

Palestine Polytechnic University



College of Administrative Sciences & Informatics

Information Technology Department

Text Summarization: Using Combinational Statistical and Linguistic Methods

Project Team

Bara' Jamal Amro

Ghadeer Abad Ghaffar Natsha

Yasmin Hussein Ta'amra

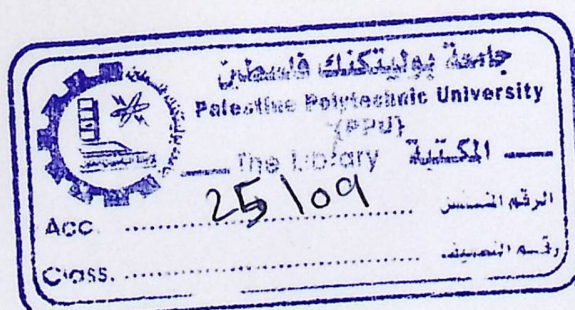
Project Supervisor

Eng. Manal Tamimi

This project is presented to the Department of Information Technology at College of Administrative Sciences & Informatics, for partial fulfillment Bachelor of Information Technology degree requirements.

June, 2011

Manal Tamimi



Abstract

Summaries are considered as useful indicators of the document content. Traditionally, summaries are created by humans, through reading a text and identifying the most important points in it. But the information overloading makes such manual work very difficult. Many attempts have been made to automate the summarization task.

This project aims at proposing a methodology for automatic text summarization based on extractive methods. Extractive summarization needs the selection of a subset of important, meaningful sentences from the source text. The work presents a method for automatic text summarization of a single computer science article using statistical and linguistic approach. About 50 experts' persons summaries were used to evaluate the proposed method. Our results show that using the combinational statistical and linguistic methods is better than using each method alone.

الملخص

هذا المشروع يصف طريقة لتلخيص النص من خلال تقليد طريقة الإنسان في تحليل النصوص وتلخيصها، وذلك من خلال دمج الطريقة الإحصائية واللغوية في عملية استخراج الخصائص من النص ، لتحديد الجمل المهمة ومن ثم الحصول على ملخص يحوي الفكرة الأساسية ، وخالي من التكرار مع المحافظة على ترابط الجمل.

لقد تم تطبيق هذه الطريقة على مقالات باللغة الانجليزية متعلقة بعلم الحاسوب ، و بعد مقارنة الملخصات الناتجة بملخصات قام بتلخيصها خمسون أستاذاً مختصاً باللغة الانجليزية، تبين أن استخدام عملية الدمج بين الطريقتين الإحصائية واللغوية حققت نتائج أفضل من استخدام كل طريقة على حدى .

Dedication

Acknowledgement

To our families for their support

To our teachers for their advices

To our brothers and sisters

To our friends

We thank our families for their support, encouragement and patience from the first step till the end.

Supervisor Eng. Manal Taha for her support and efforts.

Special thanks to Dr. Ismail Rami the Head of Information Technology Department.

Special thanks to Mrs. Suzan Sultan for her support.

Also we thank the English teachers who helped us.

Acknowledgement

We thank our families for continued support, encouragement and patience from the first step till the end.

Supervisor Eng. Manal Tamimi for her support and efforts.

Special thanks to Dr. Ismail Romi the Head of Information Technology Department.

Special thanks to Mrs. Suzan Sultan for her support.

Also we thank the English teachers who helped us.

Table of Contents

Abstract.....	II
Arabic abstrac.....	III
Dedication.....	IV
Acknowledgement.....	V
Table of contents.....	IX
List of figures.....	X
List of tables.....	X

Chapter One: Introduction

1.1 Overview.....	2
1.2 Problem statement.....	3
1.3 Objectives.....	3
1.4 project block diagram.....	3
1.5 Benefits.....	4
1.6 Expected result.....	4

Chapter Two: Background

2.1 Overview.....	6
2.2 Text mining.....	6
2.3 Text summarization.....	7
2.4 Automatic text summarization.....	7
2.5 Applications of automatic text summarization.....	7
2.6 Types of summaries.....	8
2.6.1 METHODS.....	8
2.6.2 CONTENT.....	9
2.6.3 DETAILS.....	10
2.6.4 LIMITATION.....	10
2.6.5 NUMBER OF INPUT... ..	11
2.6.6 LANGUAGE.....	11
2.7 The process of automatic text summarization.....	12
2.7.1 TOPIC IDENTIFICATION	12
2.7.2 INTERPRETATION.....	12

2.7.3 GENERATION.....	12
2.8 Text features.....	13
2.9 Evaluation of text summarization.....	14
2.10 Related terms.....	15
2.10.1 REMOVING STOPWORDS	15
2.10.2 PART OF SPEECH.....	15
2.10.3 SEGMENTATION.....	15
2.10.4 WORDNET.....	16
2.11 Summary.....	16

Chapter Three: Literature Review

3.1 Overview.....	18
3.2 E xtraction approaches in text summarization.....	18
3.2.1 STATISTICAL METHODS.....	18
3.2.2 LINGUISTIC METHODS.....	20
3.2.3 COMBINATIONAL STATISTICAL METHODS AND LINGUISTIC METHODS	21
3.2.4 RHETORICAL STRUCTURE THEORY (RST)	22
3.3 Comparison between the four approaches.....	23
3.4 Importance of literature review in this project.....	24
3.5 Summary.....	25

Chapter Four: Methodology

4.1 Overview.....	27
4.2 Preparation phases for the implementation of the project.....	27
4.3 Defining the requirements the project.....	27
4.4 Collecting samples of documents.....	27
4.5 Preprocessing documents.....	27
4.5.1 REMOVING STOPWORDS.....	28
4.5.2 REFERRING ABBREVIATIONS.....	29
4.5.3 WORD TAGGING.....	30
4.5.4 SEGMENTATION.....	30
4.6 Feature extraction.....	32
4.6.1 THE WORD LEVEL FEATURES	32
4.6.2 THE SENTENCE LEVEL FEATURES.....	36
4.7 Summary generation.....	42
4.8 Post processing.....	43

4.8.1 ORDERING THE SENTENCES	43
4.8.2 CONNECTING WORDS.....	44
4.8.3 SENTENCE TO SENTENCE SIMILARITY	45
4.9 Summary.....	47

Chapter Five: Programming

5.1 Overview.....	49
5.2 SOFTWARE ENVIRONMENT... ..	49
5.2.1 <i>netbeans ide 6.9.1</i>	49
5.2.2 RUBY LANGUAGE.....	49
5.2.3 RUBYGEMS.....	50
5.2.4 DEVKIT.....	51
5.2.5 MYSQL SERVER 5.1.....	51
5.3 Database schema.....	51
5.4 Summary.....	53

Chapter Six: Experiments and Results

6.1 Overview.....	55
6.2 Testing sample.....	55
6.3 Experiments and results.....	55
6.3.1 PREPROCESSING EXPERIMENTS AND RESULTS	56
6.3.2 FEATURES EXTRACTION EXPERIMENTS AND RESULTS.....	58
6.3.3 SUMMARY GENERATION EXPERIMENTS AND RESULTS	72
6.3.4 POST-PROCESSING EXPERIMENTS AND RESULTS.. ..	73
6.4 Evaluation.....	76
6.5 Summary.....	76

Chapter Seven: Conclusions and Future Work

7.1 Overview.....	78
7.2 Conclusions.....	78
7.3 Contributions.....	79
7.4 Project problems.....	80
7.5 Future works.....	80

Appendixes

Appendixes(A).....	81
Implementation code.....	83

Appendixes(B).....	94
1 LIST OF STOPWORDS.....	95
2 TAGS OF WORD TAGGING.....	99
3 LIST OF CUE PHRASES.....	100
4 LIST OF CONNECTING WORDS.....	101
Appendixes(C)	102.
Example of generated summaries	103
ORIGINAL TEXT A.....	103
PROPOSED SYSTEM SUMMARY FOR A.....	104
ORIGINAL TEXT B.....	105
PROPOSED SYSTEM SUMMARY FOR B.....	106
Appendixes(D).....	107
paper.....	108
References	
references	114

List of Figures

Figure (1 - 1): block diagram of extractive method.....	4
Figure (2 - 1): text mining processes.....	6
Figure (2 - 2): types of summaries.....	8
Figure(3 - 1): text rhetorical structure.....	22
Figure(3 - 2): text 1 rhetorical structure.....	22
Figure (4 - 1): preprocessing stage.....	28
Figure (4 - 2): removing stopwords step.....	29
Figure (4 - 3): referring abbreviations step.....	30
Figure (4 - 4): word tagging step.....	30
Figure (4 - 5): segmentation step.....	32
Figure (4 - 6): word features.....	32
Figure (4 - 7): step for calculation of tf-isf.....	34
Figure(4 - 8): extraction existence in the title of document.....	35
Figure (4 - 9): extraction words lengths.....	36
Figure (4 - 10): sentences features.....	36

Figure (4 - 11): extraction sentence location.	37
Figure (4 - 12): extraction sentence length.....	38
Figure(4 - 13): extraction existence of cue phrase.....	39
Figure (4-14): extraction existence of keywords feature.....	41
Figure (4-15): extraction biased feature.....	42
Figure(4 - 16): sentence selection.....	43
Figure(4 - 17): post processing stage.....	43
Figure(4 - 18): ordering the sentences.....	44
Figure(4 - 19): connecting words steps.....	45
Figure(4 - 20): sentence to sentence similarity steps.....	46
Figure(6 - 1): an example of removing stopwords from text.....	56
Figure(6 - 2): an example of referring abbreviations.....	57
Figure (6 - 3): an example of word tagging.....	58
Figure (6 - 4): an example of text segmentation.....	58
Figure (6 - 5): f-measure values at different values of tf-isf.....	60
Figure (6-7): an example of existence in title.....	63
figure(6 - 8): f-measure values at different values of α , β , and μ	65
Figure(6 - 9): f-measure values at different values of word weight.....	65
Figure (6 - 10): f-measure values at different values of sentence location.....	66
Figure(6 - 11): f-measure values at different numbers of keywords.....	67
Figure(6 - 12): example for cue phrase.....	68
Figure(6 - 13): f-measure values at different numbers of keywordS.....	69
Figure(6-14) f-measure values at different number of biased words in the sentence.....	70
Figure(6 - 15): f-measure values at different sentence weights.....	74
Figure(6 - 16): an example of connecting words step.	75
Figure(6 - 17): an example of removing urls and e-mails.....	76

5 List of Tables

Table (2 - 1): differences between extractive and abstractive.	9
Table (3 - 1): comparison between the four approaches.....	24
Table (5 - 1): paragraph table.	52

Table (5 - 2): sentences table.	52
Table(5 - 3): words table.....	52
Table(5 - 4): stopword table.....	53
Table(5 - 5):cue phrase table	53
Table(5 - 6): connecting words table.	53
Table (6-1): the result of determining tf-isf threshold experiments.....	59
Table (6-2): the result of determining word length threshold experiments.	60
Table (6 - 3): the experiments results of determining word weight factors.....	63
Table (6 - 4): the result of determining word weight threshold experiments	64
Table (6 - 5): the result of determining sentence location threshold experiments.....	65
Table (6 - 6): the result of determining sentence length threshold experiments.....	66
Table (6 - 7): the result of determining keyword existence threshold experiments.....	68
Table (6 - 8): the result of determining biased words existence threshold experiments.	69
Table (6 - 9): the experiments results of determining sentence weight factors	72
Table (6 - 10): the result of determining sentence weight threshold experiments.....	73
Table (7 - 1): comparisons of precision, recall, f-measure using different methods	78

Introduction

Contents:

1.1 Overview

1.2 Problem Statement

1.3 Objective

1.4 Project block diagram

1.5 Benefits

1.6 Expected results

1.2 Problem Statement

Students and researchers constantly face this scenario: the number of used documents becomes very huge, which makes it so difficult for researchers or readers to read all of these documents, also, the time spent in reading these documents seems to be endless.

The necessity of summarizing a document into a summarized text document becomes very important; because of the problem of huge number of documents. Automatic summarization involves reducing the original text to a short set of words that gives the text main meaning.

Recently, there are many automatic text summarization systems which aim to perform the task like a human. These systems' results suffer from the lack of readability, coherence and redundancy elimination.

1.3 Objectives

This project aims to achieve the following two objectives:

- The main objective is to develop a methodology for summarizing the text, which fulfills the following factors:
 1. Readability factor, by protecting the main idea of the text.
 2. Making connection between the sentences in the summary.
 3. Redundancy elimination, by founding the similarity between the sentences.
- Our project tries to prove that the combination between the strengths of statistical methods and linguistic methods will succeed in producing the best summaries.

1.4 Project block diagram

The main steps of this project are illustrated in figure (1-1).

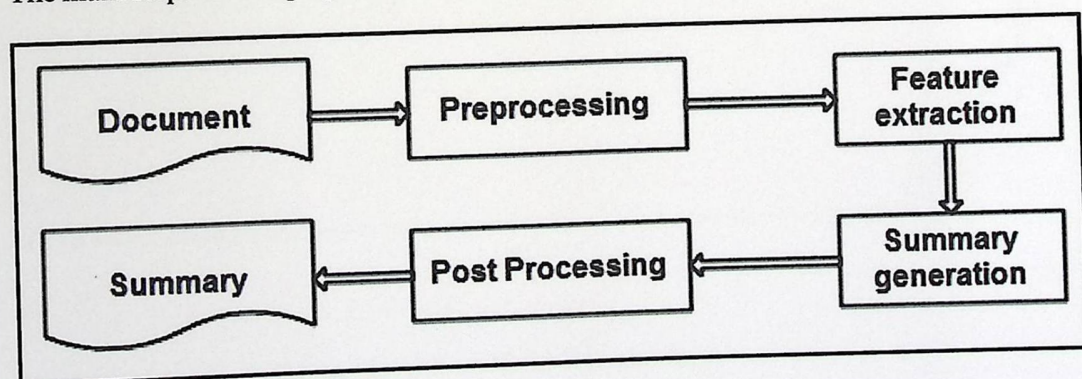


Figure (1-1): Block diagram of extractive method.

2.6 Types of summaries

Text summarization systems can be classified based on different criteria. Different papers have discussed some of these criteria. These categories are presented in figure (2 - 2), are as the following.

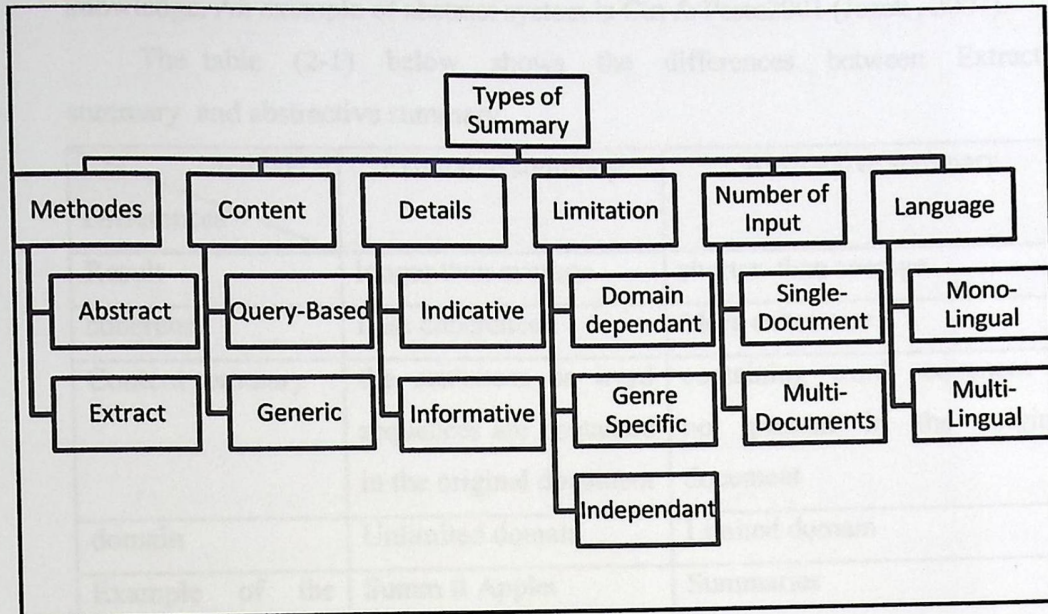


Figure (2 - 2): Types of summaries.

2.6.1 Methods

Text summarization systems are divided into extractive and abstractive methods that both are aimed at analyzing the texts and generalizing summaries from them.

- Extractive summarization method

This method works on selecting important paragraphs, sentences and key phrases from the original document and binding them into shorter forms. The extractive method has two approaches statistical and linguistic, that decides which is the important sentence based on the features of the sentences. Most of the recent automated text summarization systems are based on the extraction method; because it is conceptually simple and easy to be implemented. An example of the extractive system is NeATS11 2001 (Jezek 1,2007).

presents it as a summary. An example of a query based summarizer system is Miter's WebSumm (Aliguliyev,2009).

2.6.3 Details

This classification is based on the details and the amount of it to be included in the summary, which is important in the desired summary. The following are some of the categories used in this classification:

- Indicative

Indicative summary is used to determine what the fundamental issues that were mentioned in the original text. They can also give a brief and simple summary for each topic. The typical length of this type of summarization is 5% to 10% of the main text. The advantage that comes out of this type of summaries is giving the reader a general idea about the text. This will make the reader decide whether he would continue reading the entire document or not. An example of indicative summarizer system is ADAM (Wong,2009).

- Informative

This category attempts to include all the contained concepts in the original text to the possible extent and within the limits of compression rate. The length of informative summary is 20% to 30% of the main text. An example of informative summarizer system is DimSum 1997 (vishal,2010).

2.6.4 Limitation

Another classification is based on limitation of input text. Here are the categories of this classification:

- Domain dependant

It only summarizes the texts which their subject can be defined in the domain of the system an example of domain dependent system is LAKE 2004 (Aliguliyev,2009).

- Genre specific

The criteria of this type of classification that the system has limitation on the template of the text and accept only special type of texts as their input. There are different templates like newspaper articles, scientific papers , storie,

manuals and etc. system using the structure of these templates for generating the summary. An example of genre summarizer system is GISTexter 2002 (Aliguliyev, 2009).

- Domain independent

The independent systems do not have any predefined limitation and can accept different types of texts. An example of domain dependent system is FociSum 1999(Aliguliyev,2009).

2.6.5 Number of input

This classification is based on the number of documents which are input to the system.

- Single document

The system only accepts one (single) document or text. An example of a single document system is Luhn 1958 (Alonso, 2005).

- Multi document

A system accepts a group (cluster) of related documents, producing summary from a set of documents about an explicit or implicit main topic. An example of multi document system is SUMMONS₆ 1998 (Yang& Wan 2008) . Factors that make multi document summarization more difficult include (Radev, 2002):

- Multiple articles can be written by different authors, having different document structure and writing styles.
- Multiple articles might have conflict in views of the same event. So we need a perfect summarizer to be able to detect and handle this situation.
- Documents that have Facts and views can change over time because they written at different times.

2.6.6 Language

This kind of classification is based on the language of the input and the output (generated summary) text the categorize summarization systems consist of two main groups.

- Mono lingual

These systems only accept documents with certain languages, like

- English, and the generated summary done on that language too. An example of mono lingual system is Edmundson1969(Yang& Wan 2008).

- Multi lingual

These systems can accept documents in multiple languages, and there is a freedom for the user to select the language that is desired to be the language of the generated summary. An example of multi lingual system is Chen & Lin 2000(Alonso,2005).

2.7 The process of automatic text summarization

The process of automatic text summarization can be described by the equation:
Automatic text summarization = (topic identification + interpretation+ generation) we can explain the process in points as the following (Dalianis, 2005):

2.7.1 Topic identification

It is the first step that includes the text identification, and its essence. There are various techniques to be done in the preprocessing phase. A document has specific parts that hold an important topic, the title is always important, the first sentence and the last in each paragraph. Moreover some words or phrases are important to determine where the essence of the text is, for example, in the summary and in the conclusion.

2.7.2 Interpretation

We mentioned that text summarization has two methods; extractive and abstractive. The identification step is based on the extract method. The interpretation phase is performed by the abstract method that includes merging similar topics into one and removing redundancies etc.

2.7.3 Generation

The final phase of an automatic text summarization is the generation of final output. It consists of four methods:

- Extraction: when the process of summarizing is finished, then we print on the output of the resulting sentences and phrases.

- Topic lists: we define lists of the most frequent keywords or interpreted concepts resulting from the summary and printed on the output.
- Phrase concatenation: if there are two or more similar phrases in the generated summary, we merge them together.
- Sentence generation: new sentences from a list of concepts and their related topics are the result of sentence generator.

2.8 Text features

In the text summarization the extraction of the features from the sentences play an important role. The features which are extracted are as the following (Fattah, 2008).

- Sentence Location: Usually the first and the last sentence of a paragraph have great chances to be included in summary.
- Cue-Phrase feature: sentences that include cue phrases such as: "In conclusion", "this letter", "this report", "summary" etc, have greater chances to be included in the summary.
- Occurrence of non essential information: Some words such as: "because", "furthermore", and "additionally" are indicators of non-essential information. Sentences containing these words are the most excluded from the summary.
- Similarity between sentences: this feature determines the similarity between the sentence and other sentences depending on the vocabulary overlap between them in the document. One of the similar sentences must be included in the summary.
- Similarity with the title: this feature determines the similarity between sentences and the title depending on the vocabulary overlap between the sentence and the document's title. This feature increases the weight for the sentence to be occurring in the generated summary.
- Inclusion of proper noun: a proper noun is a name of a person, a place and a concept etc. If the sentence contains more proper nouns, it becomes the most probably included in the document summary.
- Inclusion of numerical data: if the sentences contain numerical data, they have great chances to be included in the summary.
- Sentence relative length: it is used to determine the length of the sentence. If

the sentence is too short, it will be not expected to belong to the final summary.

2.9 Evaluation of text summarization

Evaluation is an important part of any research and development effort. Each approach should be evaluated. Methods and metrics of evaluating summaries are used to determine the strengths and weaknesses of the different systems of text summarization.

There are indirect measures that determine the usefulness and completeness of a summary (Madhavi,2002) such as:

1. When the user read the summary, can he answer all the questions, as he would by reading the source document?
2. What is the compression ratio between the summary and its source document?
3. Redundancy: is there any repeated information in the summary?
4. Quality measures that are usually difficult to measure are:
5. Intelligibility
6. Coherence
7. Readability (depends on cohesion/coherence/intelligibility)

Evaluation metrics are intrinsic and extrinsic. Intrinsic determines the quality of the summary. For example, evaluation of summary coherence, this metric is difficult to measure because the qualities of the summaries are judged by direct human judgment and human judgment varies from person to another. Extrinsic measures a system's performance in a particular task. For example, redundancy (Madhavi,2002).

In extractive summarization, human summaries are used to evaluate automatic summarization. Human extract sentences from the documents to create manual-extractive summaries. Then automatic summarization evaluated by computing the number of sentences common between automatic and human summaries. These measures are called Precision and Recall. This method is currently the most used method to evaluate extractive summarization (Madhavi,2002).

Precision measure is used to evaluate the percentage of correctness for the sentences in the summary. On the other hand, Recall measure is used to evaluate the percentage of relevant sentences included in summary. F-measure combines both measures (Madhavi,2002):

3.1 Overview

This chapter contains a summary of some important contributions related to the text summarization. The chapter also includes extraction approaches to text summarization and comparison between them. In the first section of this chapter, the importance of these works is presented in this project.

3.2 Extraction approaches in text summarization

Previous studies have discussed a lot of text summarization approaches. Most of

Chapter Three

Literature Review

Contents:

3.1 Overview

3.2 Extraction approaches in text summarization

3.3 Comparison between the four approaches

3.4 Importance of literature review in this project

3.5 Summary

3.1 Overview

This chapter contains a summary of some important contributions related to the text summarization. The chapter also includes extraction approaches to text summarization and comparison between them. In the final section of this chapter, the importance of these works is presented in this project.

3.2 Extraction approaches in text summarization

Previous studies have discussed a lot of text summarization approaches. Most of them are based on sentence extraction. The extraction is defined as selecting key sentences or words from the original text and using them to form the summary. There are four different approaches for scoring and selecting key sentences as the following (Amini, et al., 2009).

3.2.1 Statistical methods

Statistical methods are the first technique of text summarization, which extracts sentences that occurred in source text, without taking into consideration the meaning of the words. There are several statistical methods for selecting important sentences (Seo, et al., 2008).

- Title method

Title method defines as how many words are commonly used between a sentence and a title. The equations below show the similarity calculation between a sentence and a title query that can be calculate the score of sentences.

$$sim(S_i, Q) = \sum_{k=1}^n w_{ik} w_{qk} \quad (3 - 1)$$

$$Score(S_i) = sim(S_i, Q) \quad (3 - 2)$$

Where S_i is an i th sentence and Q is a title query. w_{ik} is the binary weight of k th word in i th sentence and w_{qk} is the binary weight of k th word in the title query.

- Location method

This method is determined through the type of the document. For example, in technical documents, sentences in the conclusion section are ranked high, while in news articles, first few sentences are ranked higher.

- Aggregation similarity method

Aggregation similarity defines as the sum of similarities with other all sentence vectors in document vector based model. The score of sentences is calculated as the following equation.

$$\text{sim}(S_i, S_j) = \sum_{k=1}^n w_{ik} w_{jk} \quad (3-3)$$

$$\text{Score}(S_i) = \text{asim}(S_i) = \sum_{j=1, j \neq i}^m \text{sim}(S_i, S_j) \quad (3-4)$$

Where w_{ik} is the binary weight of k th word in i th sentence.

- Frequency method

This method based on how many times the term occurred in the document. The score of a sentence can be calculated as the sum of scores of the words in the sentence.

$$w_i = \text{tf}_i \times \log \frac{ND}{df_i} \quad (3-5)$$

Where tf_i is the term frequency of i th word in the document, ND is the total number of documents, and df_i is the document frequency of i th word in the whole data set.

- Term Frequency (TF)-based method

The document that is divided into topics which each of them has a title is very useful to generate a summary. However, in special cases, it can be hard to extract a title from documents that do not have a title. The TF-based query method extracts high frequent words in a document, and the extracted words are used as a query instead of title. This method is used if the document does not have a title; the TF-based query method is used instead of the Title method. Like the Title method, the equation below calculated the similarity between a sentence and the TF-based query.

$$\text{sim}(S_i, \text{TFQ}) = \sum_{k=1}^n w_{ik} w_{\text{TFQ}} \quad (3-6)$$

Where n is the number of words which is included in a document. w_{ik} is the binary weight of k th word in i th sentence and w_{TFQ} is the binary weight of k th word in the TF-based query.

- Latent Semantic Analysis

Latent Semantic Analysis is a technique that reduces the vector space in order to map “related” terms (or co-occurring terms in a collection) into the same domain. LSA method was used for multi-document summarization. (Radev, et al., 2001).

3.2.2 Linguistic methods

This method needs to be aware and know deeply in linguistic knowledge, so that computer will be able to analyze the sentences and then decide which sentence is to select depending on the position of subject, verb and name (Dongrui, et al., 2010). Different methods in this approach are discussed here:

- Lexical chain

A lexical chain is a sequence of words, each word from the sequence can be combined to the sequence that has clearly the same meaning. All of the lexical chains of a text give a representation of the lexical cohesive structure of the text (Tatar, et al., 2008)

In (Chengcheng , 2010), the steps of the lexical chain algorithm are: first, select the set of candidate words which are the results of the noun filtering process; second, the set of candidate words are exploded into senses, which are obtain from the WordNet thesaurus and each word sense is represented by distinct sets considered as levels; third, find the semantic relationship among the set of senses according to its representation. Fourth, build up chains that are sets such as:

$$\{ (\text{word}_1[\text{sense}_{11}, \text{sense}_{12}, \dots]), \\ (\text{word}_1[\text{sense}_{21}, \text{sense}_{22}, \dots]), \\ \dots \}$$

In which $\text{words}_i\text{-sense}_{ix}$ is semantically related to $\text{word}_j\text{-sense}_{jy}$ for $i \neq j$; fifth, retain the longest chains by relying on the preference criterion.

This example shows to us one implementation of the lexical cohesion approach as the driving engine of the summarization system.

- Graph theory

Graph method used for representing the structure of the text and the relationship between sentences. This method consists of nodes and edges. The nodes presented as the sentences in document and the edges presented relations between nodes. The relations are measured as a function of likelihood between contents. By using different similarity criteria, the similarity between two sentences is calculated and each sentence is scored. The sentences with higher scores are chosen for final summary. A graph based method is used for

computing relative importance of textual units for natural language processing (Vishal, 2010).

- Clustering

Clustering is used to sort the huge amount of data by grouping similar data items together. One advantage for using clustering algorithms is to provide automated tools to help in constructing text summarization and classification. This method can ensure the coherence of summaries by using the rank score of sentences with consideration of semantic features (Leil, et al., 2010). There are two different types of clustering:

- Static clustering: the vector of the cluster won't be changed by adding new document.
- Dynamic clustering: the vector of the cluster will be changed by adding a new document (Mehdi, et al., 2010).

3.2.3 Combinational Statistical methods and Linguistic methods

It is text summarization approach which uses both linguistic and statistical methods. This hybrid approach taking into account both linguistic and statistical hints to recognize terms.

In (Shams, et al., 2010), this approach is used to summarize domain specific text from single web document. To do that, they used two novel features Sentence Weight and Subject Weight to rank sentences and then used a representative domain specific corpus for the domain Direct Current (DC) electrical circuits. The Results showed that 68% of the resulted summaries satisfy the manual summaries.

In (Chang, et al., 2008) they summarized domain specific text from a single text document using both linguistic and statistical methods. The purpose of this paper is to propose a hybrid approach to text summarization. They propose a K-mixture connective-strength-based (KCS) approach to improve the quality of document summary results. Result performance under summary proportion of 40% is still quite close to the best one (i.e. 50% summary proportion).

that is, they have no satellites and the connected propositions have the same importance; otherwise, relations are mononuclear, with one nucleus and one satellite (e.g., concession and non-vol-result relations). RST originally defines around 25 relations.

3.3 Comparison between the four approaches

- Statistical methods derive weights of key terms and the total weight that determine the importance of sentence, while the linguistics methods identify term relationships in the document through Part of Speech (POS) tagging, grammar analysis, thesaurus usage, and extract meaningful sentences (Shams,etal, 2010) .
- Statistical approaches are efficient in calculation the weight of the sentences and extraction sentences but linguistic approaches look into term semantics, which may make a better summary results than statistical (Shams, et al., 2010) .
- The statistical approach is domain-independent and fast, but has an inherent upper bound of performance.
- Linguistic methods are expensive because they need deep knowledge resources and text interpretation techniques; on the other hand, they may perform better than statistical methods.
- In Combinational Statistical methods and Linguistic methods, the reliability of a statistical measure increases when applied over linguistic justified candidates. (Pazienza, et al., 2005) .
- The Rhetorical Structure Theory can generate complete, correct and readable summaries based on understanding the important sentence and the relation between them in original text (Pardo, et al., 2008) . In table (3 - 1) Comparison between the four approaches.

3.5 Summary

In this chapter we presented a summary of literature reviews related to this project. Firstly, we showed in the introduction of the chapter the four extraction approaches of text summarization, which are statistical, linguistic, combinational statistical and linguistic and rhetorical structure theory. Also we presented the different methods that are used in each approach. Then, we made a comparison between those approaches.

Depending on that, we determined the approach that we will use which is combinational statistical methods and linguistic, because it seems that the reliability of a statistical measure increases when applied over linguistic justified candidates, and this what we will prove in the following chapters.

Chapter Four

Methodology

Contents:

- 4.1 Overview
- 4.2 Preparation phases for the implementation of the project
- 4.3 Defining the requirements of project
- 4.4 Collecting samples of documents
- 4.5 Preprocessing documents
- 4.6 Feature Extraction
- 4.7 Summary Generation
- 4.8 Post Processing
- 4.9 Summary

4.1 Overview

In this chapter we will discuss the methodology that we used in this project to summarize a single English text document. The chapter includes the preparation phases for implementing this project. Then, we will define the requirements of our project. After that we will describe the preprocessing, features extraction, summary generation and post processing stages.

4.2 Preparation phases for the implementation of the project

Depending on the previous studies which talked about automatic text summarization approaches, we decided to use the combinational statistical methods and linguistic methods in our project.

4.3 Defining the requirements the project

Before talking about the methodology of this project, we will define the conditions which present the properties of the source text. These conditions are:

1. Mono language text, and the language that we chose was (English language).
2. Specific computer science articles domain was chosen to be the subject that we want to study.
3. In our study, we relied on using single document.

4.4 Collecting samples of documents

Computer science articles are taken as a case study in this project. As a result, the second step of this project is collecting articles of computer science.

4.5 Preprocessing documents

Preprocessing document stands for four stages. Firstly, stopwords will be omitted in order to deal with content words easily. Secondly, every abbreviation will be replaced with its full words. Thirdly, we mention the word's part of speech. Finally, segmentation will be done. The following figure(4 - 1) contains an explanation for these four activities.

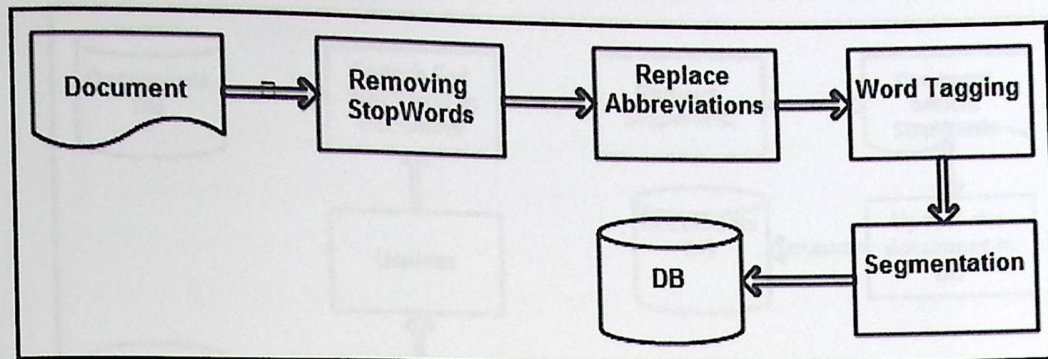


Figure (4 - 1): Preprocessing stage.

4.5.1 Removing stopwords

Stopwords are defined as general words that carry less important meanings than other words, such as “about”, “at”, “the”... etc. A list of 368 stopwords will be used in this step. Comparable lists are available in many places on the World Wide Web. Stopwords will be removed by following two steps:

1. Classifying stopwords to useless or useful words. Examples of useful words are (conclusion example, today...). But useless words are (at, the, some ...).
2. Removing useless stopwords.

See the stopwords list in the Appendix (B).

Algorithm(4 - 1) will be used for removing stopwords. The objective of removing stopwords is to reduce the document size, so reduce the parsing times.

```

N = number of words in document
k[ ] = stop word list
j[ ] = useless list
Do while word number <> N
  Read document
  If (word include? in j) Then
    Remove word
  End if
  Next word
  Store document
End do
  
```

Algorithm(4 - 1)

Removing stopwords is shown in figure (4-2).

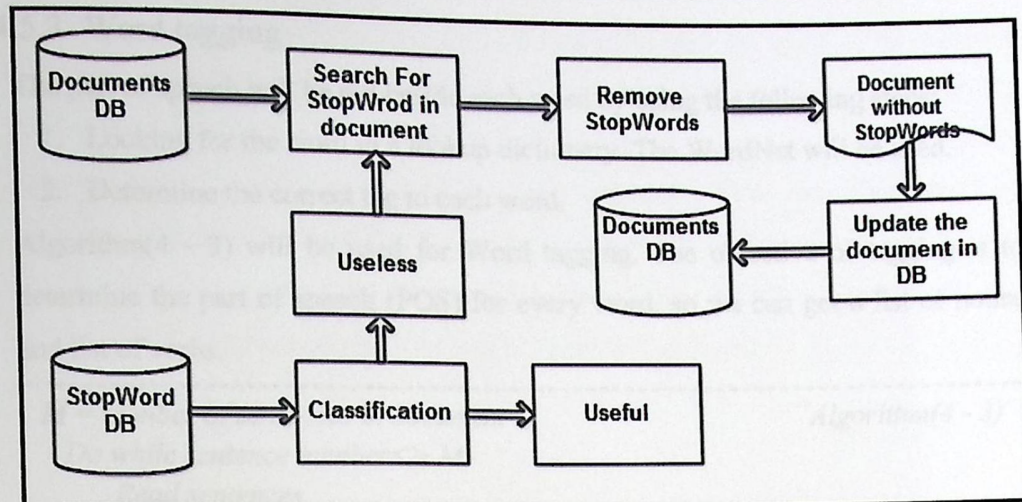


Figure (4 - 2): Removing stopwords step.

4.5.2 Referring abbreviations

Every abbreviation will be referred to its original words. Algorithm(4 - 2) will be used. The objective of this step is to know the real number of terms.

N = number of words in document

abb[] = Abbreviations list

Do while word number <> N

Read words

If (word include? in abb) Then

Replace the word by the original text

Update words

End if

Next word

End do

Algorithm(4 - 2)

Referring abbreviations is shown in figure (4 - 3).

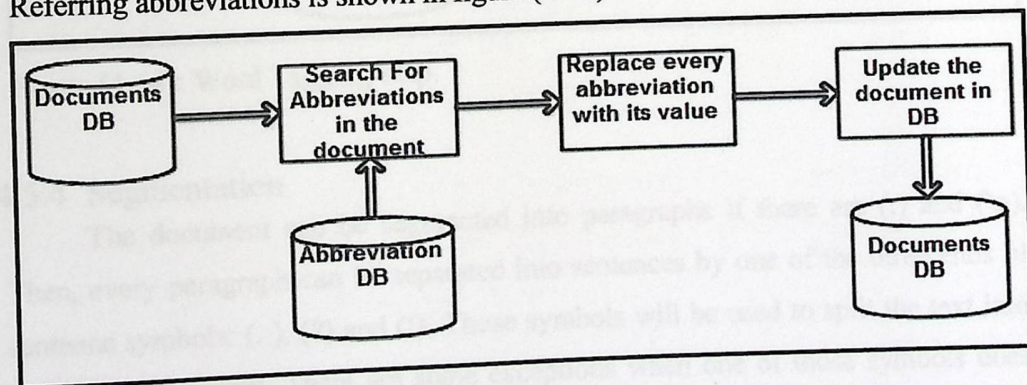


Figure (4 - 3): Referring abbreviations step.

4.5.3 Word tagging

The part of speech will be put beside each word by using the following steps:

1. Looking for the word in a lookup dictionary. The WordNet will be used.
2. Determine the correct tag to each word.

Algorithm(4 - 3) will be used for Word tagging. The objective of tagging is to determine the part of speech (POS) for every word, so we can get a list of nouns and list of verbs.

```

M = number of sentences in document
Do while sentence number <> M
  Read sentences
  Scan the sentence by WordNet
  Store WordTag
End do
  
```

Algorithm(4 - 3)

Word tagging is shown in figure (4 - 4).

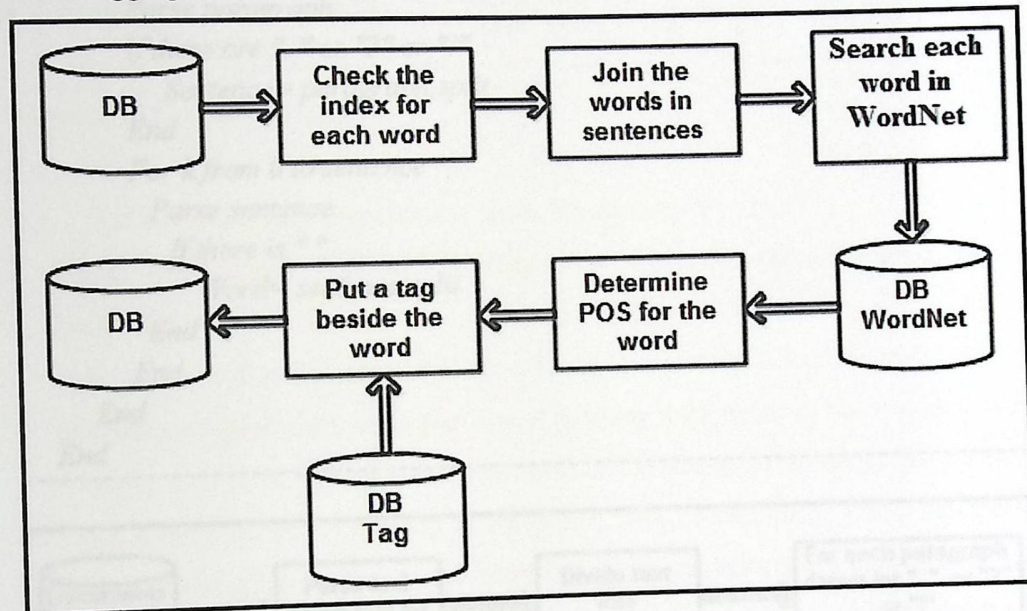


Figure (4 - 4): Word Tagging Step.

4.5.4 Segmentation

The document can be segmented into paragraphs if there are (.) and (\n). Then, every paragraph can be separated into sentences by one of the three ends of sentence symbols: (.), (?), and (!). Those symbols will be used to split the text into individual sentences. There are some exceptions when one of those symbols does not indicate a sentence boundary. For example, MS., DR., and 100.00, this symbol

(.) does not indicate sentence boundary, it must followed by space to be sentence boundary. After that every sentence can be segmented into words depending on the spaces between the words. By the end of this step, we can get a word and its mapping with its address (paragraph, and sentence). This step facilitates the processes of extracting features. Because we can easily find the words and extract their features, we can find the sentences and extract their features. Algorithm (4 - 4) will be used for segmentation. This step is illustrated in figure(4 - 5).

Algorithm(4 - 4)

```

Mytext=input document
For i from 0 to mytext.size
  Parse mytext
  If there are ("\\n" & ". ")
    Paragraph= mytext.split
  End
  For j from 0 to paragraph.size
    Parse paragraph
    If there are ". " or "?" or "!"
      Sentence= paragraph.split
    End
    For k from 0 to sentence
      Parse sentence
      If there is " "
        Word= sentence.split
      End
    End
  End
End
End
  
```

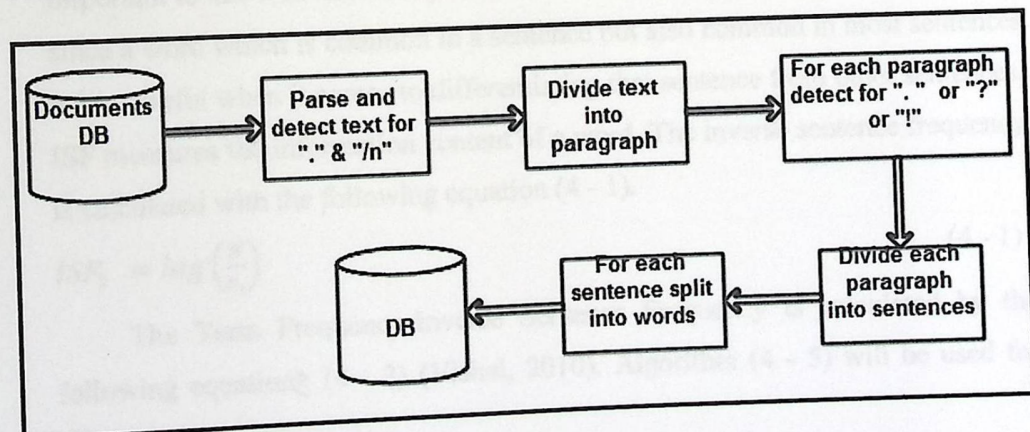


Figure (4 - 5): Segmentation step.

4.6 Feature Extraction

In this step we will extract both the sentence level and the word level features which will be used in calculating the importance or relevance of the sentence towards the document.

4.6.1 The word level features

The important words can be extracted from the text by using a set of features.

These features are:

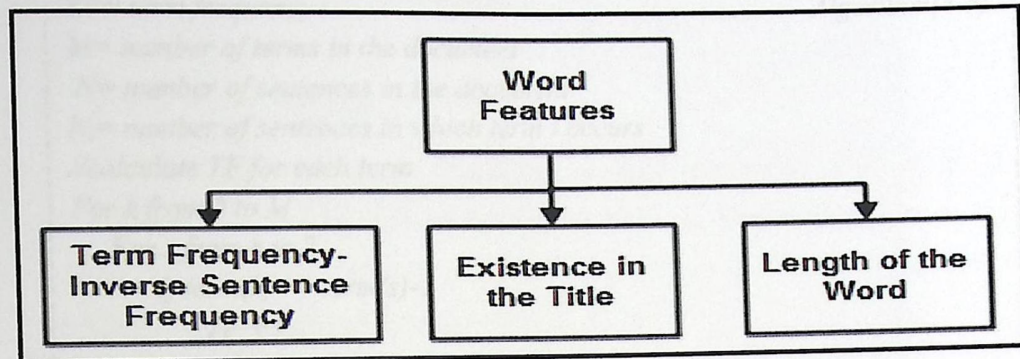


Figure (4 - 6): Word Features.

1. Term Frequency- Inverse Sentence Frequency(TF_ISF)

Term Frequency: This is the number of times the term occurred in the document

Inverse Sentence Frequency: Terms that occur in only a few sentences which are more valuable than ones that occur in many sentences. In other words, it is important to know in how many sentences of the document a certain word exists since a word which is common in a sentence but also common in most sentences is less useful when it comes to differentiating that sentence from other sentences. ISF measures the information content of a word. The inverse sentence frequency is calculated with the following equation (4 - 1).

$$ISF_i = \log \left(\frac{N}{n_i} \right) \quad (4 - 1)$$

The Term Frequency-Inverse Sentence Frequency is calculated by the following equation (4 - 2) (Vishal, 2010). Algorithm (4 - 5) will be used for TF_ISF.

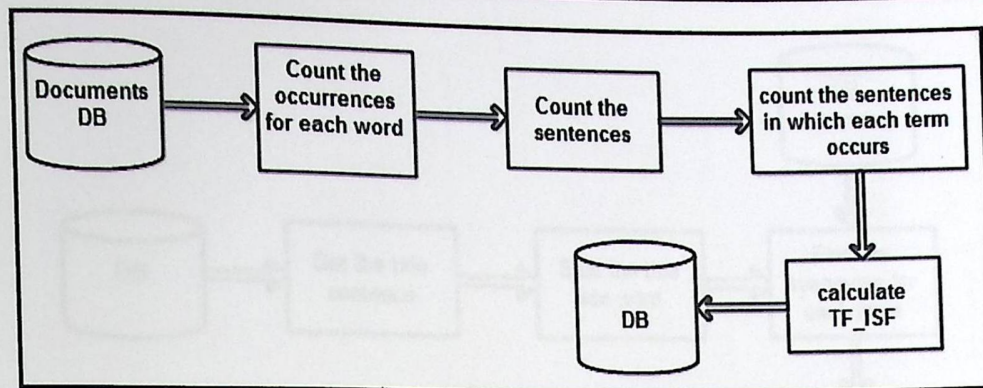


Figure (4 - 7): Step for Calculation of TF_ISF.

2. Existence in the title of document

Titles and headings are considered as short summaries of the texts. The words that exist in the title or their synonyms are important and have high scores. Algorithm (4 - 6) will be followed to extract this feature. This feature will be computed as the following:

$$\text{Existence in the title}(w) = \begin{cases} 1, & \text{if found in the title} \\ 0, & \text{if not found in the title} \end{cases}$$

M = number of terms in the document

Algorithm(4 - 6)

N = number of terms in title

For *i* from 1 to *M*

For *j* from 1 to *N*

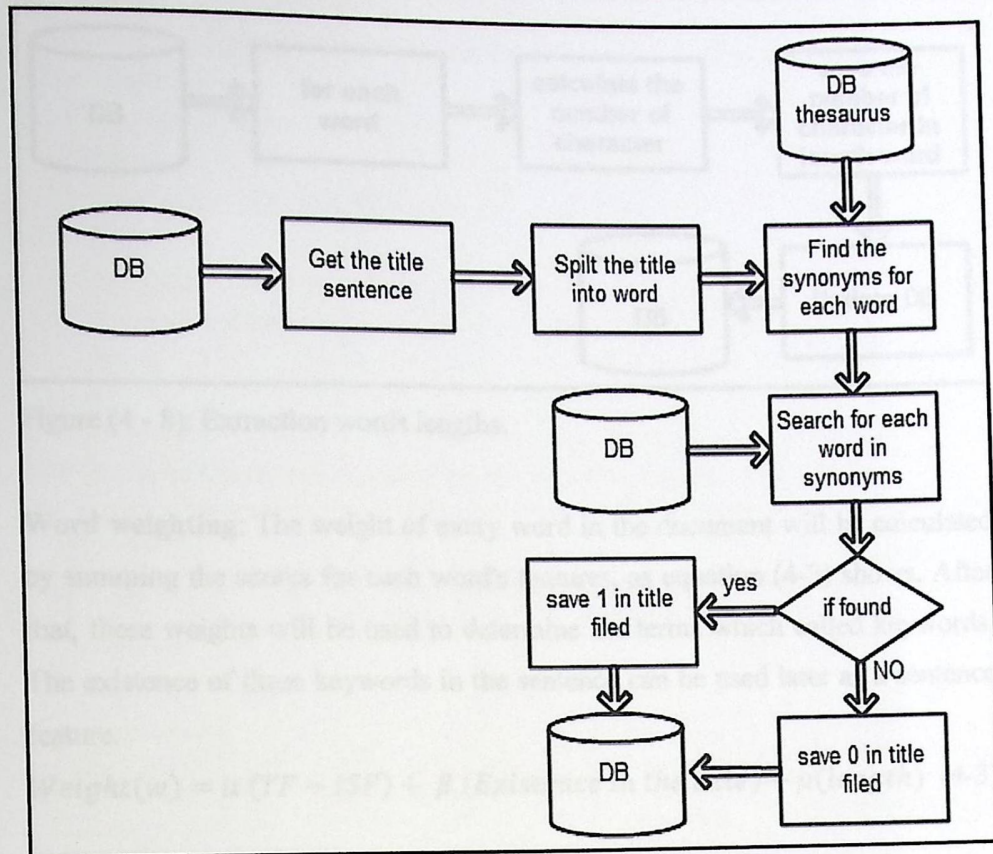
If title[*j*] == term[*i*]

score++

Next *j*

Next *i*

This step is illustrated in figure (4-8).



Figure(4 - 8): Extraction existence in the title of document.

3. Length of word

Algorithm (4 - 7) will be followed to extract the length of the word. This feature will be computed as the following:

$$Length(w) = \begin{cases} 1, & \text{if } length \geq threshold \\ 0, & \text{if } length < threshold \end{cases}$$

M = number of words in the document
Length = word length
 For *i* from 1 to *M*
 $Length[i] = term[i].length$
 Next *i*

Algorithm(4 - 7)

The figure (4 - 8) shows the extraction of lengths feature:

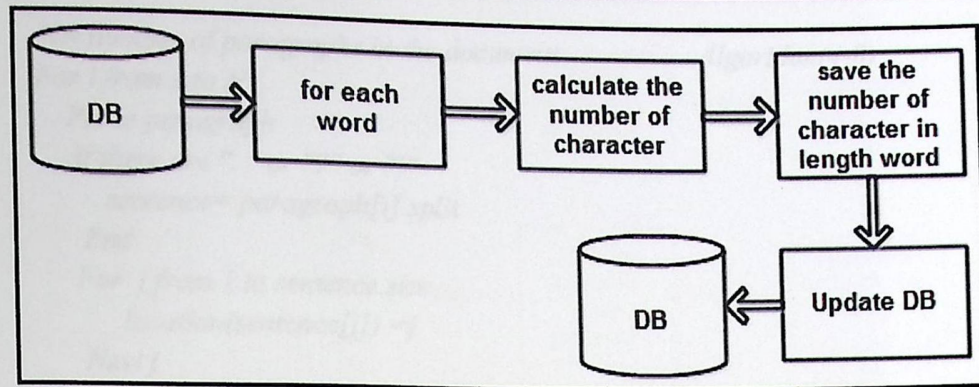


Figure (4 - 8): Extraction words lengths.

Word weighting: The weight of every word in the document will be calculated by summing the scores for each word's features, as equation (4-3) shows. After that, these weights will be used to determine the terms which called keywords. The existence of these keywords in the sentence can be used later as a sentence feature.

$$Weight(w) = \alpha (TF - ISF) + \beta (Existence\ in\ the\ title) + \mu(length) \quad (4-3)$$

4.6.2 The Sentence level features

Important sentences can be extracted by using a set of features which are described in the figure(4-10):

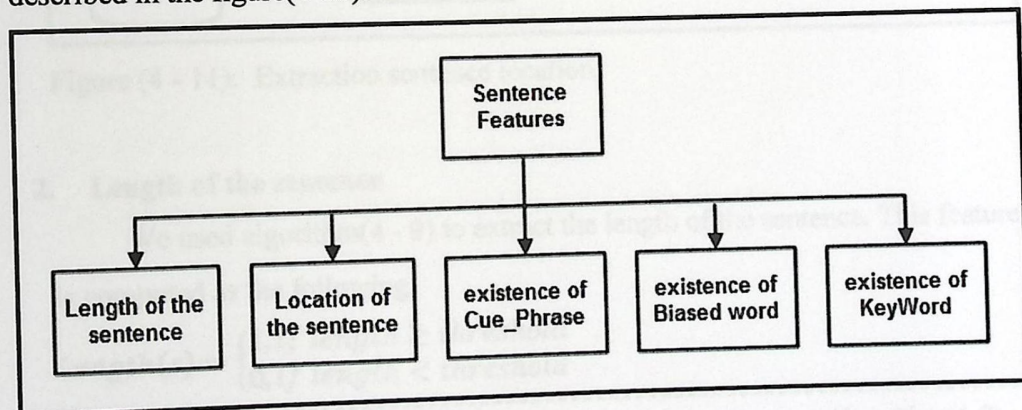


Figure (4-10): Sentences Features.

1. Location of the sentence

Algorithm (4 - 8) will be used to determine the location of the sentence .

Location can be computed as the following:

$$Location(s) = \begin{cases} 1, & \text{if the sentence location} = \text{threshold} \\ 0 & \text{Otherwise} \end{cases}$$

Algorithm(4-8)

```

N= number of paragraphs in the document
For i from 1 to N
  Parse paragraph
  if there are ". " or "?" or "!"
    sentence= paragraph[i].split
  End
  For j from 1 to sentence.size
    location(sentence[j]) =j
  Next j
Next i

```

The location feature will be extracted as the following figure (4-11)

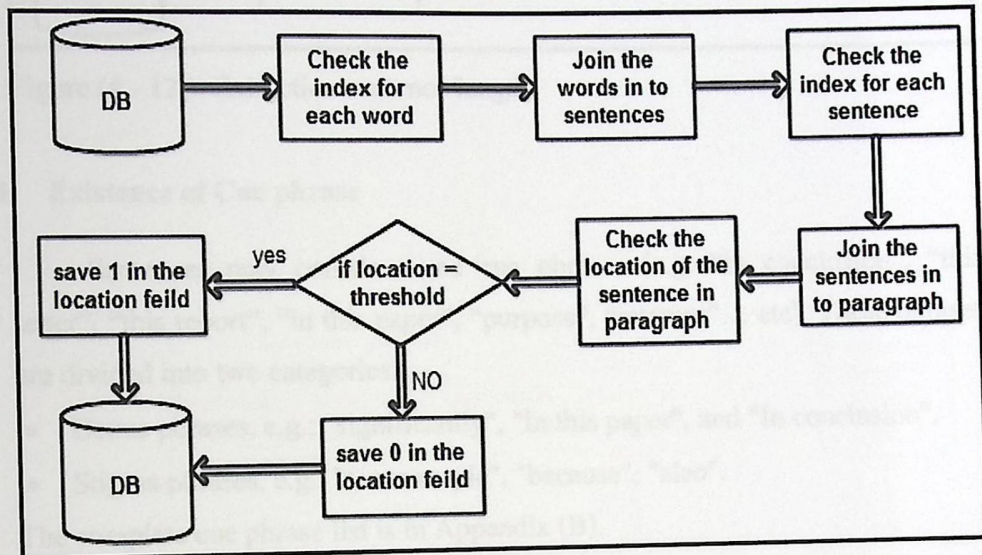


Figure (4 - 11): Extraction sentence location.

2. Length of the sentence

We used algorithm(4 - 9) to extract the length of the sentence. This feature is computed as the following:

$$Length(s) = \begin{cases} 1, & \text{if } length \geq threshold \\ 0, & \text{if } length < threshold \end{cases}$$

Algorithm(4-9)

```

N=number of sentences in the document
For i from 0 to N
  W= sentence[i].split
  Length=word.size
Next i

```

Figure (4 - 12) shows the extraction of sentence length feature:

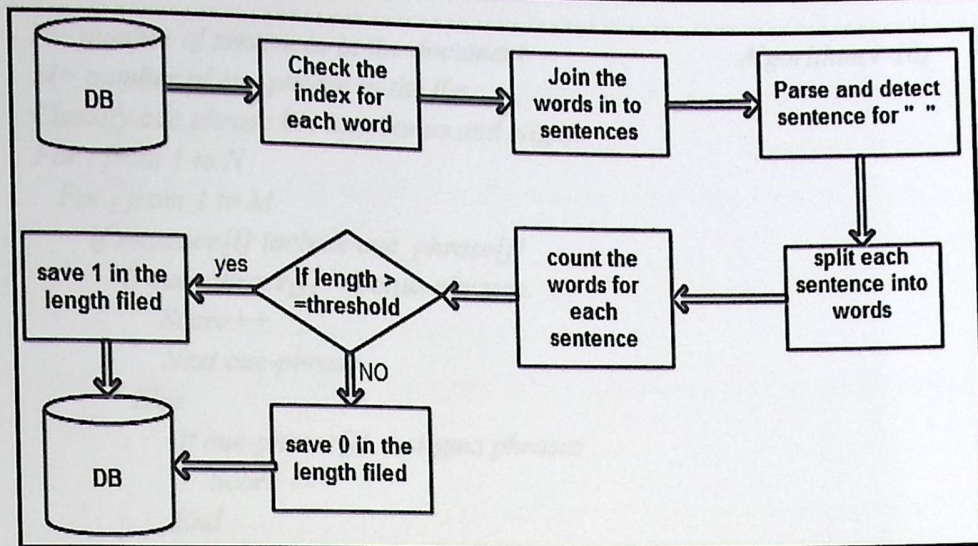


Figure (4 - 12): Extraction sentence length.

3. Existence of Cue phrase

Sentences may contain some cue phrases (e.g. "in conclusion", "this letter", "this report", "in this paper", "purpose", "attempt"... etc). These phrases are divided into two categories:

- Bonus phrases, e.g.: "significantly", "In this paper", and "In conclusion".
- Stigma phrases, e.g.: "for example", "because", "also".

The complete cue phrase list is in Appendix (B).

This feature can be computed as:

$$\text{Cue phrase}(s) = \begin{cases} 1, & \text{if bonus phrase} \\ 0, & \text{if Stigma phrase} \end{cases}$$

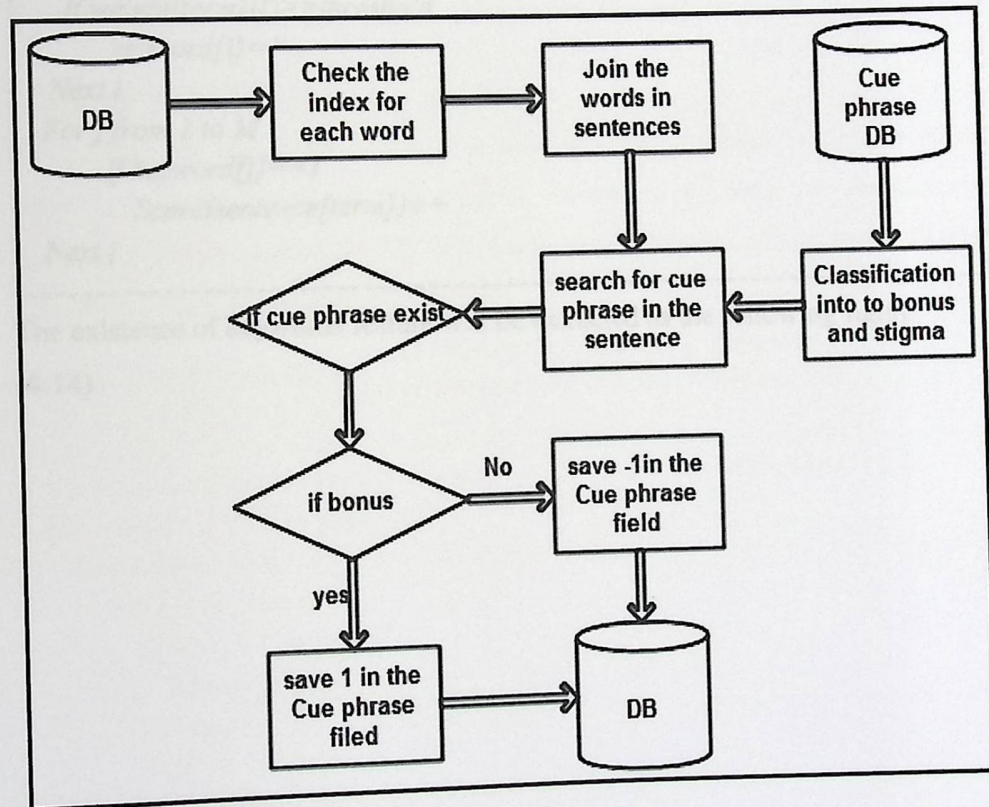
Algorithm(4-10) will be followed for extracting this feature.

Algorithm(4-10)

```

N= number of sentences in the document
M= number of cue phrase in the list
Classify cue phrase list into bonus and stigma
For i from 1 to N
  For j from 1 to M
    If sentence[i] include cue_phrase[j]
      If cue_phrase[j] = bonus phrases
        Score++
      Next cue-phrase
    Else
      if cue-phrase[j] = stigma phrases
        Score --
      End
    End
  End
Next j
Next i
  
```

The existence of cue phrase feature will be extracted as figure (4-13) shows.



Figure(4 - 13): Extraction existence of cue phrase.

4. Existence of keywords

The weights for words will be calculated as we mentioned earlier in this chapter. These weights will be used to determine the keywords from the document's words. If the word's weight will be more than such threshold, the word can be selected as a keyword, as:

$$KeyWord(w) = \begin{cases} 1, & \text{if } weight(w) \geq threshold \\ 0, & \text{if } weight(w) \leq threshold \end{cases}$$

The sentences which include keywords will be had more score, as the following formula:

$$Existence\ of\ Keywords(s) = \begin{cases} 1, & \text{if } number\ of\ keyword \geq threshold \\ 0, & \text{if } number\ of\ keyword < threshold \end{cases}$$

Algorithm(4 - 11) can be followed to extract this feature.

<pre> N= number of terms in the document M= number of terms in the document For i from 1 to N If weight(term[i])>=threshold keyword[i]=1 Next i For j from 1 to M If keyword[j]==1 Score(sentence[term])++ Next j </pre>	<i>Algorithm(4-11)</i>
---	------------------------

The existence of keywords feature will be extracted as the following figure (4-14)

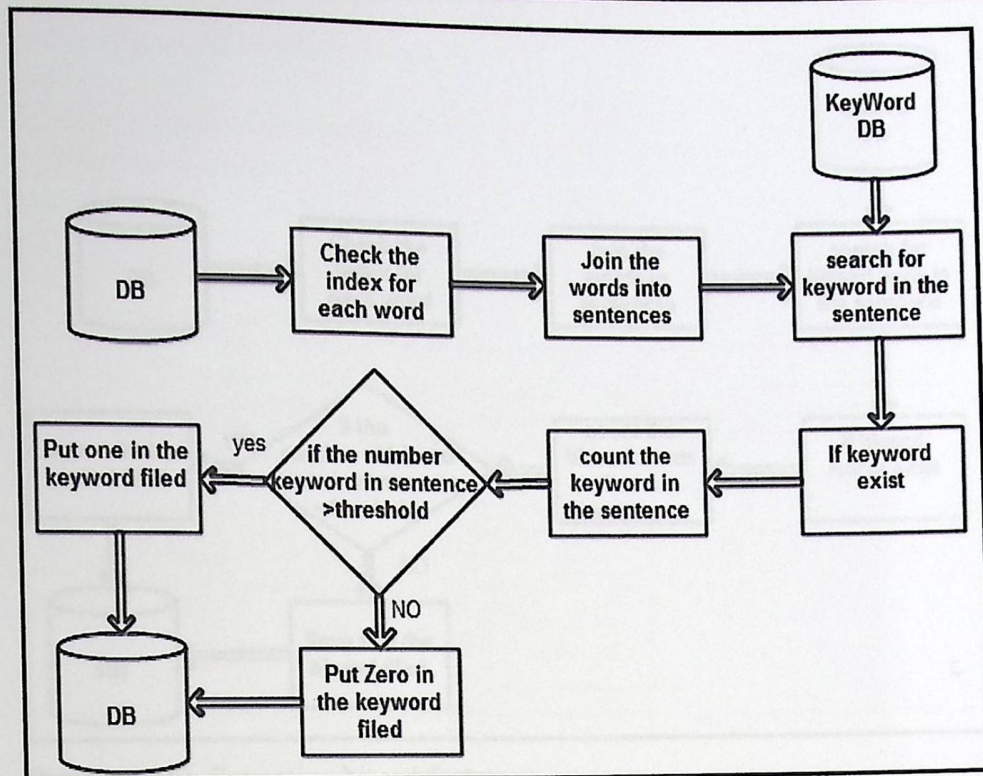


Figure (4-14): Extraction existence of keywords feature.

5. Biased Word Feature

If biased words are found in a sentence, the sentence is considered as important. Biased words list is previously defined, and it may contains domain specific words. Algorithm(4-12) will be followed for extraction biased words. This feature can be computed as:

$$\text{Biased Word Feature}(s) = \begin{cases} 1, & \text{if number of biased} \geq \text{threshold} \\ 0, & \text{if number of biased} < \text{threshold} \end{cases}$$

N = number of terms in the document

M = number of biased in the list

For *i* from 1 to *N*

For *j* from 1 to *M*

If term[*i*] = biased[*j*]

Score++

Next *i*

Else

Next *i*

Next *j*

Algorithm(4-12)

Figure (4-9) shows the extraction of biased words existence.

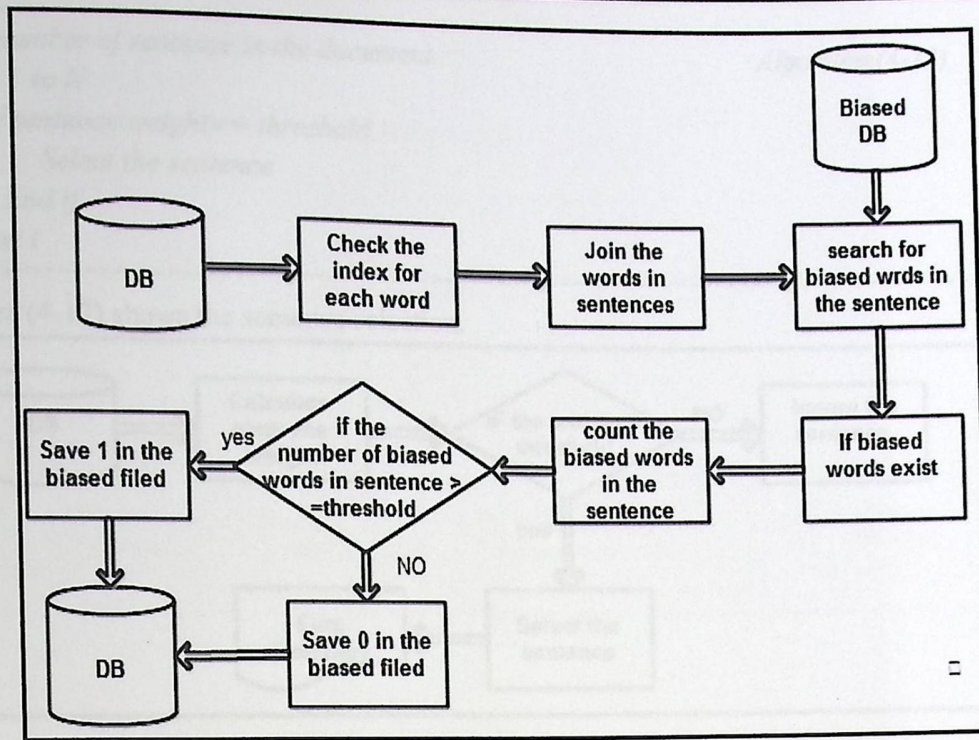


Figure (4-15): Extraction biased feature.

Sentence weighting: by the end of this stage, the weight of every sentence in the document will be calculated by summing its features scores, as equation (4-4) shows.

$$\begin{aligned}
 \text{Weight}(s) = & \alpha(\text{location}) + \beta(\text{length}) + \mu(\text{cue} - \text{phrase}) \\
 & + \gamma(\text{existence of keyword}) + \theta(\text{biased})
 \end{aligned}
 \quad (4-4)$$

4.7 Summary Generation

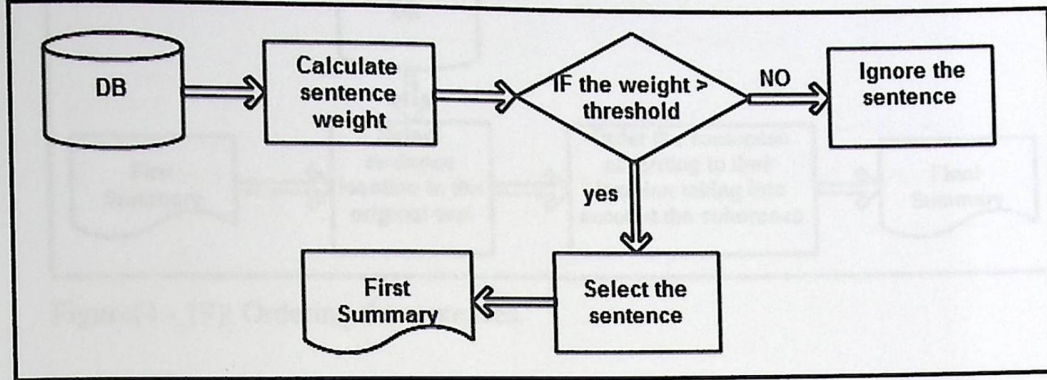
Summary generation will be the third stage in this project. This stage will produce the first summary. The sentences that have weights more than such threshold will be selected to be in the first summary. Algorithm(4 - 13) will be followed for sentence selection.

$$\text{Sentence Selection}(s) = \begin{cases} \text{Selected, if } \text{weight}(s) > \text{threshold} \\ \text{not selected, if } \text{weight}(s) < \text{threshold} \end{cases}$$

$N = \text{number of sentence in the document}$
 For i to N
 If sentence weight $\geq$ threshold
 Select the sentence
 End if
 Next i

Algorithm(4-13)

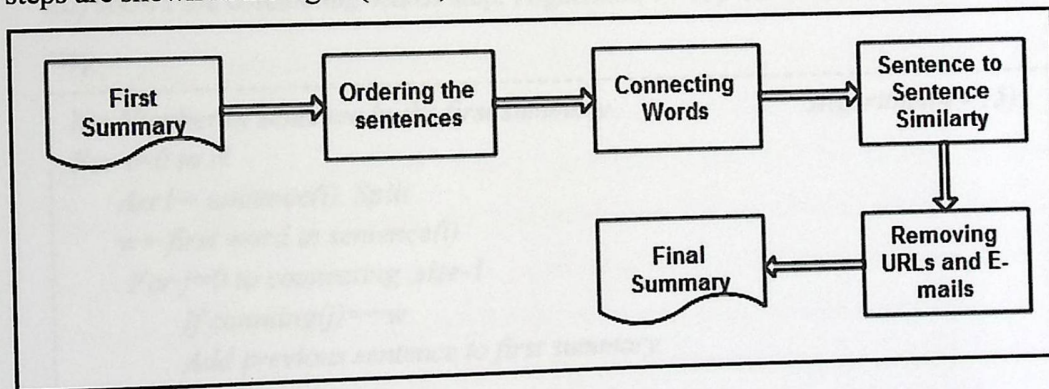
Figure (4-17) shows the sentence selection.



Figure(4 - 17): Sentence selection.

4.8 Post processing

After the extraction and generation of the first summary, some post processing steps will enhance the summary and produce the final summary. The post processing steps are shown in the figure(4 - 18).



Figure(4 - 18): Post processing stage.

4.8.1 Ordering the sentences

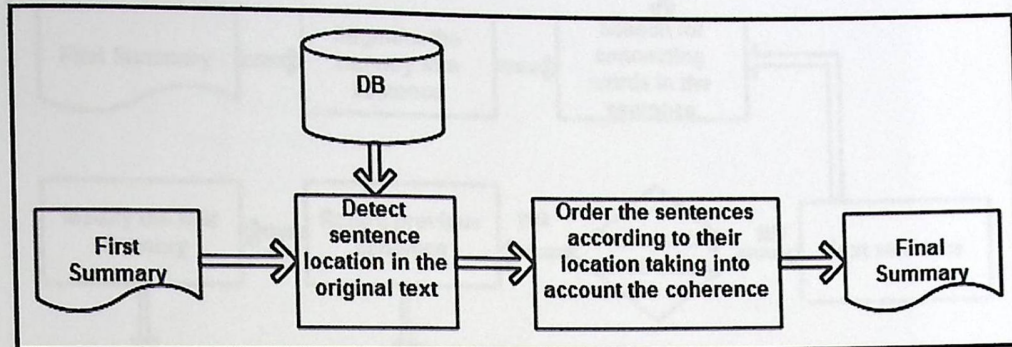
The highest weight sentences will be organized in order to get the summary of the document. Algorithm(4 - 14) will be followed for ordering the sentences, as shown in figure(4 - 19).


```

N = number of sentence in the summary
For i from 1 to N
    Sort(sentence[i]) by location(sentence[i])
End

```

Algorithm(4-14)



Figure(4 - 19): Ordering the sentences.

4.8.2 Connecting words

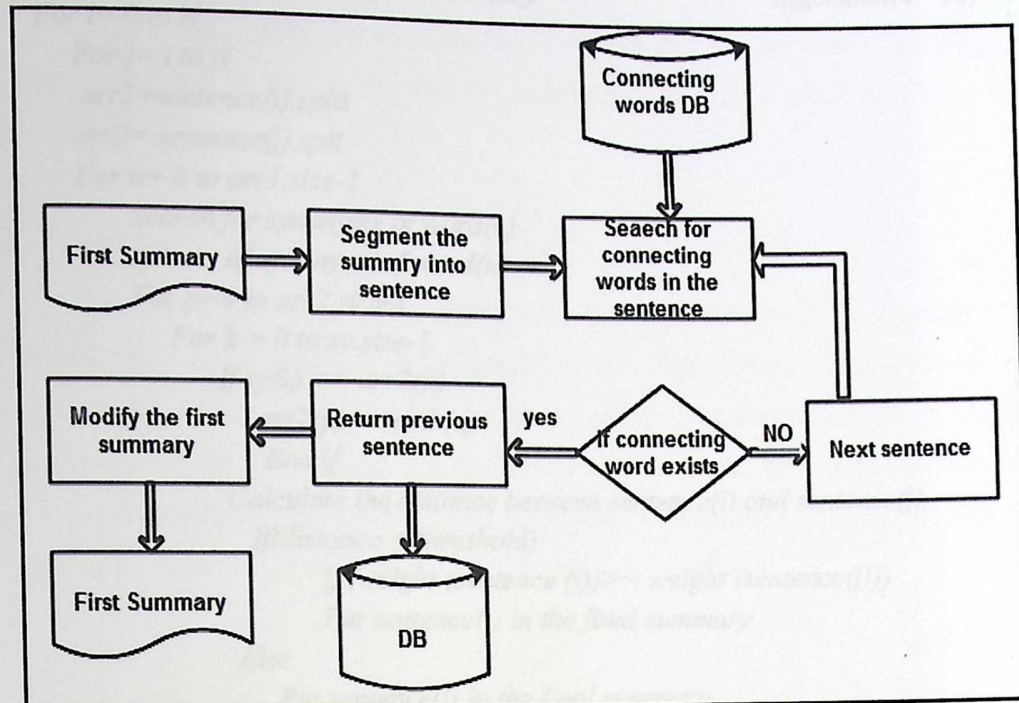
The appearance of connecting words such as "accordingly", "again", "also", "besides"... etc at the beginning of the sentence provides a coherence with the previous sentence. Linguistically, a sentence cannot start with the above words without any related introduction in the previous sentence. The following figure(4 - 20) shows the connecting words step. Algorithm(4 - 15) will be used to achieve this step.

```

N = Number of sentence in the first summary
For i=0 to N
    Arr1 = sentence(i). Spilt
    w = first word in sentence(i)
    For j=0 to connecting .size-1
        If connting(j)==w
            Add previous sentence to first summary
        Next j
    Else
        Next i
    End if
End for
End for

```

Algorithm(4 - 15)



Figure(4 - 20): Connecting words steps.

4.8.3 Sentence to sentence similarity

The sentences which are appearing in different locations of the document may have the same meaning. To produce a good summary, we should eliminate the redundancy between the sentences in the first summary. In order to solve this problem, distance can be calculated by using a dictionary to find the synonyms for each word in each sentence; also a threshold value will be defined. If two sentences' distance is less than the threshold, they will be considered repeated. So the sentence with higher weight will be selected to be in the final summary, as it is shown in figure (4-21). Algorithm(4 - 16) will be followed to achieve this step.

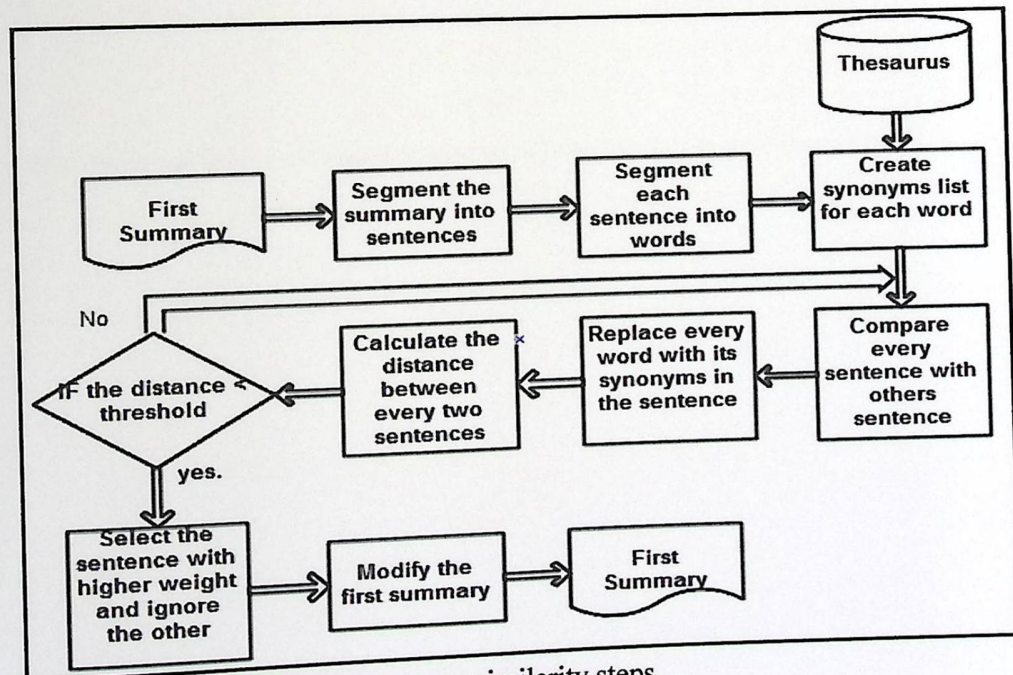
$$\text{Sentence Selection}(s) = \begin{cases} \text{Similarity, if distance}(s) > \text{threshold} \\ \text{not similarity, if distance}(s) < \text{threshold} \end{cases}$$

Algorithm(4 - 16)

```

N= Number of sentences in first summary
For i= 0 to N
  For j= i to N
    arr1=sentence(i).split
    arr2= sentence(j).split
    For u= 0 to arr1.size-1
      Search for synonyms of word(u)
      sy= set of synonyms of word(u)
      For p=0 to arr2.size-1
        For k = 0 to sy.size-1
          If sy(k) == arr2(p)
            arr2(p)= word (u)
          End if
        Calculate the distance between sentence(i) and sentence(j)
        If(distance < threshold)
          If (weight (sentence (i))>= weight (sentence(j)))
            Put sentence(i) in the final summary
          Else
            Put sentence(j) in the final summary
          End if
        End if
      End for
    End for
  End for

```



Figure(4 - 21): Sentence to sentence similarity steps.

4.8.4 Removing URLs and e-mails

URLs and e-mails are not important details in the final summary. The regular expressions will be used in this step to select these URLs and e-mails then they will be removed.

4.9 Summary

This chapter includes the methodology that will be used to generate the summary from the input text. The first stage will be preprocessing of the document to prepare it for processing. Then, the words and sentences features will be extracted. These features can give scores for the sentence. Then, the first summary can be generated by selecting the most important sentences. Finally, the first summary will be enhanced by post processing steps to produce the final summary.

Contents

5.1 Overview

5.2 Software Environment

5.3 Database Schema

5.4 Summary

5.1 Overview

This chapter describes the project simulation. It includes description of the software environment and the language that we used. We also describe the database schema of our simulation.

5.2 Software Environment

Many software packages were used to achieve the project simulation. In this

Chapter Five

Programming

Contents:

5.1 Overview

5.2 Software Environment

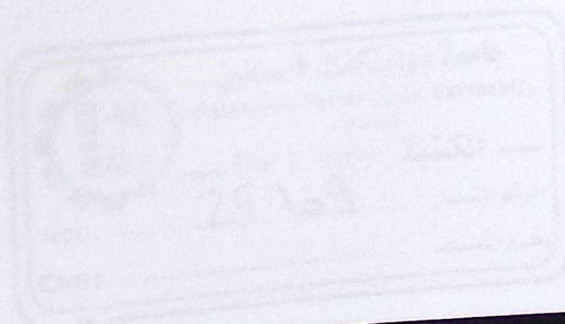
5.3 Database Schema

5.4 Summary

Perl, JavaScript, PHP, Ada and Lisp. The Ruby language was created by Yukihiro Matsumoto (commonly known as Matz), and it was first released in 1997 (Collingbourne, 2007).

Why we used Ruby?

1. Ruby is a feature-rich, fast, simple and extensible language.
2. Ruby is a powerful and powerful language.
3. Ruby has powerful built-in libraries and a set of external libraries.
4. Ruby has an English language syntax.



5.1 Overview

This chapter describes the project simulation. It includes description of the software environment and the language that we used. We also describe the database schema of our simulation.

5.2 Software Environment

Many software packages were used to achieve the project simulation. In this section, we will describe the software environment for our simulation, such as: NetBeans IDE, Ruby language and its RubyGems that we used, and MySQL Server.

5.2.1 NetBeans IDE 6.9.1

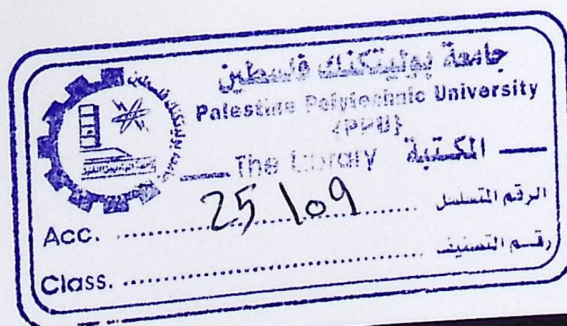
The NetBeans IDE is an integrated development environment available for Linux, Mac and Windows. The NetBeans project consists of an open source and an application platform that enables developers to rapidly create web, desktop, enterprise, and mobile applications using the Java platform, as well as, PHP, JavaFX, JavaScript and Ajax, Ruby and C/C++(Jerry,2010).

5.2.2 Ruby language

Ruby is a cross platform interpreted language which has many characteristics in common with other scripting languages such as Python and Perl. It has an English language syntax which looks somewhat like Pascal code at first sight. The languages which were mostly used in the development of Ruby are the following: Perl, Smalltalk, Eiffel, Ada and Lisp. The Ruby language was created by Yukihiro Matsumoto (commonly known as Matz), and it was first released in 1995(Collingbourne,2009).

Why we used Ruby?

1. Ruby is a feature-rich, free, simple and extensible language.
2. Ruby is a powerful text processing language.
3. Ruby has powerful built-in libraries and a set of external libraries.
4. Ruby has an English language syntax.



5.2.3 RubyGems

RubyGems is a package management system for applications and libraries of Ruby. It helped in installing RubyGems to any computer that is using Ruby. It can resolve version dependencies. So if a certain gem or the code requires a certain version of another gem, it can take care of that. Most gems will work under both Linux and Windows since Ruby is an interpreted language, and most gems are written in Ruby. However, not all gems are pure Ruby source; some include C extensions or libraries. Those gems must be compiled before they are used so we used DevKit tools to compile these kind of gems(Berube,2007).

Why we used RubyGems?

1. RubyGems facilitates installing Ruby software.
2. It handles software dependencies.
3. It facilitates using the same technology on any operating system.

We used the following RubyGems in our simulation:

1. WordNet: "Ruby-WordNet is a Ruby interface to the WordNet® Lexical Database. WordNet is an online lexical reference system. WordNet organizes English nouns, verbs, adjectives and adverbs are organized into sets"(Basu,2009). We used WordNet to determine the part of speech for each word.
2. Part-Of-Speech:"Part of speech tagger based on Mark Watsons code"(Reddavis,2010), we used it to determine the POS for each word.
3. Ruby MySQL: MySQL connector for Ruby. MySQL which supports Ruby was developed by Tomita Masahiro. He has developed a pure Ruby binding called Ruby/MySQL(Talim,2007). This gem helped us to connect with MySQL Server 5.1.
4. Metonym: We used a thesaurus in www.thesaurus.com to search for synonyms/antonyms(Yared,2011). This gem helped us in this task.
5. Text: A collection of text algorithms, Metaphone, Porter Stemming, Levenshtein, Figlet and Double Metaphone(Battley,2010). We used Levenshtein to calculate the distance between the sentences.

5.2.4 DevKit

The DevKit is a toolkit that helps in building and using native C/C++ extensions for Ruby on Windows. One of the difficulties that The DevKit is a toolkit that helps in building and using native C/C++ extensions for Ruby on Windows. One of the difficulties that many Windows users have with Ruby is how to easily install and use native RubyGems. We used DevKit to install some native RubyGems such as Linguistic gem (GitHub,2011).

5.2.5 MySQL Server 5.1

MySQL is a system like any DBMS, it allows the user to access and manipulate data within their databases, protect the data from corruption and inconsistencies, and maintain the metadata necessary to define the data being stored. MySQL is specific to relational databases. MySQL has a large set of characteristics that provides a secure environment for storing, maintaining, and data accessing.

SQL is a fast, reliable, scalable alternative to many of the commercial RDBMSs available today(Publishing, 2005). Some characteristics of MySQL are:

1. Scalability.
2. Portability.
3. Connectivity.
4. Speed.
5. Ease of use.
6. Open-source code.

5.3 Database Schema

The system database contains the whole information that the system needs to accomplish its functions and operations. The database consists of six tables which are: Paragraph, Sentences, Words, Cue-Phrase, Stopwords and Connecting word.

- Paragraph: this table stores all paragraphs that the document contains.

Filed name	Data type	Required	Key	Description
IDP	Int	Yes	PK	paragraph number
Length	Int	Yes		paragraph length

Table (5 - 1): Paragraph table.

- Sentences: this table stores all of the sentences in the document. Every sentence belongs to a certain paragraph.

Filed name	Data type	Required	Key	Description
IDS	Int	Yes	PK	sentence number
IDP	Int	Yes	PK	paragraph number
Location	Int	Yes		location in the paragraph
Length	Int	Yes		sentence length
Cue- phrase	Int	Yes		cue phrase existence
keyword	Int	Yes		keyword existence
Biased	Int	Yes		biased words existence
Weight	decimal	Yes		sentence weight

Table (5 - 2): Sentences table.

- Words: this table stores all of the words in the document. Every word belongs to a certain sentence in a certain paragraph, so by using this table we can recreate the whole text.

Filed name	Data type	Required	Key	Description
IDW	Int	Yes	PK	word number
IDS	Int	Yes	PK	sentence number
IDP	Int	Yes	PK	paragraph number
Word	Text	Yes		the word
POS	Varchar	Yes		part of speech for the word
TF	Int	Yes		term frequency
TF_ISF	Int	Yes		TF-ISF for the word
Exi_Title	Int	Yes		existence in the title
Length	Int	Yes		word length
keyword	Int	Yes		word is keyword or not
Weight	Decimal	Yes		word weight

Table(5 - 3): Words Table.

- StopWords: this table stores the stopwords list that we used.

Filed name	Data type	Required	Key	Description
IDSW	Int	Yes	PK	stopword number
SW	Varchar	Yes		stopword text
Type	Varchar			useful or useless

Table(5 - 4): Stopword table

- Cue phrase: this table stores the cue phrase list that we used. And it shows if that cue phrase is bonus or stigma.

Filed name	Data type	Required	Key	Description
IDCP	Int	Yes	PK	cuephrase number
Cue-Phrase	Varchar	Yes		cuephrase text
Type	Varchar			bonus or stigma

Table(5 - 5):Cue Phrase table

- Connecting words: this table stores the connecting words that we used.

Filed name	Data type	Required	Key	Description
IDCO	Int	Yes	PK	connecting word number
ConnWord	Varchar	Yes		connecting word text

Table(5 - 6): Connecting words table.

5.3 Summary

In this chapter we described our project simulation. In the first section we presented a description of the software environment. Also, we described the language that we used, which was Ruby. Then we discussed the main RubyGems and why we used them in the simulation, such as WordNet, Metonym and Text. After that, we discussed the DBMS that we used which was MySQL. Finally we described the database schema of our project simulation.

1 Overview

In Chapter 4, a set of algorithms were developed to apply them to the project, how algorithms are evaluated and implementation tools for project simulation (see Appendix A). In this chapter the experiments and results for this simulation are covered.

The experiments were done in four steps: pre-processing experiments, extracting

Chapter Six

Experiments and Results

Contents:

6.1 Overview

6.2 Testing Sample

6.3 Experiments and Results

6.4 Evaluation

6.5 Summary

6.3 Experiments and Results

The following section contains the experiments and results for the project steps: pre-processing, feature extraction, summary generation, and post processing.

6.1 Overview

In Chapter 4, a set of algorithms were developed to apply them in the project, these algorithms are converted into implementation code for project simulation (see Appendix A). In this chapter the experiments and results for this simulation are covered.

The experiments were done in four steps; pre-processing experiments, extracting word and sentence features experiments, summary generation experiments and evaluation experiments. The results of these experiments are also presented in this chapter.

Experiments were applied on computer science articles only. Preprocessing experiments contain the testing of preprocessing algorithms which were explained in chapter 4, such as: removing stopwords, word tagging and segmentation. After that, the extracting features experiments were applied to select the summary sentences. Finally, experiments were done to verify and evaluate the extracted features.

6.2 Testing Sample

The articles had been chosen to apply the algorithm. These articles were chosen randomly in the field of computer science. Then these articles were given to the English language professional people, to make sure that they are syntactically correct. Then, every article was summarized to five summaries by five persons who are professional in the language.

As a result, we had fifty summaries by fifty professional people. We did that because the nature of the summary differs from one person to another, so, we did more than one summary for each article. Then we took these summaries and used them in evaluation process.

6.3 Experiments and Results

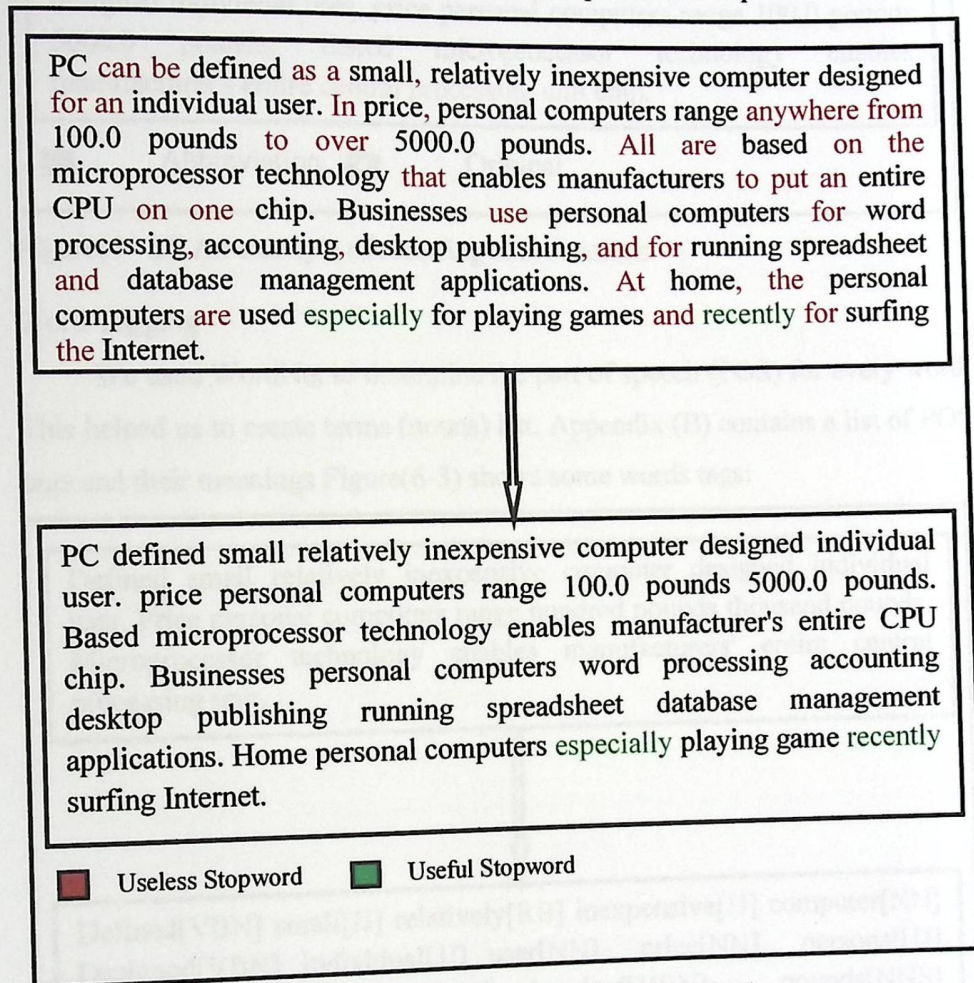
The following sections contain the experiments and results for the project steps: preprocessing, features extraction, summary generation, and post processing.

6.3.1 Preprocessing experiments and results

Preprocessing of the text is a necessary step before extracting any features from it. This section contains the experiments and results of preprocessing stage. These experiments are:

- Removing stopwords

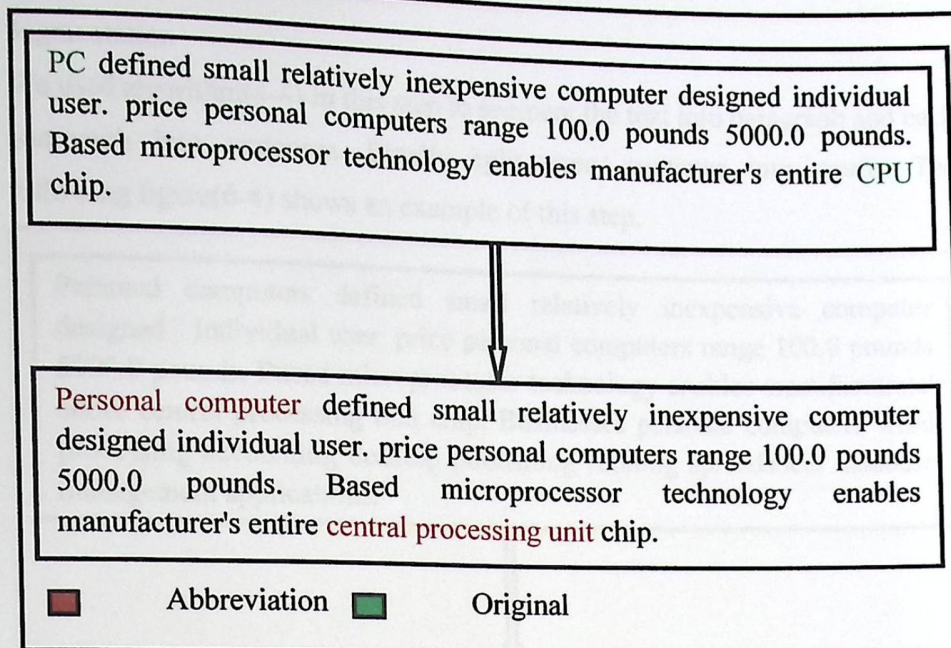
This step achieved its purpose in declining the size of the document by 30%. The following figure(6-1) shows an example of this step:



Figure(6 - 1): An example of removing stopwords from text.

- Referring abbreviations

The purpose of this step was to return every abbreviation to its original term, so as to improve the accuracy of extraction TF-ISF feature. As it is shown in the following figure(6-2):



Figure(6 - 2): An example of referring abbreviations.

- Word Tagging

We used WordNet to determine the part of speech (POS) for every word. This helped us to create terms (nouns) list. Appendix (B) contains a list of POS tags and their meanings Figure(6-3) shows some words tags:

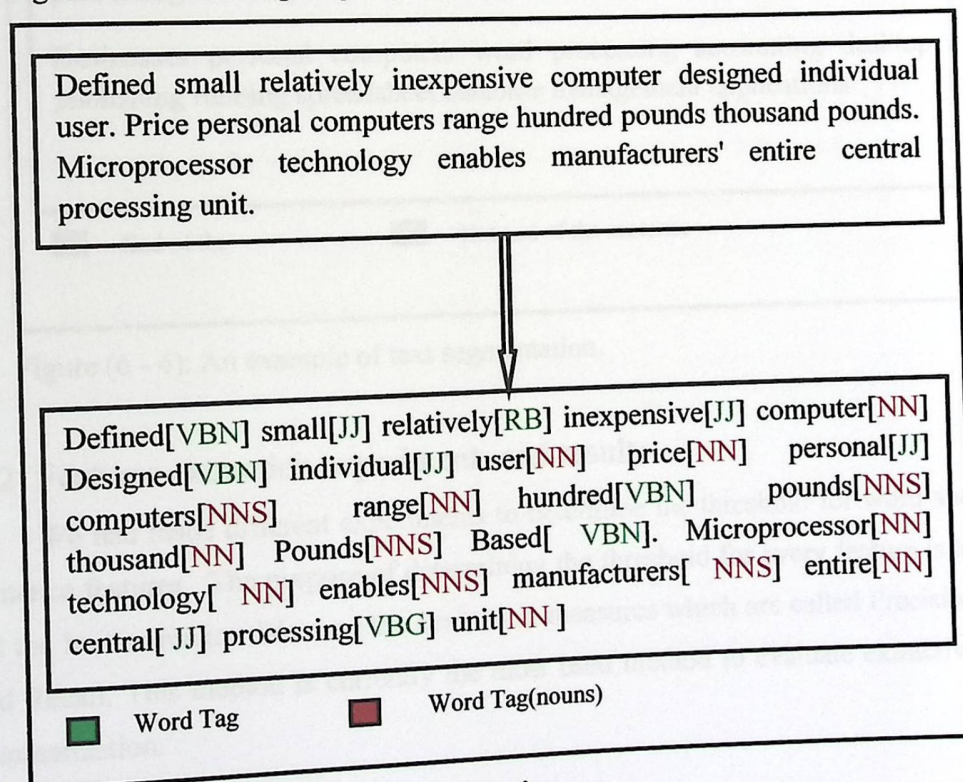


Figure (6 - 3): An example of word tagging.

- Segmentation

We used algorithm(4-4) in this step to segment the text into paragraph and each paragraph into sentences. Finally split every sentence into words. The following figure(6-4) shows an example of this step.

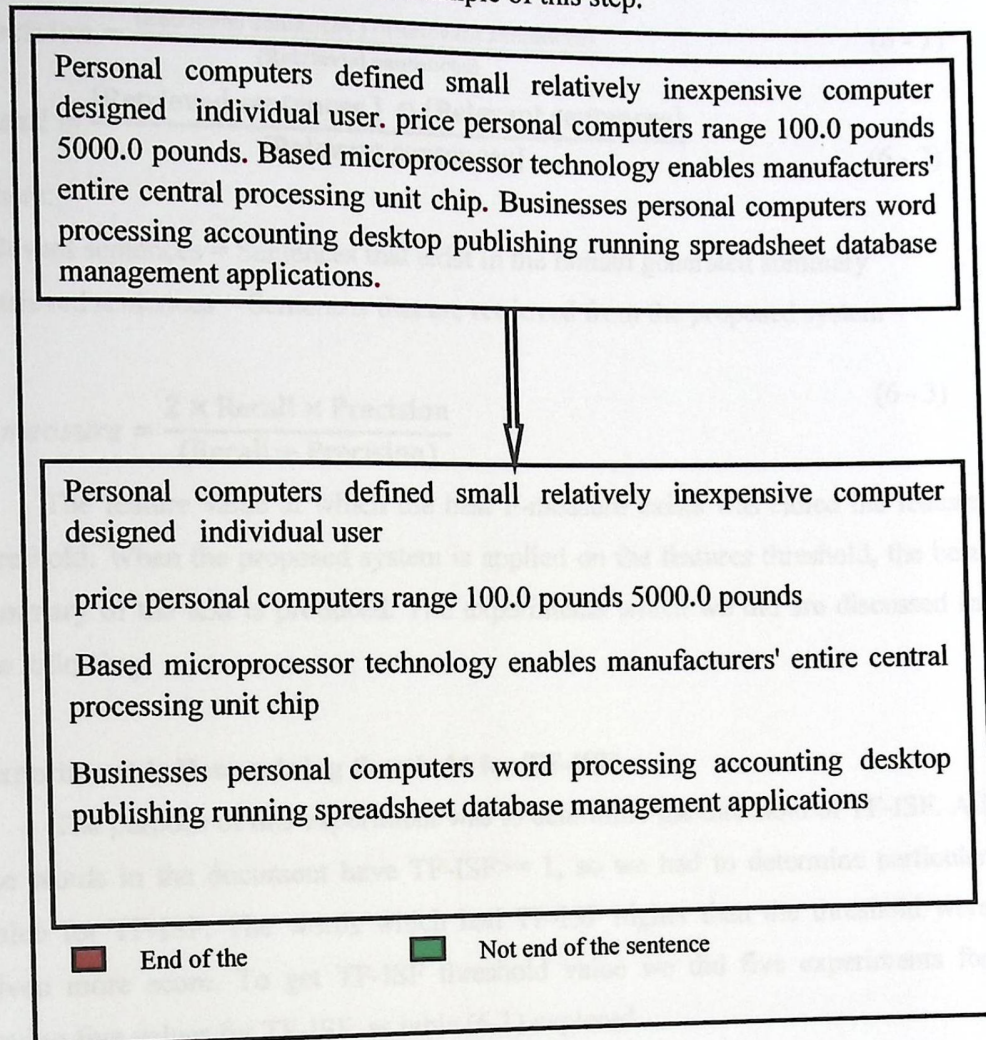


Figure (6 - 4): An example of text segmentation.

6.3.2 Features extraction experiments and results

We had made different experiments to determine the threshold for word and sentence features. The purpose of determining the threshold for every feature is to get the best summary. We used the evaluation measures which are called Precision and Recall. This method is currently the most used method to evaluate extractive summarization.

Precision is used to evaluate the percentage of correctness for the sentences in the summary. Whereas Recall is used to evaluate the percentage of relevant sentences included in summary. F-measure combines both measures (Madhavi,2002).

$$\text{Precision} = \frac{\{\text{Retrieved sentences}\} \cap \{\text{Relevant sentences}\}}{\{\text{Retrieved sentences}\}} \quad (6 - 1)$$

$$\text{Recall} = \frac{\{\text{Retrieved sentences}\} \cap \{\text{Relevant sentences}\}}{\{\text{Relevant sentences}\}} \quad (6 - 2)$$

Where:

Relevant sentences = Sentences that exist in the human generated summary

Retrieved sentences = Sentences that are retrieved from the proposed system

$$F_measure = \frac{2 \times \text{Recall} \times \text{Precision}}{(\text{Recall} + \text{Precision})} \quad (6 - 3)$$

The feature value at which the best F-measure exists was called the feature threshold. When the proposed system is applied on the features threshold, the best summary of the text is produced. The experiments which we did are discussed in the following.

Experiment 1: Determining threshold for TF-ISF

The purpose of this experiment was to determine the threshold of TF-ISF. All the words in the document have $TF-ISF \geq 1$, so we had to determine particular value for TF-ISF. The words which had TF-ISF higher than the threshold were given more score. To get TF-ISF threshold value we did five experiments for testing five values for TF-ISF, as table (6-1) explored.

Exp No	TF_ISF	F-measure										Avg F - measur e
		Doc 1	Doc 2	Doc 3	Doc 4	Doc 5	Doc 6	Doc 7	Doc 8	Doc 9	Doc 10	
1	1	0.41	0.43	0.38	0.45	0.40	0.42	0.41	0.38	0.39	0.41	0.41
2	3	0.50	0.45	0.47	0.48	0.47	0.49	0.48	0.47	0.48	0.47	0.48
3	5	0.55	0.49	0.54	0.55	0.52	0.58	0.54	0.53	0.54	0.55	0.54
4	7	0.50	0.44	0.47	0.45	0.45	0.51	0.48	0.45	0.45	0.44	0.46
5	9	0.47	0.39	0.40	0.44	0.42	0.45	0.42	0.40	0.41	0.38	0.42

Table (6 - 1): The result of determining TF-ISF threshold experiments.

As a result of these experiments, the experiment number 3 fulfilled the highest F-measure = 0.54, where the value of TF-ISF = 5. Comparing with the first experiment which was TF-ISF= 1 and F-measure =0.41, we noticed from this that the TF-ISF value became more less than 5, the F-measure for it was less. Also, we noticed from the experiments four and five that if we increase threshold more than five, F-measure value will become less. As a result of that we knew that the threshold = 5 fulfilled the best results from F-measure value, and this is explored by the figure (6-5).

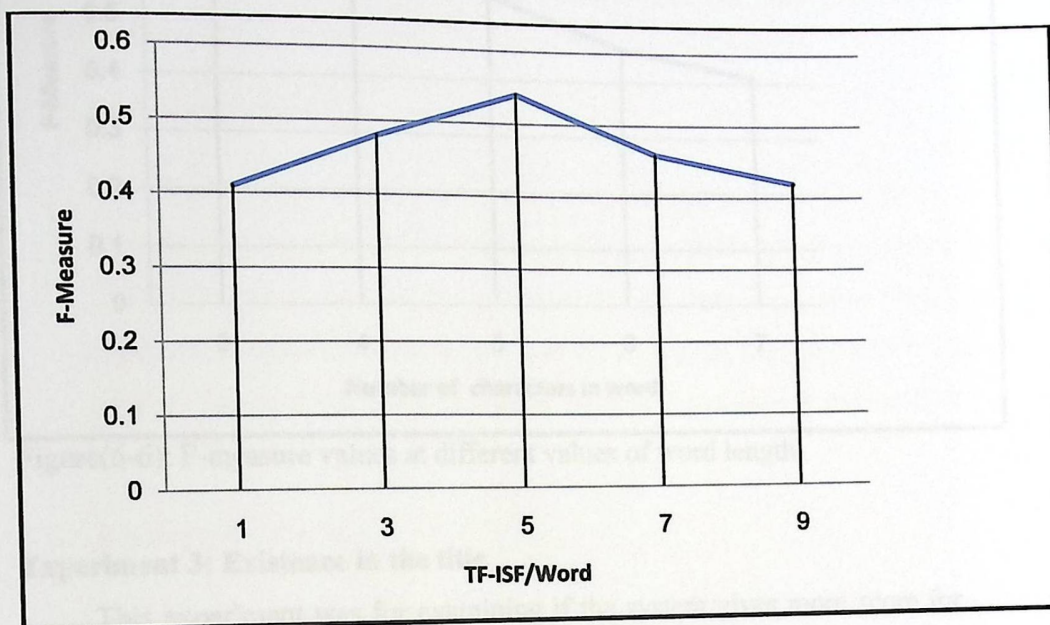


Figure (6 - 5): F-measure values at different values of TF-ISF.

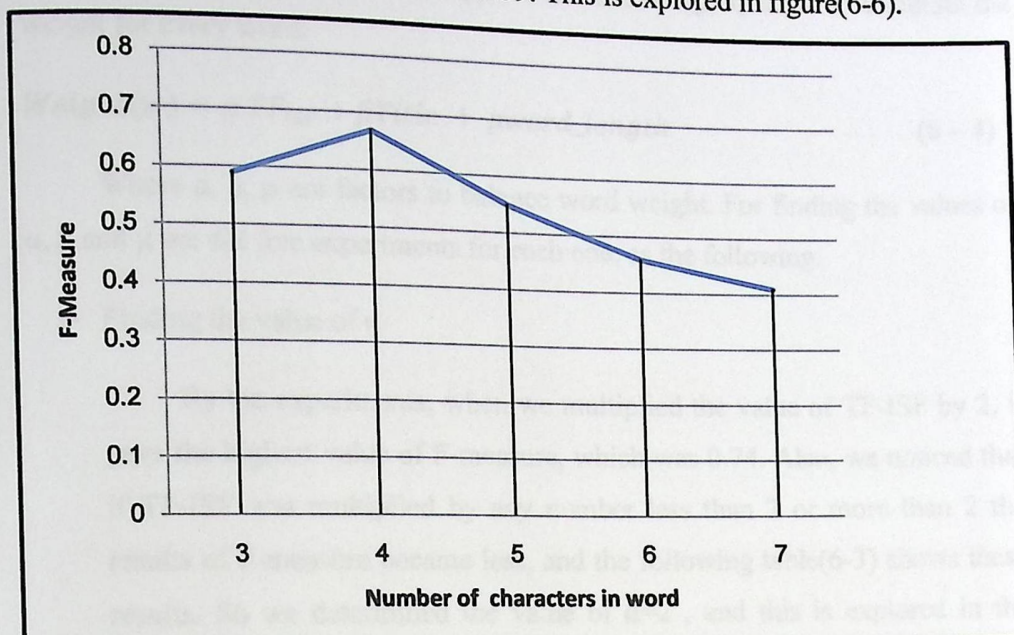
Experiment 2: Determining threshold for word length

The purpose of this experiment was to determine the threshold of word length. The words which had length higher than threshold were given more score. To fulfill that we did five experiments to test five values of word length, as it is explored in table (6-2).

Exp No	word length	F-measure										Avg F-measure
		Doc 1	Doc 2	Doc 3	Doc 4	Doc 5	Doc 6	Doc 7	Doc 8	Doc 9	Doc 10	
1	3	0.57	0.55	0.66	0.59	0.59	0.60	0.59	0.61	0.58	0.59	0.59
2	4	0.64	0.61	0.66	0.70	0.69	0.66	0.64	0.69	0.70	0.66	0.67
3	5	0.51	0.52	0.57	0.60	0.54	0.49	0.54	0.58	0.56	0.57	0.55
4	6	0.49	0.45	0.44	0.48	0.47	0.43	0.48	0.46	0.44	0.49	0.46
5	7	0.45	0.39	0.39	0.44	0.42	0.43	0.42	0.38	0.41	0.38	0.41

Table (6 - 2): The result of determining word length threshold experiments.

As a result of these experiments, the experiment number 2 fulfilled the highest F-measure = 0.67, at the value of length = 4. Whereas the first experiment had word length= 1 and F-measure =0.59. This is explored in figure(6-6).



Figure(6-6): F-measure values at different values of word length.

Experiment 3: Existence in the title

This experiment was for examining if the system gives more score for the words which exist in the title and their synonyms. The figure below shows an example of that.

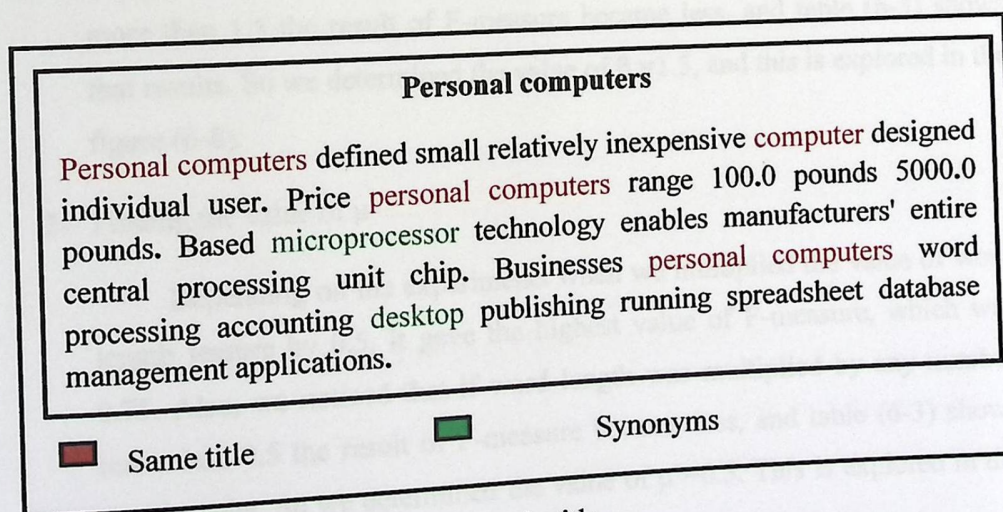


Figure (6-7): An example of existence in title.

Experiment 4: Determining word weight's factors

The purpose of this experiment was to determine the importance of every feature in order to calculate the weight for every word. Because the importances of the features are not equal, we applied the following equation to calculate the weight for every word.

$$Weight(w) = \alpha TF_{ISF} + \beta Title + \mu word_length \quad (6-4)$$

Where α , β , μ are factors to balance word weight. For finding the values of α , β and μ we did five experiments for each one, as the following:

1- Finding the value of α

By the experiments, when we multiplied the value of TF-ISF by 2, it gave the highest value of F-measure, which was 0.74. Also, we noticed that if TF-ISF was multiplied by any number less than 2 or more than 2 the results of F-measure became less, and the following table(6-3) shows these results. So we determined the value of $\alpha=2$, and this is explored in the figure (6-8).

2- Finding the value of β

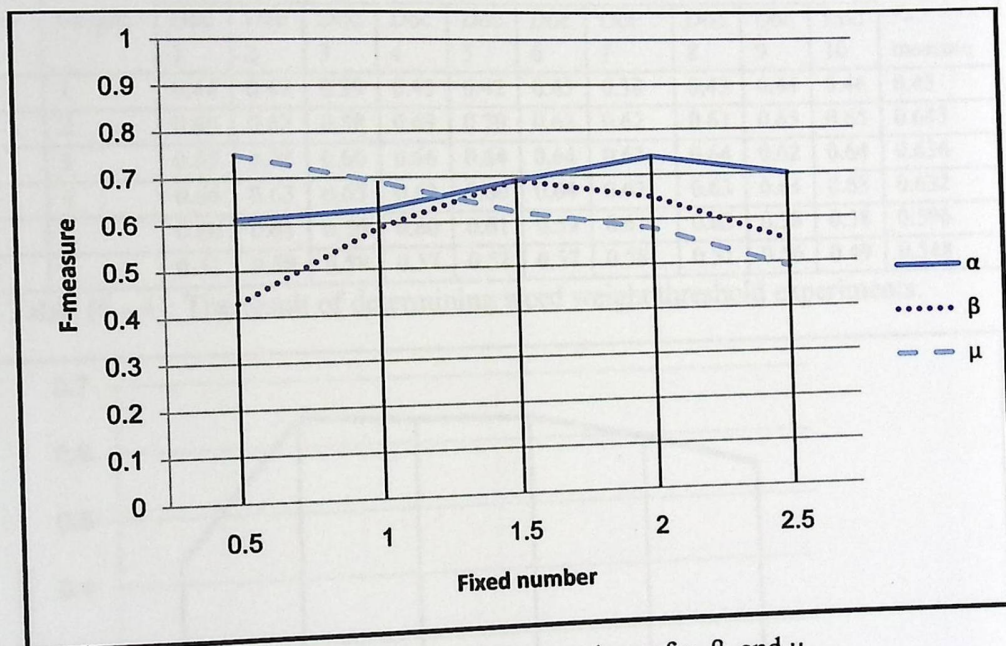
By the experiments, when we multiplied the value of title feature by 1.5, it gave the highest value of F-measure, which was 0.69. Also, we noticed that if title feature was multiplied by any number less than 1.5 or more than 1.5 the result of F-measure became less, and table (6-3) shows that results. So we determined the value of $\beta=1.5$, and this is explored in the figure (6-8).

3- Finding the value of μ

Depending on the experiments when we multiplied the value of word length feature by 0.5, it gave the highest value of F-measure, which was 0.75. Also, we noticed that if word length was multiplied by any number more than 0.5 the result of F-measure became less, and table (6-3) shows these results. So we determined the value of $\mu=0.5$. This is explored in the figure (6-8).

Exp for	Fixed No	F-measure										Avg F-measure
		Doc 1	Doc 2	Doc 3	Doc 4	Doc 5	Doc 6	Doc 7	Doc 8	Doc 9	Doc 10	
α	0.5	0.67	0.61	0.62	0.64	0.60	0.57	0.58	0.61	0.57	0.59	0.61
	1.0	0.64	0.62	0.65	0.60	0.62	0.64	0.63	0.63	0.64	0.62	0.63
	1.5	0.70	0.71	0.65	0.76	0.73	0.65	0.67	0.63	0.76	0.67	0.69
	2.0	0.74	0.72	0.70	0.78	0.74	0.74	0.73	0.73	0.74	0.75	0.74
	2.5	0.70	0.70	0.69	0.75	0.72	0.71	0.69	0.70	0.68	0.69	0.70
β	0.5	0.43	0.45	0.46	0.44	0.46	0.41	0.43	0.40	0.41	0.43	0.43
	1.0	0.57	0.63	0.61	0.59	0.53	0.62	0.61	0.56	0.58	0.63	0.59
	1.5	0.68	0.67	0.64	0.70	0.71	0.68	0.70	0.69	0.73	0.72	0.69
	2.0	0.62	0.68	0.66	0.63	0.66	0.64	0.67	0.66	0.63	0.65	0.65
	2.5	0.59	0.57	0.53	0.54	0.57	0.58	0.53	0.52	0.56	0.53	0.55
μ	0.5	0.73	0.76	0.75	0.74	0.77	0.75	0.75	0.73	0.74	0.76	0.75
	1.0	0.71	0.68	0.70	0.65	0.71	0.68	0.69	0.68	0.67	0.69	0.69
	1.5	0.62	0.64	0.65	0.58	0.63	0.65	0.59	0.59	0.63	0.64	0.62
	2.0	0.59	0.57	0.57	0.61	0.60	0.59	0.57	0.59	0.56	0.58	0.58
	2.5	0.50	0.48	0.51	0.49	0.46	0.51	0.47	0.49	0.49	0.50	0.49

Table (6 - 3): The experiments results of determining word weight factors.

Figure(6 - 8): F-measure values at different values of α , β , and μ .

Experiment 5: Determining threshold for word weight

After calculating every value of α , β , and μ , we calculated the weight for every word depending on the equation (6-1). The purpose of calculating the words weights was to determine the most important keywords which were used as a feature for the sentence. So, we did six experiments and depending on these

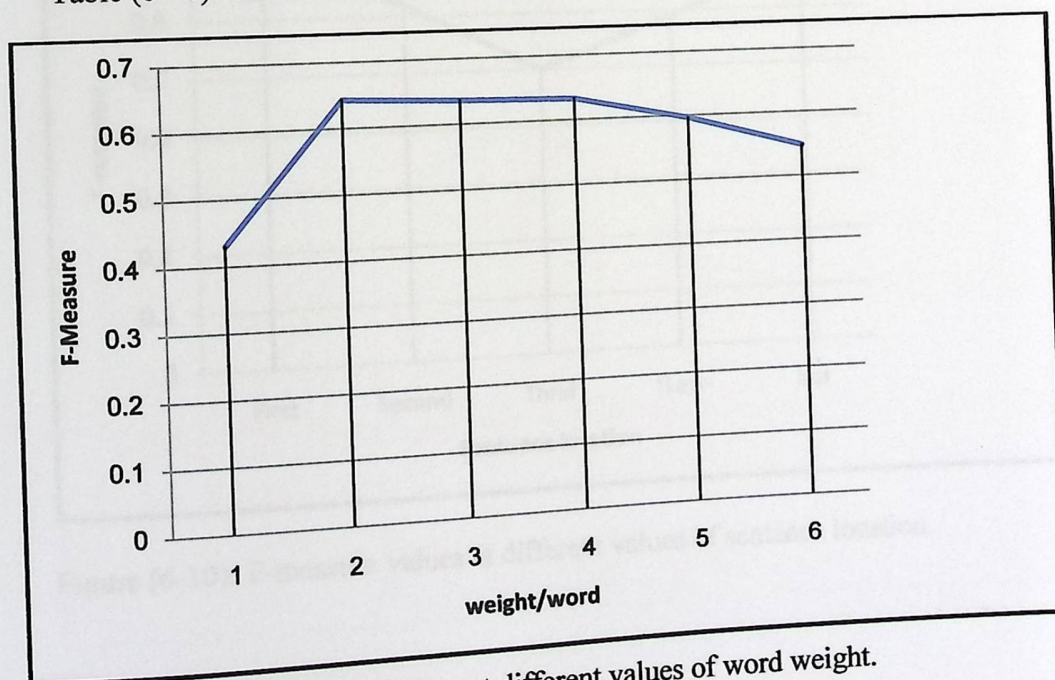
experiments we explored that the value of F-measure was approximated in the experiments 2, 3 and 4. Their values were respectively (0.643, 0.636, 0.632), as shows in figure (6-9). Also there was a simple difference between the results of experiment 5 and 6. Table (6-4) shows these results. After these six experiments, we went to the seventh experiment which calculated the average of words weights deepening on the equation (6-2).

$$\text{Weight_Average} \sum_{w=1}^n \frac{\text{weight}(w)}{n} \quad (6-5)$$

F-measure was calculated at the average for the words weights. F-measure was 0.648 which was the highest F-measure comparing with other experiments. We considered the average of weights as the threshold. So the words which had weights more than this threshold were selected as keywords.

Exp No	Word weight	F-measure										Avg F-measure
		Doc 1	Doc 2	Doc 3	Doc 4	Doc 5	Doc 6	Doc 7	Doc 8	Doc 9	Doc 10	
1	1	0.44	0.47	0.39	0.43	0.42	0.45	0.38	0.43	0.44	0.46	0.43
2	2	0.66	0.62	0.58	0.69	0.70	0.63	0.62	0.63	0.65	0.65	0.643
3	3	0.65	0.64	0.60	0.66	0.64	0.64	0.63	0.64	0.62	0.64	0.636
4	4	0.66	0.63	0.65	0.62	0.60	0.64	0.62	0.63	0.64	0.63	0.632
5	5	0.60	0.61	0.56	0.60	0.61	0.59	0.61	0.62	0.58	0.58	0.596
6	6	0.51	0.59	0.58	0.57	0.52	0.57	0.58	0.51	0.56	0.49	0.548

Table (6 - 4): The result of determining word weight threshold experiments.



Figure(6 - 9): F-measure values at different values of word weight.

Experiment 6: Determining threshold for sentence location

The purpose of this experiment was to determine the threshold of sentence location. The sentences which had location equal the threshold had more score. We tested five values for sentence location during five experiments, as it is explored in the table (6-5).

Exp No	sentence location	F-measure										Avg F-measure
		Doc 1	Doc 2	Doc 3	Doc 4	Doc 5	Doc 6	Doc 7	Doc 8	Doc 9	Doc 10	
1	First	0.73	0.69	0.75	0.69	0.70	0.71	0.75	0.73	0.68	0.69	0.71
2	Second	0.64	0.61	0.58	0.60	0.61	0.60	0.61	0.63	0.64	0.65	0.62
3	third	0.49	0.50	0.48	0.51	0.48	0.52	0.50	0.47	0.49	0.50	0.50
4	Last-1	0.58	0.63	0.55	0.59	0.61	0.59	0.61	0.58	0.59	0.58	0.59
5	Last	0.67	0.69	0.70	0.71	0.66	0.69	0.71	0.67	0.71	0.72	0.69

Table (6-5): The result of determining sentence location threshold experiments.

As a result of these experiments, the experiments number 1 and 5 fulfilled the highest F-measure, where the sentence locations were the first sentence and the last sentence, and their values were 0.71 and 0.69. Comparing with the third experiment which had sentence location= 3 and F-measure =0.50. This is explored in the figure (6-10).

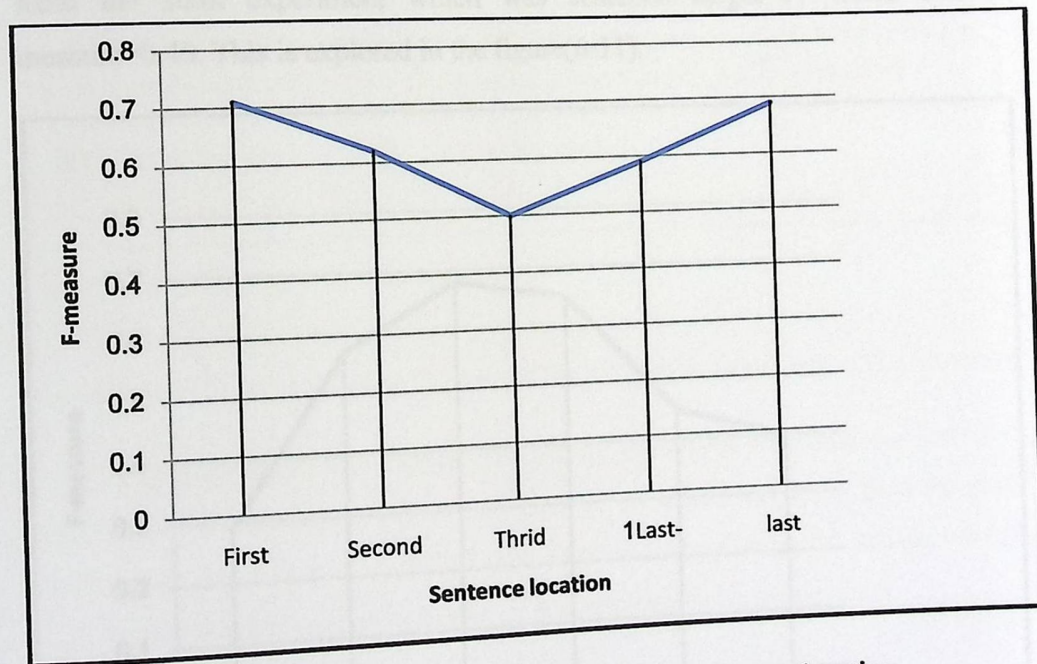


Figure (6-10): F-measure values at different values of sentence location.

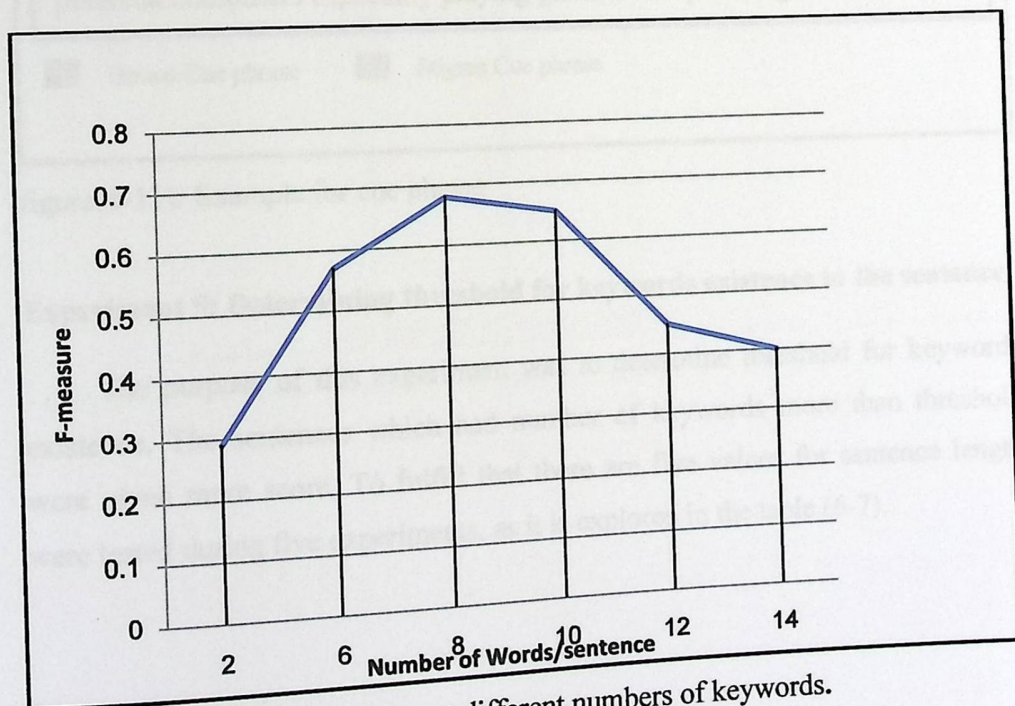
Experiment 7: Determining threshold for sentence length

The purpose of this experiment was to determine the threshold of sentence length feature. The sentences which had length more than the threshold had more score. To fulfill that we tested six values for sentence length during six experiments, as it is explored in the table (6-6).

Exp No	sentence length	F-measure										Avg F-measure
		Doc 1	Doc 2	Doc 3	Doc 4	Doc 5	Doc 6	Doc 7	Doc 8	Doc 9	Doc 10	
1	4	0.30	0.29	0.30	0.29	0.29	0.31	0.29	0.26	0.29	0.32	0.29
2	6	0.59	0.58	0.55	0.57	0.58	0.57	0.59	0.57	0.56	0.55	0.57
3	8	0.64	0.70	0.68	0.66	0.72	0.70	0.71	0.69	0.67	0.66	0.68
4	10	0.65	0.63	0.63	0.67	0.65	0.67	0.65	0.64	0.68	0.64	0.65
5	12	0.43	0.47	0.46	0.43	0.45	0.41	0.47	0.42	0.48	0.45	0.45
6	14	0.39	0.41	0.38	0.42	0.39	0.41	0.42	0.41	0.40	0.41	0.40

Table (6 - 6): The result of determining sentence length threshold experiments.

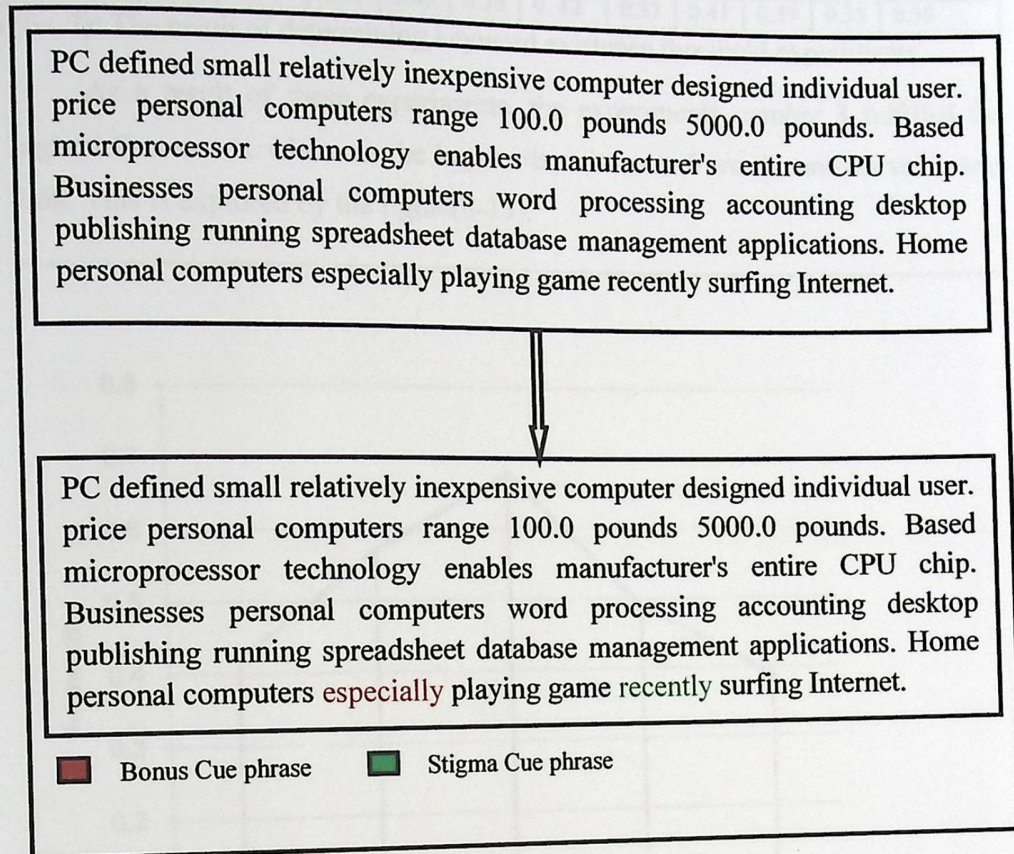
As a result of these experiments, the experiment number 3 fulfilled the highest F-measure = 0.68, at the sentence length=8 words. Whereas the first experiment which was sentence length= 4 and F-measure =0.29. On the other hand the sixth experiment which was sentence length=14 words and F-measure=0.40. This is explored in the figure(6-11).



Figure(6-11): F-measure values at different numbers of keywords.

Experiment 8: Cue phrase existence

In this experiment, we selected the sentence which contain bonus cue phrase in the sentence improved its score by 1. On the other hand, the existence of stigma cue phrase declined the sentence score by 1. Figure (6-12) shows an example of existing cue phrase types.



figure(6-11): Example for cue phrase

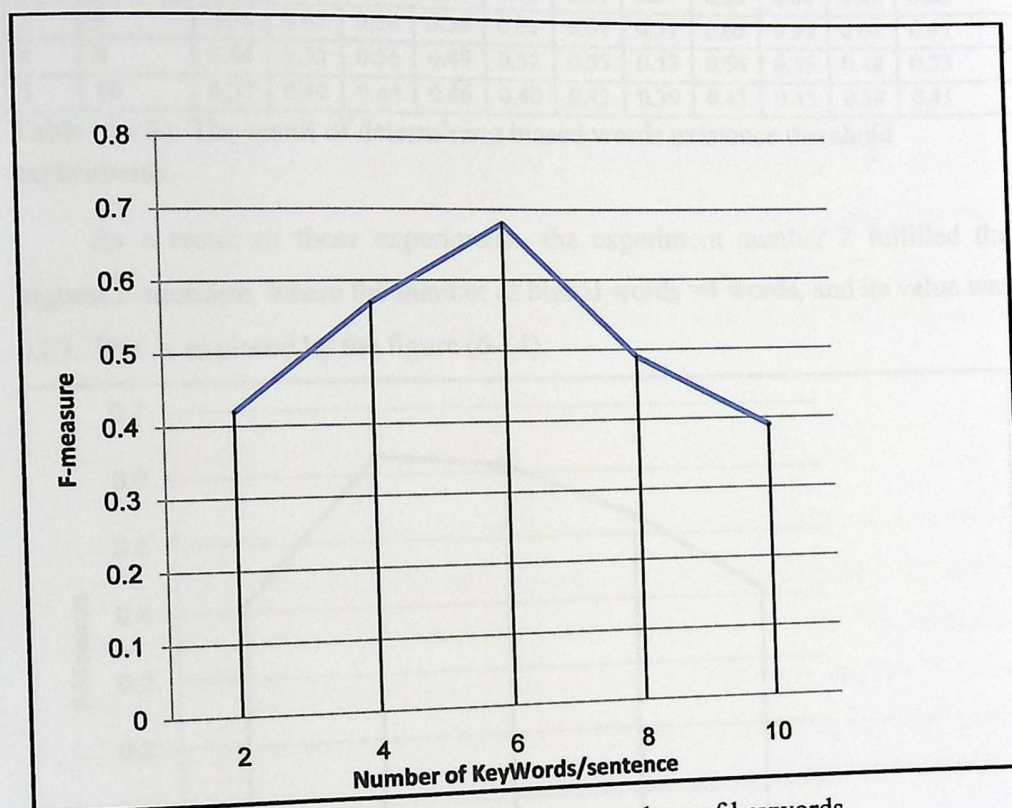
Experiment 9: Determining threshold for keywords existence in the sentence

The purpose of this experiment was to determine threshold for keywords existence. The sentences which had number of keywords more than threshold were given more score. To fulfill that there are five values for sentence length were tested during five experiments, as it is explored in the table (6-7).

Exp No	No keyword	F-measure										Avg F-measure
		Doc 1	Doc 2	Doc 3	Doc 4	Doc 5	Doc 6	Doc 7	Doc 8	Doc 9	Doc 10	
1	2	0.45	0.41	0.39	0.39	0.44	0.43	0.42	0.41	0.43	0.41	0.42
2	4	0.58	0.59	0.54	0.53	0.58	0.56	0.59	0.57	0.56	0.57	0.57
3	6	0.66	0.70	0.68	0.64	0.67	0.71	0.70	0.66	0.72	0.69	0.68
4	8	0.47	0.46	0.48	0.49	0.50	0.49	0.49	0.51	0.50	0.48	0.49
5	10	0.35	0.40	0.39	0.43	0.38	0.42	0.37	0.41	0.39	0.38	0.39

Table (6 - 7): The result of determining keyword existence threshold experiments.

As a result of these experiments, the experiments number 3 fulfilled the highest F-measure, where as the keywords existence=6 words, and its value was 0.68. This is explored by the figure(6-13).



Figure(6 - 13): F-measure values at different numbers of keywords.

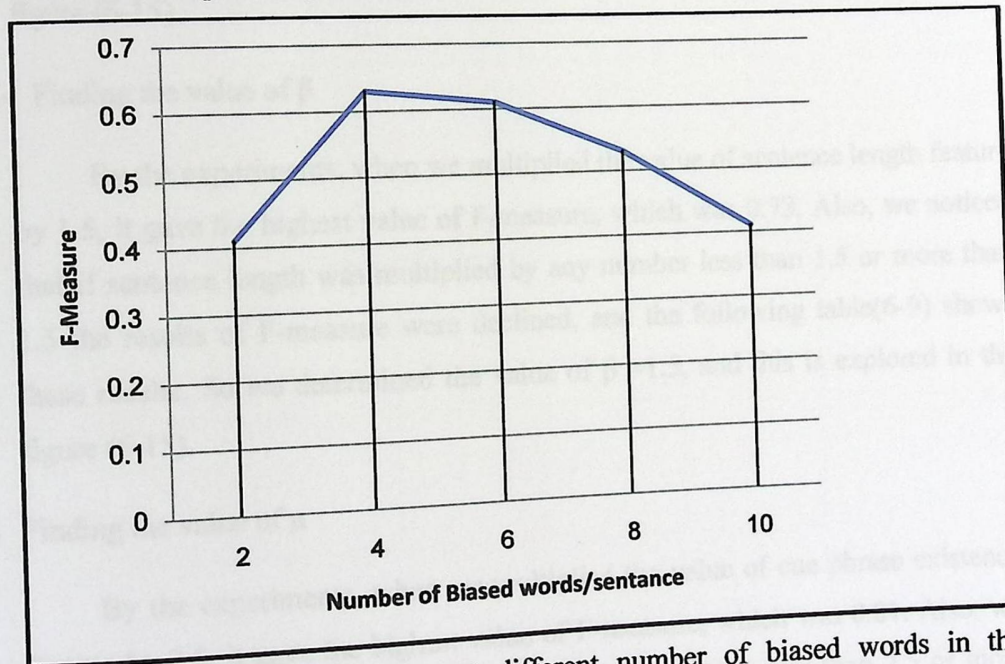
Experiment 10: Determining threshold for biased words existence in the sentence

The purpose of this experiment is to determine the threshold for number of biased words existence. The sentences which had number of biased words greater than the threshold had given more score. We determined this threshold during five experiments, as it is explored in the table (6-8).

Exp NO	number biased	F-measure										Avg F-measure
		Doc 1	Doc 2	Doc 3	Doc 4	Doc 5	Doc 6	Doc 7	Doc 8	Doc 9	Doc 10	
1	2	0.44	0.41	0.40	0.41	0.44	0.42	0.43	0.39	0.43	0.41	0.42
2	4	0.59	0.66	0.63	0.64	0.65	0.64	0.67	0.62	0.60	0.62	0.63
3	6	0.58	0.62	0.58	0.59	0.61	0.64	0.59	0.65	0.59	0.61	0.61
4	8	0.54	0.53	0.56	0.49	0.52	0.55	0.53	0.54	0.55	0.48	0.53
5	10	0.37	0.40	0.44	0.46	0.40	0.42	0.39	0.41	0.45	0.38	0.41

Table (6 - 8): The result of determining biased words existence threshold experiments.

As a result of these experiments, the experiment number 2 fulfilled the highest F-measure, where the number of biased words =4 words, and its value was 0.63. This is explored by the figure (6-14).



Figure(6-14) F-measure values at different number of biased words in the sentence.

Experiment 11: Determining sentence weight's factors

The purpose of these experiments was to determine the importance of every sentence feature in order to calculate the weight of every sentence. Because the importances of the features are not equal, we applied the following equation to calculate the weight of every sentence.

$$Weight(s) = \alpha location(s) + \beta length(s) + \mu cuephrase_{existence}(s) + \Omega keywords_{existence} + \varphi biased_{existence} \quad (6 - 6)$$

For finding the values of α , β , μ , Ω and φ we did five experiments for each one, as the following:

1- Finding the value of α

By the experiments, when we multiplied the value of sentence location feature by 2, it gave the highest value of F-measure, which was 0.79. Also, we noticed that if sentence location was multiplied by any number less than 2 or more than 2 the results of F-measure were declined, and the following table(6-9) shows these results. So we determined the value of $\alpha=2$, and this is explored in the figure (6-15).

2- Finding the value of β

By the experiments, when we multiplied the value of sentence length feature by 1.5, it gave the highest value of F-measure, which was 0.73. Also, we noticed that if sentence length was multiplied by any number less than 1.5 or more than 1.5 the results of F-measure were declined, and the following table(6-9) shows these results. So we determined the value of $\beta =1.5$, and this is explored in the figure (6-15).

3- Finding the value of μ

By the experiments, when we multiplied the value of cue phrase existence feature by 2.5, it gave the highest value of F-measure, which was 0.81. Also, we noticed that if this feature was multiplied by any number less than 2.5 or more than 2.5 the results of F-measure were declined, and the following table(6-9)

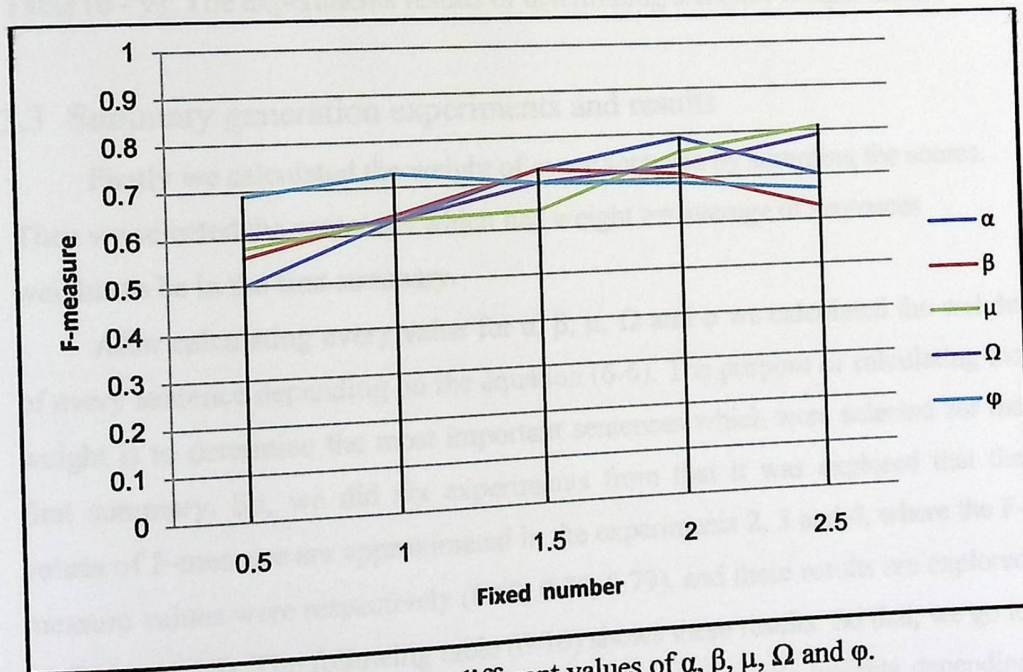
shows these results. So we determined the value of $\mu = 2.5$, and this is explored in the figure (6-15).

4- Finding the value of Ω

Depending on our experiments, when we multiplied the value of keywords existence feature by 2.5, it gave the highest value of F-measure, which was 0.79. Also, we noticed that if this feature was multiplied by any number less than 2.5 or more than 2.5 the results of F-measure were declined, and the following table(6-9) shows these results. So we determined the value of $\Omega = 2.5$, and this is explored in the figure (6-15).

5- Finding the value of ϕ

Depending on our experiments, when we multiplied the value of biased words existence feature by 1, it gave the highest value of F-measure, which was 0.73. Also, we noticed that if this feature was multiplied by any number less than 1 or more than 1 the results of F-measure were declined, and the following table(6-9) shows these results. So we determined the value of $\phi = 1$, and this is explored in the figure (6-15).



Figure(6-15) F-measure values at different values of α , β , μ , Ω and ϕ .

Exp for	Fixed number	F-measure										Avg F-measure
		Doc 1	Doc 2	Doc 3	Doc 4	Doc 5	Doc 6	Doc 7	Doc 8	Doc 9	Doc 10	
α	0.5	0.52	0.49	0.51	0.47	0.48	0.52	0.46	0.47	0.53	0.54	0.50
	1.0	0.67	0.61	0.62	0.64	0.63	0.66	0.58	0.63	0.60	0.62	0.63
	1.5	0.74	0.71	0.76	0.72	0.69	0.74	0.76	0.72	0.73	0.74	0.73
	2.0	0.80	0.76	0.74	0.78	0.77	0.79	0.83	0.81	0.78	0.79	0.79
	2.5	0.68	0.69	0.69	0.72	0.70	0.69	0.68	0.71	0.70	0.67	0.70
β	0.5	0.55	0.57	0.56	0.55	0.57	0.54	0.56	0.55	0.58	0.57	0.56
	1.0	0.67	0.61	0.65	0.66	0.65	0.62	0.62	0.67	0.63	0.66	0.64
	1.5	0.74	0.73	0.69	0.75	0.73	0.74	0.71	0.69	0.73	0.75	0.73
	2.0	0.73	0.74	0.68	0.69	0.68	0.70	0.71	0.69	0.72	0.75	0.71
	2.5	0.63	0.66	0.63	0.62	0.63	0.64	0.63	0.59	0.66	0.63	0.63
μ	0.5	0.55	0.57	0.56	0.60	0.61	0.58	0.57	0.62	0.60	0.58	0.58
	1.0	0.65	0.64	0.61	0.66	0.65	0.62	0.61	0.60	0.60	0.61	0.63
	1.5	0.66	0.63	0.64	0.67	0.63	0.61	0.66	0.62	0.62	0.66	0.64
	2.0	0.78	0.78	0.76	0.77	0.75	0.75	0.76	0.75	0.76	0.75	0.76
	2.5	0.82	0.80	0.82	0.82	0.81	0.81	0.82	0.80	0.83	0.81	0.81
Ω	0.5	0.62	0.60	0.62	0.63	0.60	0.61	0.63	0.62	0.61	0.59	0.61
	1	0.64	0.63	0.63	0.65	0.64	0.63	0.63	0.65	0.60	0.63	0.63
	1.5	0.68	0.72	0.70	0.70	0.71	0.72	0.73	0.69	0.69	0.68	0.70
	2	0.74	0.71	0.75	0.73	0.73	0.72	0.72	0.72	0.75	0.72	0.73
	2.5	0.82	0.80	0.77	0.79	0.78	0.79	0.80	0.79	0.82	0.77	0.79
ϕ	0.5	0.67	0.70	0.71	0.69	0.69	0.68	0.70	0.69	0.69	0.70	0.69
	1	0.74	0.75	0.70	0.71	0.71	0.74	0.73	0.74	0.73	0.72	0.73
	1.5	0.71	0.69	0.69	0.71	0.70	0.70	0.68	0.67	0.71	0.71	0.70
	2	0.68	0.69	0.69	0.70	0.66	0.69	0.70	0.72	0.68	0.70	0.69
	2.5	0.65	0.64	0.67	0.67	0.66	0.66	0.68	0.67	0.68	0.69	0.67

Table (6 - 9): The experiments results of determining sentence weight factors

6.3.3 Summary generation experiments and results

Firstly we calculated the weight of every sentence by summing the scores. Then we selected the sentences which had weight $\geq$ average of sentences weights to be in the first summary.

After calculating every value for α , β , μ , Ω and ϕ we calculated the weight of every sentence depending on the equation (6-6). The purpose of calculating the weight is to determine the most important sentences which were selected for the first summary. So, we did six experiments from that it was explored that the values of F-measure are approximated in the experiments 2, 3 and 4, where the F-measure values were respectively (0.60, 0.78, 0.79), and these results are explored by figure (6-16). The following table (6-10) shows these results. So that, we go to a seventh experiment which calculated the average of words weights depending on the equation (6-7).

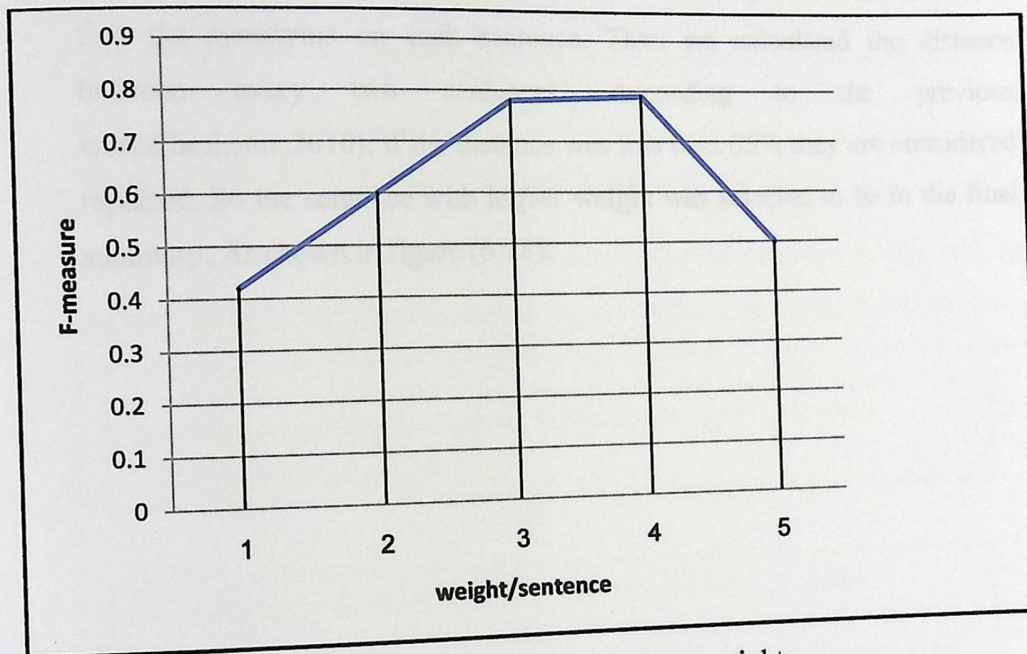
$$Weight_{Average} = \frac{\sum_{s=1}^n weight(s)}{n}$$

(6 - 7)

Then, F-measure was calculated at the average for the words weights. Its value was 0.82 so it fulfilled the highest F-measure comparing with other experiments. The weight average was considered as a threshold. So the sentences which had weight more than this threshold were selected as first summary.

Exp No	Weight sentence	F-measure										Avg F-measure
		Doc 1	Doc 2	Doc 3	Doc 4	Doc 5	Doc 6	Doc 7	Doc 8	Doc 9	Doc 10	
1	1	0.40	0.41	0.41	0.44	0.43	0.42	0.43	0.39	0.43	0.41	0.42
2	2	0.62	0.61	0.60	0.58	0.60	0.58	0.59	0.62	0.60	0.62	0.60
3	3	0.78	0.75	0.78	0.76	0.77	0.78	0.77	0.80	0.78	0.79	0.78
4	4	0.79	0.80	0.79	0.79	0.78	0.81	0.80	0.79	0.80	0.78	0.79
5	5	0.53	0.49	0.49	0.52	0.50	0.51	0.47	0.48	0.52	0.50	0.50

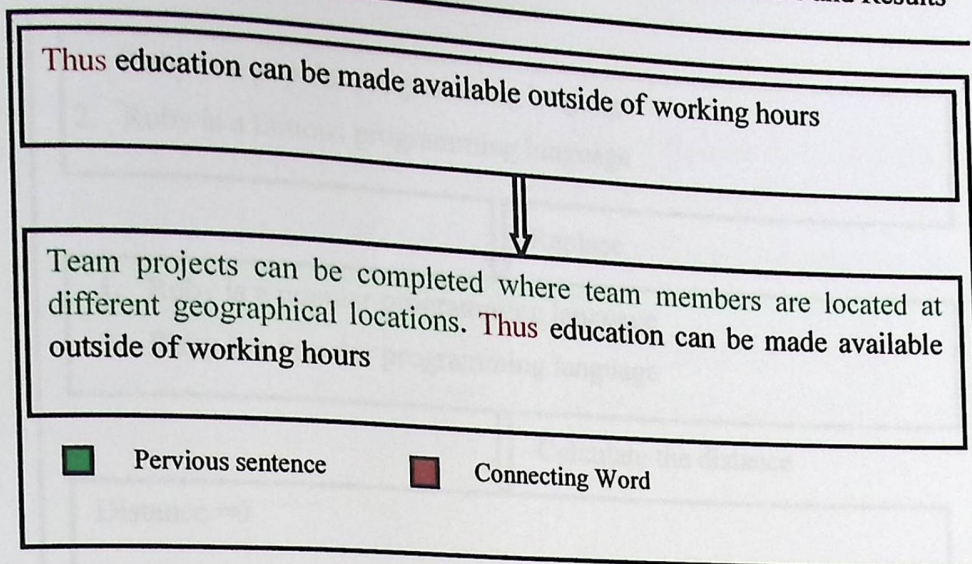
Table (6 - 10): The result of determining sentence weight threshold experiments.



Figure(6 - 16): F-measure values at different sentence weights.

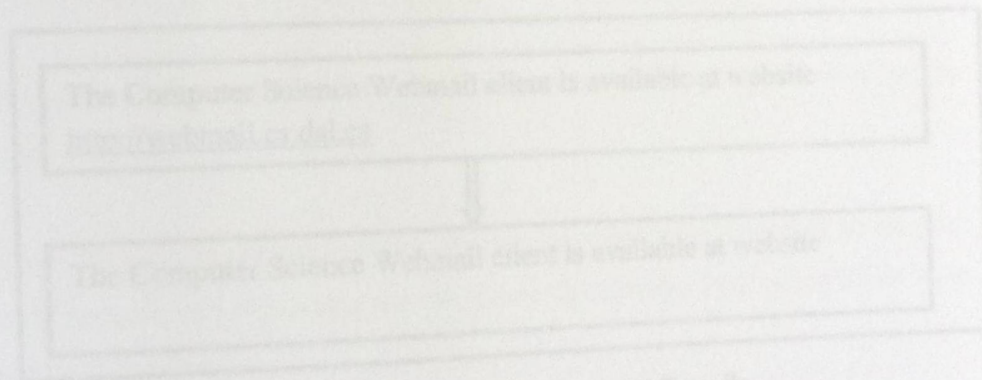
6.3.4 Post-Processing experiments and results

1. Ordering the sentences: According to the sentences locations in the source, we ordered the final summary.
2. Connecting words: This step achieved its purpose in providing a coherence between the sentences, as the following figure(6-17) shows:

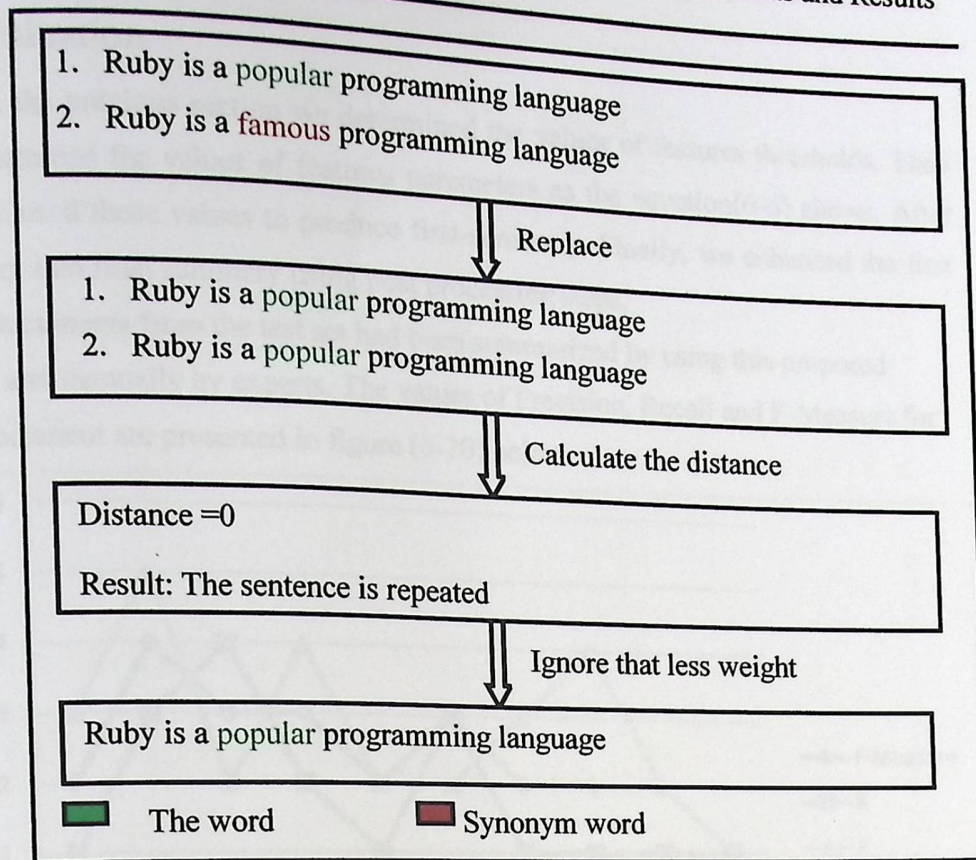


Figure(6 - 17): An example of connecting words step.

3. Sentence to sentence similarity: We used the dictionary of Thesaurus.com to find the synonyms on each sentence. Then we calculated the distance between every two sentences. According to the previous works(hasheme,2010), if the distance was less than 65% they are considered repeated. So the sentence with higher weight was selected to be in the final summary. As shown in figure (6-18):

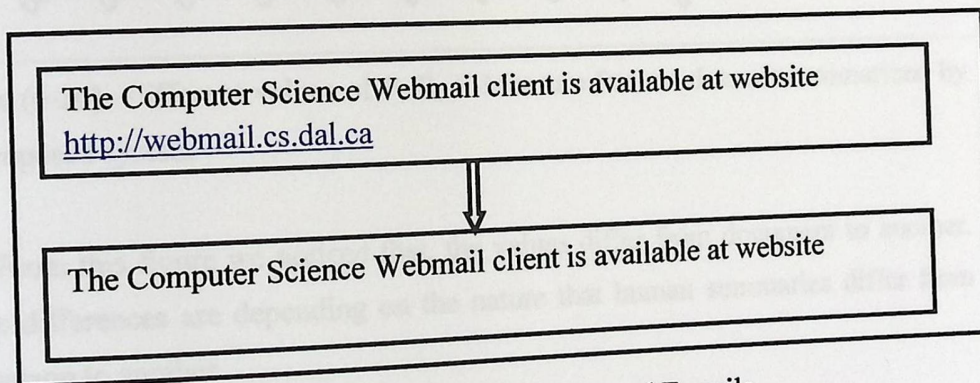


Figure(6-18): an example of removing URLs and e-mails.



Figure(6-18): An example of sentence to sentence similarity.

4. Removing URLs and e-mails: We used regular expressions in this step to select the URLs and e-mails then we removed them, as figure (6-19) shows.



Figure(6-19): an example of removing URLs and E-mails.

6.4 Evaluation

In the previous section we determined the values of features thresholds. Then we determined the values of features parameters as the equation(6-6) shows. After that, we used these values to produce first summary. Finally, we enhanced the first summary into final summary using post processing steps.

Documents from the test set had been summarized by using this proposed system and manually by experts. The values of Precision, Recall and F-Measure for each document are presented in figure (6-20) below:

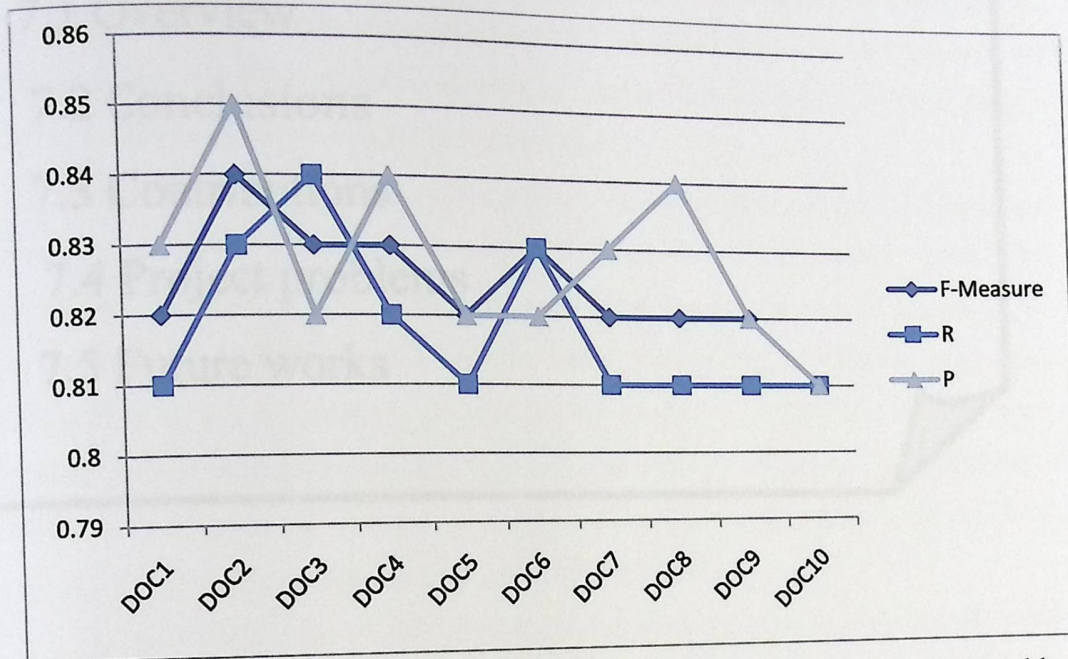


Figure (6-20): Different values of P, R, F-Measure for test data set summarized by the proposed system

From this figure we noticed that, the values differ from document to another. These differences are depending on the nature that human summaries differ from one person to another.

6.5 Summary

In this chapter we showed our experiments and results. We described the preprocessing steps by examples. Then we determined the values of features thresholds and factors on which we get the best summary. Finally, we evaluated the proposed system and found the values of P, R, and F-measure for test set which were summarized by our proposed system.

Chapter Seven

Conclusions and Future Works

Contents:

7.1 Overview

7.2 Conclusions

7.3 Contributions

7.4 Project problems

7.5 Future works

7.1 Overview

In this chapter we will present the achievements of our project. Then we will mention the main problems that we faced in the project. Finally, we will propose suggestions for the future works to improve this project.

7.2 Conclusions

In this project, new algorithms were developed to summarize domain specific text from single document, using linguistic and statistical methods. The development of these algorithms for automatic text summarization allowed us to contribute the area of natural language processing (NLP) efficiently.

As a result of our project, we conclude that the combinational statistical and linguistic method is better than each method alone. Results showed that 82% of the summaries produced by our approach were approximately the most similar to the human summaries. These results became better than results of methods used either statistical or linguistic methods or both, as we studied until 1/3/2011. For example, (Hashmi, 2010) used statistical approach and reached 70%. Whereas (ling, 2009) used clustering in linguistic approach and reached 72%. On the other hand, (Shams, 2010) used statistical and linguistic approach and reached 68%.

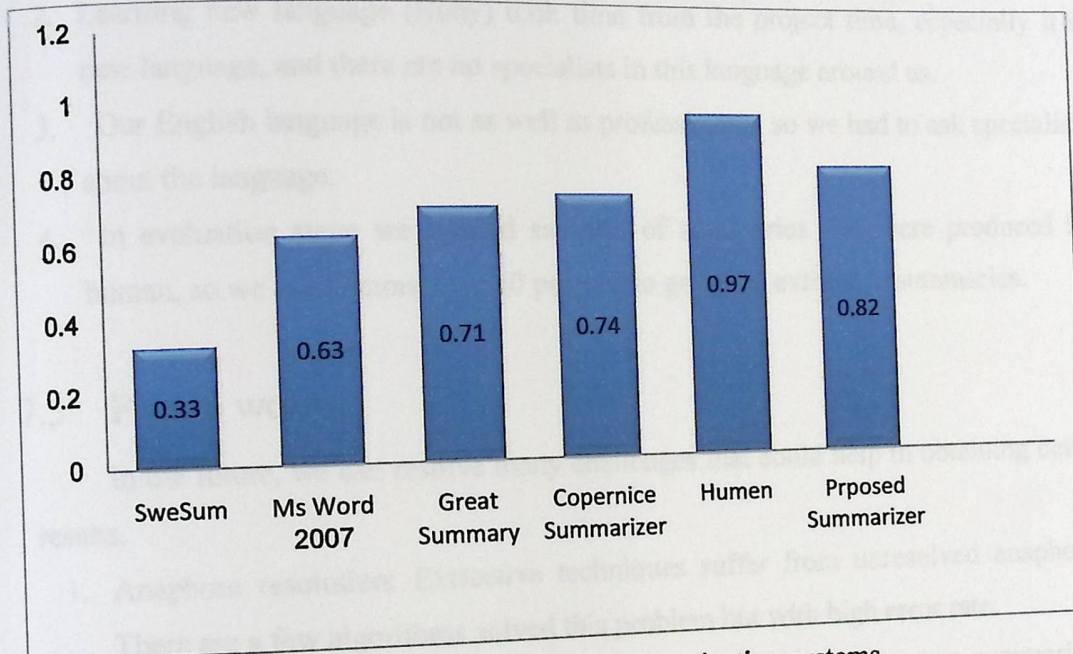
The table below shows the comparison between our proposed system and some other systems.

Methods	Precision	Recall	F-measure
Ms Word 2007	0.66	0.72	0.69
Copernice summarizer	0.73	0.75	0.74
SweSum	0.30	0.36	0.33
Great Summry	0.69	0.74	0.71
Our method	0.80	0.84	0.82
Human summary	0.98	0.96	0.97

Table (7 – 1): Comparisons of Precision (P), Recall (R), F-measure Using Different Methods.

Microsoft word 2007 summarizer reached an average precision 0.66, average recall of 0.72, and F-measure of 0.69. MS Word only simply chose some sentences defects in content expression. Copernic Summarizer reached an average precision of 0.73, recall of 0.75, and F-measure of 0.74 ; while SweSum summarizer reached an average precision of 0.30, recall of 0.36, and F-measure of 0.33. GreatSummary reached an average precision of 0.69, recall of 0.74, and F-measure of 0.71.

Our proposed system reached an average precision of 0.80, recall of 0.84, and F-measure of 0.82. Finally we evaluate human summaries to confirm that there are no 100% evaluation summary, this reached an average precision of 0.98, recall of 0.96, and F-measure of 0.97. The figure below shows these results.



Figure(7 - 1): Comparison between different summarization systems.

7.3 Contributions

Here are the main contributions of our project:

1. Identification of general stages for an automatic extractive text summarization method using combinational between statistical and linguistic approaches.
2. Development of the method for single-document summarization.
3. Proposing new approach to deal with summary generation task for English language in computer science specific field.

4. Reducing redundancy so the final summary became reduced.
5. For this project we wrote a paper called 'Automatic text summarization using combinational statistical and linguistic methods', and we aim to Publish it in the recent future (See Appendix D).

7.4 Project problems

We have faced many problems during the simulation this project. Most of these problems have been solved and others have been covered here.

1. The main problem of this project was there is no deeper knowledge for the group in data and text mining, so we had to read large number of papers and books about these topics to understand them, then to work in that field.
2. Learning new language (Ruby) took time from the project time, especially it is new language, and there are no specialists in this language around us.
3. Our English language is not as well as professionals, so we had to ask specialists about the language.
4. In evaluation stage we needed samples of summaries that were produced by human, so we asked more than 50 persons to get good extractive summaries.

7.5 Future works

In the future, we can resolve many challenges that could help in obtaining better results.

1. Anaphora resolution: Extractive techniques suffer from unresolved anaphora. There are a few algorithms solved this problem but with high error rate.
2. Extension to multi document summarization: Our proposed system summarizes single document summarization and multi document is still a challenging extension of the current work.
3. Extension to generic domain summarization: Our proposed system summarizes documents in computer science field, and need to be extended to generic domain.

Appendixes

Appendix A

Implementation Code

Appendix A

Implementation Code

In this appendix we present our project simulation code.

```
#####
#### require #####
#####
require 'rubygems'
require 'mysql'
require 'WordNet'
require 'part_of_speech'
require 'metanym'
require 'text'
#####
#### Connection & Create Table #####
#####
##### Connection database #####
db = Mysql.new('localhost','root','123','project')
##### create table Paragraph #####
db.query("CREATE TABLE IF NOT EXISTS Paragraph ( IDP INT , PRIMARY KEY (IDP), Length INT )")
# ##### create table Sentence #####
db.query("CREATE TABLE IF NOT EXISTS Sentences ( IDS INT , IDP INT,PRIMARY KEY (IDS,IDP),
Location INT, Length INT , Cue_phras INT, Keyword INT, Biased INT, Weight INT, CONSTRAINT
Sentences_ibfk_1 FOREIGN KEY (IDP) REFERENCES Paragraph(IDP) ON UPDATE CASCADE ON
DELETE CASCADE ) ")
##### create table Words #####
db.query("CREATE TABLE Words ( IDW INT, IDS INT ,IDP INT, PRIMARY KEY
(IDW,IDS,IDP),POS VARCHAR(30), TF INT , TF_ISF INT,Exi_Title INT DEFAULT 0
, Length INT DEFAULT 0,keyword INT DEFAULT 0, Weight DECIMAL(10,2)DEFAULT 0.00
,CONSTRAINT words_ibfk_1 FOREIGN KEY(IDP) REFERENCES Paragraph(IDP) ON UPDATE
CASCADE ON DELETE CASCADE, CONSTRAINT words_ibfk_2 FOREIGN KEY(IDS) REFERENCES
Sentences(IDS) ON UPDATE CASCADE ON DELETE CASCADE ) ")
##### Stopword #####
db.query("CREATE TABLE IF NOT EXISTS stopword ( IDSW INT , PRIMARY KEY (IDSW), SW
Vaechar(18),Type Vaechar(18)) ")
##### cue phrase #####
db.query("CREATE TABLE IF NOT EXISTS cue phrase( IDCP INT , PRIMARY KEY (IDCP), Ceu
phrase Vaechar(18),Type Vaechar(10)) ")
##### Connecting word #####
db.query("CREATE TABLE IF NOT EXISTS connecting word( IDCO INT , PRIMARY KEY (IDCO),
Connword Vaechar(15)) ")
#####
#### Preprocessing #####
#####
##### Removing Stopwords #####
mystopword= mystopword1.split(" ")
ar1=db.query("select * from paragraph")
while row=ar1.fetch_row do
  u3[g]=row[1]
  g=g+1
end
```



```

for dd in 1..u3.size-1
  u4= u3[dd].split(" ")
  for i in 0..mystopword.size-1
    for j in 0..u4.size-1
      if(mystopword[i].include?(u4[j]))==true
        u4[j]=" "
      end
    end
  end
  puts u5[dd]= u4
  puts u5.join(" ")
  db.query ("UPDATE paragraph SET par='#{u5}' WHERE idp=#{dd}#")
end
##### Referring abbreviations #####
abb=fa.read
faa= abb1=faa.read
ar1=db.query("select * from paragraph")
while row=ar1.fetch_row do
  ua[ga]=row[1]
  ga=ga+1
end
for ddd in 1..ua.size-1
  ua4= ua[ddd].split(" ")
  for i1 in 0..abb.size-1
    for j1 in 0..ua4.size-1
      if(abb[i1].include?(ua4[j1]))==true
        ua4[j1]=abb1[i1]
      end
    end
  end
  ua5= ua4.join(" ")
  db.query ("UPDATE paragraph SET par='#{ua5}' WHERE idp=#{ddd}#")
end
##### Part of speech #####
PartOfSpeech.new(lexicon_path)
ar=db.query("select * from words")
po=0
while row=ar.fetch_row do
  wpos[po]= row[0]
  spos[po]=row[1]
  ppos[po]=row[2]
  u111[po]=row[3]
  po=po+1
end
for iii in 0..wpos.size-1
  u222[iii]=PartOfSpeech.analyze(lexicon_path,u111[iii])
  db.query ("UPDATE words SET pos='#{u222[iii]}' WHERE idw= #{iii} AND IDP
    = '#{ppos}' AND IDS= '#{spos}'" )
end

```



```

db.query ("UPDATE words SET pos='#{u222[iii]}' WHERE idw= #{wpos[iii]} AND IDP
=#{ppos[iii]}' AND IDS= '#{spos[iii]}'" )
#####
### The word level features #####
##### TF #####
vsel=db.query("SELECT * FROM WORDS")
esel=0
while row=vsel.fetch_row do
  sent5[esel]= row[3]
  esel=esel+1
end
for i11 in 0..sent5.size-1
  v1 = db.query("SELECT * FROM WORDS WHERE WORD='#{sent5[i11]}'")
  lengthw[i11]=sent5[i11].length
  e=1
  while row=v1.fetch_row do
    u[e]= row[0]
    e=e+1
  end
  r= v1.num_rows
  db.query("UPDATE WORDS SET TF=#{r} , length=#{lengthw[i11]}' WHERE
WORD='#{sent5[i11]}' ")
end
##### TF-ISF #####
v3=db.query("SELECT * FROM WORDS ")
e3=1
while row=v3.fetch_row do
  u3[e3]= row[5]
  u4[e3]=row[3]
  e3=e3+1
end
v6=db.query("SELECT * FROM SENTENCES ")
e5=1
while row=v6.fetch_row do
  u6[e5]= row[0]
  e5=e5+1
end
m6=v6.num_rows
for i3 in 0..sent5.size-1
  v5=db.query("SELECT * FROM WORDS WHERE WORD='#{sent5[i3]}' GROUP BY IDS=IDP")
  e4=1
  while row=v5.fetch_row do
    u5[e4]= row[0]
    e4=e4+1
  end
  n5=v5.num_rows
  f3=0
  for j3 in 0..u3.size-1
    u3[j3]=(u3[j3])

```



```

f3=(u3[j3]).to_i
db.query("UPDATE WORDS SET TF_ISF=#{f3*(Math.log(m6/n5))}' WHERE WORD=#{u4[j3]}' ")
end
end
##### Existence in Title#####
for i2 in 0..sent5.size-1
v2=db.query("SELECT * FROM WORDS WHERE IDS=1 and IDP=1")
e2=1
while row=v2.fetch_row do
u6[e2]= row[3]
e2=e2+1
end
f2=0
for j2 in 0..u6.size-1
if sent5[i2]==u6[j2]
f2=f2+1
end
end
if f2>0
db.query("UPDATE WORDS SET EXI_TITLE=1 WHERE WORD=#{sent5[i2]}' ")
end
end
##### Store 1 in Length word#####
wvone = db.query("SELECT * FROM words")
weone=0
while row=wvone.fetch_row do
wwone[weone]=row[0]
wpone[weone]= row[2]
wsone[weone]=row[1]
wone[weone]=row[9]
weone=weone+1
end
for wione in 0..wwone.size-1
if (wone[wione].to_i>=4 )

db.query("UPDATE words SET length=1 WHERE IDP=#{wpone[wione]}' AND
IDS=#{wsone[wione]}' AND IDW=#{wwone[wione]}")
else
db.query("UPDATE words SET length=0 WHERE IDP=#{wpone[wione]}' AND
IDS=#{wsone[wione]}' AND IDW=#{wwone[wione]}")
end
end
#####Store 1 in TF-ISF word#####
tvone = db.query("SELECT * FROM words")
teone=0
while row=tvone.fetch_row do
twone[teone]=row[0]
tpone[teone]= row[2]
tsone[teone]=row[1]
tone[teone]=row[6]

```



```

teone=teone+1
end
for tione in 0..twone.size-1
  if (tione[tione].to_i>=5 )
    db.query("UPDATE words SET TF_ISF=1 WHERE IDP='#{tpone[tione]}' AND
IDS='#{tsone[tione]}' AND IDW='#{twone[tione]}'")
  else
    db.query("UPDATE words SET TF_ISF=0 WHERE IDP='#{tpone[tione]}' AND
IDS='#{tsone[tione]}' AND IDW='#{twone[tione]}'")
  end
end
##### Word weight #####
v0=db.query("SELECT * FROM WORDS")
e0=0
while row=v0.fetch_row do
  ws4[e0]=row[0]
  ss4[e0]=row[1]
  ps4[e0]=row[2]
  tf_isf[e0]=row[6]
  tit[e0]=row[7]
  len[e0]=row[9]
  e0=e0+1
end
sum=0.0
for iw in 0..ws4.size-1
  wd[iw]=2*(tf_isf[iw]).to_f + 1.5*(tit[iw]).to_i + 0.5*(len[iw]).to_i
  db.query("UPDATE WORDS SET WEIGHT='#{wd[iw]}' WHERE IDW='#{ws4[iw]}' AND
IDS='#{ss4[iw]}' AND IDP='#{ps4[iw]}'")
  sum = sum + (wd[iw]).to_f
end
wea=0.0
wea=sum / ws4.size
for iwa in 0..ws4.size-1
  if wd[iwa].to_f >= wea
    db.query("UPDATE WORDS SET keyword=1 WHERE IDW='#{ws4[iwa]}' AND
IDS='#{ss4[iwa]}' AND IDP='#{ps4[iwa]}'")
  end
end
#####
##### The Sentence level features #####
##### Location of the sentence #####
db.query("UPDATE sentences,paragraph
SET location='1'
WHERE sentences.IDP= Paragraph.IDP AND( sentences.IDS=1 OR sentences.IDS =
Paragraph.length)")
##### Biased words Feature #####
biased=mytext5.split("\n")
v9=db.query("SELECT * FROM SENTENCES")
e9=0

```



```

while row=v9.fetch_row do
  s3[e9]= row[3]
  pp3[e9]=row[1]
  k[e9]=row[0]
  e9=e9+1
end

for i9 in 0..s3.size-1
  ba=0
  q=s3[i9].split(" ")

  for i0 in 0..q.size-1

    for i5 in 0..biased.size-1
      if q[i0]==biased[i5]
        ba=ba+1
      end
      db.query("UPDATE SENTENCES SET BIASED=#{ba} WHERE IDS='#{k[i9]}' AND IDP='#{pp3[i9]}'")
    end
  end
end

##### Cue phrase existence#####
  cue2=mytext55.split("\n")
  for i55 in 0...cue2.size-1
    v9=db.query("SELECT * FROM WORDS WHERE WORD='#{cue2[i55]}'")
    e9=1
    while row=v9.fetch_row do
      s22[e9]= row[1]
      pp22[e9]=row[2]
      e9=e9+1
    end
    for i66 in 0..s22.size-1
      db.query("UPDATE SENTENCES SET cue_phras=-1 WHERE IDS='#{s22[i66]}' AND
        IDP='#{pp22[i66]}'")
    end
  end
end

##### Existence of keywords #####
  vk=db.query("SELECT * FROM SENTENCES")
  ek=0
  while row=vk.fetch_row do
    sk[ek]= row[3]
    pk[ek]=row[1]
    kk[ek]=row[0]
    ek=ek+1
  end
  ws0=[]
  for ik9 in 0..sk.size-1
    vs=db.query("SELECT * FROM WORDS where keyword=1 AND IDS='#{kk[ik9]}' AND
      IDP='#{pk[ik9]}'")
    bk=0
  end

```



```

es0=0
while row=vs.fetch_row do
  ws0[es0]= row[3]
  es0=es0+1
end
qk=sk[ik9].split(" ")

for ik0 in 0..qk.size-1
  for ik1 in 0..ws0.size-1
    if qk[ik0]==ws0[ik1]

      bk=bk+1
    end
  end
end
db.query("UPDATE SENTENCES SET keyword=#{bk} WHERE IDS='#{kk[ik9]}' AND
IDP='#{pk[ik9]}'")
end
#####Store 1 in Biased #####
bvone = db.query("SELECT * FROM SENTENCES")
beone=0
while row=bvone.fetch_row do
  bpone[beone]= row[1]
  bsone[beone]=row[0]
  bone[beone]=row[8]
  beone=beone+1
end
for bione in 0..bsone.size-1
  if (bone[bione].to_i>=4 )
    db.query("UPDATE SENTENCES SET Biased=1 WHERE IDP='#{bpone[bione]}' AND
IDS='#{bsone[bione]}' ")
  else
    db.query("UPDATE SENTENCES SET Biased =0 WHERE IDP='#{bpone[bione]}' AND
IDS='#{bsone[bione]}' ")
  end
end
##### Store 1 in keyword #####
kvone = db.query("SELECT * FROM SENTENCES")
keone=0
while row=kvone.fetch_row do
  kpone[keone]= row[1]
  ksone[keone]=row[0]
  kone[keone]=row[7]
  keone=keone+1
end
for kione in 0..ksone.size-1
  if (kone[kione].to_i>=6 )
    db.query("UPDATE SENTENCES SET keyword=1 WHERE IDP='#{kpone[kione]}' AND
IDS='#{ksone[kione]}' ")
  else

```



```

db.query("UPDATE SENTENCES SET keyword =0 WHERE IDP='{kpone[kione]}' AND
IDS='{ksone[kione]}' ")

end
end
##### Store 1 in length sentence #####
vone = db.query("SELECT * FROM Sentences")
eone=0
while row=vone.fetch_row do
  pone[eone]= row[1]
  sone[eone]=row[0]
  lone[eone]=row[5]
  eone=eone+1
end
for ione in 0..sone.size-1
  if (lone[ione].to_i>8 )

    db.query("UPDATE SENTENCES SET length=1 WHERE IDP='{pone[ione]}' AND
IDS='{sone[ione]}'")
  else
    db.query("UPDATE SENTENCES SET length=0 WHERE IDP='{pone[ione]}' AND
IDS='{sone[ione]}'")
  end
end
#####
##### Summary Generation #####
#####
##### Sentence weighting & Sentence selection #####
vs0=db.query("SELECT * FROM sentences")
es=0
while row=vs0.fetch_row do
  wes4[es]= row[0]
  psw[es]=row[1]
  wws[es]=row[2]
  loc[es]=row[4]
  lens[es]= row[5]
  cp[es]=row[6]
  kw[es]=row[7]
  bw[es]=row[8]
  es=es+1
end
sum1=0.0
for iws in 0..wes4.size-1
  wds[iws] =2*(loc[iws]).to_i + 1.5*(lens[iws]).to_i + 2.5*(cp[iws]).to_i + 2.5*(kw[iws]).to_i
+ 1*(bw[iws]).to_i
  db.query ("UPDATE SENTENCES SET WEIGHT='{wds[iws]}' WHERE IDS='{wes4[iws]}' AND
IDP='{psw[iws]}'")
  sum1 = sum1 + (wds[iws]).to_i
end

```



```

y=[]
summary=[]
weas =0.0
es1=0
weas =sum1 / wes4.size
puts weas
for iwas in 0..wes4.size-1
  if wds[iwas].to_i >= weas
    summary[es1]=wws[iwas]
  end
end

#####
####                               Post Processing                               ####
#####
##### Connecting words #####
conn=mytext5.split("\n")
sconn=[]
ppconn=[]
qconn=[]
kconn=[]
vconn=db.query("SELECT * FROM SENTENCES ")
econn=0
while row=vconn.fetch_row do
  sconn[econn]= row[3]
  ppconn[econn]=row[1]
  kconn[econn]=row[0]
  econn=econn+1
end
for iconn in 0..sconn.size-1
  baconn=0
  qconn=sconn[iconn].split(" ")
  for jconn in 0..biased.size-1
    if qconn[0]==conn[jconn]
      baconn=baconn+1
    end
  end
  db.query("UPDATE SENTENCES SET BIASED=#{ba} WHERE IDS='#{k[i9]}' AND
IDP='#{pp3[i9]}'")
end
end
end
end

##### Removing URLs and E-mails #####
regex = Regexp.new '([^\s]+)([^\s]+)*@([^\s]+\.[^\s]+)([^\s]+)'
regex1 = Regexp.new '((https?://|www\.)+([^\s]+\.[^\s]+)+(:\d+)?(\/[^\s]+\.[^\s]+)*(\s+)?)?'
xxxx = summary.to_s
y= xxxx.gsub!( regex, "" )
y= xxxx.gsub!( regex1, "" )
##### Sentence-to-Sentence Similarity #####
for sim in 0..summary.size-1
  b1000=summary[sim]

```



```

vsim=db.query("SELECT * FROM SENTENCES where sentenceo='{summary[sim]}")
esim=0
while row=vsim.fetch_row do
  simw[esim]= row[9]
  esim=esim+1
end
for sim2 in 0..summary.size-1
  b1000=summary[sim]
  vsim2=db.query("SELECT * FROM SENTENCES where sentenceo='{summary[sim2]}")
  esim2=0
  while row=vsim.fetch_row do
    simw2[esim2]= row[9]
    esim2=esim2+1
  end
  d1000= c1000.split(" ")
  f1000= b1000.split(" ")
  for i1000 in 0..f1000.size-1
    query = Metonym.new f1000[i1000]
    a1000=query.synonyms
    for n1000 in 0..d1000.size-1
      for j1000 in 0..a1000.size-1
        if(d1000[n1000]==(a1000[j1000]))
          d1000[n1000]=f1000[i1000]
        end
      end
    end
  end
  dd1000=d1000.join(" ")
  ff1000 = f1000.join(" ")
  dist= Text::Levenshtein.distance(dd1000,ff1000)
  if dist<(65/100)
    if (simw[0]).to_f>=(simw2[0]).to_f
      low[z100]=summary[sim2]
      z100=z100 + 1
    else
      low[z100]=summary[sim]
      z100 = z100 +1
    end
  end
end
end
end
end
fsummary=[]
for fsum in 0..summary.size-1
  for fsum2 in 0..low.size-1
    if summary[fsum]==low
      summary= summary[fsum]
    end
  end
end
end
end

```


Appendix B

Lists of

1. *Stopwords*
2. *Tags of word tagging*
3. *Cue phrases*
4. *Connecting words*

1. List of stopwords

Stop Word	Type	Stop Word	Type
a	useless	Beginning	Useless
about	useless	Co	Useless
Above	useless	Company	Useless
According	useless	corp	Useless
Across	useless	Corporation	Useless
Actually	useless	Could	Useless
Adj	useless	Couldn't	Useless
After	useful	Did	Useless
Afterwards	useless	Didn't	Useless
Again	useless	Do	Useless
Against	useless	Dose	Useless
All	useless	Doesn't	Useless
Almost	useless	Don't	Useless
Alone	useless	Down	Useless
Along	useless	During	Useless
Already	useless	Each	Useless
Also	useful	Eg	Useless
Although	useful	Eight	Useless
Always	useless	Eighty	Useless
Among	useless	Either	Useless
Amongst	useless	Else	Useless
An	useless	Elsewhere	Useless
And	useful	End	Useful
Another	useless	Ending	Useless
Any	useless	Enough	Useless
Anyhow	useless	Etc	Useless
Anyone	useless	Even	Useless
Anything	useless	Every	Useless
Anywhere	useless	Everyone	Useless
Are	useless	Everything	Useless
Aren't	useless	Everywhere	Useless
Around	useless	Except	Useless
As	useful	Few	Useless
At	useless	Fifty	Useless
Be	useless	First	Useless
Became	useless	Five	Useful
Because	useful	For	Useless
Become	useless	Former	Useless
Becomes	useless	Formerly	Useless
Becoming	useless	Forty	Useless
Been	useless	Found	Useless
Before	useless	Four	Useless
beforehand	useless	From	Useless

Further	useful	Like	Useful
Had	Useless	Likely	Useless
Here	Useless	It'd	Useless
Heres	Useless	Made	Useless
Hereafter	Useless	make	Useless
Hereby	Useless	behind	Useless
Herein	Useless	Being	Useless
Hereupon	Useless	Below	Useless
Hers	Useless	Beside	Useless
Herself	Useless	Besides	Useless
Him	Useless	Between	Useless
Himself	Useless	Beyond	Useless
His	Useless	Billion	Useless
How	Useless	Both	Useless
However	Useful	But	Useful
Hundred	Useless	Moreover	Useful
I	Useless	Most	Useless
Id	Useless	Mostly	Useless
Ill	Useless	Mr	Useless
Im	Useless	Mrs	Useless
Ive	Useless	Much	Useless
Ie	useful	Must	Useless
If	Useless	My	Useless
In	Useless	Myself	Useless
Inc	Useless	Namely	Useless
Indeed	Useless	Neither	Useless
Instead	Useful	Never	Useless
Into	Useless	Nevertheless	Useful
Is	Useless	Next	Useless
Isn't	Useless	Nine	Useless
It	Useless	Ninety	Useless
Its	Useless	No	Useless
Its	Useless	Nobody	Useless
Itself	Useless	None	Useless
Last	Useful	Nonetheless	Useless
later	Useless	No one	Useful
Latter	Useless	Nor	Useless
Latterly	Useless	Not	Useless
Least	Useless	Nothing	Useless
Less	Useless	Now	Useless
Let	Useless	Nowhere	Useless
Lets	Useless	Of	Useless

Off	Useless	The	
Often	Useless	Theire	Useless
On	Useless	Them	Useless
Once	Useless	Themselves	Useless
One	Useless	Then	Useless
Ones	Useless	Thence	Useless
Only	Useless	There	Useless
Onto	Useless	Thered	Useless
Or	Useful	There'll	Useless
Other	Useless	There're	Useless
Others	Useless	Theres	Useless
Otherwise	Useless	There've	Useless
Has	Useless	Makes	Useless
Hasn't	Useless	Many	Useless
Have	Useless	Maybe	Useless
Haven't	Useless	Me	Useless
He	Useless	Meantime	Useless
Hed	Useless	Meanwhile	Useless
Hell	Useless	Might	Useless
Hes	Useless	Million	Useless
Hence	useful	Miss	Useless
Her	Useless	More	Useless
Several	Useless	Throughout	Useless
She	Useless	Thru	Useless
Shed	Useless	Thus	Useless
Shell	Useless	To	Useless
Shes	Useless	Together	Useless
Should	Useless	Too	Useless
Shouldn't	Useless	Toward	Useless
Since	Useless	Towards	Useless
Six	Useless	Trillion	Useless
Sixty	Useless	Twenty	Useless
So	useful	Two	Useless
Some	Useless	Under	Useless
Somehow	Useless	Unless	Useless
Someone	Useless	Unlike	Useless
Something	Useless	Unlikely	Useless
Sometime	Useless	Until	Useless
Sometimes	Useless	Up	Useless
Somewhere	Useless	Upon	Useless
Still	Useless	Us	Useless
Stop	Useless	Used	Useless
Such	Useless	Using	Useless
Taking	Useless	Very	Useless
Ten	Useless	Via	Useless
Than	Useful	Ve	

That	Useful	Was	
Thatll	Useless	Wasnts	Useless
That's	Useless	We	Useless
Wed	Useless	They've	Useless
Well	Useless	Thirty	Useless
Were	Useless	This	Useless
Weve	Useless	Those	useless
Well	Useless	Though	useful
Were	Useless	Thousand	Useless
Weren't	Useless	Three	useless
What	Useless	Through	Useless
Whatll	Useless	Whence	Useless
Whats	Useless	Whenever	Useful
What've	Useless	Where	Useless
Whatever	Useless	Wheres	Useless
When	Useless	Whreafter	Useless
Our	Useless	Whereas	Useless
Ours	Useless	Whereby	Useless
Ourselves	Useless	Wherein	Useless
Out	Useless	Whereupon	Useless
Over	Useless	Whether	Useless
Overall	Useless	Which	Useless
Own	Useless	While	Useful
Per	Useless	Whither	Useless
Perhaps	Useless	Who	Useless
Rather	Useless	Whod	Useless
Recent	Useless	Wholl	Useless
Recently	Useless	Whos	Useless
Same	Useless	why	Useless
Seemed	Useless	Theyre	Useless
Seeming	Useless	They'll	Useless
Seems	Useless	Theyd	useless
Seven	Useless	They	Useless
Seventy	Useless	These	Useless
Thereafter	Useless	Thereupon	Useless
Thereby	Useless	Therein	Useless
Therefore	useless		

2. Tags of word tagging

Definition	Tag
Adjective	
comparative adjective	JJ
semantically superlative adjective (chief, top)	JJR
morphologically superlative adjective (biggest)	JJS
cited word (hyphenated after regular tag)	JJT
singular or mass noun	NC
possessive singular noun	NN
plural noun	NN\$
possessive plural noun	NNS
proper noun or part of name phrase	NN\$S
possessive proper noun	NP
plural proper noun	NP\$
comparative adverb	NPS
superlative adverb	RBR
verb, base form	RBT
verb, past tense	VB
verb, present participle/gerund	VBD
verb, past participle	VBG
verb, 3rd. singular present	VCN
wh- determiner (what, which)	VBZ
	WDT

3. List of cue phrases

Cue phrase	Type	Cue phrase	Type
addition	stigma	paper	bonus
because	Stigma	generally	bonus
detail	Stigma	finally	bonus
enumerate	Stigma	last	bonus
example	Stigma	Notwithstanding	bonus
explain	Stigma	end	bonus
further	Stigma	purpose	bonus
However	Stigma	accordingly	bonus
i.e.	Stigma	so	bonus
illustrate	Stigma	conclusion	bonus
include	Stigma	therefore	bonus
including	Stigma	accordingly	bonus
instance	Stigma	hence	bonus
instead	Stigma	consequently	bonus
like	Stigma	result	bonus
occasionally	Stigma	particular	bonus
specifically	bonus	especially	bonus
summary	bonus		

4. List of connecting words

NO	Connecting Words
1	accordingly
2	again
3	also
4	besides
5	hence
6	henceforth
7	however
8	incidentally
9	meanwhile
10	moreover
11	namely
12	nevertheless
13	otherwise
14	then
15	therefore
16	thus
17	and
18	but
19	or
20	yet
21	so
22	once
23	Than
24	That
25	Till
26	Whenever
27	whereas

Appendix C

Examples of generated summaries

Original text A

MULTIMEDIA

is simply multiple forms of media integrated together. Media can be text, graphics, audio, animation, video, data, etc. An example of multimedia is a web page on the topic of Mozart that has text regarding the composer along with an audio file of some of his music and can even include a video of his music being played in a hall.

Besides multiple types of media being integrated with one another, multimedia can also stand for interactive types of media such as video games CD ROMs that teach a foreign language, or an information Kiosk at a subway terminal. Other terms that are sometimes used for multimedia include hypermedia and rich media.

The term Multimedia is said to date back to 1965 and was used to describe a show by the Exploding Plastic Inevitable. The show included a performance that integrated music, cinema, special lighting and human performance. Today, the word multimedia is used quite frequently, from DVD's to CD ROMs to even a magazine that includes text and pictures

Multimedia has become a huge force in American culture, industry and education. Practically any type of information we receive can be categorized as multimedia, from television, to magazines, to web pages, to movies, multimedia is a tremendous force in both informing the American public and entertaining us.

Advertising is perhaps one of the biggest industry's that use multimedia to send their message to the masses. Where one type of media, let's say radio or text can be a great way to promote an item, using multimedia techniques can significantly make an item being advertised better received by the masses and in many cases with greater results.

Multimedia in Education has been extremely effective in teaching individuals a wide range of subjects. The human brain learns using many senses such as sight and hearing. While a lecture can be extremely informative, a lecture that integrates pictures or video images can help an individual learn and retain information much more effectively.

Using interactive CD ROM's can be extremely effective in teaching students a wide variety of disciplines, most notably foreign language and music.

Proposed system summary for A

Multimedia

Is simply multiple forms of media integrated together. An example of multimedia is a web page on the topic of Mozart that has text regarding the composer along with an audio file of some of his music and can even include a video of his music being played in a hall.

Besides multiple types of media being integrated with one another, multimedia can also stand for interactive types of media such as video games CD ROMs that teach a foreign language, or an information Kiosk at a subway terminal. Today, the word multimedia is used quite frequently, from DVD's to CD ROMs to even a magazine that includes text and pictures.

Practically any type of information we receive can be categorized as multimedia, from television, to magazines, to web pages, to movies, multimedia is a tremendous force in both informing the American public and entertaining us.

Multimedia in Education has been extremely effective in teaching individuals a wide range of subjects. Using interactive CD ROM's can be extremely effective in teaching students a wide variety of disciplines, most notably foreign language and music

Original text B

Bluetooth

Bluetooth technology is named after Harald Bluetooth, a Danish king who managed to consolidate Denmark and a part of Norway in the 1900s. The choice for the name of this technology is a manifestation of how influential and central the companies from this region are to the telecommunications industry.

Bluetooth is a networking technology that does not rely on user control or large amounts of power. By keeping the transmission power to an extremely low setting (1 milliwatt), Bluetooth is ideal for mobile battery operated devices. Moreover, Bluetooth does not rely on the user since it can automatically detect and communicate with other Bluetooth devices without any user input.

Bluetooth technology relies on two things, a radio frequency technology and the protocol software enabling it to transmit data to other devices. Bluetooth-capable devices can transmit data to other devices not within the line of sight of the user. It also enables different devices to communicate using certain rules such as the amount of data that will be sent, the type of communication between the devices and the radio frequency or frequencies this communication will take place. These protocols ensure that Bluetooth devices experience the least amount of interference from other Bluetooth capable objects while communicating with each other

Bluetooth is not a one-on-one data transmission technology so it can communicate with up to eight devices within its transmission radius at one time. A Bluetooth device will use at most 1600 different and randomly chosen frequencies every second within the course of its transmission to minimize the probability of other devices using the same frequency and to minimize interference time when it does coincide with another device using the same frequency

Infrared devices, however, are less susceptible to interference than Bluetooth devices. This means that you can be sure that the data will be sent to the intended recipient

without any distortion or inaccuracies. Improvements in the Bluetooth technology however minimizes this problem by enabling the Bluetooth devices to hop frequencies and communicate within a specific frequency range. Therefore, although there is still risk of interference, the chances of it happening are very minimal. If such occurs, it will happen only in a very brief period of time and a software will be available to correct any consequent distortion.

Proposed system summary for B

Bluetooth

Bluetooth technology is named after Harald Bluetooth, a Danish king who managed to consolidate Denmark and a part of Norway in the 1900s. By keeping the transmission power to an extremely low setting (1 milliwatt), Bluetooth is ideal for mobile battery operated devices. Moreover, Bluetooth does not rely on the user since it can automatically detect and communicate with other Bluetooth devices without any user input.

Bluetooth technology relies on two things, a radio frequency technology and the protocol software enabling it to transmit data to other devices. Bluetooth-capable devices can transmit data to other devices not within the line of sight of the user.

Bluetooth is not a one-on-one data transmission technology so it can communicate with up to eight devices within its transmission radius at one time. A Bluetooth device will use at most 1600 different and randomly chosen frequencies every second within the course of its transmission to minimize the probability of other devices using the same frequency and to minimize interference time when it does coincide with another device using the same frequency.

Improvements in the Bluetooth technology however minimizes this problem by enabling the Bluetooth devices to hop frequencies and communicate within a specific frequency range.

Appendix D

Paper

References

- [1] Aliguliyev M ,(2009). A new sentence similarity measure and sentence based extractive technique for automatic text summarization. Expert Systems with Applications.
- [2] Alonso A .(2005). Representing discourse for automatic text summarization via shallow NLP techniques. Master thesis. Department de Lingüística General. Universitat de Barcelona.
- [3] Amini A& Gholamrezazadeh S.(2009) A Comprehensive Survey on Text Summarization Systems. IEEE. JOURNAL OF EMERGING TECHNOLOGIES IN WEB INTELLIGENCE, VOL. 1, NO. 1,
- [4] Arcelon d. (2008). What is automatic text summarization? Retrieved from: <http://people.dsv.su.se/~hercules/textsammanfattningeng.html> (accessed: Nov 09, 2010)
- [5] Basu L.2009.Ruby WordNet. Retrieved from <http://deveiate.org/projects/Ruby-WordNet> (accessed: Apr 09, 2011).
- [6] Battley L.2010. text . Retrieved from: <http://rubygems.org/gems/text> (accessed: May 01, 2011)
- [7] Berkeley G.(2003). What Is Text Mining?. Retrieved from: <http://people.ischool.berkeley.edu/~hearst/text-mining.html> (accessed: Nov 19, 2010).
- [8] Berube D.2007. Practical Ruby Gems. Printed and bound in the United States of America
- [9] Buss k.(2007). Literature review on preprocessing for text mining. Institute of creative technologies. Retrieved 41arcelon 25, 2009 from. <http://www.ioct.dmu.ac.uk/>.
- [10]Chakrabarti P, Basu J.(2010). text summarization and discovery of frames and relationship from natural language text. (ijcse) international journal on computer science and engineering vol. 02, no.

- [11]Chang T, Hsiao F w.(2008). A hybrid approach to automatic text summarization.IEEE.
- [12]Chengcheng L.(2010).Automatic Text Summarization Based On Rhetorical Structure Theory. IEEE. 2010 International Conference on Computer Application and System Modeling (ICCASM 2010).
- [13]Collingbourne H,2009. THE BOOK OF RUBY . Retrieved from <http://www.sapphiresteel.com/>
- [14]Dalianis S, (2005) .greeksum a greek text summarizer. Master thesis. George Pachantouris.
- [15]Dongrui W, Jerry M. Mendel, Jhiin ." Linguistic Summarization Using IF-THEN Rules".2010 IEEE.
- [16]Fattah A.(2008).Automatic Text Summarization.World Academy of Science, Engineering and Technology <http://www.doc.ic.ac.uk/~lkhw98/project>
- [17]GitHub M.2011. Development Kit. Retrieved from: <https://github.com/oneclick/rubyinstaller/wiki/development-kit> (accessed: May 01, 2011)
- [18]Jerry G.2010. NetBeans IDE 6.9.1 . Retrieved from <http://netbeans.org/community/releases/69/>
- [19]Jezek K.(2007). Automatic Text Summarization.IEEE World Academy of Science, Engineering and Technology 2007
- [20]LeiLI L, Ying X, Hongyan L.(2010). Multi-Document Summarization Based on Improved Features and Clustering. IEEE.
- [21]Louie J. (2006). A hmm – viterbi algorithm based part of speech tagger. 3rd national natural language processing symposium building language tools and resources
- [22]Manna S , b. Sumudu U.(2007). Fuzzy word similarity: a semantic approach using wordnet. IEEE.
- [23]Mehdi B, Bashiri H.(2010). Evaluation of Clustering and Summarizing in Distributed Latent Semantic Indexing . IEEE.

- [24]Pardo T, Alexandre S.(2008) . Evaluation of Automatic Text Summarization Methods Based on Rhetorical Structure Theory. IEEE. Eighth International Conference on Intelligent Systems Design and Applications.
- [25]Pazienza, M., Pennacchiotti, M. & Zanzotto, F. (2005) .Terminology Extraction: An Analysis of Linguistic and Statistical Approaches. Springer Verlag.
- [26]Publishing W. 2005. Beginning MySQL. simultaneously in Canada.
- [27]Radev D, Teufel H, Saggion, Wai J .(2001). Evaluation of Text Summarization in a Cross-lingual Information Retrieval Framework. IEEE. JOURNAL OF EMERGING TECHNOLOGIES IN WEB INTELLIGENCE
- [28]Radev D.(2002). Multi Document Centroid-based Text Summarization. Proceedings of the ACL-02 Demonstrations Session, Philadelphia.
- [29]Reddavis s.2010.Part_Of_Speech. Retrieved from: http://rubygems.org/gems/part_of_speech (accessed: May 02, 2011)
- [30]sankupellay M.(2006). Malay-language stemmer. Sunway academic journal3.
- [31]Seo J & Youngjoong Koa.(2008). An effective sentence-extraction technique using contextual information and statistical approaches for text summarization. ScienceDirect.
- [32]Shams R, Hashem M, Hossain A.(2010) . Corpus-based Web Document Summarization using Statistical and Linguistic Approach. IEEE. International Conference on Computer and Communication Engineering (ICCCE 2010), Malaysia.
- [33]Suanmali L,.(2009). Sentence features fusion for text summarization using fuzzy logic. IEEE. International conference on hybrid intelligent systems
- [34]Talim S .2007. Ruby MySQL Tutorial .Retrieved from: <http://rubylearning.com/blog/2007/05/14/ruby-mysql-tutorial/> (accessed: May 09, 2011)
- [35]Tatar D, Andreea, Gabriela S.(2008). Lexical Chains Segmentation in Summarization.IEEE. 10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing.

- [36] Vishal G.(2010). A Survey of Text Summarization Extractive Techniques. journal of emerging technologies in web intelligence, vol. 2, no. 3.
- [37] Wan X . Yang J.(2008). Multi-Document Summarization Using Cluster-Based Link Analysis. SIGIR'08, July 20–24, 2008, Singapore.
- [38] Willett P.(2006).the porter stemming algorithm: then and now. Electronic library and information systems.
http://eprints.whiterose.ac.uk/1434/1/willettp9_porterstemmingreview.pdf
- [39] Wong L ,(2009), ANSES Automatic News Summarization and Extraction System Meng Computing Imperial College Dept. of Computing
- [40] Yanmin C, Lou X .(2007). Multi-document Summarization Based on Cohesion with Disambiguation IEEE. Third International Conference on Natural Computation (ICNC 2007).
- [41] Yard L.2011. Metonym. Retrieved from:
<http://rubydoc.info/gems/metonym/0.1.0/frames> (accessed: May 02, 2011)
- [42]Ziyan S.(2010). The Design and Implementation of Domain-specific Text Summarization System based on Co-reference Resolution Algorithm. IEEE. Seventh International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2010) .