A Tic Tac Toe Learning Machine Featuring the Automatic Generation and Application of Heuristics

Jacob Peck
Computer Science Department
State University of New York at Oswego

Abstract

Description and analysis of a learning machine approach to tic tac toe game playing. Includes related research and discussion of knowledge representation, along with results, abstractions made within the program, ideas for future work, and discussion.

Keywords: Machine learning; rule bases.

Introduction

Tic tac toe is a rather simple game. This simplicity makes it ideal for an exercise in machine learning. Through the process of developing this heuristic learning machine agent, an understanding and appreciation of machine learning and heuristics may be gained, and further avenues of research opened. This approach takes the form of a heuristic learning machine player agent, allowing the use of heuristics to fine tune the behavior of the machine as it learns more winning combinations.

Background

Machine learning, the idea of a machine doing better at a task due to previous experience, has a rich history. In 1952-1962, Arthur Samuel developed a checkers playing program that was sufficiently strong enough to challenge a world champion. The strength of this program came from Samuel's use of machine learning techniques (Buchanan). The application of machine learning techniques to tic tac toe playing seems natural.

Related Research

Much research has been done in the field of game playing with machine learning, including the example from Arthur Samuel above. Another interesting example comes in the form of a machine learning approach to the game Mastermind, an M.S. thesis work by Angie Hugeback (Hugeback, 2005). Other game playing machine learning agents include chess playing agents (Gleich, 2003).

Approach

The approach taken is one of a heuristic learning machine agent. This agent plays games against another agent, and keeps track of the move list of every winning game. This allows the agent to improve its play against others based upon previous experience. The way this system works is that if the current play matches the beginning of a previous winning game, it plays the next move in the series. If, at the end of the game, the heuristic learning machine player is the winner and the rule does not already exist in the database, it adds the move list to its list of heuristics.

Knowledge Representation

IT has been repeatedly observed within the AI community that appropriate representation is key to problem solving (Simon, 1982; Winston, 1985). For a problem like tic tac toe, the knowledge representation that seems most fit is simply a move list, using notation such as `(NW S E C NE SW N W SE)` to represent an entire game. The first move in the list is for the player denoted by `X`, and alternates between `O` and `X` from that point forward. The tokens that make up the list is consistent with the following visualization:

```
NW | N  | NE
---+----+---
W  | C  | E
---+----+---
SW | S  | SE
```

Program Abstractions

This program abstracts away the concept of the game board in favor of a list-based representation of the entire game. In terms of rules, the heuristics that the machine learns are simply a list of winning sets of moves for entire games, and also that the heuristic machine learning player is always denoted as the `X` (first) player. The reasons for these abstractions are twofold: ease of representation to the agent, and reduction of complexity for the programmer.

Results

Through demos before, during, and after sufficient training, the agent has shown greatly improved performance. After running several trials (consisting of 500 games pre-learning, a 5000 game learning session, and 500 games post-learning), the vast majority of the trials pointed to increased success at winning against a random machine agent, sometimes in excess of a 17% increase.

Discussion

The heuristic learning machine player did fairly well, given its constraints. In showing improvement over a random machine agent, it has demonstrated that it is, in fact, learning. While the results are not overwhelming, the validity of the idea and the approach is firmly grounded in previous works, and shows repeatable results when the same methods are applied to different problem domains. While is it possible that this machine would eventually (given persistence and enough learning sessions) become unbeatable, that was not the goal of this research. Rather, the endeavor was a success simply because it showed improvement.

Future Work

A future iteration of this heuristic learning machine agent might include persistence, in order to maintain its rule sets between sessions. This would allow the machine to grow even stronger over time, as opposed to working in a vacuum every time it is invoked. Another possibility for this machine is to collect not only rules for winning games, but also those that is has lost, in order to avoid those moves. Should there be a conflict in these two sets of rules, the machine might play towards a draw, for example, or make a random move. In any case, there is plenty of room for improvement.

Conclusion

Like Samuel's checkers work before it, this heuristic machine learning approach to tic tac toe showed improved proficiency following periods of experience gaining (Samuel, 1967). While the results weren't astounding by any means, the machine did show progress and learning, proving that the self-generation of heuristics via a simple fitness metric is a viable option for a heuristic machine learning game-playing algorithm.

References

Buchanan, Bruce B. *Timeline: A Brief History of Artificial Intelligence*. Web: http://www.aaai.org/AITopics/pmwiki/pmwiki.php/AITopics/BriefHistory

Gleich, David. (2003). *Machine Learning in Computer Chess: Genetic Programming and KRK*. Web: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1.3082&rep=rep1&type=pdf

Hugeback, Angie. (2005). *Mastermind by Importance Sampling and Metropolis-Hastings*. Web: http://www.hugeback.com/angie/projects/Mastermind/Online/thesis.pdf

Samuel, Arthur. (1967). *Some Studies in Machine Learning Using the Game of Checkers. II—Recent* Progress. Web: https://researcher.ibm.com/researcher/files/us-beygel/samuel-checkers.pdf

Simon, H. (1982). *Sciences of the artificial*. Cambridge, MA: Addison Wesley.

Winston, P. H. (1985). *Artificial intelligence*. Cambridge, MA: Addison Wesley.