A Feature Terms based Method for Improving Text Summarization with Supervised POS Tagging

Suneetha Manne
Associate Professor
Department of Information Technology
VR Siddhartha Engineering College
Vijayawada, Andhra Pradesh

S. Sameen Fatima
PhD, Professor and Head
Dept. of Computer Science & Engg.
University College of Engineering
Osmania University, Hyderabad

ABSTRACT
Text summarization is the process of distilling the most important information from a source to produce an abridged version for a particular user and task. When this is done by means of a computer, i.e. automatically, it calls as Automatic Text Summarization. Summarization can be classified into two approaches: extraction and abstraction. Extraction based summaries are produced by concatenating several sentences taken exactly as they appear in the texts being summarized. Abstraction based summaries are written to convey the main information in the input and may reuse phrases or clauses from it. This paper focuses on extraction approach. The goal of text summarization based on extraction approach is sentences selection. One of the methods to obtain the sentences is to assign some feature terms of sentences for the summary called ranking sentences and then select the best ones. The first step in summarization by extraction is the identification of important features. In our approach 1000 computer science related research papers are used as test documents. Each document is prepared by preprocessing process: sentence segmentation, tokenization, stop word removal, case folding, lemmatization, and stemming. Then, using important features, sentence filtering features, data compression features and finally calculating score for each sentence. The proposed text summarization is based on HMM tagger to improve the quality of the summary. Here, comparing our results with the existing summarizers which are Copernicus summarizer, Great summarizer and Microsoft Word 2007 summarizers etc. The proposed system is also tested with four types' similarities: Cosine, Jaccard, Jaro-winkler and Sorenson similarities. The results show that the best quality for the summaries was obtained by feature terms method.

General Terms
Text Mining, Information Extraction, Automatic Text Summarization, Natural Language Processing, POS Tagging.

Keywords
Term frequency, Term weight, Inverse sentence frequency, Noun and verb chunking, Sentence Position, Sentence Length, Verb featured sentences, Compression Ratio, Retention Ratio, Data Compression features, HMM Tagger.

1. INTRODUCTION
Today, the amount of information available is tremendous and the number of pages available on the Internet almost doubles every year [1]. In January 2012, the number of hosts advertised in the DNS is 888,239,420 [2]. In order to find the relevant information and making use of this is becoming more difficult. Nowadays, most of the information comes from the Internet is in the form of text. Due to lack of presence of contextual and discourse awareness, most of the users are not aware of the relevance and appropriateness of the information in the web documents. To reduce the effort in finding the relative information with contextual and discourse awareness, summarization is a powerful technique which is helpful for the user to find appropriate information. In this context, automatic text summarization is an area of research that attracted by many researchers [20].

According to Spark-Jones "Text summarization is a reductive transformation of a source text into a summary text by extraction or generation" [3]. Automatic summarization is nothing but a computer generated summary. Though, the automatic text summarization is predominant research area very few software tools are available to the end users. The reason for this is that the quality of summaries produced by automatically is low. In general, creation of a good summary requires a lot of intelligence. The clear evidence of this is use of summaries in document understanding and the relevant information from the large volume of texts. Various approaches are available in the literature to perform text summarization, Information extraction (IE), NLP and others. IE is a special type of text summarization that is designed specifically for the input documents [4]. Thus, in this investigation the summary generation for academic research papers have been proposed.

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. 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. Also machine learning techniques from closely related fields such as information retrieval or text mining have been adapted to help automatic summarization [5].

Researchers and students constantly face the problem that, it is almost impossible to read most of the newly published papers to be informed of the latest progress and when they work on a research project, the time spent on reading literature review seems endless. The goal of this research is to design a domain independent automatic text extraction based summarization system to alleviate, if not totally solve, this problem. The organization of the paper is as follows. Section 2 discusses the existing techniques of text document summarization. In Section 3 we discuss the proposed approach of automatic text summarization. Experimental results and Performance evaluation are depicted in Section 4. Section 5 concludes the paper.

2. RESEARCH BACKGROUND

All In this section we will review some of the literature about automatic document summarization. Automated text summarization is an old eminent research area and the earliest work dates back to the 1950s that has lain dormant for almost three decades. As a result of the information overloading on the web there is large-scale interest in automatic text summarization.

Most early work on single-document summarization focused on technical documents and was pioneered by Luhn [6]. He presented the first exploratory research on automatic abstracting provides a simple method for creating abstracts from specialized literature in the year 1958. Luhn used an algorithm which scans the source text document for the most salient information. In this algorithm to measure the significance of each sentence within an article, word frequency and sentence scoring are used. A cutoff value for significant factor was initially set depending on which the featured sentences are extracted. The system produced by Luhn was given of reasonable quality there were not many documents in electronic form, at that time. The system also restricted too few specific areas of literature and also limited to small input data. Baxendale [7] applied sentence position as a useful feature to finding salient parts of documents.

In 1969 Edmundson proposed a new concept of cue words and is one of the most influential in the area of automatic summarization. The important in Edmunson's work was the introduction of three new parameters for calculating the weights of sentences. Those were the sentence position in text, cue words and title and heading words. He divided the entire structure of the text into two parts. One is —Body which contains the main data and second is—Skeleton which contains title, heading (e.g. Introduction, Purpose, Conclusions etc.) and format of the file. The Cue words are recognized within the text and are compared with the Cue Dictionary corpus and there by cue weights are calculated. This approach suffered with huge time complexity, lacked simplicity and even restricting the entire model to Cue Dictionary [8].

Pollock and Zamora used an interesting algorithm which was used for sentence rejection rather than selection in 1975. The aim of the paper was to develop a system which outputs a summary which conforms to the standards of the Chemical Abstracts Service (CAS) [9]. Lin and Hovy claimed that as the discourse structures change over the domains and the genres, the position method cannot be simple as used by Baxendale [10].

The first endeavor for generating abstractive summaries was succeeded by ADAM Summarizer in 1975.Rather adopting linguistic techniques; ADAMS is built on the framework of Machine Learning to generate summaries through sentence ranking. It had potential to handle new domains in addition to redundancy elimination. K.R. Mc Keown in his thesis [11] in the year 1984, generated the first summary system using Natural Language Processing (NLP) based on a computational model of discourse strategies of what to say next after a sentence.

In 1995, a research paper [12] presented Term Weighting and Sentence Weighting as important features to recognize the featured sentences. It was succeeded to some extent in solving anaphoric resolutions in summary generated. Barzilay & Elahadad [13], Boguraev & Kennedy [14], Mercer [15] in 1997, Truney and Frank [16] in 1999, common to all these systems is the approach of extracting keyphrases from text as a supervised learning task. All these systems require a separate training document set with keyphrases already assigned in order to function properly. This remained as a challenge for research community.

In 2001, Cut and Paste [17] is the first domain independent abstractive summarization tool developed using sentence reduction and sentence combination techniques. Here a sentence extraction algorithm was implemented covering various parameters like lexical coherence, tf×idf score, cue phrases and sentence positions etc.

MEAD [18] developed in the year 2001 was a multi document summarization toolkit that implemented multiple summarization algorithms such as position-based, TF×IDF, largest common subsequence, and keywords. The methods for evaluating the quality of the summaries are both intrinsic (such as percent agreement, precision/recall, and relative utility) and extrinsic (document rank).A latest version of MEAD is based on centroid based multi document summarization.

Hongyan Jing in his paper [17] called identifying the whether the summary is generated reusing the original text or by reusing the phrases. This summary decomposition problem was solved by using Hidden Markov Models (HMM) [19]. Here in this paper, using the Parts of Speech Tagging that uses the HMM Tagger set to identify the key phrases within the text.

3. PROPOSED SYSTEM

The objective of this research paper is to develop a flexible software tool to do automatic text summarization for technical text documents in the domain of Computer Science and is suitable for any domain text documents. The proposed system architecture diagram is shown in Fig. 1.

3.1 Feed Subjugation

The input document can be of any document format (doc, txt, pdf, html, rtf), hence the system first applies document converters to extract the text from the input document. In Feed subjugation the concept of evaluation of character encoding schemes is applied. In order to process the fed input we need to evaluate the encoding scheme of file to Unicode. For this need parsers for each format. Parses used for Feed Subjugation are PDF parser –AspriseJavaPdf.jar, Html parser – NekoHtml.jar, Xerces.jar, and XercsImpl.jar distributed under: Apache 2.0 and document Parser – poi. jar, distributed and developed by: Apache for manipulating Microsoft docs.

3.2 Text Prologuing

Pre-processing the text before incepting to summarization and categorization is Text Prologuing. It consists of three phases which are text segmentation, normalization and phase chunking.

3.2.1 Text Segmentation

Is the process of decomposing the given text into its constituent sentences, calculating each sentence length and word count. This module divides the document into sentences. At first glance, it may appear that using end of sentence punctuation marks, such as periods, question marks, and exclamation points, is sufficient for marking the sentence boundaries.

3.2.2 Normalization

Is the the process of converting words into normalized form. The following are the processes that come under normalization techniques.

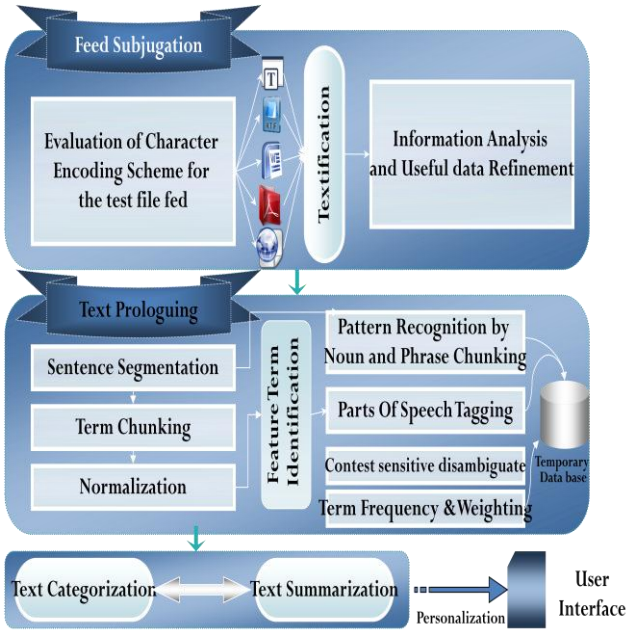

Fig. 1. Proposed Text Summarization system architecture diagram

3.2.3 Tokenization

It is the process of splitting of the sentence into words using StringTokenizer class in the java.util package.

3.2.4 Stop word Removal

During the retrieval of relevant information we have to remove few words, numbers, and special symbols etc., which have less significance.

3.2.5 Case Folding

Converting entire words in the sentences into lower case so as to avoid repetition of same word in different cases like sentence case, capital case, title case, upper case etc.

3.2.6 Lemmatizing

Extracting the commonly featured, same meaning tokenized words so as to avoid repetition (e.g. problems-problem, risks-risk, etc.). It is a subset of stemming where only the suffixes are treated to clip or few entailments needed

3.2.7 Stemming

Mechanically removing or changing the suffixes of some nouns or verbs. Stemming improves the retrieval performance because they reduce variants of the same root word to a common concept. It also reduces the size of the indexing structure because the number of distinct index terms is reduced. The design of a stemmer is language specific, and requires some significant linguistic expertise in the language. Here we proposed an integrated stemming approach which involves both Porters and Chris D. Paice stemming approaches. The proposed integrated model showed better impacting results with respect to words affected and computing time.

3.3 Feature Term Identification

The tokenized terms after applying various normalized techniques like case folding, stop word removal, stemming, lemmatizing, etc., are now considered as Feature Terms. The system extracts both the sentence level and the word level features which are used in calculating the importance of the sentence towards the document.

Parts of Speech Tagging: Identifying the parts of speech of each feature term by creating an interface that specifies a means of doing sentence segmentation from arrays of tokens and whitespaces.

3.3.1 Various Approaches of POS Tagging

Tagging allows us to use a predefined model (simply say the feature term) to assign part of speech tags to text. A model file which is manually tagged already used to tag the predefined model is considered as training. Training corpus used is pos-en-generalbrown. Hidden Markov Model can be considered a generalization of a mixture model where the hidden variables, which control the mixture component to be selected for each observation, are related through a Markov process rather than independent of each other [15].

3.3.2 Pattern Recognition by Noun and Verb Chunking

Extract high level structures like phrases can be possible using Noun and Verb Chunking. Nouns may start with determiners, adjectives, common nouns or pronouns and they continued with any category that may start a noun, or adverbs or punctuation. Verbs may start with verbs, auxiliaries, or adverbs and may be continued with any of the tags, or with punctuation. These sets are defined statically by using a set of determiner tags to a Noun or Verb. The n-best output for taggers could be used to define chunks. Certain challenges raised while tagging in accordance to Noun and phrase chunking are overcame by acquiring patches. Patches improved the overall performance of the Tagger. Patches are of the forms which are shown Table 1.

Table 1. Patches used to improve the tagger performance

1. If a word is tagged a and it is in context C, then change that tag to b e.g. The Broad gauged rails are more serviceable than a narrow gauged. Here, gauged is verb in real but being a noun as a whole phrase “The Broad gauged rails”, gauged is now a adjective with broad 2. If a word is tagged a and it has lexical property P, then change that tag to b 3. If a word is tagged a and a word in region R has lexical property P, then change that tag to b

Tagging of such Noun chunk and Verb chunk will retain POS of word at lexical level. Few Context Sensitive Ambiguities can be solved by simple cases which are shown in Table 2. Thus a successful POS tagging technique can be implemented in our proposed system with near 100% accuracy in tagging so as to select Feature Terms.

3.3.3 Term Frequency and Term weight

The Strength of the POS based filtered Feature terms can be found by using the classical techniques of Term Frequency (TF) and Term Weighting (TW). The term frequency tf (t, d) of term t in document d is defined as the number of times that t occurs in d.

Term Frequency (TFi)=no. of times a term repeated

Term Weight (TWi)= [TFi *1000] /total no. of terms

Table 2. Context sensitive ambiguities rules

```
ContextSens (String[] words, String[] result){
//Inputs: ArrayList of N words after Feature Term Identification
//ArrayList of Result Tags after primary POS Tagging for each word at i=1,…..N
if word[i].startsWith("VB") && result[i-1].equals("DT") then result[i]:="NN"
if word[i].indexof(".">-1) && result [i].startswith("N") then result[i]="CD"
if result[i].startsWith("N") && (words [i].endsWith("ed") then result[i] = "VBN";
if words[i].endsWith("ly") then result [i] = "RB“;
if result[i].startsWith("NN") && (words [i].endsWith ("al") then result [i] = "JJ";
if result[i].startsWith("NN") && words [i - 1].equals("would") then result [i] = "VB";
if result[i].equals("NN")&& words[i].endsWith("s")then result[i]= "NNS";
if result[i].startsWith("NN") && words[i].endsWith("ing") then result[i] = "VBG";
if result[i].startsWith("NN") && words[i].length() > 1 then result[i] = "NNP";
}
```

3.4 Text Summarization

For best extractive summary results the following features are used:

3.4.1 Sentence length

We consider length of a sentence as a feature because we observe that if a sentence is too short, but it occurs in the beginning paragraph of a document it is sometimes selected due to its positional advantage. So, we eliminate the sentences which are too short.

Normalized sentence length=

$$\frac{\text{no.of words occuring in the sentence}}{\text{no.of words ocurring in the longest sentence of the document}}$$

3.4.2 Sentence Position

Number of sentences in a paragraph is n, then n/2 top sentences are considered top priority than that the next n/2 sentences. Paragraph can be recognized using ends with "//s//s//s" (sentence ended with four or more spaces).

3.4.3 Sentence weight

If any sentence contain featured words selected by applying POS tagging, then that sentence is considered as important even they placed in remaining n/2 sentences. This can accomplish with the help of sentence weight

Sentence weight (SW) =

$$\frac{\text{no.of featured terms within the sentence} * 1000}{\text{total no.of terms in a paragraph}}$$

3.4.4 Inverse Sentence Frequency

The problem of TF weighting in IR is that, when a term appears in almost all the sentences, this term is useless for discriminating relevant documents. For such useless terms for discriminating the relevant sentences, Inverse Sentence Frequency is calculated. It can be defined as

$$ISF(ti) = log\frac{(N)}{ni}$$

Where ti is the term, N is the number of sentences and ni is the number of Sentences where ti appears. The less is the ISF the more is the probability for a term to be useless. It is hence when TW is multiplied ISF times may give best statistic results than usual TW. This can be represented as

$$R\,Weight_{(whole)}(\text{ti}) = TWi * ISF(ti)$$

3.4.5 Data Compression with less Information Loss

Sub-Categorization is a key for data compression. For noun and clause heads, in order to preserve the sentence coherence (subject or noun, verbal group, articles) some of our complements have been identified as systematically mandatory. Other heads (verb, adjective, adverb, preposition and pronoun) may admit optional or mandatory complements, depending on either the lexical head category or a particular head instance (a lexical entry).

E.g. Indian Hockey Team won the Asian Cup against Pakistan here in this stadium three days ago.

Here in this sentence, the italic part is optional complements when the summary deals on the title "Hockey Play". Even Pakistan is not mandatory when summary aims at Indian Hockey team story as head category. Indeed, prepositions are systematically requiring a complement, while other heads must be considered on a case-by-case. basis. Once we get the sub-categorization information for a head, we are able to determine whether its complement (s) can be removed without causing incoherence.

3.4.5.1 Compression Ratio

How much shorter the summary is than the original. It can be represented as

$$\text{CompresstionRatio(CR)} = \frac{\text{Length\_of\_Summary\_needed}}{\text{Length\_of\_Full\_Text}}$$

3.4.5.2 Retention Ratio

How much information is still retained in the summary is represented as

$$\text{RetentionRatio(RR)} = \frac{\text{Information\_in\_Summary}}{\text{Information\_in\_Full\_Text}}$$

However we choose to measure the length and the information content, we can say that a good summary is one

in which CR is small (tending to zero) while RR is large (tending to unity). We can characterize summarization systems and/or text types by plotting the ratios of the summaries produced under varying conditions.

3.4.5.3 Position Feature induced in Compression

The sentences cited first in most positions of n. Consider an example data which contains a paragraph with 6 sentences, then first sentence has 6/6 value, the second sentence has 5/6 value and so on. The last sentence has 1/6 value. The value up to which the sentences are allowed is determined on the basis of compression ratio (CR) defined initially.

$$\text{PositionFeature}\left(PF_{S_{i,k}}\right) = \frac{\text{Position\_of\_S\_in\_k\_words}}{(CR / 100)}$$

Where s is the sentence array from i=0 to k. CR is Compression Ratio given as input.

3.4.5.4 Verb Featured Sentences (VFS)

In general any sentence must contain at least one verb which defines the entire functionality of the sentence. It is hence verbs are most probably included in the document summary. The formula we used to calculate the inclusion of verbs in the sentence.

$$\text{VerbFeatureSeng}\left(VFS_{s_{i,k}}\right) = \frac{\text{No.\_of\_words\_with\_POS\_verb}}{\text{No.\_of\_total\_words\_in\_s}}$$

3.4.5.5 Co-Reference Resolution in Summarization

The featured sentences extracted from the input may sometime contain the same fragments of data. This can be avoided by comparing patterns among resulting sentences and sorting out the sentence which have maximum value of VFS. Here First sentence is informative than second as VFS is more for first.

3.4.5.6 Content Proximity in binding

The summary generated must contain appreciable amount of binding sequence between two successive sentences. This implies how correctly or closely does a sentence fit to precede another sentence. This is can be done by verb linkage between the successive sentence. The sentences that usually start with Hence, Then, So, Therefore, But, No sooner, etc. need less distance between the sentences and preceding sentence must have VFS value more.

4. Experimental Results and Performance Analysis

A test document of any of the prescribed format (pdf, html, doc, txt, xml, etc.) and percentage of summary will be given to proposed system as input. The processing parameters at user end are as follows:

- Term Frequency, Featured Terms, Term Weighting, Parts Of Speech Tag
- Words after Term Repetition Removal, before & after Normalization
- Bounded Frequencies (Possible, Selected, Best)

The generated summary by proposed system is tested with the following existing summarizers for 1000 documents. The user interface for proposed system is shown in Fig. 2.

- SSSummarizer
- MS Word Summarizer Tool
- Copernic Summarizer
- Great Summary –Online Summarizer
- Tools 4 Noobs –Online Summarizer
- SweSum –Online Summarizer
- Open Text Summarizer –Local Summarizer
- QuickJist –Local Summarizer

Fig 2. User interface with results for proposed system.

4.1 Evaluation Methods

The quality of the generated summary is verified with four similarity functions as cosine, Jaccard, Jaro-winkler and Sorenson similarities. Totally 100 documents summary quality is tested. The similarity percentage is found between abstract of given technical document and the generated summary.

4.1.1 Cosine Similarity

Cosine similarity can literally be defined as the angular difference between two vectors. Here one thing important is if we get same word twice or more, then the performance degrades even the result is good.

$$CosSimilarity = \cos\theta = \frac{A.B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2}\sqrt{\sum_{i=1}^{n}(B_i)^2}}$$

Where A –The Abstract in the document stored in a local string arraylist.

B -The Summary generated by our software for the same document.

The feature terms in the entire documents are already founded. Now each word is a dimension and matrix will be created.

4.1.2 Jaccard Similarity

$$Jaccard\ Similarity = \frac{|A \cap B|}{|A \cup B|}$$

The inputs are the same here too with no additional calculations needed. Jaccard similarity is simply defined as the length of the intersection divided by the length of the union.

The inputs are the same here too with no additional calculations needed. Jaccard similarity is simply defined as the length of the intersection divided by the length of the union

4.1.3 Jaro – winkler Similarity

$$d_j = \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right)$$

Here,
dj = Jaro Distance
m = number of matching characters
t = number of transposed characters (found words but not in the same positions)
|s1| = length of the first string
|s2| = length of the second string

It is good similarity measuring parameter because till now all the similarity measures are irrespective of order of occurrence. But Jaro-Winkler is order dependent.

For example if we gave two inputs MEDIA, AIMED, the above similarity measures show 100% similar, but Jaro-Winkler doesn't since the order of letters is not the same here though the letters are same. Jaro-Winkler similarity has shown better results than any other measures.

4.1.4 Sorenson Similarity

It follows the same procedure as Jacaard but with little difference.

$$Sorensen\ Similarity = \frac{2 \times |A \cap B|}{|A| + |B|}$$

The no. of documents vs similarity charts for 50 documents are shown in Fig 3, Fig 4 and Fig 5. The proposed system has given the good results as compared with the existing summarizers. The below charts show that proposed system has more similarity regarding the generated summary with the comparison of existing summarizers.

5. CONCLUSIONS

In this paper, an extractive automatic text summarization approach by sentence extraction is evaluated with the help of one of the supervised tagging method Hidden Markov Model (HMM). A feature term based text summarization technique with HMM tagger is designed and implemented successfully in a popular and challenging higher level programming language Java. We extracted the important features for each sentence of the document and finally represented as the summary with features consisting of the following elements: term frequency, term weight, inverse sentence frequency, sentence length, sentence weight, sentence position, data compression features without information loss and sentence to sentence similarity. Ranked sentences are collected by identifying these feature terms, key phrases, the final summary is obtained. Such type of summary generated can also be used for Text Categorization systems in order to label the documents so as to organize easily on web.

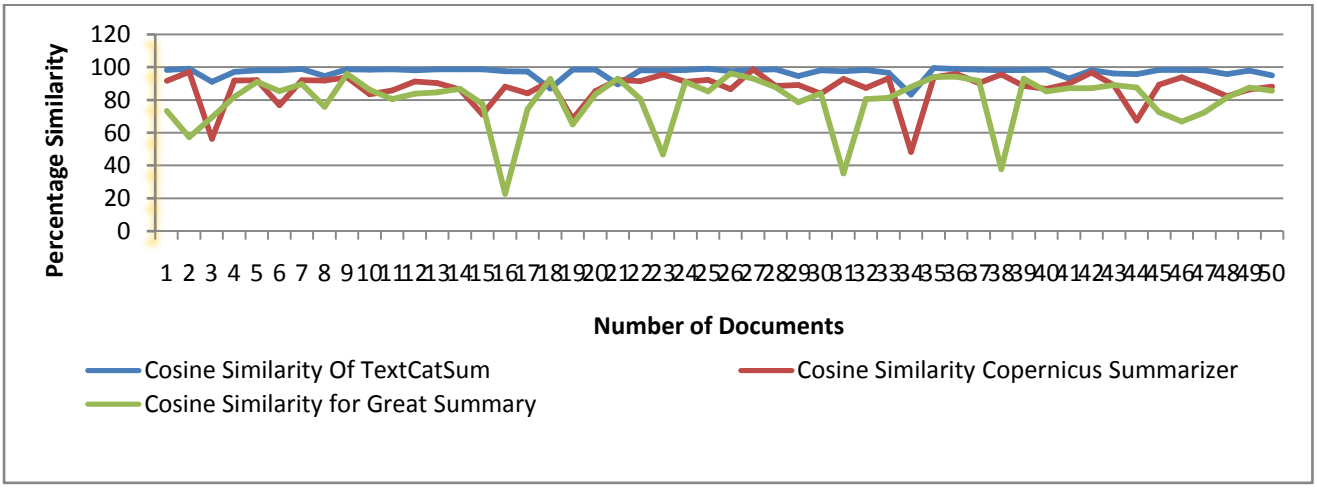

Fig. 3. Cosine Similarity chart

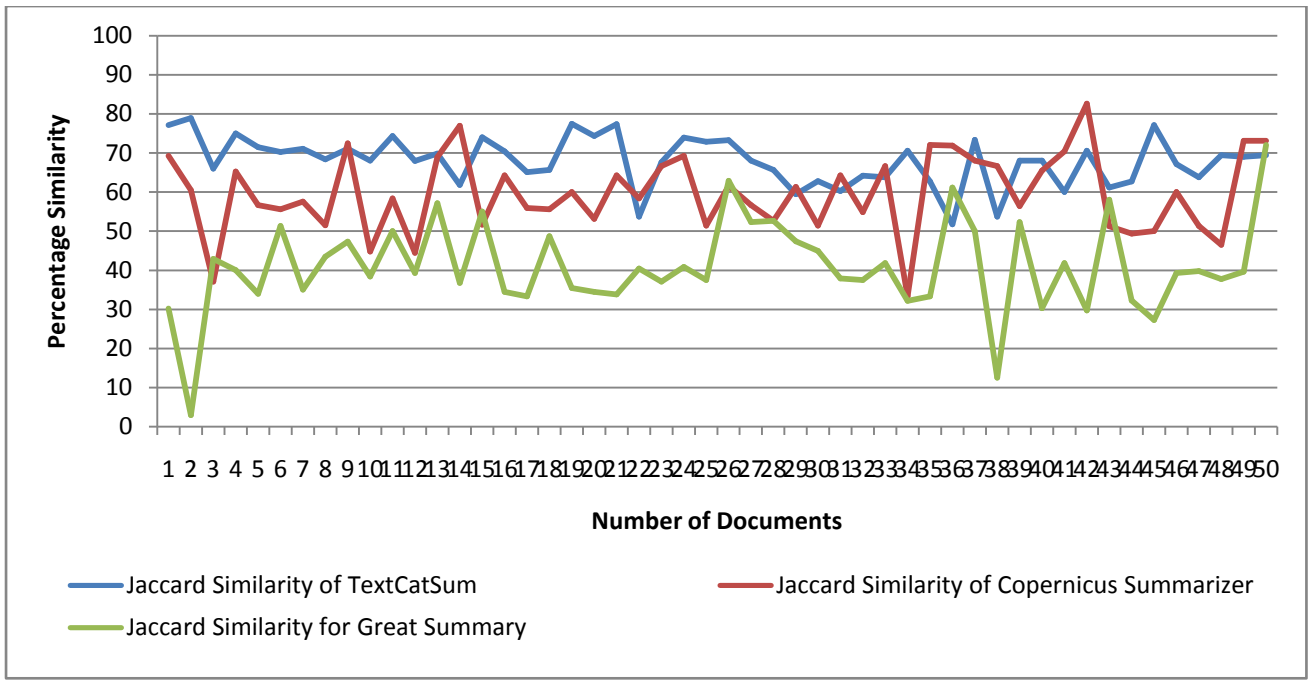

Fig. 4.Jaccard Similarity chart

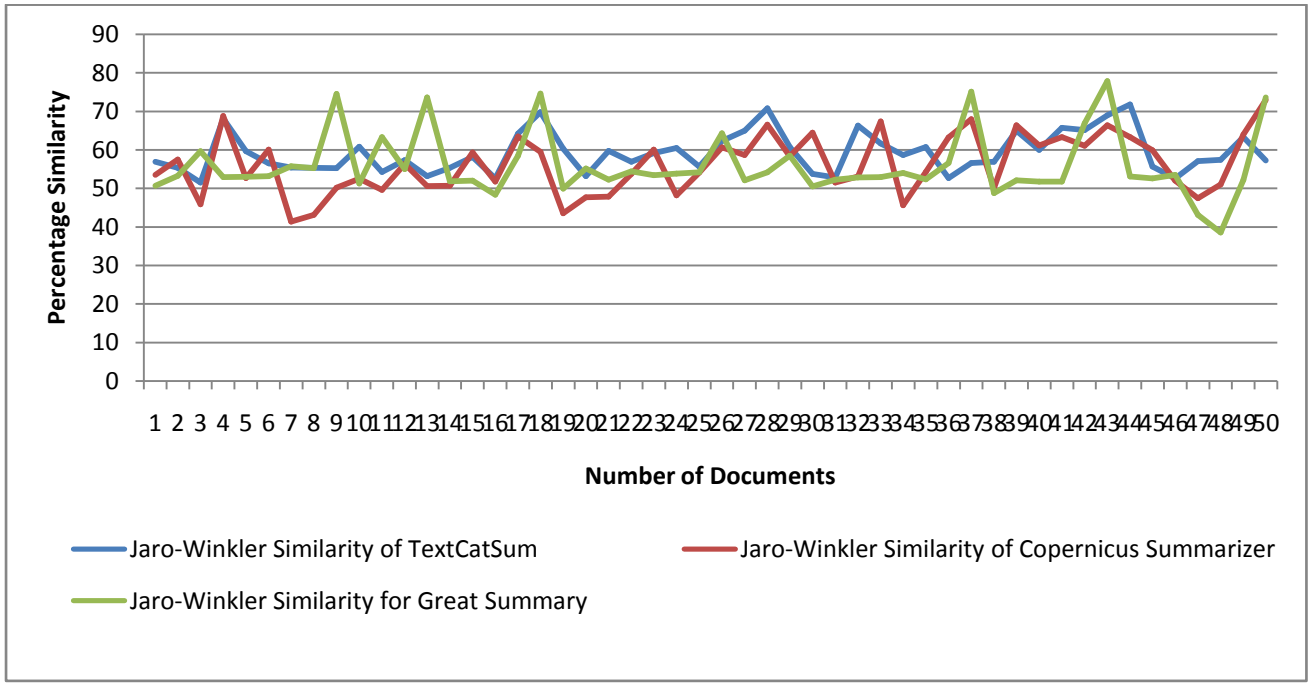

Fig. 5. Jaro - Winkler Similarity chart

6. REFERENCES

[1] D. R. Radev and W. Fan, "Automatic summarization of search engine hit lists", Proceedings of the ACL-2000 workshop on recent advances in natural language processing and information retrieval, Hong Kong, 2000, pp. 99-109

[2] ISC "ISC Internet Domain Survey", Available at: http://ftp.isc.org/www/survey/reports/current/

[3] Sparck-Jones, K. "Automatic Summarizing: Factors and Directions" in Mani, I. And Maybury, M., editors, Advances in Automatic Text Summarization, pp.1-12. MIT Press, 1999

[4] C. H. Chang, M. Kayed, M. R. Girgis, and K. Shaalan, "A survey of web information extraction systems," IEEE Transactions on Knowledge and Data Engineering, Vol. 18, 2006, pp. 1411-1428

[5] Vishal Gupta and Gurpreet Singh Lehal, "A Survey of Text Summarization Extractive Techniques", Journal of Emerging Technologies In Web Intelligence, Vol. 2, No. 3, August 2010.

[6] Luhn H.P, "The Automatic Creation of Literature Abstracts", IBM Journal April 1958 pp. 159–165.

[7] Baxendale, P. (1958), 'Machine-made Index for Technical Literature –An Experiment', IBM Journal of Research Development, Vol. 2, No.4, pp. 354-361.

[8] Edmundson H.P, " New Methods in Automatic Extracting", Journal of the Association for Computing Machinery, Vol 16, No 2, April 1969, PP. 264-285.

[9] J.J.Pollock and A. Zamora , "Automatic Abstracting Research at Chemical Abstracts Service", Journal of Chemical Information and Computer Sciences, 15(4), 226-232(1975).

[10] Lin, C.-Y. and E. Hovy (1997). Identifying topics by position. In Proceedings of the Applied Natural Language Processing Conference (ANLP-97), Washington, DC, pp. 283-290.

[11] Kathleen R. McKeown, "Discourse Strategies for Generating Natural Language Text", Department of Computer Science, Columbia University, New York, 1982

[12] Brandow, R., Mitze, K., Rau, L. F. Automatic condensation of electronic publications by sentence selection. Information Processing anagement,31(5):675-685, 1995.

[13] Barzilay R., Elhadad M., Boguraev & Kennedy M., Using Lexical Chains for Text Summarization, Workshop on Intelligent Scalable Text Summarization, Ben Gurion University of the Negev, Be'er Sheva Israel, 1997.

[14] B K Boguraev, C Kennedy, R Bellamy, "Dynamic presentation of phrasally-based document abstractions." 32nd International Conference on System Sciences, 1999.

[15] Marcu, D. 1999. The automatic construction of large-scale corpora for summarization research. In Proceedings of the 22nd International Conference on Research and Development in Information Retrieval, University of California, Berkeley, August.

[16] Turney. 1999. Learning to extract keyphrases from text. Teical chnReport ERB-1057. (NRC#41622), National Research Council, Institute for Information Technology.

[17] Jing, Hongyan and Kathleen McKeown. 2000. Cut and paste based text summarization. In 1st Conference of the North American Chapter of the Association for Computational Linguistics

[18] Radev, R., Blair-goldensohn, S, Zhang, Z. Experiments in Single and Multi-Docuemtn Summarization using MEAD. In First Document Understanding Conference, New Orleans, LA, 2001.

[19] L. Bahl and R. L. Mercer, "Part-Of-Speech assignment by a statistical decision algorithm", IEEE International Symposium on Information Theory, pages: 88 - 89, 1976.

[20] H. Dang and K. Owczarzak, "Overview of the TAC 2008 Update Summarization Task," in Proceedings of Text Analysis Conference, 2008, pp. 1–16