2014

WikiSearch: An Information Retrieval System

Hari Kishore Muvva
Grand Valley State University

WikiSearch
An Information Retrieval System

By
Harikishore Muvva
April, 2014

WikiSearch

An Information Retrieval System

By

Harikishore Muvva

A project submitted in partial fulfillment of the requirements for the degree of

Master of Science in

Computer Information Systems

at

Grand Valley State University

April, 2014

Jonathan Leidig **04-24-2014**

Table of Contents

Abstract

WikiSearch is an information retrieval system (based on the vector space model) that can be used for searching Wikipedia, one of the largest knowledge bases in the world. Clustering techniques are utilized to group semantically related documents and improve the efficiency of the search system. Clustering allows relevant documents that do not match a query's explicit form to be retrieved. Cluster labels are automatically generated using document features to provide a faceted browsing service for exploration and discovery. We also propose a storage scheme for creating and managing inverted index and clustering information using a NoSQL database. Finally, performance results are provided for the search system.

1. Introduction

Information retrieval (IR) involves finding data (usually documents) of an unstructured nature (usually text) that satisfies an information need utilizing collections. There were several models developed for information retrieval such as Boolean retrieval, vector space model, probabilistic model, and language model. Among the existing models for information retrieval the vector space model is one of the most effective model (and thus used). Vector-space models rely on the premise that the meaning of a document can be derived from the document's constituent terms [1]. Vector space models involve document preprocessing, e.g., removing the stop words, stemming, calculating the term frequency, document frequency and calculating the inverse document frequency.

Using the indexing techniques (keyword-based), documents are retrieved if they contain term specified by the user's query. Many documents in a collection contain desired semantic information, even though they do not contain the user specified keywords. This limitation can be addressed by clustering or classification of the documents. Clustering is the grouping of text documents into semantically related groups so that the documents in the same cluster are more similar to each other than to the documents in other clusters. Document clustering is very useful in improving the precision and recall of information retrieval systems and also gives user a better way to efficiently browse the retrieved results. Experimental evidences show that IR application can benefit from the use of document clustering [1].There are two main

steps in clustering process, the first one is to preprocess the documents i.e. transforming the documents into a suitable features or concept representation .The second stage is to analyze the prepared data and divide into clusters. Further the clusters need to be labelled using the document features.

In this paper we have implemented a search system that uses the vector space model for information retrieval and a clustering technique to cluster the document collection for optimal performance of the search engine. Cluster labels were also generated, which enables us to provide the faceted browsing service in the search system. An embedded database system (The Berkeley Database) is used for storing TF-IDF scores, index position and cluster information. An appropriate storage scheme is designed to optimally use the features of NoSQL database system, minimizing the index building time and to support faster retrieval of search results. The focus of this paper is on building an efficient search system with faceted browsing feature to process search queries on Wikipedia corpus. The rest of this paper is organized as follows. Section 2, discusses about related work and Section 3 provides information on using the application. Section 4 deals with implementation architecture of the information retrieval system, identifying the various components of the system, and presents an overview of the system. Discussion about the system design is organized into four sections – text extraction, building TF-IDF vector space model, clustering and labelling the clusters. Section 5 will provide the system performance statistics and conclusions in Section 6.

2. Background and Related Work

The research efforts in the field of information retrieval have spanned over diverse domains in terms of the input sources they use and also different techniques to improve the efficiency of information retrieval process. A large body of research has focused on using textual data as their input while a lesser percentage of research has focused on semi structured data and relational databases. A considerable amount of research has been conducted in the field of clustering and classification of documents. Clustering improves the precision and recall of the information retrieval system and also helps in classifying the search results which gives the user a greater flexibility in browsing the retrieved results. In [3] , the authors focused on clustering Wikipedia music related articles by means of a Self-Organized Map (SOM). They relied on the bag-of-words similarity between individual articles and also used the link structure between

the articles. [4] discusses an efficient approach for web document clustering that use WordNet lexical categories and a fuzzy c-means algorithm to improve the performance of clustering problem for web document. By using WordNet words lexical categories information, authors were successful in reducing the size of vector space and present semantic relationships between words. [5] proposes an algorithm for document browsing system based on clustering. It uses hierarchical clustering as a first step which is then refined by using the k-means algorithm. This approach is also well known as scatter/gather algorithm. [6] discusses about an automatic approach for finding and selecting sets of related Wikipedia articles. This approach consists of using semantic contexts. [7] presents a comparative study of different feature selection methods focusing on aggressive dimensionality reduction and suggest that document frequency threshold is reliable measure for selecting informative features. In [8], Patil and Atique proposed a TF-IDF features selection model in document clustering where they use three different feature selection methods to build the TF-IDF vector space model. The closest work to our study is presented in [9], where the authors have proposed using WordNet to cluster the text documents; however this works uses the clustering technique based on features selection model to improve the performance of the information retrieval process and also proposes an approach to automatically generate the labels to the clusters.

3. Program Requirements

WikiSearch has a client application where user can enter his query and search Wikipedia. The client program is a Java swing application with a simple user interface dialog where the user can enter search string and click the search button or hit the enter key. Internally, As shown in [Fig.1] the client program parses the query string, tokenizes it, removes all the stop words, and then performs the stemming of the search term (The stemmer is the same as one used within the search engine).The database is queried for each search term to fetch all the relevant documents that contains the search term and also the respective cluster information of the documents.

After fetching the relevant documents for each document the arithmetic mean of the inverse document frequency for all the search terms is calculated. The documents are then grouped by clusters and the average inverse document frequency of each individual cluster is calculated. The clusters are then sorted in descending order by average TF-IDF score and displayed in the list box on the right side of the

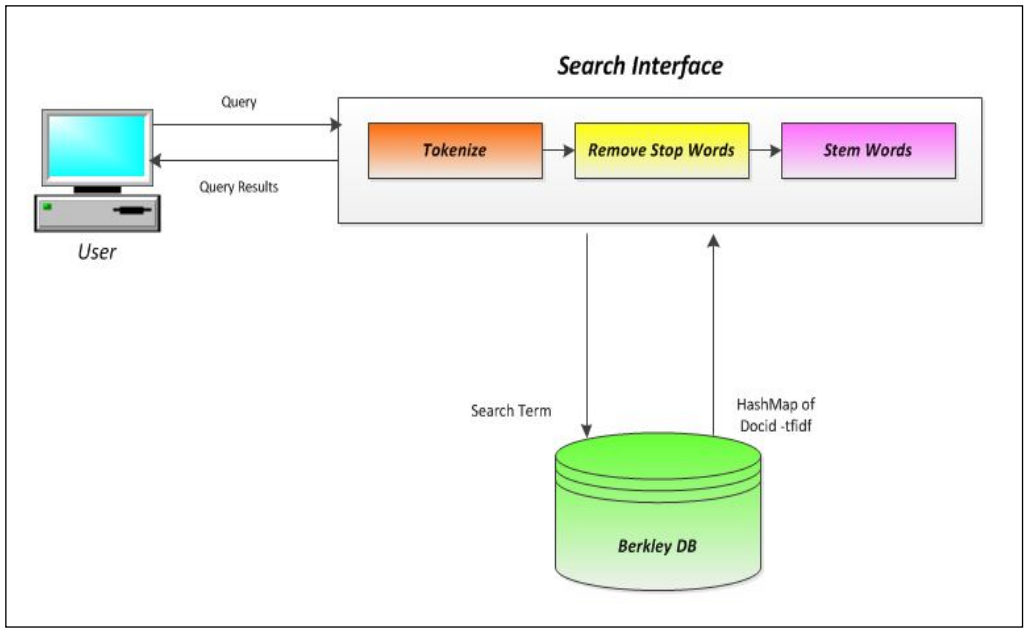

Figure 1 : WikiSearch Client Architecture

search screen. The application also displays the relevant documents in the selected cluster ordered by inverse document frequency. When the user clicks on different clusters the application displays the relevant documents in that cluster. If the user wanted to see all the documents in a particular cluster irrespective of search results, they can click the show all check box. Unchecking the show all checkbox will display only documents relevant to the search query. The application displays a title, URL, and some text about the article for each search result. The user can view the article by clicking on the corresponding URL.

4. Implementation

Implementation of WikiSearch information retrieval system can be divided into four steps.

- Text Extraction
- Building the TF-IDF vector space model for the text content
- Clustering
- Labeling the clusters

4.1 Text Extraction

Text extraction is a process of extracting textual data from the document collection which could be of any format like HTML or XML etc. Wikipedia XML dump is used as the data set used for the current system. The Wikipedia XML corpus is available as one large file (size 40 GB), or as small chunks of multiple xml files which is around 1-2GB per piece. The application contains a module called load which handles this piece of functionality. The load module parses each XML document using SAX parser and extracts the title, document id and the text content for each page. The document id is unique for each page and can be used to identify an article in Wikipedia. The document-Id, title, URL and text are stored in the database with documentId as key in the document store. The challenge in parsing the Wikipedia xml is extracting the text content of the article. The text contains a multitude of annotations, citations, and references, along with Wikipedia's formatting tags and many other HTML text formatting tags. Various complicated regular expressions were used to pre-process the text. After processing the text content, the application saves the text as text files with document-Id as name of the file.

4.2 Building a TF-IDF Vector Space Model

After the Load module has generated the text documents each one for an article, the rest of the application uses these text files. There are several steps involved in building a vector space model for the terms in text documents [Fig.2].

- Each sentence need to be tokenized to individual terms
- Stop words need to be removed.
- Words need to be reduced to its stem or root form.
- Finally the term frequency and inverse document frequency are calculated

4.2.1 Stop words Removal.

Words that are extremely common which would appear to be of little value and not convey much meaning are considered as stop words. These words are removed from the text as they are not much helpful in selecting documents matching a user query. A static list of stop words is used in this system. The application removes all words that are in the static stop word list. For example, stop words removal process will remove all the words like: she, he, all, his, from, is and so on.

4.2.2 Stemming

The stem is the common root-form of the words with the same meaning appear in various morphological forms (e.g. player, played, plays from stem play). The Porter stemmer introduced in [10] was implemented in this search system. Stemming will find the stems of the output terms to enhance term frequency counting process because terms such as "readers" and "reading" come down from the same stem "read". This process will output all the stems of extracted terms. The Porter stemming algorithm (or 'Porter stemmer') is a process for removing the commoner morphological and in flexional endings from words in English. Its main use is as part of a term normalization process that is usually done when setting up information retrieval systems. The algorithms involve six steps they are applied to the words in the text starting from step 1 and moving on to step 6. Further, they are applied sequentially one after the other as commands in a program.

- Step 1: Gets rid of plurals and -ed or -ing suffixes.
- Step 2: Turns terminal y to i when there is another vowel in the stem.
- Step 3: Maps double suffixes to single ones:-ization, -ational, etc.
- Step 4: Deals with suffixes, -full, -ness etc.
- Step 5: Takes off -ant, -ence, etc.
- Step 6: Removes a final -e.

4.2.3 Calculating TF-IDF Score

After tokenizing, stop words removal and stemming the individual terms and their term frequency and document frequency will be available .The TF-IDF measure is the product between the Term Frequency (TF) and the Inverse Document Frequency (IDF). Below are the formulas for the above mentioned term frequencies.

$$\boldsymbol{TF_{t,d}} = \frac{\boldsymbol{n_{t,d}}}{\sum_k \boldsymbol{n_{k,d}}}$$

$TF_{t,d}$: Number of times a word t appears in a document d

n : Frequency of term t in the document d

k : Number of distinct words in document d

$$IDF_i = \log \frac{|D|}{|\{d : t_i \in d\}|}$$

IDF_i : Log of the fraction between the number of documents in the corpus and the number of documents in which the word t appears

| D | : Number of documents in the corpus

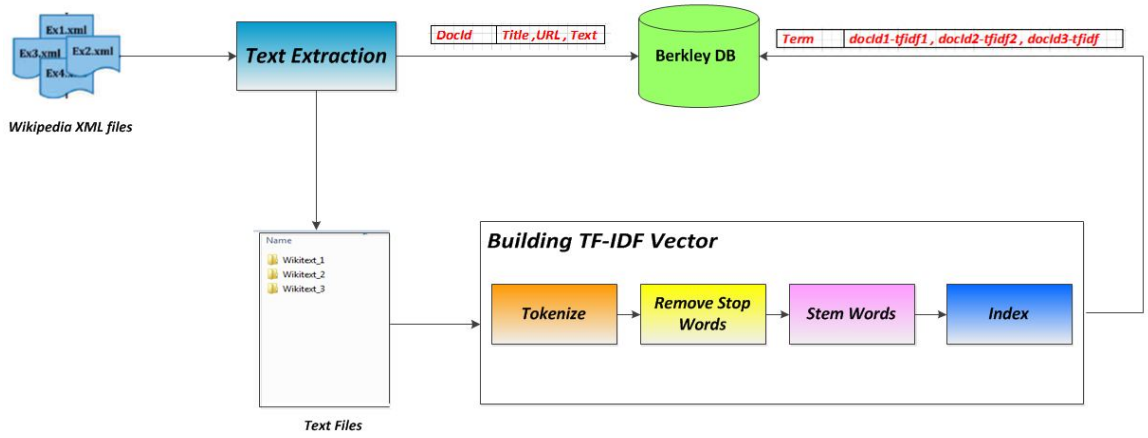

Figure 2.Text Extraction & Building TF-IDF Vector Space Model process

4.3 Clustering

Document clustering is defined as follows .Given a set of N documents DS, DS is clustered into a user defined number of k document clusters D1, D2, D3, D4...Dk (i.e {D1, D2, D3...Dk}=DS) so that the documents in a cluster are similar to one another while documents from different clusters are dissimilar [4]. Clustering algorithms can be broadly classified as hierarchical (agglomerative and divisive) and partitioning (k-means, Bisecting K-means,) clustering algorithms. Hierarchical clustering generates a hierarchical tree of clusters also called as dendogram. A comparative analysis of different clustering algorithms on different data sets done by Steinbach [11] concluded that k-means out performs the hierarchical methods. Furthermore a variant of k-means called bisecting k-means yields even better performance. Given the linear runtime performance and consistently good quality of the clusters that it produces, bisecting K-means is an excellent algorithm for clustering large number of documents. Bisecting

k-means combines the strengths of partitional and hierarchical clustering methods and thus used in this search system.

There are three main steps in the implementation of clustering [Fig.3], first is the feature extraction, second is generating a vector space model with extracted features and third is document clustering that applies clustering algorithm on document feature vectors to obtain the output clusters.

4.3.1 Feature Extraction

Feature extraction is essential to make clustering efficient and more accurate. The most important goal of feature extraction is to extract the topic terms which represent the document best. In this model a set of words extracted from the document called "bag of words" represent each document as a vector by the terms and their weights regardless of their order. Document representation based on bag of words model tend to use all the words in the documents which intern will lead to a large dimensions of the representation vector, this is called "Curse of Dimensionality". WordNet a lexical database is used to avoid this problem. The synonym and hyponym of the words can reveal the hidden similarities which lead to better clusters. After removing the stop words for each term the hyponym for the term is retrieved from WordNet and the first root entity of the hyponym tree is used for the representation vector. The top 15 terms ordered by term frequency are considered to be the terms that represent the document features best. Once the features are extracted with term frequencies and document frequencies, the TF-IDF score is calculated as described in the section. Finally a feature vector space model is made available.

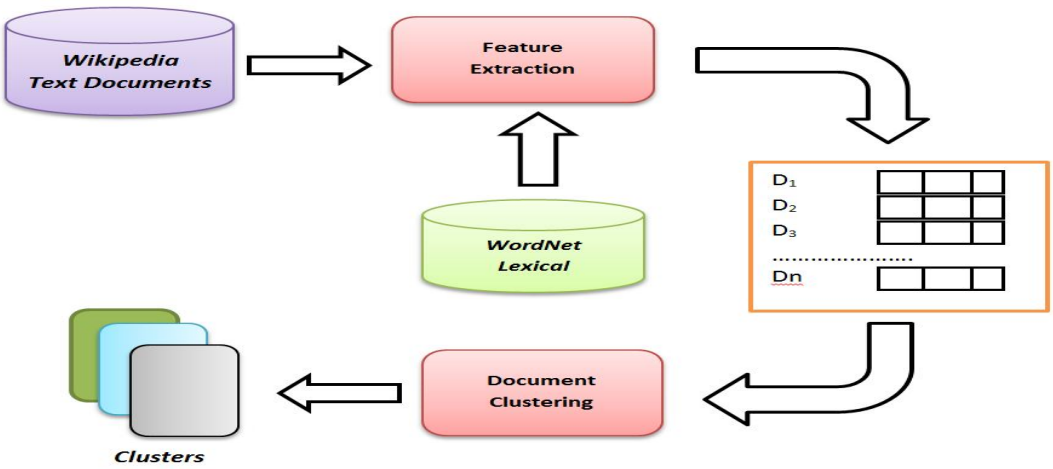

Figure 3. Clustering Process

4.3.2 Document clustering

A free software package called Cluto was used for clustering the documents.Devleloped by the University of Minnesota, Department of Computer Science & Engineering, this tool is used for clustering low and high dimensional datasets and for analyzing the characteristics of the various clusters .There are two stand-alone programs in CLUTO: vcluster and scluster. The vcluster program treats inputs as high-dimensional vectors, whereas scluster operates on the similarity space between the objects [12]. As vector space model is being used to represent the documents, vcluster program is used to cluster the documents. Cluto provides many parameters to control the clustering process among which the parameter which is used to choose the clustering algorithm is the important one. Bisecting k-means clustering algorithm is used for the search system. The optimal value for K is determined using an iterative clustering analysis with respect to total clustering distance. A random number is selected to determine the number of clusters and then create the clusters and calculate the total clustering distance. These steps will be repeated by incrementing the number of clusters. The point where the total distance variance is not significant enough, the number of clusters at that point is considered as optimal [Fig.4]. In our case we settled at 300 clusters

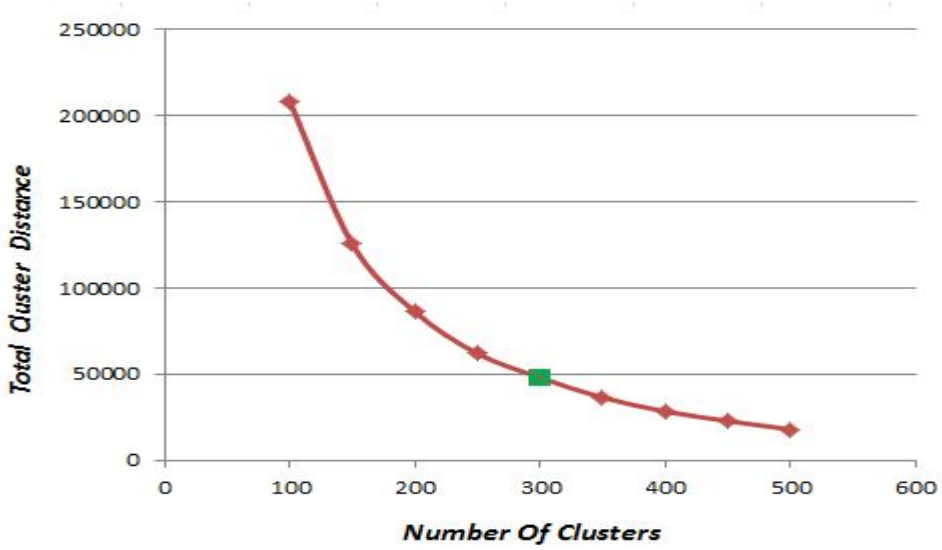

Figure 4. Iterative Cluster Analysis Graph

4.4 Labelling the Clusters

An automated approach is implemented for labelling the clusters. The distinctive terms or feature terms in a document represent the document best. Hence the feature terms of a document are used to generate the labels automatically. The feature terms are the more frequent distinctive terms in a document after excluding the stop words. WordNet is used to group the terms with similar meaning together in order to achieve accuracy in label generation. Below is the process to generate the labels for a cluster [Fig.5]. This process will be repeated for each cluster

- Access the files in a cluster

- Extract the feature term for each document in the cluster
- Calculate the term frequency and document frequency considering the cluster as the document collection
- Then select the terms whose document frequency is greater than 70% of the collection size(i.e. terms which appears in more than 70% of the documents in a cluster)
- Sort the terms by term frequency and then select the top three terms as labels

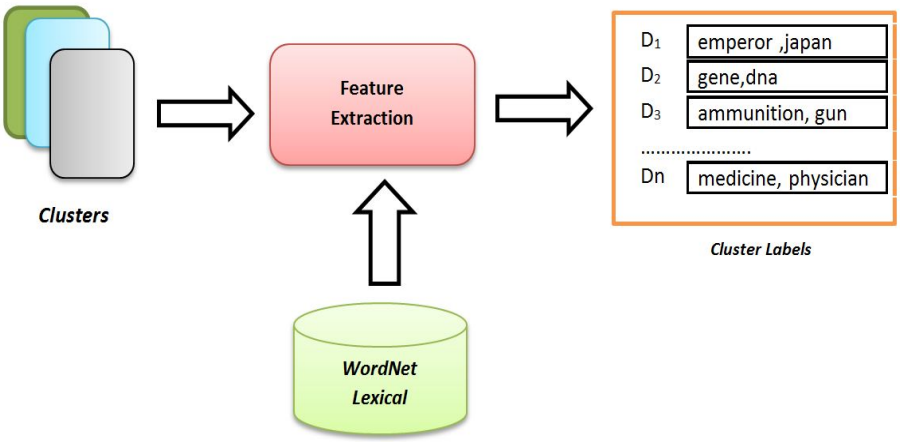

Figure 5.Label Generation Proces

4.5 Database

Berkley DB is a general-purpose NoSQL, key value pair, embedded database engine and can provide users with a wide variety of data management services. An embedded database is a library toolkit that provides database support through a well-defined programming API [13].Being an embedded database engine, it is fast and can be compiled and linked into developing applications. Berkley DB can be scaled from few bytes to terabytes of data. It is limited only by the available system resources. It provides many programming API's which gives the ability to read, write and manage the database. The in memory cache used by Berkley DB drastically improves the performance, although it requires significant memory resources to perform its tasks.

To implement the information retrieval system four key-value stores were defined in Berkley database. The storage schema was designed to take optimal advantage of the key-value pair database and for high performance.

1. Document store

Key: documentId

Value: title, URL, text

2. Term store
Key: term
Value: $docId_1$-tf |$docId_2$-tf...

3. TFIDF store
Key: term
Value: $docId_1$-tfidf|$docId_2$-tfidf |...

4. Cluster store
Key: clusterId
Value: $docId_1$, $docId_2$, $docId_3$,.| Label

4.6 WordNet

WordNet is a lexical database developed at Princeton University by a group led by George Miller. Our system uses WordNet 2.1 to identify the synonym words so that the words with semantic similarity can be grouped together. WordNet covers majority of nouns, verbs, adjectives from English Language. A total of 155327 words are mapped to 117597 sysnsets where each sysnset represents a concept. These sysnsets are very useful in identifying the document features.

5. Results, Evaluation, and Reflection

In order to evaluate the search system performance, a set of queries were prepared as shown in Table.1. Each query is executed in the system and the following attributes were recorded. The total relevant documents retrieved (true positive), total non-relevant documents retrieved (false positive), total relevant documents in the data set which are not retrieved (false negative) .Using these attributes, precision, recall and F-measure were calculated. Precision is the fraction of the returned results that are relevant to the information need. Recall is the fraction of the relevant documents in the collection returned by the system. F-Measure is used as it provides more robust evaluation criteria using precision and recall. They are calculated as follows.

$$precsion = \frac{true\ positive}{true\ positive + false\ positive}$$

$$recall = \frac{true\ positive}{true\ positive + false\ negative}$$

$$F - Measure = 2 \times \frac{precision * recall}{precsion + recall}$$

Before analyzing the results, we want to clarify the evaluation queries. Q1 is used to retrieve all entities matching "Christian gospels". Q2 is used to retrieve all entities matching keywords "surgery". Q3 is used to retrieve all entities matching keywords "bomber aircraft".

Q1	Find all the Christian Gospels (query: Christian gospels)
Q2	Find the entities matching surgery (query : surgery)
Q3	Find the bomber aircrafts (query : bomber aircrafts)

Table 1.Evaluation Queries

Queries	***Precision (%)***	***Recall (%)***	***F-Measure***
Q1	*64*	*100*	*0.78*
Q2	*66*	*100*	*0.79*
Q3	*65*	*100*	*0.78*

Table 2. Evaluation Results

Queries	***Precision (%)***	***Recall (%)***	***F-Measure***
Q1	*87*	*78*	*0.875*
Q2	*84*	*76*	*0.798*
Q3	*89*	*80*	*0.842*

Table 3.Evaluation Results at Threshold TF-IDF

The obtained results (Table.2) show that the using WordNet based text clustering technique in information retrieval system resulted in high rate of recall. This means that although some unnecessary documents are provided by the system, the relevant documents were not missed. To increase the precision search results were limited by a threshold TF-IDF score. The precision and recall values are shown in

Table-3. Limiting the search results by a threshold TF-IDF has improved the precision and reduced the recall thus improving the performance of the search system.

6. Conclusions and Future Work

This paper presents WikiSearch: an information retrieval system for searching Wikipedia corpus. The methodology of the system and the approach to implement the information retrieval system was also presented. We investigated whether introduction of clustering technique based on sematic similarity could help improve the process of information retrieval. Grouping the semantically related documents using clustering techniques improved the performance of the information retrieval system. Clustering the document corpus enabled us to group the search results so that user can easily browse the search results. An approach to automatically generate labels for the clusters using the document features was also proposed in this paper. As future work it would be interesting to carry out some experiments using other datasets (e.g. the REUTERS collection).In these experiments, due to lack of any existing concept ontologies we were not able to map the cluster labels to a single concept (e.g. Feature terms as doctor, treatment, medicine, these terms are related to a concept "Health").It would be useful to develop a concept ontology for use in systems that utilize clustering techniques.

7. Acknowledgements

I am thankful to my project advisor Prof. Jonathan Leidig for the guidance and many fruitful discussions during the work on this paper. I am also grateful to Prof. G. Karypis from University of Minnesota for making the clustering tool (CLUTO) available and to Prof. Miller and his team for making WordNet available.

8. Bibliography

[1] T. A. Letsche and M. W. Berry. Large-scale information retrieval with latent semantic indexing. Information Sciences, vol. 100, no. 4, pp.105–137, 1996.

[2] L.Muflikhah and B.Baharudin.Document Clustering Using Concept Space and Cosine Similarity Measurement. International Conference on Computer Technology and Development, vol.1, no., pp.58, 62. 2009.

[3] S. Bloehdorn and S. Blohm. A self organizing map for relation extraction from Wikipedia using structured data representations. Proceedings of the International Workshop on Intelligent Information Access, 2006.

[4] T.F Gharib, M.M Fouad and M.M Aref .Web document clustering approach using wordnet lexical categories and fuzzy clustering. 11th International Conference on Computer and Information Technology, vol., no., pp.48, 55, 2008.

[5] Douglass R. Cutting, David R. Karger, J. O. Pedersen, and John W. Tukey.
Scatter/Gather: a cluster- based approach to browsing large document collections.
In Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval, pp.318-329.1992

[6] S.E.Garza, R.F.Brena, E.Ramirez. Topic Calculation and Clustering: An Application to Wikipedia.
In Proceedings of seventh Mexican International Conference on Artificial Intelligence, pp.8- 93. 2008.

[7] Y.Yang and J. O. Pedersen. A Comparative Study on Feature Selection in Text Categorization.
In Proceedings of the Fourteenth International Conference on Machine Learning (ICML '97), pp.412- 420, 1997.

[8] L.H.Patil, M.Atique.A novel approach for feature selection method TF-IDF in document clustering.
IEEE 3rd International Advance Computing Conference (IACC), pp.858- 862, 2013.

[9] J.Sedding and D. Kazarov. WordNet-based Text Document Clustering.
In Proceedings of the 3rd Workshop on Robust Methods in Analysis of Natural Language Data, pp.104-113, 2004
.
[10] M. F. Porter. An algorithm for suffix stripping.
In Readings in information retrieval, Morgan Kaufmann Publishers Inc.pp.313-316,1997.

[11] M. Steinbach, G. Karypis, V. Kumar, A comparison of document clustering techniques.
In Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining ,KDD, 2000.

[12] G. Karypis, "CLUTO: A Software Package for Clustering High-Dimensional Data Sets," University of Minnesota, Dept. of Computer Science, Minneapolis, MN, Nov. 2003. Release 2.1.1 (www-users.cs.umn.edu/ karypis/cluto).

[13] S. Melink, S. Raghavan, B. Yang, and H. Garcia-Molina. Building a distributed full-text index for the web.
ACM Transactions on Information Systems. vol. 19, pp. 217– 241, July 2001.

[14] George A. Miller –"WordNet: A Lexical Database for English".

[15] Christopher D. Manning, P. Raghavan and H.Schütze. Introduction to Information Retrieval.
Cambridge University Press. 2008.

[16] P.Rosso, A.Molina, F.Pla, D. Jiménez and V. Vidal. Information Retrieval and Text Categorization with Semantic Indexing. In Proceedings of 5th International Conference, pp 596-600, 2004.

[17] J.Gonzalo, F.Verdejo, I. Chugur and M. Juan. Indexing with WordNet synsets can improve Text Retrieval.
Computing Research Repository, 1998.

[18] A. Hotho, S. Staab and G. Stumme. WordNet improves Text Document Clustering.
In Proceedings 26th Annual International ACM SIGIR Conference, 2003.

[19] N. Sahoo, J. Callan, Ramayya .K, G.Duncan, and R.Padman. 2006. Incremental hierarchical clustering of text documents. In Proceedings of the 15th ACM international conference on Information and knowledge management.2006