Hangman Challenge

Name- Rohit Raj
Email- rohitraj21@iitk.ac.in

Introduction

- The objective of this assignment was to develop an algorithm that improves the accuracy of predicting letters in the Hangman game. My approach focuses on utilizing N-gram models ranging from 1-gram to 9-gram, leveraging statistical language models to predict the most probable next letter based on the context provided by partially revealed words.

N-Gram Models

- **Data Preparation**: First I have made build_n_gram function which construct a nested dictionary that counts how often sequences of n letters appear in a collection of words which is already provided. It's useful for predicting what letters might come next in a sequence during word formation.

- **Probability Calculation:** The `find_n_gram_probability` function is used in solving the Hangman problem to estimate the probability of each letter being the correct guess for a missing letter in a sequence of n letters where one position is unknown (`XXXX.` format). By analyzing patterns from a n-gram model built on a training dictionary, it helps predict the next letter more accurately. This prediction is crucial in improving the guessing strategy throughout the game, aiming to maximize the chances of guessing the correct word before running out of attempts.

- **Weighted Combination**: The predictions from different N-gram models were combined using weighted averages. Initially, higher-order models (e.g., 9-gram) were given more weight. If a higher-order model did not yield a prediction (due to insufficient context), the weight was redistributed to the lower-order models, ensuring comprehensive coverage of possible predictions.

Some general strategies to improve the accuracies

- If no guess can be made based on the 1-gram model (e.g., all probabilities are zero or all letters are guessed), it defaults to focusing on high-frequency letters in the English language (`'e', 'a', 'r', 'i', 'o', 't', 'n', 's', 'l', 'c'`). This aims to increase the likelihood of a correct guess.
- Everytime the _n_gram nested dictionary is updated by removing the guessed letter from the dictionary.

- Increasing the value of n in n_gram model the accuracy is increased significantly.specially for the longer words my model is working very efficiently giving about 85-90 % accuracy. However, in 10_gram it stops working (predicts no letters)