Improvisation of the Document Summarization by combining the IR techniques with “Code-Quantity and Attention” Linguistic Principles

Pranitha Reddy, R C Balabantaray

Abstract—In this paper we have given a new statistical approach for automatic text summarization by combining the Information Retrieval (IR) techniques with the linguistic principles code quantity, memory and attention to get the relevant sentences. In our approach we are distilling the most important information from the source to get the actual concept in the abridged form. We have initially removed the redundancy of the input document by using the Synonymous Cosine Similarity and we ranked the sentences based on the linguistic principles code quantity, memory and attention. Moreover, this method has been run over the test data, obtaining satisfactory results in the evaluation when compared with the MS Word Automatic Summarizer with respect to the human judgment.

Index Terms—Summarization, Code Quantity, Memory, Attention, Synonymous Cosine Similarity.

I. INTRODUCTION

Summarization, according to [2], can be defined as a reductive transformation of source text through content condensation by selection and/or generalization of what is important in the source. Text summarization is a challenging problem these days due to the great amount of information we are provided with and thanks to the development of Internet technologies, needs of producing summaries have become more and more widespread. Summarization is a very interesting and useful task that gives support to many other tasks as well as it takes advantage of the techniques developed for related Natural Language Processing tasks. According to [3], text summarization process involves three stages: topic identification, interpretation and summary generation. To identify the topic in a document, what systems usually do is to assign a score to each unit of input (word, sentence, passage) by means of statistical or machine learning methods. The stage of interpretation distinguishes extract type summarization systems from abstract type systems. During interpretation, the topics identified as important are fused, represented in new terms, and expressed using a new formulation, using concepts or words not found in the original text. Finally, when the summary content has been created through abstracting and/or information extraction, it requires techniques of natural language generation (NLG) to build the summary sentences. When an extractive approach is taken, there is no generation stage involved. The text summarization methods can be broadly classified into extractive and abstractive summarization methods. An extractive summarization method consists of selecting important sentences, paragraphs etc. from the original document and concatenating them into shorter form. The importance of sentences is decided based on statistical and linguistic features of sentences. The abstractive summarization method proposed in [9, 10] attempts to develop an understanding of the main concepts in a document and then expresses those concepts in clear natural language. It uses linguistic methods to examine and interpret the text and then to find the new concepts and expressions to best describe it by generating a new shorter text that conveys the most important information from the original text document. In this paper we focus on novel techniques which are based on extractive text summarization methods. The rest of the paper is organized as follows. Section 2 presents the related work, Section 3 presents our recent work. In Section 4, our methodology and the proposed algorithm are presented. The results are discussed in Section 5. Section 6 concludes the paper and mentions future work.

II. RELATED WORK

In this section, we primarily aim to investigate the empirical methods that have been used to build summarization systems. Most early work on single-document summarization focused on technical documents. In the work proposed in [3] that the frequency of a particular word in an article provides a useful measure of its significance. There are several key ideas put forward in this paper that have assumed importance in later work on summarization. As a first step, words were stemmed to their root forms, and stop words were deleted. In [3], a list of content words sorted by decreasing frequency is compiled thus indexing a significance measure of the word. On a sentence level, a significance factor was derived that rejects the number of occurrences of significant words within a sentence, and the linear distance between them due to the intervention of non-significant words. All sentences are ranked in order of their significance factor, and the top ranking sentences are finally selected to form the auto-abstract. Related work proposed in [4], also done at IBM and published in the same journal, provides early insight on a particular feature helpful in finding salient parts of documents: the sentence position. Towards this goal, the author examined 200 paragraphs to find that in 85% of the paragraphs the topic sentence came as the first one and in 7% of the time it was the last sentence. Thus, a naive but fairly accurate way to select a topic sentence would be to choose one of these two. This positional
feature has since been used in many complex machine learning based systems. The approach proposed in [5] describes a system that produces document extracts. The primary contribution was the development of a typical structure for an extractive summarization experiment. At first, a protocol was proposed for creating manual extracts that was applied in a set of 400 technical documents. The two features of word frequency and positional importance were incorporated from the previous two works. Two other features were used: the presence of cue words (presence of words like significant, or hardly), and the skeleton of the document (whether the sentence is a title or heading). Weights were attached to each of these features manually to score each sentence. During evaluation, it was found that about 44% of the auto-extracts matched the manual extracts. The Trainable Document Summarizer proposed in [7], performs sentence extracting task, based on a number of weighting heuristics. Following features were used and evaluated:

1. Sentence Length Cut-O Feature: sentences containing less than a pre-specified number of words are not included in the abstract.
2. Fixed-Phrase Feature: sentences containing certain cue words and phrases are included.
3. Paragraph Feature: this is basically equivalent to Location Method feature proposed in [8].
4. Thematic Word Feature: the most frequent words are defined as thematic words. Sentence scores are functions of the thematic words’ frequencies.
5. Uppercase Word Feature: upper-case words (with certain obvious exceptions) are treated as thematic words, as well.

The ANES text extraction system proposed in [8], is a system that performs automatic, domain-independent condensation of news data. The process of summary generation has four major constituents:

1. Corpus analysis: this is mainly a calculation of the tf*idf-weights for all terms.
2. Statistical selection of signature words: terms with a high tf*idf-weight plus headline-words.
3. Sentence weighting: summing over all signature word weights, modifying the weights by some other factors, such as relative location.
4. Sentence selection: Selecting high scored sentences.

In [13] Radev et al. proposed attacks automated summarization problem using information retrieval techniques. Radev et al. uses vector space model and clustering to find the central and salient sentences. They are using weighted vectors of (tf*idf) values to represent sentences. tf is term frequency and idf is inverse document frequency. idf is the frequency of the word in all documents in the corpus. Note that this approach depends on word frequencies, so they are only taking advantage of word repetition. Word repetition is one of the lexical cohesion types.

In [11] Erkan improved the performance of the summarizer by introducing a Google’s Page rank algorithm which was proposed in [14] for the selection procedure. This summarization system is a part of MEAD summarization toolkit proposed in [15] and is an important algorithm in automated summarization research literature. Lexical chains are structures for modelling lexical cohesion computationally. Lexical chains are sets of related words. Halliday in [16] presents one of the first work on lexical cohesion. Morris and Hirst [17] discuss an algorithm for building lexical chains. St.Onge et al. in [18] presents the first algorithm where, lexical chains are built using WordNet. They used lexical chains to detect and correct malapropisms. In [19] Barzilay presented her lexical chaining algorithm and used lexical chains to extract summaries. Barzilay’s algorithm has achieved good results in evaluations. Usually, in algorithms using lexical chains, text units that are traversed by the strongest lexical chains are selected.

III. OUR RECENT WORK

In this paper we use the extractive method to get the summary of the input document. In order to extract the summary, we use the following features:

Content (Key) words: - After removing the stop words the remaining words are treated as key words. We have taken the total number of key words when assigning the weight to each term.

Code quantity and Attention: - The important sentences contain more coding material (no. of noun phrases) which, refers to code quantity. The attention of the user will be more on the sentences which are bold, italic or underlined, which refers to attention. Based on this principle weight’s are assigned to each sentence and are ranked accordingly to extract the summary.

Sentence location feature: - Usually first sentence of a text document are more important and are having greater chances to be included in summary. So in our case we have made the inclusion of first sentence of the document mandatory in the summary.

IV. METHODOLOGY

Our summarizer takes input in “.txt” or “.rtf” format. Firstly it tokenizes the text in order to find the individual tokens or terms. Then we are filtering the text by removing the stop words. After removing the stop words we are extracting the Parts Of Speech of each word by connecting to the Word Net dictionary using JWNLP which will be used in our process of retrieving summary efficiently. We are using cosine similarity to reduce the redundancy in the input document which is calculated as follows,

\[ \text{similarity} = \cos \theta = \frac{\mathbf{A} \cdot \mathbf{B}}{||\mathbf{A}|| \cdot ||\mathbf{B}||} \]  

(1)

Where \( \mathbf{A} \) and \( \mathbf{B} \) are the term frequency vectors of two sentences in the document. For cosine similarity we take term frequency vectors (tf*idf).
According to cosine similarity, term frequency (tf) refers to the count of occurrence of the term in the sentence and normalized document frequency (idf) is given as:

\[ idf = \log \left( \frac{n}{df} \right) \]  

(2)

Where, \( n \) = number of sentences in the document and \( df \) = frequency of the word in the document.

After removing the redundancy in the text a weight value is assigned to each sentence. Weight for each sentence is given based on the “Code Quantity and Attention” linguistic principle. According to this principle weight for each sentence \( w_{t_2} \) is defined as:

\[ w_{t_2} = w_{t_1} + a_f \]  

(3)

\( w_{t_2} \) Refers to the code quantity. According to [1], coding material can be considered as the noun phrase which has a syntactic structure carrying more information as per user’s needs. The weight \( w_{t_2} \) is calculated as follows,

\[ w_{t_2} = \frac{\text{number of noun phrases in the sentence}}{\text{Total number of words in the sentence}} \]  

(4)

The noun phrases in our summarizer tool are being extracted by connecting to Word Net Dictionary using Java Word Net Library (JWNL). \( a_f \) Refers to the “attention” in linguistic principle. The sentence which contains the words appearing in bold, italic, underlined, or any combination of them, is treated as important sentence and more attention of the reader will be caught. Attention factor \( a_f \) is defined as:

\[ a_f = \frac{\text{count of the special effect term in the sentence}}{\text{a_value}} \]  

Where \( a_{\text{value}} \) is taken as follows: for bold, italic, underlined, \( a_{\text{value}}=1 \), for bold-italic, italic-underlined, bold-underlined, for bold-italic-underlined, \( a_{\text{value}}=3 \).

After assigning the weight to each sentence, the next job is to rank the individual sentence according to their weight value. Finally, our summarizer extracts the higher rank sentences including the first sentence of the document. The number of sentences extracted is based on the user requirement i.e. the percentages of summary the user give as input. This percentage is calculated by dividing the percentage given by the user by total number of ranked sentences, and then taking the ceiling of that result.

(i) Algorithm:

- **Input:** A text file with .txt or .rtf extension and the Percentage of the text user want to extract in resultant summary.
- 1. Read the text file in .txt or .rtf format,
- 2. splitting the text file into individual tokens,
- 3. Removing the stop words to filter the text,
- 4. Extract the Parts Of Speech of each word by connecting to the Word Net dictionary using Java Word Net Library (JWNL),
- 5. Removing the redundancy in the text using cosine Similarity between the sentences,
- 6. Extracting the nouns in each sentence and giving a weight For each sentence as :

\[ w_{t_1} = \frac{\text{number of nouns in the sentence}}{\text{Total number of words in the sentence}} \]

7. Assign an Attention factor \( a_f \) to the sentences which appear in bold, italic, underlined or any combination of these.

\[ a_f = \frac{\text{count of the special effect term in the sentence}}{a_{\text{value}}} \]

Where \( a_{\text{value}} \) is taken as follows: for bold, italic, Underlined, \( a_{\text{value}}=1 \) for bold-italic, italic-underlined, bold- underlined, for bold-italic-underlined, \( a_{\text{value}}=3 \)

8. Calculate the total weight of the each sentence as:

\[ w_{t_2} = w_{t_1} + a_f \]

9. Rank the sentences according the weight of the sentence \( w_{t_2} \).

10. Finally, extract the higher ranked sentences including the First sentence of the input text in order to find the required Summary.

- **Output:** A relevant summarized text which is shorter than the original text.

V. RESULTS AND DISCUSSION

We have tested our system with 10 documents (five .rtf and five .txt files). Here each document contains around 20 sentences. For auto summarization we have fixed the percentage of summary as 25%, i.e. it will reduce the summary to quarter of the original document. The Screen shot of our system is given in Figure 1.

![Fig 1: Screen Short of our System](image-url)
A comparison of our system with Ms Word summarization is given in Table 1. The relevancy of the summary is calculated with respect to human judgment for both the systems. The details of the result are given in Table 1 and the graphical representation of the relevancy of both the systems with respect to human judgment is given in Figure 2. In Figure 2, the human judgment extracted sentences are considered as the best summary i.e. it is considered as 1. The results clearly show the improvement of our system over the MS Word summarizer.

VI. CONCLUSION AND FUTURE WORK

In this paper we have done a new thing which has not been taken into consideration in MS Word automatic summarization. Firstly we have removed the redundancy using cosine similarity and secondly we have extracted the relevant sentences based on code-quantity and attention, and calculated the weights and ranked the sentences of the document accordingly. For this reason the accuracy rate of our system is more than that of Ms Word automatic text summarization in most cases. Since the summarization follows the extraction method, when it extracts the important sentences it might happen that one sentence contains a proper noun and the next sentence contains a pronoun as a reference of the proper noun. In that case, if the summary considers the second sentence without considering the first one, then it does not give its proper meaning. It is a big issue in automatic text summarization. We are working to resolve this type of anaphoric problems in text summarization.

REFERENCES

[1] Elena Lloret and Manuel Palomar, “Challenging Issues of Automatic Summarization: Relevance Detection and Quality-based Evaluation”, International Journal of Informatics, Vol. 34, Issue 1, 2010, pp. 29–35.
AUTHOR BIOGRAPHY

Pranitha Reddy received her B.Tech degree in 2009 from Jawaharlal Nehru Technological University, Hyderabad, India. Currently she is pursuing her M.Tech degree from Department of Computer Science and Engineering, International Institute of Informational Technology, Bhubaneswar, Odisha, India. Her current research interests include Information Retrieval and Text Summarization.
E-mail: a110024@iiit-bh.ac.in

R C Balabantaray is currently working as an assistant professor in the Department of Computer Science and Engineering, International Institute of Informational Technology, Bhubaneswar, Odisha, India. His current research interests include Information Retrieval and Natural Language Processing.
E-mail: rakesh@iiit-bh.ac.in