Spelling Checker

Table of contents

The NLP Suite includes a range of spelling checkers: The NLTK unusual words package, A spelling checker via the Python package SpellChecker, and a purposefully built, Java spelling checker (https://www.nltk.org/book/ch02.html).

Python NLTK unusual/misspelled words

The Python NLTK package (Natural Language Tool Kit) provides a function that compares the words in your corpus against a large body of benchmark work from the NLTK corpora; any difference is listed as either an unusual word or a misspelling.

Java Spelling checker (against Wikipedia)

We have created a purposeful spelling checker in Java. The script scans all the text files inside an input folder, and automatically detects and corrects any spelling errors producing a new set of corrected files inside a new folder.

The spelling checker is able to detect the most common typos. Its corrections are based on a right-wrong spelling mistakes from Wikipedia (http://norvig.com/ngrams/).
It also makes use of a frequency distribution of words, calculated on about 3000 newspaper articles, therefore it privileges the most frequent words as candidates for the corrections.
To choose with even more accuracy the best correction, the tool includes also a list of the most common 2-Grams for English language (http://www.ngrams.info).
The recognized words are contained in the dictionary from SCOWL (and Friends) (http://wordlist.aspell.net).

CoreNLP tokenizer

The tool uses the Stanford CoreNLP toolkit (http://stanfordnlp.github.io/CoreNLP/) to scan the input text files and tokenize their content. Each token is then searched in the dictionary; if it's not present there, the spelling mistakes list taken from Wikipedia is searched for a correction. If there is no match in the previous correction list, the dictionary is scanned in order to find the nearest words to the current token. The most frequent word according to a language model of English produced by analyzing 3000 newspaper articles is then chosen. If there is any custom correction chosen by the user for the specific typo, that correction is the one that will be used without any further analysis.

Custom dictionary file

The user can write a custom file in order to force some corrections for some misspelled or unrecognized words (missing in the dictionary). This file must be in plain text (txt) format and has to contain two columns of words. The two columns must be separated by a tab. The first column should contain the unrecognized or misspelled word, the second column the wanted correction.

Example of the content of the custom corrections file:

```
beofe       before
bravler     brawler
croosing    crossing
florida     Florida
fofered     offered
forman      foreman
fullade     fusillade
hes         he's
inst        inst.
mIces       mices
nergo       negro
```

Example of log file:

```
data/2762-txt/Chicago Daily Tribune_10-21-1882_3.txt
(the) conducter -> conductor
(the) rond -> round
(he) spled -> pled
```

Input/output files

Input: An input folder with a set of text files (in txt format) to be checked for spelling errors. Optional: an input file containing the path of a special file containing custom corrections.
The only supported language is English.

Output: A folder inside the input folder, containing the corrected version of the text files where at least one error was found. Each one of these new files has the same name as the original file. A “log.txt” file is also created in the output directory; this file contains a log of all the corrections that have been made in each text file.

Python SpellChecker

The Python Pyspellchecker package provides a way to check a corpus for spelling errors. From the package website (https://pypi.org/project/pyspellchecker/) we read: “Pure Python Spell Checking based on Peter Norvig’s blog post on setting up a simple spell checking algorithm. It uses a Levenshtein Distance algorithm to find permutations within an edit distance of 2 from the original word. It then compares all permutations (insertions, deletions, replacements, and transpositions) to known words in a word frequency list. Those words that are found more often in the frequency list are more likely the correct results.”

This spelling checker is not yet implemented in the NLP Suite. Watch this space…