DRAFT VERSION MAY 15, 2022
Typeset using LaTeX **twocolumn** style in AASTeX631

How to Beat Wordle: An Optimization Approach Using Shannon Entropy, Minimax, and Frequency Analysis

JONAS HANSEN[1]

[1] *Harvard College*
Cambridge, MA 02138, USA

(Dated: May 15, 2022)

ABSTRACT

The word game Wordle has recently taken the internet by storm. Millions of people from across the world play it daily, addicted to its simplicity and ease of accessibility. But this simplicity is illusive. Finding the most optimal strategy to beat Wordle has sent game theorists and computer science hobbyists scrambling to uncover the best algorithm, both in terms of success and efficiency. In this project, we explore and develop three different approaches to optimally winning Wordle: a Shannon entropy algorithm, a minimax algorithm, and a frequency analysis approach. The three algorithms are written in Python, as well as the helper functions to extract and analyze relevant data. After running each algorithm across all the possible solution words accepted by Wordle, the lowest average number of guesses it took to find the solution word was 4.012 for Shannon entropy, 3.610 for minimax, and 4.378 for frequency analysis, with starting words "salet", "crane", and "bares", respectively, and runtimes of 1981.27 minutes, 3.66 minutes, and 12.77 minutes, respectively. While not quite optimal, these results provide insight into how information theory and statistical analysis can be powerful allies in the world of game theory and decision making.

Keywords: Wordle, Shannon entropy, Minimax, Frequency Analysis, Game Theory, Decision Making, Python, Optimization, Algorithm

1. INTRODUCTION

In October 2021, a software engineer in Brooklyn named Josh Wardle would release one of the most popular online word games today: Wordle (Wardle 2021). The objective of the game is simple: there is a secret five-letter "word of the day" that you are trying to guess within six attempts (see Section 2.1 for a more detailed explanation of the game). While the game was of course intended to be played by humans, the hunt has been on to find the best algorithms to optimally beat the game using a computer. So the question to be explored in this project is the following: how can we use mathematics and computer science to efficiently guess the word of the day using as few guesses as possible? To answer this, we will develop three different algorithmic approaches to beat Wordle: Shannon entropy, minimax optimization, and frequency analysis. We will analyze their performance, efficiency, and practicality, and address how they might improved upon and applied to additional applications.

2. BACKGROUND RESEARCH

2.1. *How Wordle Works*

If you haven't heard of it, Wordle is a deceptively simple word game were you are prompted to guess the word of the day, which is a five-letter word in English. But, you are only allowed a maximum of six guesses to do so. Luckily, every time you guess a word, the game gives you some hints: if a letter in the word you guessed is not in the word of the day, that letter turns gray. If it is in the word of the day but not in the same position as the word you guessed, the letter turns yellow. And finally, if it is in the word of the day and in the same position as the word you guessed, the letter turns green. Therefore, the goal is to get all five letters to be green using as few guess words as possible. Let's look at an example from playing Wordle on May 11th, 2022:

In the example shown in Figure 1, we can see that the word of the day was guessed in just three attempts. It takes Americans an average of 3.92 tries to solve Wor-

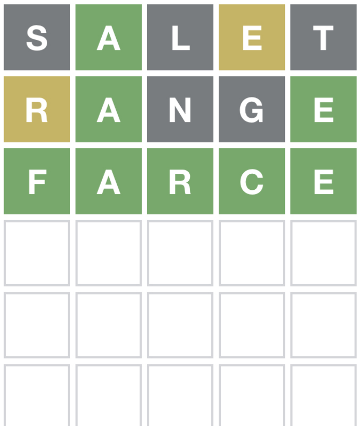


Figure 1. An example of a human playing the Wordle game from May 11th, 2022.

dle, so getting it in three is pretty impressive (Wordtips 2022)!

Humans are generally very good at picking up patterns quickly (Mattson 2014), so using the hints from the game and their background knowledge of the English lexicon should be sufficient to beat the game 100% of the time. Unfortunately, when you have a less common word of the day—such as "farce" in the above example—humans often struggle because they simply are not familiar with the word. This is where algorithmic approaches prove superior, since computers are easily able to load a list containing every possible five-letter word in English. On top of that, there is an even smaller list of words that Wordle actually uses for its solutions, which the computer can take into consideration. Let's dive into a few of these algorithmic approaches.

2.2. *Shannon Entropy*

Information theory is a branch of mathematics and computer science that involves the processing and communication of data (Martignon 2001). One of the pioneers in the field of information theory was Claude Shannon, and he presented the idea of "entropy," which, simply put, is the number of bits required to describe a random variable. The formula for entropy is:

$$H(x) = -\sum_{i=1}^{n} P(x_i) \log P(x_i) \tag{1}$$

where $P(x_i)$ is the probability of some event x_i occurring. It is important to note that as the probability of an event increases, the amount of information gained from such an event occurring decreases. In the context of Wordle, the more likely a pattern within a guess word shows up in another word, the less knowledge we gain about the solution word. Shannon entropy informs us of this level of information gain, so it is favorable to guess words with the highest expected reduction in entropy, thereby reducing the set of words to choose from for the subsequent guess.

In February of 2022, the YouTube channel 3Blue1Brown—which makes stunning animation videos explaining topics in mathematics and computer science—released a video describing a process to find the best word to open with when playing Wordle (Sanderson 2022). This was done using an exhaustive approach based on Shannon entropy, and found that the opening word that best reduced the following samples spaces was "salet" (the starting word that was used in the game depicted in 1). Trying to find the best opening word using Shannon entropy in the scope of this project would require massive computing resources and time, so we will assume "salet" is an optimal opening word based on 3Blue1Brown's findings.

2.3. *Minimax*

The minimax algorithm is an approach often seen in game theory to find the optimal move for a player, without knowledge of the opponent's move but assuming that the opponent plays optimally (Eppes 2019). In two-player zero-sum games, this algorithm essentially computes the Nash equilibrium, as the *minimax value* of a player is just the smallest value that the opponent can force the player to receive. Formally, the minimax value is defined as:

$$\overline{v_i} = \min_{a_{-i}} \max_{a_i} v_i(a_i, a_{-i}) \tag{2}$$

where i is the index of the player in question, $-i$ is the index of all the opponents, a_i is the action taken by player i, and a_{-i} is the actions taken by the opponents.

In the context of Wordle, this algorithm can help us choose the optimal guess word that minimizes the largest remaining possible world list, without any knowledge of the list of possible solution words. In this sense, we would expect this minimax algorithm to perform more efficiently than the Shannon entropy algorithm, since it does not need to compute the entropy across every possible pattern matching distribution as the entropy algorithm does.

2.4. *Frequency Analysis*

If you've ever wondered what the most common letter is in the English language, or how you might decrypt a cipher text that is based on a substitution cipher, than you've been exposed to the field of frequency analysis. Frequency analysis allows us to gather statistics on the occurrences of letters and words to deduce patterns and commonalities within a given data set. In fact, using frequency analysis, we know that the most common letter in the English language is "e", and that substitution ciphers can be broken by identifying the most recurring letters in a cipher text. To say the least, it is a simple yet powerful tool for analyzing words, making it a formidable ally in the battle to optimally beat Wordle.

Not only can we use frequency analysis to investigate the number of occurrences of letters in Wordle words but we can also compute frequency distributions for each *position* within a word. Given that all Wordle words have five positions (i.e. they are all five-letters long), this will prove to be a highly informative technique.

3. APPROACH

3.1. *Data Collection*

The main data that was used for all of the algorithms is simply a list of all possible five-letter words in the English language. Additionally, although not needed for the algorithms themselves, a text file containing a list of all of the possible Wordle solution words was also used. This was for automating the task of simulating the game so that each algorithm could be tested to see how optimally it could solve for each of these solution words (so we wouldn't have to manually do it online). The list of all possible five-letter English words contains 12,953 words (which is technically not *all* possible five-letter words, but is based on the Official Collins Scrabble Words). The list of possible Wordle solution words contains 2,309 words. These data sets were pulled from 3Blue1Brown's database (Sanderson 2022).

3.2. *The Shannon Entropy Algorithm*

Entropy, as we have discussed, is a measurement of how random something is. In the case of solving Wordle, entropy can therefore help identify the guess words that will have the most random outcome. This is favorable when analyzing the outputted pattern of hints (green, yellow, gray) since we can then make a guess that will best reduce the entropy for the subsequent round (i.e. so that we may gain more information for the next guess). This can be directly translated to the equation for entropy (eq. 1): if we let x_i be the outcome of a guess, then the probability of that outcome, $P(x_i)$, is the possible words that fit that outcome divided by all possible words. Plugging this into the log portion of the entropy expression:

$$\begin{aligned} -\log P(x_i) &= -\log\left(\frac{\text{words matching outcome}}{\text{all words}}\right) \\ &= \log(\text{all words}) \\ &\quad - \log(\text{words matching outcome}) \end{aligned}$$

So really what this means is that we are computing the difference between the list of words we started with and the list of words we now have after making a guess. If we can calculate the entropy across the entire word pattern distribution space, we can make informed choices about what guess words to pick to most optimally reduce the subsequent set of words we have to consider (by maximizing the log difference seen in the above expression).

The algorithm will therefore follow these steps, iterating through each solution word in the list of possible Wordle solution words:

1. Start with a guess word and call `calculate_hints` function to see how well it did against the solution word (returns a tuple like (1, 2, 0, 0, 1) where 0 = gray letter, 1 = yellow letter, and 2 = green letter).

2. Based on hint outcome, calculate entropy distribution across the list of words and trim the list down to words that could fit the outcome, in order of descending entropy.

3. Choose the first word in the new trimmed list (i.e. the one with the highest expected reduction in entropy) and return to Step 1 using that word as the new guess word.

4. Repeat until the guess word matches the solution word.

3.3. *The Minimax Algorithm*

Of the three algorithms explored in this project, the minimax algorithm might be the most intuitive. The goal of the algorithm is to simply choose the guess word with the smallest worst-case response. Similar to the entropy approach, we can use the hint feedback provided by the game to evaluate this response. We assign values to different word patterns such that higher values indicate a higher probability of winning. Using these pattern state values, we can then narrow down the list of words to a set that maximizes our win probability whilst minimizing the number of steps taken to do so. In a sense, this algorithm simulates a two-player game where you are one player and the Wordle game is your opponent. This algorithm attempts the minimize the value states achieved by the game to maximize your gain (by picking guesses that reveal the most information. Guesses that do not reveal much information are therefore advantageous for your "opponent," the game).

3.4. *The Frequency Analysis Algorithm*

Analyzing letter and letter position frequencies of guess words in Wordle, we can develop a simple and efficient algorithm to filter out inapplicable words and suggest the most likely solution words.

Figures 2 and 3 reveal statistics about the list of all possible five-letter words that are crucial to developing a frequency analysis approach. Figure 2 is a histogram showing how frequently each letter appears in the five-letter words, with ”s”, ”e”, and ”a” being notably the most frequent. Figure 3 shows the relative frequency of each letter appearing in each of the five possible positions in a five-letter word (showing only the top ten most frequent letters for clarity). We can see, for example, that it is very common for the letter ”s” to appear at the end of word (with a relative frequency of about 30%).

Using these statistics, the algorithm for solving Wordle using frequency analysis boils down to the following steps:

1. Given a hint pattern based on a guess made, compute the frequency distribution for each position in the word.

2. Fix the letter in the position where its frequency is the highest, and remove words from the search space that don’t have the letter in this position.

3. Compute the new frequency distribution and repeat process starting at Step 2 for the rest of the letter positions.

4. With the filtered word list, choose the word with the best letter position match and use that as the next guess word (if multiple words have the same match percentage, just pick one of them).

5. Repeat starting at Step 1 until you have found the solution word!

4. RESULTS AND ANALYSIS

4.1. *Finding the Best Opening Word*

The first point to explore was addressing which opening word was the best at minimizing the number of guesses it took to win a game. The runtimes of the Shannon entropy and frequency analysis algorithms were too high to run them multiple times for multiple different opening words, so the minimax algorithm was used to test some of words considered most optimal according to literature (Selby 2022). The opening word with the lowest average number of guesses per game, according to minimax, was ”crane”, as depicted by the distribution plot in Figure 4:

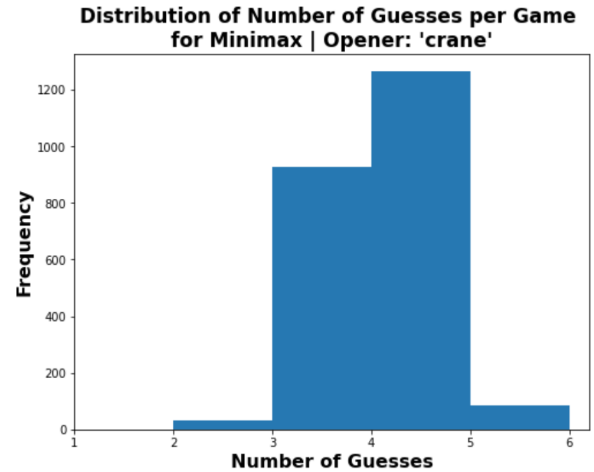


Figure 4. Plotting the distribution of number of guesses per game using the minimax algorithm with opening word ”crane”. The mean is 3.610 guesses.

One common strategy employed by many Wordle players is to use an opening word with a lot of vowels in it, such as ”adieu.” Logically, this would seem to be a legitimate approach, since all five-letter words contain vowels. Figure 5 shows the guess distribution for ”adieu” using minimax:

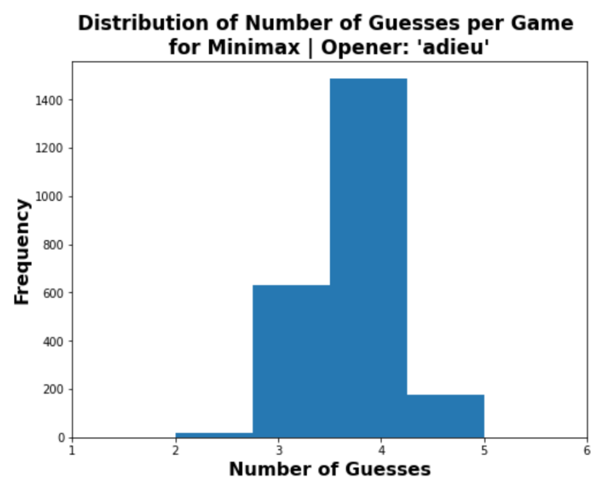


Figure 5. Plotting the distribution of number of guesses per game using the minimax algorithm with opening word ”adieu”. The mean is 3.788 guesses.

Figure 5 above shows that ”adieu” performed relatively poorly, with a mean of 3.788 guesses per game (compared to 3.610 for ”crane”, or 3.614 guesses for ”salet”). However, the worst case for ”adieu” was 5 guesses, while the worst case for ”crane” was 6 guesses. This indicates that vowels may not be an optimal approach but rather a ”safer” approach—to stay within

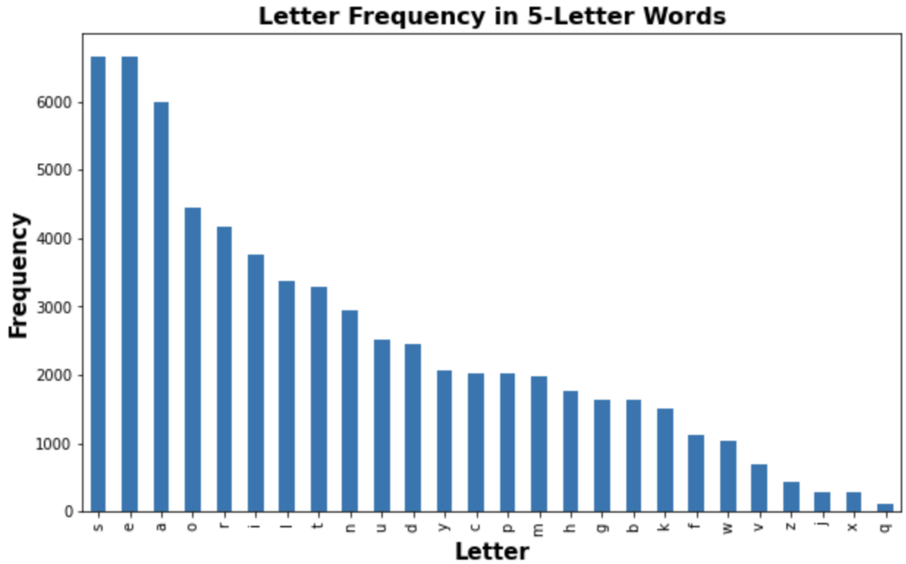


Figure 2. Plotting the frequency of each letter appearing in the list of all possible five-letter English words.

	count	freq	freq_pos0	freq_pos1	freq_pos2	freq_pos3	freq_pos4
s	6665	0.102760	0.120644	0.007169	0.041088	0.039778	0.305119
e	6662	0.102714	0.023358	0.125501	0.067993	0.179386	0.117330
a	5990	0.092353	0.056815	0.174453	0.095282	0.082794	0.052421
o	4438	0.068424	0.020197	0.161579	0.076549	0.053808	0.029988
r	4158	0.064107	0.048412	0.072464	0.092353	0.055427	0.051881
i	3759	0.057956	0.012720	0.106614	0.081021	0.067838	0.021585
l	3371	0.051973	0.044480	0.053885	0.065372	0.059436	0.036694
t	3295	0.050802	0.062828	0.018424	0.047487	0.069226	0.056044
n	2952	0.045513	0.025054	0.026596	0.074314	0.060746	0.040857
u	2511	0.038714	0.014570	0.091505	0.051418	0.030913	0.005165

Figure 3. Plotting the frequency of each letter appearing in each position of five-letter English words (showing top ten most frequent letters here).

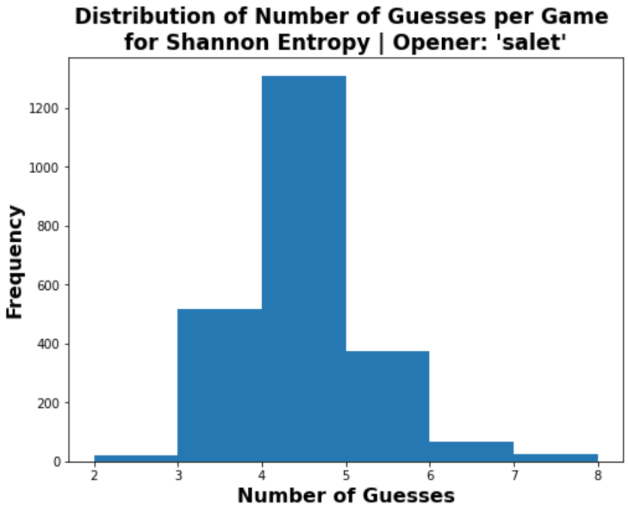
the 6-guess limit—highlighting an interesting distinction between the strategy taken by many humans and the minimax algorithm.

4.2. *So Which Algorithm Was the Best?*

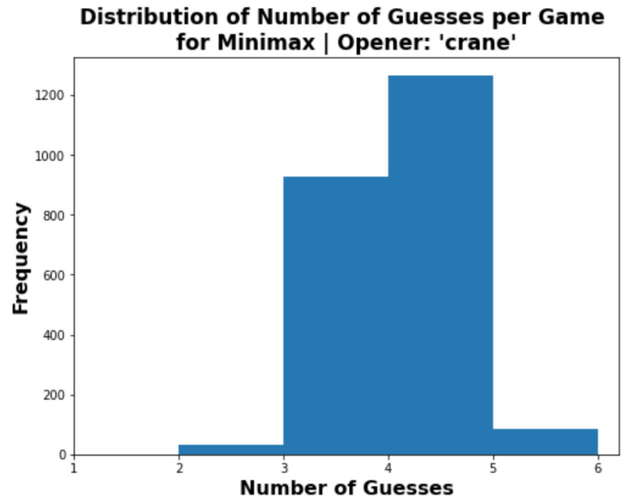
To determine which algorithm performed most optimally, we need to consider several factors. The first is the average number of guesses per game. Using the best opening words for each algorithm, the average number of guesses were 4.012 for Shannon entropy, 3.610 for minimax, and 4.378 for frequency analysis. The distribution plots for these results are shown in Figure 6.

The second factor to consider is the worst-case number of guesses. This turned out to be 8 for Shannon entropy, 6 for minimax, and 10+ for frequency analysis (some games weren't able to find the solution word within 10 guesses).

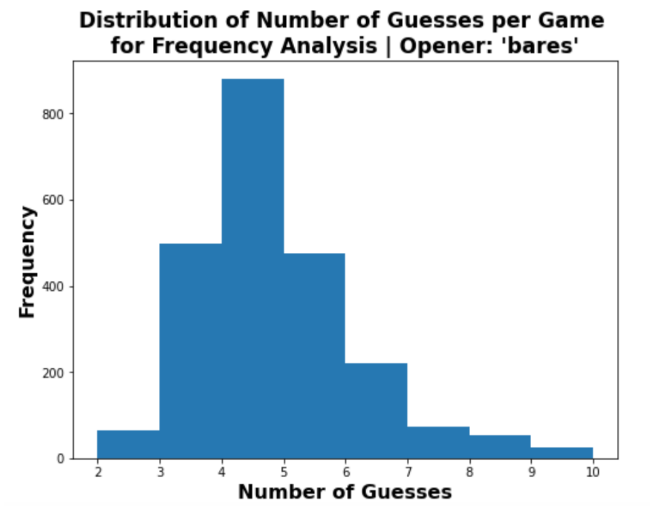
The final factor to consider is runtime. The runtime to play the game across all 2,309 Wordle solution words with a fixed starting word was 1981.27 minutes for the Shannon entropy algorithm, 3.66 minutes for the minimax algorithm, and 12.77 minutes for the frequency analysis algorithm.



(a) Shannon entropy guess distribution plot; Mean = 4.012 guesses. Worst-case = 8 guesses.



(b) Minimax guess distribution plot; Mean = 3.610 guesses. Worst-case = 6 guesses.



(c) Frequency analysis guess distribution plot; Mean = 4.378 guesses. Worst-case = 10+ guesses.

Figure 6. Best-case guess distribution plots for Shannon entropy, minimax, and frequency analysis algorithms.

Therefore, from both a performance and efficiency perspective, the minimax algorithm proved to be the best of the three. Of course, computing time and resources limited the exploration of what the best starting words would be for the Shannon entropy and frequency analysis algorithms. With more time for both code optimization and processing, better opening words could be used to improve upon these two approaches.

5. CONCLUSION

Through rigorous statistical analyses, code development, and algorithmic tests, we have explored the complexities of trying to beat a seemingly simple word game, Wordle. While challenging and not completely optimal, the three algorithms we developed were successful in beating the game, each with a unique strategy of doing so. By examining guess-distribution plots and overall runtimes, we found that the minimax algorithm performed most optimally, although future work could be done to better optimize the Shannon entropy and frequency analysis algorithms to compete with minimax's performance.

But such future work need not only be limited to these specific algorithms. Alternative methods, such as decisions trees, Alpha-Beta pruning, and reinforcement learning, would be fascinating avenues to explore to getting even better overall results. The open-source community has already made incredible progress in the search to uncover the "holy grail" of Wordle-solving algorithms, with guess averages hitting the 3.42-range (Selby 2022).

What's more, the techniques explored in this project are applicable beyond just Wordle. They are handy tools within the increasingly applicable fields of game theory and decision making. In fact, this kind of information theory-based decision making is the basis for much of machine learning, an extremely influential field in and of itself (Boruah 2021). Could a simple word game like Wordle truly begin to impact our everyday lives? Only time will tell.

ACKNOWLEDGMENTS

This research project was inspired by the insightful instruction of Sasha Petrov, Kevin Lin, Jasmine Huang, and Maira Khan. Thank you!

Software: Code available at this GitHub repository: https://github.com/JoJoFence/Math157FinalProject

REFERENCES

Boruah, P. (2021). Implications of information theory in machine learning.

Eppes, M. (2019). Game theory - the minimax algorithm explained.

Martignon, L. (2001). Information theory.

Mattson, M. P. (2014). Superior pattern processing is the essence of the evolved human brain.

Sanderson, G. (2022). Solving wordle using information theory.

Selby, A. (2022). The best strategies for wordle.

Wardle, J. (2021). Wordle game.

Wordtips (2022). Where in the world is the best at solving wordle?