Towards Efficient model for Automatic Text Summarization

Yetunde O. Folajimi
Department of Computer Science
University of Ibadan.
+2348056648530
yetundeofolajimi@gmail.com

Tijesuni I. Obereke
Department of Computer Science
University of Ibadan.
+2348137462256
tobereke@gmail.com

ABSTRACT

Automatic text summarization aims at producing summary from a document or a set of documents. It has become a widely explored area of research as the need for immediate access to relevant and precise information that can effectively represent huge amount of information. Because relevant information is scattered across a given document, every user is faced with the problem of going through a large amount of information to get to the main gist of a text. This calls for the need to be able to view a smaller portion of large documents without necessarily losing the important aspect of the information contained therein. This paper provides an overview of current technologies, techniques and challenges in automatic text summarization. Consequently, we discuss our efforts at providing an efficient model for compact and concise documents summarization using sentence scoring algorithm and a sentence reduction algorithm. Based on comparison with the well-known Copernic summarizer and the FreeSummarizer, our system showed that the summarized sentences contain more relevant information such that selected sentences are relevant to the query posed by the user

CS Concepts

• Computing methodologies →Artificial intelligence →Natural language processing →Information extraction • Information Systems →Information Retrieval →Retrieval tasks and goals →Summarization

Keywords

Automatic text summarization, extractive summary, sentence scoring, sentence reduction, query-based summarization.

1. INTRODUCTION

The area of automatic text summarization has become a widely explored area of research because of the need for immediate access to information at this age where the amount of information on the World Wide Web is voluminous. The problem is not the availability of information but users have access to more than enough information than they need, they are also faced with the problem of digging through that large amount of information to get what they really need.

Automatic text summarization is a process whereby a computer program takes in a text and outputs a short version of the text retaining the important parts only. The essence of text summarization is to bring out the salient parts of a text.

The method used for automatic text summarization can either be extractive or abstractive. Extractive summarization method involves picking important sentences from a document while abstractive method of summarization involves the use of linguistic methods to analyze and interpret a document, the system then looks for another way to portray the content of the document in a short form and still pass across the main gist of the document. Also the input of a text summarization system can either be single or multiple. Single document summarization involves summarizing a single text while Multi-document summarization involves summarizing from more than one source text.

Automatic text summarization is one of the many applications of Natural Language Processing. It can be used for question and answering, information retrieval among other things. Earlier methods of text summarization used statistical methods that assigned scores to sentences or words in a sentence, and these methods are inefficient because they didn't consider the context of words, which made the resulting summaries, incoherent. More research unveiled approaches that do not score sentences for extraction, but merged lots of knowledge bases to enable them know the part of speech of words in a sentence but do not consider keywords identification to identify important parts of documents..

Automatic text summarization system helps saves time and effort that one would have used to scan a whole document, it also helps increase productivity and with the amount of research that has been done in automatic text summarization, summaries are available in different languages [1].

This paper presents the current technologies and techniques as well as prevailing challenges in automatic text summarization, consequently, we propose a model for improving text summarization by using a method that combines sentence scoring algorithm with sentence reduction.

2. SENTENCE EXTRACTION METHODS FOR TEXT SUMMARIZATION

A system that scans a document in machine-readable form then selects from the sentences in the Article the ones that carry important information was proposed in [2]. The significance factor of a sentence is derived from an analysis of its words, whereby the frequency of words occurrence is a useful measurement of word significance and that the relative position of words within a sentence having given values of significance furnishes a useful measurement for determining the significance of sentences. In [2], a justification was made about measure of significance based on how frequent some words occurred by

CoRI'16, Sept 7–9, 2016, Ibadan, Nigeria.

pointing out that an author when trying to express his thoughts on a subject repeats some words.

Another research in IBM pointed out that the position of a sentence can be used to find areas of a document containing important information [3]. There, it was shown that sentences that occur in the initial or final parts of a paragraph contain important information. By analyzing a sample of 200 paragraphs, it was discovered that in most paragraphs the headings came first and in few it came last.

Unlike the method used in [2], which used only the frequency of word occurrence to produce extracts, [4] analyzed using cue words, title and heading words, sentence location and key method individually and together. The justification of using cue method is that sentences containing words like "most importantly", "in this paper" indicate sentence importance. For key method, scores were assigned to frequently occurring words in the document. For title method, sentences are scored based on how much of the title or heading words it contains and for the sentence location, the importance of a sentence is determined using position as criteria like words at the beginning of a paragraph are considered important. His results showed that the best match between automatic and human-written abstracts was accomplished when sentence location, cue words and title words are considered.

2.1 Beyond Sentence Extraction

A method that involved the removal of irrelevant phrases from sentences extracted for summary was introduced in [5]. The first step involves the generation of a parse tree, followed by grammar checking so as to know which of the nodes of the tree can be deleted, it then checks the parts of the sentences that contains information relating to the main topic. After doing all the above it then removes the unnecessary parts of the sentences leaving behind a concise and coherent summary.

Motivated by the fact that automatic summarizers cannot always identify where the main gist of a document lies and the way text is generated is poor, [6] introduced a cut and paste method which involved six operations:

I. Sentence reduction, where unnecessary phrases are removed from the sentences,
II. Sentence combination, where sentences are combined together,
III. Syntactic transformation, involves rearrangement of words or phrases,
IV. Lexical paraphrasing, phrases are substituted with paraphrases,
V. Generalization and specification, substituting phrases with general/specific description,
VI. Reordering, rearrangement of the sentences extracted for summary.

3. MACHINE LEARNING METHODS

Various machine learning techniques have been exploited in automatic text summarization. Some of the techniques used include: Naïve-Bayes method, Rich Features and Decision Trees method, Hidden Markov model, Log-linear models and Neural Networks and Third Party Features.

3.1 Naïve-Bayes Method

Naïve-Bayes method was first used in [7] by using Bayesian classifier to determine if a sentence should be extracted or not. The system was able to learn from data. Some features used by their system include the presence of uppercase words, length of sentence, structure of phrase and position of words. The author assumed the following:

s = a certain sentence, S = the sentences in the summary, and F_1, , F_k = the features.

$$P(s \in \mathcal{S} \mid F_1, F_2, ..F_k) = \frac{\prod_{i=1}^{k} P(F_i \mid s \in \mathcal{S}) \cdot P(s \in \mathcal{S})}{\prod_{i=1}^{k} P(F_i)} \quad \text{-- (1)}$$

In equation 1, Sentences are scored based on these features and the formula is used to calculate the score, the highest ranking sentences are extracted.

Th naïve-bayes classifier was also used in DimSum [8], which used term frequency (tf) which is the number of times that a word appears in a sentences and inverse document frequency (idf) which is the number of sentences in which a word occurs, to know words that hold point at the key concepts of a document.

3.2 Rich Features and Decision Trees

Decision trees are powerful and popular tools for classification and prediction. It is a classifier in the form of a tree structure. The following nodes make up the tree:

- Decision node: specifies a test on a single attribute,
- Leaf node: indicates the value of the target attribute,
- Arc/edge: split of one attribute,
- Path: a disjunction of test to make the final decision

In [9], the authors concentrated on text position by making an effort to determine how sentence position affects the selection of sentences. The justification for the focus on position method is that texts are in a particular discourse structure, and that sentences containing ideas related to the topic of a document are always in specifiable locations (e.g. title, abstracts, etc). They also mentioned that discourse structure significantly varies over domains, so therefore the position method cannot be easily defined.

A sentence reduction algorithm that is based on decision tree was introduced in [10]. The algorithm proposed used semantic information to aid the process of sentence reduction and decision tree to handle the fact that the orders of original sentences change after they are reduced. They extended Knight and Marcu's sentence compression algorithm [11], which was also based on decision tree by adding semantic information to theirs. To achieve this, they used a Parser to parse the original sentences and by using WordNet, they enhanced the syntax tree gotten with semantic information.

3.3 Hidden Markov Models

A hidden Markov model is a tool for denoting probability distributions over sequences of observations. If we represent the observation at time *t* by the variable $Y_{t,}$ we assume that the observation are sampled at discrete, equally-spaced time intervals, so *t* can be an integer-valued time index. The two defining properties of hidden Markov model are: the assumption that the observation at time *t* was generated by some process whose state $S_{t\ is}$ hidden from the observer and the assumption that the state of the hidden process satisfies the Markov property i.e. given the value of $S_{t-1,}$ the current state S_t is independent of all the states prior to t-1 [12].

Two sentence reduction algorithms were proposed in [13]. Both were template-translation based which means that they don't need syntactic parser to represent the original sentences for reduction. One was founded on example-based machine-translation which does a good job of in the area of sentence reduction. On the other hand in specific cases, the computational complexity can be exponential. While the second one was an addition to the template-translation algorithm through the application of Hidden Markov model, the model employs the set of template rules that was learned from examples to overcome the problem of computational complexity.

2.4.4 Log-Linear Models

Log-Linear models are generally used in Natural Language processing. The flexibility of this model is its major benefit; it allows the use of rich set of features.

In [14], log-linear models were used to bring to null the assumption that existing systems were feature independent. Consequently, it was also showeshown empirically that using log-linear models produced better extracts than naïve-bayes model. The conditional log-linear model used by the author can be stated as follow:

$$P(c \mid s) = \frac{1}{Z(s)} \exp\left(\sum_i \lambda_i f_i(c, s)\right) \text{ …………..(2)}$$

Let c = label, s = item we want to label, f_i = i-th feature, λ_i = the corresponding feature weight and Z (s) = $\sum_c \exp(\sum_i \lambda_i f_i(c,s))$.

3.4 Neural Networks and Third Party Features

The automatic text summarization system developed in [15] had learning ability. This was done through combination of a statistical approach, extraction of keywords, neural network and unsupervised learning. The process used involved three steps: step one involved removal of stop words like "a" and stemming which is done by removing suffixes and prefixes to convert a word to its stem. Step two involves keywords are extracted by computing the matrix of the term frequency against the inverse document frequency, the most frequent terms listed are the keywords to be extracted for the summary. For the final step, the model checks for stop words again to be sure that no stop word is selected as keyword after which it selects sentences containing keywords to be added to the summary.

NetSum [16] was the first to use neural network ranking algorithm and third-party datasets for automatic text summarization. The authors trained a system that learned from a train set containing labels of best sentences from which features are extracted. From the train set, the system learns how features are distributed in the best sentences and it gives a result of ranked sentences for each document, the ranking is done using RAnkNet [17]

3. SENTENCE SCORING AND SENTENCE REDUCTION MODELS

Sentence score is a value that determines the sentences that are relevant to the input text. As shown in Figure 1, in our architecture, the input to the system is a single document. Sentence scoring occurs at the first stage; significant sentences are identified and extracted. The second stage involves the sentence reduction module; the extracted sentences from the sentence scoring module are processed, grammar checking and removal of target structures is done.

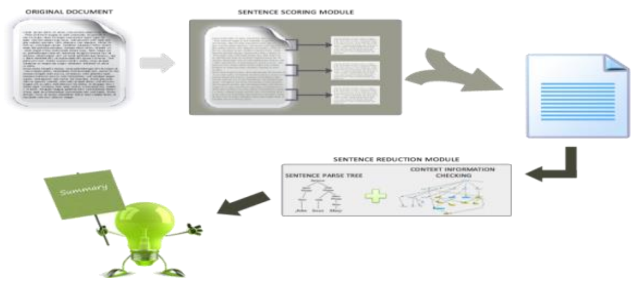


Figure 1: Text summarizer Architecture

3.1 Sentence Scoring Module

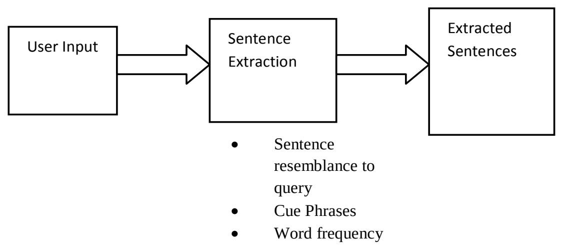


Figure 2: Sentence scoring module

In the sentence scoring module, there are two major steps involved:

1. **Preprocessing**:

This step involves the removal of stop-word and tokenization; stop-words are extremely common words (e.g. a, the, for). For this part, a stoplist which is a list of stop-words is used. Tokenization involves breaking the input document into sentences.

2. **Sentence scoring**: after the document has been broken into group of sentences. As seen in Figure 1 above, sentences are extracted based on three important features; sentence resemblance to query, cue phrases and word frequency.

- **Sentence resemblance to query:** This is modelled after sentence resemblance to title which calculates a score based on the similarity between a sentence and the title of a document. So sentence resemblance to query calculates a score based on the similarity between a sentence and the user query which means that any sentence that is similar to the query or includes words in the query are considered important. And the score will be calculated using the following formula:

$$\text{Score} = \frac{noOfQueryWordsInSentence}{nQW} \text{----(1)}$$

Where nQW = number of words in query

- **Cue Phrases**: the justification of using this feature is that the presence of some words likes "significantly", "Since" point to important gist in a document and a score is assigned to such sentences. The score is computed using:

$$\text{Score} = \frac{CuePhraseCount}{noOfTokensInSentence} \text{----------(2)}$$

- **Word frequency,** is a useful measurement of significance because it is revealed in [2] that an author tend to repeat certain words when trying to get a point across. So sentences that contain frequently occurring words are considered to be significant. The algorithm involves:

```
I.   Breaking sentence into tokens
II.  For each token, if the token already exists in array,
         Increment its count,
         Else add token to array and
     Set initial count to 1.
```

The boolean formula below is used to decide the sentences to be selected for further processing:

(SrqScore >= 0.5 || (CpScore >= 0.1 && WfScore >=3) ---(3)

Where SrqScore is Sentence resemblance to query Score, CpScore is Cue phrase score and WfScore is Word Frequency score.

3.2 Sentence Reduction Module

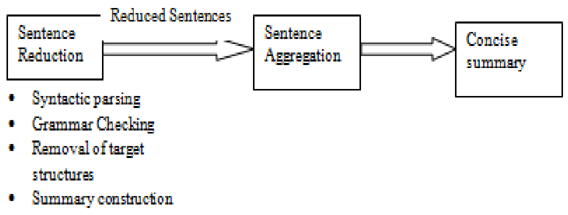


Figure 3: Sentence reduction module

In the sentence scoring module, the original document and the extracted sentences from the sentence scoring module is processed so as to remove irrelevant phrases from the document to make the summary concise, the sentence reduction algorithm is described in details in [18]. The processing involves:

Syntactic Parsing
Stanford parser, a syntactic parser is used to analyze the structure of the sentences in the document and a sentence parse tree is produced. The other stages involved in the sentence reduction module add more information to the parse tree, these information aids the final decision to be made.

Grammar checking
We go back and forth on the sentence parse tree, node by node to identify parts of the sentence are important and must not be removed to make the sentence grammatically correct. For example, in a sentence, the main verb, the subject, and the object(s) are essential if they exist.

Removal of Target Structures
For this research work we will be using the main clause algorithm for sentence reduction. In this algorithm, the main clause of a sentence is obtained and in that main clause we identify the target structures which are the structures to be removed and they are adjectives, adverbs, appositions, parenthetical phrase, and relative clauses. A reduced sentence is gotten after the targeted structures has been identified and removed from the sentence parse tree.

Summary Construction
We once again go back and forth on the sentence parse tree and see if the reduced sentences are grammatically correct. The reduced sentences are then merged together to give the final summary.
After the sentence reduction module carries out all four steps, a concise and coherent summary is expected as output.

4. IMPLEMENTATION

The system allows a user to input a document; which is then prepossessed and the sentences ranked. The high ranked sentences are then extracted for further processing. The result is then viewed by the user. As shown in figure 3, the flow of activities includes uploading of input file, preprocessing, assignment of scores, syntactic parsing, grammar checking, removal of target structures and display of output.

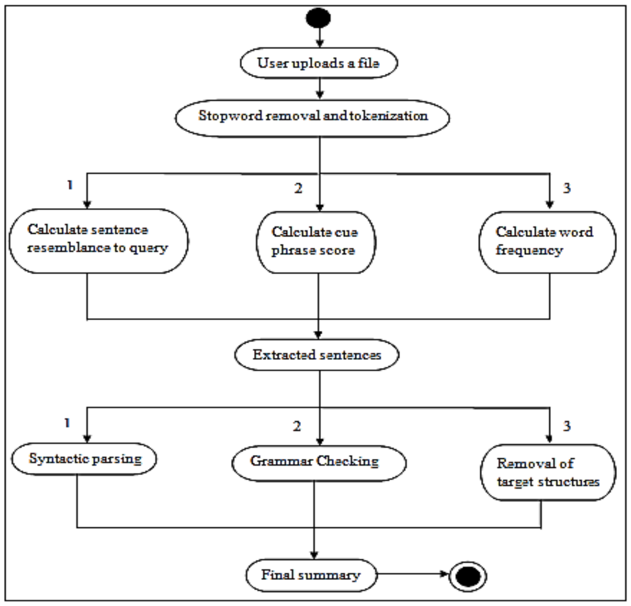

User uploads a file
Stopword removal and tokenization
1
2
3
Calculate sentence resemblance to query
Calculate cue phrase score
Calculate word frequency
Extracted sentences
1
2
3
Syntactic parsing
Grammar Checking
Removal of target structures
Final summary

Figure 3: Activity Flow

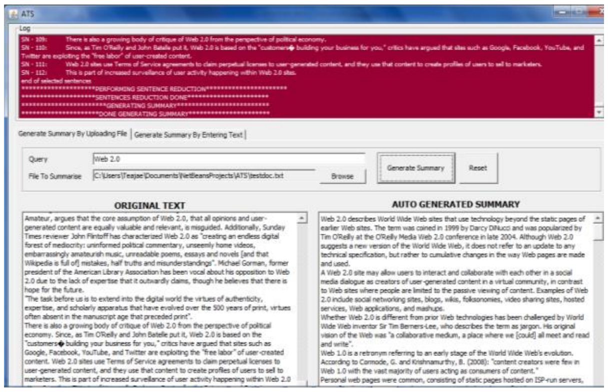
4.1 Implementation Resources

The following resources were used for the implementation of our summarization system:

- **Java:** it was considered a good choice for developing the summarization system because it makes more efficient use of memory needed for a system of this nature, it is faster and more effective, and also provides more data structures to handle the different data types used in the implementation of the proposed algorithm.
- **Stanford Parser:** The Stanford parser was used as the tagging tool in the sentence reduction module of our implementation. The Stanford parser analyses the sentences and provides us with the parts of speech of the words in a sentence, as well as the class e.g. adverbial phrase, adjectival phrases, etc., different parts of the sentence belongs to**.**
- **Gate (General Architecture for Text Engineering):** We used GATE in the development our algorithm to integrate the Stanford parser and its other modules such as the sentence splitter and tokenizer which properly handles complexities that may occur in long articles than ordinary tokenizers cannot handle properly. The integration of the parser and this other modules was used to create an application pipeline which was then used as a plugin in the implementation and development of our summarisation system.

5. RESULTS AND DISCUSSION

To test our summarisation system, we obtained random articles online from Wikipedia to use as our input document. The article is then saved in a text (.txt) file to be used in our system. Wikipedia references are disregarded during our extraction and not included in the content of the article to use as input document for our summarisation system, as they do not provide any value of importance to the overall article and the summary we want to generate. For this task, we selected an article about Web 2.0, however, it is important to note that any article could be selected and used.



```
SN - 109:      There is also a growing body of critique of Web 2.0 from the perspective of political economy.
SN - 110:      Since, as Tim O'Reilly and John Batelle put it, Web 2.0 is based on the "customers� building your business for you," critics have argued that sites such as Google, Facebook, YouTube, and Twitter are exploiting the "free labor" of user-created content.
SN - 111:      Web 2.0 sites use Terms of Service agreements to claim perpetual licenses to user-generated content, and they use that content to create profiles of users to sell to marketers.
SN - 112:      This is part of increased surveillance of user activity happening within Web 2.0 sites.
end of selected sentences
********************PERFORMING SENTENCE REDUCTION**********************
********************SENTENCES REDUCTION DONE**********************
**********************GENERATING SUMMARY************************
********************DONE GENERATING SUMMARY**********************
```

Query: Web 2.0

File To Summarise: C:\Users\Teajae\Documents\NetBeansProjects\ATS\testdoc.txt

ORIGINAL TEXT:

Amateur, argues that the core assumption of Web 2.0, that all opinions and user-generated content are equally valuable and relevant, is misguided. Additionally, Sunday Times reviewer John Flintoff has characterized Web 2.0 as "creating an endless digital forest of mediocrity: uninformed political commentary, unseemly home videos, embarrassingly amateurish music, unreadable poems, essays and novels [and that Wikipedia is full of] mistakes, half truths and misunderstandings". Michael Gorman, former president of the American Library Association has been vocal about his opposition to Web 2.0 due to the lack of expertise that it outwardly claims, though he believes that there is hope for the future.
"The task before us is to extend into the digital world the virtues of authenticity, expertise, and scholarly apparatus that have evolved over the 500 years of print, virtues often absent in the manuscript age that preceded print".
There is also a growing body of critique of Web 2.0 from the perspective of political economy. Since, as Tim O'Reilly and John Batelle put it, Web 2.0 is based on the "customers� building your business for you," critics have argued that sites such as Google, Facebook, YouTube, and Twitter are exploiting the "free labor" of user-created content. Web 2.0 sites use Terms of Service agreements to claim perpetual licenses to user-generated content, and they use that content to create profiles of users to sell to marketers. This is part of increased surveillance of user activity happening within Web 2.0

AUTO GENERATED SUMMARY:

Web 2.0 describes World Wide Web sites that use technology beyond the static pages of earlier Web sites. The term was coined in 1999 by Darcy DiNucci and was popularized by Tim O'Reilly at the O'Reilly Media Web 2.0 conference in late 2004. Although Web 2.0 suggests a new version of the World Wide Web, it does not refer to an update to any technical specification, but rather to cumulative changes in the way Web pages are made and used.
A Web 2.0 site may allow users to interact and collaborate with each other in a social media dialogue as creators of user-generated content in a virtual community, in contrast to Web sites where people are limited to the passive viewing of content. Examples of Web 2.0 include social networking sites, blogs, wikis, folksonomies, video sharing sites, hosted services, Web applications, and mashups.
Whether Web 2.0 is different from prior Web technologies has been challenged by World Wide Web inventor Sir Tim Berners-Lee, who describes the term as jargon. His original vision of the Web was "a collaborative medium, a place where we [could] all meet and read and write".
Web 1.0 is a retronym referring to an early stage of the World Wide Web's evolution. According to Cormode, G. and Krishnamurthy, B. (2008): "content creators were few in Web 1.0 with the vast majority of users acting as consumers of content."
Personal web pages were common, consisting of static pages hosted on ISP-run servers,

Figure 4: GUI interface of our summarization system.

We evaluated our summarization system using 3 standard parameters; (1) precision (the fraction of the returned summary that is relevant to the query posed by the user); (2) recall (the fraction of the relevant information contained in the document that was returned by the summarization system); and (3) F-measure (the weighted harmonic mean of precision and recall). We had a human subject read two articles, one was the web 2.0 article and the other was a Blackberry passport review.

The web 2.0 article had 179 sentences, the summary presented by the subject contained 110 sentences and our system produced a summary containing 112 sentences. 92 sentences were present in both the summary made by the subject and the summary made by our system.

The second document, a blackberry passport review contained 129 sentences. In the summary presented by the subject we had 40 sentences and our summarizers' summary, we had 57 sentences. 27 sentences were present in both the summary made by the subject and the summary made by our system. Based on comparison with the well-known Copernic summarizer which produces summary based on statistical and linguistic methods and the FreeSummarizer that produces summary based on specified number of sentences, our system gave high recall values of 83.6% and 67.5% respectively; an indication that the sentences in our system's summary contain more relevant information such that selected sentences are relevant to the query pos. ed by the user.
In conclusion, this work can be extended to multi-document summarization whereby the source text is more than one; the resulting summary may contain a larger amount of information when compared to a single document summarization. Also, it can be extended to other languages apart from English language.

7. REFERENCES

[1] Wells, M. (2009). Advantages of automatic text summarization, http://ezinearticles.com/?3-Advantages-of-Automatic-Text-Summarization&id=3465270 downloaded on 11 September, 2014.

[2] Luhn, H. P. (1958). The automatic creation of literature abstracts, IBM Journal of research and development, Vol. 2, No. 2, pp. 159-165.

[3] Baxendale, P. (1958). Machine-made index for technical literature - an experiment. IBM Journal of Research Development, 2(4):354–361

[4] Edmundson, H. P. (1969). New methods in automatic extracting. Journal of the ACM, 16(2):264–285. [2, 3, 4]

[5] Jing, H. (2000). Sentence Reduction for Automatic Text Summarization, In Proceedings of the Sixth Applied Natural Language Processing Conference, pp. 310-315. Association for Computational Linguistics.

[6] Jing, H. and McKeown, K.R. (2000). Cut and Paste Based Text Summarization, In Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics, pp. 178-185. Association for Computational Linguistics.

[7] Kupiec, J., Pedersen, J., and Chen, F. (1995). A trainable document summarizer, In Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 68-73. ACM.

[8] Larsen, B. (1999). A trainable summarizer with knowledge acquired from robust NLP techniques. Advances in Automatic Text Summarization, pp. 71.

[9] Lin, C. Y., and Hovy, E. (1997). Identifying topics by position, In Proceedings of the fifth conference on Applied natural language processing, pp. 283-290. Association for Computational Linguistics

[10] Le, N. M., & Horiguchi, S. (2003). A new sentence reduction based on decision tree model, In Proceedings of the 17th Pacific Asia Conference on Language, Information and Computation, pp. 290-297.

[11] Knight, K., and Marcu, D. (2002). Summarization beyond sentence extraction: A probabilistic approach to sentence compression, Artificial Intelligence, Vol. 139, No. 1, pp. 91-107.

[12] Ghahramani, Z. (2001). An Introduction to Hidden Markov Models and Bayesian Networks, International Journal of Pattern Recognition and Artificial Intelligence, Vol 15, No. 1, pp. 9-42.

[13] Nguyen, M. L., Horiguchi, S., Shimazu, A., & Ho, B. T. (2004). Example-based sentence reduction using the hidden markov model, ACM Transactions on Asian Language Information Processing (TALIP), Vol. 3, No. 2, pp. 146-158.

[14] Osborne, M. (2002). Using maximum entropy for sentence extraction, In Proceedings of the ACL-02 Workshop on Automatic Summarization, Vol. 4, pp. 1-8. Association for Computational Linguistics.

[15] Yong, S. P., Abidin, A. I., & Chen, Y. Y. (2005). A Neural Based Text Summarization System, In Proceedings of the 6th International Conference of DATA MINING, pp. 45-50

[16] Svore, K. M., Vanderwende, L., & Burges, C. J. (2007). Enhancing Single-Document Summarization by Combining RankNet and Third-Party Sources, In EMNLP-CoNLL, pp 448–457.

[17] Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., & Hullender, G. (2005). Learning to rank using gradient descent. In Proceedings of the 22nd international conference on Machine learning (pp. 89-96). ACM.

[18] Silveira, S. B., and Branco, A. (2014). Sentence Reduction Algorithms to Improve Multi-document