CONTEXT SENSITIVE TEXT SUMMARIZATION USING HIERARCHICAL CLUSTERING ALGORITHM

Ms. Meghana. N.Ingole, Mrs.M.S.Bewoor, Mr.S.H.Patil
Department of Post Graduate Computer Engineering,
Bharati vidyapeeth College of Engineering, Pune, India
meghna.ingole@yahoo.co.in, msbewoor@bvucoep.edu.

ABSTRACT

Text summarization addresses both the problem of selecting most important portions of text and the problem of generating coherent summaries. Information Retrieval plays an important role in searching on the Internet. Huge amount of data is present on the Web. There has been a great amount of work on query-independent summarization of documents. Most systems are limited to the query processing based on information retrieval system; the matching of the query against a set of text record is the core of the system. Retrieval of the relevant natural language text document without ambiguity is of more challenge. Today’s most search engines are based on keyword based techniques, which results in some disadvantages. We exploit NLP techniques to support a range of NL queries and snippets over an existing keyword-based search. This paper describes a simple system for choosing phrases from a document as key phrases. This phrase is made up of modules, splitting, tokenization; part-of-speech tagging; Chunking and parsing. While doing so we have also used new association algorithm to generate accurate distance matrix and then combine or divide existing group, creating hierarchical cluster of related data that reflects the order in which groups are merged or divided.

Here before clustering we are also handling text ambiguity to generate effective summary.

Keywords — NLP; Parsing; Tokenizing; Chunking; Similarity;

I. INTRODUCTION

1. Overview

Large amount of data is present on the Web. It will be easy for User to search for required information by putting particular keywords. As the number of documents available on users desktop and as the Internet access increases so does the need to provide high-quality summaries in order to allow the user to quickly locate the desired information. Text summarization is selecting important portion and generating summaries. 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 particular, First we check for word similarity by calling word net dictionary and we calculate weight for that, that is we find relation between statements using a newly introduced association algorithm and then structure is added to the documents in the pre-processing stage and converts them to a document graph where we will compare each sentence with all other sentences and calculate weight by calling word net dictionary and that is represented in the form of document graph.

Hierarchical clustering is used before summarization to generate effective summary. The main aim of this research work is to combine both approaches of Document clustering and query dependent summarization by removing ambiguity. This mainly includes applying different clustering algorithms on a text document. Create a weighted document graph of the resulting graph based on the keywords and obtain tree to get the summary of the document.

1.1 Need

In order to summarize full text effectively, it is crucial to extract relevant information from document. This need is fully satisfied by text summarization. After looking at the present scenario where Text segmentation based on hierarchical clustering algorithm which considers both Computational complexity and segmentation accuracy. The process of TSHAC consists of 4 steps. The first step is given text pre processing, tokenization, stop word removal and stemming are conducted to construct the vocabulary of the text. After text pre processing, the text can be represented as vectors, each vector represents a sentence within the text. A part of sentence (POS) similarities are then computed to construct the sentence-similarity matrix. Finally, the optimal topic boundaries are identified by the algorithm. Here we are implementing clustering approach that can be employed before summarization and if association is used before clustering then degree of relevancy of statements with each other will be more accurate. Thus also have to remove ambiguity depending on word similarity and semantics using Natural Language Processing. Before getting text summary, have to implement the algorithm, and from these performance of the algorithm can be measured.

II. RELATED WORK

To use agglomerative hierarchical 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.text) with NLP. The proposed work will be limited to clustering of text files of standard files as of now we are accepting only text. Agglomerative hierarchical method clustering are considered for generating cluster based graph. [2]This paper focuses on one approach, called shallow semantic analysis, where it accepts text, analyzes its semantic dependencies between phrases to extract key phrases. In this approach, semantic relations between phrases are usually acquired from a built thesaurus. Semantic and other statistical features are used to score the phrases where key phrase extraction is a fundamental research task in natural language processing and text mining. Standard performance evaluation metrics will be used to validate performance. [3] This paper presents a method to create query specific summaries by identifying the most query-relevant fragments and combining them using the semantic associations within the document. In particular, structure is added to the documents in the pre-processing stage and converted them to document graphs. Then, the best summaries are computed by calculating the top spanning trees on the document graphs. [4] In this paper an algorithm finds pair of sentences, one from the current summary and other from the new document that is to be swapped to improve the quality of the summary. This algorithm updates the available summary as and when a new document is made available to the system. It will replace the sentences from the present summary with the sentences from the new document. Before replacement, scoring model will check for the best pair of sentences (one from the present summary and another from the newly available document) to be swapped. The pair of
sentences that maximizes the score for summary will be chosen for swapping. The model is designed to consider both the importance of the sentence and its contribution to the summary. The method presented in this paper is obviously partial in that it only considers lexical chains as a source representation, and ignores any other clues that could be gathered from the text. Still, they have achieved results of a quality superior to that of summarizers usually employed in commercial systems such as search systems on the World Wide Web on the texts we investigated. The paper provides in depth explanation of implementation adopted for k-pragna, an agglomerative hierarchical clustering technique for categorical attributes. If there are n tuples in the dataset, then the similarity matrix can be computed in O (n^2). Before the identification of which pair of clusters has to be merged at the kth merge step, there are n-k rows of the similarity matrix with n+1-k clusters remaining. At the kth stage, the search involves n-k-1 comparisons. So, total number of comparisons for n-1 stages is O(n^2). At the kth stage, once the most similar pair has been identified, we need to update the n-k matrix entries corresponding to the cluster resulting from the merger. For n-1 stages, the number of comparisons and the updates is again of the order of O(n^2). In a typical case it would be expected that one row would be updated at each stage resulting in O (n^2). But since the loop executes for n-k times so we have the complexity as O (n^3), the hierarchical clustering algorithm have complexity this is O (n^3).

III. SYSTEM DESCRIPTION

Explanation of each of them is given in the following section:

1. Input:
First of all, original document containing set of statements is fed to the system. Generally document from which summary is to be retrieved should present relevant sentences probably from i.e. same context. Given a document related to some context further it is given to second phase.

2. OPENNLP
2.1. Split Sentences
Before implementing these modules, the input snippets should be split in to separate sentences using the Split method on this input. We can do this by treating each of the characters ‘.’, ‘!’ ‘?’ as separator rather than definite end-of-sentence markers. This split method is mainly used to detect the end of sentences.

Example:
Maharashtra is the world’s second most populous first-level administrative country sub-division. Marathi is the official language.
2.2. Tokenization:
Tokenization process is used to determine sentence boundaries, and to separate the text into a stream of individual tokens (words) by removing extraneous punctuation. It separates the text into words by using spaces, line breaks, and other word terminators in the English language. For example, punctuation marks such as ? and : are word-breaking characters.
Example:
Maharashtra | is | the | world | 's | second | most | populous | first-level | administrative | country | sub-division |. Marathi | is | the | official | language |.

2.3. Part of Speech Tagging
In POS tagging, we are classifying word under English Grammar Phrases.
CC- coordinating conjunction
CD- cardinal number
IS- existential there
DT- Determiner
NN- Noun
JJ- Adjective
DD- Common determinant
and many more.
Pre-processing of the text is needed as follows:

a) Break the text:
Apply part of speech tagging to the words in the text. This is essential to pick the correct meaning of the word in Word Net. Hence if the word “shine” is used in the text as a verb, we will not associate it with a form of clothing.
b) Identify collocations in the text:
A collocation is a group of commonly co-occurring words, for example. Identifying collocations helps in capturing the meaning of the text better than that of the individual words (just like any idiom).
c) Remove stop words:
The word like “the”, “him” and “had” which do not contribute to understanding the main ideas present in the text. The sequence of the above operations is important since we must identify collocations before removing stop words as many stop words often form part of collocations.
Example:
Maharashtra/NNP is/VBZ the/DT world/NN 's/POS second/JJ most/RBS populous/JJ first-level/JJ administrative/JJ country/NN sub-division. Marathi/NN is/VBZ the/DT official/JJ language/NN ./.

2.4. Chunking and Parsing:
It is mainly used to find phrases. It could be useful when looking for units of meaning in a sentence larger than individual words. Phrase chunking is the task of segmenting a text into information units larger than a word and possibly comprising other chunks, e.g. ‘the black board’ forms a noun phrase. Chunking recognizes the boundaries of noun phrases based on POS. Chunking is the next step on the way to full parsing. Chunks are non-overlapping groups of words forming small syntactic units (phrases) such as noun phrases consisting of an optional determiner, followed by an optional adjective, followed by a noun. Text chunking divides the input text into such phrases and assigns a type such as NP for noun phrase, VP for verb phrase, PP for prepositional phrase in the following example, where the chunk borders are indicated by square brackets:
Example:
[NP Maharashtra/NNP ] [VP is/VBZ ] [NP the/DT world/NN ] [NP 's/POS second/JJ most/RBS populous/JJ first-level/JJ administrative/JJ country/NN sub-division. Marathi/NN ] [VP is/VBZ ] [NP the/DT official/JJ language/NN ] ./.
To perform the chunking task, a POS tagged set of tokens is required.
For e.g.:

![Diagram of a parse tree]

2.5. Similarity:
- Finding similarity between words is a fundamental part of text similarity.
- Words can be similar if:
  - They mean the same thing (synonyms)
  - They mean the opposite (antonyms)
  - They are used in the same way (red, green)
  - They are used in the same context.
  - One is a type of another.

Example:
Maharashtra/NNP is/VBZ the/DT world/NN’s/POS second/JJ most/RBS populous/JJ first-level/JJ administrative/JJ country/NN sub-division/NN./.
0.508442625722466 default,ww=is, Maharashtra

Marathi/NNP is/VBZ the/DT official/JJ language/NN./.
0.508442625722466 default,ww=is, Marathi

2.6. Built Distance matrix:
Lexical semantics begins with recognition that a word is a conventional association between a lexicalized concept and an utterance that plays a syntactic role.

2.7. User fires a query:
We have distance matrix as an input to the clustering algorithm. Once the distance matrix giving associativity of different sentences is ready, user can fire a query. There has been a great amount of work on query-independent summarization of documents. However, due to the success of Web search engines query-specific document summarization has become an important problem, which has received little attention.

2.8. Clustering
Hierarchical Clustering
Given a set of N items to be clustered, and an N*N distance (or similarity) matrix, the basic process of hierarchical clustering is this:
Start by assigning each item to a cluster, so that if you have N items, you now have N clusters, each containing just one item. Let the distances (similarities) between the clusters the same as the distances (similarities) between the items they contain.
Find the closest (most similar) pair of clusters and merge them into a single cluster, so that now you have one cluster less.
Compute distances (similarities) between the new cluster and each of the old clusters.
Repeat steps 2 and 3 until all items are clustered into a single cluster of size N. (*)
Example for Hierarchical Clustering

<table>
<thead>
<tr>
<th>First Node</th>
<th>Second Node</th>
<th>Max Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>22</td>
<td>2.5</td>
</tr>
<tr>
<td>2</td>
<td>19</td>
<td>1.2</td>
</tr>
<tr>
<td>3</td>
<td>31</td>
<td>3.5</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>31</td>
<td>12</td>
<td>2.7</td>
</tr>
</tbody>
</table>

Table 1.1 Nodes and maximum weight

1) Every node is considered as a cluster.
2) Start by the first pair of nodes in the above table and add in the first cluster because they have maximum weight. Here Node 1 and 22 are closely related hence added to the first cluster. So Cluster_1 contains 2 nodes 1 and 22. Cluster_1:- 1, 22
3) The newly formed cluster Cluster_1 is compared with all other clusters (Nodes are clusters). Here the Cluster_1 and every other cluster are sent to the relation manager and maximum weight is calculated. The cluster which is having maximum weight greater than or equal to Cluster Threshold (Specified by user) is added into the first cluster Cluster_1. If the value is less than Cluster Threshold Value then that cluster is kept as it is.
4) The step 3 is repeated till all the clusters are processed.

<table>
<thead>
<tr>
<th>Cluster Number</th>
<th>Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster_1</td>
<td>1,22,13,25</td>
</tr>
<tr>
<td>Cluster_2</td>
<td>2,9,13,20</td>
</tr>
</tbody>
</table>

Table 1.2 Cluster number and nodes in cluster

2.9. Build Document Graph

The document graph can be build by considering each sentence as a single node. Complete text will be classified into nodes. Then the distance between each node with other nodes can be calculated and at the same time word net dictionary will be called. From their weight can be generated. Depending upon weight, sentence similarity can be found out. Sentence with maximum weight can be considered in summarization.

V Result

As in Figure 3, it shows Output screen of uploaded text file, processed file, document graph. The text file can be uploaded, and through parse input file, it will call word net for similarity. Fig No.4. shows performance analysis of Hierarchical and Expectation Maximization depending on Computations cycles and Space complexity. Fig. No. 5 shows summarization result where it is showing match found with sentences.
Fig No. 3 Output screen of uploaded text file, processed file, document graph

Fig No. 4 Performance analysis of Hierarchical Clustering Algorithm and Expectation Maximization Algorithm

(1) Number Of computation Cycles – Hierarchical-72, Expectation Maximization-93

(2) Space Complexity – Hierarchical-1968, Expectation Maximization-1925
VI. CONCLUSION
In this work we presented a structure-based technique to create query-specific summaries for text documents. In particular, we first create the document graph of a document to represent the hidden semantic structure of the document and then perform keyword proximity search on this graph. We show with a user survey that our approach performs better than other state of the art approaches. Furthermore, we show the feasibility of our approach with a performance evaluation.

VII. FUTURE WORK
In the future, we plan to extend our work to account for links between documents of the dataset. For example, exploit hyperlinks in providing summarization on the Web. Furthermore, we are investigating how the document graph can be used to rank documents with respect to keyword queries. 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. We are also investigating how the different phases of tokenization can be done in single technique.

VIII. REFERENCES
1. “An Efficient Linear Text Segmentation Algorithm Using Hierarchical Agglomerative Clustering”, 2011 IEEE Seventh International Conference on Computational Intelligence and Security
2. “Key phrase Extraction Based on Semantic Relatedness”, 9th IEEE Int. Conf. on Cognitive Informatics (ICCI’10)
3. Ramakrishna Varadarajan, Vangelis Hristidis, ,”A System for Query-Specific Document Summarization”
4. C. Ravindranath Chowdary P Sreenivasa Kumar “An Incremental Summary Generation System”
5. Regina Barzilay and Michael Elhadad ,”Using Lexical Chains for Text Summarization”
6. Parul Agarwal, M. Afshar Alam, Ranjit Biswas ,”Analysing the agglomerative hierarchical Clustering Algorithm for Categorical Attributes “