Neurogammon: A Neural-Network Backgammon Program

Gerald Tesauro
IBM Thomas J. Watson Research Center
PO Box 704, Yorktown Heights, NY 10598 USA

Abstract

Neurogammon 1.0 is a complete backgammon program which uses multi-layer neural networks to make move decisions and doubling decisions. The networks were trained by back-propagation on large expert data sets. Neurogammon appears to play backgammon at a substantially higher level than conventional programs. At the recently held First Computer Olympiad in London, Neurogammon won the backgammon competition with a perfect record of 5 wins and no losses, thereby becoming the first learning program ever to win any tournament.

1. Introduction

Back-propagation [12, 4, 6, 7] has now been demonstrated to be a useful learning procedure for many classes of real-world practical applications. In this paper, a research project is presented which was designed to examine whether back-propagation might be useful in higher-level tasks which are currently tackled by expert systems and knowledge engineering approaches. This project involved teaching networks to play the game of backgammon from examples of human expert play. Games in general are a useful testing ground for learning algorithms because: (a) expert-level play can be of tremendous complexity, and (b) problem inputs and performance measures are clear-cut and well-defined. The game of backgammon in particular was selected because of the predominance of judgement based on static pattern recognition, as opposed to explicit look-ahead or tree-search computations. (Readers unfamiliar with the game are referred to [5].)

Previous research in backgammon learning networks was described in [11, 9, 10]. Neurogammon 1.0 represents the culmination of this research in the form of a fully functioning program. Neurogammon contains one network for making doubling decisions, and a set of six networks for move decisions in different phases of the game. Each network has a standard fully-connected feed-forward architecture with a single hidden layer, and each was trained by back-propagation on an expert data set. Training of each network proceeded until maximum generalization performance was obtained, as measured by performance on a set of test positions not used in training. The networks

appear to have learned a great deal of expert backgammon knowledge, and the resulting program plays at a substantially higher level than conventional backgammon programs, as demonstrated by its decisive 5-0 win at the First Computer Olympiad.

In the following, we examine in detail some of the key ingredients that went into the development of Neurogammon. In section 2, a "comparison paradigm" for learning expert preferences is introduced, in which the network is told that the move preferred by the expert in a given situation should score higher than each of the other legal moves not selected by the expert. The comparison training paradigm was probably the most crucial ingredient in the achievement of Neurogammon's performance level, and in its victory at the Computer Olympiad.

Section 3 describes a separate network used to make doubling decisions, which was trained on a separate data base. Each position in the data base was ranked according to a crude nine-point scale of doubling strength in money games. Doubling decisions were made by comparing a win probability estimate, based on network judgement and on racing considerations, with a theoretically optimal doubling point depending on the match score. The whole approach to doubling can only described as a quick and dirty "hack" done in a limited time before the Computer Olympiad, yet the consensus of expert opinion at the Olympiad was that Neurogammon's doubling algorithm was probably the strongest part of the program. Thus this was an instance where supervised learning lived up to the (sometimes naive) expectation that it can be used to quickly develop a high-performance system with little effort on the part of the human programmer.

In section 4, we examine Neurogammon's performance at the Computer Olympiad. The concluding section discusses the significance of the work, and some future directions likely to yield further progress.

2. Comparison training of expert move preferences

In [11], networks were trained to score single moves in isolation, based on examples of human expert scores of moves. The input was a description of an initial board position and a transition to a final position, and the desired output was an expert's judgement of the quality of the transition from initial to final position. The expert judgement recorded in the training data ranged from +100 for the best possible move to -100 for the worst possible move in a given situation. This approach is hereafter referred to as the "relative score" paradigm (RSP).

The RSP approach gave surprisingly good average performance, but suffered from a number of fundamental limitations. The training data itself is problematic, due to the arbitrary normalization, incompleteness (the human expert cannot comment on every possible legal move), difficulty of scoring bad moves (necessary for the network, but unusual for the human expert), and possible human errors. Furthermore, in this scheme, a sophisticated scheme for coding the transition from initial to final board positions was

Type of test set	RSP net (651-12-1)	CP net (289-12-1)
bearoff	.82	.83
bearin	.49	.63
opp. bearoff	.56	.59
opp. bearin	.49	.61
other	.55	.69

Table 1: Performance of nets of indicated sizes trained in the relative score paradigm (RSP) and in the comparison paradigm (CP) on respective test sets, as measured by fraction of positions for which net agrees with human expert choice of best move.

required, using human expert concepts such as "slotting" and "stripping" points. More fundamentally, however, in the RSP approach, the network does not learn an intrinsic understanding of the position itself, but instead only learns what kinds of transitions between positions are desirable. Furthermore, the network can only judge one move at a time in isolation, without any knowledge of what other alternative are available. The latter problem is the most serious, since from the human expert point of view, the "goodness" or "badness" of a given move often depends on what other moves are available.

In [10], an alternative training paradigm called the "comparison paradigm" (CP) was reported. In this paradigm, the input to the network is two of the set of possible final board states, and the desired output represents the human expert's judgement as to which of these two states is better. This cures the most fundamental problem of the RSP method, as the network always learns to judge a particular move in the context of the other moves that are available. The comparison paradigm also has a number of other advantages. There are no problems due to the expert entering an incorrect score, no problems due to arbitrary score normalization, and no problems with uncommented moves. Also, one can present only final board positions and thus does not need a sophisticated scheme to encode the transitions from initial to final positions. Finally, it was shown in [10] that symmetry and transitivity of comparisons can be exactly enforced by a constrained architecture in which the left half of the network only looks at board position a, and the right half only looks at board position b. The constrained architecture forces each half-network to implement a real-valued absolute evaluation function of a single final board position. The network thus does develop an intrinsic understanding of the position in this training methodology.

Test-set performance of RSP nets and CP nets are compared in table 1. (The nets were trained according to the methodology of [10].) The CP nets have a much smaller and simpler input coding scheme, and have significantly fewer connections. The CP nets also have much stronger game performance. In particular, worst-case performance is dramatically improved by comparison training.

3. Neurogammon's doubling algorithm

Neurogammon 1.0 has a separate network for making doubling decisions which was trained on a separate data set. The doubling data set contained about 3000 positions taken from 64 expert games described in [3]. Each position was classified according to a crude nine-point ranking scale of doubling strength, guided by the commentary in [3]. Of these positions, 225 were set aside at random to be used as test data; the remaining positions were used as training data.

The networks trained on this data set had 9 output units corresponding to the 9 possible categories. The input representation used a total of 243 input units. This representation scheme was a simplification of the 289 unit scheme for the move selection networks.

Interpreting the network's output and measuring its performance on test data after training is a somewhat subtle matter. It was decided for simplicity to just sum up all nine output values, and interpret that as the network's score for a particular position. An error measure can then be based on the difference σ between this score and the recorded teacher score. If $\sigma < 0.5$, it indicates that the network has scored the position essentially correctly, while $\sigma > 1.5$ indicates a significant deviation from the correct answer. The best performance according this measure was obtained with a 243-24-9 network, which scored 61% of the test positions within 0.5 of the correct answer, and only 6% off by more than 1.5 from the correct answer.

In order to make optimal doubling decisions according to established theory [2, 13], one needs an estimate at any point during the game of the probability of each side winning the game. This was done in an admittedly crude way by dividing the total network score by 10; this gives a number p_n which may be interpreted as the network's estimate of White's probability of winning the game. One can also compute by standard techniques [1, 2, 13] a highly accurate estimate of the win probability based solely on the state of the race; this gives another estimate of win probability p_r. Neurogammon 1.0 makes doubling decisions by comparing an interpolated probability $\bar{p}$, lying between p_n and p_r, with the theoretical optimal doubling point p_0. The interpolation depends on the degree of contact, varying from $\bar{p} = p_n$ for fully engaged positions to $\bar{p} = p_r$ for fully disengaged positions. This interpolation scheme is *ad hoc*, but it does have the effect of reducing or eliminating the network's most serious and most frequent errors.

4. Results of the Computer Olympiad

The First Computer Olympiad, which took place Aug. 9-15, 1989, at the Park Lane Hotel in London, was a unique event in the history of computer games. Organized by computer chess expert David Levy, the Olympiad featured competitions in 15 different games, open to any computer program. In some of these games, such as chess, Go, and Othello, computer competitions have been held regularly for many years, but for several other games such as

both machine learning and computer games research. For machine learning researchers (and for neural networks researchers in particular), it provides a convincing practical demonstration that network learning algorithms can be useful in tackling hard computational tasks. It should be emphasized that tournament performance is a severe test of the networks' ability to extract good generalizations from the training data, rather than merely memorize specific individual positions. Apart from the first couple of moves at the start of each game, it is highly unlikely that any of the positions encountered by Neurogammon at the Computer Olympiad were contained in the training set.

The significance to computer games research is that it is apparently the first time in the history of computer games research that a learning program has ever won an official tournament in an established game of skill. Machine learning procedures have been studied in computer games environments for many years, but learning programs typically have not achieved the levels of performance necessary to win in tournament competition. Neurogammon's achievement suggests that learning procedures may be more widely useful in other complex games such as chess and Go.

While Neurogammon can only be described as a strong program, it clearly has to improve in a number of ways before it can be claimed to perform at human expert level. One obvious improvement would be to train on a larger and more varied expert data set containing data from several experts exhibiting a variety of styles of play. The move-selection networks in particular are dangerously inbred, having been trained only on data from one individual. Further significant improvement might be obtained by adding a small amount of look-ahead, perhaps by going to a 3-ply search.

Another way in which learning could be improved is by incorporating special knowledge about symmetry or topology of the problem either into the data representation scheme or the network architecture. For example, one knows that the evaluation function should have an exact symmetry in that, if the configuration of the Black and White pieces are swapped, and the player to move is reversed, then the evaluation should exactly invert, i.e., the score $S \rightarrow -S$. Also, one knows that backgammon has a one-dimensional spatial structure with approximate translation invariance. Explicitly building these symmetry principles into the learning system could enable it to extract better generalizations from a limited amount of training data.

Another direction which seems very promising is to abandon supervised learning altogether, and instead to learn by playing out a sequence of moves against an opponent and observing the outcome. Such a learning system could learn on its own, without the aid of an intelligent teacher, and in principle could continue to improve until it surpassed even the best human experts. Connectionist algorithms such as Sutton's Temporal-Difference algorithm [8] could be used in such an approach. It is not known whether such algorithms are efficient enough to tackle a problem of the scale and complexity of backgammon. However, it seems most likely that they could be used to learn a highly accurate end-game evaluation function. Since the late (but

backgammon, it was the first organized tournament for computer programs.

Neurogammon faced five opponents in the backgammon competition at the Olympiad: three commercial programs (Video Gammon [USA], Saitek Backgammon [Netherlands], and Mephisto Backgammon [West Germany]) and two non-commercial programs (Backbrain [Sweden] and AI Backgammon [USA]). Hans Berliner's BKG program was not entered in the competition. All five programs apparently used single-ply search algorithms with conventional hand-crafted evaluation functions. In matches to 11 points, Neurogammon defeated Video Gammon by 12-7, Mephisto by 12-5, Saitek by 12-9, Backbrain by 11-4, and AI Backgammon by 16-1. With five match victories and no losses, Neurogammon convincingly won the gold medal for backgammon at the Computer Olympiad. Neurogammon also played four unofficial matches to small numbers of points against intermediate-level humans, and won three out of the four matches. Finally, in an official challenge match on the last day of the Olympiad, Neurogammon put up a good fight but lost to a human expert, Ossi Weiner of West Germany, by 2-7. Weiner was clearly the better player, but the difference in skill levels was not as great as suggested by the score. Weiner said that he was impressed by how much the program played like a human, how rarely it made mistakes, and that he had to play extremely carefully in order to beat it.

To summarize, there were two main ingredients behind Neurogammon's success at the Computer Olympiad. First, its doubling algorithm turned in a very solid performance: although its play was not technically perfect, it did not make a single decision that would cause an expert to complain during the entire tournament. Second, in comparison to its opponents, its move selection was very aggressive and much more in the style of a human expert. Several expert strategies and tactics used by Neurogammon (e.g., slotting, blitz attacks, and back games, to name some of the most prominent) appeared to be missing completely from the other programs.

It was also interesting to observe the differences in the errors made by Neurogammon in comparison to those of its opponents. Neurogammon's errors were mainly tactical in nature; its positional judgement was usually quite good. The other programs, however, usually made tactically acceptable plays (given their lack of knowledge of expert tactics), but often displayed bad positional judgement. This leads one to speculate that a facility to search beyond one ply would be more profitably added to Neurogammon than to a conventional program. A small amount of look-ahead can cure immediate tactical problems, but is unlikely to make a significant improvement in a program with fundamentally flawed positional judgement. Other possible improvements to Neurogammon are discussed below.

5. Conclusions

Neurogammon 1.0 demonstrates that a great deal of human expert knowledge in a complex domain can be learned by multilayer neural networks from large expert data sets. Its victory at the Computer Olympiad has significance for

still engaged) stages of the game are precisely where Neurogammon currently goes most seriously wrong, learning from outcome could at least provide a useful supplement to what can be achieved with supervised learning, and at most could be used to learn the entire game to better than human world champion level.

References

[1] H. Berliner, "Experiences in evaluation with BKG– a program that plays backgammon," *Proc. of IJCAI*, 428–433 (1977).

[2] E. B. Keeler and J. Spencer, "Optimal doubling in backgammon." *Operations Research* **23**, 1063–1071 (1975).

[3] B. Kennedy and C. Papazian, *Backgammon Master Games.* Palo Alto: Shima Publishing (1981).

[4] Y. Le Cun, "A learning procedure for asymmetric network." *Proceedings of Cognitiva (Paris)* **85**, 599–604 (1985).

[5] P. Magriel, *Backgammon*, Times Books, New York (1976).

[6] D. B. Parker, "Learning-logic." MIT Center for Computational Research in Economics and Management Science Tech. Report TR-47 (1985).

[7] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning representations by back-propagating errors." *Nature* **323**, 533–536 (1986).

[8] R. S. Sutton, "Learning to predict by the methods of temporal differences." *Machine Learning* **3**, 9-44 (1988).

[9] G. Tesauro, "Neural network defeats creator in backgammon match." Univ. of Illinois, Center for Complex Systems Technical Report CCSR-88-6 (1988).

[10] G. Tesauro, "Connectionist learning of expert preferences by comparison training." In: D. Touretzky, ed., *Advances in Neural Information Processing* **1**, 99-106, Morgan Kaufman Publishers (1989).

[11] G. Tesauro and T. J. Sejnowski, "A parallel network that learns to play backgammon," *Artificial Intelligence* **39**, 357-390 (1989).

[12] P. Werbos, Ph. D. Thesis, Harvard University (1974).

[13] N. Zadeh and G. Kobliska, "On optimal doubling in backgammon." *Management Science* **23**, 853–858 (1977).