Search Engine For Ebook Portal

Prashant Kanade, Aishwarya Sadasivan, Komal Dhuri, Manaswini Muralidaran, Meghna Mohan

Abstract: The purpose of this paper is to establish the textual analytics involved in developing a search engine for an ebook portal. We have extracted our dataset from Project Gutenberg using a robot harvester. Textual Analytics is used for efficient search retrieval. The entire dataset is represented using Vector Space Model, where each document is a vector in the vector space. Further, for computational purposes we represent our dataset in the form of a Term Frequency- Inverse Document Frequency (tf-idf) matrix. The first step involves obtaining the most coherent sequence of words of the search query entered. The entered query is processed using Front End algorithms this includes-Spell Checker, Text Segmentation and Language Modeling. Back End processing includes Similarity Modeling, Clustering, Indexing and Retrieval. The relationship between documents and words is established using cosine similarity measured between the documents and words in Vector Space. Clustering performed is used to suggest books that are similar to the search query entered by the user. Lastly, the Lucene Based Elasticsearch engine is used for indexing on the documents. This allows faster retrieval of data. Elasticsearch returns a dictionary and creates a tf-idf matrix. The processed query is compared with the dictionary obtained and tf-idf matrix is used to calculate the score for each match to give most relevant result.

Index Terms: similarity modeling, clustering, elastic search, vector space model, term frequency-inverse document frequency (tf-idf) matrix, language modeling, spell checking, text segmentation.

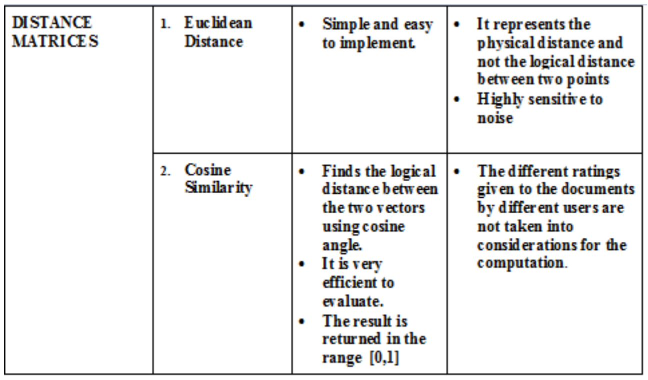
1 Introduction

Today's world revolves around technology and the internet. Humans want immediate and precise results. Even readers prefer E-books. Keeping up with the need for quick and relevant results we intend to develop an efficient search engine for an E-book portal. The search engine would be preferred by the user under two main circumstances- i) retrieving relevant results even if the search query entered is slightly vague ii) obtaining results which are similar or related to the entered the search query in addition to exact results. The objectives stated above will be accomplished by developing a search engine with a user friendly interface. A comparative study of the various algorithms used for each module is shown in Table 1.

- *Prashant Kanade is currently an assistant professor in Computer EngineeringDepartment in V.E.S.I.T, Chembur, Mumbai-400071, India, PH-+91-22-61532518 E-mail: prashant.kanade@ves.ac.in*
- *Aishwarya Sadasivan is currently pursuing Bachelor's degree in Computer Engineering in V.E.S.I.T, Chembur, Mumbai-400071, India, PH-+91-99330217169 E-mail: sadasivan.aishwarya@ves.ac.in*
- *Komal Dhuri is currently pursuing Bachelor's degree in Computer Engineering in V.E.S.I.T, Chembur, Mumbai-400071, India,*
- *PH-+91-9167198871 E-mail: komal.dhuri@ves.ac.in*
- *Manaswini Muralidaran is currently pursuing Bachelor's degree in Computer Engineering in V.E.S.I.T, Chembur, Mumbai-400071, India, PH-+91-8080236230 E-mail: muralidaran.manaswini@ves.ac.in*
- *Meghna Mohan is currently pursuing Bachelor's degree in Computer Engineering in V.E.S.I.T, Chembur, Mumbai-400071, India,*
- *PH-+91-9167150233 E-mail: mohan.meghna@ves.ac.in*

***Table 1** Comparison of the various algorithms that can be used for each module*

MODULES	ALGORITHMS	ADVANTAGES	DISADVANTAGES
SPELL CHECKER	1. Isolated term correction	• Corrects on a single query term which makes it less complex	• Fails to correct typographical errors.
	2. Context sensitive correction	• Coherent corrected search query. Topographical errors are eliminated.	• It does not cover all classes of error.
TEXT SEGMENTATION	1. Similarity based segmentation	• It uses unsupervised learning method and thus does not require a prior knowledge. • High efficiency	• Slightest change in parameter values alters the segmented result. • Information from the dictionary does not help. • Mistakes occurring in the segmented result are very close to the expected segmented output.
	2.Feature based segmentation (Bag of words model)	• It uses n-gram model for segmentation • It uses supervised learning method model and hence close mistakes can be avoided.	• The processing time to split the query into n-grams and storing them is a tedious task.
CLUSTERING	1. K means clustering	• If there are lesser number of centroids then the computation speed is good. • K-Means forms tighter clusters in comparison to other clustering algorithms.	• It is difficult to provide number of centroids • Different clusters are formed depending on the initial partitions, giving varying results with every execution.
	2. Hierarchical clustering	• Information about the number of clusters need not be given before-hand. • Easy to implement.	• Reversing actions performed with respect to forming clusters is difficult. • The algorithms used to generate distance matrices affect the clusters formed.



DISTANCE MATRICES	1. Euclidean Distance	• Simple and easy to implement	• It represents the physical distance and not the logical distance between two points • Highly sensitive to noise
	2. Cosine Similarity	• Finds the logical distance between the two vectors using cosine angle. • It is very efficient to evaluate. • The result is returned in the range [0,1]	• The different ratings given to the documents by different users are not taken into considerations for the computation.

2. Functional Requirements

2.1 FRONT-END PROCESSING

2.1.1 Spell Checker

A search engine requires a spell checker [2] to rectify a misspelt query by enumerating candidates and choosing the one with minimum modifications and having greatest frequency.

Methodology:
The entered query will get tokenized. The tokens are used to form a candidate set which is generated by performing insertion, transposition, deletion or replacement. In case of a tie, a word with a higher occurrence in the corpus is used as correctly spelt word.

2.1.2 Text Segmentation

When a query is entered without any spaces we segment this query using the concept of bag of words[2].

Methodology:
The mathematical representation of bag of words model is as follows,

P(w1…wn)=P(w1) X P(w2|w1) X (w3|w2)...X..P (wn|wn-1)….(1)

In Eq.1, P (wn|wn-1) is the probability that the nth word (wn) occurs given the previous n-1 words (wn-1) that have occurred. The product obtained indicates the probability of occurrence of a certain sequence of words. The candidate list is generated by segmenting the given query character by character and the probability of this segment with the rest of the query is calculated using the formula mentioned above.

2.1.3 Language Modeling

Grammar correction is implemented using n-gram model[2] where probability of each word is estimated based on n-1 words of prior context.

Methodology:
The words of a sentence are split into bi-grams. The product of probabilities of the words in the bigram and the conditional probability between them helps establish the correct sequence giving us a grammatically correct sentence.

2.2 BACK-END PROCESSING

2.2.1 Similarity Modeling

The similarity [3] between the documents will be calculated using the concept of vector space model [1]. Documents are represented in space, where the axes represent the document vectors. Similarity is measured by calculating the distance between the document vectors.

Methodology:
The dataset is first tokenized, stemmed[5] and stopwords are removed. A vocabulary is then created consisting of the pruned terms along with their indices. Now in order to deduce the similarity, the documents are represented using the Vector space model. The vector space model is represented by the term frequency-inverse document frequency (tf-idf) matrix. The tf-idf matrix is computed using two attributes- term frequency and inverse document frequency. The term frequency establishes the fact that greater the frequency of a term in a document more is its importance in the document. Inverse document frequency signifies a term occurring very frequently across documents is less important to a particular document. So in order to incorporate both of these concepts and assign optimum weights to the terms of a document, the following formula is applied-

$$\textit{tf-idf}\,(t,d) = \textit{tf}(t,d) \times \textit{idf}(t)\ldots.(2)$$

$$\textit{idf}\,(t)=\log\left(\frac{n_d}{1+df(d,t)}\right)\ \ldots.(3)$$

n_d is the total number of documents and df(d,t) is the number of documents that contain term t. idf(t) is the inverse document given the term t The resulting tf-idf vectors are then normalized by the Euclidean norm:

$$v_{norm}=\frac{v}{\sqrt{v2_1+v2_2+\ldots\ldots\ldots+v2_n}}\ \ldots.(4)$$

The next step is establishing similarity by calculating the cosine angle between these vectors. This serves two purposes- i) Document-Document similarity ii) Word-Word similarity. The Document-Document similarity established can be used for Genre based classification. The Word-Word similarity helps establish the context of usage of words.
Cosine angle between vectors is computed as follows-

$$K(X,Y) = <X,Y>/||X|| * ||Y||\ldots.(5)$$

This is nothing but the dot product of the vectors X and Y.
Now in order to compute the document-document similarity we compute the dot product of tf-idf matrix with itself. Word-Word similarity is computed by finding transpose of tf-idf matrix and then computing dot product of the transposed matrix with itself. Thus similarity modeling helps deduce the similarity between documents and terms across various documents.

2.2.2 Clustering

Similar documents will be partitioned into non-overlapping groups called clusters, which serves the purpose of suggesting books that are similar to the search query entered.

Methodology:
K- Means Clustering [3] is used. The first step is to choose the value of K. Initially K random seed points are taken as centres of the K clusters. Each data point is then assigned to one of the clusters depending on the minimum distance from the centroid. The centroid of the cluster is then recomputed by calculating the mean of the data points in the cluster. The above process is repeated until the centroid value doesn't change any further. One of the main objectives to be achieved is selecting an optimal 'k' value or number of clusters for formation of clusters. Instead of using the Euclidean distance between the data points and the centroid we use the cosine distance which works better in VSM as compared to Euclidean distance. As Euclidean considers the actual distance than logical distance between vectors it may give larger values. Thus, we compute the average cosine distance of an iteration which is compared to that of the previous iteration. If the value is consistent we say we have found the optimal value of 'k'.

2.2.3 Indexing
Indexing involves parsing, storing and analyzing data for the purpose of quick retrieval. We have used elasticsearch engine for indexing. As mentioned earlier documents have been represented using the Vector Space Model. Mathematically, this model is represented as a TF-IDF matrix. Elasticsearch[6] Engine uses the TF-IDF representation and hence giving a better understanding of the working of Elasticsearch. This engine facilitates fast searching, provides multi language support, allows full text search, can be used to store big data and scales to Terabytes. Elasticsearch is based on Lucene[4]. Elasticsearch is essentially a document store. The documents looks like json files and are accessed over HTTP. The conceptual model of elasticsearch includes indices, doctype and contents in a format similar to a hash table. The document store can have multiple doctypes and does not have any particular schema but doctype parameter allows us to specify a user defined doctype. The data is ingested into the indexer. A dictionary consisting of document id, book name, author name, cluster id, score and contents of the document is formed by parsing through the dataset. Full text search requires every document to be assigned a unique Document ID. The index has to be finely grained. The document store will contain a record of the occurrence of terms in the document along with the position of occurrence of the term in the document. This facilitates full text search.

2.2.4 Retrieval
Retrieval of relevant results forms an integral part of a search engine. The relevance is established using the concept of TF-IDF matrix. The relevance of the retrieved results is deduced based on the scores returned by elasticsearch. The score is evaluated using the tf-idf value. This value helps determine the relevance of the query to the retrieved results. Further, these scores help in ranking the documents for the order of relevance. Thus, elasticsearch uses the VSM representation to give accurate search results. Further, to enhance the functionality of retrieving relevant search results, the cluster information is used. The cluster information helps to suggest related books in the search result. The query structure, in dictionary format is passed to elasticsearch. Searching can be done using queries or filters. Filters return Boolean values and do not consider score values for retrieval while query returns documents according to the scores. We have used query for searching. We then specify the criterion for retrieval such as match, match_phrase, etc. Match_phrase especially helps us retrieve results based on content as it searches for the exact phrase in the content ingested. This is the advantage of a full text search.

E.g.
```
query={“query”:{“match”:{“specific_part_of_doc_to_be_sear
ched”:search_query}}}
es.search(query0,index=name_index)
```

Here es is an instance of elasticsearch which helps us invoke the various methods of elasticsearch. The query structure can be defined in a way to match on multiple conditions using the bool criteria. Thus, elasticsearch helps in efficient retrieval through clean codes.

3 PROPOSED ARCHITECTURE WITH MODULAR DESIGN

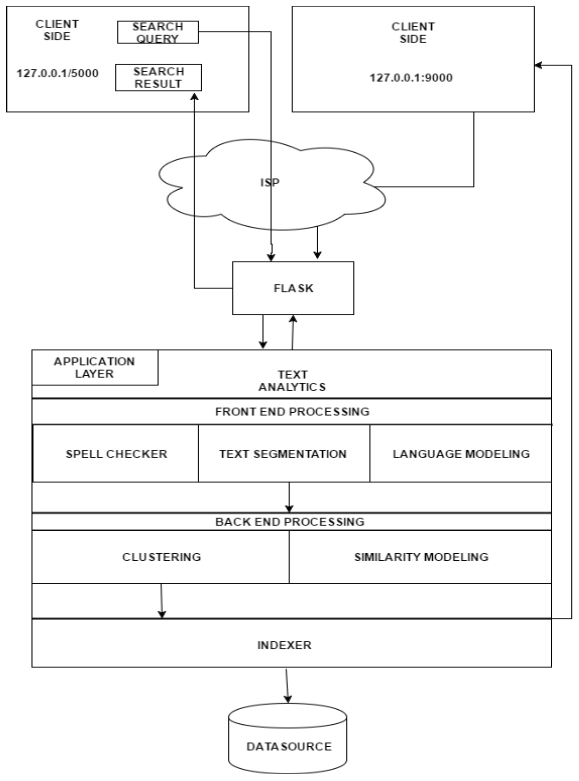


Fig 1. *System Architecture*

4 EXPECTED OUTCOME
The search engine will consist of a user friendly tabbed interface. This interface would serve 2 main purposes 1) Visualization of usage of various words in a particular context. 2) Retrieval of results. The user enters a search query. This query is processed and subjected to various front end processing algorithms. Query is compared to the datastore. Relevant results will then be retrieved and displayed. In addition the user will be provided with a

graphical representation of the usage of words in a given context. Clustering performed will help display similar results.

4.1 DATA VISUALISATION
Visualisation[7] is the simplest and most powerful technique to provide a visual access to large amounts of data. Chord Diagram is a graphical technique that displays the inter relationships between the documents. The formation of chord diagram requires two files as input viz. Json matrix and a .csv file. Json matrix contains the similarity between the documents, calculated using cosine similarity. The .csv file contains the names of the documents and the corresponding colours used to represent each document.

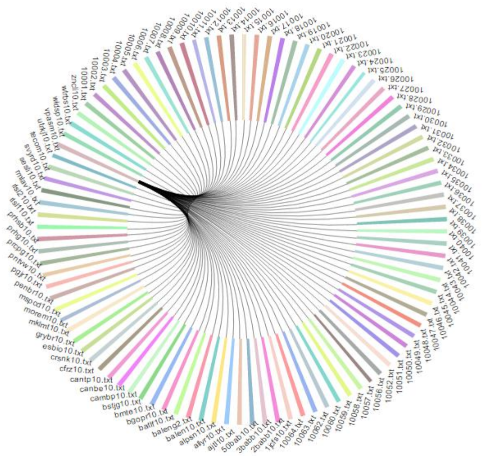
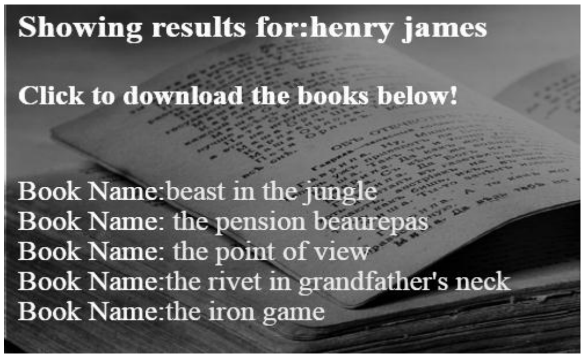


Fig 2. *Chord Diagram*

In the above fig. The circumference of the circle is the sum of the Json matrix. The circumference is then divided into arcs depending on the number of books in the dataset. The arcs represent documents and the ribbons represent the relation between two documents. When hovered above the ribbons, the percentage measure of similarity between the two documents is obtained.

4.2 RETRIEVAL OF RESULTS
The retrieval of books is achieved by querying the author/book name or by content of the books. Using the concepts of clustering and indexing explained above, books similar to the query entered can also be obtained.

Fig. 3 *- Search tabs*



Showing results for:henry james

Click to download the books below!

Book Name:beast in the jungle
Book Name: the pension beaurepas
Book Name: the point of view
Book Name:the rivet in grandfather's neck
Book Name:the iron game

Fig 4*-Results of search by author/book name*

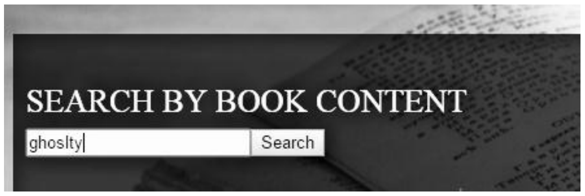
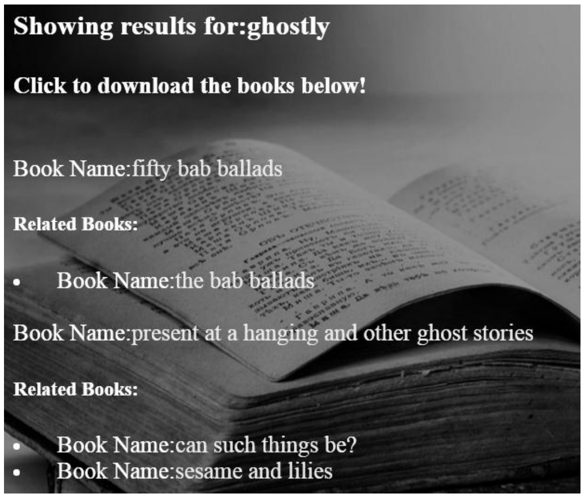


Fig 5*-search by content tab*



Showing results for:ghostly

Click to download the books below!

Book Name:fifty bab ballads

Related Books:

- Book Name:the bab ballads

Book Name:present at a hanging and other ghost stories

Related Books:

- Book Name:can such things be?
- Book Name:sesame and lilies

Fig 6*- Results with suggestions of similar books*

5 CONCLUSION
An e-book portal would require the retrieval of books with respect to title, excerpts as well as authors of the books. This brings us to involving analytics in developing a search engine to be able to extract most relevant books given the above three instances. Also the ability to group similar books encourages the user to read more books of similar liking. In addition, the chord diagram that has been developed helps visualize the similarity established between books and between words.

6 ACKNOWLEDGEMENTS
We would like to thank Mr. Sidharth Shah for his constant support and guidance throughout the project.

7 REFERENCES

[1] P. D.Turney and P. Pantel (2010)“From Frequency to Meaning: Vector Space Models of Semantics'', Volume 37, pages 141-188.

[2] List of IPython (Jupyter) Notebooks by Peter Norvig: How to do things with words or Statistical Natural Language processing in Python.

[3] Programming Collective Intelligence,Building Smart Web 2.0 Applications, by Toby Segaran, O'Reilly Media (Chapter 3).

[4] http://lucene.apache.org/java/docs/.

[5] http://snowball.tartarus.org/algorithms/lovins/stemmer.html.

[6] https://www.elastic.co/guide/en/elasticsearch/reference/current/getting-started.html

[7] http://bl.ocks.org/AndrewRP/7468330