IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 16, Issue 4, Ver. V (Jul – Aug. 2014), PP 66-72
www.iosrjournals.org

Frequent Itemset-based Text Clustering Approach to Cluster Ranked Documents

Snehalata Nandanwar[1], Geetanjali Kale[2], and Sheetal Sonawane[3]
[1](Department of Computer Engineering,Pune University,India)
[2](Department of Computer Engineering,Pune University,India)
[3](Department of Computer Engineering,Pune University,India)

Abstract: *In most of the search engines documents are retrieved and ranked on the basis of relevance. They are not necessarily ranked on the basis of similarity between the query and the respective document. The ranked documents are in the form of a list. So there is a need to rank the retrieved documents. We implement Language model that is efficient to retrieve the text documents satisfying given query. The ranking of documents is based on the similarity coefficient calculated using language model. To cluster the ranked retrieved documents, we use FITC (Frequent Item set-based text Clustering) algorithm. The algorithm partitions the documents and returns the cluster in each partition. The algorithm identifies clusters with no overlap. The system accepts user request and returns all the relevant documents partitioned in the form of clusters which satisfy the query. The results of applying the algorithm to document retrieval demonstrate that the algorithm identifies non-overlapping clusters and is therefore of widespread use in many of the search engines.*
Keywords: *Clustering algorithms, Frequent Itemset-based text Clustering, Language model*

I. Introduction

Over the past two decades, many approaches for document retrieval are studied extensively. Retrieval strategies give similarity measure between a query and a document. There are many retrieval strategies viz., Vector space model, Probability retrieval model, Language model, Inference networks, Boolean indexing etc. In our paper we implement the language model. In this documents are ranked on their probability of creating the query.

The general systems generate the results on the basis of keyword matching which does not capture the semantic meaning of the request. Most of the systems return the results ranked by their relevance to a new request. Users need to go through entire list one by one to analyse their desired search. It is a time consuming task if the result list is too long. To make it easy we group the similar documents in the form of clusters. It would help users not only in identifying their desired context but also by minimizing the efforts of attending all the documents.

To address the problem we rank the documents on the basis of language model which is really more influential than the most common probabilistic model or cosine similarity model or inference networks.

Document clustering is well-known in the field of Information Retrieval (IR). There are three basic challenges while clustering the documents:

1. A huge collection of documents
2. Data with multi dimensions
3. Specific description of cluster

While document clustering, documents are characterized as vectors with many dimensions where each dimension corresponds to a unique keyword occurred in the document. While clustering, the similarity coefficients of each pair of documents need to be calculated and nearest neighbour of each document is found out. It is really a tedious task.

Many algorithms are available for the purpose of clustering. These algorithms fall into basic four categories: partitioning methods, hierarchical methods, density based methods and grid-based methods.

The rest of this paper is organized as follows. The discussion on retrieval strategies and clustering methods is explained in section II. Section III briefly introduces the framework and mathematical model. Section IV presents the procedure for pre-processing the cases and performing semantic role analysis. Section V describes our proposed case-ranking approach. The case-clustering algorithm is proposed in Section VI. Conclusion is given in the VII section.

II. Related Work

In this paper, we develop a document retrieval system that can automatically search and rank the existing documents and group the top-ranked documents into clusters. Some related topics and techniques are reviewed in this section.

1) Case-based systems: Many keyword-based systems search the solution on the basis of the entered keywords [2],[3]. In this, the first candidate set is retrieved by using an initial information supplied by the user and then to narrow down the result set the system ask the informative keywords to user. The procedure is repeated till few documents remain or the most suitable documents are found.

2) Search and ranking: In document retrieval, many methods have been proposed to find out the similarity measurements and rank the results of a query, viz., Cosine similarity, Probabilistic model, Language model, Inference networks [5]-[8],[14]. However, similar to the case-based systems, the similarity is measured based on keyword matching, which have difficulty to understand the text deeply.

3) Clustering search results: Since existing search engines often return a long list of search results, clustering technologies are often used in search for result organization [9],[10]. However, the existing document-clustering algorithms do not consider the impact of the general and common information contained in the documents. In our work, by filtering out this common information, the clustering quality can be improved, and better context organizations can then be obtained.

Now we have a brief look through the major available document retrieval and ranking models. In vector space model both query and the documents are represented as vectors in term space. Each component of these vectors is a distinct term that occurs in the document collection. The similarity coefficient is calculated between a query and each document by using a dot product [14]. The model is based on the idea that the meaning of a document is conveyed by the words used in it. If one can represent the words in the document by a vector, it is possible to compare documents with query to determine how similar their content is. Documents whose contents, as measured by the terms in the document, correspond most closely to the contents of the query are judged to be most relevant. To determine an importance of the term user can manually assign the weight to each term or weight can be assigned automatically based on the frequency of the term that occurs in the document collection. The probabilistic model computes the similarity coefficient (SC) between a query and a document as a probability that the document will be relevant to the query [14]. In this probabilities are assigned to the components of the query and then each of these is used in calculating the final probability of the document relevance with the query. To assign the probability sufficient training date about the term occurrence in relevant and non-relevant documents is required. It is very difficult to obtain sufficient training data about these parameters. This is the problem with the probabilistic model. Inference networks use evidential reasoning to estimate the probability that a document will be relevant to the query [14]. The core of the inference network is to consider the known relationships and use them to "infer" other relationship. In binary inference network, events are used where every event will have either a true or false value. The likelihood of event is indicated by a prior probability. Once the initial values at the top level are assigned, a node on the network is instantiated and is able to compute the belief that the node is true or false. The belief is computed based on the belief that all of its parent nodes are true or false. It is always not possible to have the prior knowledge of all the events. So this model cannot be used frequently.

We use the Language model for document retrieval and ranking purpose. In this, the likelihood of documents of generating the query can be used for ranking the documents. It is different from the vector space model where only term frequency and inverse document frequency are considered in the calculation of SC of documents. And there is no need to collect training data for assigning probability to individual components like it is in probabilistic retrieval. Also it does not require any inference network to understand the relationships. Each calculation is independent and depends on the current data.

To cluster the ranked documents we use Frequent Term based text Clustering as the algorithm is considering all the possible combinations of the words occurred in the retrieved documents. In the next paragraph, we compare the different clustering algorithms which are used most frequently.

Basically there are only four broad categories which are mentioned earlier. In partitioning methods, for a given set of n objects, k partitions of data are created, where each partition represents a cluster and $k \leq n$. Most partitioning methods are distance-based. A partitioning method creates k number of partitions. Algorithms like k-means, bisecting k-means, k-medoids fall into this category. In k-means algorithm the centroid of each cluster is defined as the mean value of the points present in that cluster. It randomly selects k objects from D. Each of them is considered as the centroid of cluster. It iteratively assigns each of the remaining documents to the cluster based on the Euclidean distance between the object and cluster and calculates the new mean using the newly assigned objects. K-medoids method work in the same way like k-means. Only difference is in each iteration it tries to replace the representative item with any non-representative item present in the same cluster if the SC of the newly added object and the non-representative is better than the SC of the newly added one and the previous one.

A hierarchical method creates a hierarchical decomposition of the given set of objects. It groups the data objects into a hierarchy. A hierarchical method is either agglomerative or divisive. The agglomerative approach is called bottom-up approach. It starts with each object forming a separate group. It successively merges the objects or groups close to one another, until all groups are merged into one or a termination condition holds. The divisive approach, also called as top-down approach. It starts with all the objects in the same cluster. In each successive iteration, a cluster is split into smaller clusters, until eventually each object is in one cluster or a termination condition holds. Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) and Chameleon are the algorithms which fall under this category. In BIRCH the clustering feature (CF) of the cluster is defined as a triplet of n, number of data objects; LS, the linear sum of n points and SS, the square sum of data points. In the first phase of BIRCH, it scans the data set to build an initial CF-tree and in the second phase, the clustering algorithm is applied to cluster the leaf nodes of the CF-tree which groups dense clusters into larger ones. In Chameleon, similarity between the clusters is assessed based on well connected objects within it and closeness between them.

In density-based methods, the dense regions in the data space are modelled as clusters. Density of any object is measured by the number of objects close to it. Cluster is continuously growing as long as the density in the "neighbourhood" exceeds some threshold. Density –Based Clustering based on Connected Regions with High Density (DBSCAN) and Ordering Points To Identify the Clustering Structure (OPTICS) fall into this category. In DBSCAN, the user specified parameter is used

The grid-based clustering approach uses a multi-resolution grid data structure. It quantizes the object space into a finite number of cells that form a grid structure on which all of the operations for clustering are performed. STING is a grid-based multi-resolution technique in which the embedding spatial area of the input objects is divided into rectangular cells. CLIQUE is a simple grid-based method for finding density-based clusters in subspaces [10].

Lots of algorithms are available to cluster the data. We have chosen FITC because it captures maximum possible combinations of terms.

III. Framework

Fig.1 shows the framework of proposed system. The input of the system is the user request. On the other hand, the collected documents are cleaned by removing formatting characters and stop words. They are stored in the form of keywords that form the context of documents. In the document-ranking module, the documents are ranked on the basis of their semantic importance to the pre-processed input request. The proposed ranking method is discussed in section V. Along with searching and ranking the documents, the system also groups the related documents using method FTC.

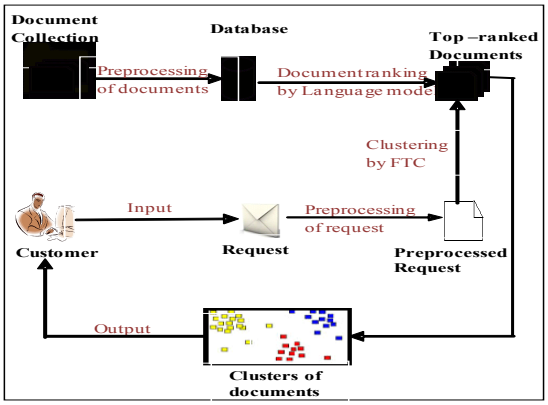

Fig. 1. Framework of the proposed system

Mathematical Model

Let I be the set of input of the system and O be the set of output. I= {q , D}

where q represents the user query, D represents the document collection.Both q and D need to execute a set of pre-processing functions, F.

F= {F_1, F_2, F_3} where F_1 = Function that separates tokens from document and query; F_2 = Function for tagging parts of speech; F_3 = Function for stemming.

$$O=\begin{cases} \{C_1, C_2, \ldots \ldots C_k\} & \text{when query matches with documents} \\ \Phi & \text{when query does not match with any of the documents} \end{cases}$$

where C_i represent a cluster; $C_i=\{D_j \mid D_j \in D\}$

IV. Document Preprocessing

In the proposed system we use OpenNLP which is a machine learning toolkit for the processing of natural language text. It supports the most common NLP tasks such as tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing. After separating out the tokens we remove the stop-words. Documents and query are represented as a collection of keywords. For each pair of query and document, we discover the semantic relations of terms using WordNet [13]. If two words are identified by the semantic relation such as synonym, the words are considered as "related".

The words appear in documents and queries often have many morphological variants. Thus, pairs of terms such as "computing" and "computation" will not be recognised as equivalent without some form of natural language processing(NLP). Morphological variants of words have similar semantic interpretations and can be considered as equivalent. For this reason, Porter's algorithm is applied which attempt to reduce a word to its stem or root form. The key terms of a query and documents are represented by stems rather than by the original words. Based on these keywords the search is carried out. The result consists of the list of documents.

V. Language Model

After retrieving the initial list we apply the language model to find out the similarity between our entered query and all documents in the list. Depending on the similarity coefficients the documents are ranked in the final display list. The statistical language model is a probabilistic mechanism for generating a piece of text. It thus defines the distribution over all possible word sequences. The core idea is that documents can be ranked on their likelihood of generating the query. In this case the probability of the query is calculated as the product of probabilities for both the terms in the query and terms those are absent in the same [14].That is,

$SC(Q,D_i) = \Pi_t$

To avoid the problem caused by the terms in the query that are not present in a document, various smoothing approaches exist which estimates non-zero values for these terms. The implementation of language model is explained by using the following equations. The following parameters are listed out:

dl_d - document length i.e. total number of tokens in the document; df_t - the document frequency of term t (the number of documents in which term t appears) ;$tf_{t,d,}$- the term frequency of each term for the given document d (the number of times term appears in the document d) ; cf_t - The raw count of each term t in the collection.

First, $P_{ml}(t|M_d)$, the maximum likelihood estimate of the probability of term t under the term distribution for document d is calculated. For each term, t,

$$P_{ml}(t|M_d) = \frac{tf_{t,d}}{dl_d} \qquad (1)$$

Second, we calculate the mean probability of term t in documents which contain the term. The equation is:

$$P_{avg}(t) = \frac{\sum_{d(t\epsilon d)} P_{ml}(t|M_d)}{df_t} \qquad (2)$$

Third, evaluate risk values for a term t for a document d.

To do that, $f_{t,}$ the mean term frequency of term t in a document d is computed by the following equation

$f_t = P_{avg}(t) \times dl_{d.}$. We then use the following risk function

$$R_{t,d} = \left(\frac{1.0}{1.0+f_t}\right) \times \left(\frac{f_t}{1.0+f_t}\right)^{tf_{t,d}} \qquad (3)$$

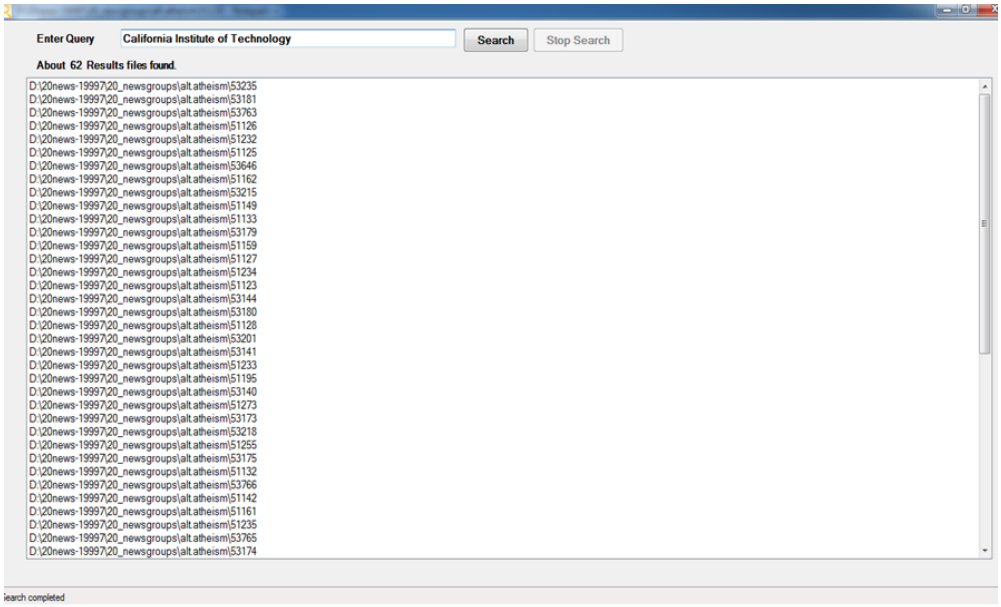

Fig.2 GUI for document retrieval

Now we use the risk value as a mixing parameter to calculate P ($Q|M_d$), the probability of producing the query for a given document model. It consists of two steps. Initially, we calculate $P(t|M_d)$ for term which matches a query term. For all other term occurrences, the smoothing estimate $P(t|M_d) = \frac{cf_t}{cs}$ is used where cs is the total number of terms in document collection.

$$P(t|M_d) = P_{ml}(t|M_d)^{(1.0-R_{t,d})} \times P_{avg}(t)^{Rt,d}$$

Using the above equation, we compute a final measure of similarity, $P(Q|M_d)$.

$$P(Q|M_d) = \prod_{t\epsilon Q} P(t|M_d) \times \prod_{t\notin Q}(1.0 - P(t|M_d))$$

After calculating the similarity coefficients of the relevant documents they are ranked in accordance with their scores. The documents are ranked in non-increasing order of their similarity scores.
The screenshot of the ranked documents is given.

VI. Frequent Itemsets-Based Text Clustering

Frequent item sets form the basis of association rule mining. Algorithm like Apriori is used for that [10]. Frequent item sets can also be used for the purpose of classification which will group the documents where these sets find. Since we are dealing not with transactions but with documents, we will use the notion of item sets. An item set is any collection of terms within a document and a document can be considered if and only if an item set found in that document at least once.

The key idea is first divide the set of documents into different partitions based on the occurrence of frequent item sets and then we form the clusters within each of the partition by calculating the similarity between them. In this section, we present the algorithm for frequent item sets-based text clustering.

Definitions

Let $D = \{D_1, D_2, \ldots\ldots, D_n\}$ be a database of text (hypertext) documents and T be the set of all terms occurring in the documents of D. Each document D_i is represented by the set of terms occurring in D_i, i.e. $D_i \subseteq T$.
Let minsup be a real number, $0 \leq minsup \leq 1$. For any set of terms S, $S \subseteq T$, let cov (S) denote the cover of S, the set of all documents containing all terms in S, i,e. the set of all documents supporting S. More precisely, $cov(S) = \{D_i \in D \mid S \subseteq D_i\}$.

Mining of Frequent Itemsets

Let $F = \{F_1, F_2, \ldots.., F_k\}$ be the set of all frequent term sets in D with respect to minsup, i.e.
$F = \{F_i \subseteq T \mid |cov(F)| \geq minsup \mid D|\}$
This F is a set of unique terms, $F= [t_1, t_2, \ldots\ldots, t_q]$

The binary database B is formed from these top p-frequent terms where binary data indicates the presence and absence of that term in the document D_i .

$$B_t = \begin{cases} 1 & \text{if } tj \in Di \\ 0 & \text{if } tj \notin Di \end{cases}$$

We then apply Apriori algorithm to find out the frequent item sets of all possible lengths q where q= 1 ,2,3……..k.

There are two steps in Apriori algorithm:

1) The join step

To find item set of length k i.e. L_k , two item sets of lengths L_{k-1} are joined. Suppose l_1 and l_2 are those two item sets. The notation $l_i[j]$ refers to the j^{th} item in item set ⊠ The join l_1 l_2 is done if all k-2 items are similar in both sets so that final set contains these [k-2] items plus one different item from each set.

2) The prune step

C_k is a set of all item sets of length L_k. These members may or may not be frequent. To remove infrequent item sets, a document scan is done and all those item sets are removed from C_k which are not satisfying the minsup. So final C_k will consist of only frequent item sets.

Partitioning the documents based on frequent item sets

An Apriori generates the set of frequent item sets of varying lengths q where q = {1,2,……,k}. First the set of all frequent item sets of each length q is sorted in descending order of their minsup values.

$F_s = \{f_1 , f_2,\ldots\ldots , f_k\}$; $1 \leq q \leq k$ where $minsup(f_l(1)) \geq minsup(f_l(2)) \geq minsup(f_l(3)) \ldots\ldots\ldots \geq minsup(f_l(n))$.

$f_q = \{f_{l(m)} \mid 1 \leq m \leq p\}$ where p denotes the number of frequent item sets in set f_q.

Now we are ready to create the partitions. The procedure of creating the partitions is mentioned below. Initially, a sorted list $f_{k/2}$ is taken. Initial partition P_1 is created by finding out all the documents those are having item set $f_{k/2}(1)$. Then we take the next element in the same list i.e. $f_{k/2}(2)$ to form the new partition P_2. This partition contains all the documents containing frequent item set $f_{k/2}(2)$ and takes away the documents those were present in partition c_1 which also contain $f_{k/2}(2)$. The procedure is repeated until all the documents are not included in any of the partition P_r. Furthermore, if the above procedure is unable to partition all documents with the sorted list $f_{k/2}$ then the subsequent sorted list ($f_{(k/2)-1}$, $f_{(k/2)-2}$) are considered. This results a set of partitions, P and every P_r contains a set of documents.

$P = \{P_r ; 1 \leq r \leq s\}$ where s denotes the number of partitions.

$P_r = \{D_i [f_{q(m)}] \mid D_i \in D\}$ where $D_i [f_{q(m)}]$ denotes
the document containing frequent term m of length q.

Clustering the documents within each partition

This section describes how to cluster the documents in each partition. Documents can be clustered based on their similarities which is apart from the frequent item set those are binding them together. In this phase, we first identify the documents and frequent item sets in each partition. Then we find the derived keywords by taking the set difference of the set of top p frequent words of documents and the set of keywords in frequent item set in each partition.

$K_d (P_r) = \{T_j\} - \{f_{q(m)}\}$; $T_j \in D^x_{cr}$ and D^x_{Pr} denotes document D in partition P_r.

So $K_d(P_r)$ returns a set of words those are frequent but not present in frequent item set of that partition. From those keywords the representative keywords can be found out. The representative keywords are the keywords those are derived keywords and also satisfy the cluster support(cl_sup).

Definition: Cluster support(cl_sup) of a keyword is defined as the number of documents in the cluster those contain the keyword.

Now, find the similarity coefficient of the documents and the representative words. The similarity S is computed as follows,

$$S(D_i , R_w) = \frac{|Kd\ (Di) \cap Rw\ (cr)|}{Rw\ (cr)}$$

The documents within the partition are sorted according to their similarity coefficient and if necessary create a new cluster if similarity coefficient exceeds the minimum threshold value.

VII. Conclusion And Future Scope

We feel that ranking of documents definitely help in finding proper match. Instead of going through the list of documents and deciding its priority manually, our system will do this for user. Also partitioning and clustering of ranked documents will be definitely helpful to explore the search. Because many times we are looking for a fixed combination of terms which is not possible with the help of the normal search technique . So this clustering strategy will be helpful to go and refer the appropriate group.

Data Set

We are referring a Newsline data set. (http://www.trec.nist.gov) This test data set consists of news from 20 different categories.

References

[1] D. Wang, T. Li, S. Zhu, Y. Gong, "iHelp : An Intelligent Online Helpdesk System," IEEE Trans. on System, Man and Cybernetics– Part B: Cybernetics, vol.41, no. 1, pp. 173-182, Feb. 2011.

[2] S.Murali Krishn, S.Durga Bhavani "An Efficient Approach for Text Clustering Based on Frequent Itemsets," European Journal of Scientific Research, 1450-216X Vol.42 No.3 (2010), pp.385-396

[3] D. Bridge, M. Goker, L. Mcginty, B. Smyth, "Case-based recommender systems," Knowl. Eng. Rev., vol. 20, no. 3, pp.315–320, Sep. 2005.

[4] P. Cunningham and B. Smyth, "A comparison of model-based and incremental case-based approaches to electronic fault diagnosis," Dept. Comput. Sci., Trinity College Dublin, Dublin, Ireland, Tech. Rep. TCDCS- 94-21, 1994

[5] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is nearest neighbor meaningful?" in Proc. ICDT, 199, pp. 217– 235.

[6] S. Agrawal , S. Chaudhuri , G. Das, and A. Gionis, "Automated ranking of database query results," in Proc. CIDR, 2003, pp. 888–899.

[7] K. Chakrabarti, V. Ganti, J. Han, and D. Xin,"Ranking objects based on relationships," in Proc. SIGMOD, 2006, pp. 371–38.

[8] S. Chaudhuri , G. Das , V. Hristidis, and G. Weikum, "Probabilistic ranking of database query results," in Proc. VLDB, 2004, pp. 888–899.

[9] G. Das, V. Hristidis, N. Kapoor, and S. Sudarshan, "Ordering the attributes of query results," in Proc. SIGMOD, 2006, pp. 395–406.

[10] A. Leuski and J. Allan , " Improving interactive retrieval by " Improving interactive retrieval by combining ranked list and clustering," in Proc. RIAO,2000, pp.665–681.

[11] J. Han , M. Kamber , J. Pei, "Data Mining – Concepts and Techniques," 3rd ed.,Morgan Kaufmann , pp. 443-519.

[12] C. Fellbaum. WordNet : An Electronic Lexical Database. Cambridge, MA: MIT Press,1998[Online] Available: http://www.Download.cnet.com/WordNet-online-dictionary

[13] D.Grossman, O.Frieder, "Information Retrieval Algorithms and Heuristics,",2nd ed.,Springer International Edition,pp. 9-84

[14] F. Beil , M. Ester, X. Xu (2002). Frequent Term- Based Text Clustering. Presented at SIGKDD 02 Edmonton, Alberta, Canada

[15] R. Baeza- Yates, B. Ribeiro – Neto, "Modern Information Retrieval", ACM Press, pp. 433-436