Context Sensitive Text Summarization using K Means Clustering Algorithm

Harshal J. Jain, M. S. Bewoor, S. H. Patil

Abstract:- The field of Information retrieval plays an important role in searching on the Internet. Most of the information retrieval systems are limited to the query processing based on keywords. In the information retrieval system matching of words with huge data is core task. Retrieval of the relevant natural language text document is of more challenging. In this paper we introduce the concept of OpenNLP tool for natural language processing of text for word matching. And in order to extract meaningful and query dependent information from large set of offline documents, data mining document clustering algorithm are adopted. Furthermore performance of the summary using OpenNLP tool and clustering techniques will be analysed and the optimal approach will be suggested.

Keywords:- K means algorithm, Document graph, Context sensitive text summarization.

I. INTRODUCTION

1. Overview

In today's information technology number of people is searching for informative data on web, but every time it is not possible that they could get all relevant data in single document, or on a single web page. They could get number of web pages as a search result. This problem has given the new solution that is associated to data mining which returns query specific information from large set of offline documents and represents as a single document to the user[1][2][3]. So it is cumbersome to extract the meaningful information from large data set to provide high quality summaries in order to allow the user to quickly locate the desired information. Summarization is the process of automatically creating a compressed version of a given text that provides useful information for the user. An application of document summarization is the snippets generated by web search engines for each query result. In this paper we represent the framework where user uploads the document files (currently supporting .txt and .doc files), after that the clusters of similar data will be formed. The clusters are query independent i.e. they are formed before the arrival of search query, and helpful result summaries are generated based on user's search query (query dependent)[8][9]. The paper consist of 5 sections where section II represents system description, section III represents future work, section IV represents the conclusion and finally section V represents the References.

Manuscript received on April 26, 2012.

Harshal J. Jain, Department of computer Engineering, Bharati Vidyapeeth College of Engineering, Pune, Maharashtra, India, 9850883090(e-mail: harshal.jain24@gmail.com)

Prof. M. S. Bewoor, Asst. Professor, Department of computer Engineering, Bharati Vidyapeeth College of Engineering, Pune, Maharashtra, India, 9850627767, (e-mail: msbewoor@bvucoep.edu.).

Dr. S. H. Patil, Head, Department of computer Engineering, Bharati Vidyapeeth College of Engineering, Pune, Maharashtra, India, 9850883090 (e-mail: shpatil@bvucoep.edu.).

1.1. Need

World Wide Web is a huge collection of data of different file formats. With the coming of the information revolution, electronic documents are becoming a principle media of business and academic information. In order to fully utilize these on-line documents effectively, it is crucial to be able to extract the valuable information of these documents. To serve this need we are having a text Summarization system. In order to generate a summary, we have to identify the most important pieces of information from the document, omitting irrelevant information and minimizing details, removing ambiguity in document and assembling them into a compact Coherent report. A particular algorithm is best suited for query dependent text document summarization. As every document we can convert into text, this strategy is much needful for the end users.

1.2. Goal

The goal is to use K means clustering algorithm for query dependent clustering of nodes in text document, and finding query dependent summary. The summary will be compared and best algorithm will be suggested for query dependent clustering using different clustering techniques. This technique will help end users to prepare query dependent summary of text documents in which they are interested.

-The proposed work will be mainly focused on summarization of text files (i.e. .txt) with NLP.

-The proposed work will be limited to clustering of text files of standard files related to the topic popular amongst researchers will be used.

-Only K means clustering algorithm method are considered for generating cluster based graph.

II. SYSTEM DESCRIPTION

Framework model of summarization system contains different stages shown in following diagram.

Explanation of figure 1 is given below

2.1 *Input*

User uploads the document files such as text or doc file. The input data is then spitted by the new line character and converted into the array of paragraphs. Every paragraph in the array is called as node. The syntactic relationship among every node is measured using traditional edge weight formula. This structure is called

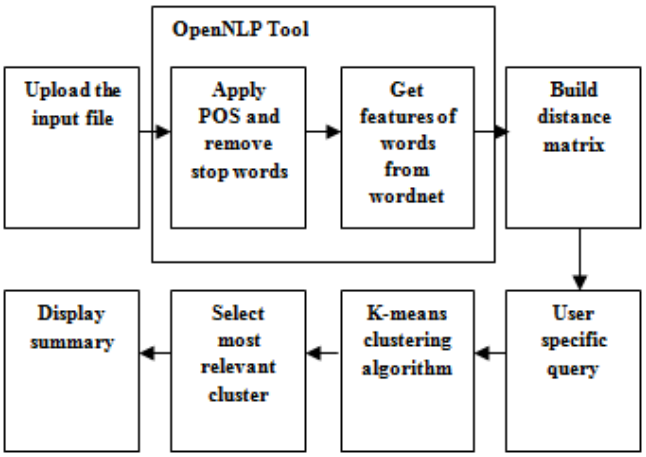

Fig 1: framework model of summarization system

as node. The syntactic relationship among every node is measured using traditional edge weight formula. This structure is called as document graph where every node is a paragraph and every edge between two nodes represents the association (similarity) between two nodes, to which the edge is connecting to. The document graph is input to the clustering algorithm, where effective clustering algorithm K-Means is used, which groups the nodes according to their edge weights and builds the clusters. Every node in the cluster maintains association with every other node. This association (edge weight) is greater than or equal to the cluster threshold value which is taken as input from user.

2.2 OpenNLP

OpenNLP library is a machine learning based toolkit for the processing of natural language text [10]. It supports the most common NLP (natural language processing) tasks, such as tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing, and coreference resolution. OpenNLP performs all the grammatical checking with the help of wordnet dictionary. These tasks are usually required to build more advanced text processing services.

OpenNLP performing the following tasks such as

Splitter- It find outs the sentences. It can detect that a punctuation character marks the end of a sentence or not. In this sense a sentence is defined as the longest white space trimmed character sequence between two punctuation marks. The first and last sentence makes an exception to this rule. The first non-whitespace character is assumed to be the begin of a sentence, and the last non whitespace character is assumed to be a sentence end. For example- The tulip flower grows in a variety of vibrant colours, shapes and sizes and is widely recognized throughout the world. According to the website springvalleys.com, tulips are the third most popular flower worldwide, ranking next to the rose and chrysanthemum. After splitting

The tulip flower grows in a variety of vibrant colours, shapes and sizes and is widely recognized throughout the world.

According to the website springvalleys.com, tulips are the third most popular flower worldwide, ranking next to the rose and chrysanthemum

Tokenizer- The OpenNLP Tokenizer segments an input character sequence into tokens. Tokens are usually words, punctuation, numbers, etc.

For example- The | tulip | flower | grows | in | a | variety | of | vibrant | colours |, | shapes | and | sizes | and | is | widely | recognized | throughout | the | world |.

POS tagger- The Part of Speech Tagger marks tokens with their corresponding word type based on the token itself and the context of the token. A token might have multiple pos tags depending on the token and the context. The OpenNLP POS Tagger uses a probability model to predict the correct pos tag out of the tag set.

Sample tag sets meaning -

DT singular determiner/quantifier (this, that), NN singular or mass noun, NNS plural noun, VBZ verb, 3rd. singular present, IN preposition

For example -The/DT tulip/NN flower/NN grows/VBZ in/IN a/DT variety/NN of/IN vibrant/JJ colours/NNS, /, shapes/NNS and/CC sizes/NNS and/CC is/VBZ widely/RB recognized/VBN throughout/IN the/DT world/NN. /.

Chunker - Text chunking consists of dividing a text in syntactically correlated parts of words, like noun groups, verb groups, but does not specify their internal structure, or their role in the main sentence.

For example-[NP The/DT tulip/NN flower/NN] [VP grows/VBZ] [PP in/IN] [NP a/DT variety/NN] [PP of/IN] [NP vibrant/JJ colours/NNS ,/, shapes/NNS and/CC sizes/NNS] and/CC [VP is/VBZ widely/RB recognized/VBN] [PP throughout/IN] [NP the/DT world/NN] ./.

Parser- it generate the parse tree for given sentence. Parser can read various forms of plain text input and can output various analyses formats, including part-of-speech tagged text, phrase structure trees, and a grammatical relations (typed dependency) format.

For example A rare black squirrel has become a regular visitor to suburban garden.

(INC (NP (DT A) (JJ rare) (JJ black) (NN squirrel)) (VBZ has) (VBN become) (NP (DT a) (JJ regular) (NN visitor)) (TO to) (NP (DT a) (JJ suburban) (NN garden)) (. .))

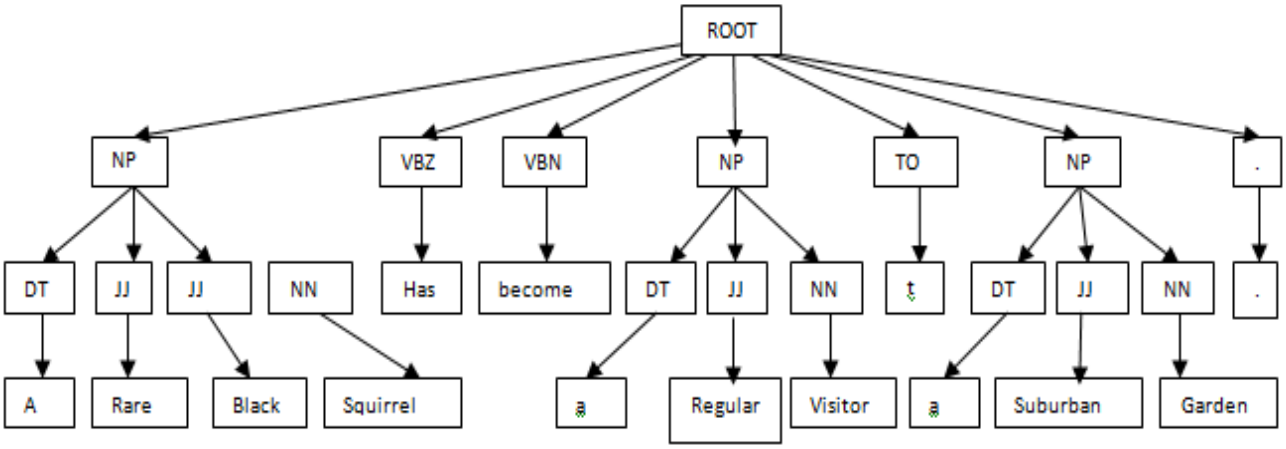

Fig 2: Parse tree for given example generated by Open NLP

Fig 2 showing the parse tree generated by openNLP parser

Name finder- The Name Finder can detect named entities and numbers in text.**For example** - The tulip <organization>flower</organization> grows in a <organization>variety</organization> of vibrant colours, shapes and sizes and is widely recognized throughout the world

2.3 Built Distance matrix: Lexical semantics begins with recognition that a word is a conventional association between a lexicalized concept and remark that plays a syntactic role. Here we have to find out the nature and organization of the lexicalized concepts that words can express, those dealing with the semantic structure of the English lexicon [11].

Word forms are imagined to be listed as headings for the columns; word meanings as headings for the rows. An entry in a cell of the matrix implies that the form in that column can be used (in an appropriate context) to express the meaning in that row.

Table 1: Sample distance matrix

Word meaning	**Word forms** W1 W2 W3 . . . . Wn
M1 M2 M3 . . Mm	E1.1 E1.2 E2.2 E3.3 . . Em,n

2.4 User specific query

We have distance matrix as an input to the clustering algorithm. Once the distance matrix gives relationship of different sentences is ready, user can fire a query. And query can be processed further.

2.5 K means clustering algorithm

After accepting a query from user, we apply K means clustering algorithm on the distance matrix to group cohesive sentences with each other according to degree of association in distance matrix. For each point, K means work like this; if the number of data is less than the number of cluster then we assign each data as the centroid of the cluster. Each centroid will have a cluster number. If the number of data is bigger than the number .Of cluster, for each data, we calculate the distance to all centroid and get the minimum distance. This data is said belong to the cluster that has minimum distance from this data. Since we are not sure about the location of the centroid, we need to adjust the centroid location based on the current updated data. Then we assign all the data to this new centroid. This process is repeated until no data is moving to another cluster anymore. By using these algorithms clusters will be formed according to the keywords in the user's query and related clusters will be formed.

2.6 selecting most relevant cluster

To select the most relevant cluster we have to build document clustering graph.

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 syntactic association among the nodes is measured and the documents graph G (V, E) is prepared [1]. The document graph G (V, E) of a document d is defined as follows:

D is split to a set of non-overlapping nodes t (v), v ϵ V.

An edge e (u, v) ϵ E is added between nodes u, v ϵ V if there is an association between t (u) and t (v) in d. 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, so they can be pre-computed. The following input parameters are required at the pre computation stage to create the document graph:

1. Threshold for edge weights:

Only edges with weight not below threshold will be created in the document graph. (A threshold is user configurable value that controls the formation of edges).Adding weighted edge is the next step after generating document graph. Here for each pair of nodes u, v we compute the association degree between them, that is, the score (weight) EScore (e) of the edge e (u, v). If Score (e) ≥ threshold, then e is added to E. The score of edge e (u, v) where nodes u, v has text fragments t (u), t (v) respectively is:

$$\text{EScore} = \frac{\sum ((tf(t(u),w) + tf(t(v),w)) \cdot idf(w))}{\text{size}(t(u)) + \text{size}(t(v))}$$

Where t f (d, w) is the number of occurrences of w in d, id f (w) is the inverse of the number of documents containing w, and size (d) is the size of the document (in words).That is, for every word w appearing in both text fragments we add a quantity equal to the tf.idf score of w. Notice that stop words are ignored. If EScore >= Threshold, the edge is added to the document graph

Table 2:

First_Node	Second_Node	Edge Weight
1	2	0.5
1	3	0.7
.	.	.
30	31	0.8

2.7 Query specific summery

Weights are assigned to each node of this sub-graph using a strategy similar to the Google Page ranking algorithm. And top five nodes are considered giving the best relevancy between statements according to user's query.

III. FUTURE WORK

In the future, we plan to extend our work to account for links between documents of the dataset. Also we will try to create the system for different application by using clustering algorithm. Finally, we plan to work on more elaborate techniques to split a document to text fragments and assign weights on the edge of the document graph.

IV. CONCLUSION

The prototype is very helpful for finding the effective query specific information in short span of time. The clustering algorithm plays an important role in result effectiveness and time consumption. Different clustering algorithms can be used on the same framework and their performance can be measured in term of quality of result and execution time.

REFERENCES

1. Ramakrishna Varadarajan, Vangelis Hristidis,"A System for Query-Specific Document Summarization"
2. Ravindranath Chowdary P Sreenivasa Kumar "An Incremental Summary Generation System"
3. Regina Barzilay and Michael Elhadad,"Using Lexical Chains for Text Summarization"
4. Mohamed Abdel Fattah, and Fuji Ren,"utomatic Text Summarization"
5. Jackie CK Cheung,"Comparing Abstractive and Extractive Summarization of Evaluative Text: Controversialist and Content Selection"
6. Jie Tang, Limin Yao, and Dewei Chen,"Multi-topic based Query-oriented Summarization"
7. R.M.Aliguliyev,"Automatic Document Summarization by Sentence Extraction"
8. Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In Proceedings of the ACL-04 Workshop: Text Summarization Branches Out, pages 74–81, Barcelona, Spain.
9. Luhn H. P. 1958, the automatic creation of literature abstracts, IBM Journal, pages 159-165
10. http://opennlp.apache.org/
11. L. Kaufman and P. J. Rousseeuw. Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons,