IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

Enhancing Text Mining Using Side Information

Prof. P.S. Toke[1], Rajat Mutha[1], Omkar Naidu[1], Juilee Kulkarni[1]

Department of Computer Engineering, PVPIT, Savitribai Phule Pune University[1]

Abstract: One of the widely researched data mining problem in the text domains is Clustering. Classification, document organization, visualization, indexing and collaborative filtering are some of the various applications of this problem. Large amount of information present within documents is in the form of text. We cannot always retrieve data in a refined text format. It also contains a lot of Side Information, int the form of different links in the document, user-access behaviour, and document provenance information from web - logs or other non-textual attributes. Large amount of information may be contained in these attributes for clustering purposes. To estimate the relative information is very difficult and cumbersome in most of the cases, particularly when some of information is noisy data. In such situation, integrating this side information into mining process can be risky, because it might result in either:
1. Addition of noise into the data
2. Improvement of quality of data mining process
An ethical way is needed for performing the data mining process, and for maximizing the advantages of the use of this available side information. We are proposing the use of K-means algorithm for improved and efficient clustering of the information in this paper.

Keywords: Information-Retrieval, Text-mining, Clustering, K-means, Side Information.

I. INTRODUCTION

The problem of text clustering appears in the context of many application domains such as the web, social networks, and other digital collections. Auxiliary information also known as Side Information or Meta data is available within text documents in several text mining application. Links in the document, Document provenance information, and other non textual attributes which are contained in the document and web logs are the different kind of side information.

1. Information Retrieval

The activity of obtaining information resources relevant to an information need from a collection of information resources is termed as Information Retrieval (IR). These searches can be based on full-text indexing."Information overload" is reduced using the automated information retrieval systems. The most visible IR applications include Web search engines. An information retrieval process is initiated when a user enters a query into the system. Queries are formal statements of information requirements, for example search strings in web search engines.

2. Text Mining

Text mining is termed as the process of deriving high-quality information from text. Text mining refers to the process of structuring the input text, deriving patterns within the structured data, and finally evaluation and interpretation of the output. Common text mining tasks include text categorization, text clustering, concept extraction, document summarization, and entity relation modelling. A typical application is to scan a set of documents containing text written in a natural language and either populate a database or search index with the information extracted or model the document set for predictive classification purposes.

3. Side Information

A tremendous amount of side information is also associated along with the documents in many application domains. This is mainly because text documents typically occur in the context of a variety of applications in which there may be a large amount of other kinds of database attributes or meta-information .This metadata can be useful to the clustering process. Some examples of such side-information include:

- In an application which tracks user access behaviour of web documents, the user-access behaviour may be captured in the form of web logs. Such kind of logs can be used for enhancing the quality of the mining process in a way. This is because these logs can often pick up subtle correlations in content, which cannot be picked up by the raw text alone.
- Most text documents containing links among them can also be treated as attributes. These attributes may provide insights about the correlations among documents in a way which may not be easily accessible from raw content.
- Many web documents contain meta-data which provides ownership, location, or even temporal information that may be informative for mining purposes.

4. Clustering

Clustering is the process of grouping a set of objects in such a way that objects in the same group known as cluster, are more similar to each other than to those in other clusters. It is a main activity of exploratory data mining. It also is a typical technique for statistical data analysis, used in many domains. Various algorithms are used to achieve clustering. These algorithms differ significantly in their view of what constitutes a cluster and how to efficiently

find them. Clustering can be stated as a multi-objective optimization problem. The working of an appropriate clustering algorithm depends on the individual data set and intended use of the results. Cluster analysis is not an automatic task in a way, but an iterative process of knowledge discovery or interactive multi-objective optimization based on trial and failure. Cluster analysis can be a tool to gain insight into the distribution of data to observe characteristics of each cluster.

II. DATA AND METHODOLOGY

Our approach is to build a system based upon existing search engines like Google, Bing, Yahoo and many others that return web documents using numerous approaches of identifying relevant information from the observable internet. After obtaining these links (which will be referred to as original links henceforth), we try to remove noisy information which may not be relevant to the query entered and diagnose additional links and web documents which may be appropriate relevant to the query.

Data mining algorithms for clustering are one of the best ways to obtain related information, by grouping similar objects together. For text clustering, the actual text needs to be transformed in a suitable format for clustering. The scheme to be followed is:

A. Initialization Phase
B. Main Phase
C. Result Generation Phase

A. Initialization Phase

Step 1: The documents refer to the text that is acquired from web documents. These are achieved by doing a simple keyword search on a search engine. These documents represent our original seeds. The additional documents which may be relevant are gained by recursively obtaining web links from each of the original document contents.
Step 2: The obtained documents cannot be used directly for clustering. The reason behind this is that natural language contains a lot of redundant and irrelevant words which can mislead algorithms which focus on word count and word weight. Stopping technique is applied to filter unwanted irrelevant information, redundant words such as “is”, “am”, “are”, “there”, “when”, “then”,"of" etc.
Step 3: These filtered documents are saved separately and then used for processing in the main phase. Each of the filtered documents is collected and then passed to the main phase.

B. Main Phase

Step 1: Filtered documents are processed one by one by applying text mining techniques i.e. Shared Word Count, Word Count Bonus (to calculate weight of words), similarity measures. Similarity measures considered are cosine similarity and Pearson measure. Euclidean distance or Minkowski distance are the different other methods used to calculate the distance in terms of similarity between different documents.
Step 2: Step 1 provides the list of attributes and base for clustering and classification of documents. K-means clustering is the prime way to arrange objects into different clusters. All the filtered documents are reiterated to devise their clusters using weighted characteristics and similarity measures. Nearest neighbour approach is also applied before assignment of any document to a particular cluster. Each technique has some weight and sum of the weights is used for finalizing the documents. This step establishes the core process of the working of the system. A detailed explanation of how k-means algorithm works is given further.
Step 3: The performance of the method is increased by applying clustering incrementally.

C. Result Generation Phase

Evaluation of the implemented system shall be done using different data mining measuring features such as accuracy, sensitivity, f-measure etc. Precision and Recall are two important measures of accuracy in any information retrieval algorithms. Precision provides us the number of retrieved documents that are actually relevant. Recall measures the total number of relevant documents which were retrieved. We need to identify which documents might be considered relevant by the user for calculating precision. Recall may consider the original links and their sub links as the total number of documents to be considered.
Results shall be compared with the results of original COATES algorithm provided in base paper.

III.MODELLING

K-means:
K-means (MacQueen, 1967) is one of the classic unsupervised learning algorithms that can be used to solve the well known clustering problem. The procedure includes easy and simple steps to classify a given data set. The initial idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because different location can cause different result. It is necessary to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When all the points get associated to respective cluster, the first step is completed and an early grouping is done. Now we need to re-calculate k new centroids as binary centres of the clusters resulting from the previous step. After the calculation of k new centroids, a new binding has to be done between the same data set points and the nearest new centroid. A loop gets generated because of which we may notice that the k centroids change their location step by step until no more changes are done. In other words centroids do not move any more.
Finally, the aim of this algorithm is to minimize an objective function, here a squared error function. The objective function is as follows:

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2$$

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

where |xi-cj| is a chosen distance measure between a data point xi and the cluster centre cj, which indicates the distance of the n data points from their respective cluster centres.

The algorithm is composed of the following steps:
1. Place K points into the space represented by the objects that are being clustered. These points represent initial group centroids.
2. Assign each object to the group that has the closest centroid.
3. When all objects have been assigned, recalculate the positions of the K centroids.
4. Repeat Steps 2 and 3 until the centroids no longer move. This produces a separation of the objects into groups from which the metric to be minimized can be calculated.

TF-IDF:
TF-IDF refers to term frequency-inverse document frequency. TF-IDF weight is a weight often used in information retrieval and text mining. How important a word is to a document in a collection is evaluated using this statistical measure. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus. Various TF- IDF weighting schemes are usually used by search engines as a central tool to score and rank a document's relevance to a given user query.

How to Compute:
The TF-IDF weight comprises of two terms: the first computes the normalized Term Frequency (TF), aka. the number of times a word appears in a document, divided by the total number of words in that document; the second term is the Inverse Document Frequency (IDF), calculated as the logarithm of the number of the documents in the corpus divided by the number of documents where the specific term appears.

TF: Term Frequency measures how frequently a term appears in a document. As every document differs in length, it is possible that a term would appear much more times in long documents than shorter ones. Hence, the term frequency is often divided by the document length (aka. the total number of terms in the document) as a way of normalization:

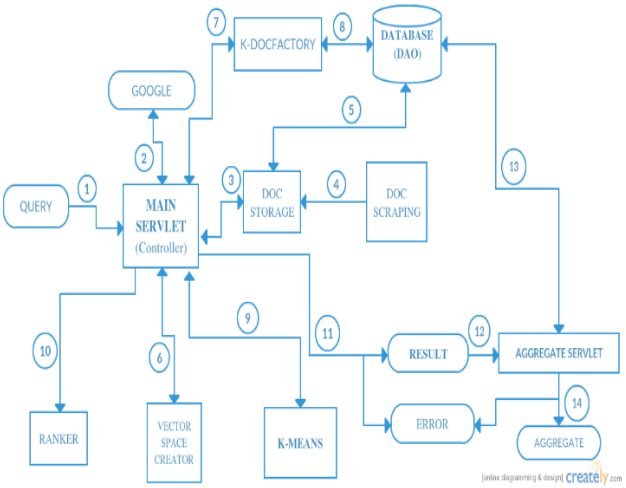
TF (t) = (Number of times term t appears in a document) / (Total number of terms in the document).

IDF: Inverse Document Frequency measures the importance of a term in a document. All terms are considered as of equal importance while computing TF, However certain terms such as "is", "of", and "that", may appear frequently but have little importance. Thus we need to weigh down the frequent terms while scale up the rare ones. This can be computed using the following:

IDF (t) = log_e (Total number of documents / Number of documents with term t in it).

Frequency count plays a major role in ranking data, it has to counts the frequency of word occurrence in text document and also determine which are all the document contain that a word. With the help of frequency count compute the importance of document for that word, suppose the word is repeated for more number of times then the word is related to the subject and it play a major role in that document. By using support vector machine measure the distance between the word and the document with the help of cosine similarity method.

IV. DESIGN



1. Query
This is the first step where the user will enter a query to the system. This query will be directed to the class Servlet.

2. Servlet
This is the main base class which will act as a controller of all the processing to be carried out. All the processing will be redirected by this Servlet class. Firstly it is connected to Google to which the query is forwarded.

3. Google
This will return an array list of strings or links associated or relevant to the query entered by the user. An arraylist is a dynamic array that is of no fixed size.

4. Docstorage and Docscraper
The system returns to the Servlet and it further connects to the 'DocStorage' class which visits the links using the 'Docscraper' class. This class uses Jsoup.Jsoup is a java html parser. It is a java library that is used to parse HTML documents. Jsoup provides API to extract and manipulate data from URL or HTML files. Hence DocScraper class extracts links and text within the links using Jsoup and returns it to the DocStorage class. Further the DocStorage is connected to a database which uses DAO.

5. DAO
DAO is Data Access Object. DAO is used to avoid conflicts and provides flexibility to change the database anytime. The content is stored by DocStorage using DAO.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

6. Vector Space Creator
The Servlet class is further connected to a 'Vector Space Creator' class. Servlet sends the original query to this vector space creator .Here this class performs stop word removal process. Stopwords are common words that carry less important meaning than the keywords. Stop word removal process removes stopwords from the keyword phrase to give the most relevant result.

After this process, a vector is generated out of this original query and sent back to the Servlet.

7. K-DocFactory-
Servlet sends the vector generated to the 'K-docFactory' class. This class is connected with the DAO through which the content stored is accessed. Using this vector, the tf-idf is generated.TF-IDF is the Term Frequency-Inverse Document Frequency is intended to reflect how important a word is to a document in a collection or corpus. The result is in the form of a K-DocList. This K-DocList is sent back to the Servlet class.

8. K-Means
Servlet sends this K-DocList, vector generated, size of the vector and K-value to the 'K-Means' class. This class performs the k-means clustering .The result is obtained in the form of clusters which hold the most relevant links extracted and processed. A clustered arraylist is sent back to the Servlet.

9. Ranker
This clustered arraylist is sent by the Servlet to the 'Ranker' class. This class is used for the ranking process. This process returns the documents with highest frequency being placed first and then in ascending order in respective clusters. After ranking, if there exist any empty clusters then those are also removed.

10. Result
A successful result can be obtained in the form of clusters containing all the relevant links to the query entered by the user. An error can also be encountered as result if the requirements of the system are not satisfied or if a wrong query is entered.

V. RESULT AND ANALYSIS

We tried two different distance measures for calculating the distance between documents and the cluster centroid, which are Euclidean distance and Cosine distance. Euclidean distance allows us to calculate the straight line distance between two points in a vector plane, and is usually one of the most widely used measurements for calculating distances. On the other hand, cosine similarity allows us to calculate both aspects of a vector - direction and magnitude. Direction is the "preference" or "sentiment" of the vector, while the magnitude indicates how potent it is along that direction. Cosine similarity is popular in text mining applications due to the ability to categorize them by their overall sentiment using angular distance. We can use cosine distance as an inverse of cosine similarity in order to find the magnitude of difference rather than similarity. Since our algorithm considers a vector representation of the query for clustering to promote the use of side information in the form of a user query, and not the bag-of-words based representation of the entire text, using cosine similarity or distance doesn't provide as much of an advantage as it usually does in unexplored and unlabelled data. In fact, in many of our analyses, we found that using Euclidean distance usually provides more distinguished and appropriate clusters. This can be observed in the following graphs that we plotted by trying out various queries.

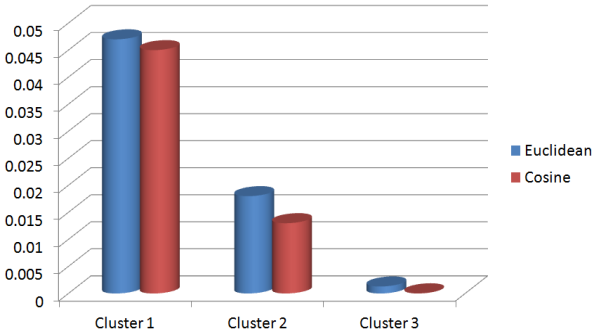


Figure 1 - Query "Game of Thrones"

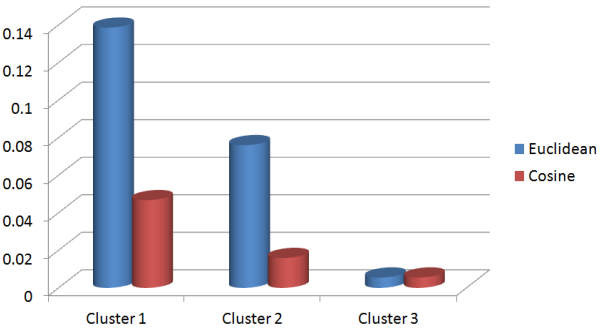


Figure 2 – Query "Programming Languages"

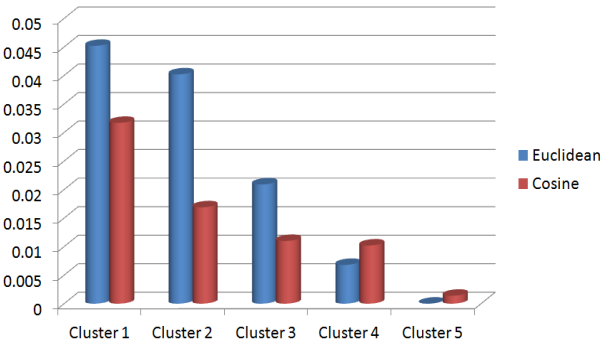


Figure 3 - Query "Harry Potter and the Chamber of Secrets"

From the above graphs where we plot the Total TF-IDF of the clusters using both Euclidean and Cosine distance, we can observe that Euclidean distance gives higher magnitude for most clusters and provides more divergent clusters as well, which enhances the uniqueness of each individual cluster.

VI. CONCLUSION

Computer science is a field which builds and iterates over previously done work to make it more effective. The process of text mining and retrieving information can be greatly enhanced by using an approach that can be effective as well as fast. We suggest a similar approach where processing the results acquired by using a search engine can help us get more appropriate results, faster. Out of the various methods and algorithms available for reducing the dimensionality of text and clustering, we chose k-means algorithm and a bag-of-words approach over more advanced approaches involving Artificial Intelligence and Natural Language Processing. The reason behind this is that the original documents that we receive already serve as a good base for relevant information. Using our approach simply acts as an extra layer to enhance the quality of text mining activity.

The significance of side information for effective clustering and mining includes number of text mining applications that contain side-information with them; this information may be of various kinds including provenance information of the documents, the links in the document, web logs which depict user-access behavior and meta-data. Lot of research work has been carried out in recent years on the clustering issue in text collections in the database and information retrieval society. This work is basically designed for issue of pure text clustering in the lack of other kinds of attributes. These attributes may contain a lot of information for clustering purposes. In this work, we are studying various techniques, for effective text clustering and mining .After studying these techniques we have come to the conclusion that, considering side information for text data clustering and mining is an excellent option because if the side information is related then it can give extremely beneficial results .On the other side if side information is noisy it can be hazardous to merge it into the mining process, as it can add noise to the mining process and degrade the quality of the mining process. So by removing this kind of noise the quality of mining process can be enhanced to a great extent.

Therefore, discussion suggests a way to design efficient algorithm which combines classical partitioning algorithm with probabilistic model for effective clustering approach, so as to maximize the benefits from using side information.

REFERENCES

1. Charu C. Aggarwal, Yuchen Zhao, Philip S. Yu., On the use of Side Information for Mining Text Data, IEEE Transactions, Vol.26, No.6, JUNE 2014.
2. H. Frank, “Shortest paths in probabilistic graphs,” Operations Research, vol. 17, no. 4, pp. 583–599, 1969.
3. L. G. Valiant, “The complexity of enumeration and reliability problems,” SIAM J. Comput., vol. 8, no. 3, pp. 410–421, 1979.
4. N. J. Krogan, G. Cagney, and al., “Global landscape of protein complexes in the yeast saccharomyces cerevisiae,” Nature, vol. 440, no. 7084, pp. 637–643, March
5. S. Guha, R. Rastogi, and K. Shim, CURE: An efficient clustering algorithm for large databases, in Proc. ACM SIGMOD Conf.,
6. R. Ng and J. Han, Efficient and effective clustering methods for spatial data mining, in Proc. VLDB Conf., San Francisco, CA, USA, 1994, pp. |144155.
7. T. Zhang, R. Ramakrishnan, and M. Livny, BIRCH: An efficient data clustering method for very large databases, in Proc. ACM SIGMOD Conf., New York, NY, USA, 1996, pp. 103114.
8. Lei Meng, Ah-Hwee Tan, Dong Xu Semi-Supervised Heterogeneous Fusion for Multimedia Data Co-Clustering IEEE Transactions On Knowledge And Data Engineering, vol. 26, no. 9, pp.2293-2306, 2014.
9. Vishal Gupta, Gurpreet S. Lehal, A Survey of Text Mining Techniques and Applications , in Journal of Emerging Technologies in Web Intelligence, Vol. 1, No. 1, August 2009, pp.60-76
10. C. C. Aggarwal and C.-X. Zhai, Mining Text Data. New York, NY, USA: Springer, 2012
11. I. Dhillon, Co-clustering documents and words using bipartite spectral graph partitioning, in Proc. ACM KDD Conf., New York, NY, USA, 2001, pp. 269274.
12. T. Liu, S. Liu, Z. Chen, and W.-Y. Ma, An evaluation of feature selection for text clustering, in Proc. ICML Conf., Washington, DC, USA, 2003, pp. 488495.
13. R. Shamir, R. Sharan, and D. Tsur, “Cluster graph modification problems,” Discrete Applied Mathematics, vol. 144, no. 1-2, pp. 173–182, 2004.
14. M.Steinbach, G. Karypis, and V. Kumar, “A comparison of document clustering techniques,” in Proc. Text Mining Workshop KDD, 2000, pp. 109–110.