ISSN : 2231-0711

Query Dependant Single Document Summarization using Partitional Clustering: K-Means Clustering Approach

Giri Virat ,Mrs.Bewoor Mrunal ,Dr. Patil S.H
Computer Engineering from Bharati Vidyapeeth University College of Engineering, Pune,India

***Abstract*-A huge amount of on-line information is available on the web, and is still growing. While search engines were developed to deal with this huge volume of documents, even they output a large number of documents for a given user's query. Under these circumstances it became very difficult for the user to find the document he actually needs, because most of the users are reluctant to make the cumbersome effort of going through each of these documents. Therefore systems that can automatically summarize one or more documents are becoming increasingly desirable.**

A summary can be loosely defined as a text that is produced from one or more texts. Automatic summarization is to use automatic mechanism to produce a finer version for the original document.

This paper presents the results of an experimental study of K-means document clustering techniques. We have implemented the query dependant single document summarization by using clustering approach. We have implemented k-means algorithm for only .txt file.

The performance of the algorithm is analyzed on different evolution factors like execution time, number of words in summary, number of computational loops etc.

Keywords- K-means Clustering, Summarization, weighted Document graph,Clustered graph.

1.0 INTRODUCTION

Sparck Jones [1] defines summary to be a condensed derivative of source text, i.e. a content reduction through either selection or generalization on what is important in the source. It is a short version of document with only the important information. The definition of the summary, though is an obvious one, emphasizes the fact that summarizing is in general a hard task, because we have to characterize the source text as a whole, we have to capture its important content, where content is a matter of both information and its expression, and importance is a matter of what is essential as well as what is salient.

Different kinds of summaries were identified [2, 9, 6, 4] based on the function of the summary. Indicative summaries provide an idea of what the text is about without conveying specific content and informative ones provide some shortened version of the content. Topic-oriented summaries concentrate on the reader's desired topic/topics of interest, whereas generic summaries reflect the author's point of view. Extracts are summaries created by picking portions (words, sentences, etc.) of the input text verbatim, while abstracts are created by regenerating the extracted content. Till now most of the researchers focused on producing extracts, with their concentration being on making the extract either indicative or informative. Indicative summaries usually serve the functions of announcement and screening. By contrast informative summaries are of function of substitution. Critical summaries criticize an approach or an opinion expressed in the text document. Extract can be of announcement and replacement. In general, all of the four types of summaries are retrospective. Indicative and informative summaries are the most important types in the current internet environment.

Summaries are influenced by a broad range of factors. The problem is that all the factors and their various manifestations are hard to define, so capturing them precisely enough to guide summarizing in particular cases is very hard. Sparck Jones [9] broadly classified these factors into three types:

1. **Input factors:** *source form, subject type and unit*

2. **Purpose factors:** *situation, audience and use*

3. **Output factors:** *material, format, style, expression and brevity*

The summarization is the process of generating Output guided by summary purpose under constraints from input features.

Sparck Jones argues automatic text summarization to be a three stage model:

I: source text *interpretation* to source text representation.

T: source representation *transformation* to summary text representation.

G: summary text *generation* from summary representation.

Here we discuss general areas of research including single-document summarization, multi document summarization and query based summarization.

1.1 Single-Document Summarization

Despite the research in alternatives to extraction, majority of the work still relies on extraction of sentences from the original document to form the summary. These approaches focused on the development of relatively simple surface-level techniques that tend to signal important passages in the source text. Although most systems use sentences as units, some work has been done with larger passages, typically paragraphs. These techniques for sentence extraction computed a score for each sentence based on features such as position in the text [3,5], word, phrase frequency [10] and key phrases (e.g., \it is important to note") [5]. Recent extraction approaches use more sophisticated techniques for deciding which sentences to extract; these techniques often rely on machine learning to identify important features, on natural language analysis to

identify key passages [7], or on relations between words rather than bags of words.

1.2 Multi-document Summarization

Multi-document summarization, the process of producing a single summary of a set of related source documents, has gained a researchers attention in the past decade. The three major problems introduced by having to handle multiple input documents are:

1. Recognizing and coping with redundancy
2. Identifying important differences among documents
3. Ensuring summary coherence

1.3 Query-Based summarization

With the belief that in the Information Retrieval scenario, if users could see the sentences in which their query words appeared, they could better judge the relevance of the documents, [8] considered generating the query based summaries. The query based summarization task is to generate a summary of single/multiple documents which is focused towards the user's query. In general, a query score was calculated for each sentence based on the distribution of query terms and added to its overall score obtained by sentence extraction methods. The top scoring sentences were used as a summary for each of the retrieved document. Based on the observation that human generated query-based summaries will contain significant of document sentences which doesn't contain the query terms [12] studied the effect of sentence and document related features along with the logistic regression model to generate query based summaries. Named entities and question words were used to calculate how well a sentence satisfies predefined constraints [11] and top scoring sentences were selected to form the summary.

2.0 SYSTEM

System can be divided into four different parts. In first part we will create cluster of document based , in second part document graph will be created .Third part will be adding weight to the document graph and in last part weight will be assigned to nodes in document graph and summary will be created.

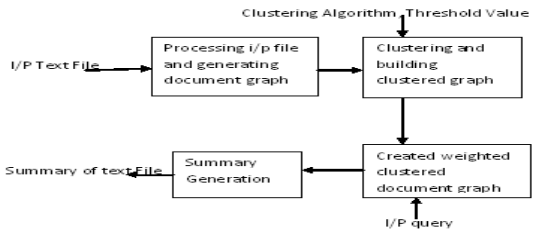

Fig1.Architecture of system

2.1 Pre-processing and Creating The Document Graph.

Each cluster becomes a node in the document graph. The *document graph G(V,E)* of a document *d* is defined as follows: *d* is split to a set of non-overlapping clusters *t(v)*, each corresponding to a node $v \in V$. An edge $e(u,v) \in E$ is added between nodes $u, v \in V$ if there is an association between *t(u)* and *t(v)* in *d*.

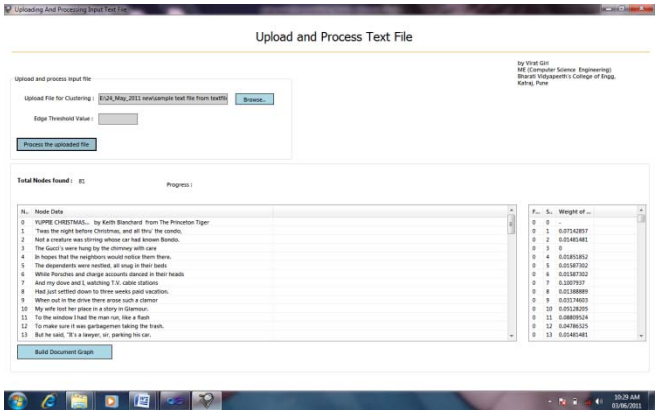

Fig 2. Input file processing

Hence, we can view *G* as an equivalent representation of *d*, where the associations between text fragments of *d* are depicted. A weighted edge is added to the document graph between two nodes if they either correspond to adjacent cluster node or if they are semantically related, and the weight of an edge denotes the degree of the relationship. Here two clusters are considered to be related if they share common words (not stop words) and the degree of relationship is calculated by "***Semantic parsing***". Also notice that the edge weights are **query-independent**, so they can be pre-computed.

2.2 Adding Weighted Edges To The Document Graph

(Note : Adding weighted edge is query independent)

A weighted edge is added to the document graph between two nodes if they either correspond to adjacent node or if they are semantically related, and the weight of an edge denotes the degree of the relationship. Here two nodes are considered to be related if they share common words (not stop words) and the degree of relationship is calculated by "Semantic parsing". Also notice that the edge weights are **query-independent**

The system accepts input text file. The file is read and stored into a string. The string is then split by the newline keyword. The split file is assigned to the string array as the split function returns the string array. The array contains paragraphs which are further treated as nodes**.**

The next stage is to find the similarity between the nodes that means finding the similarity edges between nodes and finding their similarity or weight. Each paragraph becomes a node in the document graph.

2.3 k-means clustering

Clustering is grouping of similar nodes (The nodes which shows degree of closure greater than or equal to the Cluster Threshold specified by the user) into a group. The k-means approach of clustering is used.

The *k*-means algorithm assigns each point to the cluster whose center (also called centroid) is nearest. The center is the average of all the points in the cluster — that is, its coordinates are the arithmetic mean for each dimension separately over all the points in the cluster.

Example: The data set has three dimensions and the cluster has two points: $X = (x_1, x_2, x_3)$ and $Y = (y_1, y_2, y_3)$. Then the centroid Z becomes $Z = (z_1, z_2, z_3)$, where $z_1 = (x_1 + y_1)/2$ and $z_2 = (x_2 + y_2)/2$ and $z_3 = (x_3 + y_3)/2$

The algorithm steps are:

- Choose the number of clusters, *k.*
- Randomly generate *k* clusters and determine the cluster centers, or directly generate *k* random points as cluster centers.
- Assign each point to the nearest cluster center.
- Recomputed the new cluster centers.
- Repeat the two previous steps until some convergence criterion is met (usually that the assignment hasn't changed).

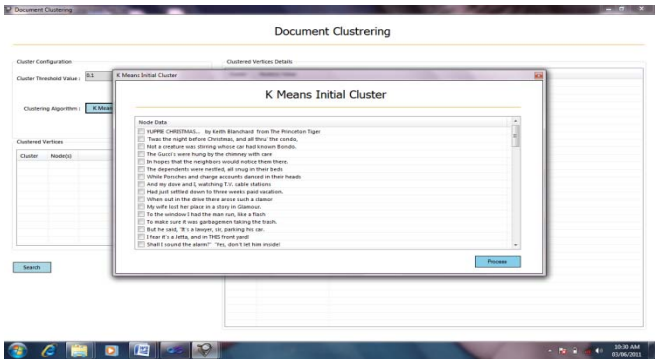

Fig 3. Initial cluster for K-means

2.4 Adding Weight To Nodes In Document Graph

When a query *Q* arrives, the nodes in *V* are assigned query-dependent weights according to their relevance to *Q*. In particular, we assign to each node *v* corresponding to a text fragment *t(v)* node score *NScore(v)* defined by the Okapi formula as given below.

$$\sum_{t \in Q,d} \ln \frac{N - df + 0.5}{df + 0.5} \cdot \frac{(k_1 + 1)tf}{(k_1(1-b) + b\frac{dl}{avdl}) + tf} \cdot \frac{(k_3 + 1)qtf}{k_3 + qtf}$$

tf is the term's frequency in document, *qtf* is the term's frequency in query, *N* is the total number of documents in the collection,*df* is the number of documents that contain the term,*dl* is the document length (in words), *avdl* is the average document length and *k1* (between 1.0–2.0), *b* (usually 0.75), and *k3* (between 0–1000) are constants.

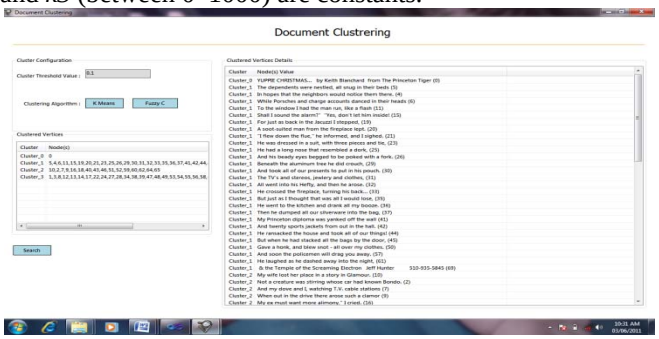

Fig 4 k-means clustered document graph

3. CONCLUSION

File Name	Size in Kb	Word	K Means summary word	K-mean Loops	Kmeans Time
beerwarn	2 kb	319	275	116	161
beerwarn	2kb	319	274	116	138
amhack	13kb	2134	1968	454	234
amhack	13kb	2134	1756	454	279
8bitcomp	24kb	4067	2936	706	1872
8bitcomp	24kb	4067	2878	706	468
crackam1	62kb	11899	9877	1936	14787
crackam1	62kb	11899	9746	1936	1587

Table 1. Results of different file

Table 1 is showing the different file that used for testing the application. Results are encouraging one and we are at the conclusion that performance of k-means algorithm is depend upon the initial cluster that we select. As the cluster threshold value is changing performance parameter ae also changing .We are at a conclusion that cluster threshold near to 0.75 giving us good summary.

4. REFERENCES

[1] Karen Sparck Jones. What might be in a Summary? In Knorz, Krause, and Womser-Hacker, editors, Information Retrieval 93: *Vonder Modellierung zur Anwendung*, pages 9-26, Konstanz, DE, 1993.

[2] H. Borko and C. Bernier. Abstracting Concepts and Methods. Academic Press, 1975.

[3] P.B. Baxendale. Man-made index for technical literature An experiment. *In IBM Journal of Research and Development*, pages 354-361, 1958.

[4] Edward T. Cremmins. *The Art of Abstracting*. Information Resources Press, Arlington, VA, second edition, 1996.

[5] H. P. Edmundson. New Methods in Automatic Extracting. Journal of . ACM, 16(2):264-285, 1969

[6] Hovy Eduard and Chin-Yew Lin. *Automated text summarization in SUMMARIST*. In I. Mani and M. T. Maybury, editors, Advances in Automatic Text Summarization, pages 81{94. MIT Press, Cambridge, 1999.

[7] Ernesto D Avanzo, Bernardo Magnini, and Alessandro Vallin. *Keyphrase Extraction for Summarization Purposes:* The LAKE System at DUC2004. In Proceedings of Document Understanding Conferences, 2004.

[8] Anastasios Tombros and Mark Sanderson. Advantages of Query Biased Summaries in Information Retrieval. In Research and Development in Information Retrieval, pages 2-10, 1998.

[9] Karen Sparck Jones. *Automatic summarizing: Factors and Directions.* In I. Mani and M. T. Maybury, editors, Advances in Automatic Text Summarization, pages 1{13. MIT Press, Cambridge, 1999.

[10] H. P. Luhn. The Automatic Creation of Literature Abstracts. *In IBM Journal of Research and Development*, pages 159-165, 1958.

[11] Jiayin Ge, Xuanjing Huang, and LideWu. *Approaches to Event-Focused Summarization Based on Named Entities and Query Words.* In Proceedings of Document Understanding Conferences, 2003.

[12] Judith D. Schlesinger and Deborah J. Baker. *Using Document Features and Statistical Modeling to Improve Query-based Summarization.* In Proceedings of Workshop on Document Understanding Conferences, DUC01, New Orleans, LA, 2001.

[13]*'A System for Query-Specific Document Summarization*', by Ramakrishna Varadarajan, Vangelis Hristidis. ACM CIKM 2006 at *Arlington*, Virginia.