NEA Proposal

March 19, 2025

1 Proposal

Idea

Using enthropy and other aspects of information theory, the idea is to make a bot/AI that could suggest the best word to play in scrabble.

Finer details

The bot should be able to suggest not only the best move, but also alternative suggestions (similar to a chess engine which suggests the top line to play, and also lines which are still winning, but not as good). To do this, the bot should take a heuristic approach and calculate the potential positions after n turns, and decide the best variant. This could be done with Monte Carlo methods.

As scrabble is a game with imperfect information, it is impossible to know what tiles the opponent has until the last stage of the game. However, with mutual information, it is possible to guess which tiles the opponent *probably* has. To do this, it should store what letters it hasn't seen yet and based on the entropy of the unseen tiles, it could predict the various tiles the opponent has. Alternatively, (**as an extension**), it should be able to make a slightly more educated guess, based on what the opponent has played (for example, if the opponent has played short words (2-3 letter), it can assume that the opponent is either saving tiles for a bingo or the tiles they have cannot make good words)

With letter frequency data, it is possible to optimise tile management. For example, given that 3 of 4 "s" tiles have been played, the bot should decide whether it is worth playing the "s" tile in its next turn, or to hold on to it, allowing it to make a hook off another move. Another example, if the bot holds the tiles "AAEEILO", it should be able to recognise that it has far too many vowels, and try to place down the vowels, to get better tiles for "bingo"ing.

1.1 Limitations

- The lexicon (valid words) will be CSW19 as NSWL (the lexicon used in North America) and CSW24 are not publicly available
- There will be no exchanging tiles or passing turns

2 Notes

Definition 1. *Entropy. The entropy of $H_b(X)$ in base b of a discrete random variable X is defined as*

$$H_b(X) = -\sum_{x\in\mathcal{X}} \mathbb{P}(X = x)\log_b \mathbb{P}(X = x)$$

where we use the convention $0\times\log_b 0 = 0$ *and the base is 2 unless specified otherwise.*

What this means is the negative sum of the probability of x in $\mathcal{X}$ multiplied by the log of that probability. For example

$$\mathcal{X} = \{1, 2, 3\}, \mathbb{P}(X = x) = \frac{1}{x}$$

$$H_b(X) = -(\frac{1}{1}\cdot\log_2(1) + \frac{1}{2}\cdot\log_2(\frac{1}{2}) + \ldots)$$

Since entropy only depends on the probability mass function, 2 separate variables with the same pmf have the same entropy. Alternatively, the entropy of a variable is the negative expectation of the log of probability of x. So in maths terms:

$$H(X) = -\mathbb{E}\left[\log(p(X))\right]$$

What this means to me is it helps prioritise likely letters. Reminder, our maximum is $\log_2 26 \approxeq 4.7$ and if our entropy is roughly that, then all the letters are equally likely, so it isn't helpful here (sad face). If our entropy is low (ie. 0.1) then we can assume that our most frequent letters (E,A,R,I,O) (see main.py to see where I pull these numbers from) will be used.