Spell Checker

CS6370- Natural Language Processing
Assignment -I

Abil N George [CS13M002]
Bharadwaj J [CS13M058]
Anil Kumar [CS13M004]

Spell Checker

There are several approaches for designing a spell checker. Each approach has its own merits and demerits. The aim is to develop a spell checker, with maximum correctness while attempting to minimize the execution time.

Specifications

- **Programming Language:** JAVA
- **Dictionary used:** Wordnet 3.1 (for checking correctness in input)
- **Training Corpus:** Brown Corpus (for finding replacement words)

Assumptions

- All valid words are present in the dictionary.
- Noisy Channel Assumption: A misspelt word may differ from the actual word by a maximum edit distance of 3
- 2 -order Markovian assumption: A word in a sentence may depend upon at most 2 previous words.
- To improve efficiency we assume a word also depends upon next word.

Procedure

The following steps are performed:

1. Training
2. Input parsing
3. Misspelt word detection
4. Generation of valid suggestions.
5. Scoring of the suggestions based on N-Gram probability
6. Output

1. Training

Training involves the following steps:

- The Brown corpus is read.
- Individual words are parsed form the corpus and part of speech tags are removed.
- The words and their respective number of occurrences in the corpus are stored as key-value pairs.
- The same procedure is repeated for bigrams and trigrams.
- The trained data are stored as serialized objects for ease of access.

2. Input parsing

The input will be in the form of paragraphs. Hence, it requires the following modifications:

- Sentences are appended with separate markers used in place of punctuation marks.
- Each sentence is split into words, which are checked for spelling errors

3. Misspelt word detection

- Scan the list of words and check whether the word is available in the dictionary.
- If not found, capture the word as well as the previous 2 words in that sentence and also the word following it.
- If it is the first word, the markers are added in place of the previous words.
- Similarly markers are added instead of the word following the last word.

4. Generation of valid suggestions

For each misspelt word the following actions are performed:

- Based on the value of a pre-initialized minimum edit-distance, a list of valid suggestions are obtained by applying substitutions, additions, deletions and transpositions.
- From these newly formed words, the invalid words are filtered out by checking the corpus.
- The valid suggestions, along with their respective scores are stored as key-value pairs.

5. Score Calculation

- For each valid suggestion, the previous two words are extracted.
- Using these, a set of bigrams and trigrams are generated along with the original word.
- The probability of the bigram and trigram are computed, based on their number of occurrences determined during training.
- Each of these probabilities is assigned a weight.
- The weighted sum of probabilities is normalized and saved as the score of the suggested word.
- The list of suggestions is then sorted according to the score.

6. Output

- Each misspelt word is output along with its suggestions and their respective scores in percentage.

Results

- The program was executed repeatedly for a different set of sentences.
- Mean Reciprocal rank measure was used to measure the correctness.
- The correctness varies from document to document.
- If several words can be obtained by slight modifications to the misspelt word, then the rank of the actual word may not be the highest.
- However, in other cases a 90% accuracy can be achieved.

Failure cases

- If the misspelt word differs from the actual word by an edit distance of more than 3.
- If space is omitted between two correctly spelled words. The application considers it as a single word and tries to find a closest match which is always incorrect.
- If the misspelt word is dependent on a word that occurs 3 words before it or after it, then the actual word might get a low rank.

Advantages

1. Single Dictionary Lookup per word

- Since dictionary lookups can be quite expensive, the program performs dictionary lookup only once for each word, i.e for detecting the misspelt word.
- Corrections are performed on the misspelt word using trained Brown corpus data.

2. Multi-threading and Synchronization

- Once a misspelt word is detected, a list of valid words can be obtained by performing insertions, deletions, transpositions and substitutions.
- These 4 operations on the misspelt word, are performed in parallel using multi-threading and hence reducing the execution time considerably.

3. Serialization of Data

- Once the data are trained the results are serialized and stored as objects rather than text files.
- This considerably reduces the data access time

Limitations

1. Pre-initialized Minimum Edit Distance

- The program requires pre-initialization of the minimum edit distance between the misspelt word and suggested words.
- The program generates all possible suggestions that differ from the misspelt word by not more than the pre-initialized value of edit-distance and ranks them based on the context.
- An ideal value of 3 minimum edit distance would be ideal, as the probability of the correct word differing from the misspelt word by more than 3 is very minimal.

2. N-Grams limited to Trigrams and Bigrams

- While using the N-Gram probability to assign scores the suggestions, a maximum of 3-grams (trigrams) have been considered and higher grams will not be evaluated.