IR Assignment 4: Spell Checking and Auto complete feature for the search engine developed using Solr

Agenda:

The agenda of this assignment is to provide spelling correction and auto complete functionalities to the search engine that was developed in the previous assignment

1. Steps you followed to complete this assignment. Include the details of what tools and techniques you used to implement spelling correction and autocomplete.

Details:

Tool: Solr 5.3.1
School Indexed: USC Gould School of law
Root URL: http://www.gould.usc.edu/
Spelling Correction Algorithm: Php version of Peter Norvig's Spell Corrector.
Auto Complete: FuzzyLookupFactory feature of Solr/Lucene.

Steps to be followed:

1. **Installations and indexing the data**
   - Solr 5.3.1 was installed according to the instructions provided under assignment 3 guidelines
   - XAMPP software was installed to host the apache server
   - **bin/solr create –c myexample** command was used to create the core
   - Once the core was created, the file ***managed-schema*** was changed to ***schema.xml*** and the changes were made to the file as provided under assignment 3 guidelines
   - Indexing of the crawled data was done as follows: **bin/post –c myexample <path to the folder>**

2. **Adding a suggest component**

- Solr is configured to return upto 5 auto complete suggestions. A search component is added to solrconfig.xml as below:

```
<searchComponent class='solr.SuggestComponent' name="suggest">
    <lst name="suggester">
        <str name="name">suggest</str>
        <str name="lookupImpl">fuzzyLookupFactory</str>
```

```
                    <str name="field">_text_</str>
                </lst>
            </searchComponent>
```

- We use the "_text_" field to obtain terms for suggestion.
- We add a requestor as follows:

```
<requestHandler class='solr.SearchHandler' name="/suggest">
        <lst name="defaults">
                <str name="suggest">true</str>
                <str name="suggest.count">5</str>
                <str name="suggest.dictionary">suggest</str>
        </lst>
        <arr name="components">
                <str>suggest</str>
        </arr>
</requestHandler>
```

3. Autosuggest Feature

- The user enters the query in the search box and the keyup component of jQuery monitors the keystrokes
- When the keyup event occurs an AJAX request is sent to autofill.php with the query string as the parameter
- Autofill.php then sends the query to SpellCorrector.php and the AutoComplete feature of Solr
- When the results are returned, we filter the results in JSON format and return this formatted result to index.php to be displayed

4. Spell Corrector Algorithm by Peter Norvig

- The spelling corrector model has to be trained specifically for Gould school of law
- The file big.txt contains a collection of words. This data is used to train the spelling corrector
- Python's beautiful soup package is used to extract the data from the crawled files
- Serialized_dictionary.txt is generated when the spelling corrector algorithm is run for the first time. It caches the training data and uses it during the later executions of the algorithm
- Spelling correction is done for each word the user enters in the query box

5. Stop Word Removal

- Solr has inbuilt support for stop words removal. Solr by default is configured to remove stop words.

- Stop words can be added to stopwords.txt which is present under conf folder.

6. **Stemming:**

- Solr's support for stemming can be enabled by adding the following line to schema.xml

  <filter class ="solr.SnowballPorterFilterFactory"/>.

2.Analysis of the results:

Spelling Correction:

When a user enters "facultie", the auto correct feature suggests faculties

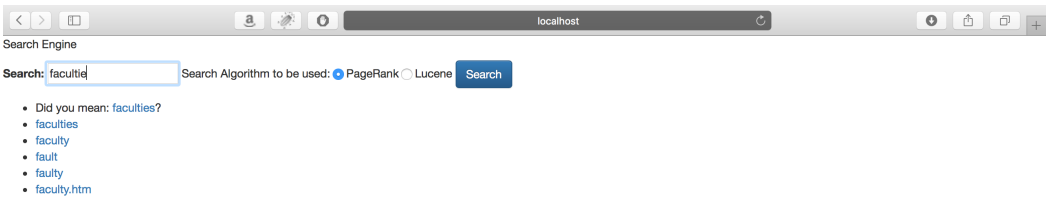
facultie -> faculties

We might think that Faculty also could be the correct word but Norvig's Algorithm suggests "faculties". This is because, an addition of one letter is needed to change facultie to faculties whereas as it requires two deletions and an addition (it can also be viewed as one deletion and a transposition) to change facultie to faculty. Also, since the algorithm considers Baye's Theorem, we have to compute the error model P(w|c).

Norvig's Algorithm uses a trivial model which says "all known words of edit distance 1 are more probable than known words of edit distance 2. A known word is a word that has been seen in the language model training data." Considering the above two factors, Norvig's Algorithm ranked faculties higher than faculty.

Another example of autocorrect is

gouldi -> gould The spelling corrector comes up with the correct spelling as the model is trained specifically from the data crawled from USC Gould School of Law



Auto Complete:

- ***event -> event, events, even, evening, eventid***
- ***require -> require, required, requirements, requires, requirement***