Applying Statistical Learning Ideas to Wordle

Samuel J. Hart

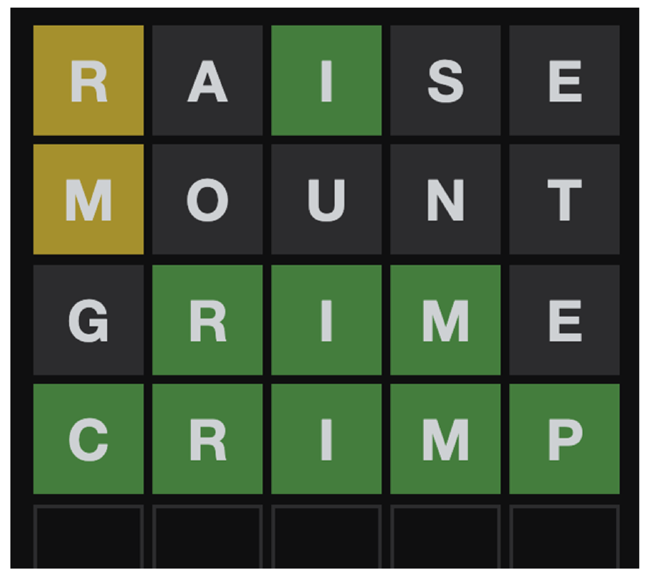
Introduction

Wordle is a word game created by Josh Wardle, who wanted to make a fun game for his significant other. The goal of the game is to guess five-letter words, collecting hints as you do, until you guess the correct pre-determined answer. You are only allowed six guesses.

This project uses the original Wordle guess and answer list curated by Josh's partner, and does not make use of any changes made by the New York Times. Note that there are 12,972 words that one may guess and 2,315 possible answers. Note that these possible answers were curated from the allowed guess list by Josh Wardle's significant other and so, there is some sense that these words are more common than the rest.

Wordle Explained

Before we state some goals of this project, it is important to get a grasp on how the Wordle game works. Below is an example game.



Greyed-out letters are not in the answer, yellow letters are but in a different location, and green letters are in the answer in that location.

Questions to Answer

With a grasp on the game, we hope to answer the following.

- What makes a guess good and can we quantify it?
- How do we use this metric to measure guess and playing strategies?

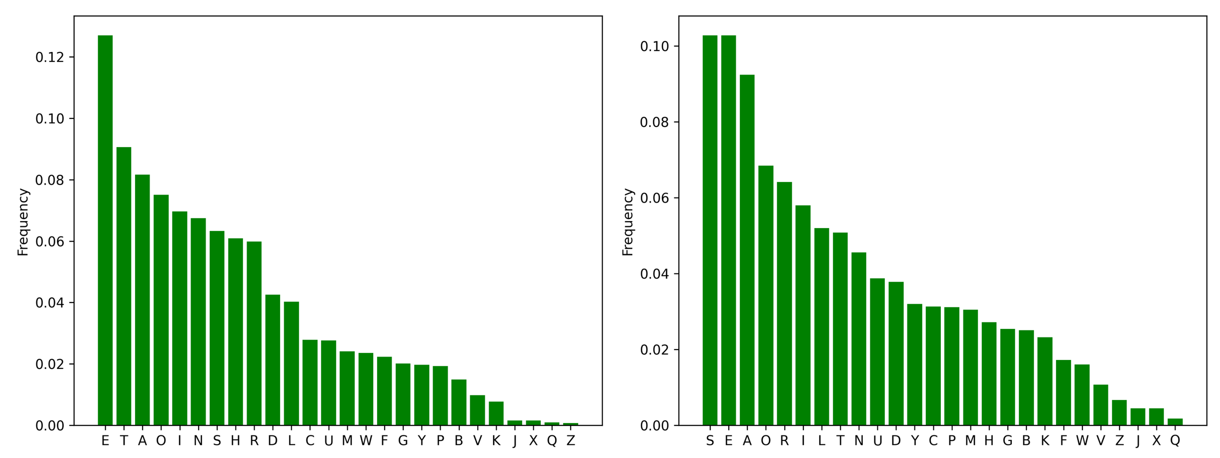
Once we answer these questions, we will develop a Wordle Solver that solves the puzzle from a starting word in the fewest number of guesses possible and which does so in a way that is somewhat similar to how a human player would also solve the puzzle.

References

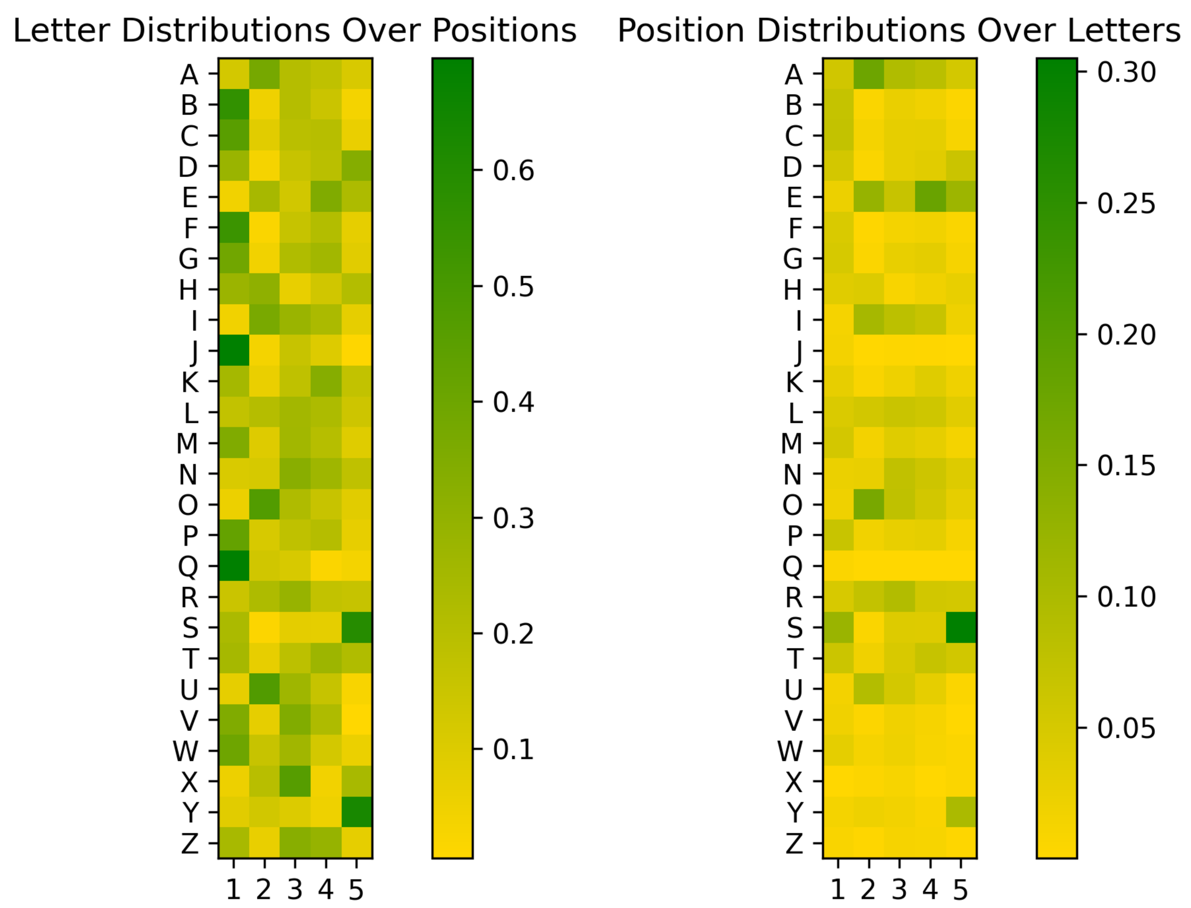
[1] Sanderson, G. "Solving Wordle using information theory"; YouTube, 2022.

[2] Emory University, The Department of Mathematics and Computer Science. "Letter Frequencies in English"; accessed 2022.

Initial Ideas



There has been debate about the best starting word that gets you to the answer quickest. Some say you should focus on vowels since the answer must have a vowel, hence "ADIEU" became a popular start. Others seem to think your start word should include some of the most common letters in the English language (above left) [2]. From this, we get "SHARE" and "OTHER". However, given that the Wordle allowed guess list is a sample of five-letter words of the English language, we may want to take a look at its letter frequencies (above right). From this, we can see that maybe "OTHER" and "SHARE" are not as good, and that "OILER" and "STARE" could be suitable replacements. Furthermore, ideas like these put too much emphasis on the letter inclusion and don't consider letter position. For example, "STARE" can be rearranged to "TEARS". Is that S better at the end or the beginning? What about the E?



Above we have two heat maps one which shows the distribution for each letter over all positions (above left), and the other shows the distribution of each position over all letters (above right). This can give us some insights into the placement of our letters. The left shows that S is most likely at the end of a word it is in, but the right shows you that the answer most likely starts with an S. Furthermore, if a word contains a J or Q, it is often at the beginning. Everything we know about English says this makes sense, and so we should guess words with letters in their most likely positions and if they come up yellow, then guess them in their next likely position and so on. Despite these insights, this still feels too random and doesn't tell us much about what is likely to get the answer the quickest. So, can we do better to quantify the goodness of a guess?

Informed Strategy

Due to the nature of gaining pieces of information at each guess, we chose to apply the ideas behind information theory and decision trees to create a Wordle strategy. We can think of each possible guess at a step as a one of our features and the pattern of colors that it gives as the value of its feature.

Note that for our strategy, the algorithms consider all allowed guesses as possibly being the answer. Using the answer list would essentially be overfitting to a test set. When we refer to a pattern we refer to the sequence of colors that could arrive from a guess. Now, similar to creating splits on a feature in a decision tree, we will choose a word (feature) to guess based on how much information we expect to gain from guessing it.

To quantify this expectation of information, we say that a guess and the pattern associated with it give one bit of information if it eliminates half of all possible answers. Note that if it eliminates half of all possible answers, that we have probability 1/2 of seeing this pattern when guessing the word. So to compute the expected information gain of a guess, we iterate over all the patterns we could see given what possible answers are left and take the sum of each probability of a pattern times the number of bits of information it has given us. That is,

$$E(\text{Info} \mid \text{Word}) = \sum_{x \in \text{patterns}} p(x) \log_2(p(x)).$$

Thus, at each step after the start word, we compute expected information for all of the guesses. Note that this expected information is a heuristic, and that it only looks at one step ahead. For example, there may be one word that gives a lower expected information on this guess, but a guess after it could give much larger amounts of information. The most optimal program would do this step look-ahead to make the choice for each guess, however, this was too computationally expensive. For the purpose of this project we will only look at the current step.

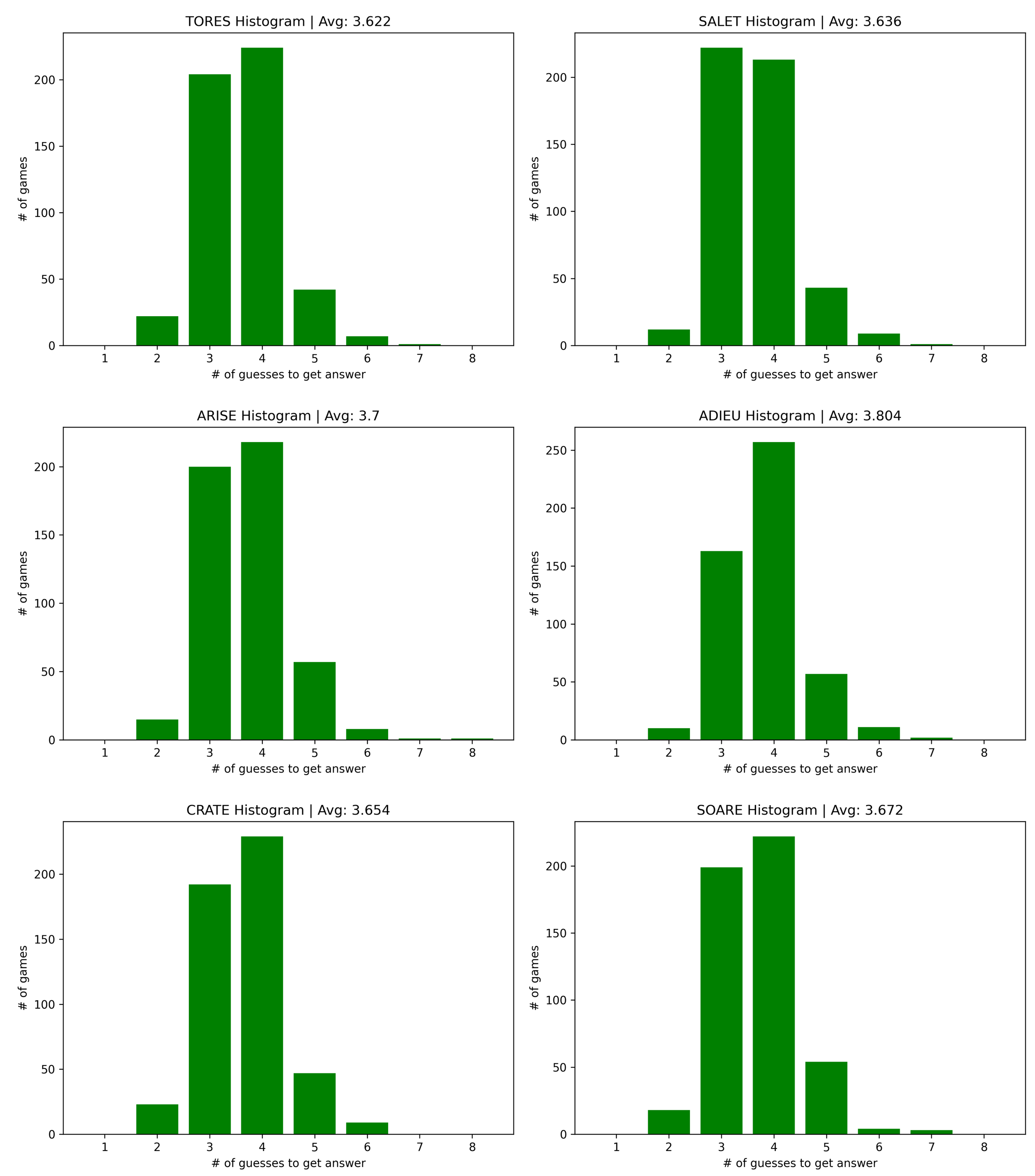
Lastly, we also make use of English word frequencies to give more common words a high probability of being the answer. The reason for this is that the answer list is human-curated from the allowed guess list and there is some semblance that the words chosen were more common. To do this, we only needed to include a Python package that conveniently fetches their English language frequencies, and then build a normalized frequency distribution of the possible answers.

Results

We've determined what makes a guess good, and how to use the metric to measure guess and playing strategies. To collect results we calculated the top 100 starting words by their expected information. We then simulated 500 games starting from each of these 100 candidates. At each step the computer chooses it's guess based on three conditions:

- If there is a guess with greater than 50% probability of being the answer, guess that,
- If there are four or less possible answers, guess the most probable,
- Otherwise, guess the word with the highest expected bits of information to be gained.

Below are the 2 start words that resulted in the lowest average number of guesses to the answer, as well as some of the more popular start words. Also, since these are simulating on a sample of the Wordle list, their true averages could be lower or higher depending on the words in the sample.



Conclusion

First, we quantified the quality of a guess based on the amount of information it gives. We then developed a Wordle Solver that solves a Wordle puzzle from a start word following the choice criterion described in the Results box. Note that we made departures from the strategy in [1]: by only calculating expected information of just one step or guess instead of looking two steps ahead; we used a normalized frequency distribution for determining the probability that any possible answer was the answer instead of using a sigmoid function; and we chose the next guess based on expected information rather than expected score. These choices lead to a less computationally expensive solver that gave similar results as our best word had an average number of guesses only about 0.2 higher of that found in [1]. Furthermore, it would seem that this model better mimics human behavior as we often only consider our next best guess, not the next best 2 guesses. Note that all our results from the 100 candidate starting words had averages within a range from 3.6 to 3.9, however, few like CRATE never failed the puzzle and always managed to solve it in six or less guesses.