Recommender Model for a Document to be Read Next

Kushal P
2014B4A70624G
BITS Pilani, Goa Campus

Prashant Pandey
2014A7PS0100G
BITS Pilani, Goa Campus

Keshav Prasad
2015A7PS0079G
BITS Pilani, Goa Campus

Group number: 6

***Abstract*—On the Internet, where the number of choices is overwhelming, there is need to filter, prioritize and efficiently deliver relevant information in order to alleviate the problem of information overload, which has created a potential problem to many Internet users. Recommender systems solve this problem by searching through large volume of dynamically generated information to provide users with personalized content and services. In this paper, we combine two most efficient methods of recommendations namely collaborative filtering and content based filtering to form a hybrid system.**

***Keywords*—Recommendation, User Profile, Feature Vector, Cosine Similarity.**

I. Collaborative Filtering

Collaborative Filtering (CF) is making automatic predictions (filtering) about the interests of a user by collecting preferences or taste information from many users (collaborating). The underlying assumption of the CF approach is that if a person A has the same opinion as another person B on an issue, then A is more likely to have Bs opinion on a different issue than that of a randomly chosen person.

- look for the most similar researcher.
- recommend the most similar paper from that researcher.

II. Content Based Filtering

Content-based filtering, also referred to as cognitive filtering, recommends items based on a comparison between the content of the items and a user profile. The similarity of the user profile with each of the recommendable papers are calculated and the most similar among these is recommended. Because content-based recommendations rely on characteristics of objects themselves, they are likely to be highly relevant to a users interests. This makes them especially valuable for organizations with massive libraries of a single type of content.

III. Feature Vector Construction

The terms in a paper along with the term count is extracted and each of the term count value is divided by the total number of words in the paper. The resulting vector is stored in a text file.

IV. User Profile

The description of what information is of interest to a user is commonly referred to as that users profile. The feature vector of a researchers paper, its references and citations are used to calculate the user profile for that researcher.

A. User Profile Construction

As research interests of researchers change over time, the user profile construction process must model this. We capture this by using a tunable forgetting factor that assigns less weight to papers published further in the past. The feature vector of the most recent paper is added with the featue vector of the other papers multiplied by a forgetting factor 0.9 to the power of recency. Recency is 0 for the most recent paper, 1 for the next and so on.

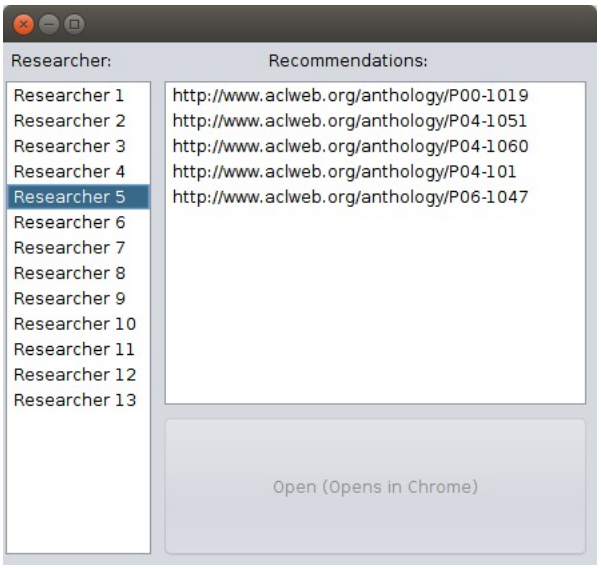

fig 1:GUI created for recommendation

V. Methodology

A. Files Constructed

- smatrix.txt - Contains the matrix of the order 13 X 597 where 13 is the number of researchers and 597 is the number of recommendable papers. The matrix stores the value 0 if the Researcher has not read the paper and 1 if he has.
- userprofiles.txt - Contains the user profile values for each researcher obtained by the method explained before.
- mat.txt - Contains a matrix of the order 13 X 13 where each coordinate [r1,r2] represents the similarity value of a researcher r1 with the researcher r2 excluding himself.
- 597.txt - Contains a matrix of the order 13 X 597. coordinate [r1,r2] represents the similarity value of a researcher r1 with the reccomendable paper r2.

- parser.java - Extracts words from a pdf file, removes stopwords and creates feature vector for the pdf.
- stemmer.java - Implements porter stemmer algorithm after stopwords have been removed by parser.java.
- Userprofilesenior.java - Constructs user profile values for each researcher and stores them in a text file
- recom.cpp - File contains the C++ code to recommend the paper to the researcher whose number is given as the input.
- Recommender.java - Calls recom.cpp on clicking the researcher number on the GUI and displays the recommended papers. The researcher is redirected to the page containing the paper on clicking any of the recommended papers.

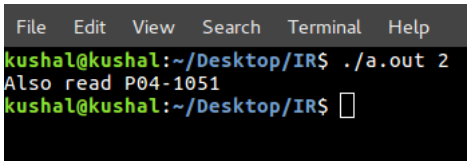

fig 2:output provided by recom.cpp for researcher 2

B. Cosine similarity

The cosine similarity between two vectors (or two documents on the Vector Space) is a measure that calculates the cosine of the angle between them. This metric is a measurement of orientation and not magnitude, it can be seen as a comparison between documents on a normalized space because were not taking into the consideration only the magnitude of each word count (tf-idf) of each document, but the angle between the documents. Similarity between two feature vectors is calculated by the method of cosine similarity which involves taking the dot product of two feature vectors.

C. Recommendation Method

The matrices mat.txt and 597.txt are created using cosine similarity. The researcher index is taken as the input. The most similar researcher R to that researcher r is found and the paper marked as read in smatrix.txt by R ,but not by r is recommended to r(collaborative filtering). If all the papers read by R is already recommended to r or if there isn't any paper read by R, then the maximum value corresponding to r in 597.txt is computed and it's corresponding position in smatrix.txt is extracted. If the value is 0, then it is marked as 1 and recommended else the next most similar paper is recommended using the same method explained above(content based filtering). The recommended feature vector file has the paper ID as its file name. The file name is extracted and redirected to the link containing the paper. Thus a hybrid model which is a combination of collaborative filtering and content based filtering is implemented in this attempt of ours to provide a better model for document recommendation.

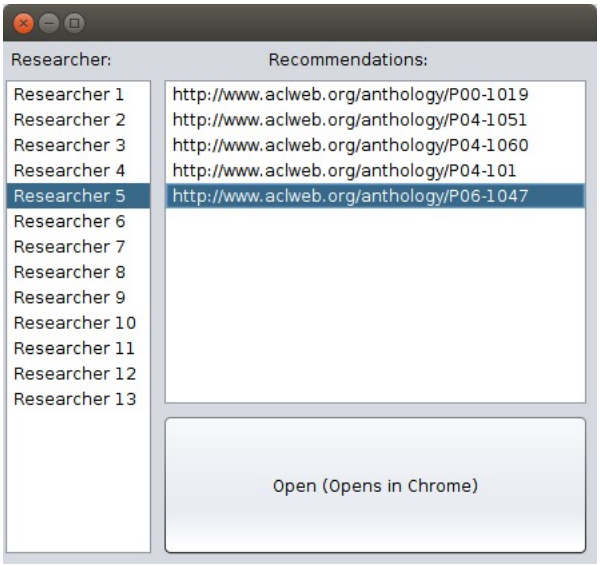

fig 3:recommendation on the GUI

VI. Conclusion

The papers recommended for each researcher are compared and found to be very similar most of the time. We have used this as our mode of evaluation as the dataset provided does not include any evaluable data. With this, we conclude that the papers for each researcher is recommended with high accuracy.

VII. Acknowledgements

We would like to extend our gratitude to Dr Swati Agarwal, Instructor In-charge of the course Information Retrieval, for her guidance and support.

References

[1] Kazunari Sugiyama,Min-Yen Kan, *Scholarly Paper Recommendation via Users Recent Research Interests*, 2010.

[2] Kazunari Sugiyama , Min-Yen Kan, *Towards Higher Relevance and Serendipity in Scholarly Paper Recommendation* ACM SIGWEB Newsletter, 2015.

[3] Kazunari Sugiyama, Min-Yen Kan, *A Comprehensive Evaluation of Scholarly Paper Recommendation Using Potential Citation Papers* International Journal on Digital Libraries,2015.