COMS4030A: Adaptive Computation and Machine Learning Term Project

An Investigation into AI Agents for Tic-Tac-Toe

Jason Chalom - 711985

Background Knowledge

Game Problems

- Games provide fascinating problems for computers to solve
- They can produce intractably large game search spaces so more clever ways of approximating the right solution need to be found
- An issue with many machine learning algorithms is that for even the slightest domain change (i.e. size) training has to be redone

Background

The Game of Tic-Tac-Toe

- Simple and popular game
- Has been analytically solved
- Has a perfect move
- The first player should always win or draw a match
- The second player should always draw a match

Technology Used

Technology

- Everything was programmed in Python
- I extensively made use of numpy

Strategies Used

Data sets for Algorithms

- I generated a million game action dataset for each board size
- They were generated from playing two random agents against each other and only keeping games where the player being scrutinized wins or draws the match

Strategies

Neural Network for 3x3 Board

- Accepts 9 inputs (one for each element of board)
- Has a hidden layer of 9 nodes
- Output layer returns a threshold value. 9 Output nodes where one of the has the value 1 (which is the move) and the rest are zero

Neural Network for 5x5 Board

- Accepts 25 inputs (one for each element of board)
- Has a hidden layer of 25 nodes
- Output layer returns a threshold value. 25 Output nodes where one of the has the value 1 (which is the move) and the rest are zero

Strategies

Attempt to make an AI which can play any sized board

- I attempted to use principal components analysis to make any sized board being played have the same inputs and output for a neural network.
- I failed because PCA does not lend itself to scaling up components. These approximations mean nothing. Also the dataset generated was not very good because I relied on random games for data. There was not enough control.

Strategies

Attempt to make an AI which can play any sized board

- I moved on to trying to use an auto-encoder to produce an intermediary set of inputs for a general neural network agent.
- Here each board would need its own auto encoder
- I failed again because I was trying to use the same bad dataset I had generated. I should have generated a new dataset which was just every state the board can be in. I could then train the auto-encoder better. I do believe this method will still produce terrible results because the accuracy of the compression was 70% which was not good enough.

Results

Results

3x3 Neural Network

- Over an averaged set of games versus the random agent the network performed terribly.
- It lost 59% of the time
- Drew 40% of the time
- And won only 1% of the time

- Two reasons this happened:
- The dataset used was actually not good as it generated based off random games where the move may have been bad but that player still won.
- The network structure was not complex enough to approximate the optimal move function

Results

5x5 Neural Network

- Over an averaged set of games versus the random agent the network performed terribly.
- It lost 65% of the time
- Drew 34% of the time
- And won only 1% of the time

- Two reasons this happened:
- The dataset used was actually not good as it generated based off random games where the move may have been bad but that player still won.
- The network structure was not complex enough to approximate the optimal move function – the 5x5 board probably needs eve more data and a larger network than the 3x3 due to have much more decision states.

Problems Encountered

Problems

- Underestimated the amount of time required
- PCA is not a good fit for attempting to standardize board size
- Dataset generated was terrible
- Training time was extremely long
- Assumptions of how to train the neural network and other methods was wrong

Conclusions

Conclusions

- This project was a failure. I should have used a better data-set and not attempted to generalise the board size. The agents produced were not good at playing the game, rather they were worse than random.
- A more complex network structure and a better data-set would have produced far better results.

Works Cited

- Sager, J. (2015, October 1). Free PowerPoint Template. Retrieved 2016, from http://sage-fox.com/
- Russell, Stuart and Norvig, Peter,Artificial Intelligence: A ModernApproach, 3rd ed. Pearson, 2009.
- Mitchell, T.M. Machine Learning,McGraw-Hill, 1997.