Some Studies in Machine Learning Using the Game of Checkers: A Critique

Adam Nelson
Machine Learning - WVU - Fall '09

September 9, 2009

Contents

1 Summary

When computers were essentially still in their infancy, using lights, switches and magnetic tape, already there were individuals attempting to tackle the concept of *machine learning*. A.L. Samuel attempted this through the notion of gameplay. In the paper, Samuel states that games are ideal for the development of learning procedures because "a game provides a convenient vehicle for such study as contrasted with a problem taken from life, since many of the complications of detail are removed." As a result, Samuel focuses entirely on the game of checkers to test developing these learning procedures as opposed to more complex games, such as chess.

On an IBM 704, Samuel developes a checker-playing program that can either play against itself, or with humans. The typical game uses very specific techniques developed just for this program. However, a typical game is played as follows:

- A look-ahead procedure is carried several moves in advance

The *look-ahead* procedure carries out in much the same way that a human checkers player would; by dynamically looking into the next few possible moves by each player, and determining the *best* move to make according to this intuition.

Samuel states that moves are stored in a "tree", in which the root note denotes the initial segment of the game, and all proceeding branches and leaves represent a move made by either party. Each move is known as a *ply*. For instance, a ply of degree 2 would indicate a move by the computer, and a resulting move by its opponent.

Thus a "backtrace" is done on this tree, to determine the highest scoring move to make.

- Board states are then "scored" and selected based on their quality

In order to determine if a board configuration is good or bad, board positions are rated based on a linear polynomial. The scoring polynomial uses 38 (22 kept as reserve) terms to derive a value for each configuration, and then records these periodically on magnetic tape using clever cataloging and culling methods.

- Repeat the above steps until either the program or its opponent wins a game

In order to convey the playing quality of the program, Samuel discussed two tests that were run, each yielding interesting results. The first test concluded that while the program's opening and end games were good, middle games were lacking, and that even though more and more games were being played, it was obvious that the learning of the program was "shifting" and never converging. However, the overall playing quality of the program was that of "a better-than-average player." The second test was conducted after fixing numerous problematic features of the program, such as changing the correlation coefficients less drastically, reducing the frequency with which terms were introduced into the scoring polynomial, etc. While the second test didn't yield a checkers champion playing ability, it did show that by altering these variables, the program's learning was much more stable, and produced fewer losses.

The experiments conducted by Samuel showed that it was in fact possible to "devise learning schemes which will greatly outperform an average person and that such learning schemes may eventually be economically feasible as applied to real-life problems." It is evident that in the very early years of machine learning research, the idea of constructing paradigms in which a machine could improve through experience was profound, and Samuel did his part by instructing a program to learn to play a better-than-average game of checkers against a better-than-average checker-playing human, and provide a challenge.

2 Limitations and Benefits

2.1 Limitations

While the paper and its concepts were ground-breaking for when it was written 50 years ago, their limitations are fairly obvious. The most striking shortcoming noticable is the implementation of shortcuts in order to save on machine resources, such as processing time and especially storage. As an example to save on exection time and memory, the "looking-ahead" operation performed in order to select the best moves is restricted to a ply of 20, where any value greater than 20 may provide better results. To reduce storage space and resources in obtaining information that is stored, techniques were developed to decrease the amount of information needing saved to tape. For instance, an age score is carried with each board position's score. The age score is used to keep track of the frequency of use of board positions by assigning an arbitrary age to each new board position, and then whenever the board position is referenced, its age is divided by 2. At an average of every 20 moves, this age is incremented, and when it reaches an arbitrary maximum the board state is expunged. Even though a board state is rarely referenced, it still remains pertinent and could possibly result in a victorious game that would otherwise be lost. Samuel refers to this as a *forgetting* “feature” that was apparently (as a last resort) “adopted on the basis of reflections as to the frailty of human memories.” This can be seen as a definite flaw, since machine learning not only takes advantage of extremely fast computations, but also of massive amounts of stored knowledge.

2.2 Benefits

There is no doubt that when the paper was written, Samuel paved the way for more advanced techniques. His dedication to machine learning and game theory gave an impressive and applicable outlook on how computers can be made to improve through experience and exposure to practice. The paradigms given can be duplicated and expanded in order to construct other game-playing learning mechanisms that can be applied to game and non-game purposes alike. With minimal resources, it becomes increasingly difficult to build a learning program. Even given serious computer restrictions, Samuel showed that it was possible to develop methods of implementing machine learning through checkers.

3 Possible Extensions

While Samuel produced remarkable experiments in the dawn of machine learning, there are features that one might suggest for extending the techniques in the paper. Some of the features are as follows:

- Increase Experience (Number of Games): Even though it was shown that the system's learning began to stabilize after 31 or 32 games, this stability could improve further after thousands of games. If so, with more practice, the system may in fact be able to play checkers at a level greater than a “better-than-average” player.

- Increase Ply: At a maximum ply of 20, the system may not perform ample “looks-ahead” in order to report on the best possible move. Thus, increasing this ply could yield better moves from the program. Note that Samuel's system of dynamically choosing a ply can still be applied by simply increasing its value.

- Increase Saved Board States: Samuel states in the paper that “At the present time the memory tape contains something over 53,000 board positions (averaging 3.8 words each) which have been selected from a much larger number of positions by means of the culling techniques described.” He then goes on to say “at least 20 times the present number of board positions would be needed to improve the midgame play significantly” By increasing the number of saved board states, and thereby improving the system’s midgame play (which was reported as the most lacking), the system’s ability to play checkers should be increased as well.

- Learning Weights: In the paper, Samuel reports that since kings are more valuable than standard pieces, they receive a higher weight than non-king pieces. This is to be assumed as per the rules of checkers. However, these weights are arbitrarily chosen, and could be optimally chosen based on *learned* weights via additional features. These learned weights could provide a more accurate assignment of importance on kings in varying board positions and thus possibly better moves.

By implementing all of the suggestions that I have given above should improve performance of the system based on the paper’s description of the program’s shortcomings. However, with that in mind it is also necessary to include that most of the possible extensions given would not have been possible on 1950s computers due to memory, storage, and processing constraints. For instance, massive increases in ply as well as saved board states were not feasible options when the experiments for this paper were conducted. Thus, through developing clever algorithms and strategies to deal with these hardware issues of the time, one cannot deny that Samuel used his resources to a potential maximum in order to create a program that *learns*.