Name: Chinedu Emeka
Class: CS 597
Assignment: Project Report

AI Strategies For Scrabble

Introduction

Scrabble is a popular board game in which two to four players alternately place tiles (words) on a 15 * 15 square board. The game is relatively easy to learn, but arguably difficult to master. Significant effort has been invested into developing heuristics and strategies to provide an advantage to informed players. The purpose of this project is to examine the game further and to provide insight as to whether computers are able to play the game effectively with as little preprogrammed information as possible.

Unlike other popular board games such as Chess and Checkers, Scrabble is not a fully observable game. It is a stochastically and partially observable game; there is some randomness involved, and players do not have all relevant information about the game state (players' cannot see the words that their opponents have available). Consequently, traditional evaluation methods such as a heuristic function with minimax alpha beta search are insufficient to ensure victory.

Previous Work

One of the most significant breakthroughs in Scrabble was Brian Sheppard's development of a computer scrabble player, Maven, that could defeat the strongest human players. Maven performed simulations to determine the likely outcome of certain positions and

also focused on rack evaluation (i.e. the tiles a player had at hand). The rack evaluation function and other parameters were carefully tuned to maximize performance.

Sheppard examined various human "expert" strategies, such as the advice that players should avoid placing vowels adjacent to bonus squares because this opened up significant opportunities for opponents to gain extra points. When he used computer analysis to estimate the impact of placing vowels in this manner, Sheppard found that it was greatly overestimated. This flaw in human strategy highlights the risks involved with computers basing their evaluation of game positions primarily on human knowledge; human knowledge can be wrong and extremely misleading.

Sutton took a different approach to developing a computer program which could play Scrabble. He deployed TD learning, which is a machine learning technique whereby an agent starts out with essentially no information about the value of state action pairs. The agent learns by exploring and updating Q(s,a), the quality of a given action given a particular state. Though Sutton's scrabble program was defeated by Maven, he contended that it could perform better with more exploration and learning. For this project, I investigated if computers could use neural networks to learn the game of Scrabble

Data and Methods

First, I developed a Scrabble game in Python. Certain features in Scrabble were not implemented (such as support for more than 2 players), so as to simplify the model and allow me to focus on the machine learning component of the project.

Unlike some other programs which only search for word combinations of up to a certain length n, I searched for all possible word combinations given the board and a player's rack. This allows for better representation of possible transitions to another state from the current state at the cost of efficiency.

A baseline agent was also developed. This baseline agent plays the move which maximizes its current score, without any consideration for any other features in the environment. This is not a recommended strategy for Scrabble. This agent was developed to serve as a basis for comparison. Given that the agent played as a novice, it was expected to be defeated by other agents.

Next, I attempted to get data of expert Scrabble games to be used for training the neural network. Data was unusually difficult to acquire; there were no extensive online databases which housed games of either strong human or AI Scrabble players. As such, I generated games with my baseline agent to use for analysis. However, because the agent used a simple heuristic, learning from it was not optimal. Eventually, I was able to get data from a researcher using Quackle, the strongest Scrabble AI available today.

With data, I proceeded to train a neural network. The network was trained using the scikit_learn library, with the rectified linear unit activation function.