Adding Spell Checking, AutoComplete and Snippets to Your Search Engine

Steps followed:

1. Spell check

* For spell check I have used Norvig’s Spell check program that is available at url:

https://www.phpclasses.org/package/4859-PHP-Suggest-corrected-spelling-text-in-pure-PHP.html#download

* This spell check program uses “big.txt” as the input. To create “big.txt”, we use Apache Tika. A java program using HTML parser was written to parse html files. When user enters a query, the input is sent to the SpellCorrector::Correct($query) method, which returns the corrected output. In case of spelling errors, the input and the output do not match. In which case, a suggestion is provided with the link that directs to the page displaying results to the spell corrected output.

2. Autocomplete

* Autocomplete was implemented using the Solr/Lucene’s FuzzyLookupFactory feature. To configure it, the following changes were made to solrConfig.xml

<searchComponent class="solr.SuggestComponent" name="suggest">

<lst name="suggester">

<str name="name">suggest</str>

<str name="lookupImpl">FuzzyLookupFactory</str>

<str name="field">\_text\_</str>

<str name="suggestAnalyzerFieldType">string</str>

<!-- optional elements with defaults

<str name="distanceMeasure">org.apache.lucene.search.spell.LevensteinDistance</str>

<str name="accuracy">0.5</str>

-->

</lst>

</searchComponent>

<requestHandler class="solr.SearchHandler" name="/suggest" >

<lst name="defaults">

<str name="suggest.dictionary">suggest</str>

<str name="suggest">true</str>

<str name="suggest.count">5</str>

</lst>

<arr name="components">

<str>suggest</str>

</arr>

</requestHandler>

* I used jQuery for the front end part of the autocomplete. Whenever a single character input is applied on the input text box, an ajax call is made to solr with suggest query.
* The response is a json which contains the suggestions for the query keyword, which is used to display the suggestions in the form of a drop down.

Snippet:

* We need to convert the html files to txt files, Jsoup parser is used for this purpose.
* Next, for each result retrieved, we need to find the first occurrence of the specified string query in the text document. We then extract required amount of data to be displayed as snippet.
* Misspelled words:

Misspelled words -> corrected words

1. hurrcani -> hurricane
2. donuld tramp -> donald trump
3. whitd hause -> white house
4. britnay spebrs -> britney spears
5. desart -> desert

* Auto-completion

First few characters -> auto-completed suggestions

1. hurr -> hurricane, hurt, hurricanes, harry, harris
2. mint -> mint, minimum, monthly, min, month
3. grea -> great, green, grand, grey, greater
4. jock -> jock, jackson, jack, joke, jokes
5. temp -> temp, tampa, temperatures, temperature, temporary