IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727
PP 60-72
www.iosrjournals.org

Personalized Document Similarity and Search

[1]Vaibhav Khatavkar, [2]Rohan Karhadkar, [3]Trupti Pimparkar, [4]Shivraj Wabale
[1]Assistant Professor, [2;3;4]B.Tech. IT, Students (Research Scholar), Department of Computer Engineering and Information Technology College Of Engineering, Pune, MS, India

Abstract: *In todays world with ever increasing volume of text resources over digital libraries and internet, organizing these resources with one's likes and interests have become a practical need. This paper will experiment two well known algorithms for calculating text similarity along with personalizing it to user specific interest and representing it in graphical format. To make user comfortable with the software, similar documents are represented as a node in a graph. Key contributors in finding similarity between the documents in this software are cosine similarity and concept similarity measures. Along with measuring similarity, the software modifies user concepts' weight from user submitted documents. This makes it easy to read and explore the similar documents guiding the user to widen the knowledge. Accordingly, history of the user's interest is maintained, modified and is eventually used in subsequent tasks of calculating correspondence for that particular user.*

Keywords: *Document Similarity, Bag of Words, Bag of Concepts, Personalization, Graph Theory*

I. Introduction

Document similarity is simply to find how similar two documents are. Now-a-days number of files are increasing with increase in need and interest in documentation. It is very important to get interested files among a huge set of available data files. Based on the browser cookie records, Google enables the Google Personalized Search - this paper introduces the methodologies to add another dimension to per-sonalize search for the documents. The paper also introduces the personalized similarity which adds another element to social networking sites' recommendations, which are planned on user's interest. Also, it can be very useful if software could get files according to personal interest without explicitly men-tioning it. This Simplest way of finding document similarity is by bag-of-words concept[1] i.e. by comparing common words from both files and calculating relative similarity which is done in cosine similarity. Here, the frequency of every term is calculated to find the importance of that term in the document. But the question is, "Is it sufficient to calculate similarity by considering only similar words? What if two documents have similar words, but totally different meaning?"

To check this, concept similarity comes into picture. Con-cept is the term having semantic role in the sentence[2]. It can either be a word or a phrase depending on the semantics of the sentence. Weights are assigned to these concepts according to their importance in the meaning of that sentence and in the document. In this modern age, it is very simple to create

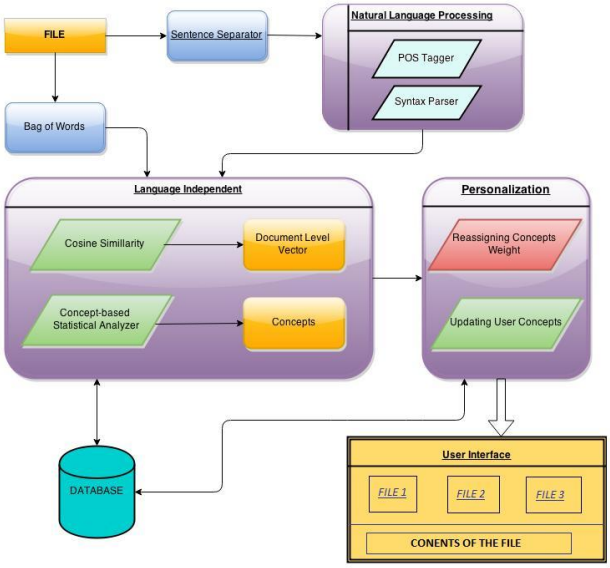

Fig. 1. Architecture

Captivating user interfaces. It is more convenient to have Graphical User Interface for the display of results and handling of events.

Explanation of important terms used in paper are as follows[2]:

Term - a word or a phrase (sequence of words) Concept - labeled term

The rest of this paper is organized as follows. Section II presents the architecture of the application. Section III introduces the cosine similarity with an illustration. Section IV introduces the concept extraction or bag-of-concept model. Section V introduces the database design. Section VI gives the actual concept of personalization in this application. Section VII gives the representation of the software in graphical user interface. The last section summarizes and suggests future work.

II. Architecture

Architecture of the software pictorially depicts the working of this system as the user submits the document; hence the flow of the application.

A. File

The user can enter text in a file, which can be in the form of a single or multiple paragraphs. The user is provided with a browse button which facilitates the user to select the desired file.

B. Sentence Separator

The Sentence Separator is a simple piece of code, which splits a paragraph in English into simple sentences. Given a string, assumed to be English text, it returns a list of strings, where each element is an English sentence. To split the paragraph into sentences, it makes use of sentence delimiters like '.', '?' and '!'. Moreover, it does its best to determine when an occurrence of '.' does not have this role of breaking the sentence (e.g. in numbers, abbreviations, URLs etc.).

The splitter is designed to work for English, but it is simple enough to adapt and customize it to other (similar) languages.

C. Natural Language Processing

1) POS Tagger: A Part-Of-Speech Tagger (POS Tagger) is a software component that takes in some language text as input and assigns each word the appropirate part of speech such as noun, verb, adjective, etc.

2) Syntax Parser: A parser is a piece of software that takes input data (text) and builds a data structure.

This structure can be in any form like a parse tree or abstract syntax tree or other hierarchical structure. This also gives a structural representation of the input and checks the correctness of the syntax in the process.

3) Shallow Semantic Parsing: Shallow semantic parsing, consisting of the detection of the semantic arguments asso-ciated with the predicate or verb of a sentence and their classification into their specific roles, is a task in NLP (natural language processing) also known as Semantic role labeling.

D. Language Independent Processing

1) Cosine Similarity: The cosine measure calculates the cosine of the angle between their corresponding word vectors to find the similarity between two documents [8][16]. Top similar documents are selected from a complete set of reuter database using cosine similarity and these documents are submitted to further step as input.

2) Concept Based Statistical Analyzer: Concepts are ex-tracted from a file and its tf and ctf values are calculated which in turn gives us the weight of each concept in a document. This step is repeated for all ten files the software received as input from the previous step. Now these ten files are sent further for computing idiosyncratic weight to a particular user.

E. Personalization

Sorting data with once likes and interest is the need of industry in which every day number of documents are pro-liferating. To sort these documents as per personal interests and likes, is really a formidable challenge and this is where personalization is taken into account. The statistical properties moving average has been used to modify the concepts' weights when new documents have been submitted.

F. Database

Learning stage of the software heavily uses database where concepts from each file are extracted and stored. They are further used in the trained stage for reassigning weights for a specific user.

G. User Interface

Graph is drawn on the user interface with submitted file at the top and three[1] similar files connected to that file with connecting lines. A single click on file name displays content of that file and double click expands tree further. That is, double clicked file is again submitted and again top three similar files are displayed.

III. Cosine Measure

The cosine measure is a bag of words concept[1], which calculates the similarity between two documents as the cosine of the angle between their corresponding word vectors[4]. It is simply the frequency of occurrence of common words from both documents as compared to frequency of occurrence of all words from both the documents. Formally, if $\tilde{d}_A$ and $\tilde{d}_B$ are the word vectors of documents d_A and d_B, their similarity is computed as[8]

$$\text{cosine}(d_A; d_B) = \frac{\tilde{d}_A : \tilde{d}_B}{|\tilde{d}_A| : |\tilde{d}_B|} = \frac{\sum_{t=0}^{V} w(t; d_A)\, w(t; d_B)}{\sqrt{\sum_{t=0}^{V} w(t; d_A)^2}\ \sqrt{\sum_{t=0}^{V} w(t; d_B)^2}}$$

Where:

$w(t, d_A)$: the number of occurrences of t in d_A represents the weight of word t in d_A that is, the tf - idf weight.

V: represents the amplitude of the vocabulary of the document collection.

This formula[8] precisely calculates the inter-document similarity. Comparable to many other measures, the cosine measure excludes connections among features. Therefore, it has been only put into the word based (bag-of-words) representation[1]. In-spite of its shortcomings, it plays a major role in the robustness of the software - even in the cases where no concepts are detected in the input document.

Now, to explain the working of cosine similarity in detail, let's take an example of two small files.

A. Example

File 1: My name is Sachin. I am from Mumbai. I study in COEP computer.

File 2 : My name is Ajay. I study in VJTI computer. My friend's name is Vijay. He is from Pune. He studies in COEP computer.

[1]Three files just for the sake of explanation, but the number can be customized for the software as per use.

B. Preprocessing of Data

Removing stop words

– words like is, am, from, of, etc.

Natural language processing

– Eg: it (pronoun) vs. IT (abbreviation for Information Technology)

Suffix stripping

– Eg: studying - study, laughs - laugh, news - news

C. Removing stop-words

File 1: name sachin mumbai study coep computer

File 2: name ajay study vjti computer friend name vijay pune study coep computer

D. Assigning weights

WORD	FREQUENCY
name	1
sachin	1
mumbai	1
study	1
coep	1
computer	1

TABLE I COSINE EXAMPLE: VECTOR d_A

WORD	FREQUENCY
name	2
ajay	1
study	2
vjti	1
computer	2
friend	1
vijay	1
pune	1
coep	1

TABLE II COSINE EXAMPLE: VECTOR d_B

$$p\ p\ jd\tilde{}_A j = 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 = 6$$

$$jd\tilde{}_B j = \ _p^p\ 1^2 + 2^2 + 1^2 + 2^2 + 1^2 + 1^2 + 2^2 + 1^2 + 1^2$$
$$= \quad 18$$

E. Calculating Cosine Similarity

Using the formula shown above,

$$d\tilde{}_A : d\tilde{}_B = 7$$

$$jd\tilde{}_A j : jd\tilde{}_B j = \quad p\ \overline{6}\ p\ \overline{18}$$

$$\text{cosine}(d_A;\ d_B) = 0{:}67$$

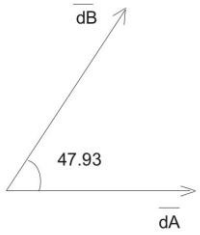

Fig. 2. Cosine Example: Angle

IV. Concept Extraction

Only cosine measure is not sufficient for the calculation of similarity between the documents. What if two documents have same concept but different text? What if two documents may describe same thing but in different manner? What if the two documents have same words, but different meanings? Let's take an example:

Document 1: Yesterday, I had a peaceful evening on the banks of the river Ganga.
Document 2: Tomorrow, I will deposit Rs. 2000 in HDFC bank.

Here, the two sentences have nothing in common. But, cosine similarity will show some false degree of similarity because of the word bank. In the first document, 'bank' means the river-bank and in the second document, 'bank' refers to a financial institution. Therefore, instead of just words, concept level extraction is also necessary and here the concept extractor comes into picture where concepts from sentences and the document are extracted and compared with the given document.

At first, sentence is passed to the Stanford Parser[2] of forming a tree of sentence by dividing sentence into sub-parts as noun, verb, etc. Here sentence is divided into many concepts according to the structure shown below: target verb: The verb on whom the sentence depends or which forms the root of a sentence tree

arg-0: The sentence argument before the target verb (the actor) argm-pnc: Purpose of the argument
arg-1: The sentence argument after the target verb (the thing acted upon)
argm-tmp: Temporal information relative to some verb argm-loc: Additional locational information relative to some verb

The number of such structures from each sentence depends on the number of target verbs in that sentence.

After this, the structure is proceed for cleaning step:

[2]The software uses Stanford Parser, but any other parser can be used. Moreover, the software can be made compatible with other languages as well using parsers of different languages.

A. Eliminating stop-words

All the trivial words like a, an, the, for, from etc. Are removed as they may add to some similarity which is not required.

B. Stemming

The words are stemmed using a popular Porter Stemmer algorithm[15]. Here, all the words are reduced to the root words. eg: laughs, laugh and laughing are the same words in different tense, so they will be reduced to a single root word laugh.

The terms generated after this step are called concepts. In this example, the stop words are removed but the concepts are shown without stemming for better readability.

Later on, the term frequency(tf) and concept term fre-quency(ctf) values are calculated. 'ctf' value represents the importance of a concept in that particular sentence while 'tf' value represents the importance of the sentence in the entire document.

tf and ctf values are calculated as follows[2]:

$$\text{tfweight} = \frac{tf_{ij}}{\sqrt{\sum_{j=1}^{cn} (tf_{ij})^2}}$$

where cn is the total number of the concepts which has a term frequency value in the document d.

In next equation, the ctf_{ij} value is normalized by the length of the document vector of the conceptual term frequency ctf_{ij} in the document d where j = 1, 2, ..., cn

$$\text{ctfweight} = \frac{ctf_{ij}}{\sqrt{\sum_{j=1}^{cn} (ctf_{ij})^2}}$$

The algorithm for extracting the concepts is as follows:

1) New document doc
2) An empty List L (L is a top argument list)
3) for each labeled sentence s in doc do
4) find all the VBN and VBD in s as c
5) for each c_i in s do
6) find the parent "S" in the tree
7) argument 1 = find the "NP" of c_i in the tree
8) temporary argument = find the "VP" before the target verb c_i in the tree
9) target verb = c_i
10) argument 2 = latter part of the sentence of the grandparent of c_i
11) add argument to L
12) end for
13) end for

Let's take an example to understand the concept extraction in detail: Feshbach said that in early January he predicted that the Dow Jones Industrial average, the most closely watched market barometer, would reach the 2400 level by April and the 2500 level by June.

This sentence is passed to a parser to form a tree of sentence, which also tags all the parts of speech. The shallow semantic parsing, used on this sentence identifies three target words or target verbs, that represent the semantic structure of the meaning of the sentence [2]. These verbs are said, predicted and watched.

Each one of these verbs has its own arguments as follows:

– arg-0: Feshbach

- – target: said
- – arg-1: that in early January he predicted that the Dow Jones Industrial average , the most closely watched market barometer , would reach the 2400 level by April and the 2500 level by June

– arg-0: he

- – target: predicted
- – arg-1: that the Dow Jones Industrial average , the most closely watched market barometer , would reach the 2400 level by April and the 2500 level by June

- – arg-0: the Dow Jones Industrial average , the most closely
- – target: watched

- – arg-1: closely watched

After this, the structure is passed for cleaning step. The terms generated after this step are called concepts. In this step, the stop words are detached and concepts are shown without stemming for better readability as follows:

1) First structure
   feshbach said
   early January predicted Dow Jones Industrial aver-age closely watched market barometer reach level April level June

2) Second structure
   predicted
   Dow Jones Industrial average closely watched mar-ket barometer reach level April level June

3) Third structure
   Dow Jones Industrial average closely watched
   closely watched

Later on, the term frequency(tf) and concept term fre-quency(ctf) values are found out. ctf value represents the importance of a concept in that particular sentence while tf value represents the importance of the sentence in the entire document.

The ctf values are shown in the table given below:

Sr. No.	Individual Concepts	(CTF)
1	early	1
2	January	1
3	Dow	3
4	Jones	3
5	Industrial	3
6	average	3
7	closely	4
8	market	2
9	barometer	2
10	reach	2

11	level	4
12	April	2
13	June	2
Sr. No.	Sentence Concepts	(CTF)
14	feshbach	1
15	said	1
16	early January predicted Dow Jones Industrial average closely watched market barometer reach level April level June	1
17	predicted	2
18	Dow Jones Industrial average closely watched market barometer reach level April level June	1
19	Dow Jones Industrial average closely	1
20	watched	4
21	closely watched	1

TABLE III CONCEPT EXAMPLE: CONCEPT-BASED STATISTICAL ANALYZER

A. Security

When a new user is created, he/she is assigned a token which is idiosyncratic to that user. First the username is appended with the id and md5 hash of that string is taken. The resultant is the token for that user as now other user with same username or id is allowed, the token will remain unique. Now, to store the password in the database, the software first appends the token with the password entered by the user and then take md5 hash value. The result now is stored in database as the password.

Now, to login, when the user enters the username and password, the database is queried for the password with entered username and stored in a temporary string. Since md5 is a one-way hashing, the software has to generate the password as done before. So, the token of the user is taken and is appended with the entered password and md5 hash is calculated again.

Now, the result is compared with the password retrieved from the database. If they match then the password is correct and the user is given access further.

B. Learning

In this example, the topic of the sentence is about closely watching the Dow Jones Industrial average levels. These concepts have the highest ctf values of 3 and 4. In addition, the concept of entire sentence which has the lowest ctf , has no significant effect on the topic of the sentence.

V. Database Design

For performing user login, username and passwords must be stored in database. Along with this, some other required information is also stored in database for future use. The class diagram is shown below:

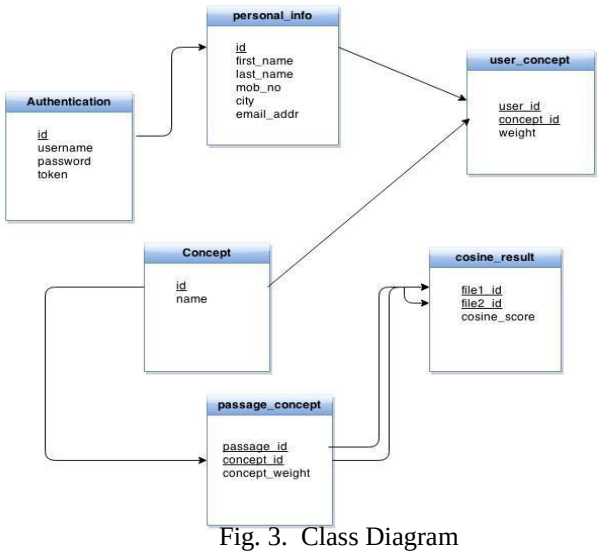

Fig. 3. Class Diagram

Before the software is launched, it goes through the learn-ing phase, where all the files are scanned and concepts are extracted, weights are assigned and then inserted in the database[17].

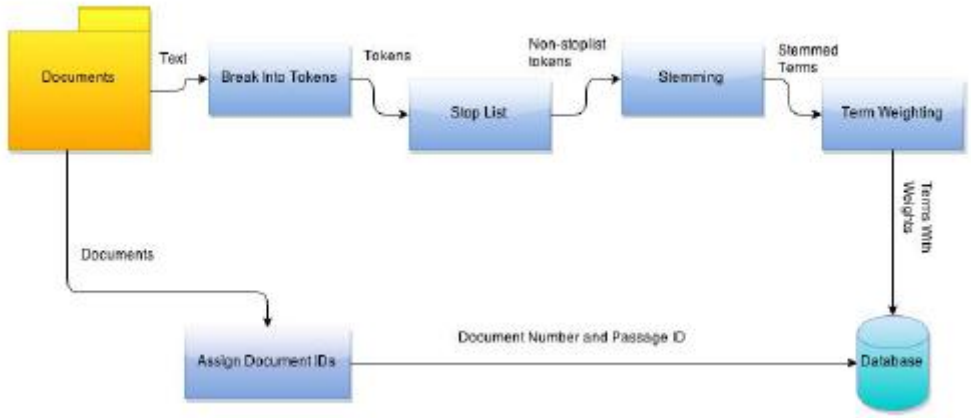

Fig. 4. Learning

VI. Personalization

Personalization is the essence of this software. It has been implemented by modifying concepts' weights such that highest priority is awarded to the latest submitted documents and lower priority is assigned to the matured documents.

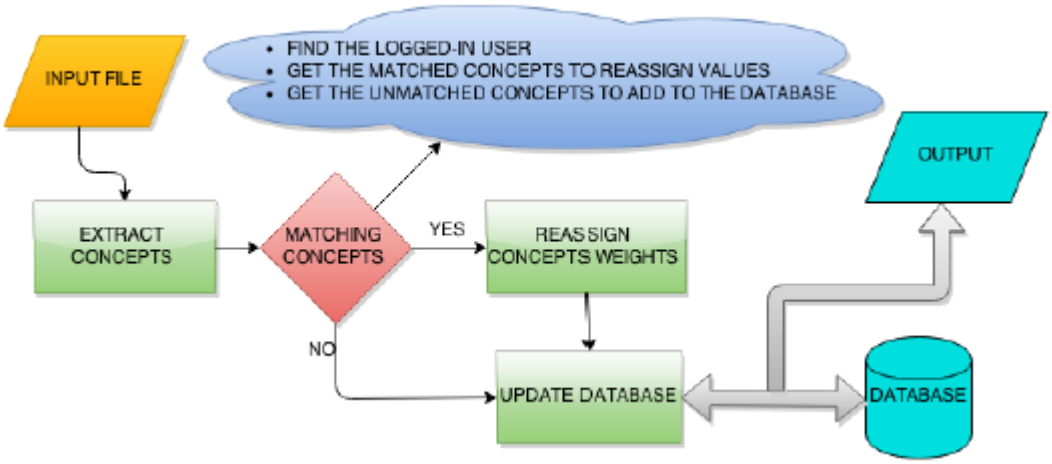

Fig. 5. Personalization

A. Exponentially Weighted Moving Average

The EWMA for a series X may be taken recursively:

$$r_1 = X_1$$

for $t > 1$; $R_t =$:$X_t + (1 \quad):R_{t\ 1}$

Where:

R_t : the value of the EWMA at 't' time period.

The coefficient : the degree of weighting decrease, a constant smoothing factor($0 < < 1$) is varies from 0 to

1. A larger decreases older observations faster.

X_t : the value at a 't' time period.

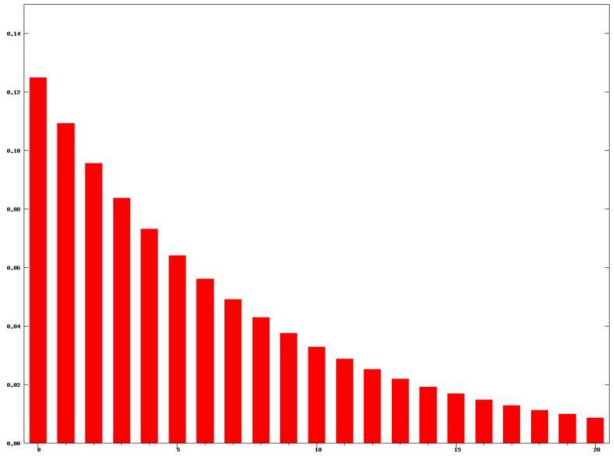

Fig. 6. EWMA

EMA considers the sequence of submission of documents to calculate the moving average. EMA uses the degree of weighting decrease represented by coefficient , as per new document submitted as shown in fig. 6. While assigning the concepts' weights, the newly submitted documents are given higher priority than the older ones. Also, the older documents' concepts' weights should not be neglected or else its weights may reach zero. So EMA considers the weight and the older datum which never reaches zero.

B. Value of for EWMA

The value of varies from 0 to 1. For example consider W1 as the oldest weight of particular concept, to make W1 relatively more important choose smaller values are take than higher values, because a larger discounts earlier data sets quickly, i.e. value of determines how fast a given value vanes but it never completely drops out.

The very standard method to find the value ofis by try and error method. By experimenting the modification of the concepts' weights the final value for comes out to be:

= 0:83

C. The Personalized Similarity Percentages

To calculate the personalized similarity percentage between two file, the similar concept from these two passages are selected and there weight are compared to find percentage similarity.

To determine the relative difference of two weights, the absolute difference between two weights are divided by the some functional value of the two weights.[3]

$$\text{Percent difference} = \frac{|W_1 - W_2|}{\left(\frac{W_1+W_2}{2}\right)} \times 100 = \frac{|(W_1 - W_2)|}{\left(\frac{W_1+W_2}{2}\right)} \times 100$$

The percentile difference of each and every concepts in these two document are calculated and the average of all these percentile differences have been taken. Now since the calculated average is the actually the percentile difference. Subtract the previously calculated average(Percent Difference) from 100 to achieve the similarity between two file. So, the percentile similarity between two files has been achieve.

VII. Graphical User Interface

The main aim of the Graphical User Interface is to intrigue the user and make the interaction as simple and efficient as possible. The software is user-oriented; easy for the user to enter data, get results and performing other events.

In this application, first window is for user login. If user has registered then he will log in using his/her username and password. There is an option available for the registration of a new user where he/she will be asked for username and password.

Fig. 7. Login GUI

These usernames and passwords are stored in the database. For more security, MD5 is calculated when user registers using his password and new token is granted.Details of security are discussed section V. Once user is logged in, using the browse button on homepage, user can select file from the machine for its submission. Top three similar files will be displayed on the

[3]In this project the the function used to divide the absolute different is arithmetic mean of two weights.

central section of the user interface in the form of a graph. Root node will be the file submitted by user and similar files will be connected to root node by lines.

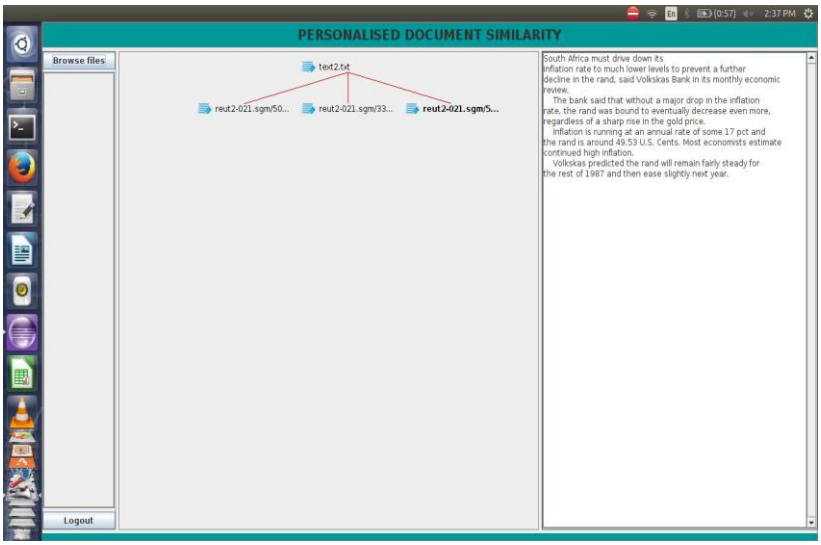

Fig. 8. Home Page GUI

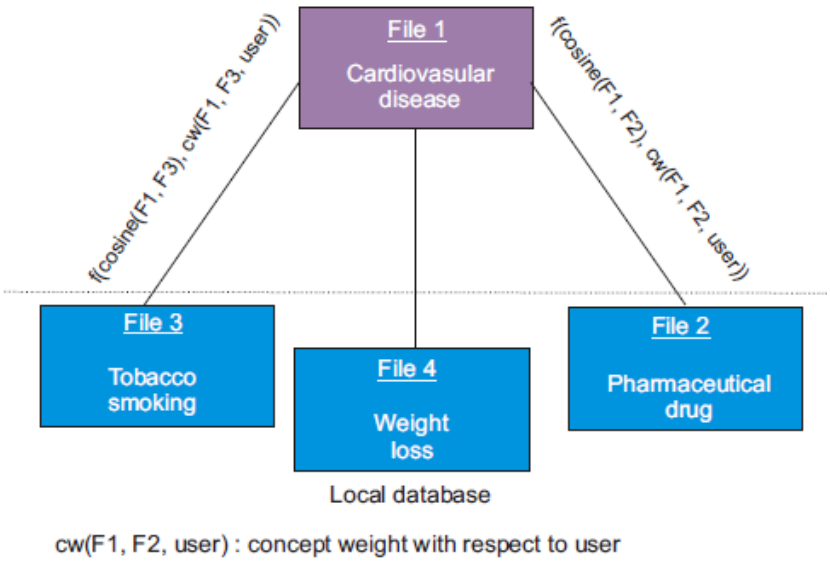

Fig. 9. Tree Structure

Double click on the file will expand the graph i.e. that signified file will be submitted for computing and again top three files will be displayed expanding the previous graph. During this user is able to see content of any file in the graph. Single click will display file contents on the right section of the user interface along with the filename and percentage similarity of that file with its parent file.

Thus, GUI makes user comfortable to find the documents close to his interests and likes, GUI further guides the user to read and explore his thoughts and ideas.

VIII. Conclusion

This application successfully describes the relation between files by calculating the similarity between them. There are three components used in this application. First component is the cosine similarity, which calculates similarity between documents by considering term frequency. Along with cosine similarity, second component is concept similarity. As concept similarity is also considered, it is more efficient for documents having same concept but different content. Hence by analyzing semantic structure of each sentence, sentence concepts are captured as conceptual term frequency ctf measure. Third component is personalization. With the use of personaliza-tion, document similarity search becomes user specific by considering his/her interest. Hence history of concepts used by the users is stored in database and used when user submits other files in future. Due to this, it is possible for two people to get different documents as output even if they happen to submit the indistinguishable document as their interests might be different. This is the idiosyncratic feature which can be used anywhere and extended further. It introduces one more dimension for the suggestion of interested groups for a person in already active softwares in industry.

There are numerous suggestions to extend this work.Technologies like Hadoop, openMP-MPI can be use to speed up and scale up the performance of the software. It takes too long to handle large database, the software can be optimize by reduce complexity by use of new technologies like CUDA C/CPP for parallel computing using GPU (Graphics Processing Unit). Collaborate with other data sets and include their databases which will increase the software scope for different applications[3].

References

[1] Mausam and Based on slides of W. Arms. Document similarity in information retrieval. https://courses.cs.washington.edu/courses/cse573/12sp/lectures/17-ir.pdf.
[2] Shady Shehata, Fakhri Karray, and Mohamed Kamel. A concept-based model for enhancing text categorization. In Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 07, pages 629637, New York, NY, USA, 2007. ACM.
[3] Edward Grefenstette and Stephen Pulman. Msc computer science disser-tation analysing document similarity measures. 2010.
[4] Timothy J Hazen. Direct and latent modeling techniques for computing spoken document similarity. In Spoken Language Technology Workshop (SLT), 2010 IEEE, pages 366371. IEEE, 2010.
[5] Thomas Hofmann. Learning the similarity of documents: An information-geometric approach to document retrieval and categorization. 2000.
[6] Anna Huang. Similarity measures for text document clustering. In Pro-ceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand, pages 4956, 2008.
[7] Anna Huang, David Milne, Eibe Frank, and Ian H Witten. Clustering documents with active learning using wikipedia. In Data Mining, 2008. ICDM08. Eighth IEEE International Conference on, pages 839844. IEEE, 2008.
[8] Lan Huang, David Milne, Eibe Frank, and Ian H. Witten. Learning a concept-based document similarity measure. J. Am. Soc. Inf. Sci. Technol., 63(8):15931608, August 2012.
[9] Christopher D Manning, Prabhakar Raghavan, and Hinrich Sch utze. Introduction to information retrieval, volume 1. Cambridge university press Cambridge, 2008.
[10] Donald Metzler, Yaniv Bernstein, W. Bruce Croft, Alistair Moffat, and Justin Zobel. Similarity measures for tracking information flow. In Proceedings of the 14th ACM International Conference on Information and Knowledge Management, CIKM 05, pages 517524, New York, NY, USA, 2005. ACM.
[11] Rada Mihalcea, Courtney Corley, and Carlo Strapparava. Corpus-based and knowledge- based measures of text semantic similarity. In Proceed-ings of the 21st National Con- ference on Artificial Intelligence - Volume 1, AAAI06, pages 775780. AAAI Press, 2006.
[12] Akiyo Nadamoto, Eiji Aramaki, Takeshi Abekawa, and Yohei Mu-rakami. Acquiring the gist of social network service threads via com-parison with wikipedia. Web-Based Multimedia Advancements in Data Communications and Networking Technologies, page 85, 2012.
[13] Akiyo Nadamoto, Yu Suzuki, and Takeshi Abekawa. Gist of a thread in social network services based on credibility of wikipedia. In System Sciences (HICSS), 2011 44th Hawaii International Conference on, pages 110. IEEE, 2011.
[14] P Perumal, R Nedunchezhian, and M Indra Priya. Concept-based document similarity based on suffix tree document.
[15] Martin F Porter. An algorithm for suffix stripping. Program, 14(3):130137, 1980.
[16] Gerard Salton, Anita Wong, and Chung-Shu Yang. A vector space model for automatic indexing. Communications of the ACM, 18(11):613620, 1975.
[17] Anastasios Tombros and C. J. Van Rijsbergen. Query-sensitive similarity measures for information retrieval. Knowledge and Information Systems, 6:2004.
[18] M Yasodha and P Ponmuthuramalingam. An advanced concept-based mining model to enrich text clustering. IJCSI International Journal of Computer Science Issues, 9(4):417422, 2012.