The Applicability of Machine Learning Concepts to Game Artificial Intelligence

Garrett Gutierrez (garrettg@stanford.edu) and Qiujiang Jin (qiujiang@stanford.edu)

Stanford University, CS229, 13 December 2016

Summary

The motivation of our project was to assess the applicability of Machine Learning concepts to General Game Playing. The applicability of linear regression and reinforcement learning were analyzed. An analysis of linear regression revealed the potential for a greedy game playing algorithm that performs better than Monte-Carlo Tree Search (our baseline for measuring performance) under certain conditions.

General Game Playing (GGP)

Rules of **General Game Playing** regarding artificial players:

General Game Players know nothing about the rules of the game or the amount of time they will be given before the game begins.

Passed rules of the game at the beginning of the game.

Player has an amount of time, referred to as the **start clock**, at the beginning of the game before any turns have been taken to do any calculations.

Player has an amount of time, referred to as the **turn clock**, during each turn to decide what action to take. Must decide an action before the end of the turn clock.

All players take simultaneous turns.

Data

There is no precompiled game state data because, by the rules of GGP, the player is permitted no knowledge of the game before the game begins. Games used are those defined at **http://gamemaster.stanford.edu**.

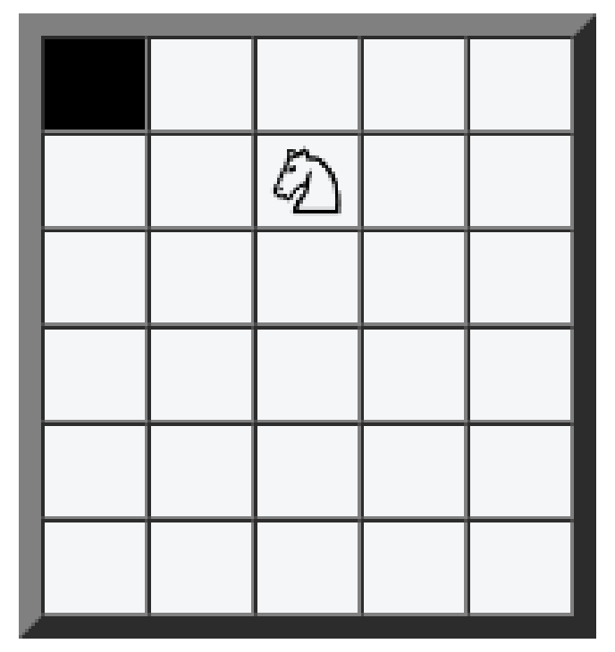

Figure 1: Knight's Tour, a single player game that MCTS performs reasonably well on that the linear regression model performed poorly on.

Features

Feature vectors for states are n length vectors where n is the number of propositions in the game. For example, if proposition j in checkers is "the red player has captured 4 total black pieces", then $x_j^{(i)} = 1$ if the red player has captured 4 total black pieces in state i. Otherwise it is $x_j^{(i)} = 0$.

First Model: Post-game Linear Regression

To begin, we attempted to get an upper bound on the efficacy of linear regression by running a standard Monte-Carlo Tree Search player with Propositional Nets on various games. Then, at the end of the game, we trained a linear regression model on the state data compiled during the game, with the game states being the x input vectors and the utility derived by Monte-Carlo Tree Search being the y target values. The results are shown in **Table 1**. Accuracy measures how often the Monte-Carlo Tree Search algorithm and the derived hypothesis function agreed on the best action to take given a state.

Game	Degree	Depth	Accuracy
3-puzzle	2	6	0.61
Buttons and Lights	3	6	0.21
Hunter	2-4	14	0.08
Knight's Tour	2-6	27	0.03
Connect Four	8	14	0.23
Checkers	9-10	49	0.71
Alquerque	10	29	0.72

Table 1: Results for each game tested. Both degree and depth describe the game tree (where each node is a state) and are approximations.

Interpretation of Results

Linear regression appeared to be most effective in cases where specific state propositions are associated strongly with a specific level of utility. The goal in both Checkers and Alquerque is determined by how many of the enemy pieces have been captured. The average breadth and depth of the game tree did not appear to have as much of a bearing on the efficacy of the algorithm.

Second Model: Pre-game Linear Regression

Since linear regression seemed to be accurate on games with advantageous game propositions, we realized that we could use linear regression to create a greedy player that prefers states that have as true propositions that are recognized to be advantageous. Such a player could be effective in situations where the start clock gave the player enough time to compile sufficient training data but the play clock was minimal enough that a regular MCTS player would essentially be forced to return a random action. Algorithm 1 shows the adjusted back propagation function used by the linear regression player during the start clock. During the play clock, it merely judges the best action based on which resulting state maximizes $h_\theta(s)$.

```
Data: Terminal state $s$ and associated utility $u$ derived by single depth
      charge of MCTS. $\theta$ derived from previous backward
      propagations and step size $\alpha$.
Result: Updated Monte-Caro game tree and $\theta$.
while $s \neq null$ do
    $s.utility \leftarrow s.utility + u$;
    if $s.recorded == false$ then
        for $j \leftarrow 0$ to $n$ do
            $\theta_j \leftarrow \theta_j + \alpha(u^2 - h_\theta(s.features))s.features_j$;
        end
        $s.recorded \leftarrow true$;
    end
    $s \leftarrow s.parent$;
end
```

Algorithm 1: Altered backwards propagation algorithm incorporating linear regression.

Results

The player that conducts pre-game linear regression, in its current state, does not appear to perform any better than a random player during games such as Checkers and Alquerque. There is a significant degree of difficulty involved in choosing an appropriate step size. In addition, training θ on the initial judgement of the utility of a state means we are training it on the weakest possible meaningful assessment of a state's utility. An alternative algorithm would be to collect the data first during the first portion of the start clock (by exploring the game tree using MCTS) and then during the second portion training on the collected data. It is difficult to judge, however, how to divide the time and how, specifically, to judge how much time will be necessary for convergence.

Future

Future potential research could involve determining a relationship between the size of the game tree and the feature vector (I.E. number of propositions per state) and assessing from there how much data and time is necessary for convergence. Alternatively, a greedy algorithm that involves directly examining the propositional network to determine propositions associated with specific goal values may have superior performance in certain circumstances in addition to being easier to implement.

References

[1] C. Browne et al, "A Survey of Monte Carlo Tree Search Methods" in *IEEE Transaction on Computational Intelligence and AI in Games*, vol. 4, no. 1, March 2012.

[2] M. Genesereth et al, "General Game Playing: Overview of the AAAI Competition" in *AI Magazine*, vol. 26, no. 2, Summer 2005.