Adaptive Information Systems: HW2

(gar43@pitt.edu)

Gangeshwari Rajavelu

Tools Used: Php, Xampp and Apache Server

Interesting RSS Feed used: RSS Feed from Eventur

Second RSS Feed: Can input any other RSS Feed.

(Please run the homepage.php on the server, and having all the rest of the files included in the same directory)

The program works as follows:

1. There are 2 input fields, one for building the user profile (in this case Eventur RSS feeds URL is provided, but it can be any other URL as well, as long as there would be sufficient textual content to extract 500 frequent stems) and the other field takes as input any other URL for which similarity of contents is to be compared and calculated.
2. The RSS feeds from the Eventur website is parsed completely where every item's description tag contents is treated as a document.
3. All the words in the document are extracted
4. The html tags are stripped using functions in php
5. The stop words are removed from the content (by comparing with an array of stop words that is read from a text file containing the list of stop words obtained from the web).
6. The words are then normalized, turning into lower case.
7. The words are then stemmed using Porter Stemmer algorithm(This implementation in php was used for the homework).
8. The stems and their corresponding frequency in the collection are stored in a vector after indexing using array functions.
9. The user profile is built into a vector by extracting the 500 top frequent stems and the values stored for each stem are the key which is term itself and the value which is given as
   Key = term
   Value = frequency of every stem/highest frequency term in the collection (ranges between 0 to 1)
10. When the second url of RSS is passed, the document vectors are generated for every item read in the feed.(stop words removal, normalizing, stemming performed as above)
11. This is followed by calculating the cosine similarity between every document vector and the user profile vector as follows:

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

12. So also, for every document another frequency value (co-occurrence) for the content as well as the titles of every item is calculated which is
    Content – co occurrence percent
    (No. of words in the document that exist in the user profile vector/No. of words in the user profile vector)
    Title– co occurrence percent
    (No. of words in the title that exist in the user profile vector/No. of words in the user profile vector)

13. The above gives a measure of co occurrence of the words in the document vector and the user profile vector.
14. Finally the rank for every document is calculated by adding the cosine similarity value obtained in step 11 and co occurrence value measured in step 12. This value ranges between 0 and 1 again which are then ordered in descending and the corresponding documents with higher values are given higher rank and those with lower values are given with lower ranks.
    The overall rank calculation for every document is as follows:

    $rank = $cosine_similarity + $cooccurence_percent + $title_cooccurence_percent;

This ranking is interesting as it takes into account the frequency of co occurrence of words both in the title and the description of the items in the rss feeds and compares their occurrence in the user profile of interest. Further the ranking considers the cosine similarity of all the terms in the document with that of the user profile vector which gives a measure of the relative orientation of the text in the document in comparison to the text in the user's interests.