, Is-smiling = yes, Holding = sword, Jacket_color = red, Has_tie = yes. Positive and negative training examples are marked by + and -, respectively.

The individual diagrams in *Figure 2* present the target concept C1 (sparsely shaded), and the concept learned (densely shaded) by BpNet, CFS, C4.5 and AQ15, respectively. In this case, the training set consists of 6% of all possible positive examples (84), and of 3% of all possible negative examples (348). The areas of the *target concept* not covered by the *learned concept*

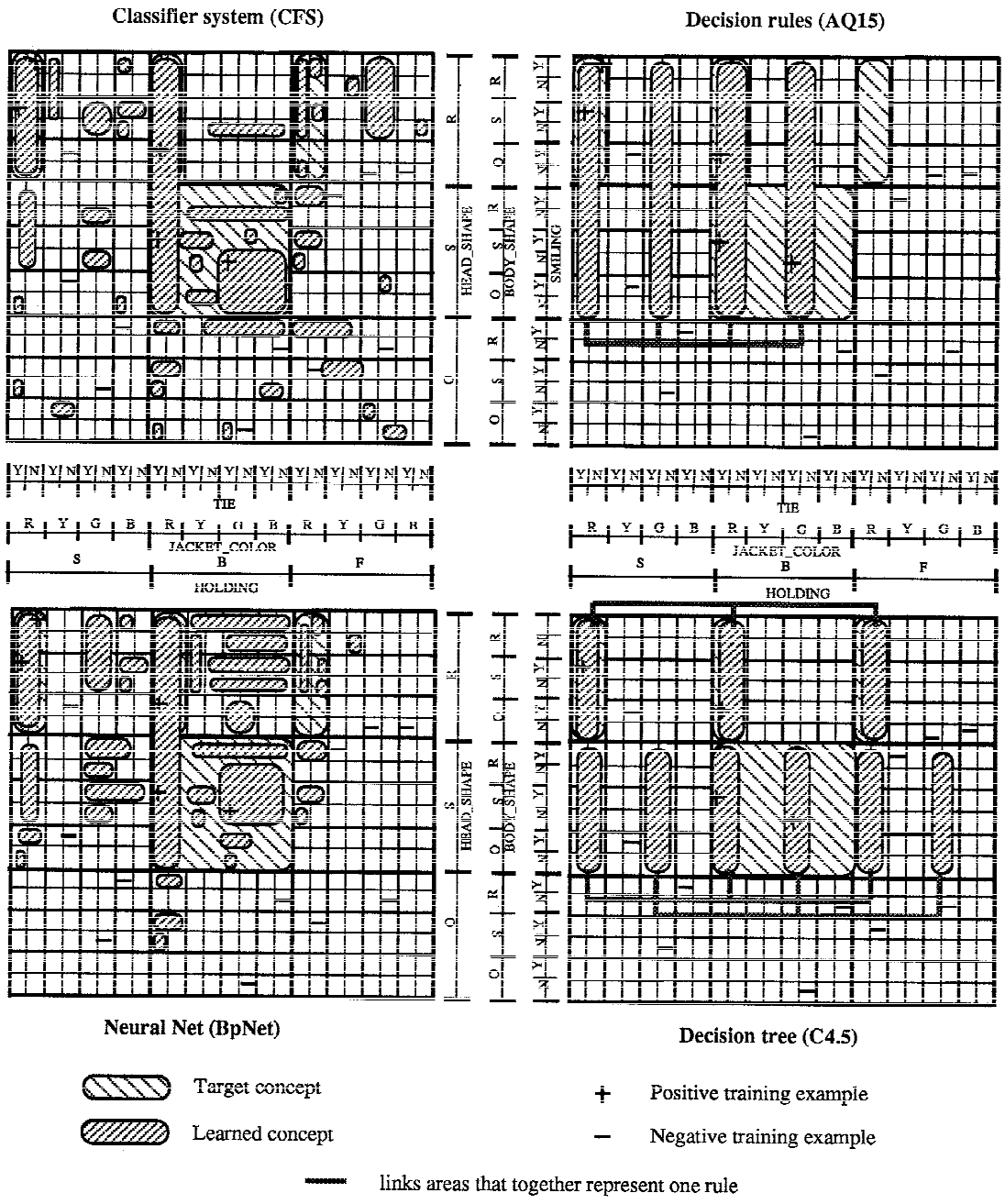


Figure 2. Images of the *target* and *learned concept C1*
(from a training set of 6% positive and 3% negative examples)

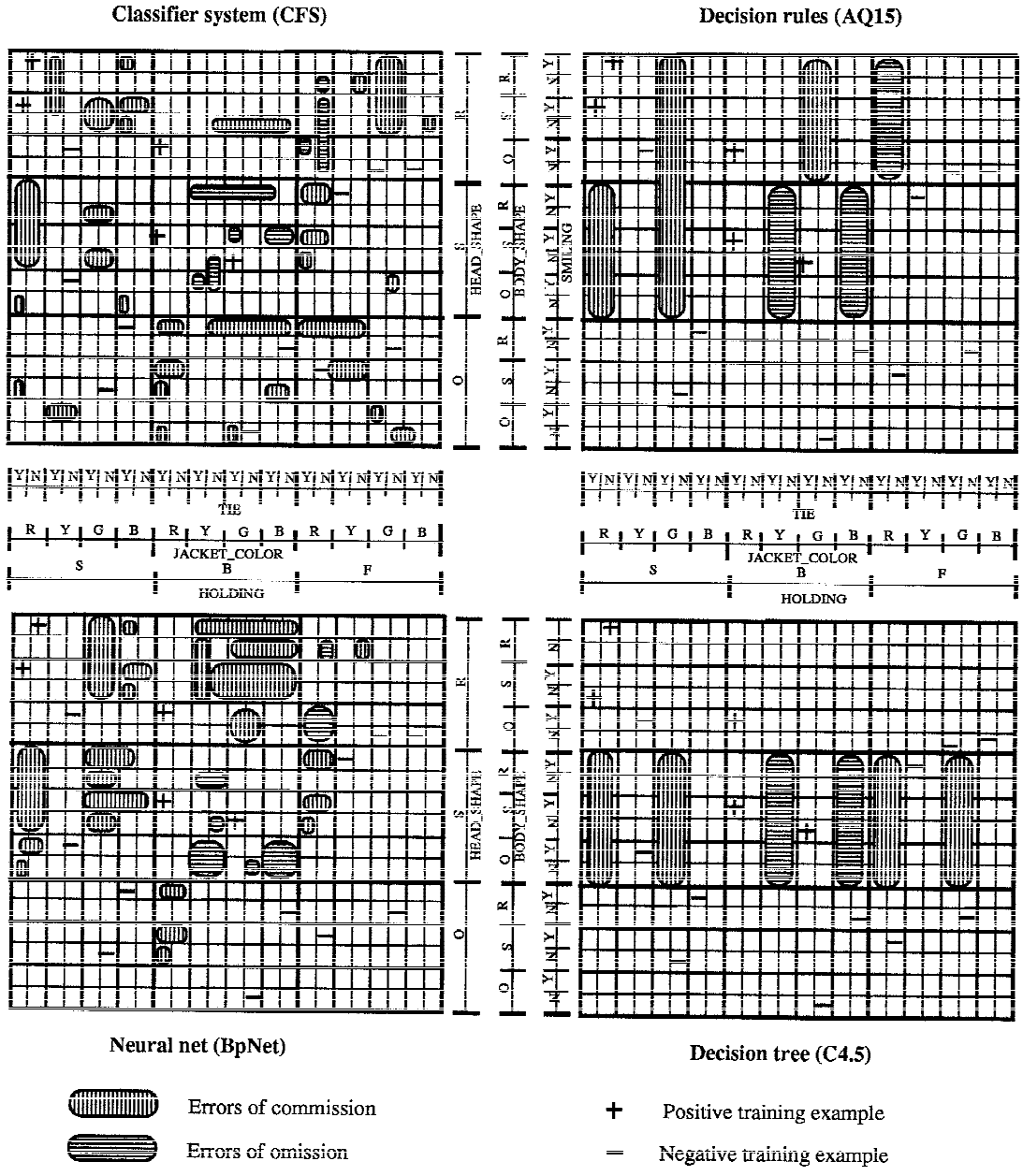


Figure 3. Error images of the learned *concept C1*
(from the training set of 6% positive and 3% negative examples)

represent *errors of omission*, while areas of the learned concept not covered by the target concept represent *errors of commission*. The union of both types of errors represents the *error image* (Figure 3).

5. EXPERIMENTAL RESULTS

Tables 1 and *Table 2* summarize results of all experiments. *Table 1* gives the average exact error rate, and *Table 2* gives the average D-complexity of descriptions learned from different training sets. Pairs (a,b) in the top row of each table denote the percentage of positive and negative examples, respectively, used in experiments. In Table 1, each number in every column represents the average exact error rate for all five concepts learned from two different subsets of training examples (Set1a and Set1b, Set2a and Set2b, ...), i.e., an average of 10 error rates. In these experiments, target concepts were precisely defined and there was no noise in training data sets. Therefore, as results from AQ15 were used complete and consistent concept descriptions, rather than truncated descriptions (Bergadano, et al., 1990). For the same reason, as results from C4.5 were used unpruned decision trees.

	(6%, 3%)	(10%, 10%)	(15%, 10%)	(25%, 10%)	(100%, 10%)
Genetic Alg. (CFS)	21.3%	20.3%	22.5%	19.7%	16.3%
Neural Nets (BpNet)	9.7%	6.3%	4.7%	7.8%	4.8%
Decision trees (C4.5)	9.7%	8.3%	1.3%	2.5%	1.6%
Decision rules (AQ15)	22.8%	5.0%	4.8%	1.2%	0.0%

Table 1. The dependence of the average error rate on the number of training examples.

The average error rate (4.8% in the case of 100% positive and 10% negative training) of the BpNet-generated concepts was primarily due to an inadequate learning of concepts C1 and C4. The error rate of the CFS-C-generated descriptions was much higher then that of the other descriptions, and what is most surprizing, did not improve much with the growth of training sets. For AQ15-generated descriptions, the error rate of descriptions learned from less than 100% positive examples was primarily due to the description of concept C1. Decision trees

generated by C4.5 produced some error rate[3] even when 100% positive examples were given. This error was due to the description of concept C4.

The average D-complexity was determined over five concept descriptions learned for each of the 10 training sets. The D-complexity of the CFS- and BpNet-generated descriptions was derived by counting the estimated minimum number of rules needed to represent (to cover exactly) the image of the concepts in the diagrammatic representation.

	(6%, 3%)	(10%, 10%)	(15%, 10%)	(25%, 10%)	(100%, 10%)
Genetic Alg. (CFS)	49	45	51	48	41
Neural Nets (BpNet)	35	26	12	22	12
Decision trees (C4.5)	3.1	2.8	2.5	2.5	2.5
Decision rules (AQ15)	2.6	2.2	2	1.6	1.6

Table 2. The dependence of the D-complexity on the number of training examples.

6. CONCLUSION

Four learning methods, CFS, BpNet, C4.5 and AQ15, were compared in terms of the average exact error rate and the average D-complexity of concept descriptions learned from the same sets of training data The results have shown that for the cognitively oriented target concepts used in the experiments, the symbolic methods, AQ15 and C4.5, outperformed the non-symbolic methods, BpNet and CFS, both in terms of the average exact error rate and the D-complexity of the descriptions learned. One interesting finding is that increasing the number of training examples from 6%pos and 3%neg to 100%pos and 10%neg resulted in only a slight improvement of the performance of the CFS-generated descriptions (from 21.3% to 16.3%). Other interesting findings are that even with 100% positive examples the neural net, the genetic algorithm, and to a smaller degree the decision tree method did not learn the concept precisely.

[3] C4.5 has a feature for converting trees into rules (which involves pruning a tree and simplifying rules.) In this case, concept C4 was classified with 27% error for (6%, 3%) training set, and no error for the other training sets.

The diagrammatic visualization has proven to be a very useful method for illustrating learning processes. For symbolic learning systems there is a direct mapping between the description acquired by the system and the concept image in the diagram. For non-symbolic methods, the concept image is determined by mapping into the diagram the decisions suggested by the description for each possible instance. The so obtained concept image can then be used for re-reprepresenting the concept in the form of a DNF description (or logic-style rules). As knowledge encoded in a neural net or a population of classifiers is hard to comprehend, this way the diagrammatic visualization enables us to get an insight into what was learned by these methods. Because the size of the description space that can be represented in a diagram is limited by the size of the display, the effect of the technique is the strongest for relatively small domains (no more than 10-12 attributes). For larger problems one can create a hierarchy of diagrams.
The CFS classifier system seems to be not well suited for the classification-type problems. To test further this finding, in the future research we plan to experiment also with another implementation of CFS and another genetic algorithm-based system.

As mentioned earlier, target concepts were generated by human subjects, and therefore the study favored methods that use symbolic representations, as such representations are more closely related to human representations. Studying how systems learn such human-generated concepts is important for applications where knowledge that needs to be acquired is in such cognitively-oriented forms, and/or applications where the knowledge learned needs to be understandable by human experts.

There are problem domains in which these factors are not relevant. Therefore, to make a complete evaluation of the relative performance of these methods, the future research will investigate other problem types, for example, special technical problems, such as text-to-speech mapping (Dietterich 1990), and concepts generated randomly (Wnek, 1990). Such studies should give us an insight into the performance of different methods in learning any kind of concepts. Future research might also compare the performance of the methods in learning from noisy data or from inconsistent example sets, and in learning imprecisely defined concepts.

REFERENCES

Bergadano, F., Matwin, S., Michalski, R.S., Zhang, J., "Learning Two-Tired Descriptions of Flexible Concepts: The POSEIDON System," submitted to *Machine Learning Journal,* 1990.

Dietterich, T.G., Hild, H., Bakiri, G., "A Comparative Study of ID3 and Backpropagation for English Text-to-Speech Mapping," *Proceedings of the 7th International Conference on* Machine Learning, Austin, TX, 1990.

Fisher, D. H., McKusick, K. B., "An Empirical Comparison of ID3 and Back-propagation," *Proceedings of IJCAI-89*, Detroit, MI, pp. 788-793, August 1989.

Kaufman, K.A., Michalski, R.S., and Schultz, A.C., "EMERALD1: An Integrated System of Machine Learning and Discovery Programs for Education and Research," George Mason U., *Reports of Machine Learning and Inference Laboratory,* No.MLI-89-12, 1989.

Michalski, R.S., and Larson, J.B., "Incremental Generation of VL1 Hypotheses: The Underlying Methodology and the Description of program AQ11, George Mason University, *Reports of Machine Learning and Inference Laboratory*, No. MLI-83-11, 1983.

Michalski, R., Mozetic, I., Hong, J., and Lavrac, N., "The AQ15 Inductive Learning System: An Overview and Experiments," George Mason University, *Reports of Machine Learning and Inference Laboratory*, No. MLI 86-6, 1986.

Michalski, R.S., "Diagrammatic Visualization of Learning Process," George Mason University, *Reports of Machine Learning and Inference Laboratory*, No. MLI-90, 1990.

Mooney, R., Shavlik, J., Towell, G., Gove, A., "An Experimental Comparison of Symbolic and Connectionist Learning Algorithms," *Proceedings of IJCAI-89*, Detroit, MI, pp. 775-780, 1989.

Quinlan, J. R., "Induction of Decision Trees," *Machine Learning*, pp. 81-106, 1986.

Rendell, L. A., Cho, H. H., Seshu, R., "Improving the Design of Similarity-Based Rule -Learning Systems," *International Journal of Expert Systems*, 2, 1, pp. 97-133, 1989.

Riolo, R.L., "CFS-C: A Package of Domain Independent Subroutines for Implementing Classifier Systems in Arbitrary, User-Defined Environments," Logic of Computers Group, Division of Computer Science and Engineering, University of Michigan, Ann Arbor, 1988.

Rumelhart, D. E., Hinton, G. E., Williams, J. R., "Learning Internal Representations by Error Propagation," *Parallel Distributed Processing, Vol.1*, Rumelhart & McClelland (eds.), pp. 318-362, 1988.

Sejnowski, T.J., and Rosenberg, C.R., "Parallel networks that learn to pronounce English text. *Complex Systems*, Vol. 1, pp. 145-168, 1987.

Utgoff, P. E., "ID5: An Incremental ID3," *Proceedings of the 5th International Conference on* Machine Learning, Ann Arbor, MI, 1988.

Weiss, S. M., Kapouleas, I., "An Empirical Comparison of Pattern Recognition of Neural Nets, and Machine Learning Classification Methods," *Proceedings of IJCAI-89*, Detroit, MI, pp. 781-787, 1989.

Wnek, J., "Learning to Classify with the CFS Classifier System," George Mason University, *Reports of Machine Learning and Inference Laboratory*, No. MLI-90, 1990.

Wnek, J., "A Comparison of Learning Paradigms on Concepts Generated Randomly," George Mason University, *Reports of Machine Learning and Inference Laboratory,* No. MLI-90, 1990 (to appear.)