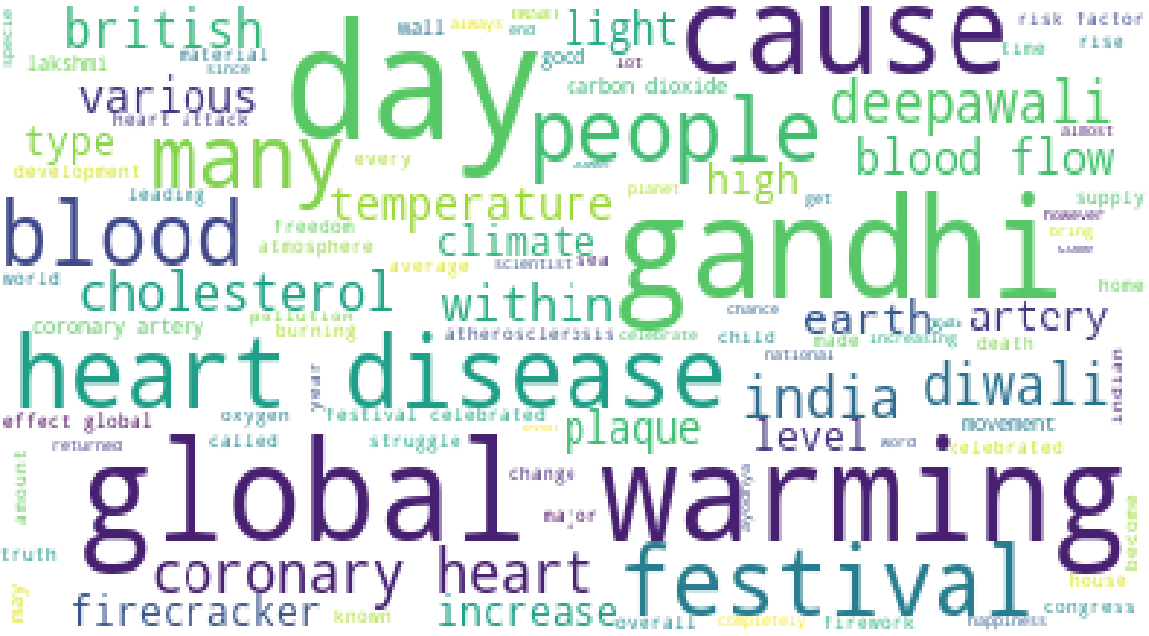
KEYWORD GENERATION USING TF-IDF AND LATENT SEMANTIC ANALYSIS



Rakshith RP
Sarvesh Kumar Thapa
Dawn Elza Zachariah

ABSTRACT

In recent times, data is being generated in humongous numbers. Textual data is an integral part of said data. Hence, understanding this data becomes a primary cause of concern. One way of understanding a large body of text (corpus) is by generating keywords which will help us in getting a vague idea of what is present in the text and what it is trying to tell us. Therefore, we have chosen keyword generation and key concepts generation as the topics of research and will be using TF-IDF and Latent Semantic Analysis respectively to do so. The basic design of the study includes taking in a text corpus (in .txt format), cleaning the text, and then finally running the algorithm. The output consists of a word cloud, grams, a set of keywords and finally key concepts.

INTRODUCTION

Text analysis is an important limb of data analytics. A majority of unstructured data is in the form of text. There are umpteen ways in which textual data can be analysed. It is mainly because there are as many conclusions one can draw from them. To be more precise, given a corpus, a researcher can be interested in either the volume of the text, the diversity of text, the language of the text or the essence of what the text is trying to convey. The focus of this paper falls under the final category. Keyword prediction and key concept generation is basically a method to summarize a large body of text to arrive at the central topic which the paper is trying to convey. By generating keywords, we can broadly understand what domain the data belongs to. Key concepts will further narrow down the study by classifying the words under a certain “concept”.
It is for this reason that we have embarked on the journey of writing this paper because it forms a union between concepts and words. There have been individual studies on them separately but they have never been part of the same study. It is to fill this gap that the study was undertaken. The main aim of this paper is to determine whether the non inclusion of these aspects was a conscious decision due the understanding that they are unrelated or hopefully find an interesting relation between them, thereby creating a new parameter for text prediction.

REVIEW OF LITERATURE

Latent semantic analysis is a natural language processing technique. It is a statistical model of word usage that permits comparisons of the semantic similarity between pieces of textual information. LSA was originally designed to improve the effectiveness of information-retrieval methods by performing retrieval based on the derived "semantic" content of words in a query as opposed to performing direct word matching. This approach avoids some of the problems of synonymy, in which different words can be used to describe the same semantic concept. We read

Latent Semantic Analysis for Text Based Research **by Peter W. Foltz** to understand the concept of LSA.

This paper summarized three experiments that illustrated how LSA may be used in text-based research. Two experiments described methods for analyzing a subject's essay for determining from what text a subject learned the information and for grading the quality of information cited in the essay. The third experiment described using LSA to measure the coherence and comprehensibility of texts.

Here, an experiment in which 24 college students were asked to read 21 texts related to the events leading up to the building of the Panama Canal. The texts included excerpts from textbooks, historians' and participants' accounts, and primary documents such as treaties and telegrams. After reading the texts, the subjects wrote an essay on the same. The texts which showed the greatest influence in the subjects' essays were to be determined. Foltz et al. (1994) reanalyzed the essays, using LSA to make predictions about which texts influenced the subjects' essays. The goal was to match individual sentences from the subjects' essays against the sentences in the original texts read by the subjects. Sentences in the essays that were highly semantically similar to those in the original texts would probably indicate the source of the subject's knowledge.

The first experiment of “Matching Summaries to the Texts Read” used the information on which text was the most similar. The second one of “Characterizing the Quality of Essays” used information on how semantically similar what a subject wrote in an essay was to what the subject had read.

Two measures of the quality of the essays were computed using LSA for this. The first examined the amount of semantic overlap of the essays with the original texts. Each sentence in each essay was compared with all sentences in the original texts, and a score was assigned on the basis of the cosine between the essay sentence and the closest sentence in the original texts. Thus, if a subject wrote a sentence that was exactly the same as a sentence in the original text, they would receive a cosine of 1.0, while a sentence that had no semantic overlap with anything in the original texts would receive a cosine of 0.0. The second measure determined the semantic similarity between what a subject wrote and the 10 sentences the expert grader thought were most important. In this way, it captured the degree of overlap with an expert's model of what was important in the original texts. In short LSA seemed to be a promising application for researchers. The method is automatic and fast, permitting quick measurements of the semantic similarity between pieces of textual information.

To understand keyword prediction, we referred to ***Metadata Tagging and Prediction Modeling: Case Study of DESIDOC Journal of Library and Information Technology (2008-17)*** **by M. Lamba and M. Madhusudhan**. The paper described the importance and usage of metadata tagging and prediction modeling tools for researchers and librarians. This study was divided into

two phases. The first phase determined the core topics from the research articles which was based on latent dirichlet allocation (LDA), whereas the second phase employed prediction analysis to annotate the future research articles on the basis of the modeled topics. The core topics (tags) were found to be digital libraries, information literacy, scientometrics, open access, and library resources for the studied period. This study further annotated the scientific articles according to the modelled topics to provide a better searching experience to its users.

Latent Dirichlet Allocation (LDA) modeling technique was used to model the topics (tags), and Support Vector Machine (SVM) and Naive Bayes were used as the predictors to predict the performance of created models. The paper had both methodological and applicative objectives

(i) to identify the main topics of the scientific articles

(ii) to annotate the scientific articles according to the modeled topics

(iii) to statistically evaluate the predictive model

The performance of the built models was analysed by the statistical evaluation measures: accuracy, precision, and recall.

A total of 387 articles were downloaded from DESIDOC Journal of Library and Information Technology (DJLIT) for the period 2008-17. The data were analysed according to the LDA probabilistic topic modeling method. For the 10 years period, five topics were identified. Each topic contained a probability value. These topics were ranked according to their probability values. Top five words were chosen as most representative of the topic according to their probability.

METHODOLOGY

In this paper we have used TF-IDF to generate the keywords and LSA to obtain the concepts.

These had been implemented as follows. The user was asked to input the paths of the .txt files. The file for which keyword was to be generated was also mentioned. The files were made into a data frame. Exploratory data analysis like word count, descriptive statistics, identifying common words and uncommon words were done. The pre-processing includes removing punctuations, stopwords, tags, special characters, digits, stemming, lemmatization and converting to lower case. Once that was done, it was appended to a list.

Word cloud is generated for this corpus list obtained. Then most frequently occurring words, bigrams and trigrams were found and plotted with their frequency. TF-IDF and TF-IDF scores were generated using `TfidfTransformer` and sorted in descending order to pick up the top

10 as keywords for the specified document. The concepts were obtained for the documents using `TfidfVectorizer` and `randomized_svd` based on LSA.

TF-IDF

TF-IDF in NLP stands for Term Frequency–Inverse document frequency. TF-IDF is a statistical measure that evaluates how relevant a word is to a document in a collection of documents. This is done by multiplying two metrics:

- **how many times a word appears in a document:** The term frequency of a word in a document. There are several ways of calculating this frequency, with the simplest being a raw count of instances a word appears in a document. Then, there are ways to adjust the frequency, by length of a document, or by the raw frequency of the most frequent word in a document.
- **the inverse document frequency of the word across a set of documents:** This means, how common or rare a word is in the entire document set. The closer it is to 0, the more common a word is. This metric can be calculated by taking the total number of documents, dividing it by the number of documents that contain a word, and calculating the logarithm. So, if the word is very common and appears in many documents, this number will approach 0. Otherwise, it will approach 1.

Multiplying these two numbers results in the TF-IDF score of a word in a document. The higher the score, the more relevant that word is in that particular document. To put it in more formal mathematical terms, the TF-IDF score for the word t in the document d from the document set D is calculated as follows:

$$tfidf(t, d, D) = tf(t, d) \cdot idf(t, D)$$ where

$$tf(t, d) = log(1 + freq(t, d))$$

$$idf(t, D) = log(\frac{N}{count(d \in D : t \in d)})$$

It's not that easy for algorithms or any techniques to handle text input as compared to numerical input. So, we need to transform that text into numbers, otherwise known as text vectorization. A word vector represents a document as a list of numbers, with one for each possible word of the corpus. Vectorizing a document is taking the text and creating one of these vectors, and the numbers of the vectors somehow represent the content of the text. TF-IDF enables us to give us a way to associate each word in a document with a number that represents how relevant each word is in that document.

TF-IDF has many uses, most importantly in automated text analysis, and is very useful for scoring words in machine learning algorithms for Natural Language Processing (NLP). TF-IDF was invented for document search and information retrieval.

- Information retrieval - TF-IDF was invented for document search and can be used to deliver results that are most relevant to what you're searching for. It's likely that every search engine you have ever encountered uses TF-IDF scores in its algorithm.
- Keyword Extraction - TF-IDF is also useful for extracting keywords from text. The highest scoring words of a document are the most relevant to that document, and therefore they can be considered *keywords* for that document.

LSA

Latent Semantic Analysis (LSA) is an efficient way of analyzing the text and finding the hidden topics by understanding the context of the text. It is used to find the hidden topics represented by the document or text. These hidden topics then are used for clustering the similar documents together. LSA is an unsupervised algorithm and hence we don't know the actual topic of the document. LSA uses a bag of word (BoW) model, which results in a term-document matrix (occurrence of terms in a document). Rows represent terms and columns represent documents. LSA learns latent topics by performing a matrix decomposition on the document-term matrix using singular value decomposition. LSA is typically used as a dimension reduction or noise reducing technique.

We have m documents and n words as input. An m * n matrix can be constructed. You can use count occurrence or TF-IDF score. However, TF-IDF is better than count occurrence in most of the time as high frequency does not account for better classification.

The LSA returns concepts instead of topics which represent the given document. The concepts are a list of words which represents the document in the best possible way. For example, in the dataset of sports document the concepts can be

Concept 1: ball, shoes, goals, win

Concept 2: ball, bat, score, umpire

As we can see two concepts — concept 1 represents football and concept 2 represents cricket. But we can see that the concepts can have overlapping words, so the whole set of words represent one concept together rather than the individual word. LSA tries to find the best set of words known as concept to represent the document using the term co-occurrence matrix. Concept is also a way of representing the document through dimension reduction.

The LSA is the pioneer for LSI and dimensionality reduction algorithms. The LSA is used for dimensionality reduction. We can reduce the vector size drastically from millions to mere

thousands without losing any context. This will help us in reducing the computation power and the time taken to perform the computation. The LSA is used in search engines. Latent Semantic Indexing(LSI) is the algorithm developed on LSA. The documents matching the search query are found using the vector developed from LSA. LSA can also be used for document clustering. As we can see that the LSA assigns topics to each document based on the assigned topic we can cluster the documents.

WORD CLOUD

For text data, word cloud can convey crucial information easily. Word cloud generators can help simplify lengthy databases or long pages of text and obtain insights. Word clouds (also known as text clouds or tag clouds) work in a simple way: the more a specific word appears in a source of textual data (such as a speech, blog post, or database), the bigger and bolder it appears in the word cloud. A word cloud is a collection, or cluster, of words depicted in different sizes. The bigger and bolder the word appears, the more often it's mentioned within a given text and the more important it is. Also known as tag clouds or text clouds, these are ideal ways to pull out the most relevant parts of textual data, from blog posts to databases.

Bigrams: Bigram is 2 consecutive words in a sentence.

E.g. "The boy is playing football". The bigrams here are:

The boy

Boy is

Is playing

Playing football

Trigrams: Trigram is 3 consecutive words in a sentence.

For the above example trigrams will be:

The boy is

Boy is playing

Is playing football

RESULTS

We took four different text files and tried to choose text files whose contents were different from each other. Each file was around thousand words,

	document	word_count
0	Deepawali is the main festival of Hindus. Ever...	966
1	The Biological Basis and Risk Factors for Coro...	1052
2	Mahatma Gandhi - Father of The NationBy Ritu J...	844
3	Global warming is an increase in the average t...	1030

Then found some descriptive statistics, Four the number of total number of text files. Mean is for the total number of words in four text files divided by four text files.

```
count          4.000000
mean         973.000000
std           93.416629
min          844.000000
25%          935.500000
50%          998.000000
75%         1035.500000
max         1052.000000
Name: word_count, dtype: float64
```

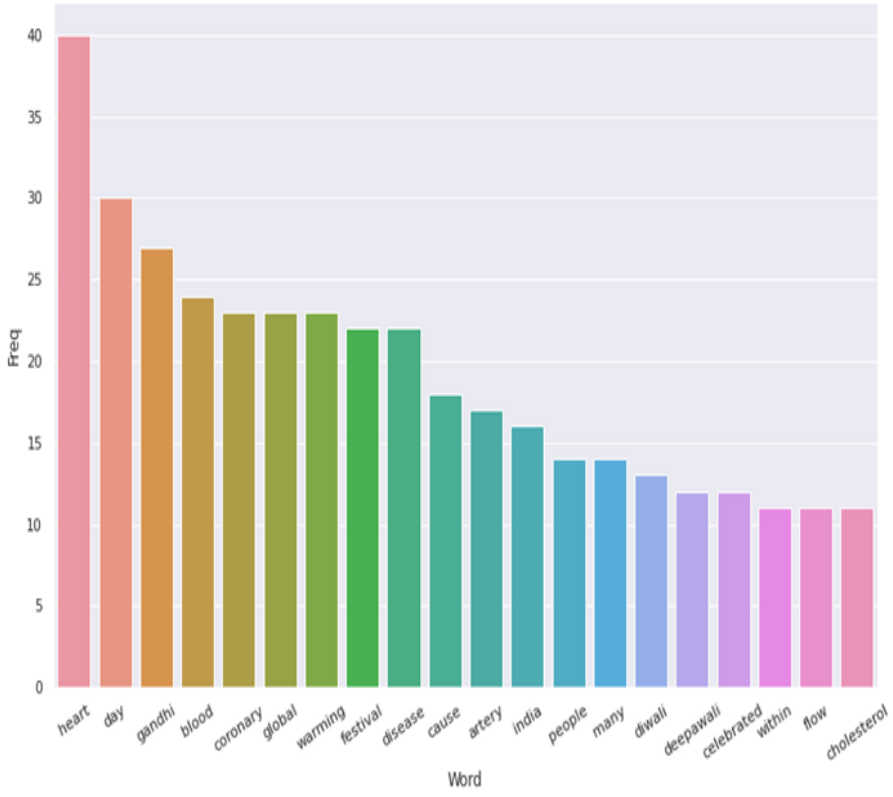
To find some common words, from the entire given text files, for this we print the most frequent words. To find uncommon words we print the last 20 words from the frequency list.

```
the          247
of           188
and          115
to            87
in            79
is            77
a             54
heart         37
are           36
that          32
The           31
for           27
on            27
blood         23
this          22
as            21
with          21
Gandhi        20
was           20
coronary      19
dtype: int64
```

```
worshiping                1
avali                     1
cadmium.                  1
cells                     1
biggest                   1
participation             1
look                      1
cholesterolSmokingLack    1
salt                      1
simulations,              1
atmosphere                1
worshiped.Bhai            1
twenty                    1
glacier                   1
activities                1
current                   1
walls,                    1
Co-operation              1
crows                     1
Government                1
dtype: int64
```

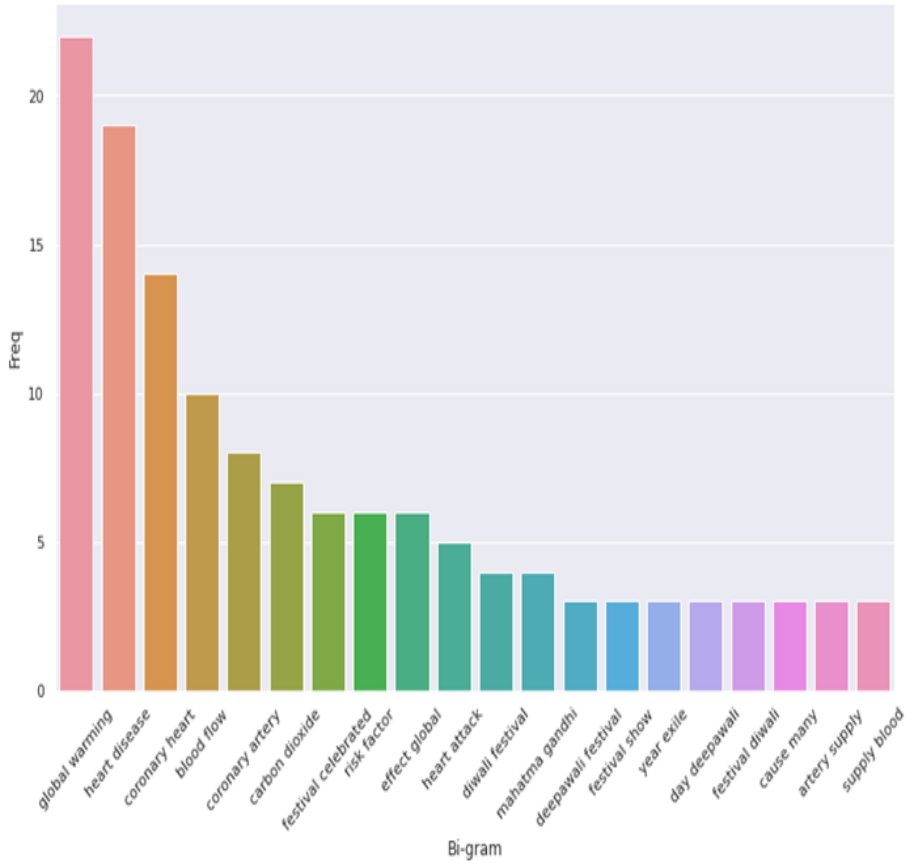
Word cloud is an easy and inexpensive way to visualize text data, but the challenge of interpreting a word cloud is that display emphasizes frequency of **words**, not necessarily their importance.

We found most repeating words from corpus also known as uni grams,



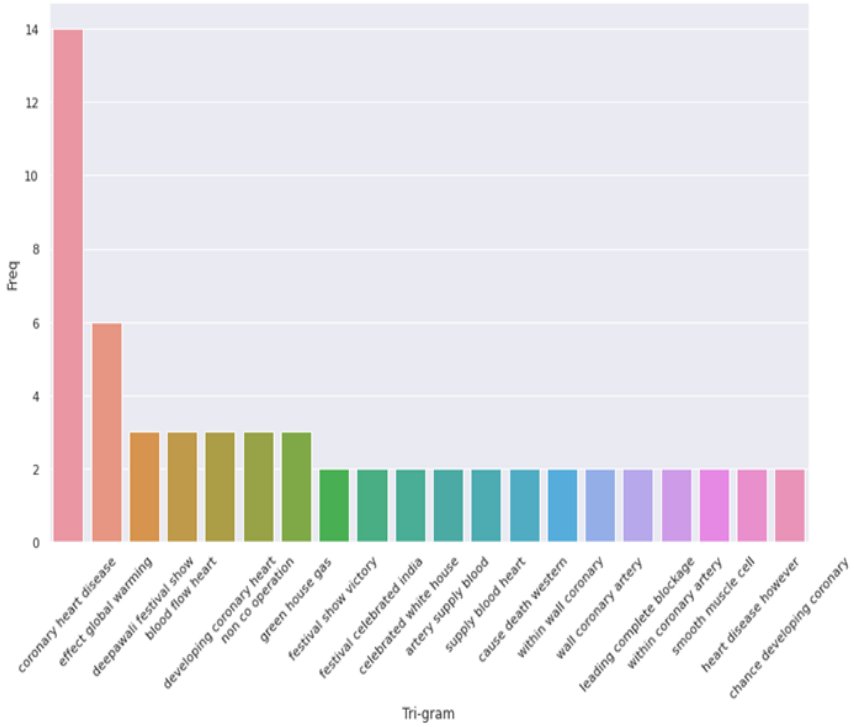

```
[Text(0, 0, 'heart'),
 Text(0, 0, 'day'),
 Text(0, 0, 'gandhi'),
 Text(0, 0, 'blood'),
 Text(0, 0, 'coronary'),
 Text(0, 0, 'global'),
 Text(0, 0, 'warming'),
 Text(0, 0, 'festival'),
 Text(0, 0, 'disease'),
 Text(0, 0, 'cause'),
 Text(0, 0, 'artery'),
 Text(0, 0, 'india'),
 Text(0, 0, 'people'),
 Text(0, 0, 'many'),
 Text(0, 0, 'diwali'),
 Text(0, 0, 'deepawali'),
 Text(0, 0, 'celebrated'),
 Text(0, 0, 'within'),
 Text(0, 0, 'flow'),
 Text(0, 0, 'cholesterol')]
```

Bi grams,




```
[Text(0, 0, 'global warming'),
 Text(0, 0, 'heart disease'),
 Text(0, 0, 'coronary heart'),
 Text(0, 0, 'blood flow'),
 Text(0, 0, 'coronary artery'),
 Text(0, 0, 'carbon dioxide'),
 Text(0, 0, 'festival celebrated'),
 Text(0, 0, 'risk factor'),
 Text(0, 0, 'effect global'),
 Text(0, 0, 'heart attack'),
 Text(0, 0, 'diwali festival'),
 Text(0, 0, 'mahatma gandhi'),
 Text(0, 0, 'deepawali festival'),
 Text(0, 0, 'festival show'),
 Text(0, 0, 'year exile'),
 Text(0, 0, 'day deepawali'),
 Text(0, 0, 'festival diwali'),
 Text(0, 0, 'cause many'),
 Text(0, 0, 'artery supply'),
 Text(0, 0, 'supply blood')]
```

Tri grams




```
[Text(0, 0, 'coronary heart disease'),
 Text(0, 0, 'effect global warming'),
 Text(0, 0, 'deepawali festival show'),
 Text(0, 0, 'blood flow heart'),
 Text(0, 0, 'developing coronary heart'),
 Text(0, 0, 'non co operation'),
 Text(0, 0, 'green house gas'),
 Text(0, 0, 'festival show victory'),
 Text(0, 0, 'festival celebrated india'),
 Text(0, 0, 'celebrated white house'),
 Text(0, 0, 'artery supply blood'),
 Text(0, 0, 'supply blood heart'),
 Text(0, 0, 'cause death western'),
 Text(0, 0, 'within wall coronary'),
 Text(0, 0, 'wall coronary artery'),
 Text(0, 0, 'leading complete blockage'),
 Text(0, 0, 'within coronary artery'),
 Text(0, 0, 'smooth muscle cell'),
 Text(0, 0, 'heart disease however'),
 Text(0, 0, 'chance developing coronary')]
```

Final results,

```
Keywords:
festival 0.403
diwali 0.238
deepawali 0.22
celebrated 0.22
light 0.165
firecracker 0.147
lakshmi 0.128
festival celebrated 0.11
firework 0.092
child 0.092
```

Keywords were generated for the chosen file, but concepts were obtained using Latent Semantic Analysis(LSA). These concepts were obtained By performing LSA on all the submitted text files.

```
Concept 3:
heart
blood
coronary
heart disease
artery
coronary heart
coronary heart disease
cholesterol
blood flow
risk

Concept 4:
festival
diwali
celebrated
deepawali
light
firecracker
lakshmi
festival celebrated
child
firework
```

CONCLUSION

A keyword extraction method was proposed using the LSA method combined with a ranking scheme from a single document. LSA has already been used for sentence extraction in previous studies. The contribution of this study has four steps, creation frequency matrix, computation of singular value decompositions of the frequency matrix followed by vector identification where we eliminate least significant singular values from the matrix and last step index creation which organizes the data into hierarchical structure. With any given cut-off number N, the top N most important keywords were extracted from the first column. Output keywords were compared to automated keyword extraction methods, ranked TF-IDF .

We aimed for generating a key concept from a corpus. Firstly, started out by performing exploratory data analysis for finding out common and uncommon words in our corpus. We define 'Document preprocessing' for many documents which contain many things which are not core or relevant for the required text analysis. After performing stemming and lemmatization, we convert words into lowercase and remove the stop words. By doing this we get clean and preprocessed corpus. Then we made a word cloud to depict the tags from the corpus but that plot emphasizes frequency of words, not necessarily their importance. Vectorization, characterize each text as a vector. Each text has some common and uncommon words compared to each other

to account for all possibilities, a word set is formed which consists of words from both the documents. After this we found out the most frequently occurring word, Bi grams and trigrams to get a more accurate idea about the corpus. Then we take abstract and keywords from the new corpus on which we have performed TF-IDF. At last we apply latent semantic analysis on the new corpus, Result which we obtained were from LSA, the concept in question, as well as all documents. LSA analysis recovers the original semantic structure of the space and its original dimensions. The new dimensions by LSA analysis are a better representation of documents and queries and by using this reduced dimension noise from data can be removed; the noise is a data which could be described as rare and less important usage of certain terms.

The main advantage of TF-IDF is the mere utilization of the knowledge of the whole document set in order to compute the inverse document frequency. Efficiency becomes crucial for large document sets where keywords need to be extracted automatically from daily submissions incrementally. The LSA-based keyword extraction method is language-independent, knowledge-lean, and easy to apply in a diverse set of settings. It does not require any language-specific information or external knowledge sources other than the input text. Major limitation of this LSA is that vectors require large storage and due to this large size of data it requires an additional storage space and computing time which reduces the efficiency of LSA. Another objection is with SVD that it is designed for normally distributed data, but such a distribution is inappropriate for all types of data.

Future scope of the paper can be to implement this LSA using interactive UI, so that results of which can be obtained easily and compared. This can help in faster information retrieval.

REFERENCES

Lamba, Manika & Margam, Madhusudhan. (2019). Metadata Tagging and Prediction Modeling: Case Study of DESIDOC Journal of Library and Information Technology (2008-2017). 12. 33-89. 10.18329/09757597/2019/12103.

Foltz, Peter. (1996). Latent Semantic Analysis for Text-Based Research. Behavior Research Methods. 28. 197-202. 10.3758/BF03204765.