CS6370: Natural Language Processing
Spell Check Assignment

1 Problem statement

All of you must have experienced the power of the Google Spell Checker. This is an opportunity to get a hands-on experience into building such a system. The assignment consists of 3 parts:

- **Word spell check** - standalone erroneous words are given and you are supposed to suggest corrections.
- **Phrase spell check** - words present in the phrases need to be checked for spelling errors and corrected.
- **Sentence spell check** - entire sentence needs to be checked for spelling errors.

Only for **Word spell check**, you may assume that the incorrect word is not present in the dictionary. This may not be the case in phrases and sentences. For example, consider the following query: "peace of cake ". Here, the incorrect word *peace* would be in the dictionary. But the correct replacement, *piece*, needs to be suggested. However, you can assume that there will be only one incorrect word in a query (phrase or sentence).

The code and the executable of these three tasks form the basis of the assignment. You will be provided with resources (see Section 2) and a sample dataset of typos (for all the three tasks)

Think of the various issues that might come into play when you make your spell checker. How much of distortion would your program be able to recover from? Can you intelligently prune down the search space of candidate replacements? For the **Phrase spell check** and **Sentence spell check**, the context is important. You can try different things to improve your spell checkers like part-of-speech tags or phonetic algorithms. Remember that good performance on the sample dataset is a necessary but not sufficient condition for your algorithm, since the results on the test data might still surprise you. Think of inputs different from the sample dataset that might be given to a standard spell checker.

2 Resources

You must use the following resources for designing your spell check program:

- The expected minimal dictionary is: `http://norvig.com/ngrams/count_1w100k.txt`
- The following link provides the frequencies of various bigrams and trigrams - `http://norvig.com/ngrams/`.
- You can use the free frequent n-grams data based on Corpus of Contemporary American English. You need to register your email to get the resources. The URL is `http://www.ngrams.info/`.
- To improve your system, you may use any additional data or resources along with this. *(but this should be notified in advance and the TAs must approve of such use)*

In addition to that, some suggestions regarding the background knowledge and datasets to make your system better:

Papers (present in moodle)

- Kernighan, Mark D., Kenneth W. Church, and William A. Gale. "A spelling correction program based on a noisy channel model." *In Proceedings of the 13th conference on Computational linguistics-Volume 2*, pp. 205-210. Association for Computational Linguistics, 1990.
  (Note that the above paper deals only with single correction per word. But the input to your program can contain more than one corrections per word. For example, ocassion contains a deletion as well as an addition.)

- Golding, Andrew R. "A Bayesian hybrid method for context-sensitive spelling correction." *arXiv preprint cmp-lg/9606001* (1996).

Additional Datasets

- WordNet as a dictionary. WordNet is downloadable from `http://wordnet.princeton.edu/`
- The Brown Corpus (tagged with part-of-speech): `http://nltk.googlecode.com/svn/trunk/nltk_data/packages/corpora/brown.zip`
- Reuters dataset: `http://www.daviddlewis.com/resources/testcollections/reuters21578/reuters21578.tar.gz`

For any other resource(s)/help that may be required, you may approach the TAs.

3 Evaluation

For **Word spell check**, the score will be given based on the cardinality of the intersection of the suggestions and the expected corrections.

For **Phrase spell check** and **Sentence spell check**, the evaluation will be done using the Mean Reciprocal Rank measure. Please go through the url `http://en.wikipedia.org/wiki/Mean_reciprocal_rank` for more information.

Consider the following example of evaluation with Mean Reciprocal Rank Measure.

Incorrect Input sentence : The departments of the institute offer corses, conducted by highly qualified staff.
Incorrect Word : corses.
Correct word : courses.

Depending on the ranking of the suggestions, Mean Reciprocal Rank will differ.

Output #1 : courses, horses, corset - correct word is at rank 1, so Mean Reciprocal Rank is 1.
Output #2 : horses, courses, corset - correct word is at rank 2, so Mean Reciprocal Rank is 1/2.
Output #3 : horses, corset, courses - correct word is at rank 3, so Mean Reciprocal Rank is 1/3.
Output #4 :horses, corset, scores - correct word not present, so Mean Reciprocal Rank is 0.

The MRR for all the test cases will be summed up to get a measure of the performance of the spell checker system you designed. The time taken by your program to execute will also be taken into consideration. Along with the code, you will also have to submit a report containing the details of your algorithm and the observations you have made.

Please note that the evaluation for the assignment is automated. You need to strictly abide by the following **guidelines**:

- **Folder structure:**

```
TeamNumber.zip (For e.g., Team10.zip)
............TeamNumber/
....................Report_TeamNumber.pdf (E.g., Report_Team10.pdf)
....................src/ (Contains all the source codes)
....................bin/ (Contains all executables generated by the Makefile)
....................data/ (Contains all data required for the programs to run)
....................Makefile
....................README (All packages dependencies should be mentioned
                            along with the instructions to run your program
                            on the terminal)
```

- **Output format:**

  The output must be in the following specific format:

  query ⟨**tab**⟩ **suggestion1** ⟨**tab**⟩ **score** ⟨**tab**⟩ **suggestion2** ⟨**tab**⟩ **score**
  query2 ⟨**tab**⟩ **suggestion1** ⟨**tab**⟩ **score**

Here, *query* is the incorrect word and *score* refers to the values based on which you have ranked your suggestions. For **Word spell check**, you can print a maximum of 10 suggestions per query in descending order of relevance in the above format. For **Phrase spell check** and **Sentence spell check**, you are allowed to print 3 suggestions.

It is very important that the submissions follow the guidelines specified. Only programs that are bug-free and produce sensible results will be evaluated by early October. The evaluation of remaining programs will be deferred till the end of the semester, with a heavy penalty.

Plagiarism in any form - report or code - is intolerable. Copying contents verbatim from any paper or even references is strictly not allowed. If you are found to engage in plagiarism, it will be treated very seriously and will result in a U grade (irrespective of other course evaluations). This may also be forwarded to the Dean, Academic Courses for further action. All your project reports and codes will be verified using plagiarism detector tools.

Intermediate deadlines for the assignment are as follows:-

- Sept 20 - Word Spell Check
- Sept 30 - Phrase and Sentence Spell Check
- Oct 4 - Report and Code Submission

The final deadline has been set at $\mathbf{4}^{th}$ **October 2015 23:55**. This is a **hard deadline and non-negotiable**.