A GRAPH-BASED APPROACH TOWARDS AUTOMATIC TEXT SUMMARIZATION

by

Qandeel Fatima

Bachelors of Electrical Engineering, Center for Advanced Studies in Engineering 2011

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN
COMPUTER SCIENCE

UNIVERSITY OF NORTHERN BRITISH COLUMBIA

December 2017

Abstract

Due to an exponential increase in number of electronic documents and easy access to information on the Internet, the need for text summarization has become obvious. An ideal summary contains important parts of the original document, eliminates redundant information and can be generated from single or multiple documents. There are several online text summarizers but they have limited accessibility and generate somewhat incoherent summaries.

We have proposed a Graph-based Automatic Summarizer (GAUTOSUMM), which consists of a pre-processing module, control features and a post-processing module. For evaluation, two datasets, Opinosis and DUC 2007 are used and generated summaries are evaluated using ROUGE metrics. The results show that GAUTOSUMM outperforms the online text summarizers in eight out of ten topics both in terms of the summary quality and time performance. A user interface has also been built to collect the original text and the desired number of sentences in the summary.

Table of Contents

Abstract ii
List of Tables v
List of Figures vi
Acknowledgement ix
1. Introduction 1
1.1 Extractive Summarization 5
1.2 Types of Summaries 6
1.3 Graph-based approach 8
1.4 Methodology 9
1.5 Contributions 11
1.6 Organization of Thesis 12
2. Literature Review 13
2.1 Machine Learning Approach 15
2.1.1 Naïve Bayes Method 17
2.1.2 Rich Features and Decision Trees 18
2.1.3 Hidden Markov Model 18
2.1.4 Log-Linear Method 19
2.1.5 Neural Networks 19
2.1.6 Summary of Machine Learning Approach 20
2.2 Clustering Based Approach 21
2.2.1 Summary of Clustering Based Approach 24
2.3 Lexical Chaining Approach 24
2.3.1 Summary of Lexical Chaining Approach 27
2.4 Frequent Term Approach 27
2.4.1 Summary of Frequent Term Approach 30
2.5 Information Retrieval Approach 31
2.5.1 Summary of Information Retrieval Approach 32
2.6 Graph-based approach 32
2.6.1 Summary of Graph-based approach 35
3. Design and Implementation 37
3.1 LexRank 37

3.2 Proposed Method 41
3.2.1 Pre-processing Module 44
3.2.2 Jaccard Distance 47
3.2.3 Control Features 49
3.2.4 Interface of GAUTOSUMM and Post-processing Module 50
3.3 Benefits of the Proposed Method 51
3.4 Limitations of the Proposed Method 54
4. Evaluation and Results 57
4.1 ROUGE 58
4.2 Datasets 59
4.3 Online Text Summarizers 61
4.4 Results Using Opinosis Dataset 63
4.5 Evaluation and Results Using DUC Dataset 68
4.5.1 Document Length 72
4.5.2 Time Performance 84
4.5.3 Impact of pre-processing 87
4.5.4 Complexity of GAUTOSUMM 93
Conclusion and Future Direction 97
5.1 Future Work 99
Appendix 1 101
Bibliography 102

List of Tables

Table 1: Adjacency matrix used by Radev and Erkan in LexRank [66] 40

Table 2: Sentence Similarity Matrix 43

Table 3: ROUGE evaluation metrics from SMMRY (sample from Opinosis dataset) 63

Table 4: Comparison of ROUGE evaluation metrics for online summarizers and GAUTOSUMM (without pre-processing) 64

Table 5: Comparison of ROUGE evaluation metrics for online summarizers and GAUTOSUMM (extensive pre-processing) 67

Table 6: Comparison of ROUGE evaluation metrics for online summarizers and GAUTOSUMM (basic pre-processing) 68

Table 7: Impact of pre-processing on number of edges for GAUTOSUMM 90

Table 8: Number of Edges and Sentences of samples together with execution time of GAUTOSUMM. 92

List of Figures

Figure 1: Graph-based approach showing different topics covered in a document [7].....33

Figure 2: Graphical representation of LexRank [4].....39

Figure 3: Selected sentences [65] for graphical representation of LexRank in Figure 2 39

Figure 4: Proposed Method 42

Figure 5: Sum vector and Sorted Sum.....43

Figure 6: Illustration of proposed method (Without pre-processing).....44

Figure 7: Illustration of proposed method (With pre-processing).....46

Figure 8: Generated output from GAUTOSUMM with post-processing.....51

Figure 9: An example to illustrate True and False edges 53

Figure 10: F-score for online text summarizers and GAUTOSUMM for 250 words output..70

Figure 11: F-score for online text summarizers and GAUTOSUMM for 5% output 71

Figure 12: Size of samples with respect to increasing number of sentences.....72

Figure 13: Comparison of F-score (250 words output) with increasing number of sentences: 210 - 365.....73

Figure 14: Comparison of F-score (250 words output) with increasing number of sentences: 374 - 542.....74

Figure 15: Comparison of F-score (250 words output) with increasing number of sentences: 548 - 786.....75

Figure 16: Comparison of F-score (250 words output) with increasing number of sentences: 799 - 962.....76

Figure 17: Comparison of F-score (250 words output) with increasing number of sentences: 1029 - 1693 77
Figure 18: Comparison of F-score (5% output) with increasing number of sentences: 210 - 365 79
Figure 19: Comparison of F-score (5% output) with increasing number of sentences: 374 - 542 80
Figure 20: Comparison of F-score (5% output) with increasing number of sentences: 548 - 786 81
Figure 21: Comparison of F-score (5% output) with increasing number of sentences: 799 - 962 82
Figure 22: Comparison of F-score (5% output) with increasing number of sentences: 1029 - 1693 83
Figure 23: F-score showing outliers of online summarizers and GAUTOSUMM for 250 words 85
Figure 24: Comparison of time performance for GAUTOSUMM 86
Figure 25: Time Performance of online text summarizers and GAUTOSUMM 87
Figure 26: Impact of pre-processing on execution time with increasing number of sentences: 210 - 480 88
Figure 27: Impact of pre-processing on execution time with increasing number of sentences: 482 – 811 89
Figure 28: Impact of pre-processing on execution time with increasing number of sentences: 849 – 1693 89
Figure 29: Execution time comparison for three levels of pre-processing: Extensive pre-processing, Basic pre-processing and No Pre-processing 91

Figure 30: Number of Sentences and Edges for samples from DUC 2007 dataset.....93

Figure 31: Complexity of GAUTOSUMM 94

Figure 32: Complexity of GAUTOSUMM (Sentences Vs Time).....94

Figure 33: Number of Sentences Vs Number of Edges with basic pre-processing.....95

Figure 34: Number of Sentences Vs Number of Edges with no pre-processing.....96

Acknowledgement

Firstly, I would like to thank Dr. Waqar Haque for his esteemed guidance, motivation and immense knowledge. His expertise and support have been very helpful in completing my thesis. Throughout this research, he guided me in the right direction to overcome many difficult situations. Without his constant support and generosity, my thesis would not have been possible.

I also would like to thank my supervisory committee, Dr. David Casperson and Dr. Reza Chowdhury for their support and direction in my research. I would like to thank my colleagues and friends for their support and help.

Last but not the least, I would like to thank my parents and siblings for having faith in me and always encouraging to pursue dreams in my life.

Chapter 1

1. Introduction

In today's busy world, access to compressed and meaningful information extracted from large volumes of data is very important and necessary. We often find ourselves surrounded by a huge quantity of data and come across information that needs to be summarized eloquently, regardless of whether we search for topics related to business, sports, politics, medicine, religion, or science. Big companies generate internal reports related to their business on an annual, monthly, weekly and daily basis. Business owners or senior executives want these reports in a compressed form which is meaningful to them and saves time and space. Similarly, search engines provide access to massive volumes of information based on keywords, but this information still needs to be sifted through by the user. However, sometimes we are presented with summaries that we often take for granted. For instance, we cannot have a newspaper without headlines, or books and movies without reviews and trailers, scholarly articles without abstracts, or search results without summarized extracts from each page. In general, summaries are of great potential value to CEOs, journalists, lawyers, students, researchers and casual browsers of the Internet.

Summaries are generated by text summarizers to reduce the length of the original document in a way that highlights important contents of the document. This process should be

automatic, i.e. it must be generated by a machine to extract the most significant information in a shorter form. A good summary should preserve the principal semantic content and help the user to quickly understand large volumes of information. Radev [1] has formally defined a summary as "a text that is produced from one or more texts, that conveys important information in the original text(s), and that is no longer than half of the original text(s) and usually significantly less than that". Using work from other researchers, we can simplify this definition as:

- Generation of summaries from single or multiple documents.
- Preservation of important parts which are present in the original text.
- Reduction of the original text, or elimination of redundant parts from the original text.

In 1958 Luhn [2], a computer scientist at IBM was likely the first person who formally acknowledged the need for automatically generated summaries and abstracts. Though at that time there were no modern computers or computing devices, he suggested a system that could take full advantage of the capabilities of modern electronic data processing. He generated abstracts of technical literature and magazine articles to facilitate quick and precise topic identification of the document. This work motivated Edmundson to publish a book in 1969 [3], in which he proposed new methods for automatic extraction of information. The concept of automatic text summarization evolved at that time, which demonstrated that summaries can be generated by computers without any human supervision.

Over the past 50 years, researchers have been trying to develop a generic mechanism, which can compress large documents into meaningful summaries. Recently, text summarization has gained more importance due to an exponential increase of information on

the Internet. Because of easy access to the Internet, it is very difficult for users to narrow down their query to a specific topic because of a large amount of inter-related information available online. This problem has led to the development of methods and algorithms for automatic text summarization, which can be implemented with or without human supervision. Several methods and techniques have been proposed for text summarization, but there is still a need for an efficient generic text summarizer which can generate summaries on any topic and in any language. To date, text summarization remains a big challenge for researchers.

Depending on the methods or techniques, text summarization is divided into two broad categories – extractive and abstractive. *Extractive summaries* [4] [5] are generated by taking out the exact sentences or keywords from the original document that are important. These sentences are then combined into a shorter form to produce a meaningful and coherent summary. Extractive summaries are usually based on statistical analysis of individual or mixed surface level features such as word or phrase frequency, location, and cue words to locate the sentences to be extracted [6] [7]. Most of the research today focuses on extractive summarization.

Abstractive Summaries [7] [4] are generated by first understanding the main concept of a document and then stating them in a clear, simple language. It aims to produce the main concept of the document and important material in a new way. Abstraction is essentially a replacement of the original text while keeping the same theme as of original document. It uses linguistic methods to examine and interpret a text document. Abstractive summaries are difficult to produce because current computing devices cannot generate semantic representation, inference, or natural language to a level that is equivalent to humans. Summaries produced by humans are not extractive; instead, we make or write summaries

depending on our background knowledge, attitude or disposition. Humans have their own interpretation of ideas [2] and the quality of a summary may vary widely among them. In fact, a person may generate two different summaries of the same article at different times in their lives depending on a variety of external factors.

There are several problems associated with both extractive and abstractive summaries. The main problem with the former is that longer sentences are often selected which results in potential inclusion of less important parts in the generated summary. Longer sentences have a higher probability to be included in the generated summary because they may have more important words than shorter sentences. Important information within a document is usually distributed in different parts of the original document and extractive summaries cannot capture this information in scenarios where multiple themes are introduced. Summaries with longer sentences can eliminate this problem to some extent, but then the summaries may include redundant information. Representation of conflicting arguments in the generated summary is also a big challenge for both abstractive and extractive summaries [7]. For example, an article on advantages and disadvantages of a product or a comparative study on a topic may include arguments in favor of the topic and against the topic. For extractive summaries, it is very difficult to present conflicting arguments correctly because decision making by summarizer and subsequent identification of the related argument is necessary. Extraction of sentences from the document can lead to lack of coherence in the generated summary. Additionally, sentences may contain pronouns, which lose their reference when extracted out of the context. This problem is more intense in multi-document summarization but can be avoided by replacing the pronouns with their antecedents [7]. On the other hand, abstractive summaries have a representation problem because they involve reformulation of contents, a process which

makes it domain dependent and requires human supervision [8]. The biggest challenge for abstractive summarization is that the capabilities of current computing devices are limited because they cannot generate rich representations of a language. To first understand the natural language and then produce the summary by connecting words and sentences in a similar way a human would, is beyond the ability of today's technology. Abstractive text summarizers provide an overview of the topic rather than going into the details contained within the original document. Thus, abstractive summaries do not have high quality in terms of evaluation, and they can deviate from the topic of the document very easily. Simplicity, speed, domain independence, and non-requirement of background knowledge [8] are the reasons that we opted to focus on extractive text summarization.

1.1 Extractive Summarization

Extractive text summarization consists of two sub-methods – Fusion and Compression [1]. As the name indicates, fusion [1] coherently combines extracted parts of the original text, whereas compression excludes the unimportant sections of the text. The process of extractive text summarization is divided into two steps [7]: pre-processing and processing. Pre-processing [7] step is the removal of determiners (a, an, the, this, that, these, those), auxiliary verbs (have, many, can, shall, be), prepositions (at, in, on, under, over, of) and/or conjunctions (and, but, or). This step includes the identification of sentence boundaries and it may include stemming - retaining the stem or radix of each word. Pre-processing is a very important step in summarization because it eliminates insignificant words that occur most frequently and thus, can affect the calculation of statistical features such as sentence length and cue words or

phrases. The processing step may include one or more of the summarization approaches such as graph-based approach, lexical chains, or a frequent term approach to obtain an absolute summary [9] (these approaches are discussed further in Chapter 2). This step mainly includes document analysis of the original text, statistical selection of significant sentences, sentence weighting, and sentence extraction as the last step.

1.2 Types of Summaries

Text summaries can also be classified as: indicative, informative, generic, query based and user-focused. Indicative summaries [10] provide reduced information on the main topic of a document. These summaries usually preserve the most important passages and help users to decide whether the original document is worth reading or not. The purpose of these summaries is to focus on a problem, event or topic. Informative summaries [10] are intended to cover most of the topics in the original document. These summaries are more likely to provide information about the article and they are longer in length as compared to the indicative summaries. Generic summaries [10] [8] are intended to convey the author's perspective, which also represents the theme of the document. These summaries inform the broad community of readers and include a general idea of the original document. Query-based summaries [10] are generated when the result is based on a question, e.g. "what is text summarization?" They are also known as user-focused summaries because they depend on the input from the user. These summaries are produced depending on the interest of a particular user and focus on the topics specified by the user.

Summaries can be generated from a single document as well as from multiple documents, known as single document summarization and multi-document summarization, respectively. Text summarization can be implemented for monolingual documents, as well as for multilingual documents [11]. For monolingual summarization, the input and the output summary is in the same language, but for multilingual summarization, input or original text is in a different language than the target language of the final output summary.

Another important aspect of text summarization is the evaluation of generated summary [7]. Generally, summaries are evaluated using two measures: Intrinsic and Extrinsic [7]. Intrinsic measures are used to evaluate the quality of a summary in terms of linguistics, non-redundancy and reference clarity. Intrinsic measures also check for precision, recall and F-score used by evaluation software (discussed in Chapter 4). Precision determines how precise is the generated summary, recall measures the flow and structure, and F-score represents harmonic mean of precision and recall. Extrinsic measures are used to determine the quality of a summary by following a specified task based performance metric such as document categorization, question answering and information retrieval [12].

For text summarization, there are many approaches which can be used to generate a summary for the given document, for example, machine learning approach, clustering based approach, lexical chaining approach, frequent term approach, information retrieval approach and Graph-based approach (discussed in Chapter 2). In our research, we are using extractive summarization and Graph-based approach to find the most important sentences within the document. Graph-based approach is used in many fields such as computer science, biology, social sciences and information systems. In a Graph-based approach, a relation between nodes and related edges is defined by a similarity measure. The generated graph can be directed or

undirected and the edges can be specified by a weight. This approach is further discussed in the next section.

1.3 Graph-based approach

In this approach, each sentence of a document is represented as a node. A node can be connected to another node through an edge when there is a similarity between them [4]. The number of edges between the nodes (or weight of the edges) depends on the similarity measure. Several statistical features such as cosine similarity, cohesion and discounting (further explained in Chapter 3) are used to calculate similarity measure. The number of edges between sentences are determined to find the nodes with most edges drawn from, and to, other nodes. The sentences (nodes) with the highest number of edges are considered as the most significant sentences. To present this concept in terms of an algorithm, a matrix is generated which contains the number of edges between the nodes. The row-wise (or column-wise) addition of this matrix generates a sum vector that recognizes rank of each sentence. Sum vectors are sorted in a descending order to obtain the most significant sentences.

One of the main reasons for using a Graph-based approach is that the graph of the original document provides a clear visualization of the nodes which helps in locating the important sentences. The selection of salient sentences in a graph depends on the edges drawn from and to other nodes, which creates a democratic system within the graph [13]. A democratic system is like casting a vote for a representative node, where a node recommends a number of other nodes which may be selected for the generated summaries. Another reason for choosing a Graph-based approach is that it does not need to go through a training process such as machine

learning approaches or neural networks. The training process requires a significant amount of time to feed in all the available information to the algorithm about given topics. This procedure needs extensive datasets to be processed and learned by the algorithm, but still the algorithm may not be able to cover all the topics to generate the required summary. On the other hand, a Graph-based approach can produce summaries in real time, without the help of any trained templates. It does not include complex equations and calculations, but still provide a clear representation of sentences in a document. Graph-based approach has been used by companies like Facebook and Google [14] to provide a better representation of data to users. The key feature of a Graph-based method is to provide a clear representation of nodes, edges and to locate the important nodes by their relations through connecting edges. Such an approach provides advantages to businesses through a "better customer experience, targeted content and increased revenue opportunities" [14]. There are some other areas where a Graph-based approach is valuable such as "customer support portals, content portals, product catalogs and social networks" [14].

1.4 Methodology

We have adopted the Graph-based approach which was originally proposed by Radev and Erkan in 2004 [4]. Since then, many improvements have been proposed by other authors to enhance this approach. As Graph-based approach mainly depends on nodes and edges, Radev and Erkan used a cosine similarity equation [4] as a similarity measure to define a relationship between the nodes and edges. This equation involves calculation of variables such as centroids and sentence centrality. A centroid is a file which contains *tf-idf* scores of words above a

predefined threshold [4], where *tf* is the term frequency within the document(s) and *idf* is inverse document frequency. *idf* is a logarithmic function which ignores the words that occur frequently, in addition to eliminating prepositions or articles. The sentences that contain more words from the centroid are considered as central and determine the sentence centrality. The degree of sentences is also calculated by using a threshold method so that sentences with more significant similarities are selected [4]. A sum vector is calculated by the addition of rows or columns of the adjacency matrix to determine the most significant sentences.

In the proposed summarizer (GAUTOSUMM – Graph-based Automatic Summarizer), the Graph-based approach is implemented in a different way. We first identify important words within the sentences and then calculate their frequency to determine a score for each sentence. In this way, redundant frequency calculations are avoided and only important words are considered for calculations of the score. The other methods like LexRank [4], use *tf-idf* for the calculation of cosine similarity and calculate term frequencies (*tf*) for the whole document. In our method, *tf's* are calculated for important words only. With this module, we get a part of *idf* by eliminating the frequently occurring words like articles and prepositions. By using a pre-processing module, our proposed summarizer does not exclude frequently occurring terms other than articles, prepositions, conjunctions, determiners and auxiliary verbs, which can play an important role in the selection of good quality summary. Another important consideration in GAUTOSUMM is that the comparisons are made across the sentences (i.e. from one sentence to the other), but not within the sentence itself. The comparisons within a sentence introduce calculation of false edges and may include redundant information in the generated summary.

To measure the quality of generated summaries for GAUTOSUMM and online text summarizers, an evaluation software, ROUGE, is used. The values of precision, recall and harmonic mean show that summaries generated by GAUTOSUMM are superior than online text summarizers in terms of quality, readability and accuracy in eight out of ten topics. These results are discussed in Chapter 4.

1.5 Contributions

The contributions of this research consist of the following:

- Our method includes calculation of edges in two ways:
    - By comparisons of edges between the sentences.
    - By Jaccard distance [15], which helps in the calculation of similarities and dissimilarities between the sentences by using sentence lengths. Jaccard distance is modified in a way to calculate the true number of edges between the sentences so that longer sentences do not affect the generated summary.
- Pre-processing module is used instead of *idf*. Results of precision, recall and F-score are calculated with pre-processing, without pre-processing and with extensive pre-processing. The results showed that GAUTOSUMM generated good quality summaries with pre-processing and extensive pre-processing. Further, these summaries are better than existing online text summarizer's summaries.
- Control features are added to avoid the problem of longer sentence selection in the generated summary. These control features are dependent on sentence lengths and

are combined with Jaccard distance to produce a good quality summary with better flow and structure. The following control features are added to the algorithm:

- A function which selects an optimized range of sentences from the document. The range of selected sentences is dependent on the size of the document and sentence lengths, which may differ for each sample.
- The length of selected sentences for the generated summary can be controlled. Too short and too long sentences are eliminated from the summary to make it more precise and to provide a better flow. This is implemented via a threshold which can be adjusted by the topic or user.

1.6 Organization of Thesis

This thesis consists of five chapters. In chapter 1, we introduced the background of text summarization and discussed the details of extractive text summarization, its comparison with abstractive text summarization, and pros and cons of both methods. Chapter 2 provides a literature review and history of text summarization. It describes different approaches and methods proposed by researchers. Chapter 3 presents a discussion on Graph-based approach and proposed method used by the summarizer together with its benefits and limitations. The evaluation and results for GAUTOSUMM are provided in chapter 4. It includes a comparison of ROUGE metric results and time performance with online text summarizers. Chapter 5 concludes the work and provides direction for future research.

Chapter 2

2. Literature Review

With the exponential growth of online available information and accessibility to the Internet, text summarization has become a necessity. In this chapter, a background of text summarization together with an overview of the approaches adopted by researchers is presented. The early experimentation of automatic text summarization began in 1950's by Luhn [16]. Since then, many techniques have been proposed. As our research is focused on extractive text summarization and due to the extensive amount of research done in this area, we will discuss extractive approaches.

One of the most cited papers on text summarization is by Luhn [16], which describes research carried out at IBM. The main idea presented in this work was that the frequency with which a word occurs in an article provides a useful measure of its importance within that article. Other ideas presented in the paper include word stemming to their root form and deletion of stop words. He collected a list of "content words" which are sorted by decreasing frequency and marked these words with an index to provide a significance measure of the word. A "significance factor" is then calculated that reflects the number of occurrences of important words within a sentence. Finally, the score of all sentences is calculated and ranked in order of their significance factor.

Another related work published in 1958 by Baxendale [17] describes the importance of statistical features of sentences, such as "sentence position", which can help to identify the salient sentences in the document. Towards this effort, the author examined 200 paragraphs and concluded that in more than 80% of the paragraphs, the topic sentence is the first sentence and in 7% paragraphs it is the last sentence [17]. Sentence position feature can be an accurate way to select a topic sentence from the first or last sentence of a paragraph. This positional feature is a basic way of finding important sentences, and is used in many complex text summarization algorithms including machine learning approaches (which will be addressed in next section). In 1969, extractive text summarization was introduced by Edmundson [18] which describes a system that produces document extracts. His main contribution was the development of a structure for an extractive summarization experiment [18]. As a first step, Edmundson created a protocol for generating manual extracts which he used as a reference comparison standard for a set of 400 technical documents. From previous work, word occurrence and positional importance were incorporated along with two new features: cue words (presence of important words such as "significant" or "conclusion") and the overall skeleton (to identify sentences or words as title or heading). The weights were then attached to each of these features to calculate a score of the sentences.

The approaches used for text summarizers can be grouped into six groups depending upon the characteristics of the algorithm implemented within the summarizer. These include Machine Learning Approach, Clustering Based Approach, Lexical Chaining Approach, Frequent Term Approach, Information Retrieval Approach and Graph-based approach. A comparison of readability, redundancy and similarity for all approaches is shown in Appendix 1. These approaches and related summarizers are discussed in the next section.

2.1 Machine Learning Approach

Several Machine Learning approaches [7] [19] [11] developed in 1990s have been used for text summarization. These include Naïve Bayes Methods, Rich Features and Decision Trees, Hidden Markov Models, Log-Linear Models and Neural Networks. A Naïve Bayes Method [20] calculates Naïve Bayes classifier that categorizes each sentence as worthy of extraction or not. Rich features and decision trees use a position method [21] which arises from the idea that texts generally follow a predictable discourse structure, and that the sentences of greater topic centrality tend to occur at certain specifiable locations (e.g. title, abstracts, etc.) [19]. Hidden Markov Model (HMM) is a sequential method to define local dependencies between the sentences [22]. It uses three features: position of the sentence in the document (built into the state structure of the HMM), number of terms in the sentence, and likeliness of the sentence terms given the document terms. Log-Linear Model includes an exponential method that has two labels: sentence to be extracted for a summary or not [23]. In neural networks, document sentences which contain keywords used in news search engine or entities found in Wikipedia articles present a greater chance of having those sentences in the highlight for the generated output summary [19].

In 1998, SUMMARIST was proposed by Hovy and Lin [24]. This summarizer provides single document text summarization in the field of "news" and is also a multi-lingual system. SUMMARIST used Natural Language Processing (NLP) techniques combined with symbolic concept-level world knowledge. The process of text summarization in SUMMARIST is divided into three steps: topic identification, interpretation and generation of output summary. COLUMBIA MDS [25] was introduced in 2002 which is a multi-document summarizer in the

field of “news” and generates both extractive and abstractive summaries. It uses statistical techniques and is a composite system which uses different algorithms from other summarizers. For example, depending on the input, it uses a module known as MultiGen for single document input and DEMS (Dissimilarity Engine for multi-document Summarization) for multiple documents or biographical documents. The evaluation scores of the summarizer showed that some improvements are needed to enhance the output summary generated by DEMS. Coherence is one of the main improvements mentioned by the author because DEMS is multi-document summarizer which may generate a summary that is not coherent and well connected. Another problem faced by DEMS was lack of training data to cover a diverse range of topics. The same authors evaluated DUC 2003 dataset [26] with the suggested improvements which showed better results but, still encountered by lack of training data for the summarizer. NTT [27] is a single document text summarizer which employs Support Vector Machine (SVM). SVM is a Machine Learning technique to classify a sentence as relevant or non-relevant. Some of the control features used by NTT are: position, length, weight, similarity with headline and presence of verbs and prepositions. The evaluation and results of NTT showed that it generated better quality summaries in most of the topics but for some topics, generated summaries lacked readability, grammar, cohesion and organization. To overcome these problems, Named Entities, Modalities and Rhetorical Relations are the suggested improvements by the authors. Karamuftuoglu [28] proposed another algorithm for single document extractive summaries. It uses a pattern matching method for lexical links and bonds with the implementation of SVM. It is mainly based on extract-reduce-organize paradigm [28]. The algorithm tends to pick sentences which appear earlier in a document and it may select consecutive sentences which introduce lack of coherence and flow in the generated summary. Lal & Rueger [29] introduced single-document text summarizer that uses Bayes classifier which is one of the machine

learning methods. It uses several modules to resolve the problem of anaphora for extractive summaries. The problem of anaphora arises when a word is used in the document which refers or replaces a word used earlier, to avoid repetition of the word or phrases.

2.1.1 Naïve Bayes Method

This method was used by several researchers in 1990's. Kupiec et al. [20] extended Edmundson's work to design an algorithm that uses a Naïve Bayes method which learns from data and includes a Naïve Bayes classifier to determine whether a sentence should be included in summary or not. It includes control features such as sentence length and presence of uppercase words. A set of technical documents was used to perform evaluation by manual mapping of generated text with actual document sentences. Feature analysis of the output concluded that an algorithm using sentence position, cue words and sentence length feature performed the best. This summarizer does not include pre-processing step which affects the frequency analysis of word sequences. To identify articles, prepositions, common adverbs and auxiliary verbs, a stop list is suggested to break the words in different sentences into phrases. Aone et al. [30] also incorporated Naïve Bayes classifier with richer features. Their proposed system employed features like term frequency (*tf*) and inverse document frequency (*idf*) [30]. *idf* is calculated by taking a set of multiple documents, as it is a calculation of the ratio between total number of documents by the number of documents which contains a given term. A Naïve Bayes method is mainly suitable for small datasets. The method generates low quality summaries for larger documents even with the inclusion of control features.

2.1.2 Rich Features and Decision Trees

Lin and Hovy [21], studied the importance of a single feature such as "sentence position" and assumed that features are independent of each other. A position method needs to be combined with other control features like cue words or phrases because only a position method does not support goal-oriented topic search. Later, Lin changed this assumption and implemented the problem of sentence extraction using decision trees instead of Naïve Bayes classifier [24]. Several statistical features such as sentence position, sentence length, and cue words were examined to study their effect on sentence extraction for summaries. A reference dataset was created for evaluation of summaries generated by the summarizer. Some common features used by Lin are: query signature (score given to sentences depending on query words that they contain), IR signature (the m most salient words in the dataset) and identification of numerical data, names, pronouns, weekdays and quotations.

2.1.3 Hidden Markov Model

Before this method, all the approaches were mostly feature based and non-sequential. Conroy and O'Leary [22], proposed the extraction of sentences from a document using Hidden Markov Model (HMM). The main focus of the method was to use a sequential model that can account for local dependencies between sentences. They used only three features: position of the sentence in the document (built into the state structure of HMM), the number of terms in the sentence or sentence length, and similarity of the sentence terms given in the document terms. Evaluations were carried out by comparing the results with human generated extracts. Another summarizer known as CLASSY [31], is a multi-document summarizer for the field of "news". It is a query based system and uses Hidden Markov Model for sentence scoring and selection. It categorizes sentences as those which are to be included or excluded from the

summary [31]. One of the issues related to this summarizer is anaphora resolution. The performance of the summarizer can be increased in terms of linguistic quality questions by including anaphora resolution.

2.1.4 Log-Linear Method

Osborne [23] claimed that all existing approaches for summarization have always assumed feature independence. The author introduced a log-linear method to eliminate this assumption and showed empirically, that the system produced a better extract than a Naïve Bayes model. The evaluations were done by using a standard f-score where f-score=2pr/(p+r), precision (p) and recall (r), which were measured against human generated extracts. Some of the features incorporated in this method are: word pairs (a pair of words with all words truncated to ten characters), length of the sentences, position feature, and discourse features such as inside introduction and inside conclusion. Similarly, SimFinder [32] uses a log-linear model to organize text into tight clusters and then reduces each cluster to a single sentence. A comparison study revealed that SimFinder showed better results with log-linear model than *tf-idf* and other models. The performance of SimFinder can be improved by combining it with control features such as sentence length and position. The summarizer is multi-lingual and resilience during translation from the source language to target language is a challenge for authors.

2.1.5 Neural Networks

Svore et al. [33] proposed an algorithm, NetSum, based on neural networks and the use of third-party database. Thee trained model could infer the proper ranking of sentences in a test document using a neural network which incorporates a gradient descent method for the

training. The authors concluded that a sentence which contains keyword used in search engine by the user possess a greater chance to be selected in the generated output summary. NetSum is a single document summarizer and uses neural networks to enhance sentence features, then extracts a maximum of three sentences which best match the document highlights. ROUGE evaluation and results for NetSum are very low mainly because it is very rare to match the content of single sentence with the keywords. To improve the performance of NetSum, sentence simplification, sentence splicing and merging are identified as suggested improvements. The other problem faced by NetSum is lack of control features. These features can be beneficial to the performance of the summarizer and would help to identify inputs provided by user. In 2005, Nenkova [34] proposed a summarizer to generate a 100-word summary of a single news article. The author proposed a system for the evaluation of generated output summaries using neural networks and designed a baseline system with statistical significance. During the evaluations, the best performing systems could not outperform the baseline system proposed by Nenkova which corresponds to the selection of first n sentences of a newswire article [34].

2.1.6 Summary of Machine Learning Approach

To describe a model for text summarization, most of the models based on machine learning approaches are embedded with features such as position, cue words, signature words and term frequency. Machine learning methods use several statistical features on sentences from the document collection and then calculate the probability of each sentence for inclusion or exclusion from the generated output summary.

A Machine learning approach requires large databases and training sets for learning process. In terms of text summarization, this learning process may take a significant amount

of time and processing of information. An algorithm which deploys a machine learning approach requires continuous training of data to incorporate new cases and improve accuracy. These algorithms may fail in situations where proper or sufficient training is not applied. Large dataset requirements by machine learning algorithms still may not guarantee best quality summary. Availability of relevant and large datasets is another challenge for machines learning algorithms in text summarization.

2.2 Clustering Based Approach

Clustering based approach [11] [35] is suitable for both single document and multi-document summarization. The algorithms which use this approach makes clusters of documents which are to be summarized and then find relationships among the selected documents [36]. Each cluster is then indexed depending on the theme of the cluster. After indexing, sentences are ranked within each cluster and their saliency scores are calculated. In the last step, high score sentences from each cluster are extracted to generate a summary. Control features such as cosine similarity, cohesion, discounting and relevancy can be added to the algorithm to enhance the performance and summary quality. The main steps of the approach remain same, which are: segmentation into clusters, theme cluster identification, sentence score calculation and summary extraction.

Jade Goldstein [37] proposed a method based on clustering-based approach. The proposed method mainly focuses on “relevant novelty”, which is a metric for minimizing the redundancy and maximizing the relevance and diversity in generated summary. The method uses clustering, coverage (to cover most of the important parts of the summary), anti-redundancy

and summary cohesion criteria (to extract sentences in a way which produces a cohesive flow in the summary). The author proposed some improvements such as generating coherent temporally based event summaries and effectively using multi-document summarization through interactive interfaces to learn large documents and datasets. Chin & Lin introduced a multi-document text summarizer in the field of "news" to extract summaries [38]. The summarizer produces multilingual summaries in English and Chinese. It uses clustering techniques along with "meaning units' detection" to find similarities between topic chains and linking elements. The main challenge faced by authors is translation ambiguity. This summarizer produces good quality summaries when the original document and the respective generated output summary is in same language i.e. either English or Chinese. Early in 2000's, many researchers contributed towards text summarizers and proposed new algorithms. Most of the summarizers were multi-document, extract based and for the field of "news". One of the clustering based approach summarizer is CENTRIFUSER [39], which produces query-driven summaries and is based on a document topic tree. A similarity function is used for query mapping with structural information from topic trees. This summarizer is domain specific for health-care articles. Control features and additional document features can be added to increase performance of the summarizer. A probabilistic method can be replaced by the existing template-based realizer to produce appropriate sentence patterns based on contextual analysis. Judith [40] proposed a clustering based algorithm for multi-document summarization. The summarizer uses linguistic trimming and statistical methods to generate summaries. The structural design of the summarizer is made up of five steps: preparation of raw texts, trimming of sentences, scoring, redundancy elimination and sentence organization for final summary. A problem faced by the summarizer is the evaluation of each component on languages other than

English. For example, redundancy removal effectiveness is not established for non-English languages.

Xiaojun Wan [41] also proposed a clustering-based approach algorithm which uses two models for the process of sentence ranking. One model is used to incorporate the cluster-level information and the second model is used to consider the clusters and sentences as hubs for the calculation of sentence scores. To improve the performance of algorithm, subtopic information can be included which is more fine-grained than the original document. Nitin Agarwal [42] contributed towards a clustering-based approach to generate a summary without human supervision. The summarizer consists of four principal modules: text tilling, clustering, ranking and summary presentation. The main concern of the summarizer is that it cannot generate good quality summaries for longer documents. This would require more filtering and ranking to avoid lack of focus in generated output summaries. NeATS [43], is a multi-document extractive summarizer for the field of "news". It uses control features like sentence position, term frequency, topic signature and term clustering. Stigma words and time stamps are used by the summarizer to improve cohesion and coherence of the output summaries. Newsblaster [44] is also a multi-document extractive summarizer for the field of "news". The algorithm uses clustering-based approach to group news articles using Topic Detection and Tracking (TDT) mechanism. One of the main challenge for Newsblaster is to retrieve the caption from the images of newspaper. For this, categorization of images needs to be done first which is based on shallow parsing of captions and simple word similarity metrics. K.U. Leuven [45] introduced a single and multi-document extractive summarizer. It uses a clustering based approach and employs a topic segmentation techniques for multi-document tasks. To improve the performance of the summarizer, features such as sentence weight, sentence scoring and

position, proximity to the topic, topic segmentation and compression are used for single-document summarization.

2.2.1 Summary of Clustering Based Approach

The advantages of clustering based approach are: it's anti-redundant (to minimize redundancy and to maximize relevance and diversity) and includes summary cohesion criteria. Due to this reason, clustering based approaches are more suitable for multi-document text summarization. A disadvantage of clustering based approach is that it includes cohesion and coherence which are not very easy to implement. Advanced systems are required for computations of an algorithm which deploys cohesion and coherence because of the complexity and processing. Another disadvantage of clustering based approach is that it is not suitable for single document summarization because a set of documents is required for clustering so that related documents can be divided into respective clusters.

2.3 Lexical Chaining Approach

A Lexical chaining approach [46] include chains of words which are defined as semantically related words spread over the entire document. A word is included in the chain if it is cohesively and coherently related to existing words in the chain. Each chain of words represents a semantically related cluster of words. Words from different passages of a document are grouped together into meaningful clusters of chains to identify various themes within a document. These clusters are then arranged systematically to form a binary tree structure. Lexical chains start building up when the first word of the document appears. The

second word is then examined to estimate whether it is semantically related to first one or not. If the words are related then the second word is also added in first chain, otherwise a new chain starts. There are several methods for the calculation of semantics, cohesion and coherence between words, but it involves implementation of a complex algorithm. For lexical chaining, pre-processing is an essential and important step because relevancy of articles and prepositions makes no sense. Similar to previous approaches, a lexical chaining approach also uses various statistical features to enhance the quality of generated summaries. This approach can be applied to both single and multi-document summarizations.

Barzilay and Elahadad [47] proposed an algorithm to compute lexical chains using text from a given document. They introduced a WordNet thesaurus, a part-of-speech tagger, shallow parser for the identification of nominal groups and a segmentation algorithm. Topic identification of the text was then accomplished by grouping words into lexical chains. Extraction of sentences was carried out by identification of strong chains of words. Cut-and-Paste [48] is a single document, domain independent and abstractive summarizer. It implements control features such as lexical coherence, *tf-idf* score, cue phrases and sentence positioning to identify key sentences by a sentence extraction algorithm. Cut & Paste is an abstractive summarizer and generates summaries which are generic in terms of topic. More focused and structural summaries are the challenges for this summarizer. A summarizer proposed by Brunn, Chali and Barbara [49] performs topic segmentation and sentence extraction techniques along with the computation of lexical chains for each segment. It introduces heuristics to make summaries more coherent and readable. The summarizer faces the prospect of longer sentence selection for the generated output summary because typically longer sentences contribute towards longer chains.

ALEXIA, Acquisition of Lexical Chains for Text Summarization [46], uses lexical cohesive relationships instead of standard linguistic resource “WordNet”. An automatically constructed lexico-semantic knowledge base was implemented to identify cohesive relationships between the words. Unlike the approaches which use “WordNet” as a knowledge source, authors examined lexical cohesive relationships that cannot be defined in terms of thesaural relationships, but are considered "intuitively" related due to their regular co-occurrence in the text. ALEXIA uses pre-built software to identify candidate words for the chaining process. For example, it tags the words to be considered in the chain and then extraction of compound nouns is carried out by SENTA software. Filtered nouns, compound nouns and proper nouns are considered as candidate words for further processing of lexical chains. To assign a given word in a lexical chain, a calculation of semantic similarity between all clusters is performed. Two words are semantically related when the semantic similarity between them is greater than a given threshold. Finally, to identify strong chains, a chain score is computed as the sum of similarities between all chain members. Another effort was made by Maheedhar [50] towards an algorithm for automatic text summarization using lexical chains. The algorithm computes lexical chains for the given document set and then uses them to extract important parts of the document. Three tasks are mainly considered for the design of the algorithm: headline generation, multi-document summarization and query-based summarization. The design of algorithm also consists of document processing and segmentation, text chunking, noun extraction and lexical chaining. It has two separate modules: one for single-document and the second for multi-document summarization. The summarizer faces a problem of document clustering to generate coherent summaries for multi-document summarization. It is very important to identify clusters of documents because these clusters

point towards different themes. A good similarity measure can overcome this problem and would improve the performance of the summarizer.

2.3.1 Summary of Lexical Chaining Approach

The main concern with lexical chains approach is that huge databases and memory is required to build the chains. Semantics play an important role for lexical chains, especially for synonyms and to understand the actual meaning of the words. Implementation of these two aspects is beyond the capability of today's computing devices. Without the implementation of semantics, there are some limitations on lexical chains approach. First is the selection of longer sentences because they usually contribute towards longer chains and get included in the summary [47]. The other problem is that; the extracted text mostly contains anaphora links to the rest of the sentences in the document. It is very difficult to identify these links and then to replace anaphora with their referent is another challenge. Another problem is that, there is no way to control the length and the level of details of the summary using lexical chains.

2.4 Frequent Term Approach

A Frequent term approach [51] [52] [53] is a relatively new technique and it requires pre-processing as an essential step. This method checks for terms which are frequent and semantically similar. There are some methods like *tf-idf* (term frequency – inverse document frequency) used for the calculation of frequent terms in the document. *tf-idf* method involves frequency of both frequent and non-frequent terms. *tf* is the frequency of a term and *idf* is the inverse document frequency. *idf* is a logarithmic function which eliminates most frequent and

least frequent terms. For calculation of semantic similarity, it checks the length of the path linking the terms, position of the terms, measures the difference in information content of two terms and the similarity between two terms like definitions and synonyms. The summarizer then filters the sentences which have most frequent and semantically related terms to extract the related sentences for final summary.

Mani and Marbury published a book [54] which presented key developments in the field and suggested future research areas. They defined different approaches of text summarization such as Classical Approaches, Corpus-Based Approaches, Exploiting Discourse Structure, Knowledge-Rich Approaches, Evaluation Methods, and New Summarization Problem Areas. An algorithm known as ADAM was also proposed. It takes single document input and generates indicative summaries for the field of chemistry. The main features used by the algorithm are: cue phrases, term frequencies and sentence selection or rejection. The summarizer does not include semantics and other natural language processing tasks like text classification, collocation extraction and word sense disambiguation. Authors believe that addition of these improvements would contribute towards even better results. Mitze and Rau [55] described a system ANES, which performs domain-independent automatic condensation of news from a large commercial news service. They focused on evaluation of the generated summaries which included 250 documents of three different lengths resulting in 1500 evaluations. This was the largest evaluation of human versus machine in text summarization in late 1990's. They used term frequency and sentence weighting which is also known as *tf-idf*. In their proposed algorithm, first sentences of the documents or passages are given most importance and get selected in the generated output summary. This is also a disadvantage of

the algorithm because it may not generate good summary where the first few paragraphs do not contain important information.

GOFAISUM [56] is a multi-document extractive summarizer in the field of "news". It is mainly based on symbolic approach and implements basic techniques like *tf-idf* and syntactic pruning. The summarizer then extracts the sentences with highest score to build the summary. However, anaphora resolution remains a challenge for the summarizer. A better anaphora resolution module which can solve the problem of question-answering could significantly improve the quality of generated output summaries. Subsequently, Ledeneva [52] presented an algorithm which combines frequent term approach with Graph-based approach. A method with multiword descriptions was proposed which consists of four steps: term selection, term weighing, sentence weighing and selection. The same method was later used for Graph-based approach. The evaluation and results of the summarizer revealed that a pre-processing module included in the summarizer did not affect the results and quality of the summary. However, pre-processing improved the time performance of the algorithm. As a suggested improvement, more extensive pre-processing might increase the quality of summaries along with time performance. In 2011, a frequent term and semantic similarity based text summarization algorithm was proposed for single documents [51]. This algorithm is designed to work in three steps. In the first step, the given document is processed to eliminate stop words and then performs stemming. The second step involves the calculation of frequent terms from the document which are used to find semantically equivalent terms. Finally, the output summary is generated which contains the most frequent terms identified in the second step. The algorithm is designed for single document summarization but can be extended to a multi-document using the similar concept and open source technologies. FociSum [57] merges

information extraction with sentence extraction techniques to get extractive summaries. The topic of the text (which is known as "foci" in the algorithm) is determined dynamically from name entities and multiword terms. The evaluation results shown by the summarizer are inconclusive mainly because a task-based evaluation scheme was not used. The authors faced many problems in evaluation because they conducted qualitative analysis only. The main concern raised by judges was time constraints for reading the original documents and their respective generated output summaries. Gupta [7] used simple statistical measures to find the most significant passage of the source document that consists of most frequent terms, known as kernel. The kernel is then used as a guideline to choose other sentences from the document for generated output summary. The biggest challenge for this summarizer is to generate a good quality summary (in terms of language, format and size) from a number of textual and semi-structured sources (including databases and web pages) for a specific user.

2.4.1 Summary of Frequent Term Approach

The main advantage of frequent term approach is that it is easy to implement and covers a part of pre-processing. This approach is very famous for extractive single and multi-document text summarization. *tf-idf* principle is a weak method because it does not capture the position of the text in the document, semantics and co-occurrences. Therefore, the algorithm which implements *tf-idf* needs to be combined with some other control features like a position method and cue phrases or words. *idf* implements a part of pre-processing but it also excludes the most occurring terms along with least occurring terms. Another problem with this approach is that it tends to select longer sentences which sometimes include redundant information. *tf-idf* cannot capture semantics and it becomes complex when combined with semantics, cohesion and coherence.

2.5 Information Retrieval Approach

Information retrieval approach [35] [58] is an enhancement on two Graph-based methods introduced by Radev and Erkan, which are LexRank (threshold) and LexRank (continuous). In this method, the main feature used is logical closeness i.e. how two sentences are logically related to each other rather than just the topical closeness. In addition to this, it considers that sentences must be coherent. In this approach, more related sentences are picked up in a chain to produce a logical summary. This technique is very similar to Graph-based approach and lexical chaining.

Information retrieval approach focuses on logical closeness which is the key advantage of this approach. Topical closeness is based on synonymy which is not strong enough to measure the coherence of sentences [35]. Negi et al. [58] proposed a model for text summarization that uses pattern recognition techniques to improve retrieval performance of relevant information. The proposed model is designed for multi-document summarization and requires an input query from a user. For the calculation of relevance of a document, identifiers are first defined and constructed. These identifiers are based on grammatical aspects of English language pertaining to which information is retrieved from the given document. They used English Language features such as tense, voice and speech form to retrieve information from the set of documents. Alfonseca and Rodriguez [59] contributed towards a single document text summarizer which uses an information retrieval approach to generate very short length (up to 10 words) summaries for articles. It uses a genetic algorithm to identify relevant sentences. Relevant words and phrases are then extracted by keeping coherence of the output. The evaluation and results from DUC 2003 showed that the summarizer ranked at 7th position out

of 13 systems that participated in the first task. The summarizer generates good quality summaries for short documents and is mainly designed for short summaries.

2.5.1 Summary of Information Retrieval Approach

There are some advantages and disadvantages associated with this approach. The biggest disadvantage is that the implementation of topical closeness is very difficult and involves complex algorithms and computations. Techniques like logical or even topical closeness with information retrieval using pattern recognition involve new modeling and estimation methods that are beyond the scope of traditional approaches and are very complex to be implemented. An advantage of this approach is that it can be adopted for both single and multi-document text summarizations. Moreover, it can be used for both abstractive and extractive summarization.

2.6 Graph-based approach

Graph-based approach [4] [13] [8] [7] [60] [35] [1] is used for both single document and multi-document summarization. For text summarization, it first introduced by Mani [54], for an algorithm that applies a spreading activation technique to identify nodes (sentences) related to the main theme of the document. In this approach, each sentence is treated as a node. Two nodes (or two sentences) are connected to each other by an edge if they have similarity between them. Calculation of similarity depends on many aspects, for example, two nodes can be connected if the sentences have some commonality or have cosine similarity between them. There are numerous other methods to calculate similarity between the sentences like discounting, cumulative sum method [13] and position weight [8]. In a Graph-based approach,

sub-graphs are generated which may or may not be connected to each other. These sub-graphs show the number of topics covered in the documents as shown in *Figure 1*. From this representation, we can identify the most important sentences of the documents. The nodes or sentences which have more edges connected to other nodes are considered as the most important sentences. Graph-based approach depends on sentence centrality and centroid [4], which are the measures used by researchers to calculate similarity between the nodes.

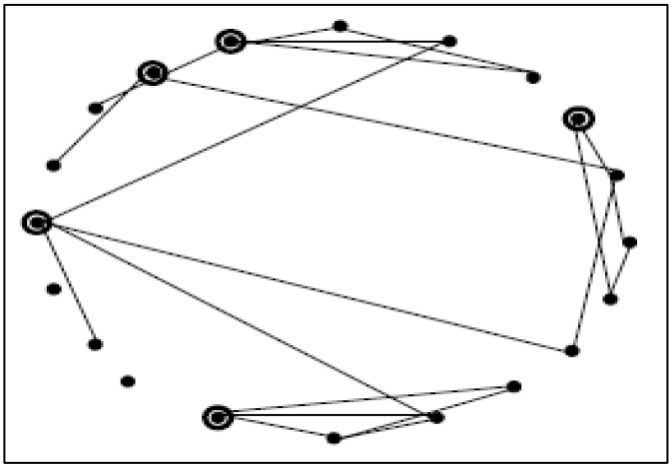

Figure 1: Graph-based approach showing different topics covered in a document *[7]*

In 2004, Rada and Paul introduced TextRank [61], which is a Graph-based ranking algorithm for text summarization. The algorithm visualizes text as a graph and translates the graph in a way that interconnects words and test entities with meaningful relations. The algorithm starts by adding a vertex (node) for each sentence which can be linked together. The links between vertices are defined by sentence similarity relation that leads to the selection of top scored sentences to generate the summary. LexRank was also proposed by Radev and Erkan [4] during the same period, that uses a model based on matrices. The matrix depends on intra-sentence cosine similarity which is also known as adjacency matrix. The algorithm uses

centroid based methods to find sentence centrality which helps in the calculation of degree of the sentences. This leads to the selection of top scored sentences. MEAD [62] is a multi-document extractive summarizer for the field of "news". It is based on sentence extraction through features such as, position, overlap with the first sentence and centroid score. It automatically discards sentences which are very similar to each other. The evaluation and results of the summarizer showed satisfying results with DUC 2001. The evaluation of MEAD included ten summarization systems (also a trainable version of MEAD) for both single and multi-document summaries. Similarly, Zhang [35] proposed a Graph-based method that combines the content of the text with cues and tries to investigate sub-topics from the document to generate the expected output summary. A drawback of this summarizer is that it tends to include longer sentences in generated output summary which introduces redundancy in the generated output summary. A MSR-NLP summarizer [63] is a multi-document extractive summarizer for the field of "news". It uses a graph scoring algorithm to identify highly weighted node relations. Its main objective is to identify important events in the given dataset. The summarizer's goal is to generate summaries which are more human-like and coherent. The summarizer showed the expected results but needs exploration in diverse topics as it uses a completely event-centric method.

Xiaojun Wan [60] presented a model based on a Graph-based approach which is a two-link graph that claims to select the most important and highly correlated sentences for the summary. The model includes document-level information and sentence to document relationship into the Graph-based ranking process. DUC 2001 and 2002 was used to demonstrate the effectiveness of the proposed algorithm. The model uses coarse grained document-level information which can be used to find a sub-topic within a document. Sub-

topic may contain more refined information and can be implemented using Text Tilling Algorithm [60]. In 2009, Hariharan and Srinivasan [13] proposed some enhancements on LexRank. They defined discounted cumulative sum method and discounted degree centrality method. Two metrics were used to evaluate the summaries generated by their proposed methods. Further enhancements on LexRank were presented in 2013 [8] introducing techniques such as discounting and position weight in LexRank and Continuous LexRank. These enhancements on LexRank improved the evaluation and results and the quality of summaries. FemSum [64] is also a single and multi-document text summarizer, designed to answer complex questions using a syntactic and semantic representation of the sentences. It uses Graph-based representation to establish links between the candidate sentences. FemSum is organized in three language independent components: Relevant Information Detector (RID), Content Extractor (CE) and Summary Composer (SC). In addition, it also consists of a language dependent Linguistic Processor (LP) and a Query Processor (QP).

2.6.1 Summary of Graph-based approach

The main advantage of Graph-based approach is that it provides a vertex (node) representation of sentences and the similarity between them. It considers each sentence as a node and the similarity is represented by edges, which gives representation similar to a network. It is very easy to visualize neighbor nodes which have high scores and their edges drawn to other nodes. One of the disadvantages of a Graph-based method is that it does not include semantics and mainly works on statistical features. If the sentences selected in the summary are from each sub-graph, then there is a chance that summary is not correlated or some redundant information is selected. In some cases, Graph-based algorithms tend to select

longer sentences when the method is not combined with a controlling or cut off function for sentence lengths.

Researchers have been working on automatic text summarization for many years. They combined their efforts to generate summaries that can be of as good quality as human generated summaries. The struggle is still going on because the capabilities of today's technology are limited. Text summarization has many applications including the summarization of scientific papers and journals for researchers, description of a book for readers, summaries of lectures for students, a short description of TV or radio programs, navigation of online information, reviews of the products or services and much more. As revealed earlier, we are opting towards graph based text summarization along with added control features because of the advantages offered by Graph-based approach discussed in section 2.6. The power of Graph-based search [14] benefited many businesses like Facebook and Google. Many large enterprises which have online software to maintain their databases are choosing Graph-based search engines, such as Walmart, eBay, Cisco, HP and Telenor [14]. In the next chapter, we discuss Graph-based approach in further detail together with our proposed method.

Chapter 3

3. Design and Implementation

In this chapter, we discuss the design and implementation of the proposed method and its benefits. In this method, we use a Graph-based approach to extract summaries and incorporate pre- and post-processing modules together with control features. Jaccard distance is also used to calculate similarities between sentences. The chapter contains four sections. First, we explain the underlying LexRank [4] algorithm which is followed by a description of the proposed method. The next section illustrates interface of the proposed summarizer (GAUTOSUMM – Graph-based Automatic Summarizer) which is designed and implemented using MATLAB GUI. We conclude with discussion and the benefits/limitations of the proposed method.

3.1 LexRank

The application of the Graph-based approach to text summarization was originally proposed by Mani [35] and then by Radev and Erkan [4]. Improvements to this method were subsequently proposed for single and multi-document text summarization which included

better evaluation process and techniques. The method proposed by Radev and Erkan is known as LexRank which works on the concept of sentence salience to identify the most important sentences in a document (*Figure 2*). Sentence salience is defined in terms of the presence of important words or in terms of similarity to a centroid pseudo-sentence [4]. To understand LexRank, some of the terms used in the method are defined below:

Centroid: It is a pseudo-document which contains words that have *tf-idf* scores above a predefined threshold. In centroid-based summarization, a sentence that contains more words from centroid is considered as central and defines the sentence centrality; this kind of sentence is called **Centroid pseudo-sentence.**

Sentence centrality: The necessary and sufficient amount of information provided by a sentence related to the main theme of a document is sentence centrality. A common way of determining centrality of words is "to look at the centroid" [4].

Sentence salience: It is defined in terms of the presence of important words in a sentence or in terms of similarity to a centroid pseudo-sentence. Sentence salience consists of two factors, cosine similarity between two sentences, and the sentence centrality as compared to other sentences in a document.

Cosine similarity equation: Each sentence is considered as a vector and the corresponding value of *tf* in this representation defines the number of occurrences of the word times the *idf* of the word. The cosine similarity between two sentences is calculated by taking "cosine between their two corresponding vectors" [4].

Degree: It is defined as the number of edges from one sentence to other sentences. An edge (or a vertex) exists between two sentences when there is a similarity between them. Such similarity is calculated using a cosine similarity equation [4].

Cosine threshold: In a document, many sentences are expected to be similar to each other since they are all on the same topic. To select sentences with significant similarities, edges with low values of cosine similarity are eliminated by defining a threshold. For example, for a threshold of 0.3 in Figure 2, the edge between S3 and S4 would be eliminated. Cosine threshold decreases redundancy and repetition in the generated summary by avoiding sentences which are very similar to each other.

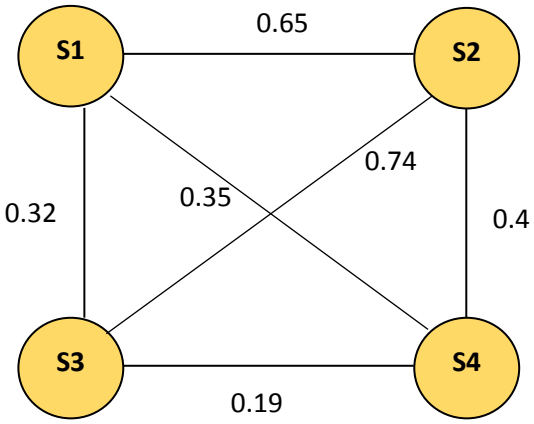

Figure 2: Graphical representation of LexRank [4]

S1: Good food is a basic need of human body and its growth.

S2: It is of prime importance in the attainment of normal growth and development.

S3: The role of nutrition food cannot be neglected in the promotion of good health and prevention of disease in human body.

S4: In recent years the influence of mal-nutrition in the area of mental retardation is being actively investigated.

Figure 3: Selected sentences [65] for graphical representation of LexRank in Figure 2

The graph shown in Figure 2 is an undirected graph and the values of edges between sentences S1, S2, S3 and S4 (in Figure 3) show the measure of similarity. For example, cosine similarity between S1 and S2 is 0.65 which is calculated by taking the cosine of the vectors of both sentences, where the vectors contain *tf-idf* of each word [4]. The cosine similarity between two identical sentences is 1 which means self-directed edges are considered (Table 1). The inclusion of self-directed edges is to avoid a divide by zero situation, when the two sentences are completely dissimilar.

From the values of similarity between four sentences in Figure 2, an adjacency matrix or cosine similarity matrix is generated (Table 1). It is a symmetric matrix which includes two vectors: Sum and Degree. The sum vector is obtained by adding the matrix either row-wise or column-wise (because the matrix is symmetric). The degree of each sentence is calculated by considering the cosine threshold. For example, the degree of S1 is 4 because it generates four edges: from S1 to S2, S1 to S3, S1 to S4 and S1 to S1. All these edges are included in the degree of S1 because the similarity between them is greater than the threshold (0.3). The degree of S3 is 3 (edge from S3 to S1, S3 to S2 and S3 to S3) because the value of edge between S3 and S4 is 0.19, which is less than the threshold of 0.3. Once the values of Sum and Degree vectors are determined, the sentences with highest value are selected for the summary. In this example, the descending order of sentences in terms of Sum and Degree is S2, S1, S3, and S4.

	S1	S2	S3	S4	Sum	Degree (0.3)
S1	1	0.65	0.32	0.35	2.32	4
S2	0.65	1	0.74	0.4	2.79	4
S3	0.32	0.74	1	0.19	2.25	3
S4	0.35	0.4	0.19	1	1.94	3

Table 1: Adjacency matrix used by Radev and Erkan in LexRank [66]

In next section, the proposed method is discussed which is used to identify edges between sentences using Graph-based approach. The proposed method does not include calculation of cosine similarity and *tf-idf*. However, term frequencies (*tf's*) are calculated for the words which are shared among sentences.

3.2 Proposed Method

The proposed method consists of a pre-processing module, control features (like sentence length) and a post-processing module. Pre-processing is an essential step because it helps to improve the readability of generated summary and time performance of the summarizer. Jaccard distance is incorporated in the method to calculate similarities between sentences in terms of sentence lengths. The post-processing module aligns the output sentences for better representation. The Graph-based approach is adopted mainly because of the following reasons:

1. It provides a clear visualization of nodes that represent sentences. Selection of sentences that are to be included in the generated summary depends on the edges drawn from other sentences. This creates a "democratic system" to locate the important sentences by their relations through connecting edges [66].
2. The method does not need to go through a training process unlike machine learning approaches and can produce summaries without the help of trained templates.
3. Control features such as sentence length and position can be easily incorporated to generate a summary with better quality and sentence structure.

In the proposed method, generated graphs are undirected and weighted, i.e. nodes and edges do not include any specified direction and the relative weights are attached to each edge. Sentences are represented by nodes which are connected through edges when there is a word shared among them. Additionally, our proposed method is unsupervised, meaning that summaries are automatically generated without any human supervision. Currently, the proposed method is designed for a single document and monolingual text summarization.

Similar to LexRank, a visual representation of the nodes and number of connected edges between them is derived (Figure 4). The sentences (represented by nodes) may not have any edge or connection to or from other sentences if there are no common words shared among them. In Figure 4, there are eight edges between S1 and S2 because there are eight words shared between these two sentences. Once all the edges are identified, a symmetric matrix is formed as shown in Table 2. This matrix consists of number of edges between the four sentences: S1, S2, S3, and S4. An edge between a sentence itself (self-directed edge) is taken as 0.

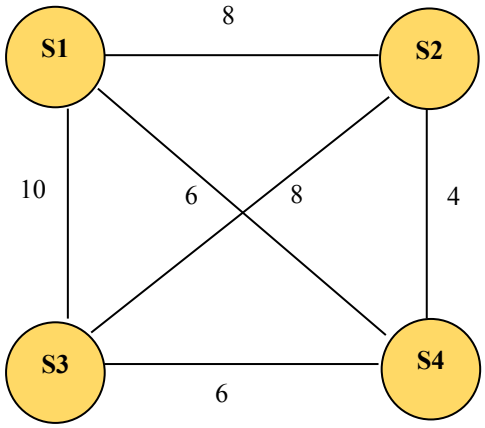

Figure 4: Proposed Method

	S1	S2	S3	S4
S1	0	8	10	6
S2	8	0	8	4
S3	10	8	0	6
S4	6	4	6	0

Table 2: Sentence Similarity Matrix

After generating the matrix, a sum vector is obtained by adding the matrix row-wise or column-wise. For example, S1 has 24 edges in total which is the sum of all the edges shown in Figure 4 (0+8+10+6 = 24). The sum vector is sorted to obtain the highest ranked sentences (Figure 5) which are selected for the output summary.

Sum:

Sentences	Sum
S1	24
S2	20
S3	24
S4	16

Sorted Sum:

Sentences	Rank
S1	24
S3	24
S2	20
S4	16

Figure 5: Sum vector and Sorted Sum

To understand the working of our proposed method, Figure 6 shows an example of edges drawn between two sentences. The number of edges are drawn in both directions which generate a symmetric matrix. For example, both sentences contain word “special”, therefore, an edge between sentences 1 and 2 is drawn. When a word is repeated n times in a sentence, it draws n edges to the other sentence and vice versa. For example, word “Summarization” appeared once in sentence 1 and repeated twice in sentence 2, due to which two edges are drawn between sentences 1 and 2.

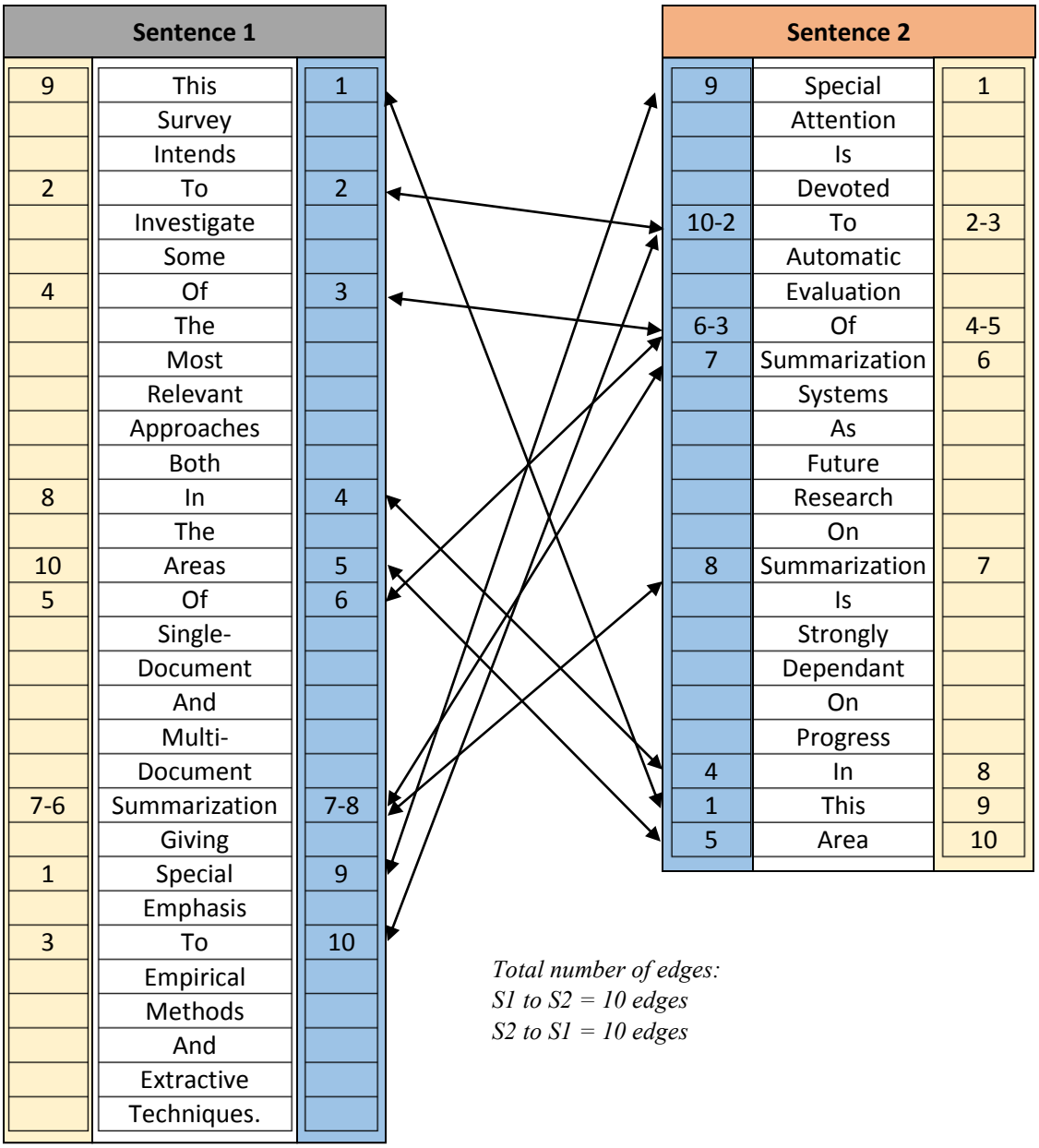

Figure 6: Illustration of proposed method (Without pre-processing)

3.2.1 Pre-processing Module

A pre-processing module is included to clean the text provided by the user, improve time performance, detect sentence boundaries and identify correct number of edges. Cleaning of the text includes removal of stop, bad words, special characters (only for finding the edges), brackets, prepositions, articles, and conjunctions. In English language, word order depends on

lexical categories which are parts of speech. These lexical categories include determiners, auxiliary verbs, prepositions, and conjunctions, which are grouped in the following manner:

- Determiners: “a”, “an”, “the”, “this”, “that”, “these”, “those”, pronouns and quantities.
- Auxiliary verbs: forms of be, “have”, “may”, “can” “shall”.
- Prepositions: “at”, “in”, “on”, “under”, “over”, “of”.
- Conjunctions: “and”, “but”, “or”.

Removal of all the lexical categories does not guarantee good quality summaries (discussed in chapter 4). Due to this reason, only articles, prepositions, and conjunctions are removed during the pre-processing of text provided by the user.

For the same sentences shown in Figure 6, Figure 7 illustrates the number of edges between both the sentences after pre-processing. The pre-processed sentences do not contain stop words, special characters, text (in brackets), prepositions and articles from the text. The number of edges is reduced by 50 percent for this selected example. This highlights the importance and necessity of having a pre-processing step. The impact of pre-processing on the number of edges varies from sample to sample but the inclusion of a pre-processing module always generates compact summaries with improved performance. For larger documents, the accumulated impact of pre-processing becomes even more significant.

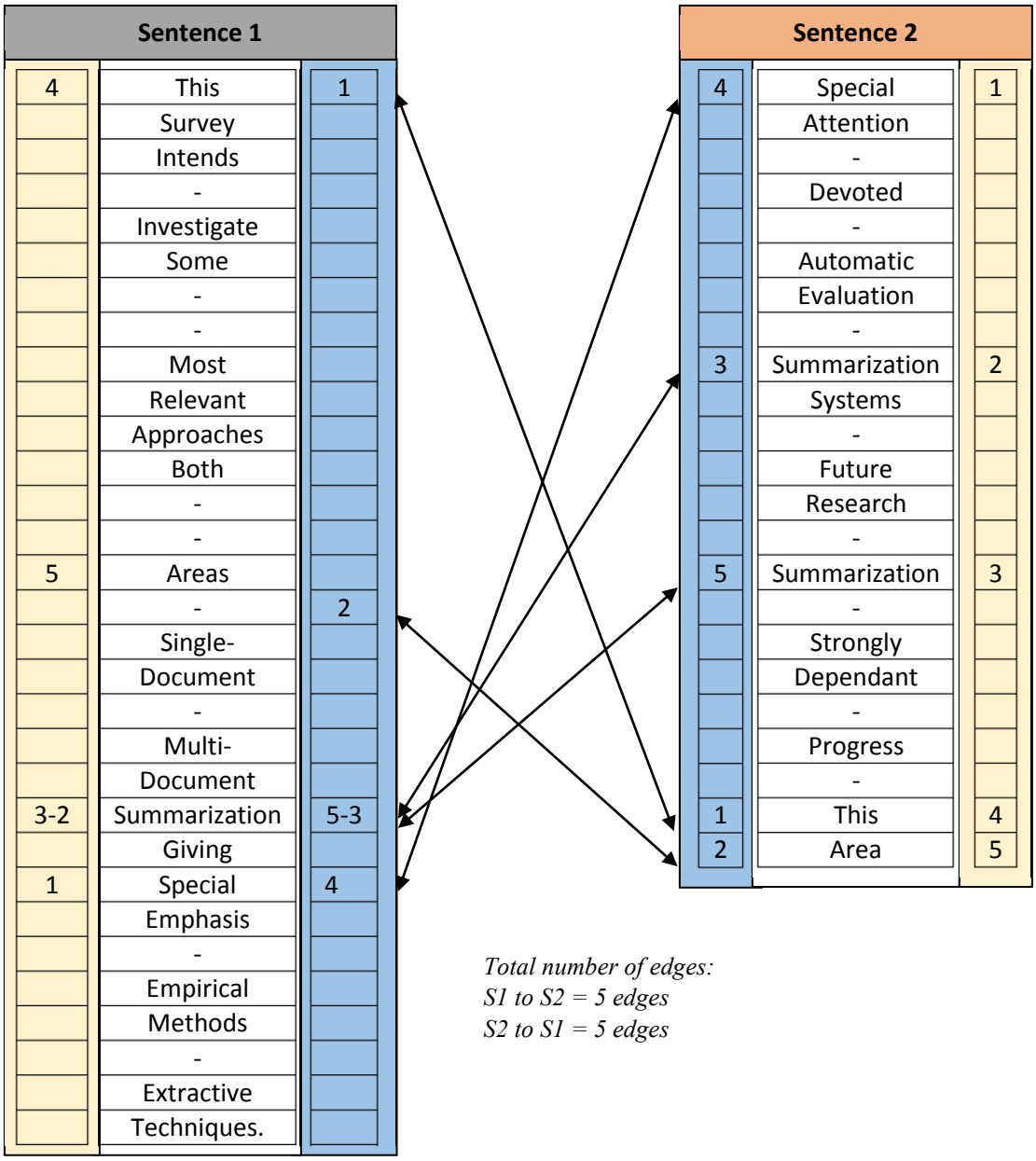

Figure 7: Illustration of proposed method (With pre-processing)

The pre-processing module also removes text written in brackets, references and any special characters connected to the words so that correct number of edges can be calculated. For example, a word “summary” appears with a comma in one sentence as “summary,”. An edge must be generated for the word “summary” appearing in another sentence. Without pre-

processing, this edge would not be generated because of the punctuation attached to the word. This also illustrates that adding a pre-processing module results in a more accurate representation of edges.

In GAUTOSUMM, the pre-processing module performs the following actions:

- Sentence boundaries are identified by period "." or a question mark "?".
- Basic pre-processing, i.e. removes articles, prepositions and conjunctions.
- Removes brackets "[{( )}]" and anything within brackets.
- Removes special characters attached to words (to find true number of edges).

3.2.2 Jaccard Distance

The main concern in a Graph-based approach is that it tends to extract longer sentences. By managing sentence lengths, longer and less important sentences can be eliminated. The proposed method is incorporated with Jaccard distance [15] which calculates similarity between sentences by considering the lengths of the sentences in a more logical way, that is, by calculating the ratio of similar content between the two sentences. The Jaccard distance for two variables A and B [15] can be calculated as:

$$D = | (A \cup B - A \cap B) | / | A \cup B | \quad (1)$$

We modified this equation for sentence lengths and similarities between two sentences A and B such that:

$$\text{Dissimilarity } (A, B) = \frac{| \text{Length } (A+B) - (\#\text{ of edges between sentences A, B}) |}{\text{Length } (A+B)} \quad (2)$$

$$\text{Similarity } (A, B) = 1 - \text{Dissimilarity } (A, B) \quad (3)$$

The absolute value of the numerator in Equation 2 ensures that Dissimilarity and Similarity remains in the interval [0,1]. Consider, for example, the following sentences:

Buy, buy, buy computers. Buy, buy, buy now.

$$\text{Dissimilarity} = | (8 - 9) | / | 8 |$$

$$\text{Dissimilarity} = 0.125$$

$$\text{Similarity} = 0.875$$

The similarity between sentences is determined by the length of sentences and the number of edges between them. For example, if we have the same number of edges between two pairs of sentences, i.e. 7 edges between sentences 1 and 2, and 7 edges between sentences 3 and 4, then it is not conclusive that the similarity between the two pairs would be same. Consider the length of first pair of sentences as 12 and 14 words, respectively. The similarities between sentences 1 and 2 is 7, which calculates Dissimilarity and Similarity from equations (1) and (2) as follows:

$$\text{Dissimilarity}(1, 2) = | ((12 + 14) - 7) | / (12+14)$$

$$\text{Dissimilarity}(1, 2) = 0.73$$

$$\text{Similarity}(1, 2) = 0.27$$

For the second pair of sentences, sentence 3 is 20 words and sentence 4 is 18 words long. The Dissimilarity and Similarity from equations (1) and (2) for this pair are:

$$\text{Dissimilarity}(3, 4) = | ((20 + 18) - 7) | / (20+18)$$

$$\text{Dissimilarity}(3, 4) = 0.81$$

Similarity (3, 4) = 0.19

The number of edges being equal, the sentences which are longer in length contain more dissimilarity than shorter sentences. In the selected example, sentences with shorter sentence lengths i.e. sentences 1 and 2 would be selected in the generated summary instead of sentences 3 and 4. By calculating similarity based on shorter sentence lengths, the summarizer generates compact and good quality summaries.

3.2.3 Control Features

In order to overcome the problem of longer sentence selection, we have included control features which are dependent on sentence lengths. The control feature works such that it takes a range of all the sentence lengths in a document and then modifies this range, depending on the difference of sentence lengths from the average length. The function checks for minimum and maximum sentence length and then, calculates optimal length. In this way, the range of the selected sentences is adjusted for each document to pick sentences with optimized lengths. Average sentence lengths are used instead of excluding the bottom and top range of sentences because sentence lengths are variable in each document and may depend on the topic of original document provided by the user.

Another control feature is enforced by a function which can limit the length of selected sentences for the generated summary. An optimal sentence length can range from 15 to 25 words [67]. Sentence lengths in the range from 35 to 50 words have also been recommended to avoid short and choppy sentences, depending on the topic [68]. To maintain a proper flow in generated summaries with meaningful content, sentence length may have to be more than 25 words. Short sentences often lead to a disconnected summary with less coherence. Due to

this reason, we selected maximum sentence length of up to 50 words. Microsoft Office also contains a built-in function that provides a similar feature which highlights sentences containing more than 60 words and suggests that the user reconsider the sentence.

Very long sentences not only overwhelm the user but may lose its meaning and content. Moreover, longer sentences may introduce more than one theme which also adds redundancy in the generated summary. Therefore, both control features are used in the proposed method to select sentences with optimal lengths.

3.2.4 Interface of GAUTOSUMM and Post-processing Module

An interface is designed for the proposed summarizer (GAUTOSUMM) by using MATLAB's graphical user interface, GUIDE (GUI Development Environment). The user provides two inputs to GAUTOSUMM: text to be summarized and the number of desired sentences in the output summary. A post-processing module cleans the generated summary in order to make the output readable and easy to understand. The final output contains formatted sentences with no redundant special characters, bullets or spaces (Figure 8).

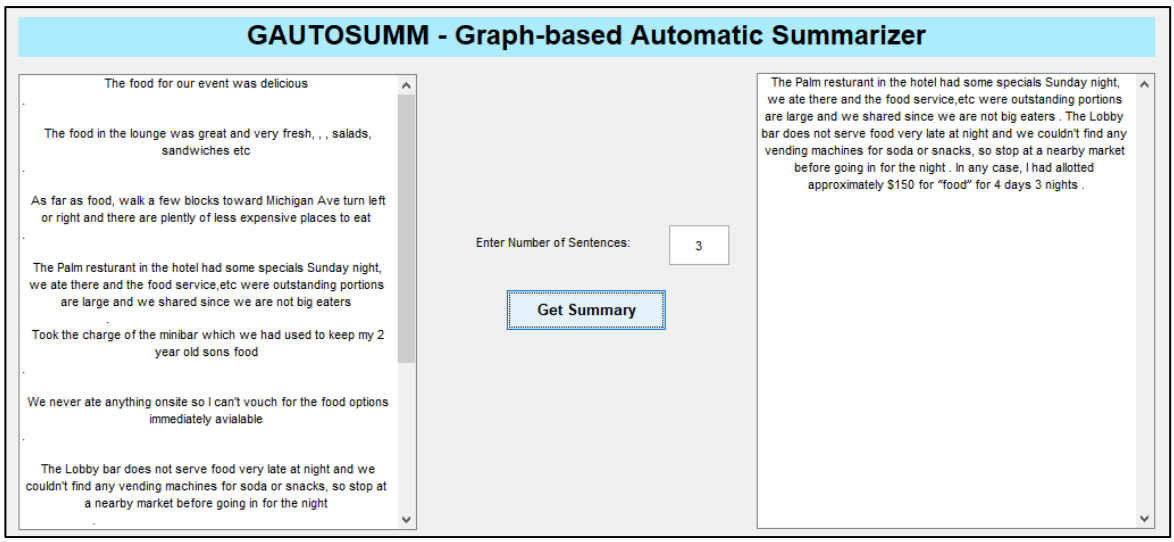

Figure 8: Generated output from GAUTOSUMM with post-processing

3.3 Benefits of the Proposed Method

The proposed method provides the following benefits:

- The method does not include complex mathematical equations like a cosine similarity equation [4] used by LexRank. LexRank uses *tf-idf* for the calculation of cosine similarity, where *tf-idf* is a statistical feature which calculates term frequencies *(tf)* of the whole document. In contrast, our proposed method calculates *tf's* for significant words only (which contribute towards the edges), thus making it more efficient in terms of finding edges which in turn improves the performance of the summarizer. Moreover, the proposed method does not include multiplication and dot product of matrices like a cosine similarity equation, which increases the execution time of algorithm.

- A pre-processing module is implemented rather than *idf*. With a pre-processing module, we get a part of *idf* by eliminating frequently occurring words such as articles and prepositions. In addition, pre-processing keeps words that occur frequently other than prepositions and articles, as described in the following examples:

    i. Suppose there are five sentences with an article "the" in four sentences. The *idf* is calculated as:

    $$idf = \log (5/4) = 0.0969$$

    ii. Consider another example where total number of sentences is also five and an article "a" is present only in two sentences. In this case, *idf* is calculated as:

    $$idf = \log (5/2) = 0.3979$$

    iii. This illustrates that the number of word occurrences in more sentences gives lower values of *idf*. Let us consider another example, where a word "summary" is present in all the five sentences of a document, then

    $$idf = \log (5/5) = \log (1) = 0$$

From this, we can infer that any term which is present in all the sentences will be eliminated, because most likely that term is an article or preposition. Logarithmic functions only consider a range of values. Thus, *idf* can eliminate words that are frequently occurring but are not articles or prepositions. For instance, in example (iii), the word "summary" would be eliminated even though it might be an important word and related to the theme of the document. Those frequently occurring words which are

not from lexical categories can play an important role in the selection of good quality summary and must be included in the calculation of edges. Due to this reason, articles and prepositions are excluded along with bad/stop words (special characters, references in brackets and anything written in brackets) and conjunctions during the pre-processing step of proposed method, instead of *idf*.

- Pre-processing module has also improved time performance of the proposed method by eliminating stop or bad words, special characters, brackets, prepositions, articles and conjunctions, which in return removes false edges from the text.
- The comparisons occur between the sentences i.e. from one sentence to the other but not within the sentence itself (Figure 9). This gives better results in a way that each sentence has its own significance/rank. Edges are not determined within a sentence to avoid any false edges which may lead to a generated summary that contains less important sentences.

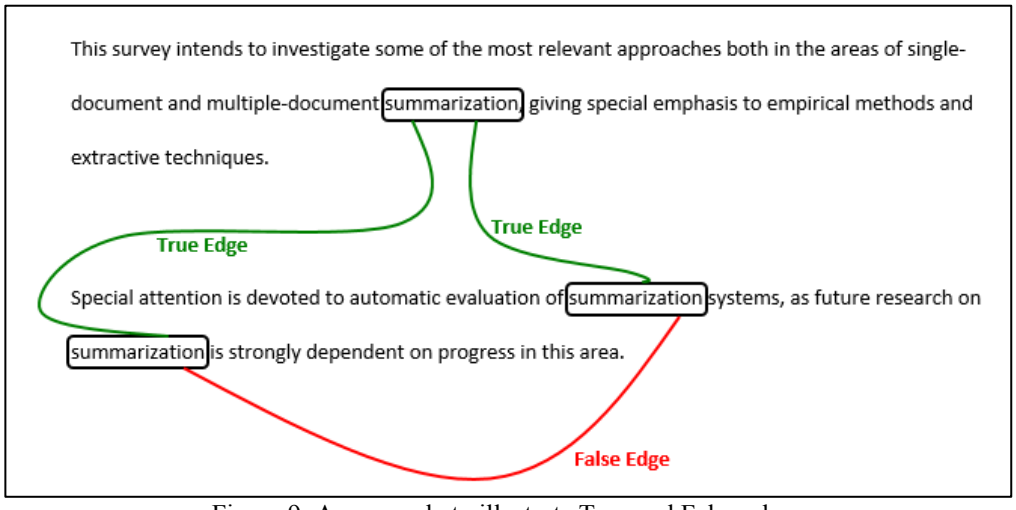

Figure 9: An example to illustrate True and False edges

3.4 Limitations of the Proposed Method

Though we have demonstrated clear benefits of our proposed method, there are some limitations which are listed below:

- Semantics are not included in the pre-processing module of the proposed method. Semantics consist of an add-on dictionary in the module along with anaphora resolution (when a synonym is used to replace a word to avoid repetition). This reduces false edges and results in precise summary with improved time performance. For example, consider two sentences:

  "Red car hit red house. Red car destroyed."

  Here "red" is used for both car and house which shows that semantics are very important in terms of meaning and understanding of the text.
- Post-processing module displays the summary in one paragraph instead of breaking it into smaller paragraphs. This part also belongs to semantics, where a summarizer must be able to recognize different themes in the generated summary.
- The user interface of GAUTOSUMM currently only allows number of sentences as the input along with the original text. An improved interface with more input options (like number of words and keywords) would provide better impact and feel to the reader.
- The proposed summarizer includes control features on sentence lengths. Control features play very important part in tweaking the summary to fit the reader's requirement. The presence of additional control features like sentence position and cue

words (such as “in conclusion”, “therefore”) would improve the quality of generated summary.

- The proposed summarizer is monolingual, i.e. it generates summaries only in English language and can be further extended to multi-lingual.
- The summaries generated by GAUTOSUMM are mostly from documents which consists of general topics instead of scientific or more specialized papers like bank reports or medical procedures.
- The size of document has a significant impact on execution time of the algorithm. Generally, documents exceeding 950 sentences, or those larger than 100 KB, take substantially more time to generate a summary.

In this chapter, we discussed the design and implementation of our proposed method which uses a Graph-based approach and consists of a pre-processing module, control features and a post-processing module. The pre-processing step has been demonstrated to be useful for generating summaries with better quality and improved time performance. Moreover, it overcomes the deficiencies resulting from *idf* by keeping the frequently occurring words (other than articles, prepositions and conjunctions) for the calculation of edges. Selection of longer sentences in the generated summary is avoided by incorporating Jaccard distance and control features. Jaccard distance helps to identify the sentences with fewer redundancies and precise information, whereas, control features eliminate the sentences with longer lengths to avoid redundant information in the generated summary. Post-processing module produces a user-friendly and coherent summary. When compared with online text summarizers, GAUTOSUMM performed better in eight out of ten topics (chapter 4) for the datasets DUC 2007 and Opinosis. The quality of summary is measured by using the ROUGE evaluation software.

Chapter 4

4. Evaluation and Results

This chapter includes an evaluation of generated summaries from proposed summarizer (GAUTOSUMM) and online text summarizers, both in terms of quality and time performance. We have used Opinosis [69] and DUC 2007 datasets [70] provided by NIST, and ROUGE software in the evaluation process of GAUTOSUMM and online summarizers. DUC 2007 dataset includes samples from ten topics and provides gold standard summaries which are produced by human judges. The summaries produced by human judges focus on measuring readability, coherence, precision, grammar, and content. Human evaluations are easy to produce when the dataset is small and small number of documents are involved in the evaluation process. For larger datasets, even with simple and few linguistic quality questions, thousands of hours are required to read original samples and produce corresponding summaries. Generally, a large dataset is required from different topics to make sure that the summaries generated by text summarizer are of good quality and fulfill the need of the user.

4.1 ROUGE

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [71] evaluates the quality of extracted summary by comparing it to model or gold standard summaries created by human judges. ROUGE uses precision, recall, and F-score metrics for this evaluation. To understand precision and recall, consider two summaries 'A' and 'B' where A ("retrieved") is the summary generated by the summarizer and B ("relevant") is the gold standard summary [8]:

$$\text{Precision, } P = |(A \cap B)| / |A|$$

$$\text{Recall, } R = |(A \cap B)| / |B|$$

F-score (harmonic mean of precision and recall) is then defined as:

$$F = 2PR / (P+R)$$

Precision and recall measures the quality of a summary and F-score is the harmonic mean of both the metrics. The value of precision metric reveals the correctness of the content of generated summary with respect to the gold standard summaries. Usually short summaries are similar in length to gold standard summaries and thus yield higher values of precision. On the other hand, recall metric provides the information about flow and structure of a generated summary. A higher value of recall is attained when number of sentences increases in the summary. As precision and recall are inversely related, a higher value of recall decreases precision proportionately, and vice versa. Due to this reason, a harmonic mean of the two metrics is generated to determine the overall quality of a summary.

ROUGE requires reference summaries and system summaries as its inputs. Reference summaries are the gold standard summaries (model summaries) and system summaries are those produced by the summarizer. ROUGE includes a configuration file where the settings like ngram size, stop words file and ROUGE type can be changed. The output file generates three metrics: Avg_recall, Avg_precision and AvgF_score whose values range from 0 to 1. The values of these metrics depend on the content of reference and system summaries. For example, if the reference and system summaries are identical, then a score of 1 is generated for all three metrics. When there is no match between reference and system summaries, then a score of 0 is generated for all metrics. The values of ROUGE evaluation metrics depend on the gold standard summaries provided by human judges. This is a disadvantage because metric values completely depend on the sentences provided in gold standard summaries which may produce biased results. We believe that ROUGE evaluation combined with qualitative analysis of generated summaries would likely provide better assessment of summaries and unbiased results.

4.2 Datasets

Opinosis dataset is available online, while DUC 2007 dataset requires access from NIST (National Institute of Science and Technology).

- **Opinosis Dataset:** Opinosis [69] is "topic-oriented opinion sentences" from different topics such as, hotels, cars and products (such as iPods, GPS and kindle). It contains reviews from the users of product or service. The dataset contains 51 samples in total and each sample consists of approximately 100 sentences. Each sample has 4 to 5 gold

standard summaries that are produced by human judges. The gold standard summaries in Opinosis are short in length, often 2 to 3 sentences, because the dataset was originally designed for abstractive summarizers which summarize the topic in a broader concept.

- **DUC 2007 Dataset:** DUC (Document Understanding Conference) is a series of summarization evaluations conducted by National Institute of Standards and Technology (NIST) since 2001 [72]. The main objective of DUC is to help the progress in automatic text summarization and to enable researchers to participate in large-scale experiments for the implementation, development and evaluation of text summarization systems [72]. DUC comprises of two tasks: Main task and Update task. The Main task consists of real world "complex question answering", in which a question cannot be answered by simply stating a "name", "date" and "quantity" [72]. The Main task contains 45 samples in total with 25 documents in each sample. The task expects a summary of 250 words and it must be well organized and structured in terms of the topic statement. The Update task expects a summary of 100 words from each sample and is designed to "inform the reader about new information" which is related to the given sample [72]. For the evaluation and comparison of our proposed summarizer, a Main task dataset is selected to have the summaries in the range of 250 to 280 words. The evaluation from ROUGE shows that for most of the topics (eight out of ten), GAUTOSUMM generated good quality summaries. We have selected four online summarizers (SMMRY, Tools4noobs, FreeSummarizer and SplitBrain) for comparisons, as most of the other summarizers are unable to process long documents and they limit online access after multiple usage of the software.

4.3 Online Text Summarizers

Eleven online summarizers were considered for the evaluation process. These are listed below:

- SMMRY [73]
- Tool4noobs [74]
- FreeSummarizer [75]
- AutoSummarizer [76]
- SplitBrain [77]
- Text Compactor [78]
- Shvoong [79]
- HelpfulPapers [80]
- Article Summarizer Online [81]
- MS Word Summarization [82]

Most of these online summarizers (for example, SplitBrain [77] and Shvoong [79]) do not allow access to the information about their underlying algorithm or the platform. However, some summarizers provide a brief introduction. For instance, FreeSummarizer [75] is more suitable for news and long texts. Tool4noobs [74] generates a summary for the given text by ranking each sentence considering the relevance. Text Compactor [78] is mainly designed for busy students, teacher or professionals. SMRRY's [73] task is to provide an efficient way of understanding the text, which is done primarily by reducing the text to the most important sentences. AutoSummarizer [76] uses several algorithms to produce better summaries. According to the developer, it uses K-means for clustering the sentences and a Bayer's Naïve

Bayes classifier to calculate the probabilities between words and sentences. Article Summarizer Online [81] underlines the main ideas, provides a brief overview, reflects the writing style and rephrases the original text. Article summarizer helps in figuring out the main message of the document and breaks down the text to identify different aspects of the original document. Finally, MS Word Summarization [82] includes an AutoSummarize tool in MS Word 2007 which identifies the key points in a document by analyzing and assigning a score to each sentence. Sentences that contain frequently used words are given a higher score. User can select a percentage of highest-scoring sentences to display in the summary. It works best on well-structured documents like reports, articles and scientific papers [83].

We began by using all these online text summarizers to generate summaries. By studying their extracted output summaries, some of the limitations were identified. The analysis of evaluation was done by using a dataset of 1000 passages from six topics: business, sports, politics, medical, religion, and science. The identified limitations are briefly described below:

- Most summarizers could not generate summaries for longer text. For example, HelpfulPapers could not generate a summary for the documents larger than 60KB in size.
- Generated summaries are difficult to read due to the presence of special characters and broken sentences (which may contain text from header or footer).
- Summarizers ignored questions in the document, for example, text appearing as a question.
- Most summarizers consider series of questions as one sentence, which introduces redundancy.

- Summarizers are unable to identify sentence boundaries if there is no space between two sentences.
- Some summarizers include only a part of the sentence because of semicolons. This is good for few examples or topics but in most part this feature produced meaningless summaries.
- Summarizers [80] could not deal with punctuation properly.
- If a period comes within a sentence, such as, “Sr. Analyst”, many summarizers consider this as two sentences.

After discovering these limitations, four of the online text summarizers are selected out of eleven for evaluation and comparison with GAUTOSUMM. The selected online summarizers are: SMMRY, Tools4noobs, FreeSummarizer, and SplitBrain. In the next section, results for Opinosis are assessed.

4.4 Results Using Opinosis Dataset

The evaluation of summary quality is carried out by comparing the results generated by ROUGE for proposed method and online text summarizers. An output from ROUGE using a sample from Opinosis and the respective gold standard summaries is shown in Table 3.

Avg_Recall	Avg_Precision	AvgF_Score
0.62528	0.0655	0.11858

Table 3: ROUGE evaluation metrics from SMMRY (sample from Opinosis dataset)

The significance of these numbers depends on the lengths of the generated and gold standard summaries. By observing different outputs from ROUGE, a higher value indicates that summary quality is good. The values of recall and precision significantly depend on gold standard summaries. For example, when the generated output summary is longer than gold standard summary, then the value of recall increases but precision drops. A higher value of precision is obtained for short and precise summary. SMMRY [73] generated good quality summaries for documents which are not very long (one paragraph to one page). For longer documents, the performance of SMMRY is not significant. A comparison between the evaluation metrics of online text summarizers and proposed method is shown in Table 4 for a sample selected from Opinosis [69] dataset. In this table, only those values of metrics are highlighted which are lower than the proposed summarizer.

Summarizer	**Avg_Recal**	**Avg_Precision**	**AvgF_Score**
SMMRY	0.62528	0.0655	0.11858
Tools4noobs	0.54834	0.0859	0.15254
FreeSummarizer	0.43640	0.11860	0.18652
Split brain (5%)	0.45769	0.10000	0.16414
Split brain (10%)	0.63053	0.06460	0.11720
Proposed	0.58796	0.06061	0.10989

Table 4: Comparison of ROUGE evaluation metrics for online summarizers and GAUTOSUMM (without pre-processing)

Example:

We have taken a sample from Opinosis to observe the changes in values of metrics and the corresponding generated summary. The original text can be found online from the Opinosis dataset [69]. The model summaries produced by human judges are as follows:

Model Summary 1:

"The battery life is longer then 5 hours.

But due to the battery charger this may decrease or not work at all."

Model Summary 2:

"Battery lasts about 5 hours.

Time is shorter when running many drives or using bright backlight."

Model Summary 3:

"battery-life is fantastic and good.

The 6 hours battery life is great."

Model Summary 4:

"Battery lasts about 5 hours.

Time is shorter when running many drives or using bright backlight."

Model Summary 5:

"Battery lasts about 5 hours.

Time is shorter when running many drives or using bright backlight."

For the same sample, one and two sentence summaries generated by SMMRY and GAUTOSUMM together with their ROUGE metric values are:

SMMRY (two sentences):

"The warranty card plainly states that the warranty covers everything but, the battery or free accessories such as mice or laptop bag.

Unboxed the netbook, put in the battery, charged it up, everything was great."

Avg_Recall	Avg_Precision	Avg_FScore
0.17955	0.08333	0.11383

GAUTOSUMM (two sentences):

"I spend a lot of time in battery, power environments for other technical devices, so I'm a bit more forgiving when it comes to expected battery life versus real battery life . 5 hours of battery life, bluetooth, a better webcam can someone confirm this?"

Avg_Recall	Avg_Precision	Avg_F-Score
0.3178	0.11304	0.16677

SMMRY (one sentence):

"Unboxed the netbook, put in the battery, charged it up, everything was great."

Avg_Recall	Avg_Precision	Avg_FScore
0.11477	0.13846	0.12551

GAUTOSUMM (one sentence):

"5 hours of battery life, bluetooth, a better webcam can someone confirm this?"

Avg_Recall	Avg_Precision	Avg_F-Score
0.21553	0.24286	0.22838

The recall of generated summaries of two sentences is higher than that of one sentence for both the summarizers. Conversely, the precision is higher for one sentence summaries as compared to two sentences. From this example, we can infer that the summaries which contain words that are present in model summaries and are shorter in length generate higher values of precision and F-score.

When a pre-processing module is added along with control features, significant improvement in the values of recall, precision and F-score is observed (Table 5 and Table 6). These results show that with pre-processing, summaries are more precise and contain less redundant information. Table 5 shows results of a sample where extensive pre-processing is applied by removing articles, prepositions, and conjunctions. In Table 6, only articles and few prepositions are removed from the original document. Removing all lexical categories (which includes determiners, conjunctions etc.) from the sentences or extensive pre-processing does not produce summaries with better quality. This can be observed for the selected sample in Table 5 and Table 6.

Summarizer	**Avg_Recal**	**Avg_Precision**	**AvgF_Score**
SMMRY	0.57975	0.06604	0.11857
Tools4noobs	0.56052	0.06490	0.11634
FreeSummarizer	0.43278	0.07246	0.12414
Split brain	0.61056	0.07353	0.13125
Our (3 sentences)	0.50179	0.08871	0.15077

Table 5: Comparison of ROUGE evaluation metrics for online summarizers and GAUTOSUMM (extensive pre-processing)

Summarizer	Avg_Recal	Avg_Precision	AvgF_Score
SMMRY	0.61073	0.05063	0.09351
Tools4noobs	0.51503	0.06182	0.11039
FreeSummarizer	0.60535	0.06154	0.11172
Split brain	0.54470	0.06154	0.11058
Our (3 sentences)	0.65184	0.07544	0.13523

Table 6: Comparison of ROUGE evaluation metrics for online summarizers and GAUTOSUMM (basic pre-processing)

4.5 Evaluation and Results Using DUC Dataset

The second dataset, DUC 2007, is used for the evaluation of generated summary quality for all summarizers. For GAUTOSUMM's summary quality, evaluation metrics of ROUGE are calculated after the implementation of control features and post-processing module. DUC 2007 dataset contains 45 samples from ten topics which are used to generate output summaries of 250 words and 5% of the original document. The 250 words limit is a requirement of the main task of DUC 2007 and the model summaries. We have limited the output summary to 5% of the original document because SplitBrain only allows to generate summaries in percentages instead of number of words or sentences. 5% is the lowest percentage allowed by SplitBrain for the generated summary. A percentage higher than 5% is not used to generate summaries because of the length of documents provided by DUC 2007; the resulting longer summaries produce lower values of ROUGE metrics as shown in Figure 11.

Tableau software [84] is used to generate and evaluate results produced by ROUGE. The outliers are removed for a more representative analysis of results. Figure 10 shows the results of generated summaries from GAUTOSUMM and online text summarizers for 45

samples using ROUGE metric: F-score. A median is calculated to define a cut off in Tableau to generate the bar charts for each summarizer. For 250 words length summaries, GAUTOSUMM showed better results in eight out of ten topics. For summaries restricted to 5% of the original document (Figure 11), Tools4noobs showed the least performance in terms of summary quality. SplitBrain, SMMRY and FreeSummarizer presented almost the same values of F-score while GAUTOSUMM performed better than other summarizers. F-scores are lower when output is limited to 5% of the original document as compared to 250 words output.

In the next section, we discuss the effects of document length on F-score of the generated summaries.

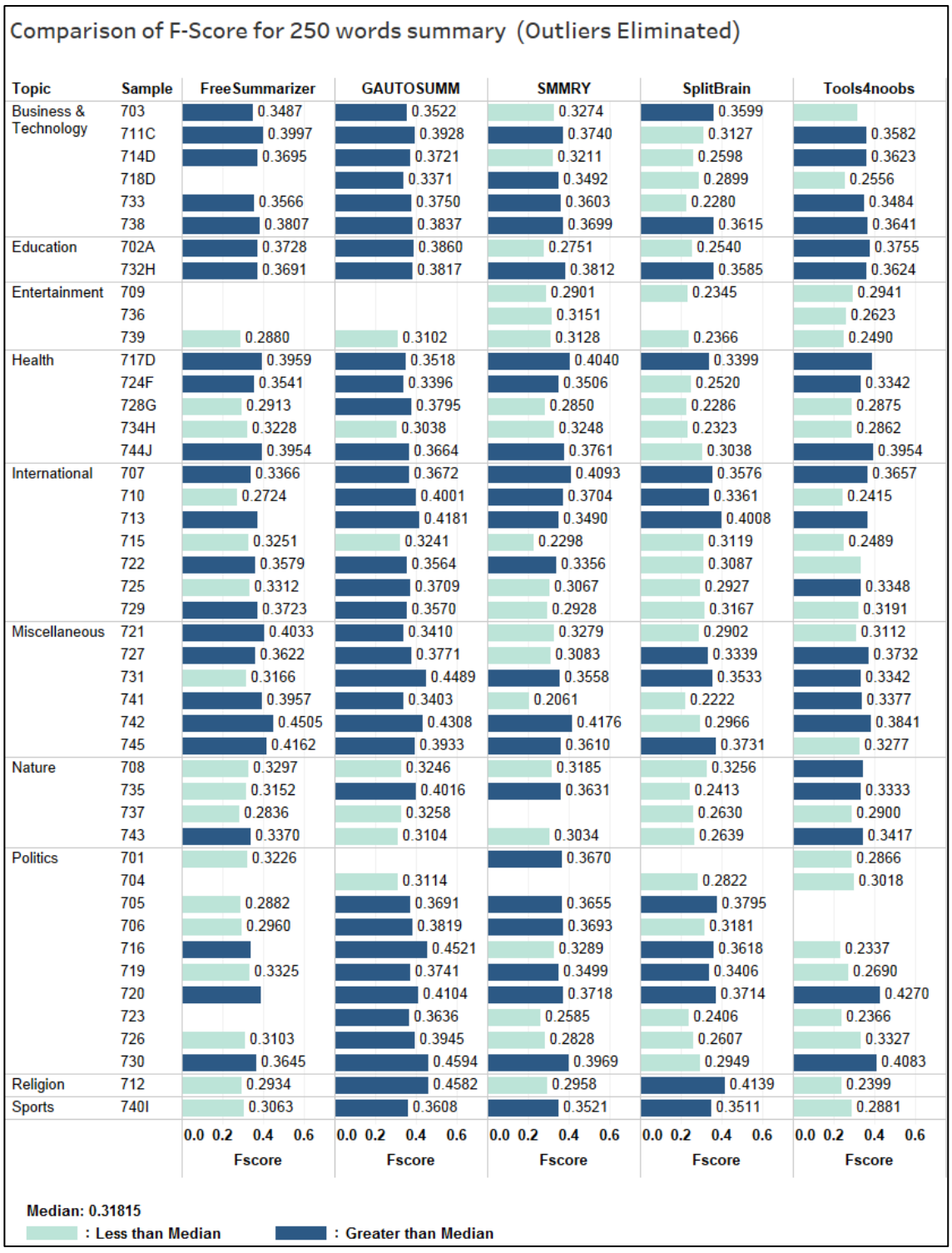

Figure 10: F-score for online text summarizers and GAUTOSUMM for 250 words output

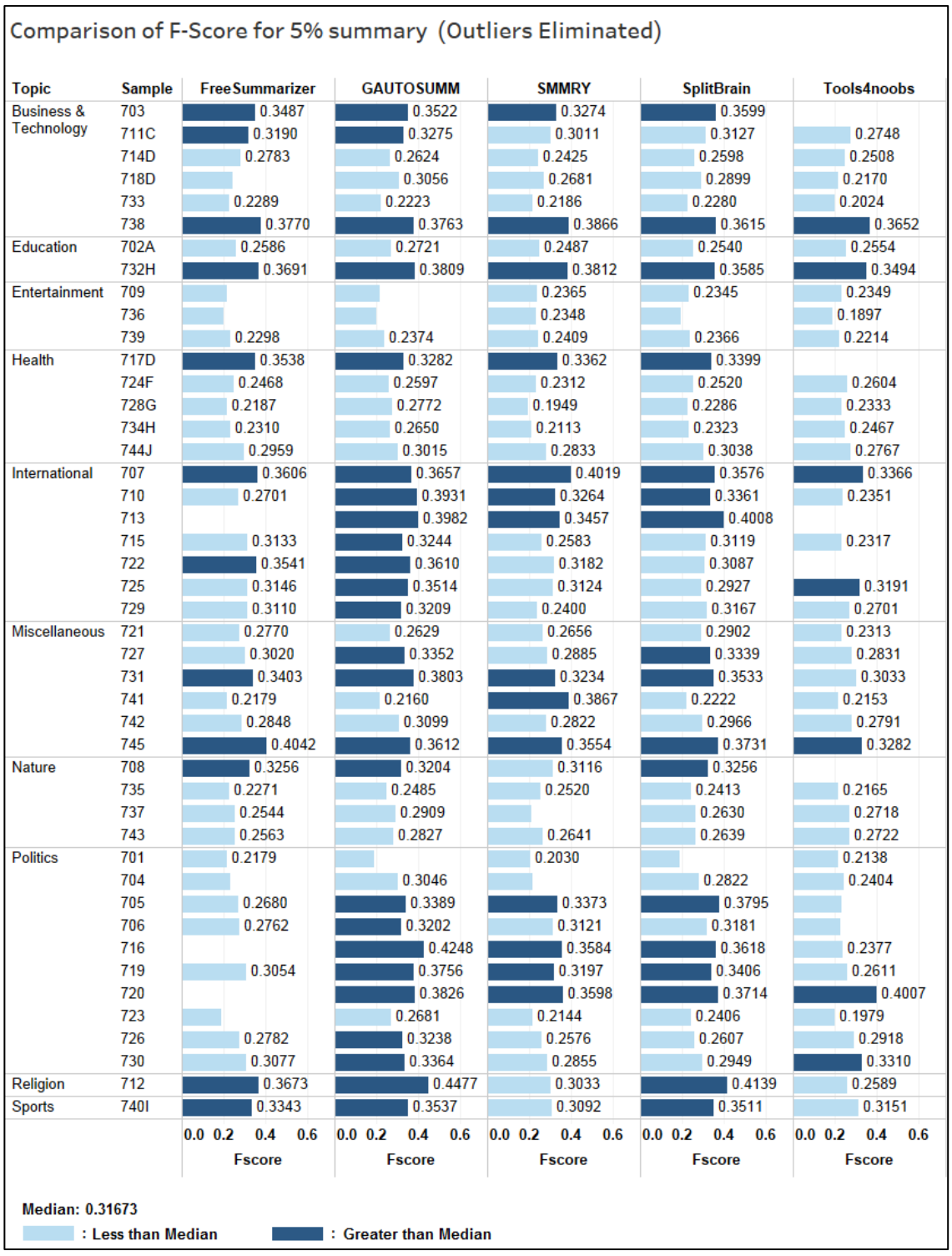

Figure 11: F-score for online text summarizers and GAUTOSUMM for 5% output

4.5.1 Document Length

Figure 12 represents the size of each sample in number of sentences which shows that sample 741 is the largest (1693 sentences) and sample 703 is the smallest (210 sentences). It is observed that summaries which are longer in length generally produce lower values of F-score.

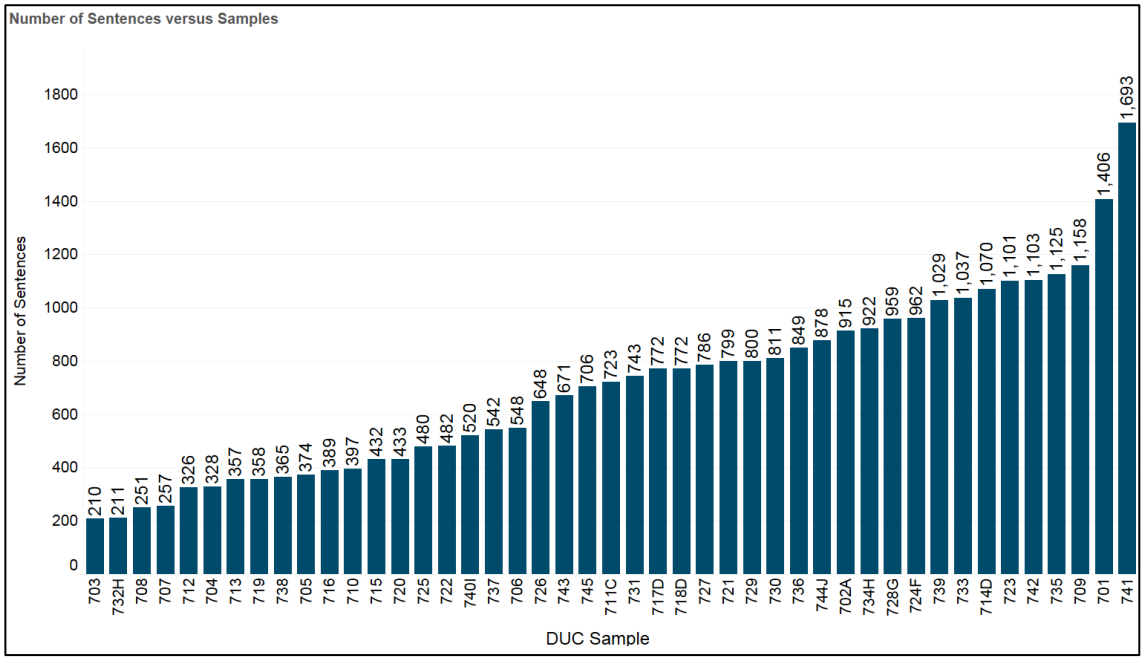

Figure 12: Size of samples with respect to increasing number of sentences

Figure 13 to Figure 17 shows the F-score, with increasing number of sentences, for all the samples producing 250 words of generated summary. From these figures, we observe that GAUTOSUMM generated good quality summaries and outperformed online text summarizers in 24 out of 45 samples. FreeSummarizer showed better results than other summarizers in 9 out of 45 samples, SMMRY was superior in 7 samples and Tools4noobs ranks lowest with 5 samples.

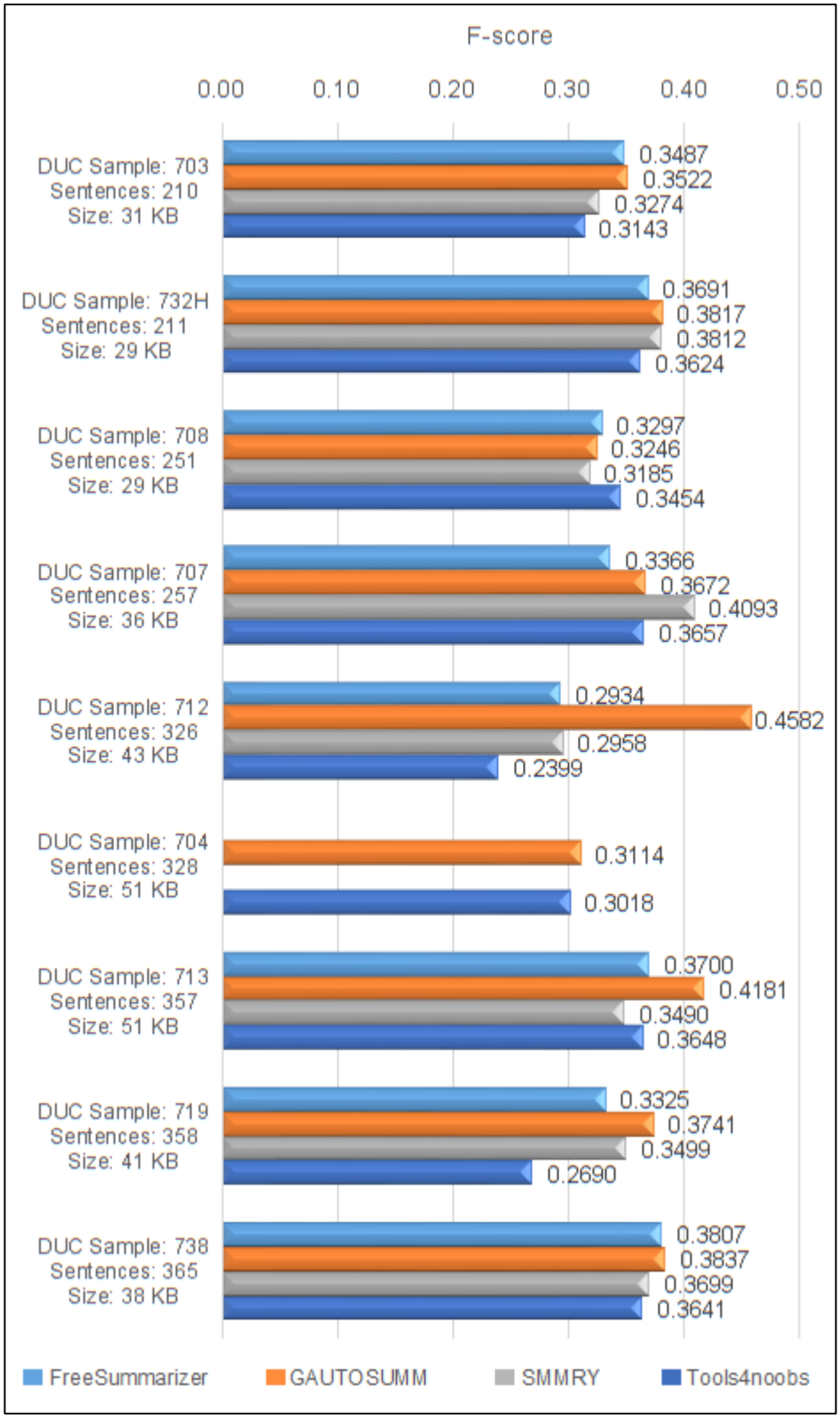

Figure 13: Comparison of F-score (250 words output) with increasing number of sentences: 210 - 365

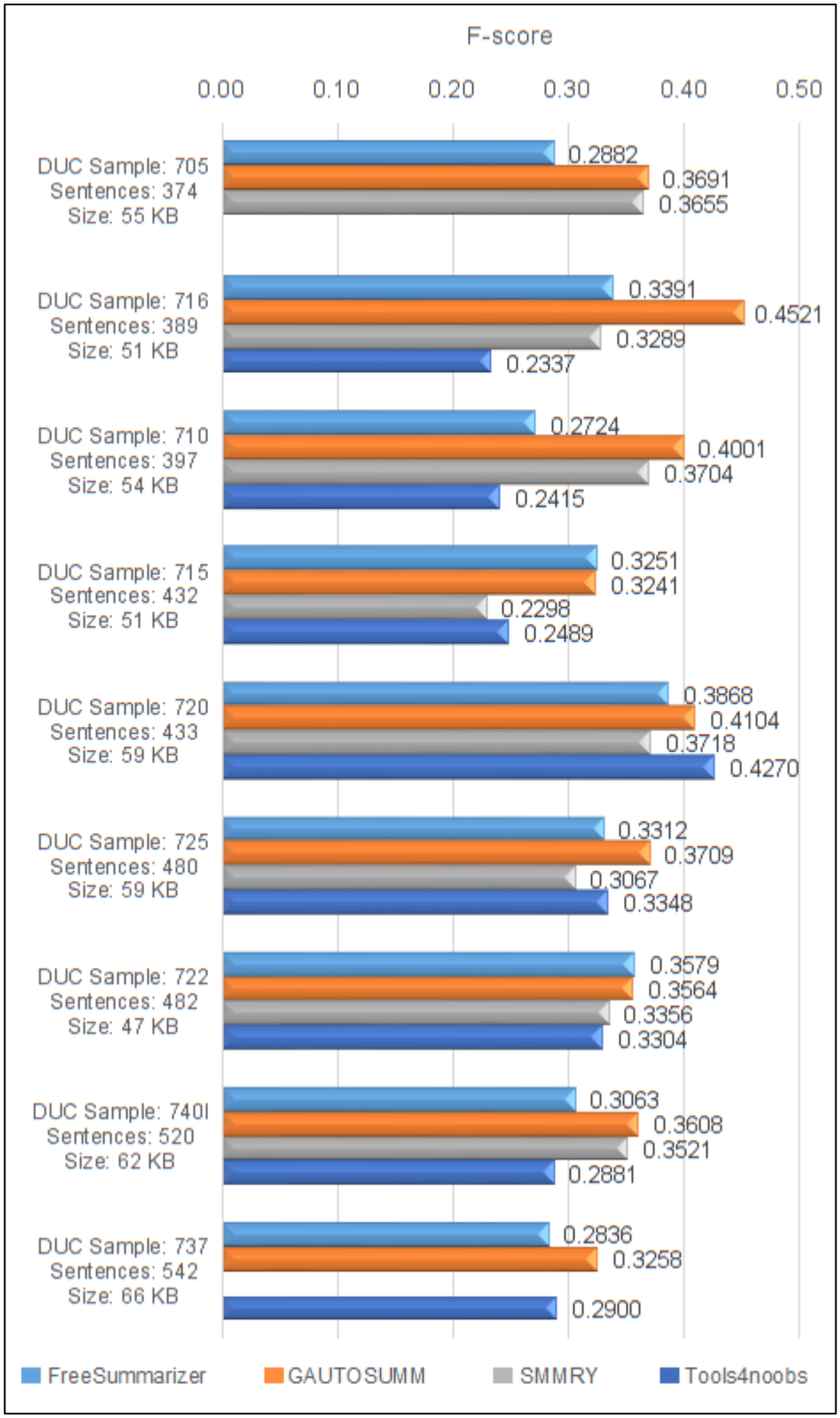

Figure 14: Comparison of F-score (250 words output) with increasing number of sentences: 374 - 542

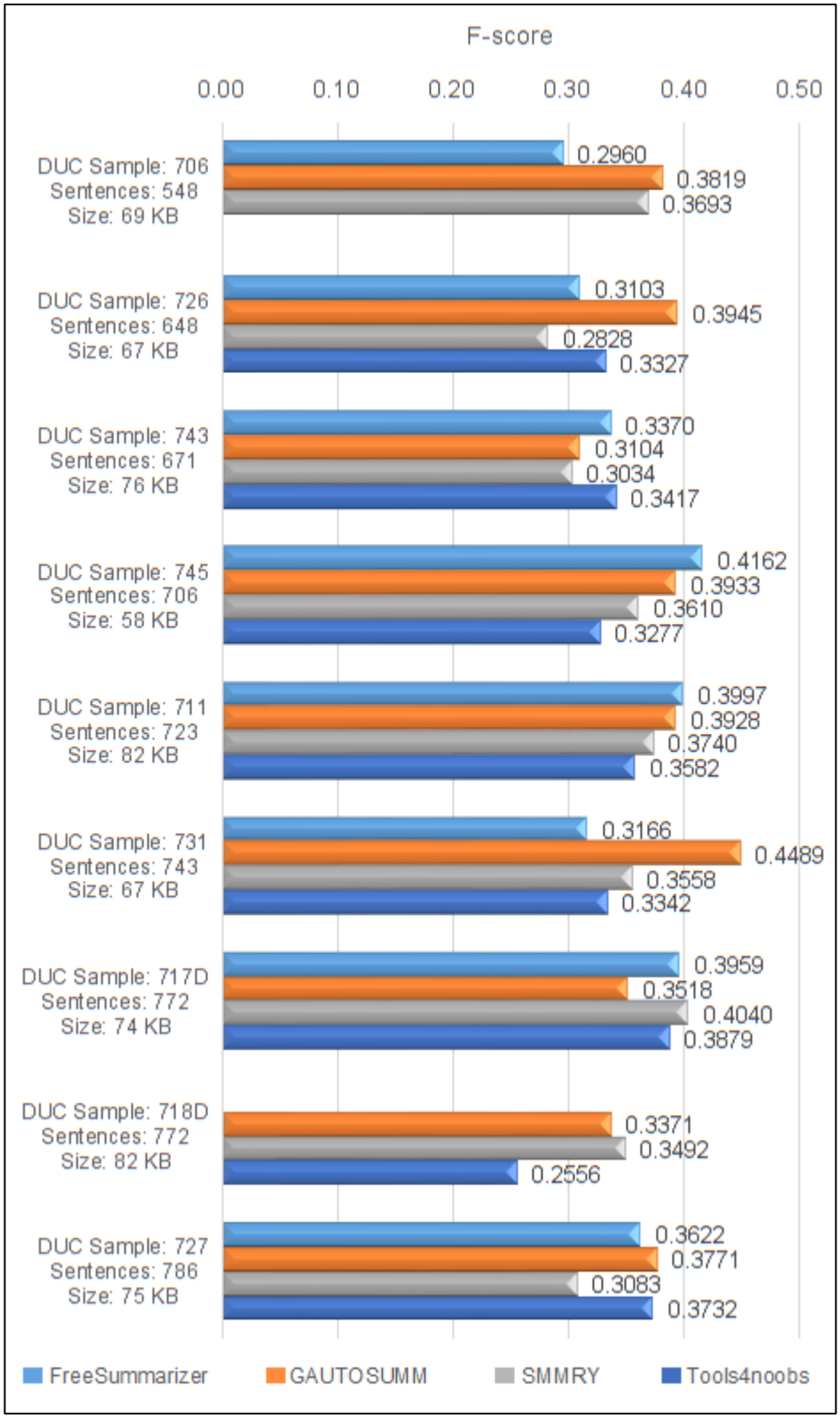

Figure 15: Comparison of F-score (250 words output) with increasing number of sentences: 548 - 786

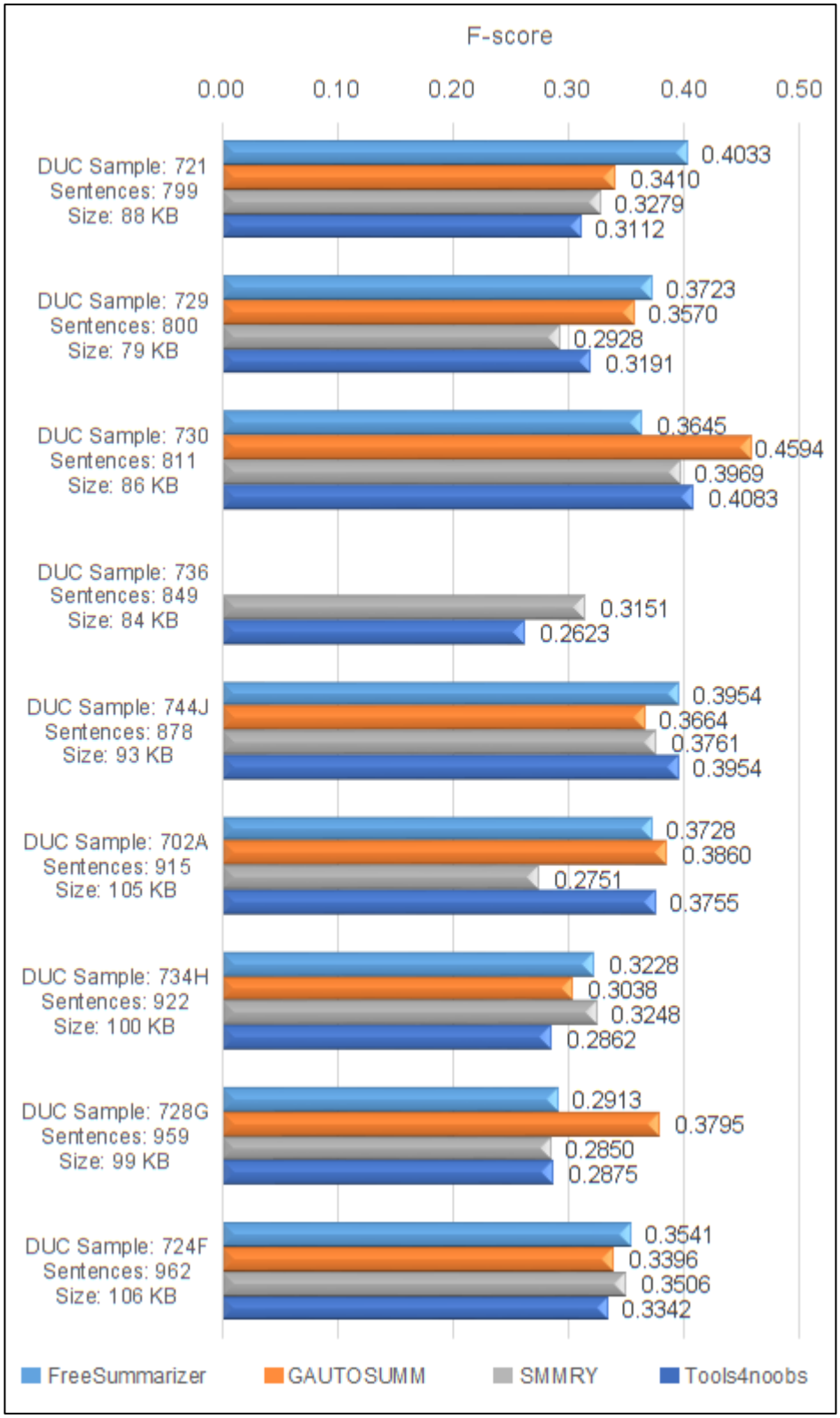

Figure 16: Comparison of F-score (250 words output) with increasing number of sentences: 799 - 962

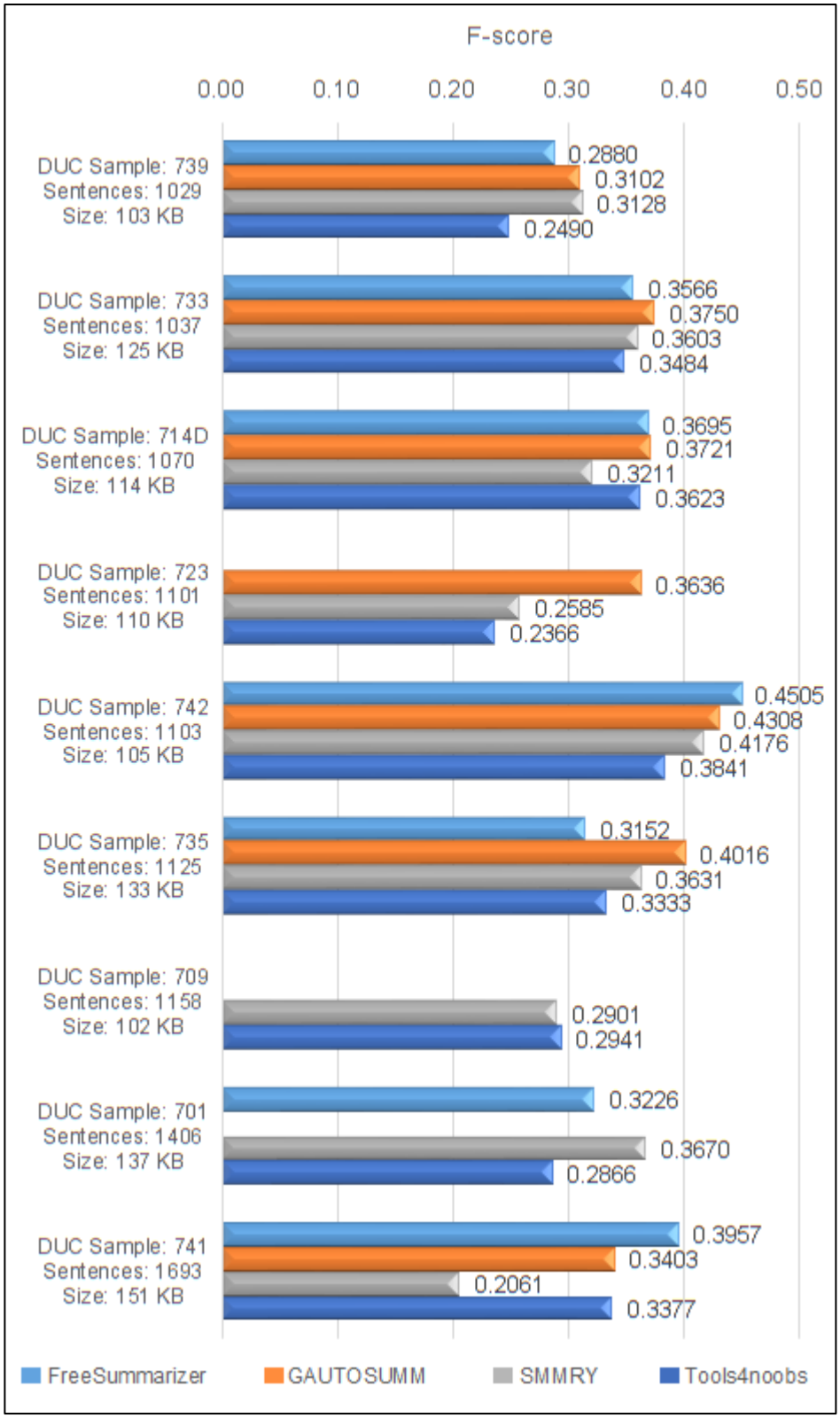

Figure 17: Comparison of F-score (250 words output) with increasing number of sentences: 1029 - 1693

In the next study, to include SplitBrain in the evaluation, summaries were restricted to 5% of the original document[1] (Figure 18 to Figure 22). Once again, GAUTOSUMM performed well in 23 out of 45 samples among all other summarizers and Tools4noobs remained the lowest with only 2 better summaries out of 45 samples. Similarly, SMMRY, FreeSummarizer and SplitBrain performed better in 9, 6 and 5 samples, respectively.

The length of a document affects the quality of generated summary. This effect is more prominent with 5% output (Figure 21 to Figure 22), where all the summarizers showed lower values of F-score as compared to 250 words output. The reason for degraded summary quality is that the original documents provided by DUC 2007 dataset are varying in length (between 210 to 1693 sentences). When the results of 5% output are calculated for samples larger than 60 KB (or approximately 500 sentences), the value of F-score drops. This happens because the evaluation metrics of ROUGE change such that recall produces higher value (for longer summaries) while precision drops by a significant amount thus decreasing the values of F-score. Ideally, for longer documents, a summarizer must generate a summary which is not affected by length of the document. The control features in GAUTOSUMM are added to overcome this problem and thus showed better results as compared to online text summarizers.

[1] SplitBrain does not allow summaries less than 5%

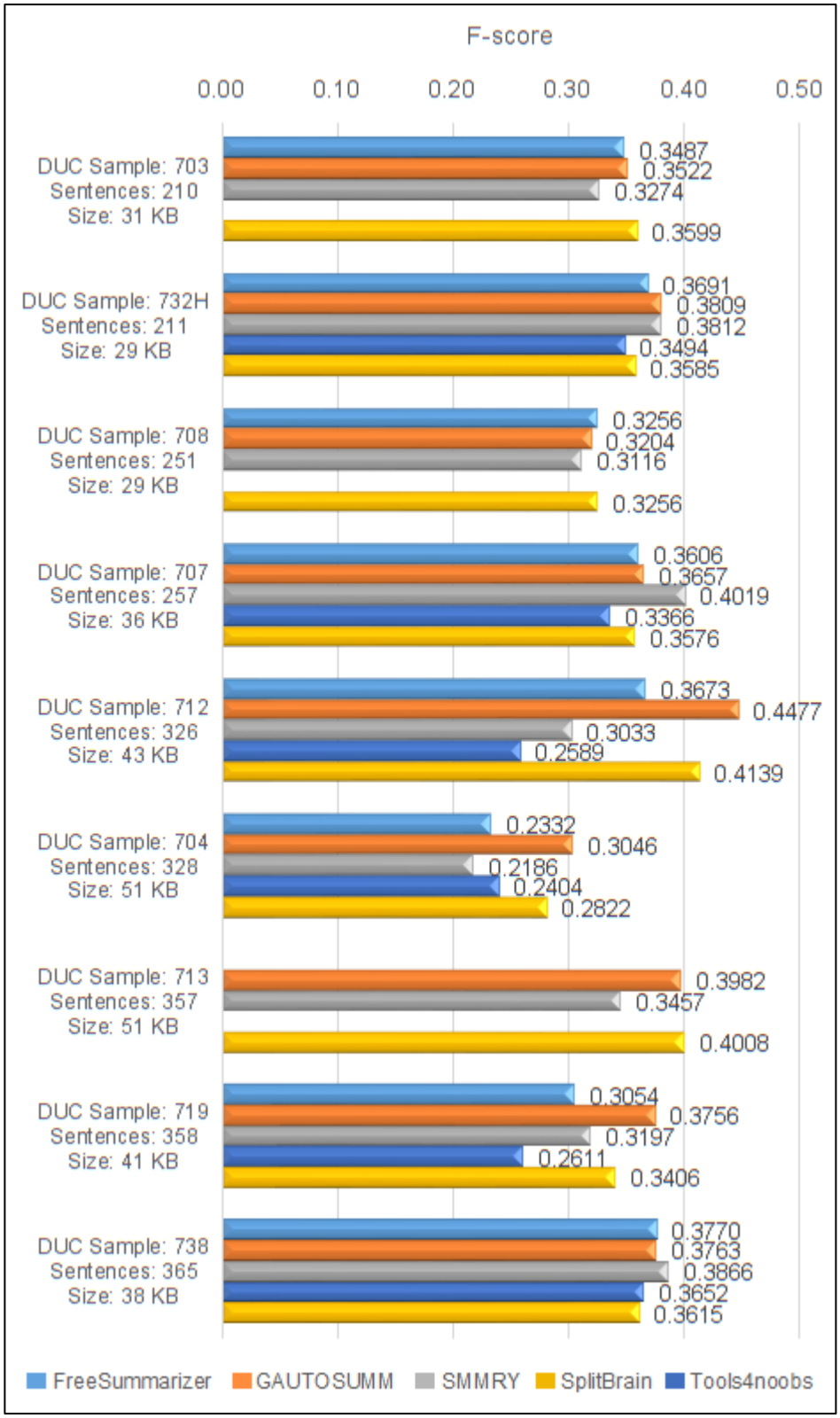

Figure 18: Comparison of F-score (5% output) with increasing number of sentences: 210 - 365

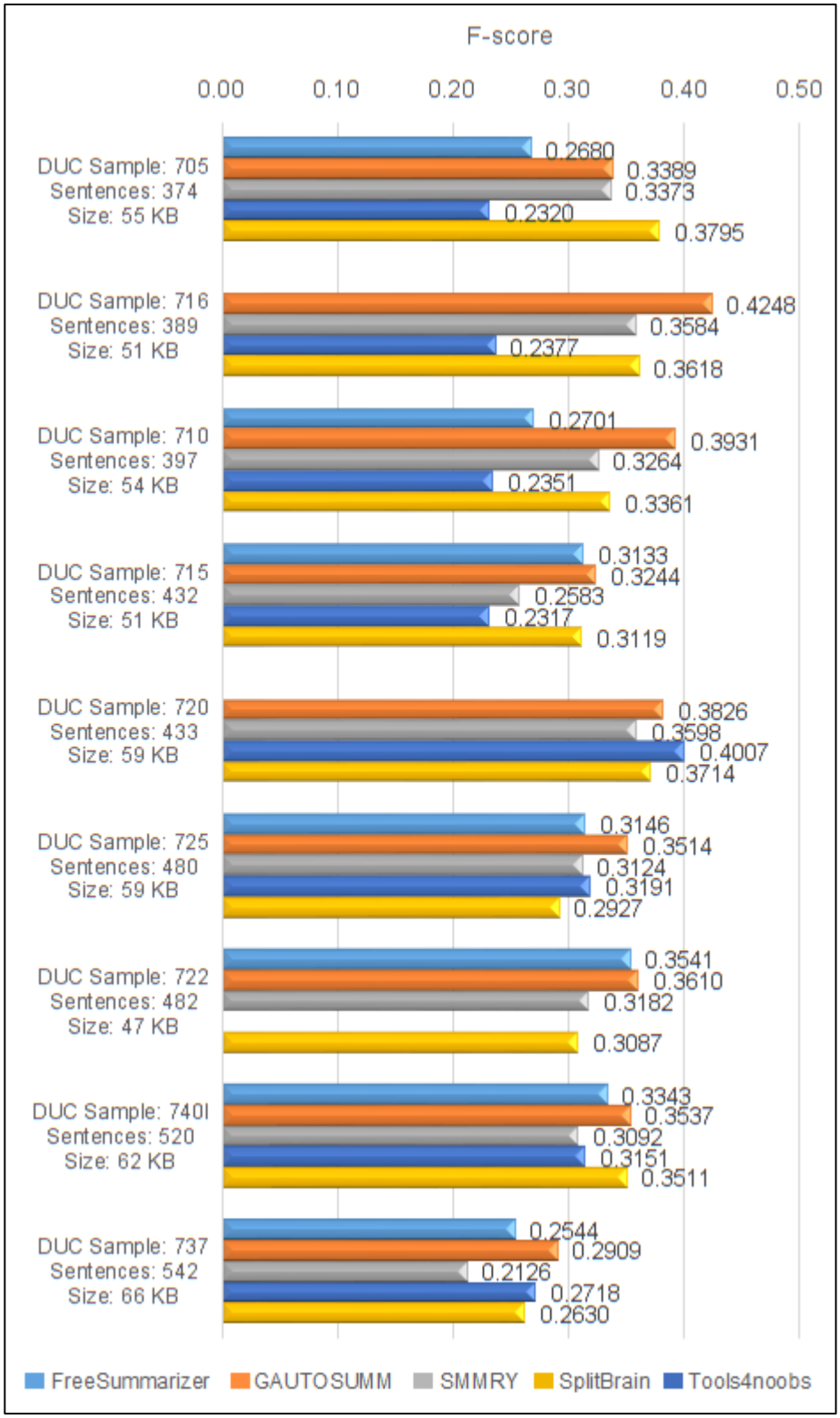

Figure 19: Comparison of F-score (5% output) with increasing number of sentences: 374 - 542

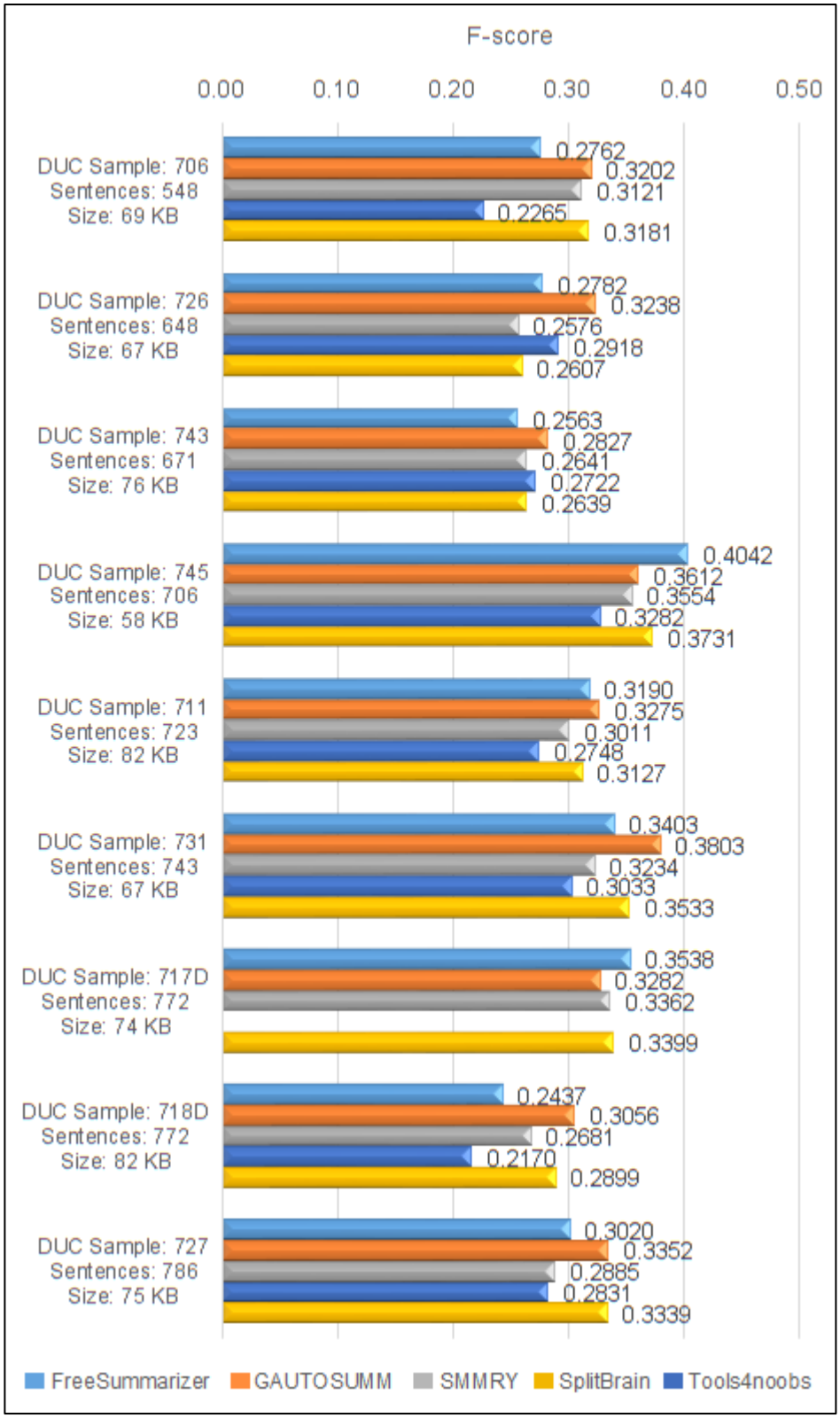

Figure 20: Comparison of F-score (5% output) with increasing number of sentences: 548 - 786

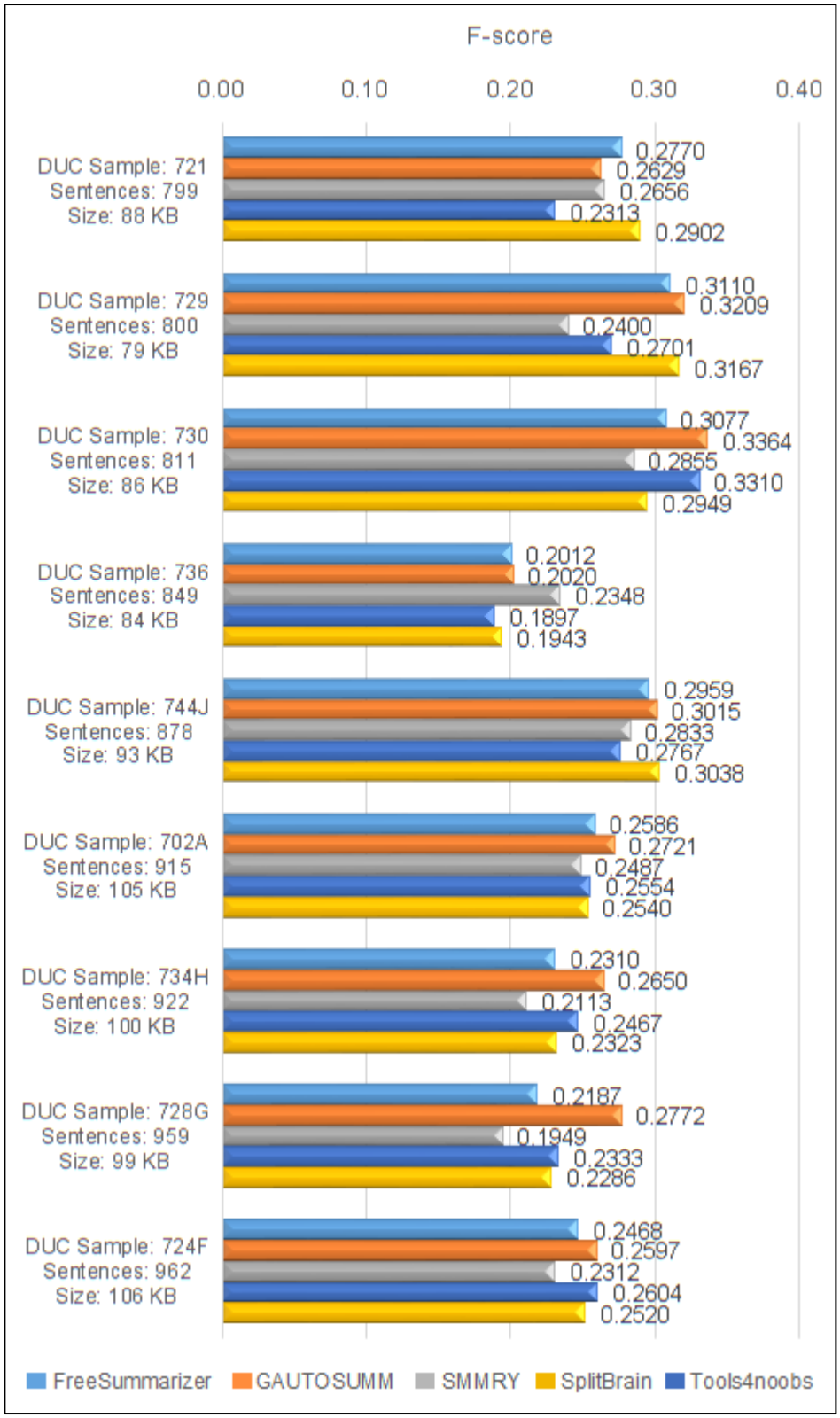

Figure 21: Comparison of F-score (5% output) with increasing number of sentences: 799 - 962

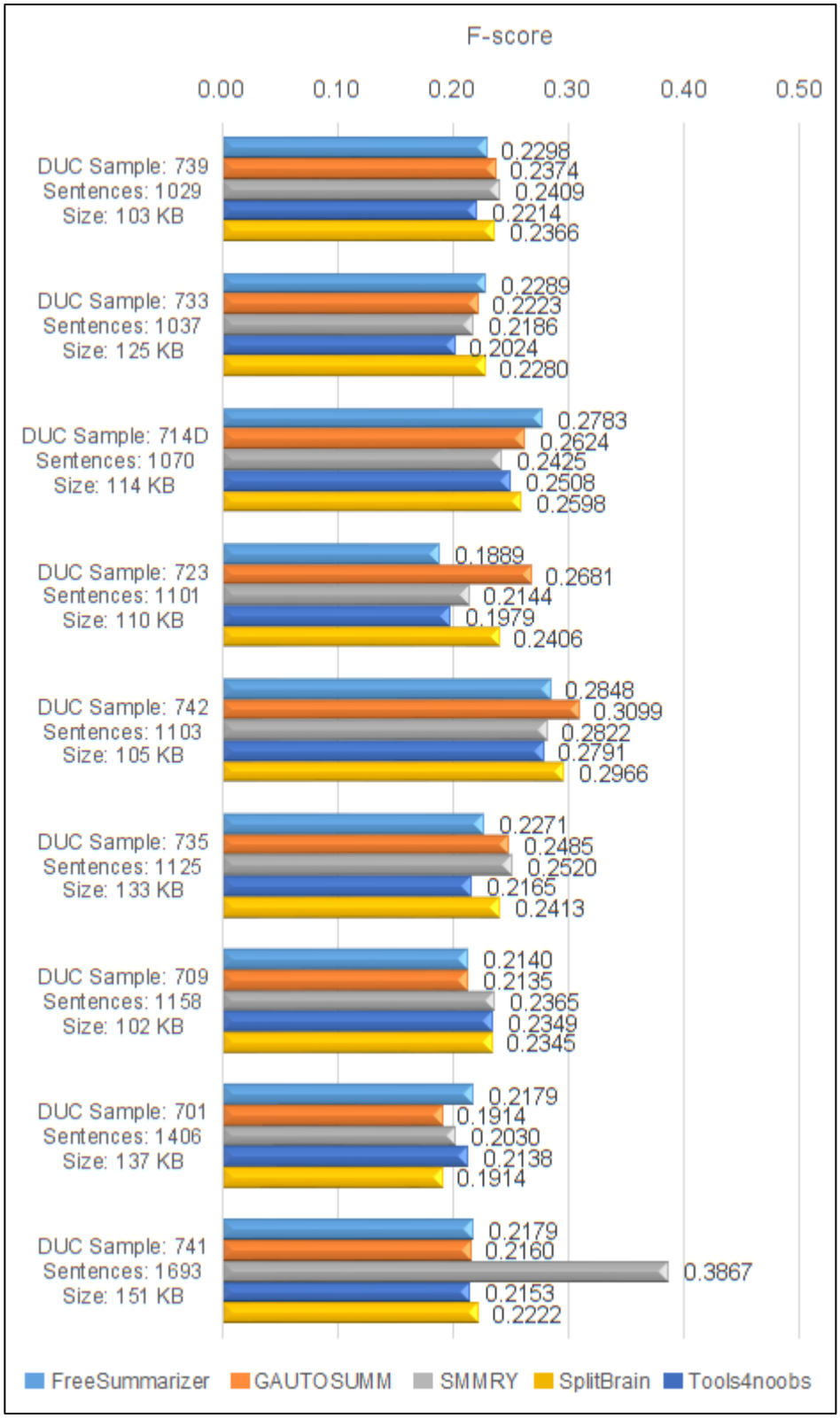

Figure 22: Comparison of F-score (5% output) with increasing number of sentences: 1029 - 1693

The outliers are eliminated from the results of all summarizers. Figure 23 illustrates the values of F-score (with outliers), as compared to the results shown in Figure 10, where outliers are removed. The percentage difference in the values of F-score is 4.01 without outliers. The figure shows a color scheme from yellow (Min) to green (Max) for 250 words of generated summary with a median at 0.31661. Even with outliers, the F-score of GAUTOSUMM for most of the samples are above the median and towards the green (Max) range.

4.5.2 Time Performance

The performance of GAUTOSUMM is measured by execution time on MATLAB. The performance of MATLAB is same on Windows or any other operating system but there are hardware dependencies such as CPU (number of processors), the bandwidth of the system bus, memory (RAM), hard disk, GPU (Graphics Processing Unit) for display and computations [85]. The execution time of GAUTOSUMM is compared between two systems: System 1 and System 2 (Figure 24). System 2 contains more cached memory, RAM and high-speed processor as compared to System 1. From the figure, we can see that execution time increases with number of sentences for both the systems. The hardware dependencies show that System 2 generated summaries 1.5 times faster than System 1. Figure 25 shows that Tools4noobs provides the best performance among all summarizers though it generated summaries with lower F-score.

Fscore - 250 words

Topic	Sample	Summarizer				
		FreeSummarizer	GAUTOSUMM	SMMRY	SplitBrain	Tools4noobs
Business & Technology	711C	0.3997	0.3928	0.3740	0.3127	0.3582
	738	0.3807	0.3837	0.3699	0.3615	0.3641
	733	0.3566	0.3750	0.3603	0.2280	0.3484
	714D	0.3695	0.3721	0.3211	0.2598	0.3623
	703	0.3487	0.3522	0.3274	0.3599	0.3143
	718D	0.1917	0.3371	0.3492	0.2899	0.2556
Education	702A	0.3728	0.3860	0.2751	0.2540	0.3755
	732H	0.3691	0.3817	0.3812	0.3585	0.3624
Entertainment	739	0.2880	0.3102	0.3128	0.2366	0.2490
	736	0.2478	0.2669	0.3151	0.1943	0.2623
	709	0.1867	0.1872	0.2901	0.2345	0.2941
Health	728G	0.2913	0.3795	0.2850	0.2286	0.2875
	744J	0.3954	0.3664	0.3761	0.3038	0.3954
	717D	0.3959	0.3518	0.4040	0.3399	0.3879
	724F	0.3541	0.3396	0.3506	0.2520	0.3342
	734H	0.3228	0.3038	0.3248	0.2323	0.2862
International	713	0.3700	0.4181	0.3490	0.4008	0.3648
	710	0.2724	0.4001	0.3704	0.3361	0.2415
	725	0.3312	0.3709	0.3067	0.2927	0.3348
	707	0.3366	0.3672	0.4093	0.3576	0.3657
	729	0.3723	0.3570	0.2928	0.3167	0.3191
	722	0.3579	0.3564	0.3356	0.3087	0.3304
	715	0.3251	0.3241	0.2298	0.3119	0.2489
Miscellaneous	731	0.3166	0.4489	0.3558	0.3533	0.3342
	742	0.4505	0.4308	0.4176	0.2966	0.3841
	745	0.4162	0.3933	0.3610	0.3731	0.3277
	727	0.3622	0.3771	0.3083	0.3339	0.3732
	721	0.4033	0.3410	0.3279	0.2902	0.3112
	741	0.3957	0.3403	0.2061	0.2222	0.3377
Nature	735	0.3152	0.4016	0.3631	0.2413	0.3333
	737	0.2836	0.3258	0.2319	0.2630	0.2900
	708	0.3297	0.3246	0.3185	0.3256	0.3454
	743	0.3370	0.3104	0.3034	0.2639	0.3417
Politics	730	0.3645	0.4594	0.3969	0.2949	0.4083
	716	0.3391	0.4521	0.3289	0.3618	0.2337
	720	0.3868	0.4104	0.3718	0.3714	0.4270
	726	0.3103	0.3945	0.2828	0.2607	0.3327
	706	0.2960	0.3819	0.3693	0.3181	0.2095
	719	0.3325	0.3741	0.3499	0.3406	0.2690
	705	0.2882	0.3691	0.3655	0.3795	0.2232
	723	0.2133	0.3636	0.2585	0.2406	0.2366
	704	0.2246	0.3114	0.1981	0.2822	0.3018
	701	0.3226	0.2649	0.3670	0.1914	0.2866
Religion	712	0.2934	0.4582	0.2958	0.4139	0.2399
Sports	740I	0.3063	0.3608	0.3521	0.3511	0.2881

Fscore :

Median (0.31661)

Min Max

0.1867 0.4594

Figure 23: F-score showing outliers of online summarizers and GAUTOSUMM for 250 words

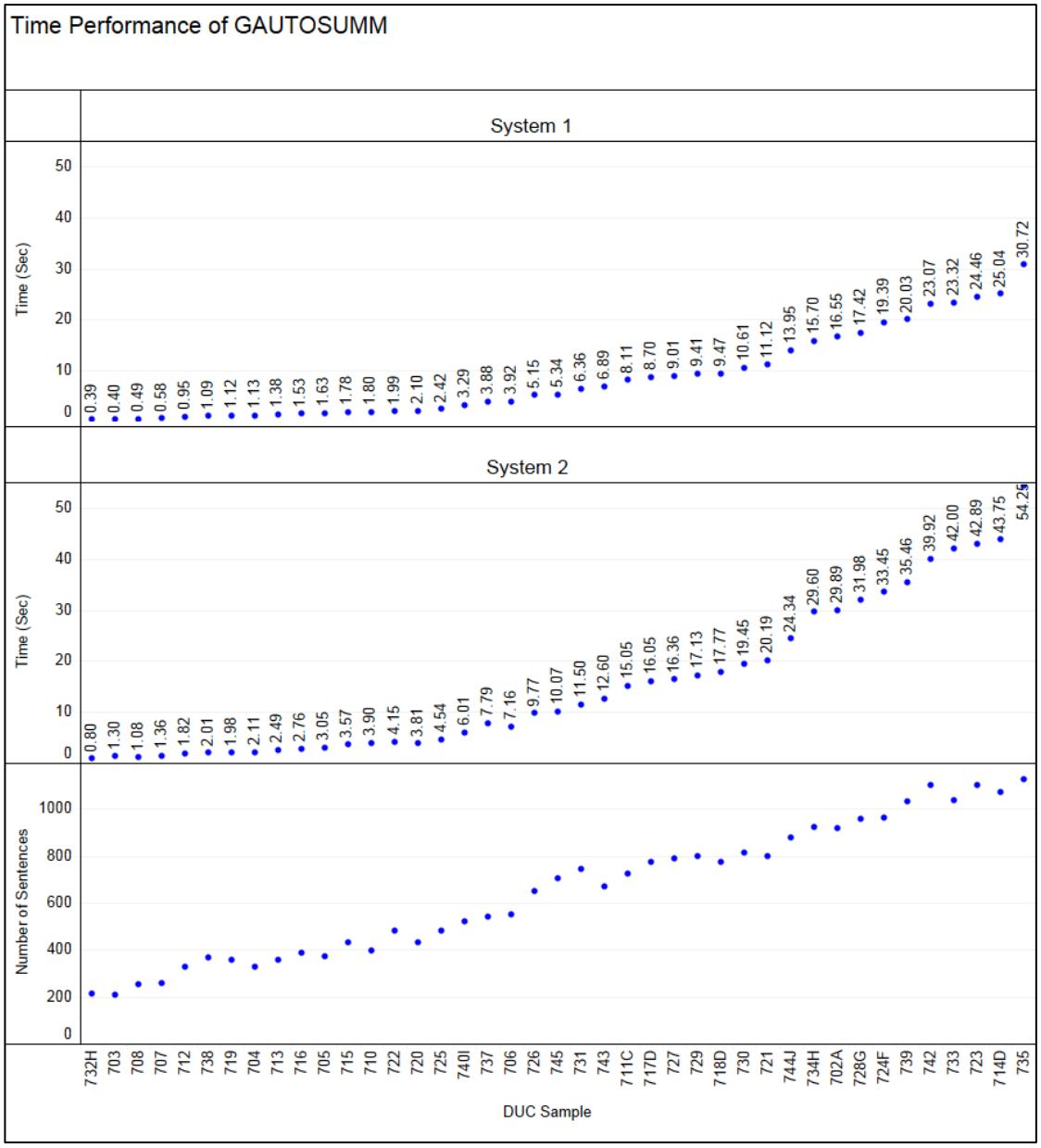

Figure 24: Comparison of time performance for GAUTOSUMM

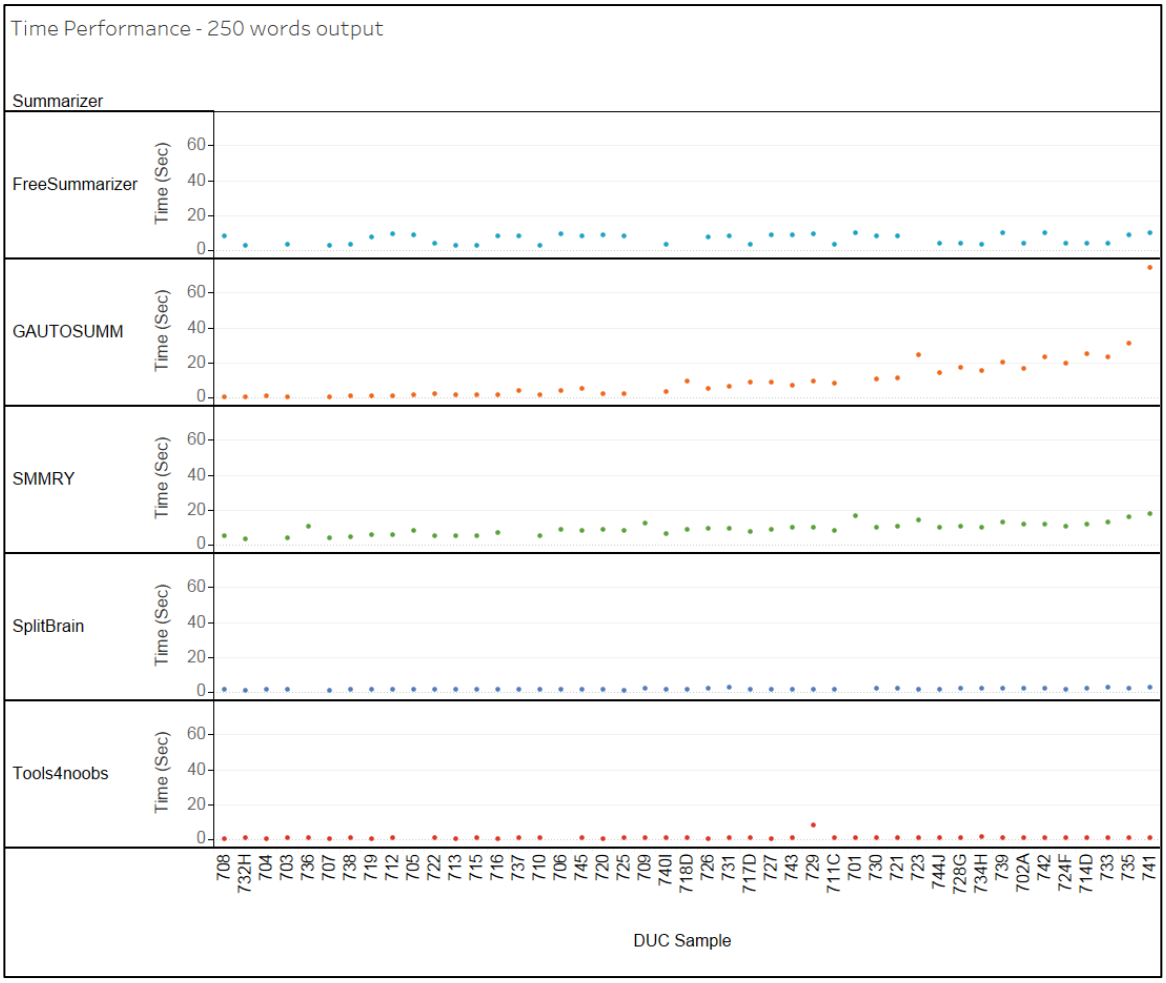

Figure 25: Time Performance of online text summarizers and GAUTOSUMM

4.5.3 Impact of pre-processing

Three levels of pre-processing were examined for the proposed summarizer: 1) Extensive pre-processing (removal of all lexical categories), 2) Basic pre-processing (removal of articles, prepositions, and conjunctions) and 3) No pre-processing. Figure 26 to Figure 28 shows the impact of pre-processing on execution time for all 45 samples for the three levels of pre-processing with increasing number of sentences. We observe that without pre-processing, the largest sample 741 took 82 seconds to generate the output summary and the smallest sample 703 took 0.41 seconds. The difference in execution time is a result of an increase in number of

sentences which in turn generates more edges. Extensive and basic pre-processing reduces the execution time because the number of edges are reduced after the removal of lexical categories (Table 8). Figure 29 provides a different perspective of the same observations. The samples are represented by circles of corresponding size. Most of the samples provided by DUC 2007 are large which impacts the number of sentences and edges. By including a pre-processing step, number of edges are greatly reduced and subsequently, the time performance of the proposed summarizer is improved. Without pre-processing, the largest sample of DUC 2007, 741, contributed towards more than five million edges, whereas with basic pre-processing, number of edges are reduced to almost one million. Extensive pre-processing further reduced this number to around 891,000 (Table 7). The smallest sample, 703, generated over three million edges without pre-processing and the number of edges are reduced to 58,717 with pre-processing and then to 55,942 with extensive pre-processing. These figures clearly demonstrate the importance of pre-processing step and its impact on time performance of GAUTOSUMM.

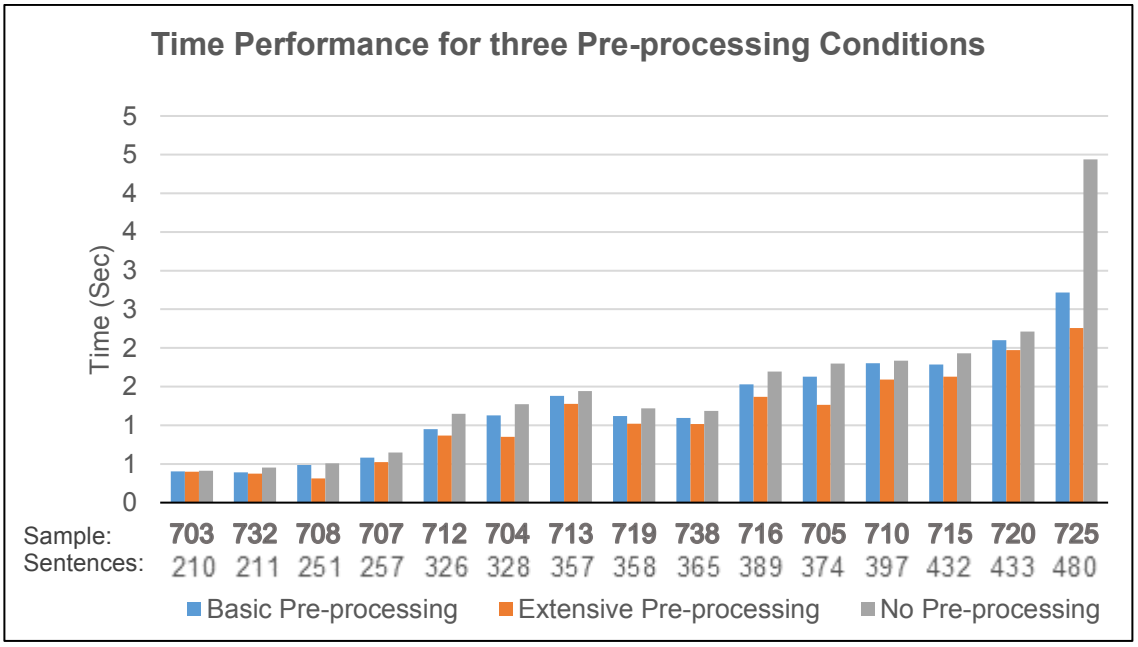

Figure 26: Impact of pre-processing on execution time with increasing number of sentences: 210 - 480

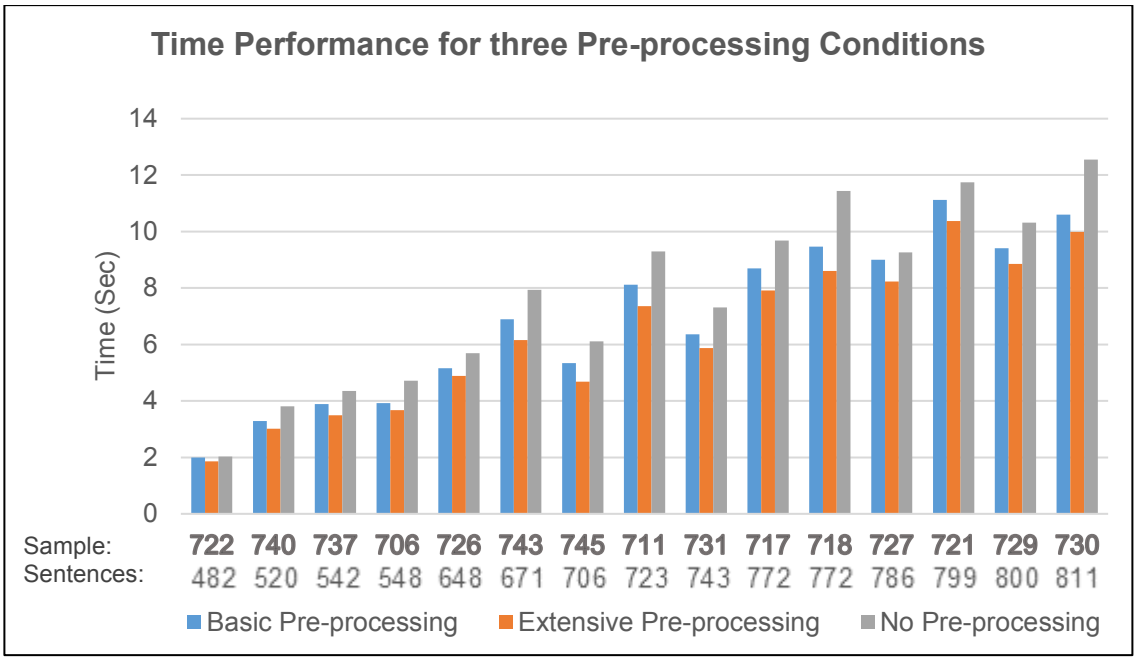

Figure 27: Impact of pre-processing on execution time with increasing number of sentences: 482 – 811

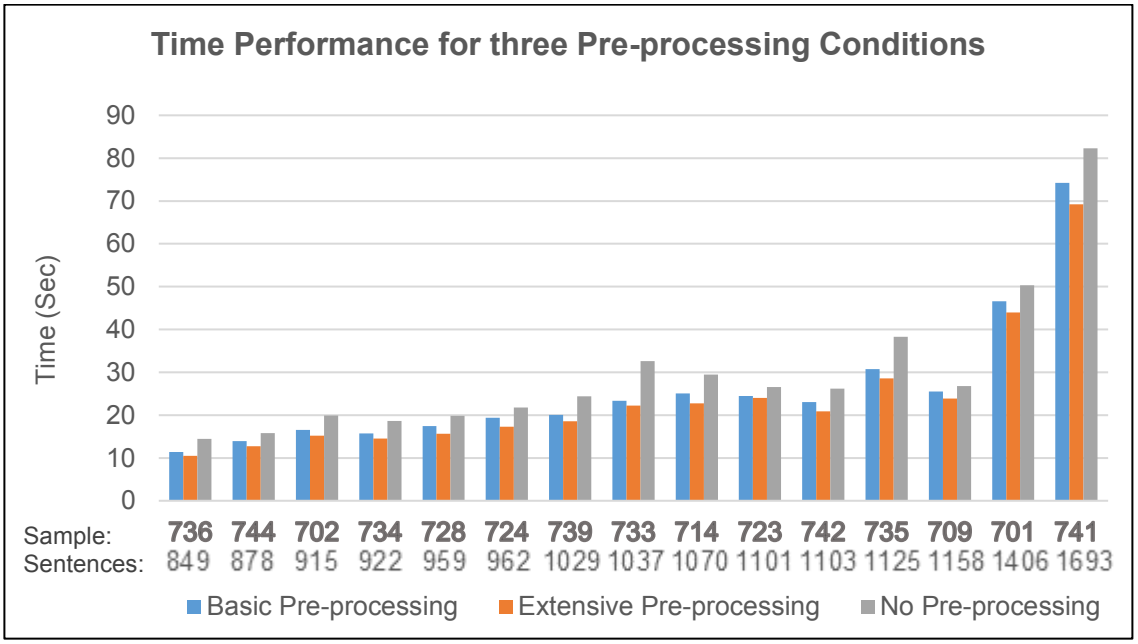

Figure 28: Impact of pre-processing on execution time with increasing number of sentences: 849 – 1693

Impact of Pre-processing on Number of Edges			
DUC Sample	Condition		
	Extensive Pre-processing	Basic Pre-processing	No Pre-processing
741	891,349	1,129,390	5,091,039
701	303,732	404,821	4,741,800
735	403,057	498,813	4,590,733
733	217,587	266,995	4,580,376
742	352,740	432,039	2,860,243
714D	401,987	494,078	2,778,699
702A	201,139	238,554	2,399,793
723	255,830	335,670	2,302,113
724F	239,883	294,840	2,257,691
730	243,868	268,852	2,244,683
739	230,671	274,788	2,101,225
728G	223,258	275,947	1,981,753
744J	252,288	300,525	1,772,736
734H	187,277	228,106	1,735,572
721	181,671	228,105	1,656,932
709	197,495	234,542	1,576,634
711C	218,424	244,491	1,493,050
729	223,842	289,699	1,437,863
743	106,690	148,725	1,338,808
717D	205,754	258,420	1,310,189
737	58,693	82,260	1,242,874
718D	152,543	187,153	1,181,975
736	123,002	143,510	1,155,309
706	88,484	105,099	1,127,743
720	115,623	126,391	1,106,930
727	273,412	301,252	1,106,749
740I	83,007	104,898	921,201
726	175,190	197,009	920,740
731	151,630	169,354	871,656
745	123,104	156,173	785,483
725	139,053	156,068	760,831
705	117,802	124,751	677,664
715	92,922	106,627	666,101
710	112,898	142,992	658,918
713	131,678	147,272	616,452
716	82,729	93,313	611,278
722	137,601	147,521	600,437
719	51,599	60,573	478,967
704	72,923	83,773	476,670
707	69,916	74,788	395,708
712C	68,945	78,393	378,157
703	55,942	58,717	335,401
738	41,781	47,353	319,760
708	77,176	80,595	289,951
732H	28,966	31,953	238,912

Table 7: Impact of pre-processing on number of edges for GAUTOSUMM

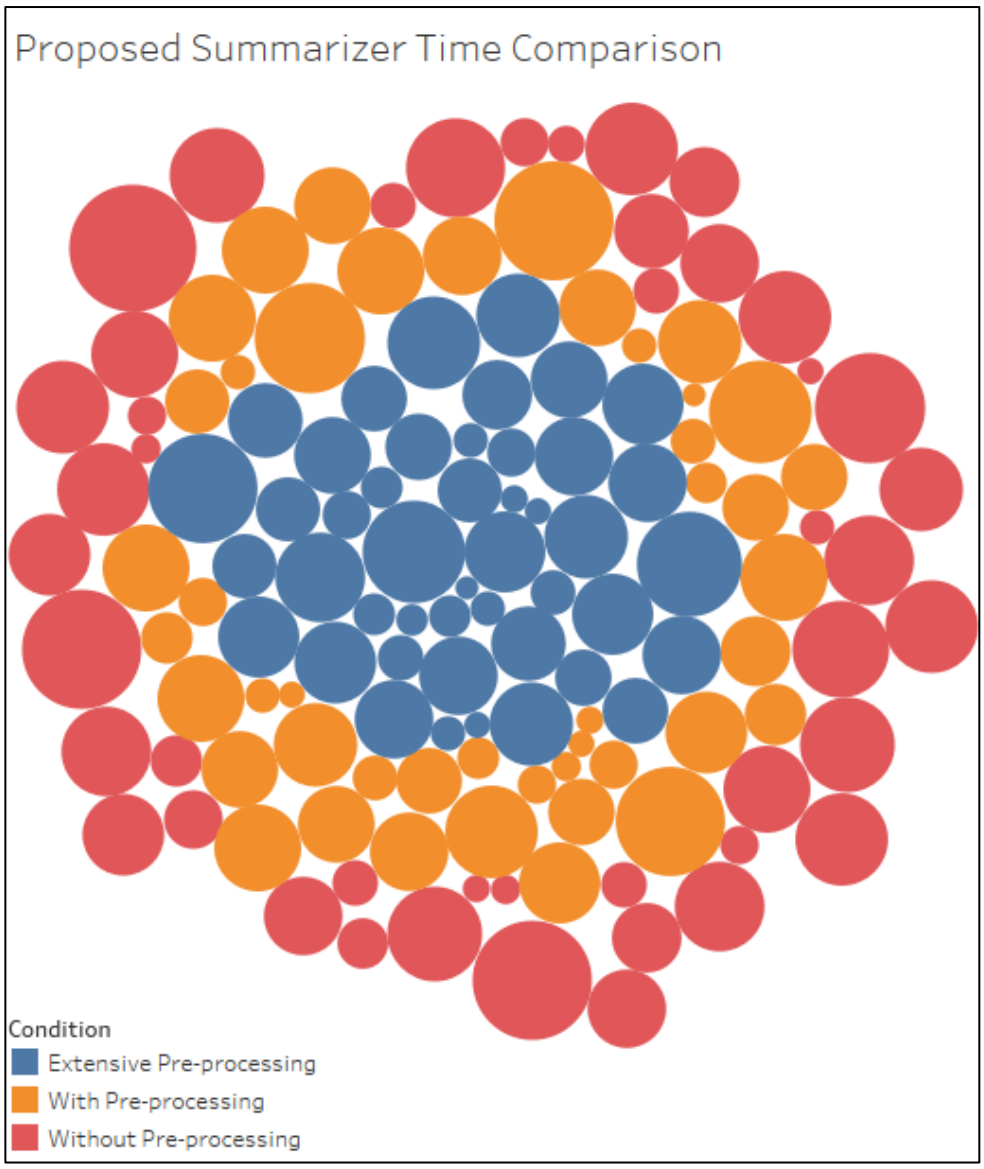

Figure 29: Execution time comparison for three levels of pre-processing: Extensive pre-processing, Basic pre-processing and No Pre-processing

Time Performance

DUC Sample	Number of Sentences	Size (KB)	Edges	Time (Sec)
732H	211	29	31,953	0.39
703	210	31	58,717	0.40
708	251	29	80,595	0.49
707	257	36	74,788	0.58
712	326	43	78,393	0.95
738	365	38	47,353	1.09
719	358	41	60,573	1.12
704	328	51	83,773	1.13
713	357	51	147,272	1.38
716	389	51	93,313	1.53
705	374	55	124,751	1.63
715	432	51	106,627	1.78
710	397	54	142,992	1.80
722	482	47	147,521	1.99
720	433	59	126,391	2.10
725	480	59	156,068	2.42
740I	520	62	104,898	3.29
737	542	66	82,260	3.88
706	548	69	105,099	3.92
726	648	67	197,009	5.15
745	706	58	156,173	5.34
731	743	67	169,354	6.36
743	671	76	148,725	6.89
711C	723	82	244,491	8.11
717D	772	74	258,420	8.70
727	786	75	301,252	9.01
729	800	79	289,699	9.41
718D	772	82	187,153	9.47
730	811	86	268,852	10.61
721	799	88	228,105	11.12
744J	878	93	300,525	13.95
734H	922	100	228,106	15.70
702A	915	105	238,554	16.55
728G	959	99	275,947	17.42
724F	962	106	294,840	19.39
739	1,029	103	274,788	20.03
742	1,103	105	432,039	23.07
733	1,037	125	266,995	23.32
723	1,101	110	335,670	24.46
714D	1,070	114	494,078	25.04
735	1,125	133	498,813	30.72
741	1,693	151	1,129,390	74.23

Table 8: Number of Edges and Sentences of samples together with execution time of GAUTOSUMM.

4.5.4 Complexity of GAUTOSUMM

Our proposed summarizer uses a Graph-based approach whose performance is dependent on the number of nodes and edges. The complexity of GAUTOSUMM is O (N x M), where N is a number of sentences (or nodes) and M is the number of shared words (or edges). Typically, the number of sentences is less than the number of words in a document. The number of sentences versus edges for all the samples of DUC 2007 dataset are shown in Figure 30.

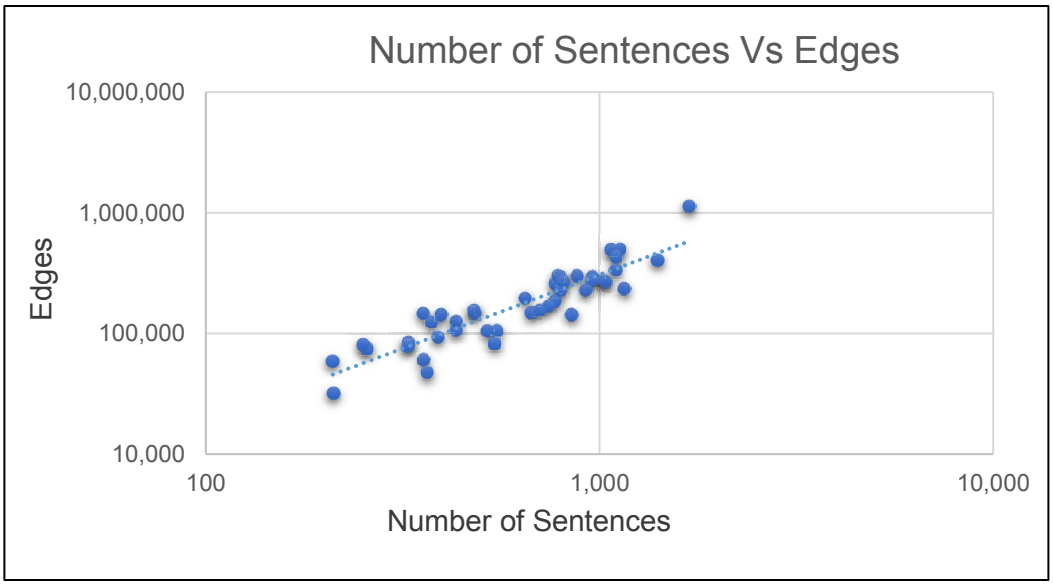

Figure 30: Number of Sentences and Edges for samples from DUC 2007 dataset

To elaborate a relation between number of sentences and edges, the samples are further grouped and plotted on a log-log scale with the best fit line (Figure 31). From the figure we can infer that, for this selected dataset, the number of edges increases with the number of sentences, with an exponent of 1.13. The value of R^2 shows that data points are 96.63% closer to the best fit line and number of sentences versus edges are linearly related.

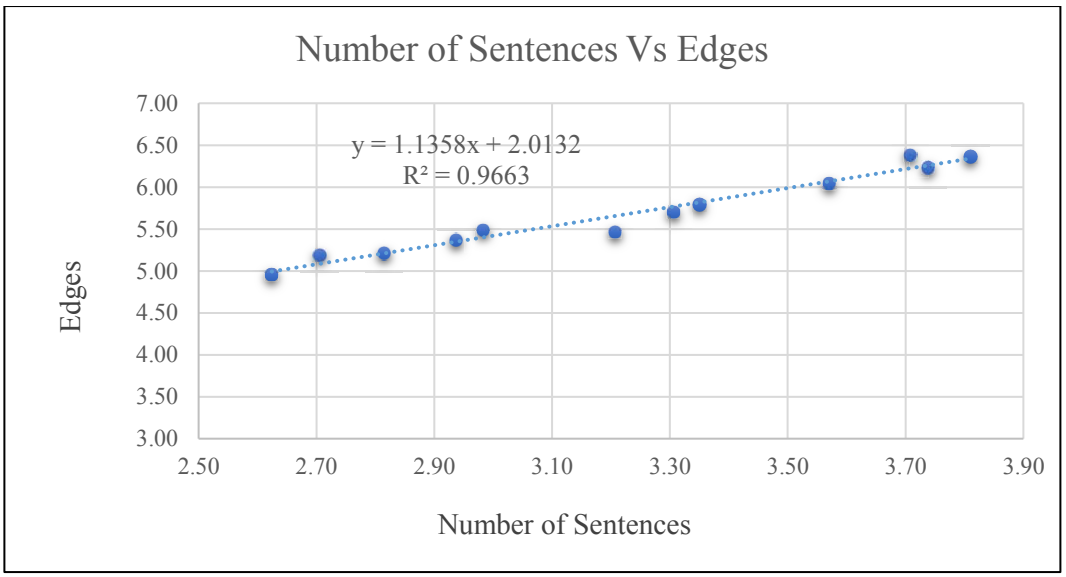

Figure 31: Complexity of GAUTOSUMM

Figure 32 shows a relation between number of sentences and execution time on a log-log scale. As compared to Figure 31, the slope of the line is linear but steeper with a value of 2.59.

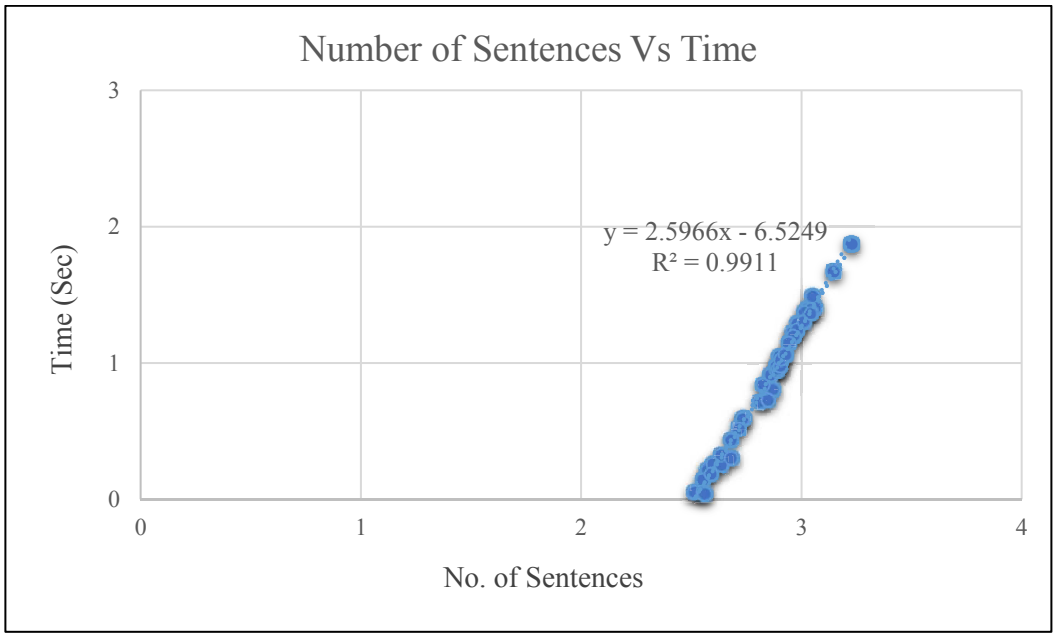

Figure 32: Complexity of GAUTOSUMM (Sentences Vs Time)

Figure 33 displays a trend between number of sentences and edges for all the samples where pre-processing was used. As discussed in the previous section, the number of edges increases by a significant amount without pre-processing and this impact can also be observed in Figure 34.

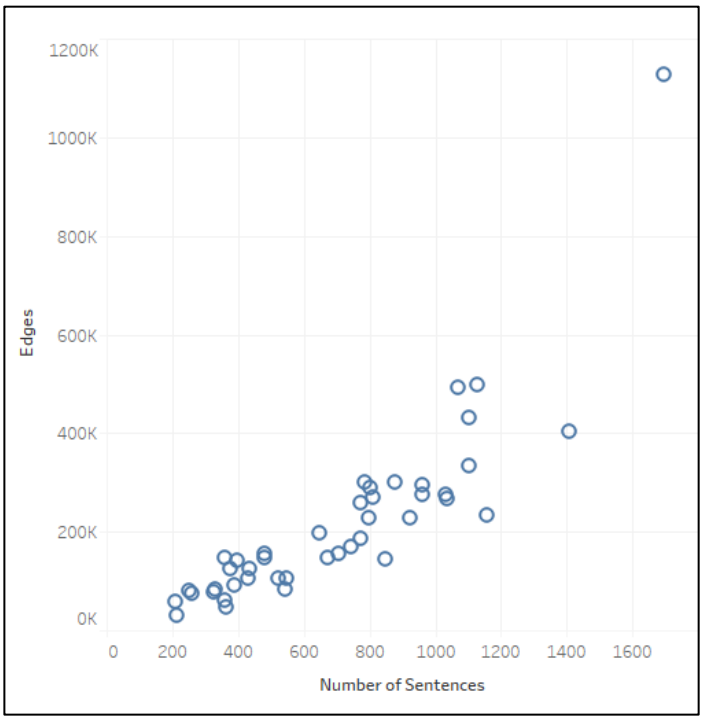

Figure 33: Number of Sentences Vs Number of Edges with basic pre-processing

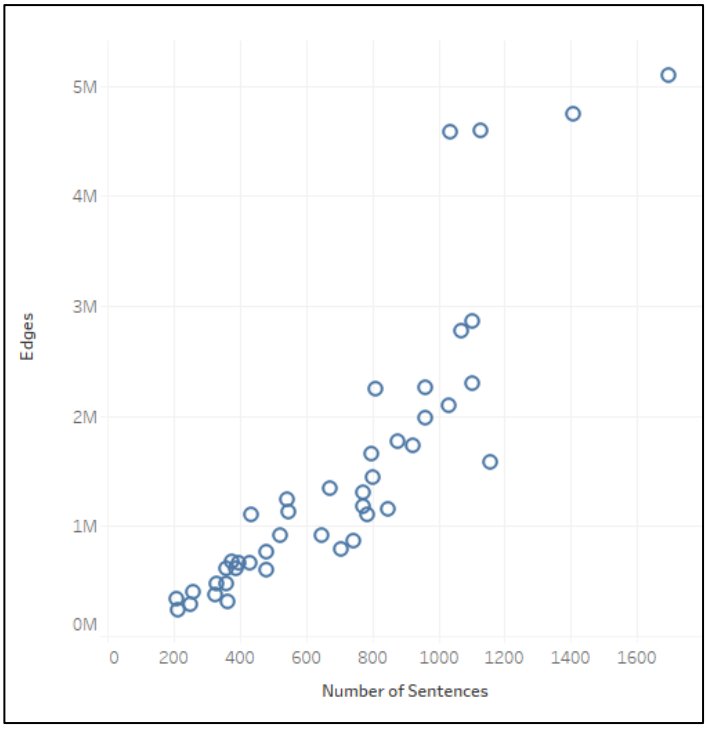

Figure 34: Number of Sentences Vs Number of Edges with no pre-processing

In this chapter, we discussed the evaluation and results of all the summarizers based on ROUGE metrics. Two datasets, Opinosis and DUC 2007, were used for the evaluation of generated summaries from online text summarizers and GAUTOSUMM. A brief introduction to the online text summarizers was also presented. Results are discussed for all summarizers using an Opinosis dataset. Finally, evaluation and results obtained from the summaries generated using DUC 2007 dataset were presented. These results showed improvements in ROUGE metric values after implementation of the pre-processing module, control features and post-processing module. The next chapter concludes this work and provides direction for future research.

Chapter 5

Conclusion and Future Direction

The need for text summarization has been long known but it has now become even more important due to large amount of data accessible on the Internet. In this thesis, we presented the concepts behind text summarization, related work and different approaches taken to generate summaries. Over the years, many methods have been proposed by researchers to produce a good quality summary in terms of cohesion and readability. These efforts led to several text summarizers which are available online to generate summaries from a variety of topics. Unfortunately, the existing summarizers generate output with poor readability, redundant information and disconnected sentences. To address these shortcomings, we have proposed a Graph-based automatic summarizer, GAUTOSUMM, and have compared results with existing online text summarizers. Initially, eleven online summarizers were selected out of which four (SMMRY, Tools4noobs, FreeSummarizer and SplitBrain) were used for evaluation due to several factors such as limited access, usage limitations for longer documents, and frequent error messages. Our proposed method is different from the existing work in many ways, for example, *tf's* are calculated only for words which are shared among other sentences. A pre-processing module was included in the algorithm to remove articles, prepositions, conjunctions, stop or bad words, brackets and text within brackets. A post-

processing module was added to generate aligned and formatted output, to maintain sentence structure and for easy readability. Intra-sentence edges were not considered by the algorithm to decrease the probability of redundant information selection. Finally, Jaccard distance was incorporated to determine the similarity between sentences.

Two datasets, Opinosis and DUC 2007, were used for evaluation of the summarizers. The documents in Opinosis ranges from 5 pages to 15 pages in length and contain different topics such as cars, hotels and food. The results showed that the metric values of Avg_Precision were very low as compared to Avg_Recall, because the gold standard summaries only consisted of 2 to 3 sentences. For longer documents, the value of precision drops even further. The decreased values of precision metric also reduced the values of harmonic mean AvgF_Score. On the other hand, DUC 2007[2], which is designed for extractive summaries, is more reliable and used as an industry standard for evaluation of text summaries. We used DUC 2007 dataset to evaluate GAUTOSUMM and online text summarizers for 45 samples. The comparison of generated summaries showed that for most of the topics (eight out of ten), GAUTOSUMM showed better metric values because of the inclusion of pre- and post-processing step together with control features which led to selection of sentences with less redundancy and precise information.

The results from ROUGE evaluation showed that GAUTOSUMM also generated good quality summaries for larger documents as compared to other summarizers. The generated summaries do not contain any stop words or redundant information due to a pre-processing module. Outliers were eliminated from the results for a more realistic analysis of summary

[2] supplied by NIST (National Institute of Standards and Technology)

quality. The execution time of GAUTOSUMM was significantly reduced by pre-processing due to elimination of redundant edges. The time performance of proposed summarizer is hardware dependent and can be further improved by adding resources such as cache, high-speed processor and RAM. GAOUTOSUMM has also been shown to be very scalable because the number of edges increase linearly with the number of sentences. The outputs from GAUTOSUMM are evaluated for general English topics rather than specialized documents. This shows that GAUTOSUMM works efficiently for documents which are from general topics such as those included in both datasets. For more specialized documents, semantics would be required to be incorporated.

5.1 Future Work

Based on the promising results, we believe that this research can be extended further in several directions. For instance:

- In our proposed method, a pre-processing module improved results significantly. A module embedded with semantics would likely produce even better results and high quality summaries. However, semantics introduces complex calculations in the algorithm and may require training that could affect the execution time significantly.
- A dictionary attached to the algorithm can also reduce the number of comparisons and false edges.
- An improved interface of GAUTOSUMM with more input options (such as number of words and keywords) would provide better impact to the reader.

- For the evaluation of GAUTOSUMM and online text summarizers, we have used a dataset from NIST. DUC 2007 dataset contains 45 samples from ten topics. A larger dataset can be used to evaluate more samples and number of topics covered.
- In GAUTOSUMM, we have included two control features which are related to length of the document. These control features seem to greatly impact the generated summary. More control features (such as sentence position and cue words) can be added to observe their impact on the generated summary.

Appendix 1

The following table shows a comparison of readability, redundancy and similarity for the approaches described in chapter 2.

	Approach	Readability	Less redundancy	Similarity
1.	Machine Learning	✓		
2.	Clustering Based		✓	
3.	Lexical Chaining	✓		
4.	Frequent Term			✓
5.	Information Retrieval			
6.	Graph-based	✓	✓	✓

As we can observe from the above table that Graph-based Approach results in better readability in terms of the content of generated summary by removing stop words from the original document. The generated summary contains less redundant information (after including control features, such as for sentence length) as compared to other approaches, and objectively calculates similarity between sentences (or the content of sentences).

Bibliography

[1] D. Dipanjan and A. F. Martins, "A Survey on Automatic Text Summarization," *Literature Survey for the Language and Statistics II course at CMU 4,* pp. 192-195, November 2007.

[2] H. P. Luhn, "The Automatic Creation of Literature Abstracts," *IBM Journal of research and development 2.2,* pp. 159-165, April 1958.

[3] H. P. Edmundson, "New Methods in Automatic Extracting," *Journal of the Association for Computing Machinery,* vol. 16, no. 2, pp. 264-285, April 1969.

[4] G. Erkan and D. R. Radev, "LexRank: Graph-based Lexical Centrality as Salience in Text Summarization," *Journal of Artificial Intelligence Research 22,* pp. 457-479, 2004.

[5] S. Suneetha, "Automatic Text Summarization: The Current State of the Art," *International Journal of Science and Advanced Technology,* vol. 1 , no. 9, pp. 1-12, November 2011.

[6] K. Jezek and J. Steinberger, "Automatic Text Summarization (The state of the art 2007 and new challenges)," in *Proceedings of Znalosti*, 2008.

[7] V. Gupta and G. S. Lehal, "A Survey of Text Summarization Extractive Techniques," *Journal of Emerging Technologies in Web Intelligence,* vol. 2, no. 3, pp. 258-268, August 2010.

[8] S. Hariharan and T. Ramkum, "Enhanced Graph Based Approach for Multi-Document Summarization," *The International Arab Journal of Information Technology,* vol. 10, no. 4, pp. 334-341, July 2013.

[9] Daniel Jacob Gillick, *The Elements of Automatic Summarization,* Berkeley: University of California, , 2011.

[10] E. Hovy and . C.-Y. Lin, "Automated Text Summarization in SUMMARIST," *Advances in Automatic Text Summarization,* 2003.

[11] Elena Lloret, *Text Summarization: An Overview,* Spain: Dept. Lenguajes y Sistemas Inform´aticos Universidad de Alicante, 2006.

[12] J. Steinberger and K. Jezek, "Evaluation Measures for Text Summarization," *Computing and Informatics,* vol. 28, pp. 1001-1026, 2009.

[13] S. Hariharan and R. Srin, "Studies on Graph based Approaches for Single and Multi-Document Summarizations," *International Journal of Computer Theory and Engineering,,* vol. 1, no. 5, pp. 519-526, Dec 2009.

[14] N. Technology, "The Power of Graph-Based Search," neo4j.com, 2015.

[15] "Jaccard Distance," [Online]. Available: https://en.wikipedia.org/wiki/Jaccard_index. [Accessed 04 April 2017].

[16] H. P. Luhn, "The Automatic Creation of Literature Abstracts," *IBM Journal of research and development 2.2,* pp. 159-165, April 1958.

[17] P. B. Baxendale, "Machine-Made Index for Technical Literature—An Experiment," *IBM Journal of Research and Development,* vol. 2, no. 4, pp. 354-361, 1958.

[18] H. P. Edmundson, "New Methods in Automatic Extracting," *Journal of the Association for Computing Machinery,* vol. 16, no. 2, pp. 264-285, April 1969.

[19] D. Dipanjan and A. F. Martins, "A Survey on Automatic Text Summarization," *Literature Survey for the Language and Statistics II course at CMU 4 (2007),* pp. 192-195, November 2007.

[20] J. Kupiec, J. Pedersen and F. Chen, "A Trainable Document Summarizer," *Xerox Palo Alto Research Center,* pp. 68-73, 1995.

[21] C. Y. Lin and E. Hovy, "Identifying Topics by Position," in *Information Sciences Institute of the University of Southern California*, California, 1997.

[22] J. M. Conroy and D. P. O'leary, "Text summarization via hidden markov models," in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, New York, 2001.

[23] M. Osborne, "Using Maximum Entropy for Sentence Extraction," in *Proceedings of the ACL-02 Workshop on Automatic Summarization*, United Kingdom, 2002.

[24] E. Hovy and . C.-Y. Lin, "Automated Text Summarization in SUMMARIST," *Advances in Automatic Text Summarization,* pp. 18-24, 1998.

[25] K. McKeown, R. Barzilay and S. B. Goldensohn, *The Columbia Multi-Document Summarizer for DUC 2002,* New York: Workshop on Automatic Summarization, 2002, pp. 1-8.

[26] K. McKeown, R. Barzilay, A. Nenkova, B. Schiffman, S. Sigelman, A. Schlaiker, S. B. Goldensohn and V. Hatzivassiloglou, "Columbia at the document understanding conference 2003.," in *Proceedings of the Document Understanding Workshop (DUC 2003)*, New York, 2003.

[27] T. Hirao, Y. Sasaki , H. Isozaki and E. Maeda, "NTT's Text Summarization System for DUC-2002," in *Proceedings of the Document Understanding Conference*, 2002.

[28] M. Karamuftuoglu, "An approach to summarizaion based on lexical bonds.," in *Proceedings of the Workshop on Automatic Summarization, 40th Annual Meeting of the Association for Computational Linguistics.*, Cambridge, 2002.

[29] P. Lal and S. Ruger, "Extract-based Summarization with Simplification," in *Proceedings of the ACL.*, London, 2002.

[30] C. Aone, M. E. Okurowski, J. Gorlinsky and B. Larsen, *A trainable summarizer with knowledge acquired from robust nlp techniques,* Fairfax: Advances in automatic text summarization 71, 1999.

[31] J. M. Conroy and J. D. Schlesinger, "CLASSY Query-Based Multi-Document Summarization," in *Proceedings of the 2005 Document Understanding Workshop*, Boston, 2005.

[32] V. Hatzivassiloglou, J. L. Klavans, M. L. Holcombe, R. Barzilay, . M.-Y. Kan and K. R. McKeown, "SIMFINDER: A Flexible Clustering Tool for Summarization," in *Proceedings of the NAACL workshop on automatic summarization*, New York, 2001.

[33] K. M. Svore, L. Vanderwende and C. J. Burges, "Enhancing Single-document Summarization by Combining RankNet and Third-party Sources," *Emnlp-conll,* pp. 448-457, 2007.

[34] N. Ani, "Automatic text summarization of newswire: Lessons learned from the document understanding conference.," *AAAI,* vol. 5, pp. 1436-1441, July 2005.

[35] M. M. Haque, S. Pervin and Z. Begu, "Literature Review of Automatic Multiple Documents Text Summarization," *International Journal of Innovative and Applied Studies,* vol. 3, no. 1, pp. 121-129, May 2013.

[36] "tools4noobs," [Online]. Available: http://www.tools4noobs.com/summarize/. [Accessed Jun 2017].

[37] J. Goldstein, V. Mittal , J. Carbonell and M. Kantrowitz, "Multi-Document Summarization By Sentence Extraction," in *Proceedings of the 2000 NAACL-ANLPWorkshop on Automatic summarization*, New Jersey, 2000.

[38] H. H. Chen and C. J. Lin, "A Multilingual News Summarizer," in *Proceedings of the 18th conference on Computational linguistics*, Taiwan, 2000.

[39] M. Y. Kan, K. McKeown and J. L. Klavans, "Applying Natural Language Generation to Indicative Summarization," in *Proceedings of the 8th European workshop on Natural Language Generation*, New York, 2001.

[40] J. D. Schlesinger, D. P. O'Leary and J. M. Conroy, "Arabic/English Multi-document Summarization with CLASSY—The Past and the Future," *Computational Linguistics and Intelligent Text Processing,* pp. 568-581, 2008.

[41] Xiaojun Wan, "An Exploration of Document Impact on Graph-Based Multi-Document Summarization," *Association for Computational Linguistics,* pp. 755-762, October 2008.

[42] N. Agarwal, K. Gvr, R. S. Reddy and C. P. Rose, "Towards Multi-Document Summarization of Scientific Articles:Making Interesting Comparisons with SciSumm," in *Proceedings of the Workshop on Automatic Summarization for Different Genres, Media, and Languages*, Portland, Oregon, 2011.

[43] C.-Y. Lin and E. Hovy, "Automated Multi-document Summarization in NeATS," in *Proceedings of the second international conference on Human Language Technology Research*, San Diego, California, 2001.

[44] K. R. McKeown , R. Barzilay, D. Evans, V. Hatzivassiloglou, J. L. Klavans, A. Nenkova , C. Sable, B. Schiffman and S. Sigelman, "Tracking and Summarizing News on a Daily Basis with Columbia's Newsblaster," in *Proceedings of the second international conference on Human Language Technology Research. Morgan Kaufmann Publishers Inc.*, New York, 2002.

[45] R. Angheluta, R. Mitra, X. Jing and M.-F. Moen, *K.U.Leuven summarization system at DUC 2004,* Belgium: In Document Understanding Conference., 2003.

[46] Claudia Sofia Oliveira Santos, "ALEXIA – Acquisition of Lexical Chains for Text Summarization," University of Beira Interior, Covilhã, Portugal, February 2006, 109.

[47] R. Barzilay and M. Elhadad, *Using Lexical Chains for Text Summarization,* Beer-Sheva, 84105 Israel: Mathematics and Computer Science Dept., 1999, pp. 111-121.

[48] H. Jing, *Cut-and-Paste Text Summarization,* Doctoral dissertation, Columbia University, 2001.

[49] M. Brunn, Y. Chali and B. Dufour, "The University of Lethbridge Text Summarizer at DUC 2002," in *Document Understanding Conferences*, Lethbridge, 2003.

[50] Maheedhar Kolla, "Automatic Text Summarization Using Lexical Chains: Algorithms and Experiments," University of Lethbridge, Faculty of Arts and Science, Lethbridge, 2004.

[51] N. K. Nagwani and S. Verma, "A frequent Term and Semantic Similarity based Single Document Text Summarization Algorithm," *International Journal of Computer Applications,* vol. 17, no. 2, pp. 36-40, March 2011.

[52] Y. Ledeneva, "Automatic Language-Independent Detection of Multiword Descriptions for Text Summarization," Instituto politecnico nacional centro de investigacion en computacion laboratorio de lenguaje natural y procesamiento de texto, MEXICO, 2008.

[53] L. Yulia, A. Gelbukh and R. G. Hernandez, "Effect of Preprocessing on Extractive Summarization with Maximal Frequent Sequences," *MICAI 2008: Advances in Artificial Intelligence,* pp. 123-132, 2008.

[54] I. Mani and M. Marbury, Advances in Automatic Text Summarization, Cambridge: MIT press Vol 293, 1999.

[55] K. Mitze and L. F. Rau, "Automatic condensation of electronic publications by sentence selection," *Information Processing Management,* vol. 31, pp. 675-685, 1995.

[56] F. Gotti, G. Lapalme, L. Nerima and E. Wehrli, "GOFAISUM: A Symbolic Summarizer for DUC," in *Proceeding of Document Uunderstanding Conference*, Montreal, 2007.

[57] M.-Y. Kan and K. R. McKeown, "Information Extraction and Summarization: Domain Independence through Focus Types," Computer Science Technical Report, Columbia University, 1999.

[58] P. S. Negi, . M. M. S. Rauthan and H. S. Dham, "Text Summarization for Information Retrieval using Pattern Recognition Techniques," *International Journal of Computer Applications,* vol. 21, no. 10, pp. 20-24, May 2011.

[59] E. Alfonseca and P. Rodriguez, *Description of the UAM system for generating very short summaries at DUC-2003,* Edmonton: In HLT/NAACL Workshop on Text Summarization / DUC 2003, 2003.

[60] Xiaojun Wan, "An Exploration of Document Impact on Graph-Based Multi-Document Summarization," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Honolulu, 2008.

[61] R. Mihalcea and P. Tarau, *TextRank: Bringing Order into Texts,* vol. 4, North Texas: Department of Computer Science, University of North Texas, 2004, pp. 404-411.

[62] D. R. Radev, S. B. Goldensohn and Z. Zhang, *Experiments in Single and Multi-Document Summarization Using MEAD,* Michigan: Ann Arbor 1001 (2001): 48109., 2001.

[63] L. Vanderwende, M. Banko and . A. Menezes , "Event-Centric Summary Generation," in *Working notes of DUC (2004)*, Redmond, 2004.

[64] M. Fuentes, H. Rodriguez and D. Ferres, "FEMsum at DUC 2007," in *Proceeding of the Document Understanding Conference 2007*, Spain, 2007.

[65] P. Articles, "302 words short essay on Food," Preserve Articles, [Online]. Available: http://www.preservearticles.com/201105156684/short-essay-on-food.html. [Accessed October 2017].

[66] S. Hariharan and R. Srin, "Studies on Graph based Approaches for Single and Multi-Document Summarizations," *International Journal of Computer Theory and Engineering,,* vol. 1, no. 5, pp. 519-526, Dec 2009.

[67] "Sentence Length," [Online]. Available: https://strainindex.wordpress.com/2008/07/28/the-average-sentence-length/.

[68] Griffith Publishing, [Online]. Available: https://jgwritingtips.wordpress.com/2010/02/17/how-long-should-a-sentence-be/. [Accessed October 2017].

[69] "Rouge 2.0," [Online]. Available: http://kavita-ganesan.com/content/rouge-2.0-documentation. [Accessed Jun 2017].

[70] "DUC/NIST," [Online]. Available: http://duc.nist.gov/data.html. [Accessed Jul 2017].

[71] C. Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," in *Proceedings of Workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL 2004.*, Barcelona, 2004.

[72] "NIST," [Online]. Available: http://duc.nist.gov/duc2007/tasks.html. [Accessed May 2017].

[73] "SMMRY," [Online]. Available: http://smmry.com/. [Accessed Jul 2017].

[74] "tools4noobs," [Online]. Available: http://www.tools4noobs.com/summarize/. [Accessed July 2017].

[75] "FreeSummarizer," [Online]. Available: http://freesummarizer.com/. [Accessed Jun 2017].

[76] "AutoSummarizer," [Online]. Available: http://autosummarizer.com/index.php. [Accessed Jun 2017].

[77] "SplitBrain," [Online]. Available: http://www.splitbrain.org/services/ots. [Accessed Jun 2017].

[78] "Text Compactor," [Online]. Available: http://textcompactor.com/. [Accessed Aug 2016].

[79] "Shvoong," [Online]. Available: http://www.shvoong.com/summarizer/. [Accessed Jun 2017].

[80] "HelpfulPapers," [Online]. Available: http://helpfulpapers.com/summarizer-tool/. [Accessed Jun 2017].

[81] "Article Summarizer Online," [Online]. Available: http://www.summarizing.biz/best-summarizing-strategies/articlesummarizer-online/. [Accessed Jun 2017].

[82] "MS Word Summarizer," [Online]. Available: http://office.microsoft.com/en-ca/word-help/automatically-summarize-adocument. [Accessed Aug 2016].

[83] Q. Fatima, S. alZahir and M. Cenek, "New Graph-Based Text Summarization Method," in *Communications, Computers and Signal Processing (PACRIM), 2015 IEEE Pacific Rim Conference*, Victoria, August 2015.

[84] TABLEAU , [Online]. Available: https://www.tableau.com/. [Accessed May 2017].

[85] "Mathworks - Matlab Answers," [Online]. Available: https://www.mathworks.com/matlabcentral/answers/92044-is-it-possible-to-choose-computer-hardware-which-best-optimizes-the-performance-of-matlab. [Accessed June 2017].