Topic Extraction

Topic extraction is possibly the most unique feature offered by our text editor. The aim of this functionality is simple- to try and find out what the user is typing and subsequently provide useful links concerning it to the user.

Our text editor broadly classifies a document typed by a user into the following buckets:

1. Letter
2. Resume
3. Code
4. Essay

The methodology used in detecting the first three types of documents is the same. The detection of essays involves the function of keyword extraction or heading extraction.

Detecting letters, resumes and code:

Letters, Resumes and Code in various programming languages have certain characteristic features or *keywords*. We maintain a set of keywords for each of the three types of documents- letter, resume and code. We then scan the document and maintain a list of words that appear in the document. Each document type is associated with a hot count. Whenever a word in the document uniquely matches a tag, the hit count of the appropriate document is increased by 1. If the hit count of a document is less than 3, it is ignored. The document with the highest hit count is detected as the document being typed. If all of the documents have their associated hit count, the document is then assumed to not fall under any of the three types. We then proceed to detecting the topic of the essay being typed.

Essay topic detection using keyword extraction or Header extraction

If the document is found to not fall under any of the three document types (letter, resume, code), we assume that the document being typed is an essay and try to detect the topic of the essay. We first scan the document for a heading; if it exists we extract the same and provide links to it. If no headings were detected, we proceed by performing keyword extraction.

Keyword Extraction

The keyword extraction methodology used in the text editor is a slight modification of the tf-idf method of obtaining the keyword, largely used in related areas of NLP. We define a function f that weighs the term positively for the number of times the term occurs within the document, while also weighting the term negatively relative to the corpus(Brown Corpus in this case). Formally:

*f*= where is the frequency of term t in the current document d and is the frequency of the term t in the corpus c. The terms are then arranged in decreasing order of their corresponding f values and the links to the top few keywords are provided.

Links are provided to the topics/keywords using Java’s Gson Module.