project description

the purpose of this project is to (i) apply the soundex code and noisy channel model for spelling correction and ranking corrected keyword queries based on the probability distribution, respectively, and (ii) generate snippets for the results retrieved for each corrected query.

1 spell checker

to identify and correct spelling mistakes in a user’s query you are required to

1.1. implement the soundex code for spelling correction. (the pseudo-code for the soundex code is given in the textbook as well as on slide #11 in the lecture notes of chapter 6). you must use the dictionary posted under http://students.cs.byu.edu/~cs453ta/projs/dictionary.txt for mapping correct words to codes. also, you must use the implementation of the edit distance algorithm available at http://students.cs.byu.edu/~cs453ta/projs/edit_distance.pdf to determine all the possible corrected spellings and select the corrected words with an edit distance value of at most 2 with respect to a given misspelled query keyword.

1.2. implement the noisy channel model to determine the most appropriate word, \( w^* \), which can replace the misspelled word \( e \) in a user’s query, among all the possible words determined using the soundex code and the edit distance algorithm as mentioned in step 1.1.

\[
w^* = \arg \max_{w \in S} P(e \mid w) P(w)
\]

where

- \( S \) is the set of possible spelling corrections of \( e \), as determined in step 1.1.
- \( P(e \mid w) \) is the probability of (an erroneous) word \( e \) in place of (the correct) word \( w \), which is the proportion of the number of sessions in a query log in which \( w \) is the correction of (the misspelled word) \( e \) over the total number of sessions in a query log in which \( w \) is the correction of any misspelled word. in estimating \( P(e \mid w) \), use the provided query log posted under http://students.cs.byu.edu/~cs453ta/projs/query_log.txt.
- \( P(w) \) is the probability of a given word \( w \), which is computed as the number of occurrence of \( w \) in a given document collection over the total number of occurrences of all the words in the same document collection (not including stopwords and without stemming the words in the collection). in computing \( P(w) \), you must use the document collection, denoted wiki collection, which is the same collection used in project 1 and is available at http://students.cs.byu.edu/~cs453ta/projs/wiki.rar.

example 1 determining \( P(e \mid w) \). based on the sample query-log shown in table 1, the number of sessions in which actor is the correct spelling of axtor is two. the number of sessions in which any misspelled word was replaced by actor is three. therefore, \( P(axtor \mid actor) = \frac{2}{3} \). \( \square \)
<table>
<thead>
<tr>
<th>Session ID</th>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Movie actor</td>
</tr>
<tr>
<td>01</td>
<td>Movie actor</td>
</tr>
<tr>
<td>02</td>
<td>Award winning actor</td>
</tr>
<tr>
<td>02</td>
<td>Award winning actor</td>
</tr>
<tr>
<td>03</td>
<td>Soap opera actor</td>
</tr>
<tr>
<td>03</td>
<td>Soap opera actor</td>
</tr>
</tbody>
</table>

Table 1: Sample query log used for computing $P(\text{actor}|\text{actor})$

### 2 Snippet Generation

You are required to create a snippet for each of the top-5 results retrieved for each query given in Section 3 by

2.1. Implementing each of the following features (as shown on Slide #24 in the lecture notes of Chapter 6): (i) a density measure of query words (i.e., significance factor), (ii) the longest contiguous run of query words in the sentence, (iii) the number of unique query terms in the sentence, (iv) the total number of query terms occurring in the sentence, (v) whether a given sentence is the 1st or 2nd line of the corresponding document, and (vi) whether a given sentence is a heading.

2.2. Implementing three additional features of your own choice.

Note that in generating the snippet of a given document $D$ you must select two sentences in $D$ for which the combined score of the features (the six provided on the lecture notes and the additional three of your choice) is the highest. The combined score is computed by adding the scores for each of the features discussed in Step 2.1 along with the scores for the three features of your choice. Also, you must bold in the snippet the keywords that belong to the same stem class as keywords in the (corrected) query. In retrieving and ranking the top-5 documents from the Wiki collection, you must use the query processing and ranking program that you have implemented for Project 1.

### 3 Query evaluation

For each of the five queries $Q$ given below, you are required to (i) correct the spelling of $Q$, using the spell checker that you have implemented in Section 1, (ii) retrieve and rank the top-5 documents\(^1\) with the highest ranking score with respect to $Q$ using your Project 1, and (iii) show the corresponding snippet (as created according to the instructions given in Section 2) for each retrieved document. (See Figure 1 for the expected output for a sample query.)

1. sentenced to prison
2. open court case
3. entertainment group
4. tv actor
5. scheduled movie screening

\(^1\)If a given query retrieves less than five relevant documents, then it is acceptable to show the snippets of the retrieved documents up till the last one.
4 Grading Criteria

The assignment is worth 150 points, and the breakdown of the point distribution is given below.

- Implementing the spell checker in Step 1 is worth 75 points.
- Generating snippets for the retrieved documents in Step 2 is worth 75 points.

To receive credit for this project assignment, you must pass off your program during the TA office hours by the due date.