Expert Prediction, Symbolic Learning, and Neural Networks

An Experiment on Greyhound Racing

Hsinchun Chen, Peter Buntin Rinde, Linlin She, Siunie Sutjahjo, Chris Sommer, Daryl Neely, University of Arizona

UNCERTAINTY IS INEVITABLE IN problem solving and decision making. One way to reduce it is by seeking the advice of an expert. When we use computers to reduce uncertainty, the computer itself can become an "expert" in a specific field through a variety of methods. One such method is machine learning, which involves using a computer algorithm to capture hidden knowledge from data. Machine learning usually encompasses different types of solutions, such as decision trees, production rules, and neural networks.[1] (See the sidebar on "AI approaches to uncertainty.)

We compared the prediction performances of three human track experts with those of two machine learning techniques: a decision-tree building algorithm[2] (ID3), and a neural network learning algorithm[3] (backpropagation).

Most of the applications on which machine learning has been tested are in engineering or biomedical domains. These domains are complex and interesting, but the data sets used for testing were relatively clean and structured. For our research, we investigated a different problem-solving scenario called *game playing*, which is unstructured, complex, and seldom-studied.

We considered several real-life game-playing scenarios and decided on greyhound racing, a complex domain that involves about 50 performance variables for eight competing dogs in a race.

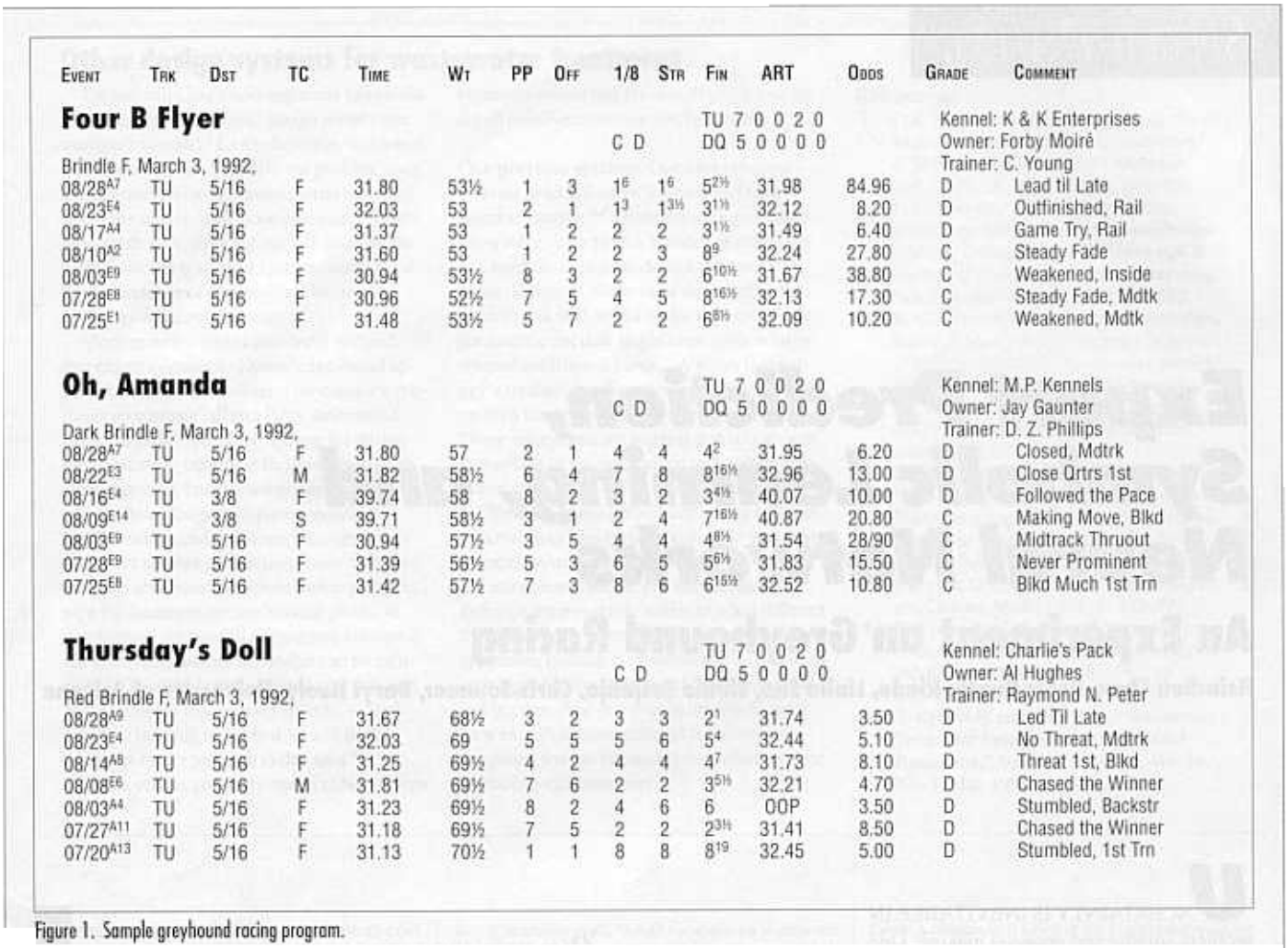
OUR RESEARCH EXPLORED TWO QUESTIONS: CAN MACHINE-LEARNING TECHNIQUES REDUCE UNCERTAINTY IN GAME-PLAYING SCENARIOS? CAN THESE METHODS OUTPERFORM HUMAN EXPERTS IN PREDICTION?

For every race, each dog's past history is complete and freely available to bettors. This is a large amount of historical information — some accurate and relevant, some noisy and irrelevant — that must be filtered, selected, and analyzed to assist in making a prediction. This large search space poses a challenge for both human experts and machine-learning algorithms. The questions then become: Can machine-learning techniques reduce the uncertainty in a complex game-playing scenario? Can these methods outperform human experts in prediction? Our research sought to answer these questions.

Setting up the experiments

The Tucson Greyhound Park in Tucson, Arizona, holds about 112 races in an average week. The park makes detailed programs available to its patrons (see Figure 1). Each program contains about 15 races, with race grades varying from A (the most competitive) to D. A few special races with grades such as "M" (maiden race) are also included, but were ignored in our experiments. Each race program includes information about eight dogs, including each dog's fastest time, the dog's total races, and its number of first, second, third, and fourth place finishes. Directly below the dog's summary data, the program lists its performance for the last seven of its

0885-9000/94/$4.00 © 1994 IEEE



Event	Trk	Dst	TC	Time	Wt	PP	Off	1/8	Str	Fin	ART	Odds	Grade	Comment
Four B Flyer										TU 7 0 0 2 0				Kennel: K & K Enterprises
								C	D	DQ 5 0 0 0 0				Owner: Forby Moiré
Brindle F, March 3, 1992,														Trainer: C. Young
08/28[A7]	TU	5/16	F	31.80	53½	1	3	1^{6}	1^{6}	$5^{2½}$	31.98	84.96	D	Lead til Late
08/23[E4]	TU	5/16	F	32.03	53	2	1	1^{3}	$1^{3½}$	$3^{1½}$	32.12	8.20	D	Outfinished, Rail
08/17[A4]	TU	5/16	F	31.37	53	1	2	2	2	$3^{1½}$	31.49	6.40	D	Game Try, Rail
08/10[A2]	TU	5/16	F	31.60	53	1	2	2	3	8^{9}	32.24	27.80	C	Steady Fade
08/03[E9]	TU	5/16	F	30.94	53½	8	3	2	2	$6^{10½}$	31.67	38.80	C	Weakened, Inside
07/28[E8]	TU	5/16	F	30.96	52½	7	5	4	5	$8^{16½}$	32.13	17.30	C	Steady Fade, Mdtk
07/25[E1]	TU	5/16	F	31.48	53½	5	7	2	2	$6^{8½}$	32.09	10.20	C	Weakened, Mdtk
Oh, Amanda										TU 7 0 0 2 0				Kennel: M.P. Kennels
								C	D	DQ 5 0 0 0 0				Owner: Jay Gaunter
Dark Brindle F, March 3, 1992,														Trainer: D. Z. Phillips
08/28[A7]	TU	5/16	F	31.80	57	2	1	4	4	4^{2}	31.95	6.20	D	Closed, Mdtrk
08/22[E3]	TU	5/16	M	31.82	58½	6	4	7	8	$8^{16½}$	32.96	13.00	D	Close Qrtrs 1st
08/16[E4]	TU	3/8	F	39.74	58	8	2	3	2	$3^{4½}$	40.07	10.00	D	Followed the Pace
08/09[E14]	TU	3/8	S	39.71	58½	3	1	2	4	$7^{16½}$	40.87	20.80	C	Making Move, Blkd
08/03[E9]	TU	5/16	F	30.94	57½	3	5	4	4	$4^{8½}$	31.54	28/90	C	Midtrack Thruout
07/28[E9]	TU	5/16	F	31.39	56½	5	3	6	5	$5^{6½}$	31.83	15.50	C	Never Prominent
07/25[E8]	TU	5/16	F	31.42	57½	7	3	8	6	$6^{15½}$	32.52	10.80	C	Blkd Much 1st Trn
Thursday's Doll										TU 7 0 0 2 0				Kennel: Charlie's Pack
								C	D	DQ 5 0 0 0 0				Owner: Al Hughes
Red Brindle F, March 3, 1992,														Trainer: Raymond N. Peter
08/28[A9]	TU	5/16	F	31.67	68½	3	2	3	3	2^{1}	31.74	3.50	D	Led Til Late
08/23[E4]	TU	5/16	F	32.03	69	5	5	5	6	5^{4}	32.44	5.10	D	No Threat, Mdtrk
08/14[A8]	TU	5/16	F	31.25	69½	4	2	4	4	4^{7}	31.73	8.10	D	Threat 1st, Blkd
08/08[E6]	TU	5/16	M	31.81	69½	7	6	2	2	$3^{5½}$	32.21	4.70	D	Chased the Winner
08/03[A4]	TU	5/16	F	31.23	69½	8	2	4	6	6	OOP	3.50	D	Stumbled, Backstr
07/27[A11]	TU	5/16	F	31.18	69½	7	5	2	2	$2^{3½}$	31.41	8.50	D	Chased the Winner
07/20[A13]	TU	5/16	F	31.13	70½	1	1	8	8	8^{19}	32.45	5.00	D	Stumbled, 1st Trn

Figure 1. Sample greyhound racing program.

races, which includes the dog's starting position, its position during the first turn (called the break position), its position in the second and third turns, and its finishing position. In addition, its race time and the grade of the race are recorded. The park also publishes the previous day's results. These contain information about how each dog fared, along with the payoff odds on the winning dogs.

The park also enlists the prediction services of three dog racing experts, who are not affiliated with the park. To our knowledge, their predictions are based solely on their own thinking, and the park does not compensate them for publishing their advice in the daily programs. These experts base their predictions on the same information available to any bettor in the daily program.

In a typical race, the program contains about 50 variables that might affect the outcome of that race. Among these are the condition of the track, the weather, the dog owner and trainer, and the physical attributes of each dog. Using a "multiple representation strategy" that bridges the gap between structural representation and performance representation,[3] most of these variables in the program can be considered structural variables, from which a small set of performance-related variables must be identified (either algorithmically or by some domain experts).

The "domain model"[4] and "abstraction space"[5] (of the underlying domain) also produce better classification results and substantial speedup (that is, they provide guidance for the rule formation program in a rule space that is too large to search exhaustively, and in a domain where trivial associations far outnumber important ones.[4])

As is typical of complex problem-solving scenarios, the first, most important, and most time-consuming task was to reduce the problem's complexity by pruning the problem space: deciding on a smaller set of relevant performance attributes. The inability of the machine learning algorithms to analyze the initial noisy and fussy problem domain prompted us to rely on human experts' knowledge and heuristics. (This may prove to be one of the weaknesses of some prevailing machine-learning techniques, and is an area that deserves more research attention.)

As in horse racing prediction,[6] the selection of performance attributes in our research was mainly based on the opinions of frequent bettors, track experts, and the management of the park. These experts succinctly identified performance variables that they thought were the most crucial in predicting winners. Their domain knowledge helped to reduce the problem space significantly. In total, the experts suggested 10 performance variables:

- *fastest time:* the fastest time in seconds for a 5/16 mile race;
- *win percentage:* the number of first places divided by the total number of races;
- *place percentage:* the number of second places divided by the total number of races;
- *show percentage:* the number of third places divided by the total number of races;
- *break average:* the dog's position during the first turn (averaged over the seven most recent races);
- *finish average:* the average finishing position over the previous seven races;
- *time 7 average:* the average finishing time of the seven most recent races;

AI approaches to uncertainty

Over the past 20 years, various AI techniques have been developed and tested to reduce complexity and uncertainty. Systems based on heuristics and algorithms with learning capability have gained significant attention among researchers.

Expert systems. One method of problem solving involves "expert" or knowledge-based systems. An expert system has been defined as "a computer program that represents and reasons with knowledge of some specialist subject with a view of solving problems or giving advice."[1] The idea is to make an expert-like system by storing large amounts of domain-specific knowledge with "condition-action" pairs, coupled with an inference engine that simulates human reasoning.

Some knowledge-based systems may exhibit "weak" problem-solving methods for various application domains, adopting human problem-solving strategies (such as means-end analysis) or general information-processing heuristics (such as the Occam's Razor heuristic).

Because expert systems aim to solve real-world problems of significant complexity and uncertainty, most of them offer rule-based solutions to real-world problems. Applications of expert systems have ranged from the analysis of chemical compounds, to the diagnosis of diseases, to the configuration of computer systems.

Despite their usefulness, expert systems are *performance systems*[2] — they only perform what they were programmed to do because they lack learning ability. In addition, significant effort is often required to obtain knowledge from domain experts, and to maintain and update the knowledge base.

Machine learning. In contrast to *performance systems,* which acquire knowledge from human experts, *learning systems* acquire knowledge automatically from data. The most frequently used techniques include symbolic, inductive-learning algorithms such as ID3 and multiple-layered, feed-forward neural networks such as backpropagation networks.

Symbolic learning and ID3. In symbolic machine learning, knowledge is represented by symbolic descriptions of the learned concepts. Symbolic machine-learning techniques can be classified based on underlying learning strategies like rote learning, learning by being told, learning by analogy, and so on.[3] Among these techniques, learning from examples — a special case of inductive learning — appears to be the most promising technique for knowledge discovery in real databases.

The ID3 decision-tree building algorithm and its descendants are simple yet powerful algorithms for inductive learning. ID3 takes objects of a known class, described in terms of a fixed collection of properties or attributes, and produces a decision tree that incorporates these attributes and correctly classifies all the given objects. It uses an information-theoretic approach aimed at minimizing the number of tests needed to classify an object, and its output can be summarized in terms of production rules. ID3 has been used successfully for various classification and prediction applications.

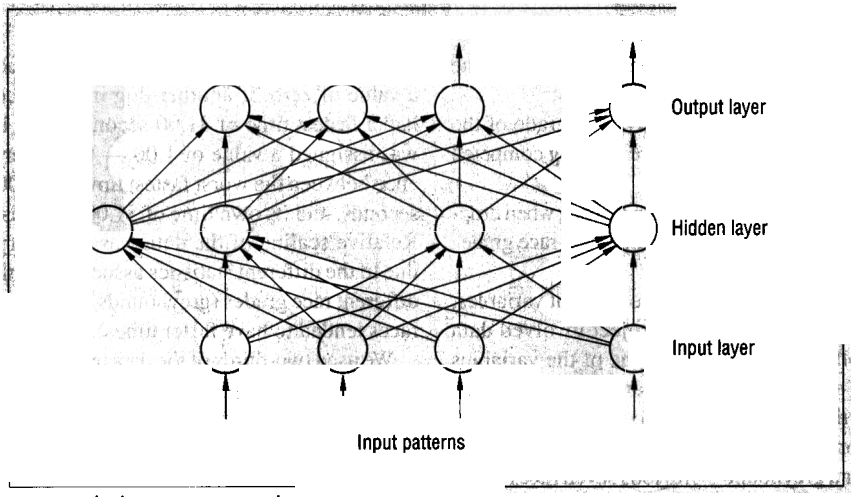


Figure A. A backpropagation network.

Neural networks and backpropagation. The foundation of the neural networks paradigm was laid in the 1950s and has attracted significant attention in the past decade as more powerful hardware and algorithms have been developed. Nearly all such "connectionist" algorithms have a strong learning component. Unlike in symbolic learning, in connectionist learning knowledge is learned and remembered by a network of interconnected neurons, weighted synapses, and threshold logic units. Learning algorithms, such as Delta rule, can be applied to adjust connection weights so that the network can predict or classify unknown examples correctly.

Among the many computational models of neural networks, the backpropagation network has been shown to be theoretically sound[4] and has demonstrated excellent capability for various complex classification and prediction problems. Activation of a backpropagation network flows from the input layer through hidden layers to the output layer (see Figure A). Each unit in a layer is connected in the forward direction to every unit in the next layer. A backpropagation network may contain multiple hidden layers in between. Knowledge of the network is encoded in the (synaptic) weights between units. The activation levels of the units in the output layer determine the output of the whole network. Despite promising results in various applications, the development of backpropagation networks still requires extensive experimentation, parameter selection, and human judgment.

Applications. Several studies have compared the performance of these techniques — as well as some systems that use hybrid representations and learning techniques — for different "toy" and real-life applications. In one study, Mooney and his colleagues found that their overall performance was comparable, but that ID3 was faster than a backpropagation net, while the backpropagation net was more adaptive to noisy data sets.[5] Weiss and Kapouleas suggested using a resampling technique such as leave-one-out for evaluation. Discriminant analysis methods, backpropagation nets, and decision tree-based inductive learning methods were found to achieve comparable performance for several data sets.[6] Fisher and McKusick found that when using batch learning, backpropagation performed as well as ID3, but was more noise-resistant.

References

1. P. Jackson, *Introduction to Expert Systems,* Addison-Wesley, Reading, Mass., 1990.
2. H. Simon, "Artificial Intelligence: Where Has It Been, and Where is it Going?" *IEEE Trans. Knowledge and Data Engineering*, Vol. 3, No. 2, June 1991, pp. 128–136.
3. J.G. Carbonell, R.S. Michalski, and T. M. Mitchell, "An Overview of Machine Learning," in *Machine Learning, an Artificial Intelligence Approach*, R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, eds., Tioga Publishing, Palo Alto, Calif., 1983, pp. 3-23.
4. D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning Internal Representations by Error Propagation," in *Parallel Distributed Processing*, D.E. Rumelhart, J.L. McClelland, and the PDP Research Group, eds., MIT Press, Cambridge, Mass., 1986, pp. 318–362.
5. R. Mooney et al., "An Experimental Comparison of Symbolic and Connectionist Learning Algorithms," in *Proc. 11th Int'l Joint Conf. AI*, Morgan Kaufmann, San Francisco, Calif., 1989, pp. 775–780.
6. S.M. Weiss and I. Kapouleas, "An Empirical Comparison of Pattern Recognition, Neural Nets, and Machine Learning Classification Methods," *Proc. 11th Int'l Joint Conf. AI*, Morgan Kaufmann, San Francisco, Calif., 1989, pp. 781–787.

- *time 3 average:* the average finishing time of the three most recent races;
- *grade average:* the average grade of the seven most recent races the dog competed in; and
- *up grade:* weight given to a dog when dropping down to less competitive race grade.

Once we identified the relevant variables, the next stage of the project involved data collection and entry. Some of the variables or attributes required manipulating the data in the racing programs to make them useful. For instance, we averaged the values listed for a greyhound's previous seven races to obtain the average break position. The finish average also required the mean computation of a variable. Two additional attributes — the time 3 average and the time 7 average — were calculated by computing the mean of the three most recent race times and the seven previous race times, respectively.

For the race grade average, we assigned values to each type of grade, with the most competitive race grade, A, receiving a value of four. B grade races received the value of three, C races were assigned a value of two, and D races received a value of one. All other types of races, such as maiden races or training races, received a value of zero. We then averaged these values to procure the race-grade-average. We also used previous race grades to assign values when a greyhound moved down a race grade category. This was necessary to account for residual effects of racing in a previous more competitive race. The value assigned for this "up grade" attribute depended upon how recently the drop in race grade occurred. If the most recent race was of a higher grade, three points were assigned; two points were given if the more competitive race was two races ago, and so on, until the value of zero was given for a greyhound that had not raced in a more competitive event in the last three races.

Since the dogs compete against each other directly in a win-lose scenario, we manipulated the data so that the values were relatively scaled. This means that after making all calculations (such as averaging a dog's finish times), we then assigned the lowest value in one single race to be zero. The other values were scaled to reflect their difference from that lowest value. For example, a race program entry contains eight pieces of data that represent each of the eight greyhounds' fastest time. The worst fastest time among these dogs, say, 32.00 seconds, was assigned a value of zero. If another dog in that race had a fastest time of 31.00 seconds, then it was assigned a value of 1.00 — the difference between the worst fastest time of 32.00 seconds, and its own time of 31.00 seconds. Relative scaling of the data was necessary due to the different statistics associated with different race grades (greyhounds in grade A races tended to have faster times).

We used two-thirds of the data training and one-third for testing. The training stage consisted of 200 races, for a total of 1600 classified greyhounds. Testing consisted of an additional 100 races, or 800 new cases. We did not use any resampling techniques for testing due to the large testing data set and the computational requirements for evaluating the backpropagation algorithm.

IT TOOK ONLY SEVERAL SECONDS TO BUILD A DECISION TREE FROM 1600 TRAINING CASES. THE RESULT WAS SIMPLE TO UNDERSTAND, AND IT WAS EASY TO TRACE THE CLASSIFICATION DECISIONS.

Testing ID3

ID3 adopts a divide-and-conquer strategy for classification. Its goal is to classify mixed objects into their associated classes, based on the objects' attribute values. In our application, each greyhound can be classified as either a winner or a loser, and described by a set of attributes such as the number of races it has won, its fastest time, and so on. The entropy value of an attribute describes how well that attribute can be used to classify a greyhound as either a winner or a loser — the lower the entropy value, the less uncertainty, and the more useful the attribute.

ID3 is designed to deal with both categorical attributes and continuous values. For categorical attributes, attribute values can be easily enumerated. For continuous values, ID3 performs a sweeping analysis of entropy reduction for all possible partition points (between any two consecutive values) and selects the partition point that reduces entropy the most.[2] In essence, the algorithm performs a binary partition. For our system, we developed an ID3 program that adopted this original design.

We also developed a version of the ID3 algorithm that performed ternary partition for continuous variables. This variant of the ID3 algorithm first sorted the continuous values in ascending order, such as (0.12 winner) (0.15 winner) (0.35 loser) (0.36 loser) (0.45 winner) (0.55 loser) (0.66 winner) (0.70 loser) (0.80 loser) (0.88 loser). It then marked the "clean" classes on the two ends of the sorted list as unique classes: {(0.12 winner) (0.15 winner)} are in the "winner" class, for example, and {(0.70 loser) (0.80 loser) (0.88 loser)} are in the "loser" class. Other values in the middle of the list were mixed and remained to be classified using other attributes.

In our experiment, we found that this algorithm tended to reduce more entropy than the original binary partition algorithm, because the two ends in the three partitions were clean classes and thus did not have any entropy value. This often resulted in a lower overall entropy value after partition than that produced by binary partition, which often generated two large mixed classes. The resulting decision tree was also less complex than that obtained through binary partition — ternary partition produced a simple ternary tree where each level of the tree contained exactly three branches. Binary partition branched out two ways at each level, which resulted in a large and more complex binary tree. However, both algorithms were equally efficient, especially in comparison with the neural network algorithms.

In our implementation (in ANSI C and running on a DECstation 5000/120), it took only several seconds to build a decision tree from 1600 training cases. ID3's result was simple to understand, and it was easy to trace the classification decisions.

The attribute break average occupied the root node. Branches represented the up grade, the show percentage, the fastest time, the grade average, the win percentage, time 7 average, finish average, time 3 average, and place percentage. The decision tree created by ID3 using ternary partition is shown in Figure 2. The first five attributes were useful for deciding first-place winners. Other attributes were helpful for deciding place (second place) and show (third place), which were not considered in our analysis.

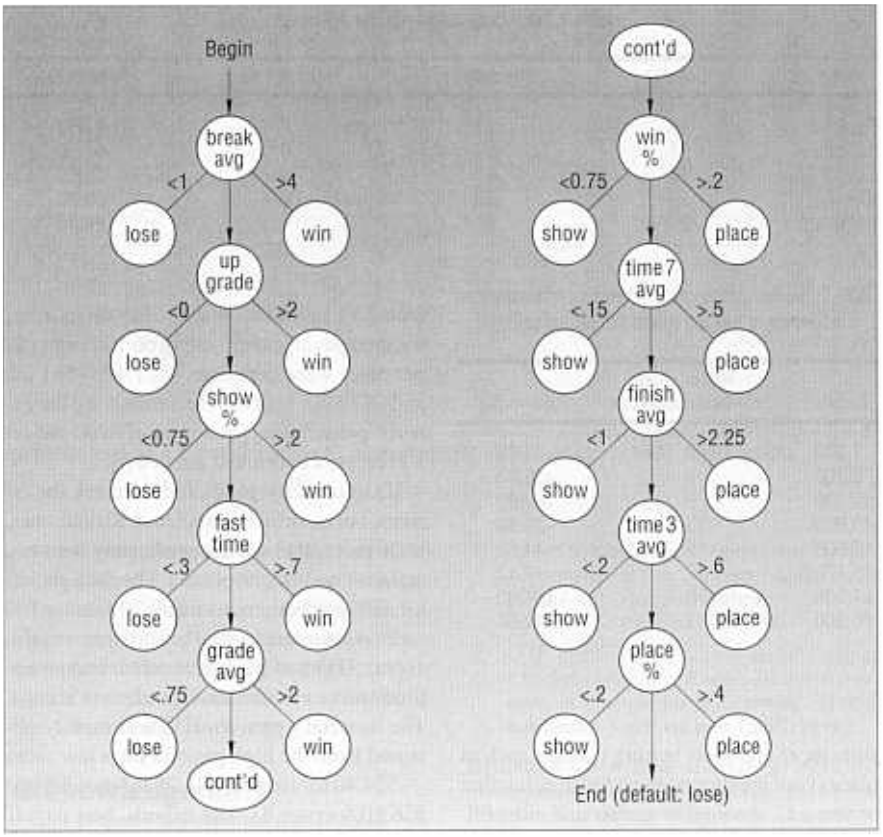


Figure 2. An ID3 decision tree.

Testing the backpropagation network

The number of units in the backpropagation input layer is determined by the number of independent variables in the data set. Similarly, output units represent the application's dependent variables (for prediction problems) or classes (for classification problems). For greyhound racing, the input layer consisted of the various dog racing attributes that could affect the outcome of winning or losing in the output layer. Given a greyhound's past history in racing and other important features, the network should be able to predict the dog's performance in a given race.

The number of hidden units in the hidden layers, as well as the number of hidden layers in a backpropagation network, are much harder to determine. They provide the added power of internal representation that can capture the often nonlinear relationships between the input and output vectors. Although these hidden units and layers play an important role in the network, deciding the exact numbers has never been easy. Researchers generally agree that a larger hidden layer results in a more powerful network, but that too much power obtained from the training data may be undesirable for predicting new, unknown data (a generalization problem).[7] In addition, the computing time needed for a backpropagation network is directly proportional (with a factor often much greater than 1) to the number of hidden units and layers in a network. A complex topology may be undesirable for computational reasons.

The network's *learning rate* (ϵ) determines how far to move toward the gradient of the surface over the weight space defined by an error function. A small ϵ will lead to slower learning, but a large one may cause a move through weight space that "overshoots" the solution vector. Based on reports by other researchers,[8] the range of values we tested for ϵ was between 0.25 and 0.35 in an attempt to avoid local minima. The *momentum factor* (α) tends to keep the weight changes moving in the same direction and allows the algorithm to skip over small local minima. It can also improve the speed of learning. But, as in the case of learning rate, a large momentum factor may cause too much skipping. In past research, momentum factors have been set between 0 and 1 for various applications.

A large number of *epochs* usually brings the network closer to a convergent state and helps improve its recall of the training data set. For different applications and network topologies, the required number of epochs can be very different — ranging from a few hundred to several thousand. However, large epochs require longer computing time and may cause an undesirable *generalization* problem.

If all possible inputs and outputs are shown to a backpropagation network, along with proper topology and training epochs, the network will likely find a set of weights that maps the inputs onto the outputs correctly. For most applications, however, it is impossible to present all possible data. The network needs to *generalize* from the training set to recognize unknown data having similar characteristics. However, there are some potential pitfalls related to overtraining.

During the initial stage of training, performance on the training set typically improves as the network adjusts its weights through backpropagation. Performance on the unknown testing set also improves, although it is never quite as good as on the training set. After a number of epochs, network performance reaches a plateau as the weights shift around, looking for a path to further improvement. Eventually, such a path is found, and performance on the training set improves again. But performance on the test set gets worse. Why? The network has begun to memorize the individual input-output pairs rather than settling for weights that generally describe the mapping for all cases. With thousands of real-valued connection weights in the network, a backpropagation network is theoretically capable of storing an entire training set. This may result in overtraining. Deciding the proper amount of training is therefore crucial to the success of neural network learning.

The results

We developed the programs in ANSI C and ran them on a DECstation 5000/120 (a 25 MIPS machine, running ULTRIX). Evaluating the performance of different techniques was simply a matter of computing the monetary payoffs. For each dog that an algorithm or an expert predicted to be winner, we bet $2.00. If the algorithms predicted more than one winner, we bet on all predicted winners; if no winner was picked, we placed no bet. Using the payoff odds given in the result sheets, we were able to compute the final payoffs after betting on 100 races. For performance comparison, we only considered the payoff for first-place winners; we did not con-

Table 1. Predictions and payoffs for 100 races.

Technique	Correct	Incorrect	Did not bet	Payoffs ($)
Expert 1	19	81	0	–71.40
Expert 2	17	83	0	–61.20
Expert 3	18	82	0	–70.20
ID3	34	50	26	69.20
Backprop.	20	80	0	124.80

Table 2. The Backpropagation network's performance as a function of training epochs (25 hidden units).

Epochs	No. of races predicted correctly	Payoffs ($)
2,500	8	–73.00
5,000	9	–53.60
7,500	9	–67.80
10,000	13	–39.80
12,500	15	44.60
15,000	17	9.40
17,500	20	124.80
20,000	19	44.60

sider more elaborate betting systems such as place (your greyhound must finish either first or second); show (your greyhound must finish either first, second, or third), quiniela (two greyhounds you bet on must finish first and second), and so on. It could be argued that the payoff odds are not necessarily correct, since each bet on a particular dog will change the payoff odds for that dog. We contend that since there are numerous bets on any given dog, the odds would not change significantly with the additional bet on a particular dog.

Table 1 summarizes the predictions by the experts, ID3, and the backpropagation network, and their final payoffs for 100 races.

Among the three track experts, the best predicted the winners in 19 races, but predicted 81 races incorrectly. For all experts, the final payoffs after betting on 100 races ($2 per race) were negative: –$71.40, –$61.20, and –$70.20. That is, when following the experts' predictions, for a total of $200 bet on 100 races, a bettor lost about $70.

ID3 correctly predicted winners for 34 races, but incorrectly predicted 50 outcomes. In 26 races, ID3 did not predict any winners, and thus no bet was placed. The final payoff for ID3 predictions was $69.20 for the 100 races. Compared with the most successful expert, ID3 had a more accurate and sometimes more conservative prediction record. The monetary gain for ID3 was mainly obtained from the high payoffs for a few races — $24.40 for race 31, $41.20 for race 59, and $26.80 for race 83. The experts' best payoff was $11.40. By analyzing the greyhound attributes objectively, ID3 identified long shots and thus realized significant monetary gain. The decision tree created by binary partition resulted in a $69.00 payoff, which was close to the $69.20 payoff achieved by the decision tree created by ternary partition.

Our backpropagation network consisted of a simple three-layer design. We tested hidden units of 15, 20, 25, 30, and 35, respectively. The network of 25 hidden units consistently made better predictions than other topologies. Learning rates varied between 0.2 and 0.4, with a 0.05 increment. Momentum factors were chosen at 0 and 0.1, respectively. Among these parameters, a learning rate of 0.2 and momentum factor of 0.1 were stable and converged quickly. After fixing these parameters, we varied the number of learning epochs from 2,500 to 20,000, with an increment of 2,500 epochs. The performances of these networks are shown in Table 2.

While the prediction accuracy for the testing races continued to improve when we increased the training epochs, the network's predictive power for the 100 testing races peaked at 17,500 epochs. The resulting optimal backpropagation network predicted 20 races correctly, and mispredicted 80 races. Even though the number of correctly predicted races was about the same as for the human experts, their monetary payoffs were very different. The backpropagation network gained $124.80 for the 100 races. As with the ID3 results, the backpropagation algorithm obtained high payoffs for several long shots — $78.00 for race 10, $30.20 for race 37, $30.00 for race 64, and $26.80 for race 83.

A summary table for the one-way analysis of variance (using Minitab) in the payoffs of the different approaches is provided in Figure 3. The means for the five underlying populations (payoffs generated by three experts and the two algorithms) were different at the 10% significance level (N = 100, P = 0.083, where N is the sample size, and P the level of significance). Two sample t-tests revealed that the backpropagation network outperformed the best expert (and the other experts) in monetary payoffs at the 10% significance level (P = 0.084). However, ID3 did not out-per-

Game scenarios

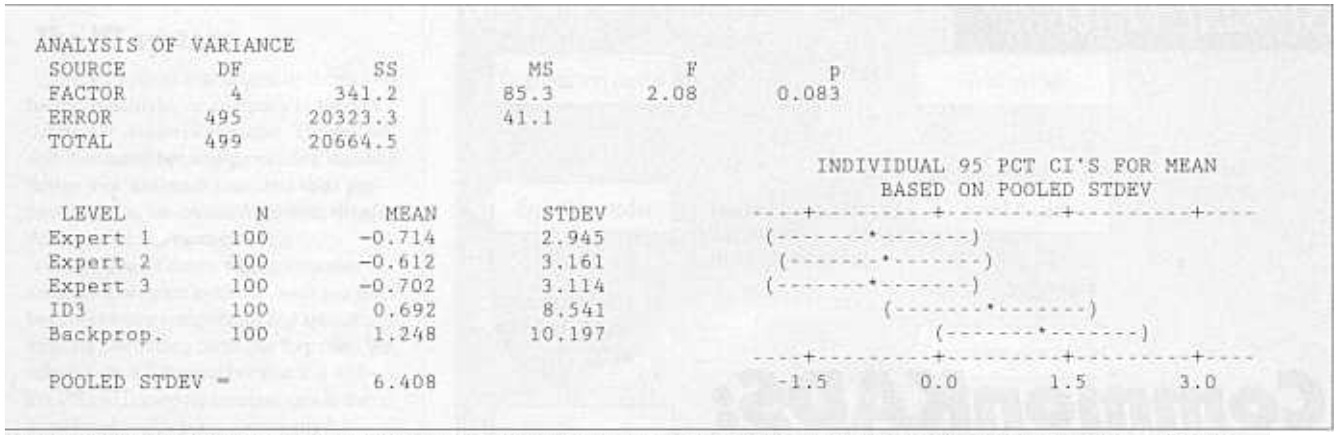
In a game scenario, participants generally compete with each other; the best performer wins, and everyone else loses. Performance, however, is based on a relative scale — each performance is relative to the other participants. A weak performer may still win a game, for example, if the other participants are even weaker.

It is assumed that participants' behaviors or capabilities are relatively stable in the short run, but may improve or degrade in the long run. Furthermore, a game is assumed to be conducted fairly and to be free from other external human factors (such as cheating). Animal racing games like horse racing and greyhound racing exhibit these characteristics. Based on these assumptions, will human experts or machine learning algorithms predict more accurately?

Salzberg was the first to test an algorithmic solution to the horse racing problem,[1] using general human heuristics to formulate hypotheses for the Handicapper prediction program. Handicapper used nine heuristics based on cognitive psychology to predict which horse will win a race. The heuristics were based on what Salzberg called "usefulness." For example, the "unusualness" heuristic focused on a rare or unusual feature of the situation preceding an unexpected event; the Occam's Razor heuristic preferred simple hypotheses over complex ones. In several trials, Handicapper picked wir ners better than human experts. The heuristics and the rationalization process designed for Handicapper provide a computer system with an excellent way to select a good hypothesis from a very large set of candidate hypotheses.

Reference

S. Salzberg, "Pinpointing Good Hypotheses with Heuristics," in *Artificial Intelligence and Statistics*, W.A. Gale, ed., Addison-Wesley, Reading, Mass., 1986, pp. 133-158.



```
ANALYSIS OF VARIANCE
SOURCE        DF          SS          MS          F          p
FACTOR         4       341.2        85.3       2.08      0.083
ERROR        495     20323.3        41.1
TOTAL        499     20664.5

                                                    INDIVIDUAL 95 PCT CI'S FOR MEAN
                                                      BASED ON POOLED STDEV
 LEVEL          N        MEAN       STDEV      ----+---------+---------+---------+----
Expert 1      100      -0.714       2.945      (-------*-------)
Expert 2      100      -0.612       3.161       (-------*-------)
Expert 3      100      -0.702       3.114      (-------*-------)
ID3           100       0.692       8.541                (-------*-------)
Backprop.     100       1.248      10.197                     (-------*-------)
                                              ----+---------+---------+---------+----
POOLED STDEV =          6.408                  -1.5        0.0       1.5       3.0
```

Figure 3. ANOVA analysis for payoffs. N = sample size; P = significance level; DF = degree of freedom; SS = sum of squares; MS = mean square; and F = the F-statistic.

form the experts at a statistically significant level (P = 0.15) and there was no statistically significant difference in payoffs between backpropagation and ID3 (P = 0.68).

In terms of prediction accuracy and monetary payoff, both the backpropagation network and ID3 performed better than human experts. Both algorithms appeared to be more robust than humans in their ability to analyze the large set of racing data objectively and reach unbiased conclusions. This characteristic is particularly evident from both algorithms' convincing prediction of numerous long shots that human experts failed to identify. In comparing the two algorithms, ID3's decision tree output was more understandable than that of backpropagation. In general, ID3 also predicted more conservatively than other approaches. The backpropagation network, on the other hand, was more computationally expensive (and thus very slow), but made excellent predictions of long shots.

IN THIS EXPERIMENT, ESPECIALLY during the early stage, experts' heuristics for refining the initial performance variables were critical to the success of the machine learning algorithms. The 50-plus candidate variables were reduced to 10 of the most relevant parameters, which effectively reduced the problem space for this complex and noisy domain. Some variables were also the results of computations, such as time 3 average, time 7 average, and so on. Also, the machine-learning algorithms that we adopted were "supervised" learning. That is, the learner was told the category membership of environmental observations, and thus the sole learning task was to summarize the commonality among members of the same categories and differences among competing ones.

These algorithms were not adaptive to unsupervised exploration of relevant variables or useful categories in the data using internalized heuristics. Nevertheless, there are other machine-learning algorithms that exhibit unsupervised learning capability.[9] This may prove to be one of the most potentially fruitful research directions for machine learning or knowledge discovery.

Acknowledgments

This project was supported mainly by an NSF grant #IRI-9211418 (1992-1994) and a University of Arizona Summer Research Grant (1992).

References

1. S.M. Weiss and C.A. Kulikowski, *Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Networks, Machine Learning, and Expert Systems,* Morgan Kaufmann, San Francisco, Calif., 1991.
2. J.R. Quinlan, *C4.5: Programs for Machine Learning,* Morgan Kaufmann, San Francisco, Calif., 1993.
3. N.S. Flann and T.G. Dietterich, "Selecting Appropriate Representations for Learning from Examples," in *Proc. Fifth Nat'l Conf. AI (AAAI-86)*, MIT Press, Cambridge, Mass., 1986, pp. 460–466.
4. B.G. Buchanan and T.M. Mitchell, "Model-Directed Learning of Production Rules," in *Pattern-Directed Inference Systems*, D.A. Waterman and F. Hayes-Roth, eds., Academic Press, New York, 1978, pp. 297–312.
5. G. Drastal, G. Czako, and S. Raatz, "Induction in an Abstraction Space: A Form of Constructive Induction," in *Proc. 11th Int'l Joint Conf. AI (IJCAI-89)*, Morgan Kaufmann, San Francisco, Calif., 1989, pp. 708–712.
6. S. Salzberg, "Pinpointing Good Hypotheses with Heuristics," in *Artificial Intelligence and Statistics*, W.A. Gale, ed., Addison-Wesley, Reading, Mass., 1986, pp. 133-158.
7. D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning Internal Representations by Error Propagation," in *Parallel Distributed Processing*, D.E. Rumelhart, J.L. McClelland, and the PDP Research Group, eds., MIT Press, Cambridge, Mass., 1986, pp. 318–362.
8. R. Mooney et al., "An Experimental Comparison of Symbolic and Connectionist Learning Algorithms," in *Proc. 11th Int'l Joint Conf. AI (IJCAI-89)*, Morgan Kaufmann, San Francisco, Calif., 1989, pp. 775–780.
9. D.H. Fisher, M.J. Pazzani, and P. Langley, *Concept Formation: Knowledge and Experience in Unsupervised Learning,* Morgan Kaufmann, San Francisco, Calif., 1991.

Hsinchun Chen is an assistant professor of management information systems at the Karl Eller Graduate School of Business, University of Arizona. His research interests include CSCW, human-computer interactions, text-based information management and retrieval, Internet resource discovery, digital libraries, knowledge acquisition and knowledge discovery, machine learning, and neural network modeling and classification. He received an NSF Research Initiation Award in 1992, and the Hawaii International Conference on System Sciences Best Paper Award and AT&T Foundation Award in Science and Engineering in 1994. He received a PhD in information systems from New York University in 1989. Chen can be reached at the MIS Department, University of Arizona, McClelland Hall 430Z, Tucson, AZ 85721; Internet: hchen@bpa.arizona.edu.

Peter Buntin Rinde is currently employed with Ernst & Young Information Technology Management Consulting. He received his BS in management information systems from the University of Arizona in 1994. He was the recipient of the Outstanding BPA Senior Award and 1994 UA Parents' Association Award.

Linlin She is currently employed with the Biosphere-2 in Tucson, Arizona. She received her MS degree in management information systems from the University of Arizona in 1994.

Siunie Sutjahjo is currently employed with the Ventana Corporation in Tucson, Arizona. She received her MS in management information systems from the University of Arizona in 1993.

Chris Sommer is currently employed with Ernst & Young Information Technology Management Consulting. He received his BS degree in management information systems from the University of Arizona in 1993.

Daryl Neely is currently employed with West Publishing. He received his BS in management information systems from the University of Arizona in 1993.