Text Summarization and Discovery of Frames and Relationship from Natural Language Text - A R&D Methodology

P.Chakrabarti[1] , J.K. Basu[2]
[1]Sir Padampat Singhania University, Udaipur-313601, Rajasthan, India
[2]Heritage Institute of Technology, Kokata-700107, West Bengal, India

Abstract
The paper deals with the concept of data mining whereby the data resources can be fetched and accessed accordingly with reduced time complexity. Resource sharing is an important aspect in the field of information science. The retrieval techniques are pointed out based on the ideas of binary search tree, Gantt chart, text summarization. A theorem has been cited regarding the summation of total length of codes of each leaf search term. Summarization is a hard problem of Natural Language Processing because, to do it properly, one has to really understand the point of a text. This requires semantic analysis, discourse processing, and inferential interpretation (grouping of the content using world knowledge). The last step, especially, is complex, because systems without a great deal of world knowledge simply cannot do it. Therefore, attempts so far of performing true abstraction--creating abstracts as summaries--have not been very successful. Fortunately, however, an approximation called extraction is more feasible today. To create an extract, a system need simply to identify the most important/topical/central topic(s) of the text, and return them to the reader. Although the summary is not necessarily coherent, the reader can form an opinion of the content of the original. Most automated summarization systems today produce extracts only. Another purpose of this paper is to addresses the problem of information discovery in large collections of text. For users, one of the key problems in working with such collections is determining where to focus their attention. Text documents often contain valuable structured data that is hidden in regular English sentences. This data is best exploited if available as a relational table that we could use for answering precise queries or for running data mining tasks. We explore a technique for extracting such tables from document collections that requires only a handful of training examples from users. In this paper we have tried to explain how to extract the different kind of relationship between the words with the help of a frame net analysis diagram of an annotation layer software.

Keywords-- *data mining , time complexity, binary search tree , Gantt chart, text summarization*

I. INTRODUCTION

Accessing information that is resources from heterogeneous data should be done in an optimum way[1,2]. The search tree can be applied for effective search. The average waiting time for successful transaction of data can easily be analyzed with the help of Gantt chart whereby we denote search transaction for an user as a process. Sometimes in case of web mining of resources, the context of text summarization is done where the search is based on some selected portion of text. Herein lies the importance of text summarization which is based on centroid-based algorithm.

II. MINING OF RESOURCES

A search can be formed based on the initial search term and its gradual sub term while the process of matching[3,4]. Thereby the level is increased, in initial search term is the root and the final term fully matching with the context of the users' desire is a leaf node. For further use if the library administrator save the time then he can save each tree in a database and denote each search term by a binary code.

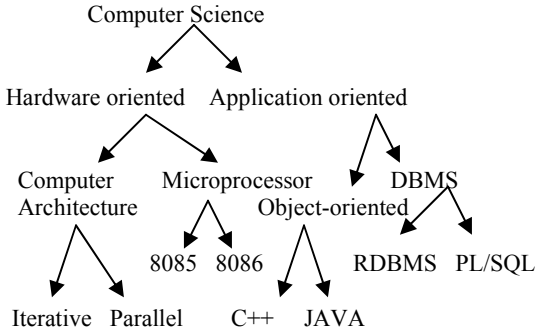


Fig1 : Binary search tree

In the above figure, computer science is the root that is initial search term. If a user wants to access library resources PL/SQL, then the database hierarchy, PL/SQL is the node in level 3 and it is a leaf node. For future purpose if the library administrator saves the model in a database and identify each search term as a binary code, then by giving the code number he can analyze the position of data in the model and acknowledge quickly as per users' request. The concept of coding is as follows:

Value = 0 if the search term is a left child of parent node

= 1 otherwise

Theorem: In the process of coding, $\sum_{i=1}^{N} 1/2^{L_i} = 1$,

where L_i is the length of code of ith leaf node in the tree, N is total number of leaf nodes and $1<i<N$.

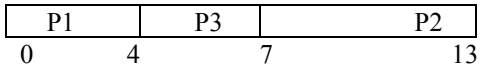
Proof:

In fig. 1 codes of leaf nodes are as follows:
Hieratic Architecture :000
Parallel Architecture : 001
8085 model : 010
8086 model : 011
C++ : 100
JAVA : 101
RDBMS: 110
PL/SQL " 111
So, N=8. Each leaf node has identical code length i.e. 3.
Therefore, $1/2^{L_i} = 1/2 = 1/8, 1/2 = 1/8, \ldots 1/2 = 1/8$

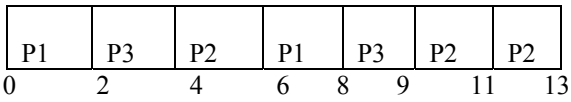
III. MINING OF THE LIBRARY RESOURCES BASED ON GANTT CHART

Let R be a resource and the users are A, B, and C. Now, for transaction of R , each of A, B and C send request to the library administrator. According based on the priority, the search schedule is performed.
Let, P1 = Process of search by user A
P2 = Process of search by user B
P3 = Process of search by user C

Let, tP1 = time for A to search successfully = 4 seconds
tP2 = time for B to search successfully = 6 seconds
tP3 = time for C to search successfully = 3 seconds
If priority of P1>P3>P2, then Gantt chart is s follows:



Waiting time of P1 = 0, waiting time of P2 = 7 seconds and waiting time of P3 = 4 seconds.
Sometimes the concept of Round Robin Scheduling is applied whereby a time slice is given and after that the process is switched to another user irrespective of completion time of search. Let, time slice = 2 seconds, then the Gantt chart is as follows:



Hence, after 9th seconds , two successive search engines are performed by user B as the other users A and C have already fetched their information successfully.

IV. MINING OF THE DATA RESOURCES BASED ON CENTROID BASED TEXT SUMMARIZATION

The mining technique is based on Centroid-based algorithm[5] which is as follows :
Input**:** A collection of related documents.
Output**:** A summary.

Steps to summarize :

A.. Finding Cluster Centroid
A cluster consisting of total number of sentences from all input documents is formed. The 'count' value for each word indicating the average number of occurrences of a word across the entire cluster is found out. Then the centroid value for each term is calculated as:

Count * idf(w)=count(w) * (log(DN / df(w)))

where df(w)=document frequency for each word.
DN=number of documents in the corpus.

B. Finding Sentence Position Score
The score of ith sentence **(Si)** is computed
Pscore*(Si)*= max(1/i , 1/(n-i-1))
where i=sentence number
n=number of sentences

C. Finding Sentence Length Score
The length here means the number of characters in the sentence. A sentence shorter than a certain length gets penalty. The length score of a sentence can be calculated as
Lscore*(Si)*= 0 if Li≤ Lmin
= (Li-Lmin)/Li otherwise
where Li=length of each sentence

Lmin=20 , i.e. sentence with 20 or fewer characters receives penalty.

D. Finding Headline Score

The idea is that greater the number of words in a sentence that match those in the headline , the more important the sentence is likely to be. The headline score can be calculated as

$$Hscore(Si) = t / N$$

where t=number of words in the sentence that match with the words in the headline

N= number of words in the sentence

E. Compute Sentence Score

$$SCORE(S) = \sum (wc.Ci + wp.Pi + wf.Fi + wl.Li)$$

where i ranges from 1 to n as $(1 \le i \le n)$

Also, Ci=Centroid value of the sentence
Pi=sentence position score
Fi=headline score
Li=sentence length score
Wc= wI = wf = wl =1
n = number of sentences in the cluster

F. Extract Sentences

Sentences are sorted according to descending order. Select d out of n sentences as an intermediate summary of the input documents. The sentences are extracted in an order.

$$d = r * n$$

where r = Compression Rate
and n = total number of sentences taken from input documents.

V. FRAME DEVELOPMENT AND FRAME NET ANNOTATION

A lexical unit (LU) is a pairing of word with a meaning. Typically each sense of polysemous word belongs to a different semantic frame, a script like conceptual structure that describes a particular type of situation, object, or event along with its participants and props. For example the Apply_heat frame describes a common situation involving a COOK, some FOOD and a HEATING_INSTRUMENT, and is evoked by words such as *bake, blanch, boil, broil, brown, simmer, stream* etc. We call these roles frame elements (FEs) and the frame evoking words are LUs in the Apply_heat frame. Some frames are more abstract, such as change_position_on_a_scale, which is evoked by LUs such as decline, decrease, gain, plummet, rise, etc. and has FEs such as ITEM, ATTRIBUTE, INITIAL_VALUE and FINAL_VALUE.

In the simplest case, the frame-evoking LU is a verb and the FEs are its syntactic dependents:

(i) (*Cook*) Mathew fried (*food*) the fish (*Heating_Instrument*) in a heavy iron skillet.

(ii) (*Item*) ITC stock rose (*Difference*) $4 (*Final_value*) to $40.

However, event noun such as *reduction* in the cause_change_of_scalar_position frame also evoke frames:

….the reduction **(***item*) of debt levels (*value_2*) to $500 million (*value_1*) from $2.5 billion

or objectives such as asleep in the Sleep frame:

(*Sleeper*) They were asleep (*Duration*) for hours.

The lexical entries for a predicting word, derived from such annotations, identifies the frame which underlies a given meaning and specifies the ways in which FEs are realized in structures headed by the word. Framenet annotations derive from two sources. In pursuing the goal of recording the range of semantic and syntactic combinatory possibilities of each word in which of it senses, we normally concentrate on a particular target LU and extract sentences from the different texts containing that LU. In another kind of work that represents a much smaller percentage of our overall annotations, we annotate running text. Full text annotation differs from sentence annotation mostly in that the sentences are chosen for us, so to speak, by the author of the text. The annotation of running text is also technically possible. Frame net lexicographers can one by one declare each word in a sentence of target, select a frame relative to which the new target is to be annotated, get a new set of annotation layers (frame element, grammatical function, phrase type) and appropriate frame element tags, and then annotate the relevant constituents. The core of the process has always been looking at attestations of a group of words that we believe to have some semantic overlap, and dividing these attestations into groups. Afterward we combine the small groups into large enough groupings to make reasonable frames at which point we may (equivalently) call the words targets, lexical units, or frame-evoking elements. In the past the criteria of such grouping have been informal and intuitive, but recently, the criteria have become more explicit. The basic semantic type for a frame element ought to be broadly constant across uses. If that is not so it suggests the need to posit distinct frame elements. In some cases, however, we still want to recognize a relationship between frame elements whose syntactic form suggests that they refer to ontologically different kinds of entities. For example,

in ***I want*** [to win] compared with ***I want*** [an orange], both complements of the verb "want" have something to do with the desiring frame, but each of the complements directly refers to something rather different.

As a technical matter, the way in which FrameNet analyzes instances of a target predicate consists of marking up parallel aligned layers of annotation with appropriate label sets. The number of layers and the kind of information that can be recorded on them is technically unlimited. But in FrameNet's current practice the four core annotation layers are the Target, frame element (FE), grammatical function (GF), and phrase type (PT) layers. On the first, the(parts of the) target predicate are marked while on the latter three, labels are applied to the constituents expressing the frame elements of the target. The next-most important set of layers consists of the layers called other; a layer called either Noun, Verb, Adj, or Prep depending on the part of speech of the target (this layer is also often called the part-of-speech-specific layer); and the Sent(sentence) layer. A final group of layers includes, among others, layers holding labels related to part of speech (POS) and Named Entity Recognition (NER). Generally FrameNet is the collection of frames consisting different types of frame elements like Noun phrase (NP), Verb phrase (VP), Adverb phrase (ADP), Adjective phrase (AJP), Preposition phrase (PP) and FrameNet analyzes the instances of different layer of annotation like Frame Elements (FE), Grammatical Functions (GF) and Phrase Types (PT).

VI. EXPERIMENT METHODOLOGY

We used the parser of Collins (1997)[6], a statistical parser trained on examples from the Penn Treebank, to generate parses of the same format for the sentences in our data. Phrase types were derived automatically from parse trees generated by the parser. Given the automatically generated parse tree, the constituent spanning

the same set of words as each annotated frame element was found, and the constituent's

nonterminal label was taken as the phrase type. In cases where more than one constituent matches due to a unary production in the parse tree, the higher constituent was chosen.

The matching was performed by calculating the starting and ending word positions for each constituent in the parse tree, as well as for each annotated frame element, and matching each frame element with the parse constituent with the same beginning and ending points. Punctuation was ignored in this computation. Due to parsing errors, or, less frequently, mismatches between the parse tree formalism and the FrameNet annotation standards, there was sometimes no parse constituent matching an annotated frame element. This occurred for 13% of the frame elements in the training set. The one case of systematic mismatch between the parse tree formalism and the FrameNet annotation standards is the FrameNet convention of including both a relative pronoun and

its antecedent in frame elements. Mismatch caused by the treatment of relative pronouns accounts for 1% of the frame elements in the training set. During testing, the largest constituent beginning at the frame element's left boundary and lying entirely within the element was used to calculate the features. We did not use this technique on the training set, as we expected that it would add noise to the data, but instead discarded examples with no matching parse constituent. Our technique for finding a near match handles common parse errors such as a prepositional phrase being incorrectly attached to a noun phrase at the right-hand edge, and it guarantees that some syntactic category will be returned: the part-of-speech tag of the frame element's first word in the limiting case.

Algorithm used for extracting new tupples using a set of patterns

```
GenerateTuples(Patterns)
For each text segment
(1) {<o,l>,< ls,t1,ms,t,rs >}=
                    CreateOccurrence(text_segment);
        TC = <o,l>;
        SimBest = 0;
        For each p in Patterns
(2) sim = Match(<ls,t1,ms,t2,rs >,p);
        if (sim>=Tsim)
(3) UpdatePatternSelectivity(p,TC);
        if(sim>=SimBest)
                SimBest=sim;
                PBest=p;
        if(SimBest>=Tsim)
        CandidateTuples[TC].Patterns[PBest]= SimBest;
return CandidateTuples;
```

VII. RESULTS

Results on identifying frame elements (FEs), including partial matches. A total of 7,681 constituents were identified as FEs, and 8167 FEs were present in hand annotations, of which matching parse constituents were present for 7,053(86%).

Type of Overlap	*Identified Constituents*	*Number*
Exactly matching boundaries	63%	5326
Identified constituent entirely within true frame element	8	653
True frame element entirely within identified constituent	7	593
Both partially within the other	0	23
No overlap with any true frame element	13	997

Table 1: Matching of overlaps

When the automatically identified constituents were fed through the role labeling system described above, 79.6% of the constituents that had been correctly identified in the first stage were assigned the correct role in the second, roughly equivalent to the performance when assigning roles to constituents identified by hand.

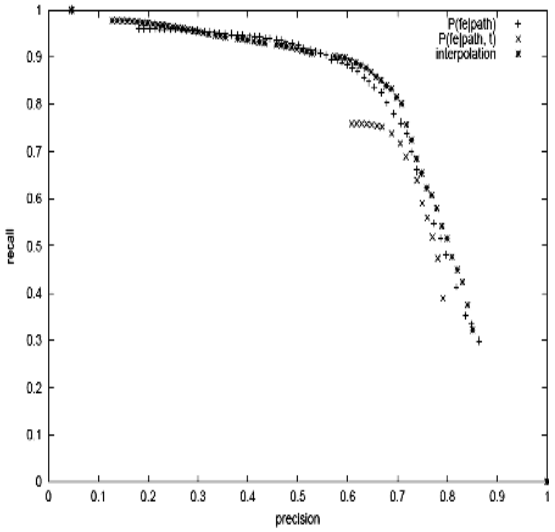


Fig 2: Precision/recall is plotted for various methods of identifying frame elements.

VIII. CONCLUSION

In the paper we have observed that efficient ways of optimum data mining reduces time complexity. In case of text summarization. Researches are going on this topic. There are many other techniques related to text summarization based on position of sentences or length of sentences of the documents. It will be more reliable if the sentences are parsed in phrase level using Link Grammar parser. For each sentence with the content of the sentence there should be associated the information of the words of the sentence. The information of the word means 'subject', 'time', 'space/ location', 'action i.e. verb' etc. Using these information the sentences are clustered on the basis of same 'subject' or 'action' etc. These clusters are ranked on the basis of size. The clusters are extracted from top order until required summary length is achieved. Experiments are also going on other several features of sentences. The results of this experiment are very encouraging. They demonstrate that the technique employed is capable of identifying numerous associations of interest in large text collections in a completely automated fashion. The approach could be used to provide a rapid overview of large volumes of text. In a dynamic collection environment it could be used to automatically alert users when text has been acquired that contains information of interest.

The approach described here has a number of important characteristics that make it particularly well-suited for intelligence analysis applications:

(1) It is independent of topic, genre, or language.

(2) Identified relationships represent the aggregate implications of high-order associations. For this reason, the approach is capable of identifying quite subtle relationships.

(3)The user has complete flexibility in specifying items of interest. Any point in the LSI representation space can be used as a basis for defining a row or column in the comparison matrices. In particular, a textual description of an entity or concept of interest may be chosen as a definition of an item of interest. That description does not have to have been derived from the documents indexed. There need only be some minimal degree of overlap in terminology between the description and the aggregate terminology employed in the documents used to generate the representation space.

(4)The technique has significant utility in identifying possible use of aliases.

(5)There is no need for predefined auxiliary structures such as taxonomies or ontologies. Nor is there need for any linguistic analysis, other than that contained in the entity extraction software

REFERENCES

[1] Daniel N ; Radev D, and Allison T (2003) - Sub-event based multidocument summarization. In HLT-NAACL Workshop on Text Summarization, Edmonton, AB, Canada

[2] Grewal A ; Allison T ; Dimitrov S and Radev D (2003) - Multi-document summarization using o_ the shelf compression software. In HLT-NAACL Workshop on Text Summarization, Edmonton, AB, Canada

[3] Kareem O and Radev D (2004) - Hierarchical text summarization for WAP-enabled mobile devices. Submitted to SIGIR 2004 Demo Session

[4] Otterbacher J and Radev D (2004) - A resource for revision-based multi-document summarization and evaluation. In LREC, Lisbon, Portugal

[5] Radev D; Blair-Goldensohn S, and Zhang Z (2001) - Experiments in single and multi-document summarization using M EAD. In First Document Understanding Conference, New Orleans, LA

[6] Collins, Michael. 1997. Three generative, lexicalised models for statistical parsing. In Proceedings of the 35th Annual Meeting of the ACL, pages 16–23, Madrid, Spain.