Spell Checking System using Bayesian Probability

Name: J Krithika
Roll Number: 22011101046

24/01/2025

Aim

To develop a robust and intuitive spell-checking system that uses Bayesian probability and edit distance techniques to identify and correct misspelled words. The system evaluates candidate corrections based on prior probabilities, likelihoods of transformations, and a detailed analysis of edit operations, ensuring accurate and contextually appropriate suggestions for users.

Source Code

```python
import re
from collections import Counter
from math import exp

# Corpus for this example
CORPUS = ["hello", "world", "word", "would", "work", "wool","whirled","curled"]

# Create a frequency table from the corpus
def create_frequency_table(corpus):
    return Counter(corpus)

# Helper function to calculate edit distance and operations
def calculate_edit_distance_and_operations(misspelled, candidate):
    len1, len2 = len(misspelled), len(candidate)
    dp = [[0] * (len2 + 1) for _ in range(len1 + 1)]
    operations = [[[] for _ in range(len2 + 1)] for _ in range(len1 + 1)]

    for i in range(len1 + 1):
        for j in range(len2 + 1):
            if i == 0:
                dp[i][j] = j
                operations[i][j] = ["Insert " + candidate[k] for k in range(j)]
            elif j == 0:
                dp[i][j] = i
                operations[i][j] = ["Delete " + misspelled[k] for k in range(i)]
            elif misspelled[i - 1] == candidate[j - 1]:
                dp[i][j] = dp[i - 1][j - 1]
                operations[i][j] = operations[i - 1][j - 1]
            else:
                delete = dp[i - 1][j] + 1
                insert = dp[i][j - 1] + 1
                substitute = dp[i - 1][j - 1] + 1
                dp[i][j], op = min((delete, "Delete"), (insert, "Insert"), (substitute, "Substitute"))
                if op == "Delete":
                    operations[i][j] = operations[i - 1][j] + [f"Delete {misspelled[i - 1]}"]
                elif op == "Insert":
                    operations[i][j] = operations[i][j - 1] + [f"Insert {candidate[j - 1]}"]
                else:
                    operations[i][j] = operations[i - 1][j - 1] + [f"Substitute {misspelled[i - 1]}     {candidate[j - 1]}"]

    return dp[len1][len2], operations[len1][len2]

# Calculate likelihood using edit distance
def calculate_p_x_given_w(edit_distance):
    return exp(-edit_distance)

# Calculate prior probability
def calculate_p_w(word, freq_table):
    return freq_table[word] / sum(freq_table.values())

# Spell check implementation with detailed evaluation
def spell_check(x, corpus):
    freq_table = create_frequency_table(corpus)

    print("Candidate Words Evaluation:")
    print("--------------------------------------------------")
    max_posterior = 0
    best_word = None
    for word in freq_table:
        edit_distance, operations = calculate_edit_distance_and_operations(x, word)
        p_x_given_w = calculate_p_x_given_w(edit_distance)
        p_w = calculate_p_w(word, freq_table)
        posterior_probability = p_x_given_w * p_w

        print(f"Word: {word}")
        print(f"  Edit Distance: {edit_distance}")
        print(f"  P(X|W) (Likelihood): {p_x_given_w:.6f}")
        print(f"  P(W) (Prior Probability): {p_w:.6f}")
        print(f"  P(W|X) (Posterior Probability): {posterior_probability:.6f}")
        print(f"  Operations to Transform '{x}'     '{word}': {operations}")
        print("--------------------------------------------------")

        if posterior_probability > max_posterior:
                max_posterior = posterior_probability
                best_word = word

    print("Final Correction:")
    print("--------------------------------------------------")
    print(f"Misspelled Word: {x}")
    print(f"Corrected Word: {best_word}")
    print("--------------------------------------------------")
    return best_word

misspelled_word = "woorld"
correct_word = spell_check(misspelled_word, CORPUS)
```

Output

```
Candidate Words Evaluation:
--------------------------------------------------
Word: hello
  Edit Distance: 5
  P(X|W) (Likelihood): 0.006738
  P(W) (Prior Probability): 0.125000
  P(W|X) (Posterior Probability): 0.000842
  Operations to Transform 'woorld' → 'hello': ['Substitute w → h', 'Substitute o → e'
--------------------------------------------------
Word: world
  Edit Distance: 1
  P(X|W) (Likelihood): 0.367879
  P(W) (Prior Probability): 0.125000
  P(W|X) (Posterior Probability): 0.045985
  Operations to Transform 'woorld' → 'world': ['Delete o']
--------------------------------------------------
Word: word
  Edit Distance: 2
  P(X|W) (Likelihood): 0.135335
  P(W) (Prior Probability): 0.125000
  P(W|X) (Posterior Probability): 0.016917
  Operations to Transform 'woorld' → 'word': ['Delete o', 'Delete l']
--------------------------------------------------
Word: would
  Edit Distance: 2
  P(X|W) (Likelihood): 0.135335
  P(W) (Prior Probability): 0.125000
  P(W|X) (Posterior Probability): 0.016917
  Operations to Transform 'woorld' → 'would': ['Substitute o → u', 'Delete r']
--------------------------------------------------
Word: work
  Edit Distance: 3
  P(X|W) (Likelihood): 0.049787
  P(W) (Prior Probability): 0.125000
  P(W|X) (Posterior Probability): 0.006223
  Operations to Transform 'woorld' → 'work': ['Delete o', 'Substitute l → k', 'Delete
--------------------------------------------------
Word: wool
  Edit Distance: 2
  P(X|W) (Likelihood): 0.135335
  P(W) (Prior Probability): 0.125000
  P(W|X) (Posterior Probability): 0.016917
  Operations to Transform 'woorld' → 'wool': ['Delete r', 'Delete d']
--------------------------------------------------
Word: whirled
  Edit Distance: 3
  P(X|W) (Likelihood): 0.049787
  P(W) (Prior Probability): 0.125000
  P(W|X) (Posterior Probability): 0.006223
  Operations to Transform 'woorld' → 'whirled': ['Substitute o → h', 'Substitute o →
--------------------------------------------------
Word: curled
  Edit Distance: 4
  P(X|W) (Likelihood): 0.018316
  P(W) (Prior Probability): 0.125000
  P(W|X) (Posterior Probability): 0.002289
  Operations to Transform 'woorld' → 'curled': ['Substitute w → c', 'Substitute o → u
--------------------------------------------------
Final Correction:
--------------------------------------------------
Misspelled Word: woorld
Corrected Word: world
--------------------------------------------------
```

Result

The system successfully corrected the misspelled word **"woorld"** to **"world"** by identifying the candidate words, calculating the posterior probabilities using Bayesian principles, and evaluating the minimum edit distance.