English Auto-correct in Python

Introduction

Make a python program to automatically detect the spelling mistakes and correct it.

Objective is to implement the auto correct algorithm to find the misspelled word and give the most probable correct words list.

Abstract of Past Similar Approaches

1. <https://github.com/somyajain99/english-autocorrect>

* In this repository it calculates the given word with all the other words vocabulary and calculate the minimum edit distance.
* Print only the words that are smaller than edit distance of one or smaller.

1. <https://github.com/filyp/autocorrect/tree/master>
   * autocorrect/typos.py

constructor generates slices of the word. This class gives possible slices of given word. By insertion, deletion, transpose, replaces,

each method generates possible typographical error.

* + - autocorrect/word\_count.py

def get\_words: this extracts the word based on each language specific regx pattern. It removes capitalized words that appear after punctuation or at the beginning of a line and yields all remaining words that match the language's word regex.

def count\_words: Extracts words from a file based on the given language and encoding. Counts the occurrences of each word. Counts the occurrences of each word. Sorts the words by frequency in descending order.

* + - autocorrect/constants.py

word\_regexes: this dictionary gives regx to extract words from each language from this regx we can get only the words that are matching only to the specific language.

alphabets: this dictionary gives regx to extract each matching letters of any given language.

* + - autocorrect/\_\_init\_\_.py

class ProgressBar: this will show the progress of downloading element

def load\_from\_tar: this will download the dictionary from the internet. This downloads the frequency dictionary from the internet. Downloads the file file\_name="word\_count.json"

class Speller:

self.nlp\_data: this is frequency dictionary in previously generated by text corpus.

def existing: this function returns only the word that are in the frequency dictionary

main logic

* + - 1. The method first checks if the exact word exists by calling self.existing([word]).
      2. If it doesn’t exist, it checks for typo candidates using w.typos().
      3. If still no valid candidates are found, it checks for more complex typo candidates (in the non-fast mode) using w.double\_typos().
      4. If nothing valid is found even after all this, it returns the original word as a fallback.

Minimum Edit Distance Algorithm

Given two string s1(target) and s2(source) this algorithm calculates the minimum edit distance to convert source to target

1. Recursion approach

min\_edit(stay,play)

Insert(p)`

Delete(p)

replace(p)

min\_edit(pstay,play)+1

min\_edit(tay,play)+1

min\_edit(ptay,play)+2

Insert(l)`

Delete(s)

replace(s)

min\_edit(plstay,play)+1

min\_edit(ptay,play)+1

min\_edit(pltay,play)+2

…

… … …

1. DP table approach

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | 0 | p | l | a | y |
| 0 | 0 | 1 | 2 | 3 | 4 |
| s | 1 | 2 | 3 | 4 | 5 |
| t | 2 | 3 | 4 | 5 | 6 |
| a | 3 | 4 | 5 | 4 | 4 |
| y | 4 | 5 | 6 | 5 | 4 |