CS771: Assignment 2

Anirudh [200128], Bhavya Gupta [22111017], Kunal Singh [200535]
Lakshmi Pravallika [200282], Sana Chaitanya [200599], Shaijal Tripathi [22111274]

05 April, 2023

1 Introduction

We have implemented three different strategies for creating a decision tree. Of the three, we have chosen the first approach as our preferred method. However, we also provide detailed information about two alternative methods that yield comparable performance for the problem.
All three methods have been provided for code evaluation too.
For each of the methods, we describe the following aspects of decision tree design.

1. criteria to select the most informative word at each node to query.
2. criteria for determining when to stop expanding the decision tree and convert a node into a leaf.
3. hyperparameter factors such as the size and depth of the node

2 Our Chosen Method

2.1 Building Frequency Dictionary of Word Characters

1. At root level, the nodes are split based on the length of the words.
2. Next level onwards, we build a frequency dictionary of l*26 cells, where l is the length of the word at that node.
3. In each cell, we store $f_i char$, the frequency of an alphabet occurring at each index in the cell, where i is the index, f is the frequency of occurrence of the character char at position i.
4. At each level, the split depends on the query word given to Melbo. At each node, there are some subgroups of words. From this subgroup of words, we choose the best word to query.

5. For each word in the words at that node, we count the frequency sum for the word by summing over the frequency of the character at that index. Example: Word = "decision"

$$f_{count}=f_1{}'d'+f_2{}'e'+f_3{}'c'+f_4{}'i'+f_5{}'s'+f_6{}'i'+f_7{}'o'+f_8{}'n'$$

6. Based on the maximum f_{count} frequency sum, we split the node further at next level.

3 Alternate Methods

3.1 Alternate Method 1: Comparing the Rhyming Potential

1. Our approach entails a systematic process for determining the best rhyming word from a group of words stored at a particular node. This method begins by analyzing each word in the set and comparing it to every other word contained in that same set.

2. To determine the rhyming potential of each word, a score is assigned to each one based on a specific concept taken from the reveal function within the Node class. This score is incremented every time a character in a word matches with the corresponding character in another word at the same position.

3. After the scores for all words in the set have been calculated using this system, the word with the highest score is identified as the best rhyming word. This word is then designated as the query to be used for the intended purpose.

4. Overall, this process provides a straightforward approach for finding the word with the best rhyming potential by systematically comparing words within a set and assigning scores based on the degree of similarity between the words at specific positions. The result of this method is a single word identified as the best candidate for rhyming, which is then used as the query word at that node.

3.2 Alternate Method 2: Max-child Heuristic for Balanced Split

1. We have considered the following heuristic to split the node: This strategy is applied for all the nodes other than the root node.

2. For every word that has reached a particular node, we assign it as the query word and compute the number of children it creates for that node.

3. We tend to pick the word that results in the maximum number of children.

4. We also calculate the entropy for each of the children and use it to make a decision to select the query word among those set of words that yield the same number of children and they also create the maximum number of children.

5. The reasoning behind this approach is simple. We thought that if the breadth of the tree would be maximised, it would, in turn, minimize of the height of the decision tree and thus, would require lesser number of queries to guess the word.

4 Comparative Performance of three methods

Below, we report the variations in training time of the decision tree, model size, win count and average queries on applying different methods.

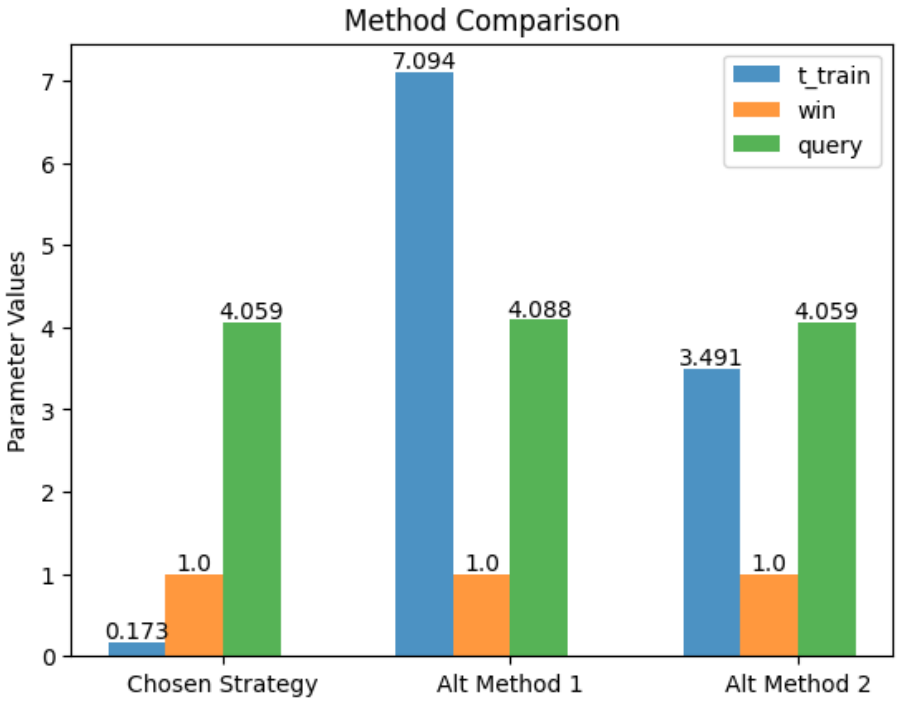

Figure 1: Evaluation Parameter Values for Various Methods

4.1 Table 1: Evaluation Parameters for Three Methods

Method	t_train	model_size	win$_c ount$	query
Method 1	0.173	1089772.0	1.0	4.059
Method 2	7.094	1072676.0	1.0	4.088
Method 3	3.491	1089643.0	1.0	4.059

Comparison of time complexity of each method:
Method 1: O(n)
Method 2: O(n^2) as we are comparing each word with every other word
Method 3: Relatively longer time