

**Semantic based Text Summarization for Single Document  
on Android Mobile Device**

by

Mellissa Lee Ai Lin

Dissertation submitted in partial fulfillment of  
the requirements for the  
Bachelor of Technology (Hons)  
(Business Information System)

SEPTEMBER 2012

Universiti Teknologi PETRONAS  
Bandar Seri Iskandar,  
31750 Tronoh  
Perak Darul Ridzuan

# CERTIFICATION OF APPROVAL

## **Semantic Based Text Summarization for Single Document on Android Mobile Device**

By

Mellissa Lee Ai Lin

A project dissertation submitted to the  
Information System Programme  
Universiti Teknologi PETRONAS  
in partial fulfilment of the requirement for the  
BACHELOR OF TECHNOLOGY (Hons)  
(Business Information System)

Approved by,

---

(Ms. Foong Oi Mean)

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

September 2012

## CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

---

MELLISSA LEE AI LIN

## **ABSTRACT**

The explosion of information in the World Wide Web is overwhelming readers with limitless information. Large internet articles or journals are often cumbersome to read as well as comprehend. More often than not, readers are immersed in a pool of information with limited time to assimilate all of the articles. It leads to information overload whereby readers are trying to deal with more information than they can process. Hence, there is an apparent need for an automatic text summarizer as to produce summaries quicker than humans. The text summarization research on mobile platform has been inspired by the new paradigm shift in accessing information ubiquitously at anytime and anywhere on Smartphones or smart devices. In this research, a semantic and syntactic based summarization is implemented in a text summarizer to solve the overload problem whilst providing a more coherent summary. Additionally, WordNet is used as the lexical database to semantically extract the text document which provides a more efficient and accurate algorithm than the existing summary system. The objective of the paper is to integrate WordNet into the proposed system called TextSumIt which condenses lengthy documents into shorter summarized text that gives a higher readability to Android mobile users. The experimental results are done using recall, precision and F-Score to evaluate on the summary output, in comparison with the existing automated summarizer. Human-generated summaries from Document Understanding Conference (DUC) are taken as the reference summaries for the evaluation. The evaluation of experimental results shows satisfactory results.

## **ACKNOWLEDGEMENT**

First and foremost, I would like to express my deepest appreciation and gratitude to the following people in helping me during the course of my final year project completion.

I offer my sincere gratitude to my supervisor, Ms. Amy Foong Oi Mean who has guided and assisted me throughout my final year project with patience and knowledge whilst allowing me to work in my own way. It has been such a great opportunity to be given the trust to proceed with the semantic based text summarization research which is a challenging field to explore.

Apart from that, I would also like to thank all of my friends and acquaintances who are willing to help me to evaluate my application for user testing purposes. Their responses helped me to acquire as much data as possible in completing this research project. Additionally, I would like to thank some of them who spend the time to guide me in the development of the application and algorithm.

Finally, I would like to thank my parents for giving me the moral support along the completion of my final year project. I hope, with the support from all of the people mentioned above, this research will be able to contribute as much as possible in improving the reliability and relevancy of the text summarizer especially on Android platform.

## TABLE OF CONTENT

<b>ABSTRACT</b>	iv
<b>ACKNOWLEDGEMENT</b>	v
<b>LIST OF FIGURE</b>	viii
<b>LIST OF TABLE</b>	viii
<b>ABBREVIATIONS AND NOMENCLATURES</b>	ix
<b>CHAPTER 1: INTRODUCTION</b>	1
1.1 Background of Study	1
1.2 Problem Statement	3
1.3 Objectives	4
1.4 Scope of Study	4
<b>CHAPTER 2: LITERATURE REVIEW.</b>	5
2.1 Introduction to Automatic Text Summarization (ATS)	5
2.1.1 Usage: Indicative or Informative	5
2.1.2 Audience: User-oriented or Generic	6
2.1.3 Derivation: Extractive or Abstractive.	6
2.2 Semantic Text Extraction	6
2.2.1 WordNet-based	7
2.3 Current development of mobile application.	9
2.3.1 Android based mobile phone application	9
2.3.2 Challenges of text summary on mobile applications	10
<b>CHAPTER 3: METHODOLOGY</b>	11
3.1 Software development Methodology	11
3.2 Project Activities	12
3.2.1. Planning Phase	12
3.2.2. Prototype Development Phase	13
3.2.3. Implementation Phase.	21
3.3 Tools	22

<b>CHAPTER 4: RESULTS AND DISCUSSION</b>	23
4.1 Preprocessing Enhancement	23
4.2 Summary Evaluation	25
4.2.1 Performance Measure	25
4.2.2 Existing Text Summarizer and Reference Summaries	26
4.2.3 Evaluation of Experimental Results	27
4.3 User Testing	30
<b>CHAPTER 5: CONCLUSION AND RECOMMENDATIONS</b>	33
5.1 Conclusion.	33
5.2 Recommendations.	34
<b>REFERENCES</b>	35
<b>APPENDIX A – SYSTEM APPLICATION’S HELP MANUAL</b>	38
<b>APPENDIX B – REVIEW ON APPLICATION AVAILABLE IN THE MARKET</b>	41
<b>APPENDIX C – PENN ENGLISH TREEBANK POS TAGS</b>	42
<b>APPENDIX D – TEXTSUMIT APPLICATION QUESTIONNAIRE</b>	43
<b>APPENDIX E – GANTT CHART</b>	44
<b>APPENDIX F – TECHNICAL REPORT</b>	45

## LIST OF FIGURES

FIGURE 2.1: DEVELOPMENT OF ANDROID APPLICATION .....	9
FIGURE 2.2: ANDROID VERSIONS DISTRIBUTION [17] .....	10
FIGURE 3.1: RAPID APPLICATION DEVELOPMENT (RAD) METHODOLOGY [19] .....	11
FIGURE 3.2: PROPOSED MODEL OF AUTOMATIC TEXT SUMMARIZER .....	15
FIGURE 3.3: MAIN HOMEPAGE OF TEXTSUMIT.....	17
FIGURE 3.4: TEXT INPUT SCREEN.....	17
FIGURE 3.5: SUMMARY OUTPUT SCREEN.....	17
FIGURE 3.6: SAVED DOCUMENTS SCREEN.....	17
FIGURE 4.1: THE AVERAGE RECALL, PRECISION AND F-SCORE GRAPH FOR TEXTSUMIT AND WORD AUTOSUMMARIZE USING ARTICLES FROM DUC 2002.....	29
FIGURE 4.2: EVALUATION RESULTS FOR EASE OF NAVIGATION OF APPLICATION .....	30
FIGURE 4.3: EVALUATION RESULTS FOR READABILITY OF SUMMARY .....	31
FIGURE 4.4: EVALUATION RESULTS FOR THE PERCENTAGE OF TIME SAVED BY USING THE APPLICATION.....	32

## LIST OF TABLES

TABLE 2.1: Examples of Semantic Relations in WordNet [15].....	8
TABLE 3.1: Functional Requirement 1 .....	14
TABLE 3.2: Functional Requirement 2 .....	14
TABLE 3.3: Functional Requirement 3 .....	14
TABLE 3.4: Functional Requirement 4 .....	14
TABLE 3.5: Functional Requirement 5 .....	15
TABLE 3.6: Non-Functional Requirement 1 .....	15
TABLE 4.1: Comparison of results between PorterStemmer and WordNet Stemmer .....	24
TABLE 4.2: Sample POS Tagging results.....	24
TABLE 4.3: The recall, precision and F-score for TextSumIt and Word AutoSummarize using DUC 2002 articles.....	28



## **ABBREVIATIONS AND NOMENCLATURES**

ADT	Android Development Tools
ATS	Automatic Text Summarizer
DUC	Document Understanding Conference
IDE	Integrated Development Environment
GUI	Graphical User Interface
POS	Part-Of-Speech
RAD	Rapid Application Development
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
SDK	Software Development Kit
TF-ISF	Term Frequency-Inverse Sentence Frequency
TSS	Total Sentence Score
KW	Keyword
WN	WordNet

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 Background of Study**

The increasing information available in the World Wide Web is overwhelming readers with immense data causing an information overload among readers. Information overload occurs when the readers are trying to deal with more information than they can process. These pools of information need to be accessed to extract its important content in order to generate constructive knowledge. In fact, this applies to companies of various industry lines who are seeking for useful knowledge to assist in their decision making.

As a result of the existence of abundant data, readers are constantly being challenged to accommodate more facts from various sources within a short period of time. Thus, reader's ultimate goal is to have a quick and easy way to retrieve the main gist of the available information, comprehend it to gain knowledge, and finally make decision efficiently and effectively based on the knowledge gained.

In this context, there is an apparent need for text summarization whereby in most cases, summaries are produced by humans. However, due to the increasing user demands, Automatic Text Summarization (ATS) has been proposed as a solution to accommodate the growing information while saving time in producing it manually. ATS extracts content of the document using an algorithm, produces coherent and correctly-deliberated summaries, and displays the most important points of the

original text to the user in a more condensed way and in accordance to each user's needs [1].

Most studies are focusing on extractive approach towards summarization. Nevertheless, summarized content would not be effective if all the sentences in the text are deemed important, as the reduction of any sentences will cause the summary to lose its information. Therefore, improved methods are proposed to identify and rate the importance of sentences to be extracted which enhance the summaries produced. Features such as Part-Of-Speech (POS) tagging, Title feature score, Position score, and WordNet score are the additional aspect proposed within this study.

Text summarization is substantially a complex task especially when involving deep natural language processing e.g., syntactic, semantic, or pragmatic. The two approaches towards creating text summaries are extraction and abstraction. Most researchers applied extractive summary as it is more difficult to develop abstractive summary due to its implementation of deep natural language processing which is still a growing field. The robustness of a summary depends on the coverage level of the summarization i.e. word-level and concept-level generalizations [2].

Today, the advanced technology era has challenged the ability to retrieve information at anytime and anywhere which result in the widespread of summarization for hand-held devices such as personal digital assistants (PDAs) and cell-phones [3]. However, due to the latest advancement in Smartphone devices, the demand has now diverged to the use of high-resolution touch screen mobile. Android has recently gain popularity among mobile users which make it a stepping stone for information retrieval on mobile devices.

According to G. Chang et. al. (2010), Android has the potential to change the Smartphone market industry since it is an open source platform, enabling customizable development for specific users [4]. The main challenge would be to generate a text summary of which a condensed but precised contents of input source text is developed on an Android mobile platform with minimal back-end processing.

In this study, it is proposed to generate a text summary of which a condensed but precised contents of input source text is developed on an Android mobile platform. The proposed approach will be able to extract text semantically as well as producing more coherent and precised sentences as the output.

## **1.2 Problem Statement**

Large internet articles or journals are often cumbersome to read as well as comprehend. More often than not, it is time-consuming and we have limited time to assimilate all of the articles which are at times, exceeding our capability to perceive. In addition, time is precious as today's fast paced era has caused people to demand for quick results.

As the information continues to grow exponentially, there exists a need to retrieve and filter the overloaded information. In fact, information overload is an increasing problem both in the workplace and in our general daily life. Moreover, readers are incapable of managing such huge pool of knowledge without a mechanism to support its readability. They either skip reading the content or not processing the information well, leading to wrong decisions made.

The problem with readers or researchers today, is that they lack a mechanism that supports the accessibility of information at anywhere and anytime. Accessing information via a computer would be burdensome if at that particular moment computers are not available. As technology advances, it becomes more convenient to access information on-the-fly by utilizing mobile devices. However, the crucial problem in comprehending texts on mobile devices is the readability of lengthy texts on the small screen of the devices.

### **1.3 Objectives of Study**

The main objective of this research work is closely related to the implementation of semantic extraction on text summarizer. The following are the objectives:

- To propose and explore the semantic based model of extractive text summarizer to generate effective and meaningful summaries.
- To implement and deploy the extractive document summarizer on Android platform and provide a higher readability of shorter text for users.
- To provide a higher readability of shorter text for users.

### **1.4 Scope of Study**

The following are the main components within the scope of this project:

- English text
- Smartphone –Android operating system
- WordNet
- Semantic extraction
- Standard datasets – Document Understanding Conference (DUC)

The scope of this research involves creating short version of any plain text documents (.txt format) in English language. The plain text documents will excludes materials such as images, diagrams, graphs, tables etc.

It will process text summaries using improved syntactic and semantic based algorithm as a combination to generate higher feature scores. WordNet is used as the lexical database to assist in the semantic extraction of the text to generate more meaningful summaries.

The proposed system is developed on an Android mobile platform. Mobile application development will be based on Android 2.3 (Gingerbread) which is the most stable platform and has the highest number of users. Application development will be build using Eclipse IDE with Android SDK and ADT.

Target users for the system are readers or anyone who needs to comprehend certain information quickly and view it at anywhere and anytime with their mobile devices.

## CHAPTER 2

### LITERATURE REVIEW

#### 2.1 Introduction to Automatic Text Summarization (ATS)

The earliest works of text summarization started in 1950s which are pioneered by Luhn [5] and Baxendale [6]. Recent research has explored the different types of summaries, techniques used to generate the summaries, and the evaluation methods implemented. The initial work started off with the implementation of statistical techniques in text summarization and gradually improves towards using natural language process (NLP), semantic analysis, fuzzy logic, swarm intelligence, and lastly hybrid fuzzy swarm [7]. The challenge to text summarization is the improvement of summary quality which remains as a key role in many researches.

There are two categories of text summarizers namely, statistical and linguistic. Statistical summarizers operate by finding the important sentences using statistical techniques such as word frequency (Luhn, 1958), text position (Baxendale, 1958), cue words and heading (Edmunson, 1969), sentence position (Lin & Hovy, 1997) and etc. On the other hand, linguistic summarizers use knowledge about the language such as syntax, semantics (Liu & Troels Andreasen, 2009) usage etc. to summarize a document [8].

Based on Hovy and Lin (1999) and Sparck Jones (1998) research, summaries can be categorized by the following criteria [2] [9]:

##### 2.1.1 Usage: Indicative or Informative.

Indicative summaries usually provide the general concepts of the text document without showing specific content. Informative summaries on the other hand, reflect part of the content which allows readers to describe the content of the input text.

### **2.1.2 Audience: User-oriented or Generic**

User-oriented summaries focus on the interest of the readers on certain topics. It favors specific themes or aspects of the text. Generic summaries convey the point of view of the authors on the input text.

### **2.1.3 Derivation: Extractive or Abstractive**

**Extractive Summary** uses a fragment of the source text (key clauses, key phrases, sentences, etc) to structure the summary, i.e., summary copied from input [10]. Generally, it selects information which is presumed the most important by the system and organized them to form the summary. These summaries lack the coherence as compared to abstractive summaries as it only conveys an approximate content of the source text.

**Abstractive Summary** reconstructs the extracted sentences, i.e., paraphrasing sentences to form a more cohesive and coherent summary. It could generate an entire different sentence structure but retain the original meaning of the sentences. This method can condense text more strongly as compared to extraction by developing an understanding and expressing main concept of documents in clear natural language [10].

Basically, there are three main parts to automatic text summarization process [9]:

- 1) *Preprocessing step* – original source text is interpreted and representation of source text is obtained.
- 2) *Processing step* – representation of source text is transformed into summary representation using applied algorithm.
- 3) *Generation step* – final summary text is generated from the summary representation.

## **2.2 Semantic Text Extraction**

Semantic extraction involves the understanding of the structure and meaning of the natural language to produce semantic information from text documents [11]. However, the critical issue in extracting text semantically is the ambiguity and uncertainty of the meaning of texts. The improvement in summaries is achieved by

applying lexical knowledge (e.g. WordNet) towards the text summaries to build a more comprehensive text. In addition, the cohesiveness of sentences can be enhanced by mapping the terms within the sentence to similar concepts using WordNet.

Based on the coverage level of processing, it can be divided into three categories, namely surface level, discourse level and entity level. Semantic extraction is part of the entity level approach. The entity-level approach builds internal representation of the text input. It then models the text entities with their relationships [12].

### **2.2.1 WordNet-based**

WordNet is a lexical database available online which contains a large repository of English lexical terms (also supports multilingual WordNet). Its structure is useful for linguistics and natural language processing implementation. In WordNet, it connects four types of Part-of-Speech (POS): nouns, verbs, adjectives, and adverbs in which it groups the words into sets of synonyms called *synsets* [13]. Each synsets consists of the word, its explanation and its synonyms.

The synsets are connected among other synsets using several semantic relations such as hypernym/hyponym for nouns and hypernym/troponym for verbs [13]. These relations consists of hierarchies i.e. holonymy (is-a-kind-of) and meronymy (is-a-part-of). If there is more than one senses for a word, it will be arranged in an order of the most frequently used sense to the least frequently used. An example is shown in Table 2.1 where words are connected through semantic relation such as synonym, antonym, hyponym, meronym, and troponym etc.

The main idea of WordNet is to combine the usage of a dictionary and thesaurus which can support text analysis and the implementation of artificial intelligence applications [14] such as word sense disambiguation, automatic text summarization, text categorization, information retrieval etc.



TABLE 2.1: Examples of Semantic Relations in WordNet [15]

Semantic Relation	Syntactic Category	Examples
Synonymy (similar)	Noun Verb Adjective Adverb	pipe, tube rise, ascend sad, unhappy quickly, rapidly
Antonymy (opposite)	Noun Verb Adjective Adverb	high, low rise, descend sad, glad rapidly, slowly
Hyponymy (subordinate)	Noun	pine tree, conifer conifer, tree tree, plant
Meronymy (part of)	Noun	brim, hat gin, martini ship, fleet
Troponymy (manner)	Verb	march, walk whisper, speak
Entailment	Verb	drive, ride divorce, marry
Derivation	Adjective Adverb	magnetic, magnetism simply, simple

There are a few terms used to describe the concept of WordNet:

- **Synsets** – sets of cognitive synonyms which is the smallest unit in WordNet that represents specific meaning or concept of a word. It is interconnected through lexical relations.
- **Sense** – specific meaning of one particular word which has one type of POS. It is allocated to different synset.
- **Gloss** – further defines the concept of the word it represents.

## 2.3 Current development of mobile application

### 2.3.1 Android based mobile phone application

Android applications have gain increasing popularity in today's large community. It has attracted many developers to write applications that extend the functionality of the devices. Google bought over a small company called Android Inc and now Android operating system is developed and maintained by Google. It is installed for mobile devices which does not limit to only tablets as well as Smartphone. It is based on Linux kernel [16].

The development of Android is applied within Eclipse IDE, tested on Android emulator and deployed on an Android mobile as shown in *Figure 2.1*. The reason Android is chosen in comparable with iOS is its open source development kit and its consumer-driven applications capabilities. Since Google has built Android as an open standard device, anyone can develop their own applications and enhance them in which its major attractiveness is its customizable features.

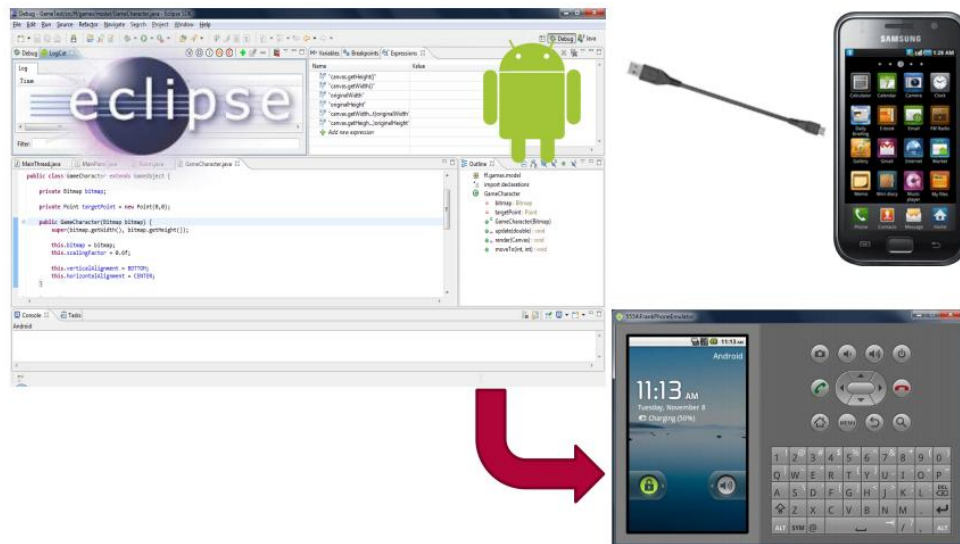


FIGURE 2.1: Development of Android application

The current distribution of the Android versions can be seen in Figure 1. According to [17], the table shows the percentage of Android devices used to access Google Play. It can be seen that Android Gingerbread API 10 (version 2.3.3-2.3.7) has the most usage yielding more than 50% of the Android users and Android Ice Cream Sandwich API 15 (version 4.0.3-4.0.4) comes second. Thus, the application in this

study will be developed on an Android Gingerbread version 2.3.3 to accommodate to the vast amount of users within this platform. Nonetheless, other versions higher than 2.3.3 will also be able to run the application with proper development.

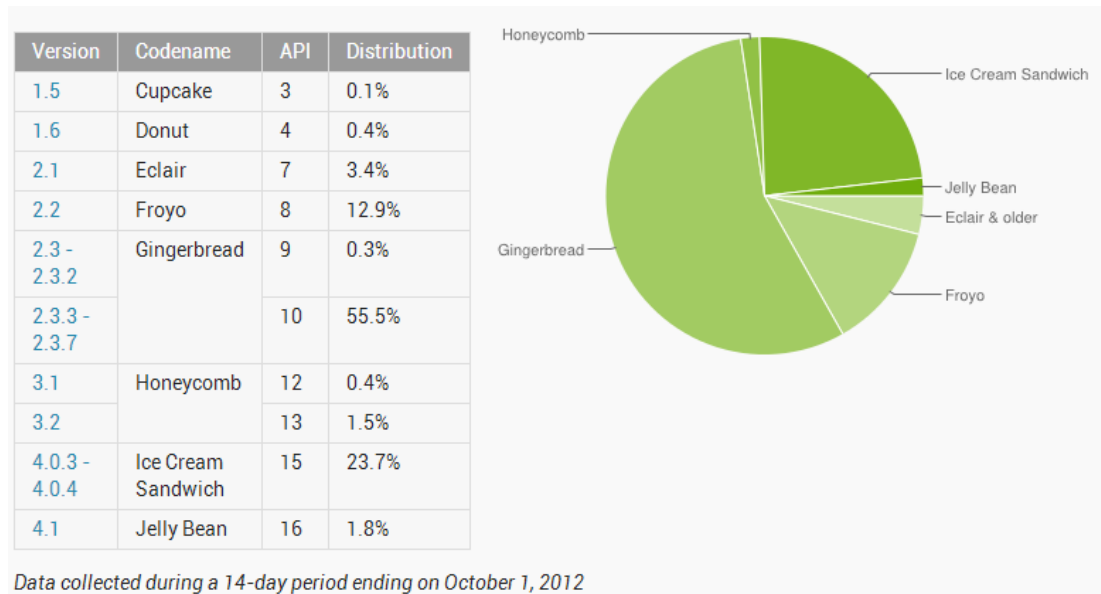


FIGURE 2.2: Android versions distribution [17]

### 2.3.2 Challenges of text summary on mobile applications

A major concern when developing applications for mobile is the small screen size. The small screen size of mobile devices has affect users in many ways: text reading rate on small screen and comprehension rate [18]. Therefore, the challenges in developing text summarization on mobile platform include a better visual display of text on smaller screens, appropriate size of content to be generated, and the overall design of text summarization application.

Generally, the specific features to be taken into account are as below:

- Small screen size
- CPU speed and memory are limited
- Limited content to be displayed
- Keyboards i.e. touch screen have limited space
- Overall user interface design

## CHAPTER 3

### METHODOLOGY

In this chapter, the project requirements, system design and implementation of system using available tools and algorithms will be further explained. This chapter constitutes the following details:

- Software development methodology
- Project activities – Planning, Analysis, Design and Implementation phases
- Tools

#### 3.1 Software Development Methodology

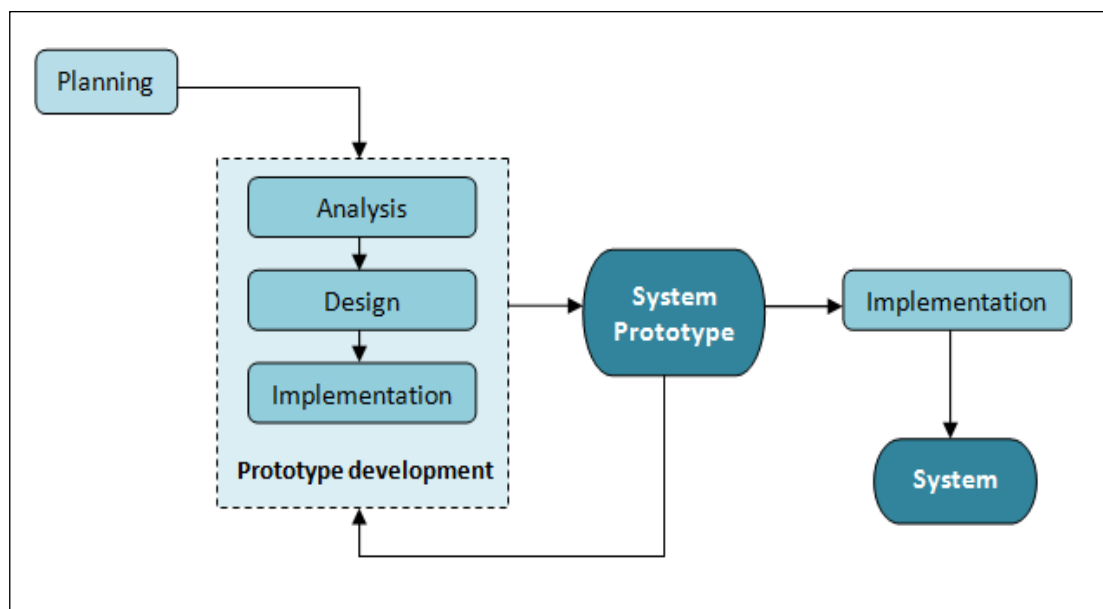


FIGURE 3.1: Rapid Application Development (RAD) methodology [19]

In this research, Rapid Application Development (RAD) methodology is chosen as the development process for the text summarizer, as shown in *Figure 3.1*. This methodology is selected as it avoid longer period of planning stage which allows system to be built quickly and concurrently during development stages.

The application system will be constantly refined and developed, making it easier to identify the most significant and useful features for implementation. In addition, as the prototype development stage iterate, it allows different features to be added and tested which is crucial for implementation of the prototype due to the limitations of time and resources. The development will continue until a final prototype is built before the real implementation is carried on to produce the real system. This will ensure that the final outcome of the system is well suited to users' requirements and functions.

## **3.2 Project Activities**

### **3.2.1. Planning Phase**

During this phase, fundamental problems are accessed and primary processes are executed. It mainly consists of two processes namely, project initiation and project management:

#### a) Project Initiation

- Review previously accomplished research done by other researchers.
- Identify the challenges in incorporating text summarization into Android application.
- Review on the existing application/system available in the market.
- Construct strategies for application development.
- Acquire list of development tools for application.
- Define system requirements for implementation.

#### b) Project Management

- Organize project activities using work breakdown structure.
- Construct project Gantt chart and key milestone.

The requirements of the system in this project are based on an existing model of summarization tool. The main focal point revolves around working algorithm and auto-generated summary in comparable with human-generated summaries as well as obtaining acceptable evaluation method. The planning stage has derived some important information for the next development phase: to obtain the gist of a text document so that readers save time in reading and comprehending the text, allowing more allowance of time for other readings and research.

### **3.2.2. Prototype Development Phase**

In the prototyping development phase, a running prototype model will be built for demonstration purposes. The model will be presented to relevant users and project supervisor for further feedbacks and evaluations. These constructive comments will then be used to reanalyze, redesign and re-implement on the next prototype model with additional features suggested. The iterative process of the prototype development will continue until the users, developer and supervisor agree on the implemented functions in the application.

The prototype development phase is described in more details as follows:

#### **a) Analysis phase**

The tasks to be performed at this phase include the design of the system model, the interface of the system as well as the algorithms used for implementation. The two approaches, semantic and syntactic relations to be implemented in this research are analyzed and apprehended. Relevant information gathering is vital at this stage to fully assimilate the research application. Tools for text analysis are widely available on the Internet. There are many open source software useful for the implementation of the system, e.g., WordNet could be helpful in providing words bank and assist in dealing with sentence cohesiveness.

Apart from that, a brief comparison is done on similar summarization application that is available in the current market. *Refer Appendix B* for the details of the application. Advantages and disadvantages of the application

are also being accessed and several information gathering has been done to understand the functionality of the application.

After gathering sufficient information on the current market applications, requirements for the prototype TextSumIt are then carefully accessed and defined in accordance to the application requirements. For the requirement gathering, it can be divided into two key aspects i.e. functional or non-functional requirements as shown in *Table 3.1* to *Table 3.6*.

TABLE 3.1: Functional Requirement 1

Function	Func1: Ability to summarize text document
Area	Functional (For User)
Description	The application should be able to summarize a single text file document (.txt format) into a shortened version of text with identified keywords.

TABLE 3.2: Functional Requirement 2

Function	Func2: Ability to save and view summarized documents
Area	Functional (For User)
Description	The application should allow users to save summaries of the documents that are summarized. It should also allow users to view saved documents in the repository.

TABLE 3.3: Functional Requirement 3

Function	Func3: Ability to adjust percentage of output of summary
Area	Functional (For User)
Description	Users can maneuver to adjust the percentage output of the summaries i.e. compression rate.

TABLE 3.4: Functional Requirement 4

Function	Func4: Ability to browse files from phone directory
Area	Functional (For User)
Description	The application should be able to display a list of directory from the phone memory and sdcard which allows users to browse the files to be selected for summarization.

TABLE 3.5: Functional Requirement 5

Function	Nav1: Automatically adjust the orientation view of the phone
Area	Graphical User Interface / Navigation (For User)
Description	The application should adjust the screen view to suit both the orientation i.e. landscape and portrait view. Text size or content should adjust accordingly to display appropriate information in correct position.

TABLE 3.6: Non-Functional Requirement 1

Function	Func5: Ability to perform and response quickly
Area	Non-functional
Description	The application should have a short response time for a simple execution of the summarization and a high throughput (rate of processing work) while maintaining the quality of the output.

**b) Design phase**

i. System architecture

The proposed model of the system architecture is shown in *Figure 3.2*. The processes and relations among them are portrayed in the following diagram.

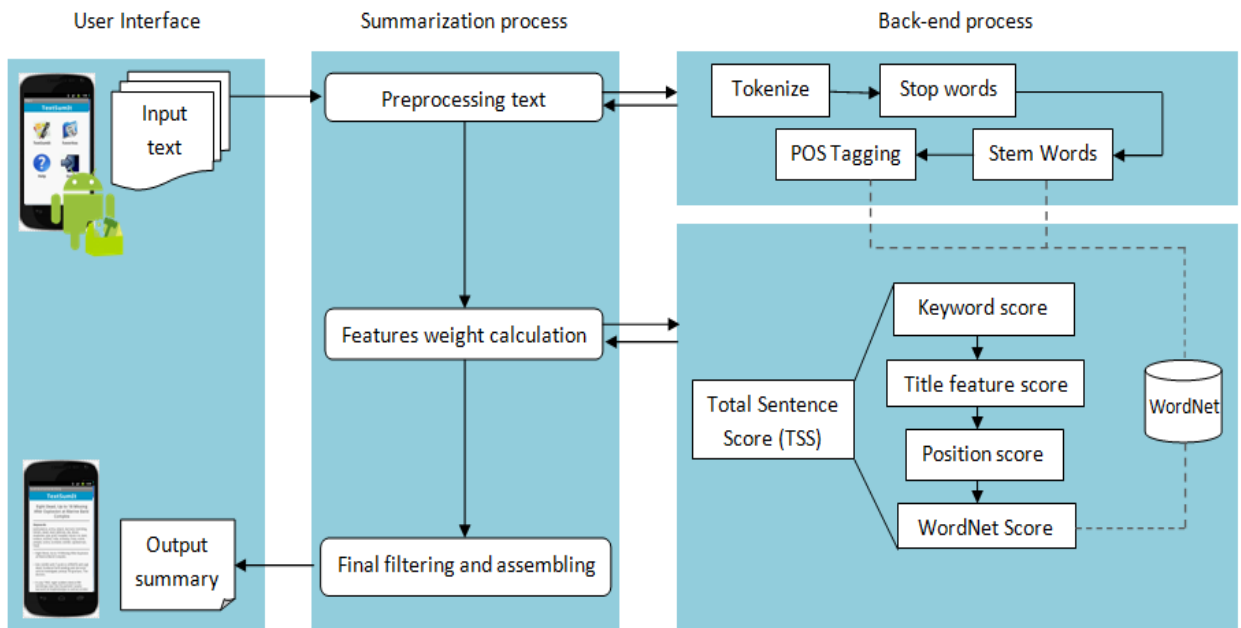


FIGURE 3.2: Proposed Model of Automatic Text Summarizer



The proposed model focuses on both the preprocessing phase and the features weight calculation. Users will input source text documents that they want to summarize into the user interface. During the **preprocessing text stage**, the text document undergoes a process of tokenization, stop words removal, WordNet stemming and finally Part-of-Speech (POS) tagging. The preprocessing is done to make it easier for the process of learning algorithm.

The next stage, **features weight calculation** involves the calculation of the Total Sentence Score (TSS) which consists of Keyword Score, Title feature score, Position Score, and WordNet Score. These features weight will determine the rank of the whole sentences from the source text. The sentence with the highest weight will be selected for the summary output. The overall summary output depends on the features weight which is enhanced with the semantic extraction algorithm using WordNet.

The last stage of the text summarization process is **final filtering and assembling**. At this stage, undefined references of sentences are being filtered, i.e., sentences with words such as “He” or “They” at the beginning of the sentence and quoted sentences will be removed. Finally, all the filtered sentences are assembled to form a complete summary to be output to users.

#### ii. Graphical User Interface (GUI)

*Figure 3.3* to *Figure 3.6* are the interfaces used for users’ navigation within the TextSumIt application on Android. *Figure 3.3* shows the main screen of TextSumIt application which consists of four simple navigation: TextSumIt (for summarization), Favorites (to view saved documents), Help (for help in using the application), and Quit (to quit the application).

*Figure 3.4* is the results of the text summarization whereby content of documents are summarized into several key points to make it easier for reading. Top 20% of keywords are also extracted from the text to give users a better idea of the summarized text. Users may save the summary for future viewing. *Figure 3.5* is the screen displaying all the saved summaries which users can select for viewing.



FIGURE 3.3: Main Homepage of TextSumIt



FIGURE 3.4: Text Input Screen



FIGURE 3.5: Summary Output Screen

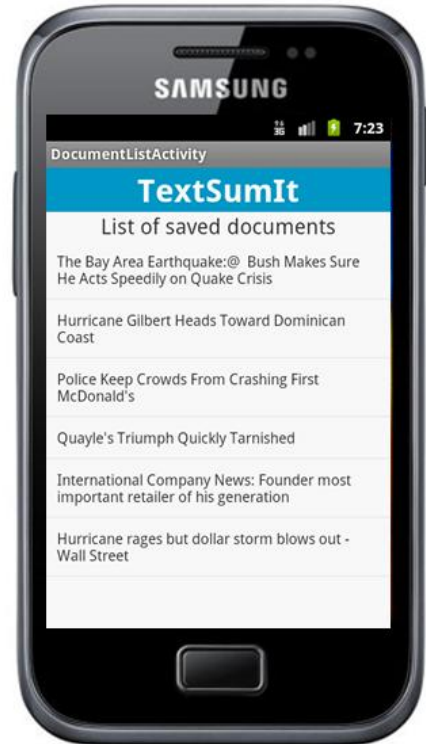


FIGURE 3.6: Saved Documents Screen

### c) Implementation phase

#### i. Preprocessing text

Source text which is input from users is being accessed whereby documents are being tokenized into sentences, then individual words. Words from the input source text are usually stored in an array to be compared with a list of predetermined stop words and these are stored in a text file. Stop words referred to similar words and meaningless words such as “a”, “an”, “is”, “the” etc. Words from the source text that appear in the stored array list will be removed before being split into its individual root words using the stemming process, e.g., running >> run. Stemming of words refers to the process of extraction of root words, i.e., base words of which is essential to discard any suffixes and prefixes [20] .

Next, a Part-of-Speech (POS) tag is produced as an annotation to each of the word (i.e. nouns, adjectives, adverbs etc). In this study, *Apache OpenNLP* tool is used to mark the tokenized words with their corresponding word type based on the token itself as well as the context of the token. Both stemming and POS tags processes use WordNet as the data dictionary to execute its process.

#### ii. Total Sentence Score (TSS)

Another significant enhancement to the summaries is focused on the features weight calculation whereby a final total sentence score (TSS) is calculated for each sentence using Eq. 1.

$$TSS(S_i) = w_1KW(S_i) + w_2Title(S_i) + w_3Pos(S_i) + w_4WN(S_i) \quad (1)$$

Where TSS is the Total Sentence Score, KW represents the keyword score, Pos represent the position score and WN is the WordNet score.  $w_1, w_2, w_3, w_4$  are the weights for each score which will be sum up to TSS. The top sentences will be selected as the output summary according to the compression rate set by users.

### iii. Keyword Score

Keyword score is based on the Term Frequency (TF) and Inverse Sentence Frequency (ISF) algorithm by Joel Larocca (2002) [21]. It is adapted from the common TF-IDF (Term Frequency-Inverse Document Frequency) measures for information retrieval of multiple documents. TF-ISF is computed for each word in the sentences and a sentence with higher values of TF-ISF has a higher probability to be selected for the output summary. It is computed as shown in Eq. 2.

$$TF - ISF(w, s) = TF(w, s) * ISF(w) \quad (2)$$

Where the term frequency  $TF(w, s)$  is the number of times the word  $w$  occurs in sentence  $s$ , and inverse sentence frequency  $ISF(w)$  is computed in Eq. 3.

$$ISF(w) = \log\left(\frac{S}{SF(w)}\right) \quad (3)$$

Where sentence frequency,  $SF(w)$  is compute as the number of sentences in which the word  $w$  occurs.  $S$  is the total number of sentences in the document.

### iv. Title Feature Score

This algorithm emphasizes on the sentences with keywords that present in the title which resemble the theme of the document and are of greater chances to be included in the summary. The resemblance of the sentence can be calculated using Eq. 4 [22]:

$$Title(S_i) = \frac{Keywords\ in\ S_i \cap Keywords\ in\ Title}{Keywords\ in\ S_i \cup Keywords\ in\ Title} \quad (4)$$

#### v. Position Score

The position score in this study comprises of two models. The *first position model* scores the sentences based on its similarity to the first and last sentence of the document at average. The average score is calculated as Eq. 5, 6, 7:

$$P_1(S_i) = \frac{\text{Keywords in } S_i \cap \text{Keywords in } S_i}{\text{Keywords in } S_i \cup \text{Keywords in } S_i} \quad (5)$$

$$P_{Last}(s_i) = \frac{\text{Keywords in } s_i \cap \text{Keywords in } S_{Last}}{\text{Keywords in } s_i \cup \text{Keywords in } S_{Last}} \quad (6)$$

$$Pos(s_i) = \frac{P_1(s_i) + P_{Last}(s_i)}{2} \quad (7)$$

The *second position model* is based on the algorithm introduced by Barrera and Verma (2012). The model assumes the sentences towards the beginning and ending of a document are deemed important and more likely to be selected in the summary. This position score can be achieved using Eq. 8 [23]:

$$P_{cos}(s_i) = \frac{\cos \frac{2\pi x}{k-1} + \alpha - 1}{\alpha} \quad (8)$$

Where  $\alpha$  refers to the dent factor in which  $\alpha=2$  is used in this algorithm,  $k$  represents total sentences,  $x$  as the position of sentence  $S_i$  in the document.  $P_{cos}$  will be equally distributed as  $\alpha$  increases and on the other hand when  $\alpha$  decreases,  $P_{cos}$  is more concentrated to the value one in the beginning and end of the document [23]. The first sentence has value  $x=0$  and last sentence has value  $x=k-1$ .

#### vi. WordNet Score

The algorithm implemented here is based on the use of WordNet whereby the approach to WordNet model in this study focuses on keyword semantics extraction as illustrated in [23]. The model's objective is to select the sentences which contain words that have the closest meaning to the keywords extracted. This not only extracts the keywords but also provide a semantic analysis by integrating WordNet into the system. The whole algorithm is

associated with a collection of similar words of which Barrera and Verma described it as *thematic word list* [23]. For each word in a sentence,  $S_i$  it will be ranked according to their relevancy to the text document using WordNet's sets of synonyms called *synsets*. The following Eq. 9 is the score allocated for each word  $w$ :

$$score(w) = \frac{1}{2^l} \quad (9)$$

Where  $l$  refers to the minimum level determined when the word  $w$  is compared with the *synsets*. If the word  $w$  is found in the sets of synonyms, then  $score(w) = 1$  whereby  $l=0$ . If it is not found in this level, then  $l$  will be increase by one ( $l=1$ ) which means that the word is compared to the words in the preceding level of *synsets*. It will increase up to a maximum level of 4. Generally, the overall equation for WordNet score is shown in the following Eq. 10:

$$WN_{syn}(s_i) = \sum_{w \in S_i} score_{syn}(w) \quad (10)$$

Hence, the closer the word  $w$  in a sentence,  $S_i$  to the *synsets*, the higher the WordNet score.

### 3.2.3. Implementation Phase

At the implementation phase, a complete system would be built according to the latest agreed prototype with all the intended features and requirements implemented. Testing which is the most critical stage plays a significant role in ensuring accuracy of the output system and the user evaluation on the application. The built system will be tested to ensure it runs and performs as expected. User testing will be done to improve on its GUI implementation.

### **3.3 Tools**

The following are the tools required for the system development:

- a) Hardware
  - i. Android mobile
  - ii. Personal Computer
- b) Software
  - i. Android 2.3 (Gingerbread) platform
  - ii. Java plug-ins for Android
  - iii. Eclipse IDE
  - iv. WordNet
  - v. ROUGE tools

## CHAPTER 4

### RESULTS AND DISCUSSION

This chapter covers the results of the research and discussion on the implementations as well as the development of prototype. This chapter is categorized into 3 sections: preprocessing enhancement, summary evaluation, and user testing.

#### 4.1 Preprocessing Enhancement

The prototype has implemented a new method of stemming using the WordNet as the lexical database to correctly stem each word to its root. Previous researcher has utilize a simple stemmer called the PorterStemmer [20] which does not gives accurate stemming results, in other words incorrectly return the root words.

On the other hand, WordNet stemmer has shown assuring results as compared to the simple stemmer. Part of the stemming results is recorded in Table 4.1 comparing the stemmed words using PorterStemmer and WordNet stemmer. It can be seen that stemming using WordNet as the data dictionary yield much better results than the PorterStemmer as the root words are properly retrieved because the tokens to be stemmed run through the dictionary (WordNet) to get the base word.

The next stage of preprocessing is assigning Part-Of-Speech (POS) tag to each of the remaining root words. Here, a pre-trained POS tagging model tools from OpenNLP [24] is used. Generally, the POS tagger marks each token with corresponding syntax which includes nouns, verbs, adjectives and adverbs. It is tagged based on the token and its context. The POS tagger tool implements a probability model in order to predict the correct pos tag from the tag set. A sample of pos tagging on a text document is shown in Table 4.2. *Refer Appendix C* for the prefixes of POS Tag [25].



TABLE 4.1: Comparison of results between PorterStemmer and WordNet Stemmer

Original words	Porter Stemmer	Positive	Negative	WordNet Stemmer	Positive	Negative
billion	billion	√		billion	√	
quake	quak		X	quake	√	
industry	industri		X	industry	√	
observers	observ		X	observer	√	
expect	expect	√		expect	√	
company	compani		X	company	√	
suffer	suffer	√		suffer	√	
financial	financi		X	financial	√	
damage	damag		X	damage	√	
public	public	√		public	√	
affairs	affair	√		affair	√	
American	american	√		american	√	
insurance	insur		X	insurance	√	
benefit	benefit	√		benefit	√	
analysts	analyst	√		analyst	√	
predicted	predict	√		predict	√	
reverse	revers		X	reverse	√	
years	year	√		year	√	
declining	declin		X	decline	√	
rates	rate	√		rate	√	
regulators	regul		X	regulator	√	
estimated	estim		X	estimate	√	
hammered	hammer	√		hammer	√	
southeastern	southeastern	√		southeastern	√	
translate	translat		X	translate	√	

\* DUC\_D062\_V2.txt is used as the source document.

TABLE 4.2: Sample POS Tagging results

Sample article: DUC_D061.txt
hurricane_NN, gilbert_NN, head_NN, dominican_JJ, coast_NN, sweep_NN, dominican_JJ, republic_NN, sunday_NN, civil_JJ, defense_NN, alert_JJ, heavily_RB, populate_JJ, south_JJ, coast_NN, prepare_VBP, high_JJ, wind_NN, heavy_JJ, rain_NN, high_JJ, sea_NN, storm_NN, southeast_JJ, sustain_VBP, wind_NN, mph_NN, gust_NN, mph_NN, civil_JJ, defense_NN, director_NN, eugenio_NN, cabral_NN, television_NN, alert_NN, shortly_RB, midnight_NN, saturday_NN, cabral_NN, resident_NN, province_NN, barahona_NN, closely_RB, follow_VBP, movement_NN, estimate_NN, people_NNS, live_JJ, province_NN, include_VBP, city_NN, barahona_NN, mile_NN, west_RB, santo_VBD, domingo_JJ, tropical_NN, storm_NN, form_NN, eastern_JJ

## 4.2 Summary Evaluation

In this research, the proposed algorithm is tested on a Java application for the evaluation of the system summary output. It is then tested on Android Java platform for its GUI implementation and user testing evaluation.

### 4.2.1 Performance Measure

The final output summaries are evaluated using Recall-Oriented Understudy for Gisting Evaluation (ROUGE) which is based on [26]. It automatically evaluates the quality of summaries using measures such as n-gram, word-pairs as well as the word sequences by comparing the machine-generated summaries with human-generated summaries. The summarized text will be evaluated using the f-measure according to the recall and precision scores based on Eq. 11, 12 and 13. More specifically, ROUGE-N (N-gram Co-Occurrence Statistics) is chosen as the evaluation function in this research.

$$R = \frac{|\{\text{Relevant Sentences}\} \cap \{\text{Retrieved Sentences}\}|}{|\{\text{Relevant Sentences}\}|} \quad (11)$$

$$P = \frac{|\{\text{Relevant Sentences}\} \cap \{\text{Retrieved Sentences}\}|}{|\{\text{Retrieved Sentences}\}|} \quad (12)$$

$$F - score = \frac{(1 + \beta^2)RP}{R + \beta^2 P} \quad (13)$$

Recall, R measures the percentage of relevant instances that are being retrieved whereas Precision, P on the other hand, measures the percentage of retrieved instances that are relevant. In short, a high recall indicates that most of the relevant results are returned and a high precision indicates that more relevant results are returned than irrelevant ones.

In this case, it is tested on the information retrieval of text summarization; recall is the number of correct results divided by the number of results that should have been returned [27]. Computing recall alone is insufficient as irrelevant results should be computed as well and thus precision is calculated for all retrieved results as shown in Eq. 11.

Finally, after computing recall and precision, F-score which measure the accuracy of the results is computed from both the equations. As shown in Eq.12, F-score is a weighted average of both recall and precision.

#### **4.2.2 Existing Text Summarizer and Reference Summaries**

In order to evaluate the effectiveness of the Automatic Text Summarizer, the aforementioned measures are used to rate the quality of the output system summary against the existing system, namely Microsoft Word AutoSummarize. Word AutoSummarize will identify key points from the text document, usually works better on well structured articles or reports. The system summarizer TextSumIt is compared to Word AutoSummarize as both has similar algorithm behind the text summarization process.

In Word AutoSummarize, each sentence in the document are assigned with a sentence score and then display the highest scoring sentences according to the percentage of output determined by users [28]. The scoring is computed based on the frequency of words in a sentence that occur in the document. In other words, sentences with words used frequently in the document have a higher score.

Similarly, for TextSumIt, the sentence selection is also based on the features weight calculation whereby more features are taken into consideration to increase the scoring system. The additional features implemented are keyword score, title feature score, position score and WordNet score. The difference would be the additional semantic analysis of the words in the document before it is extracted as implemented under the WordNet score.

As for the reference summaries, text corpus from Document Understudy Conferences (DUC) is taken for the evaluation of both summarizers. More specifically, selected articles from DUC 2002 are taken as the sample input text for summarization.

### 4.2.3 Evaluation of Experimental Results

ROUGE evaluation tool developed and maintained by Chin-Yew Lin (2004) has been utilized to automate the evaluation process. Basically, ROUGE will compare two generated output summaries produced by different applications (in this case TextSumIt and Word AutoSummarize) in comparison with the reference summaries from DUC. The final results are recall, precision and F-score of both the applications.

The evaluation is done on a selected number of articles from DUC 2002. Thirty articles are selected as an input for the system summary, TextSumIt and Word AutoSummarize. The summarization on TextSumIt and Word AutoSummarize is done with the closest length possible to yield approximate similar results. In this research, ROUGE-1 is used as the evaluation method for the summaries.

Before the evaluation can be done using ROUGE, the output summary is compiled into corresponding file format as a system input for ROUGE tools. On the other hand, the reference summaries which are from DUC 2002 are referred as “Gold” standards summary by the ROUGE tool. The same procedures are done with both TextSumIt and Word Autosummarize. The output scores are then tabularized and shown in graphs to compare the recall, precision and F-score between both the system as shown in Table 4.3 and a graphical representation is shown in Figure 4.1.

The overall evaluation of performance of the summarization system will involved the following information:

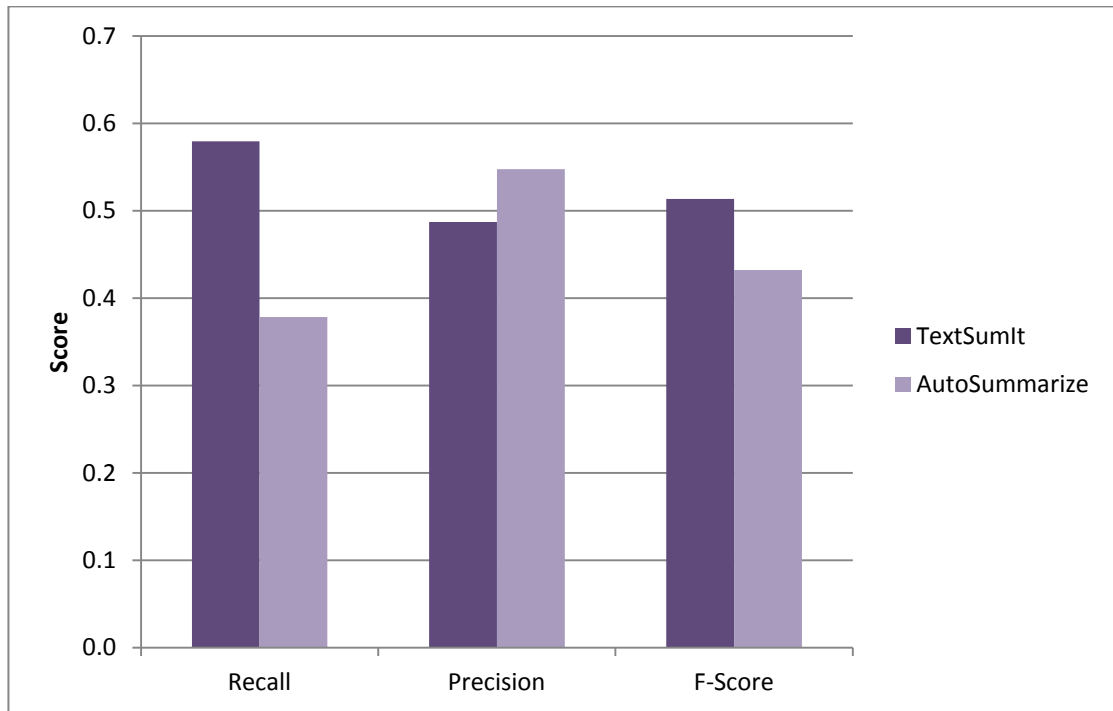
- i. Gold standards reference summaries (DUC 2002)
- ii. TextSumIt system output summaries
- iii. Word AutoSummarize system output summaries
- iv. ROUGE-1 scores of each output summaries for both systems.

**i. Tabular Data**

TABLE 4.3: The recall, precision and F-score for TextSumIt and Word AutoSummarize using DUC 2002 articles

No.	Documents (DUC 2002)	TextSumIt			Word AutoSummarize		
		Recall	Precision	F-Score	Recall	Precision	F-Score
1	DUC_D062_v2	0.80909	0.27726	0.41299	0.52727	0.42963	0.47347
2	DUC_D062j_0021	0.50485	0.45217	0.47706	0.28155	0.76316	0.41134
3	DUC_D063	0.54639	0.34194	0.42064	0.36082	0.42169	0.38889
4	DUC_D064_0193	0.64815	0.34826	0.45308	0.59259	0.48120	0.53112
5	DUC_D068_0172	0.61111	0.46809	0.53012	0.41667	0.56962	0.48129
6	DUC_D069_0064	0.56566	0.54902	0.55722	0.37374	0.54412	0.44312
7	DUC_D070_0077	0.58491	0.74699	0.65609	0.25472	0.45763	0.32728
8	DUC_D076_AP0072	0.72897	0.38806	0.50649	0.55140	0.53153	0.54128
9	DUC_D077_AP0204	0.66071	0.40000	0.49832	0.52679	0.60825	0.56460
10	DUC_D080_AP0193	0.53846	0.33333	0.41176	0.55769	0.52252	0.53953
11	DUC_D081_AP0135	0.57843	0.64835	0.61140	0.29412	0.75000	0.42254
12	DUC_D082_AP0157	0.70103	0.32536	0.44445	0.41237	0.30075	0.34782
13	DUC_D083_AP0199	0.62136	0.35556	0.45230	0.40777	0.33600	0.36842
14	DUC_D085_AP0054	0.52778	0.51818	0.52294	0.43519	0.74603	0.54971
15	DUC_D086_AP0004	0.57143	0.53333	0.55172	0.30357	0.52308	0.38418
16	DUC_D087_AP0013	0.48421	0.35938	0.41256	0.45263	0.55128	0.49711
17	DUC_D087_AP0042	0.69388	0.57627	0.62963	0.29592	0.64444	0.40560
18	DUC_D087_AP0097	0.54206	0.39726	0.45850	0.40187	0.50588	0.44792
19	DUC_D090_AP0142	0.52000	0.40000	0.45217	0.37000	0.56923	0.44848
20	DUC_D092_AP0025	0.65421	0.69307	0.67308	0.35514	0.60317	0.44706
21	DUC_D092_AP0186	0.68807	0.59524	0.63830	0.41284	0.63380	0.50000
22	DUC_D093_AP0007	0.46465	0.67647	0.55090	0.25253	0.60976	0.35715
23	DUC_D093_AP0051	0.45631	0.58750	0.51366	0.41748	0.69355	0.52122
24	DUC_D093_AP0076	0.48077	0.55556	0.51547	0.27885	0.72500	0.40278
25	DUC_D094_AP0071	0.56311	0.46032	0.50655	0.44660	0.54762	0.49198
26	DUC_D094_AP0104	0.39604	0.53333	0.45454	0.22772	0.46000	0.30463
27	DUC_D094_AP0157	0.51485	0.40625	0.45415	0.36634	0.50685	0.42529
28	DUC_D095_AP0004	0.54639	0.45299	0.49533	0.25773	0.37879	0.30675
29	DUC_D095_AP0013	0.66972	0.42442	0.51957	0.35780	0.39796	0.37681
30	DUC_D095_AP0160	0.50505	0.80645	0.62112	0.16162	0.61538	0.25600
<b>Average scores:</b>		0.57926	0.48701	0.51340	0.37838	0.54760	0.43211

## ii. Graphical Representation



**FIGURE 4.1: The average recall, precision and F-score graph for TextSumIt and Word AutoSummarize using articles from DUC 2002**

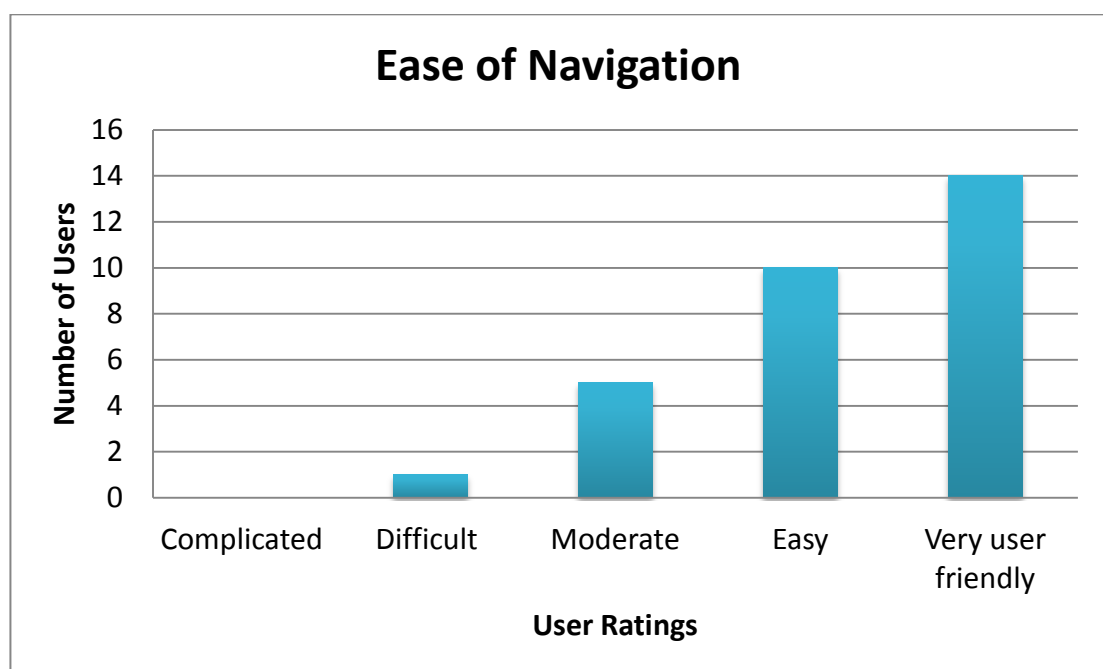
From the tabular results as shown in Table 4.3 and Table 4.4, the overall average recall, precision and F-Score indicates that TextSumIt gives a better output as compared to Word AutoSummarize. Both summaries were compared to the reference summaries which are human-generated by experts. For certain articles, the precision score of Word AutoSummarize outperform the precision score of TextSumIt. This could mean that there are more relevant sentences being retrieved by the Word AutoSummarize than irrelevant sentences. However, TextSumIt on average outperform Word AutoSummarize on its recall score which means that most of the relevant sentences are retrieved.

Overall, TextSumIt still has a higher F-score of 0.51340 as compared to Word AutoSummarize which has a lower F-Score of 0.43211. The significant difference in the results is due to the different algorithm used in the sentence extraction. TextSumIt which uses a semantic based algorithm is able to extract sentences more precisely as compared to Word AutoSummarize which only uses frequency-based algorithm. The current results could be considered satisfactory.

### 4.3 User Testing

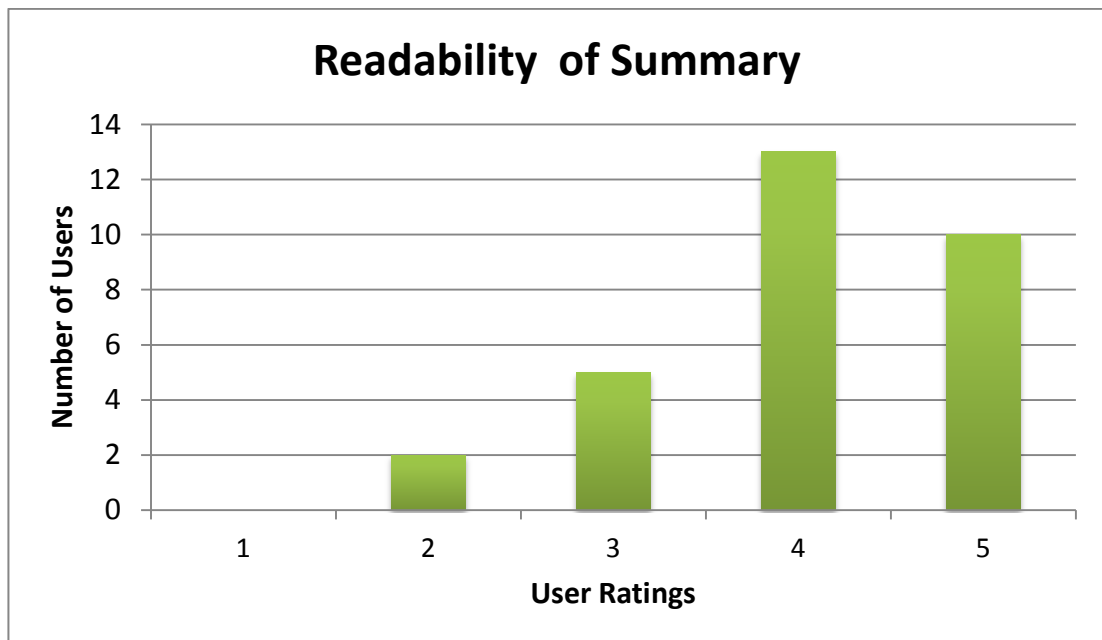
The user testing is done mainly for the evaluation of the TextSumIt application using a subjective evaluation by the users. The main features being tested on the Android application is the **ease of navigation, readability** and **time saved**. The testing targeted 30 users, mainly students who used to read lengthy documents like research papers, journals or articles and uses Android mobile devices. Targeted users are provided with the application setup on their mobile with sample text provided in the package. Users were given a questionnaire for them to respond and evaluate after testing the application. *Refer Appendix D for the sample questionnaire.*

The first part of the testing is to test the ease of navigation on the application. Users are expected to navigate through the application screens and rate appropriately according to their experiences in using it. Next, users are required to test on the summarization system and rate on the readability of the summary output. The results of the user testing for the two features; ease of navigation and readability are shown in Figure 4.2 and Figure 4.3. They were then asked if the application does help to save their time in reading lengthy text. The results of their evaluation are shown in Figure 4.4.



**FIGURE 4.2: Evaluation Results for Ease of Navigation of Application**

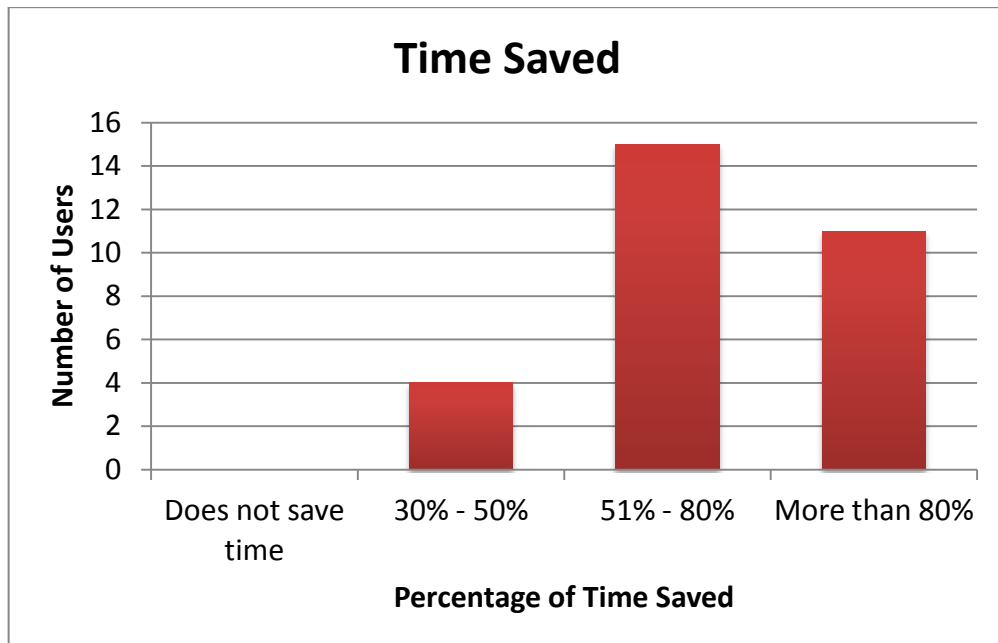
Generally, 45% of the users rated that the application is very user friendly and 35% of them said that it is easy to navigate while only a minority of them finds it difficult to maneuver through the application. This might be due to the possibility of not understanding the flow of the application itself. It might also be due to the back or menu navigation function in which different users operate it differently. However, none of the user finds it complicated to navigate through.



**FIGURE 4.3: Evaluation Results for Readability of Summary**

The readability of summary is evaluated on a scale of 1 to 5 whereby 1 is considered the least readable and 5 being highly readable. According to the graph as shown in Figure 4.5, eight out of 20 users rated 4 for the readability of the summary while 6 of them rated a 5 for its readability. Only two users find the summary less readable from the application which could be due to the output appearance of the system.





**FIGURE 4.4: Evaluation Results for the Percentage of Time Saved by Using the Application**

It is found that 12 out of the 20 users agreed that the application is able to save up 51%-80% of their reading time, 5 of them thinks that it can saves more than 80% of their time and only 3 feels that it saves only 30%-50% of their time. Additionally, there are no users who think that the application does not save any of their time which means that the text summarizer application is helpful in saving their reading time.

From the questionnaire returned by the users, there were several comments and feedbacks on the application. Generally, users like the idea of text summarization on the Android mobile device as it is able to provide them the gist of document quickly and briefly. Some of them find it handy especially when they want to assimilate lengthy documents when they are busy. There are several comments that mention the readability of the system can be enhanced with improved font appearance and size as well as the structure of the output. Overall, the response is positive and majority of them will utilize the application either for personal use or at workplace.

## **CHAPTER 5**

### **CONCLUSION AND RECOMMENDATIONS**

#### **5.1 Conclusion**

Automatic Text Summarizer (ATS) is a challenging field as there are several concerns on the output of summaries, especially the quality of summarization. Most of the researches focus on using the statistical approach in text summarization. While the demand today has shift the research towards Natural Language Processing (NLP), there are generally more in depth research to be done in this particular area in order to improve the accuracy of summary output. Although there are already many available text summarizers in the market today, researchers' ultimate findings would be getting a better algorithm to generate higher accuracy of summary.

In this research, a semantic based Automatic Text Summarizer for a single document on Android mobile device has been implemented. Semantic extraction using WordNet is used to enhance the summarization process to give a better outcome of summaries. It has enhanced the preprocessing of text in terms of its stemming algorithm and additional part-of-speech tagging. An improved features weight calculation which computes the total sentence score are also implemented to give a higher accuracy in selecting sentences for output summaries. The only limitation of the current application is the types of input document which only accepts a .txt file format. It can only generate summaries from a text-based document which means it omits documents with images, diagrams, graphs etc.

In a nutshell, the research meets the objective of the research which incorporates semantic and syntactic analysis into text summarization and deployed it on an Android mobile device that gives better readability to users.

## 5.2 Recommendations

The technology advancement today has led summarization field towards the mobile technology and it will be a great progress in the future. This research will be a stepping stone for future mobile development. However, there are several challenges to be taken into consideration as the development of application on mobile has few concerns on the output e.g. the small screen size and the limited processing memory of the mobile. Heavy text processing might be a challenge for the implementation on mobile itself. It is recommended to connect the application and run the back end processes on a server to reduce the heavy processing on the mobile device.

In addition, text classification can be extended to the mobile application whereby text documents that are summarized can be categorized into different themes according to the content of summaries. This provides a better operability of the application for users to search appropriate summaries in a better way. For example, text summaries can be categorized to several main themes such as Accidents, Earthquake, or Hurricane. Further research is needed to identify the types of classification to be implemented on such text documents.

The system application can be further extended to summarize text on the web. This allows users more flexibility in summarizing text and not bounded by only input documents. This may mean surfing the web on the mobile device and summarizing the text which connects to the server for processes. The types of text to be summarized depend on the users but the most common and basic summarization can be done on news website like Reuters.

The performance of the TextSumIt application can be further enhanced by implementing the Particle Swarm Optimization (PSO) [29] algorithm in order to increase the robustness of the system. Further research can be done that revolves around the Swarm Intelligence [30] as an approach for text summarization to increase the system's learning ability. In the course of implementing such algorithm it is assumed that the future back end processing be focused on the server rather than the system application itself as mentioned earlier.

## REFERENCES

- [1] I. Mani, "Recent Developments in Text Summarization," in *Proceedings of the tenth international conference on Information and knowledge management, CIKM*, New York, USA, 2001.
- [2] E. Hovy and C.-Y. Lin, "Automated Text Summarization and the SUMMARIST system," 1999.
- [3] O. Buyukkokten, H. Garcia-Molina and A. Paepcke, "Seeing the Whole in Parts: Text Summarization for Web Browsing on Handheld Devices," in *Proceedings of the 10th International Conference on World Wide Web*, New York, USA, 2001.
- [4] G. Chang, T. Chunguang, L. Guanhua and Z. Chuan, "Developing Mobile Applications on the Android Platform," in *Mobile Multimedia Processing Fundamentals, Methods, and Applications*, Springer Berlin / Heidelberg, 2010, pp. 264-286.
- [5] H. P. Luhn, "The Automatic Creation of Literature Abstracts," *IBM Journal*, pp. 159-165, 1958.
- [6] P. Baxendale, "Machine-made Index for Technical Literature - An Experiment," *IBM Journal of Research Development*, vol. II, no. 4, 1958.
- [7] M. S. Binwahlan, S. Naomie and L. Suanmali, "Fuzzy Swarm Diversity Hybrid Model for Text Summarization," *Information Processing and Management*, vol. 46, pp. 571-588, April 2010.
- [8] O. M. Foong, A. Oxley and S. Sulaiman, "Challenges and Trends of Automatic Text Summarization," *International Journal of Information and Telecommunication Technology*, vol. I, no. 1, pp. 34-39, 2010.
- [9] K. S. Jones, "Automatic Summarising: factors and directions," in *MIT Press*, England, UK, 1998.
- [10] V. Gupta, "A Survey of Text Summarization Extractive Techniques," *Journal of Emerging Technologies in Web Intelligence*, vol. II, no. 3, pp. 258-268, 2010.
- [11] S. Jusoh and H. M. Al Fawareh, "Semantic Extraction from Texts," in *Proceedings of International Conference on Computer Engineering and Applications IPCSIT*, Singapore, 2011.
- [12] R. K. Wong and C. Sia, "An Efficient WordNet-Based Summarizer for Large Text Documents," New South Wales, Australia, 2003.
- [13] Princeton University, "WordNet: A lexical database for English," 6 January 2012.

- [Online]. Available: <http://wordnet.princeton.edu/>. [Accessed 31 July 2012].
- [14] "WordNet," Wikipedia: the free encyclopedia, [Online]. Available: <http://en.wikipedia.org/wiki/WordNet>. [Accessed 1 August 2012].
- [15] C. Leacock, G. A. Miller and M. Chodorow, "Using corpus statistics and WordNet relations for sense identification," *Computational Linguistics - Special issue on word sense disambiguation*, vol. 24 , no. 1, pp. 147-165 , March 1998 .
- [16] F. Feinbube, "Android," 17 November 2011. [Online]. Available: <http://www.tuplespaces.net/teaching/EmbeddedOS/Slides2011>. [Accessed 22 June 2012].
- [17] A. Developers, "Platform Versions," Android , [Online]. Available: <http://developer.android.com/about/dashboards/index.html>. [Accessed 2 October 2012].
- [18] M. Jones, G. Marsden, N. Mohd Nasir, K. Boone and G. Buchanan, "Improving web interaction on small displays," *Computer Networks*, vol. 31, no. 11-16.
- [19] "Lessons from Software Development," DrTdaxp, [Online]. Available: <http://www.tdaxp.com/archive/2005/07/20/dreaming-5th-generation-war.html>. [Accessed 20 June 2012].
- [20] M. F. Porter, An algorithm for suffix stripping, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997.
- [21] J. Larocca Neto, A. A. Freitas and C. A. A. Kaestner, "Automatic Text Summarization Using a Machine Learning Approach," in *Advances in Artificial Intelligence*, Brazil, Springer Berlin Heidelberg, 2002, pp. pp 205-215.
- [22] O. M. Foong and A. Oxley, "A Hybrid PSO Model in Extractive Text Summarizer," in *IEEE Symposium on Computers & Informatics*, 2011.
- [23] A. Barrera and R. Verma, "Combining Syntax and Semantics for Automatic Extractive Single-Document Summarization," *Proceeding CICLing'12 Proceedings of the 13th international conference on Computational Linguistics and Intelligent Text Processing*, vol. Part II, pp. 366-377, 2012.
- [24] "Apache OpenNLP Developer Documentation," The Apache Software Foundation, [Online]. Available: <http://opennlp.sourceforge.net/models-1.5/>. [Accessed 10 November 2012].
- [25] M. Liberman, "Alphabetical list of part-of-speech tags used in the Penn Treebank Project," [Online]. Available: [http://www.ling.upenn.edu/courses/Fall\\_2003/ling001/penn\\_treebank\\_pos.html](http://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html).

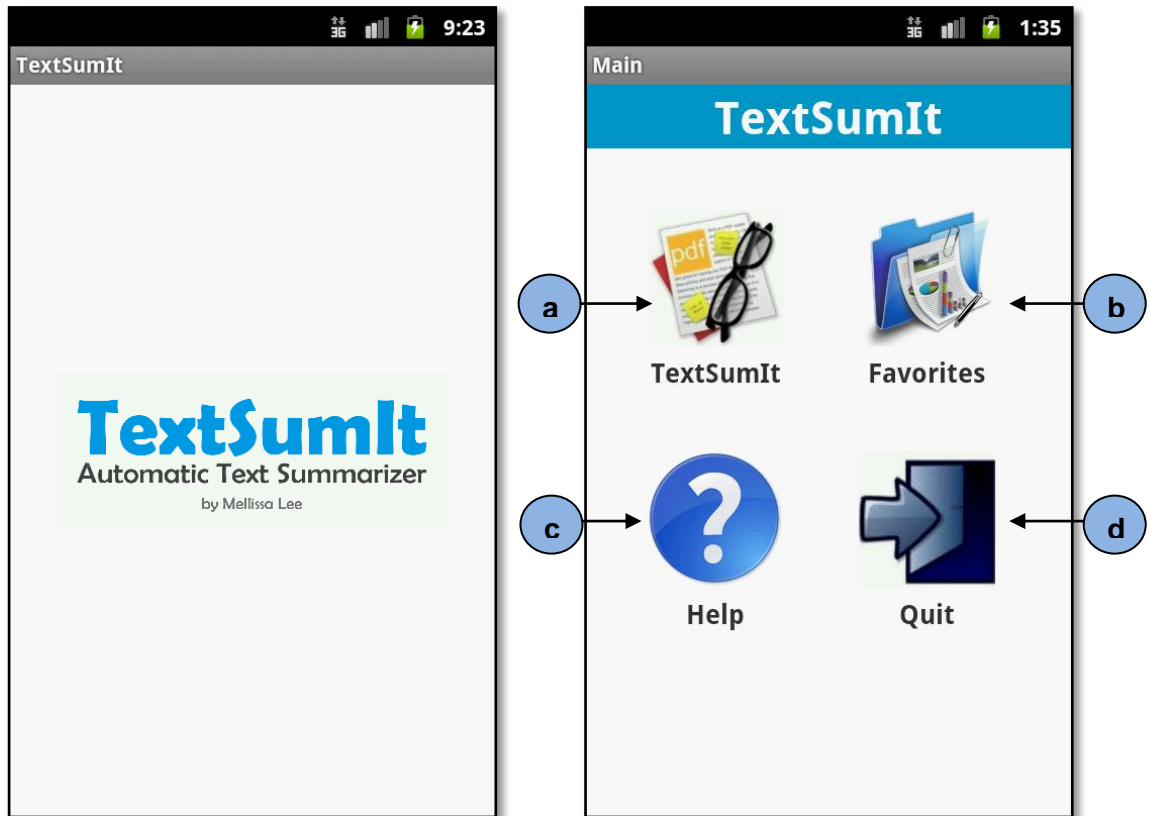
[Accessed 10 November 2012].

- [26] C.-y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," in *Proceedings of Workshop on Text Summarization Post-Conference Workshop (ACL 2004)*, Barcelona, Spain, 2004.
- [27] Wikipedia, "Precision and recall," Wikipedia, the free encyclopedia, [Online]. Available: [http://en.wikipedia.org/wiki/Precision\\_and\\_recall](http://en.wikipedia.org/wiki/Precision_and_recall). [Accessed 31 October 2012].
- [28] "Automatically summarize a document," Microsoft Corporation, [Online]. Available: <http://office.microsoft.com/en-us/word-help/automatically-summarize-a-document-HA010255206.aspx>. [Accessed 15 November 2012].
- [29] J. Kennedy and R. C. Eberhart, "Particle Swarm Optimization," in *Proceedings of the IEEE International Conference on Neural Networks, IEEE Service Center, Piscataway*, 1995.
- [30] M. S. Binwahlan and N. Salim, "Swarm Based Text Summarization," in *International Association of Computer Science and Information Technology - Spring Conference*, 2009.

## APPENDICES

### APPENDIX A

#### SYSTEM APPLICATION'S HELP MANUAL

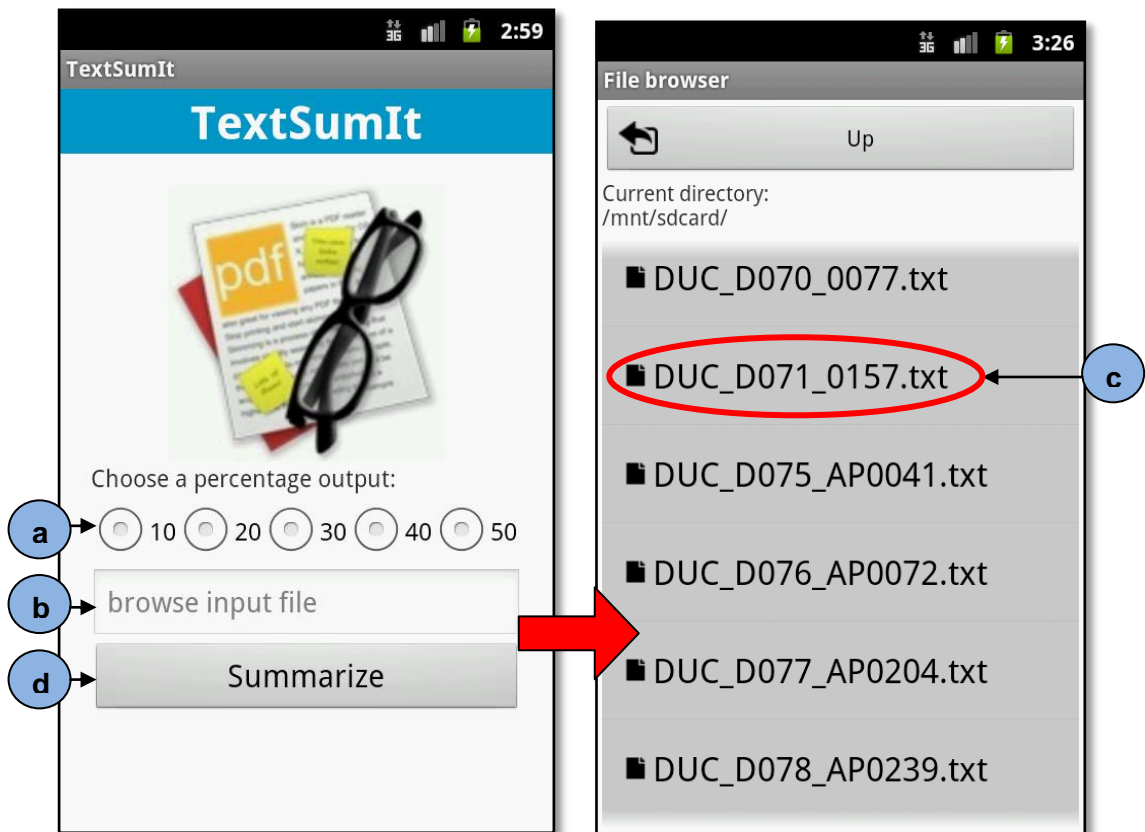


**Step 1:** Open the application to the homepage.

**Step 2:** Click on TextSumIt.

#### Navigation Functions:

- |                      |                                     |
|----------------------|-------------------------------------|
| (a) <b>TextSumIt</b> | : Main summarization function.      |
| (b) <b>Favorites</b> | : Store all saved summaries.        |
| (c) <b>Help</b>      | : User manual on using application. |
| (d) <b>Quit</b>      | : To quit the application.          |

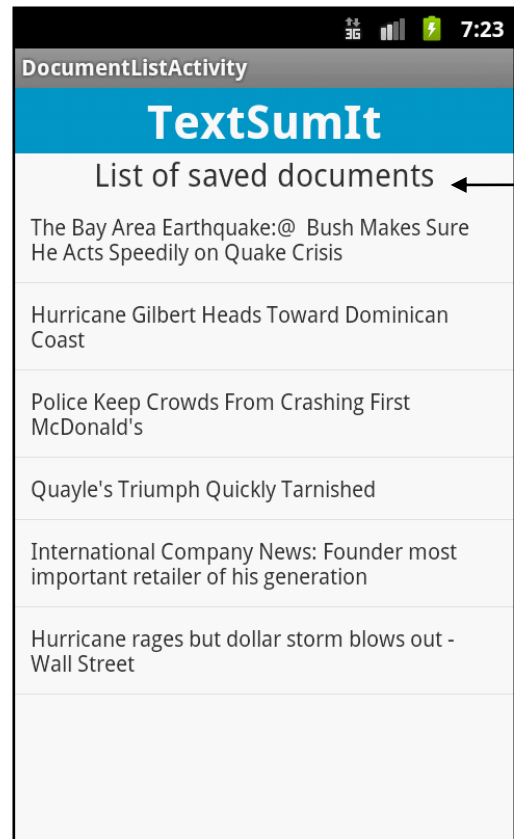
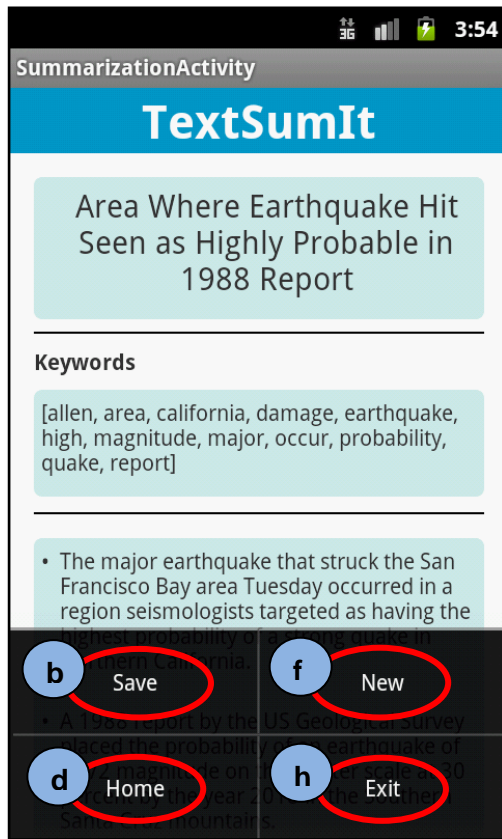


### Step 3:

**TextSumIt:**

- (a) Choose a percentage output for your summary.
- (b) Browse for an input file from your external storage directory
- (c) Ensure that only text file is selected.
- (d) Click on “Summarize” button to generate the summary for your document.




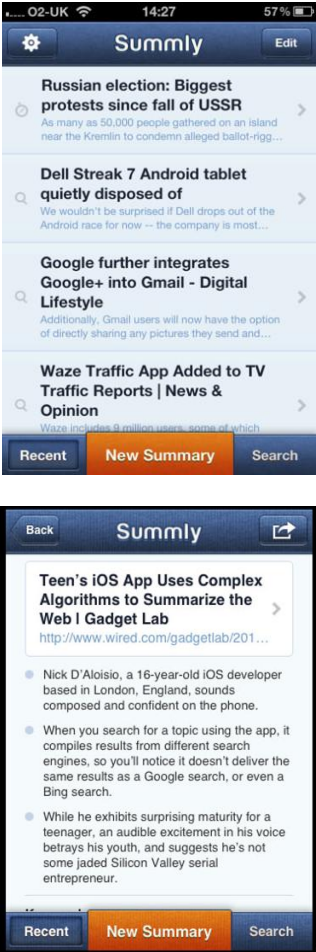


#### Step 4:

##### Summary Output:

- (a) The summary is arranged into three parts: first is the topic, then the keywords and lastly the key sentences are displayed in key points.
- (b) User may save the summary by clicking “Menu” and “Save” options.
- (c) The summary is saved under “Favorites” folder in the homepage.
- (d) User can then go back to homepage by clicking “Menu” and “Home” option.
- (e) Then click on “Favorites” to view all saved documents.
- (f) Users may click on “New” to start a new summarization.
- (g) Users may opt to “Quit” the application.

**APPENDIX B**  
**REVIEW ON APPLICATION AVAILABLE IN THE MARKET**

Application / System	Descriptions	Advantages	Disadvantages	Picture
 Summly	<p>Summly provides a simpler way to browse and search on the web whereby search results or news articles and web pages are automatically summarized using an algorithm, displaying a shortened text which makes it easier to consume. It produces results through AI, Machine Learning and Ontology techniques.</p> <p><b>Version:</b> 1.0.2 <b>Size:</b> 1.9 MB</p> <p><b>Compatibility:</b></p> <ul style="list-style-type: none"> <li>• iPhone 3GS</li> <li>• iPhone 4</li> <li>• iPhone 4S</li> <li>• iPod touch (3rd generation)</li> <li>• iPod touch (4th generation)</li> <li>• iPad.Requires iOS 4.3 or later.</li> </ul>	<ul style="list-style-type: none"> <li>• Quick web search and browsing</li> <li>• Save reading and browsing time</li> <li>• Summarize web content in many languages</li> <li>• Standalone application and extension to iPhone Safari browser</li> </ul>	<ul style="list-style-type: none"> <li>• Only available on iOS platform</li> <li>• Summarized results are inaccurate</li> <li>• Unable to summarize content of less than 500 characters.</li> <li>• Key points get jumbled up.</li> <li>• Inconsistent in summarizing</li> <li>• Fails to distinguish relevant and irrelevant information</li> </ul>	

## APPENDIX C

### PENN ENGLISH TREEBANK POS TAGS

1. *CC Coordinating conjunction*
2. *CD Cardinal number*
3. *DT Determiner*
4. *EX Existential there*
5. *FW Foreign word*
6. *IN Preposition or subordinating conjunction*
7. *JJ Adjective*
8. *JJR Adjective, comparative*
9. *JJS Adjective, superlative*
10. *LS List item marker*
11. *MD Modal*
12. *NN Noun, singular or mass*
13. *NNS Noun, plural*
14. *NNP Proper noun, singular*
15. *NNPS Proper noun, plural*
16. *PDT Predeterminer*
17. *POS Possessive ending*
18. *PRP Personal pronoun*
19. *PRP\$ Possessive pronoun*
20. *RB Adverb*
21. *RBR Adverb, comparative*
22. *RBS Adverb, superlative*
23. *RP Particle*
24. *SYM Symbol*
25. *TO to*
26. *UH Interjection*
27. *VB Verb, base form*
28. *VBD Verb, past tense*
29. *VBG Verb, gerund or present participle*
30. *VBN Verb, past participle*
31. *VBP Verb, non3rd person singular present*
32. *VBZ Verb, 3rd person singular present*
33. *WDT Whdeterminer*
34. *WP Whpronoun*
35. *WP\$ Possessive whpronoun*
36. *WRB Whadverb*

## APPENDIX D

---

### TextSumIt Application Questionnaire

---

The objective of this questionnaire is to determine the fulfillment of functional requirements of the system application. It is aim for user evaluation to better provide further enhancement on the application.

(1) How do you rate the ease of navigation of the application?

- a. Complicated    b. Difficult    c. Moderate    d. Easy    e. Very user-friendly

(2) How do you rate the readability of the summary?

*(with 1 being less readable and 5 being highly readable)*

Rating    :    1    2    3    4    5

(3) Do you think the application is useful to your daily work?

*(with 1 being least useful and 5 being very useful)*

Rating    :    1    2    3    4    5

(4) How much do you think the application helps in saving your reading time?

- a. Does not save time    b. 30%-50%    c. 51%-80%    d. More than 80%

(5) Will you continue using the application in the future?

- a. Yes    b. No

Comments/Feedbacks:

---

---

---

*Thank you for participating in the questionnaire. Your feedbacks are highly appreciated and will be useful in the research of this area.*

## APPENDIX E

### GANTT CHART

Detail Week	FYP 1														FYP 2													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Planning phase																												
Problem identification																												
Proposed solution and feasibility																												
Analysis phase																												
Information gathering i. Semantic Extraction ii. Features weight																												
System requirement analysis																												
Design phase																												
Design system architecture																												
Design graphical user interface																												
Implementation phase																												
Application development Functionality: i. Semantic Text Summarization ii. Summarize on Android																												
Evaluation of Experimental Results i. Preprocessing Results ii. Summary Evaluation iii. User evaluation																												

Key Milestone

**APPENDIX F**  
**TECHNICAL REPORT**

# Semantic Based Text Summarization for Single Document on Android Mobile Device

Oi Mean Foong<sup>1,a</sup> and Mellissa Lee Ai Lin<sup>2,b</sup>

<sup>1,2</sup>Department of Computer and Information Sciences, Universiti Teknologi PETRONAS  
Bandar Seri Iskandar, 31750 Tronoh, Perak, Malaysia

<sup>a</sup>[foongoimean@petronas.com.my](mailto:foongoimean@petronas.com.my), <sup>b</sup>[mellissalee90@gmail.com](mailto:mellissalee90@gmail.com)

**Abstract-** The explosion of information in the World Wide Web is overwhelming readers with limitless information. More often than not, readers are immersed in a pool of information with limited time to assimilate all of the articles. The text summarization research on mobile platform has been inspired by the new paradigm shift in accessing information ubiquitously at anytime and anywhere on Smartphones. In this research, a semantic and syntactic based summarization is implemented in a text summarizer. The objective of the paper is to integrate WordNet into the proposed system called TextSumIt which condenses single lengthy document into shorter summarized text that gives a higher readability to Android mobile users. The summary evaluation is done in comparison with the existing automated summarizer using human-generated summaries from Document Understanding Conference (DUC) as the reference summaries. Empirical results show that the proposed semantic based model improves the accuracy and efficiency in composing the summarized text.

**Keywords:** *Semantic, Single document, Sentence Score algorithm, WordNet, mobile device.*

## I. INTRODUCTION

### A. Background of Study

The increasing information available in the World Wide Web is overwhelming readers with immense data causing an information overload among readers. These pools of information need to be accessed to extract its important content in order to generate constructive knowledge. Ultimately, reader's goal is to have a quick and easy way to retrieve the main gist of the available information, comprehend it to gain knowledge, and finally make decision efficiently and effectively based on the knowledge gained.

In this context, there is an apparent need for an Automatic Text Summarization (ATS) as a solution to accommodate the growing information while saving time in producing it manually. ATS extracts content of the document using an algorithm, produces coherent and correctly-deliberated summaries, and displays the most important points of the original text to the user in a more condensed way and in accordance to each user's needs [1]. Improved methods have been proposed to identify and rate the importance of sentences to be extracted which enhance the summaries produced.

Today, the advanced technology era has challenged the ability to retrieve information at anytime and anywhere. Android has recently gain popularity among mobile users which make it a stepping stone for information retrieval on mobile devices. The main challenge would be to generate a text summary of which a condensed but precised contents of input source text is developed on an Android mobile platform with minimal back-end processing.

Text summarization is substantially a complex task especially when involving deep natural language processing e.g. syntactic, semantic, or pragmatic. In general, the two approaches towards creating text summaries are extraction and abstraction. Most researchers applied extractive summary as it is more difficult to develop abstractive summary due to its implementation of deep natural language processing which is still a growing field. The robustness of a summary depends on the coverage level of the summarization i.e. word-level and concept-level generalizations [2].

### B. Problem Statement

Large internet articles or journals are often cumbersome to read as well as comprehend. It is time-consuming and we have limited time to assimilate all of the articles which are at times, exceeding our capability to perceive. In addition, time is precious as today's fast paced era has caused people to demand for quick results.

As the information continues to grow exponentially, there exists a need to retrieve and filter the overloaded information. In fact, information overload is an increasing problem both in the workplace and in our general daily life. Moreover, readers are incapable of managing such huge pool of knowledge without a mechanism to support its readability. They either skip reading the content or not processing the information well, leading to wrong decisions made.

Another problem is that they lack a platform to access the information at anywhere and anytime. As technology advances, it becomes more convenient to access information on-the-fly by utilizing mobile devices.

The significance of this research is to produce a more efficient and coherent summaries using semantic models and to implement it on Android mobile platform.

### C. Objectives of Study

The objectives of this research are outlined as the following:

1. To propose and explore the semantic based model of extractive text summarizer to generate effective and meaningful summaries.
2. To implement and deploy the extractive document summarizer on Android platform.
3. To provide a higher readability of shorter text for users.

### D. Scope of Study

The scope of this research involves creating short version of any plain text documents (.txt format) in English language which excludes materials such as images, diagrams, graphs, tables etc. It will process a single text document using improved syntactic and semantic based algorithm as a combination to generate higher feature scores.

WordNet is used as the lexical database to assist in the semantic extraction of the text. The proposed system is developed on an Android mobile platform. Mobile application development will be based on Android 2.3 (Gingerbread). Application development will be build using Eclipse IDE with Android SDK and ADT.

## II. LITERATURE REVIEW

### A. Automatic Text Summarization (ATS)

The earliest works of text summarization started in 1950s which are pioneered by Luhn [3] and Baxendale [4]. The initial work started off with the implementation of statistical techniques in text summarization and gradually improves towards using natural language process (NLP), semantic analysis, fuzzy logic, swarm intelligence, and lastly hybrid fuzzy swarm [5]. The challenge to text summarization is the improvement of summary quality which remains as a key role in many researches.

There are two categories of text summarizers namely, statistical and linguistic. Statistical summarizers operate by finding the important sentences using statistical techniques such as word frequency (Luhn, 1958), text position (Baxendale, 1958), cue words and heading (Edmunson, 1969), sentence position (Lin & Hovy, 1997) and etc. On the other hand, linguistic summarizers use knowledge about the language such as syntax, semantics (Liu & Troels Andreasen, 2009) usage etc. to summarize a document [6].

Based on Hovy and Lin (1999) and Sparck Jones (1998) research, summaries can be categorized by the following criteria [2] [7]:

- **Usage: Indicative or Informative**

Indicative summaries usually provide the general concepts of the text document without showing specific content. Informative summaries on the other hand, reflect part of the content which allows readers to describe the content of the input text.

- **Audience: User-oriented or Generic**

User-oriented summaries focus on the interest of the readers on certain topics. It favors specific themes or aspects of the text. Generic summaries convey the point of view of the authors on the input text.

- **Derivation: Extractive or Abstractive**

*Extractive Summary* uses a fragment of the source text (key clauses, key phrases, sentences, etc) to structure the summary, i.e., summary copied from input [8]. These summaries lack the coherence as compared to abstractive summaries as it only conveys an approximate content of the source text.

*Abstractive Summary* reconstructs the extracted sentences, i.e., paraphrasing sentences to form a more cohesive and coherent summary. This method can condense text more strongly as compared to extraction by developing an understanding and expressing main concept of documents in clear natural language [8].

### B. Semantic Text Extraction

Semantic extraction involves the understanding of the structure and meaning of the natural language to produce semantic information from text documents [9]. However, the critical issue in extracting text semantically is the ambiguity and uncertainty of the meaning of texts. The improvement in summaries is achieved by applying lexical knowledge (e.g. WordNet) towards the text summaries to build a more comprehensive text. In addition, the cohesiveness of sentences can be enhanced by mapping the terms within the sentence to similar concepts using WordNet.

Based on the coverage level of processing, it can be divided into three categories, namely surface level, discourse level and entity level. Semantic extraction is part of the entity level approach. The entity-level approach builds internal representation of the text input. It then models the text entities with their relationships [10].

### C. WordNet-based

WordNet is a lexical database available online which contains a large repository of English lexical terms (also supports multilingual WordNet). Its structure is useful for linguistics and natural language processing implementation. In WordNet, it connects four types of Part-of-Speech (POS):



nouns, verbs, adjectives, and adverbs in which it groups the words into sets of synonyms called synset [11]. Each synset consists of the word, its explanation and its synonyms.

The main idea of WordNet is to combine the usage of a dictionary and thesaurus which can support text analysis and the implementation of artificial intelligence applications [12] such as word sense disambiguation, automatic text summarization, text categorization, information retrieval etc.

#### D. Current development of mobile application

Android applications have gain increasing popularity in today's large community. It has attracted many developers to write applications that extend the functionality of the devices. According to [13], it can be seen that Android Gingerbread API 10 (version 2.3.3-2.3.7) has the most usage yielding more than 50% of the Android users. Thus, the application in this study will be developed on an Android Gingerbread version 2.3.3 to accommodate to the vast amount of users within this platform. Nonetheless, other versions higher than 2.3.3 will also be able to run the application with proper development.

The challenges of text summary development on mobile applications can be summarized as the following:

- Small screen size
- CPU speed and memory are limited
- Limited content to be displayed
- Keyboards i.e. touch screen have limited space
- Overall user interface design

### III. METHODOLOGY

#### A. Software Development Methodology

In this research, Rapid Application Development (RAD) methodology is chosen as the development process for the text summarizer, as shown in Fig. 1. This methodology is selected as it avoid longer period of planning stage which allows system to be built quickly and concurrently during development stages due to the limitations of time and resources. The application system will be constantly refined and developed, allowing different features to be added and tested.

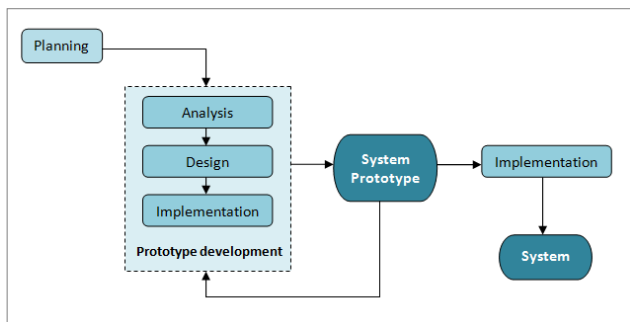


FIGURE 5: Rapid Application Development (RAD) methodology [14]

#### B. System Architecture

The proposed model focuses on both the preprocessing phase and the features weight calculation as shown in Fig.2. Users will input source text documents to summarize. During the preprocessing text stage, the text document undergoes a process of tokenization, stop words removal, WordNet stemming and finally Part-of-Speech (POS) tagging. The preprocessing is done to make it easier for the process of learning algorithm.

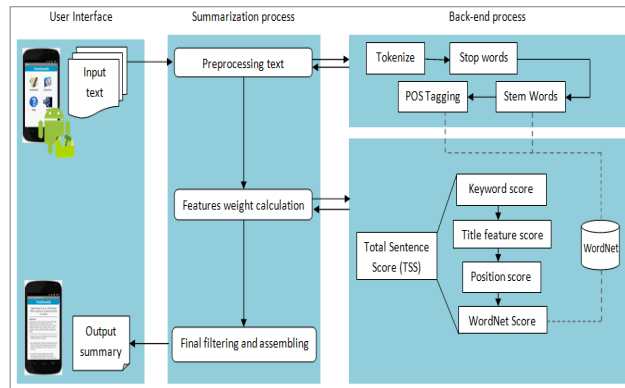


FIGURE 6: Proposed Model of Automatic Text Summarizer

The next stage, features weight calculation involves the calculation of the Total Sentence Score (TSS) which consists of Keyword Score, Title feature score, Position Score, and WordNet Score. These features weight will determine the rank of the whole sentences from the source text. The sentence with the highest weight will be selected for the summary output. It is enhanced with the semantic extraction algorithm using WordNet.

The last stage of the text summarization process is final filtering and assembling. At this stage, undefined references of sentences are being filtered and assembled to form a complete summary to be output to users.

#### C. Features Weight Calculation

- **Total Sentence Score (TSS)**

A final total sentence score (TSS) is calculated for each sentence using Eq. 1.

$$TSS(S_i) = w_1KW(S_i) + w_2Title(S_i) + w_3Pos(S_i) + w_4WN(S_i) \quad (1)$$

Where TSS is the Total Sentence Score, KW is the keyword score, Pos as the position score and WN as the WordNet score.  $w_1$ ,  $w_2$ ,  $w_3$ ,  $w_4$  are the weights for each score which will be sum up to TSS. The top scoring sentences will be selected as the output summary according to the compression rate set by users.

- **Keyword Score**

Keyword score is based on the Term Frequency (TF) and Inverse Sentence Frequency (ISF) algorithm by Joel Larocca (2002) [15]. It is computed as Eq. 2.

$$TF - ISF(w, s) = TF(w, s) * ISF(w) \quad (2)$$

Where the term frequency  $TF(w, s)$  is the number of times the word  $w$  occurs in sentence  $s$ , and inverse sentence frequency  $ISF(w)$  is computed in Eq. 3.

$$ISF(w) = \log\left(\frac{S}{SF(w)}\right) \quad (3)$$

Where sentence frequency,  $SF(w)$  is compute as the number of sentences in which the word  $w$  occurs.  $S$  is the total number of sentences in the document.

#### • Title Feature Score

This algorithm emphasizes on the sentences with keywords that are present in the title which resemble the theme of the document. It can be calculated using Eq. 4 [16].

$$Title(S_i) = \frac{Keywords\ in\ S_i \cap Keywords\ in\ Title}{Keywords\ in\ S_i \cup Keywords\ in\ Title} \quad (4)$$

#### • Position Score

The position score in this study comprises of two models. The *first position model* scores the sentences based on its similarity to the first and last sentence of the document at average [16]. The average score is calculated as Eq. 5, 6, 7.

$$P_1(S_i) = \frac{Keywords\ in\ S_i \cap Keywords\ in\ S_1}{Keywords\ in\ S_i \cup Keywords\ in\ S_1} \quad (5)$$

$$P_{Last}(S_i) = \frac{Keywords\ in\ S_i \cap Keywords\ in\ S_{Last}}{Keywords\ in\ S_i \cup Keywords\ in\ S_{Last}} \quad (6)$$

$$Pos(S_i) = \frac{P_1(S_i) + P_{Last}(S_i)}{2} \quad (7)$$

The *second position model* is based on the algorithm introduced by Barrera and Verma (2012). The model assumes the sentences towards the beginning and ending of a document are deemed important. It can be achieved using Eq. 8 [17]:

$$P_{cos}(S_i) = \frac{\cos\left(\frac{2\pi x}{k-1}\right) + \alpha - 1}{\alpha} \quad (8)$$

Where  $\alpha$  refers to the dent factor in which  $\alpha=2$  is used in this algorithm,  $k$  represents total sentences,  $x$  as the position of sentence  $S_i$  in the document.  $P_{cos}$  will be equally distributed as  $\alpha$  increase and on the other hand when  $\alpha$  decrease,  $P_{cos}$  is more concentrated to the value one in the beginning and end of the document.

#### • WordNet Score

The algorithm focuses on keyword semantics extraction as illustrated in [17]. The model's objective is to select the

sentences which contain words that have the closest meaning to the keywords extracted. The whole algorithm is associated with a collection of similar words of which Barrera and Verma described it as *thematic word list* [17]. The following Eq. 9 is the score allocated for each word  $w$ :

$$score(w) = \frac{1}{2^l} \quad (9)$$

Where  $l$  refers to the minimum level determined when the word  $w$  is compared with the synsets. If the word  $w$  is found in the sets of synonyms, then  $score(w)=1$  whereby  $l=0$ . If it is not found in this level, then  $l$  will be increase by one ( $l=1$ ) which means that the word is compared to the words in the preceding level of synsets. the overall equation for WordNet score is shown in the following Eq. 10:

$$WN_{syn}(S_i) = \sum_{w \in S_i} score_{syn}(w) \quad (10)$$

The closer the word  $w$  in a sentence,  $S_i$  to the synsets, the higher the WordNet score.

#### D. Development Tools Required

The following are the tools required for the system development:

- Android mobile (Gingerbread 2.3)
- Android Emulator
- Java plug-ins for Android
- Eclipse IDE
- WordNet
- ROUGE tools

## IV. RESULTS AND DISCUSSION

### A. Performance Evaluation Measures

The final output summaries are evaluated using Recall-Oriented Understudy for Gisting Evaluation (ROUGE) which is based on [18]. It automatically evaluates the quality of summaries using measures such as n-gram, word-pairs as well as the word sequences by comparing the machine-generated summaries with human-generated summaries. The summarized text will be evaluated using the f-measure according to the recall and precision scores based on Eq. 11, 12 and 13. More specifically, ROUGE-1 (N-gram Co-Occurrence Statistics) is chosen as the evaluation function in this research.

$$R = \frac{|\{Relevant\ Sentences\} \cap \{Retrieved\ Sentences\}|}{|\{Retrieved\ Sentences\}|} \quad (11)$$

$$P = \frac{|\{Relevant\ Sentences\} \cap \{Retrieved\ Sentences\}|}{|\{Relevant\ Sentences\}|} \quad (12)$$

$$F - score = \frac{(1 + \beta^2)RP}{R + \beta^2 P} \quad (13)$$

Recall,  $R$  measures the percentage of relevant instances that are being retrieved whereas Precision,  $P$  on the other hand, measures the percentage of retrieved instances that are

relevant. F-score is a weighted average of both recall and precision.

### B. Existing Text Summarizer and Reference Summaries

Word AutoSummarize will identify key points from the text document, usually works well on well structured articles or reports. The system summarizer TextSumIt is compared to Word AutoSummarize as both has similar algorithm behind the text summarization process.

In Word AutoSummarize, each sentence in the document are assigned with a sentence score and then display the highest scoring sentences according to the percentage of output determined by users [19]. The scoring is computed based on the frequency of words in a sentence that occur in the document.

As for the reference summaries, text corpus from Document Understudy Conferences (DUC) is taken for the evaluation of both summarizers. More specifically, selected articles from DUC 2002 are taken as the sample input text for summarization.

### C. Experimental Results

The evaluation is done on a selected number of articles from DUC 2002. Thirty articles are selected as an input for the system summary, TextSumIt and Word AutoSummarize. The summarization on TextSumIt and Word AutoSummarize is done with the closest length possible to yield approximate similar results. In this research, the evaluation method used for the summaries is ROUGE-1.

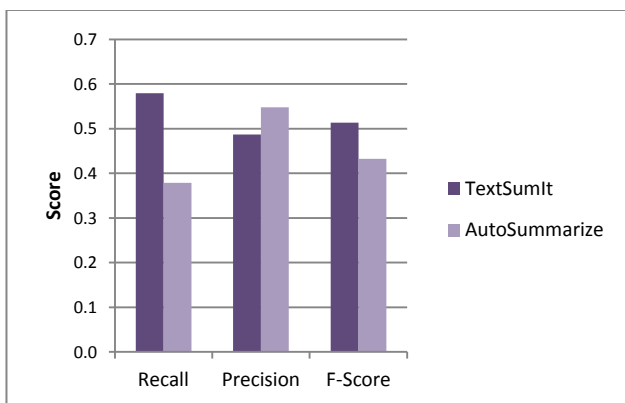


FIGURE 7: Overall average performance comparison between TextSumIt and AutoSummarize

As shown in Figure 3, the overall average recall, precision and F-Score indicates that TextSumIt gives a better output as compared to Word AutoSummarize. Both summaries were compared to the reference summaries which are human-generated by experts. In general, TextSumIt has a higher F-score of 0.51 as compared to Word AutoSummarize which has a lower F-Score of 0.43. The significant difference in the results is due to the different algorithm used in the sentence extraction.

TextSumIt which uses a semantic based algorithm is able to extract sentences more precisely as compared to Word AutoSummarize which only uses frequency-based algorithm. In spite of the higher precision score obtained by the Word AutoSummarize, the proposed semantic model of TextSumIt yield the best F-score of 0.51 as compared to the other model. The current results could be considered satisfactory.

### D. Graphical User Interface (GUI)

The interfaces used for users' navigation within the TextSumIt application on Android are shown in Fig. 4 to Fig. 6. The output summary of the text summarization is shown in Fig. 6 whereby the content of documents is summarized into several key points to make it easier for reading.

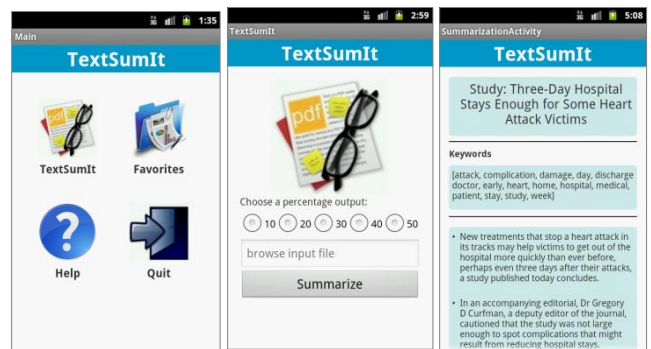


FIGURE 8: Home Screen

FIGURE 9: Text Input Screen

FIGURE 10: Summary Output Screen

### E. User Testing

The user testing is done mainly for the evaluation of the TextSumIt application and is a subjective evaluation by the users. The main features being tested on the Android application is the ease of navigation, readability and time saved. The testing targeted 30 users, mainly students who used to read lengthy documents like research papers, journals or articles and uses Android mobile devices. The results of the user testing are shown in Fig. 7 to Fig. 9.

Based on Fig. 7 to Fig. 9, it can be concluded that more than half of the users find the application very user friendly, provides high readability, and able to save 51%-80% of their time assimilating lengthy documents. The results are considered satisfactory as it meets the objective of the research.

From the questionnaire returned by the users, there were several comments and feedbacks on the application. Generally, users like the idea of text summarization on the Android mobile device as it is able to provide them the gist of document quickly and briefly. Some of them find it handy especially when they want to assimilate lengthy documents when they are busy. Overall, the response is positive and majority of them will utilize the application either for personal use or at workplace.

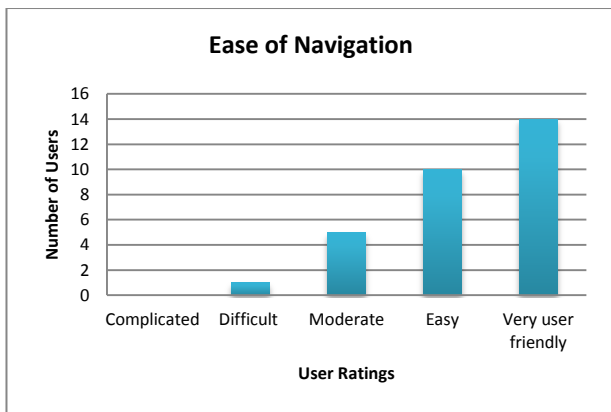


FIGURE 11: Evaluation Results for Ease of Navigation of Application

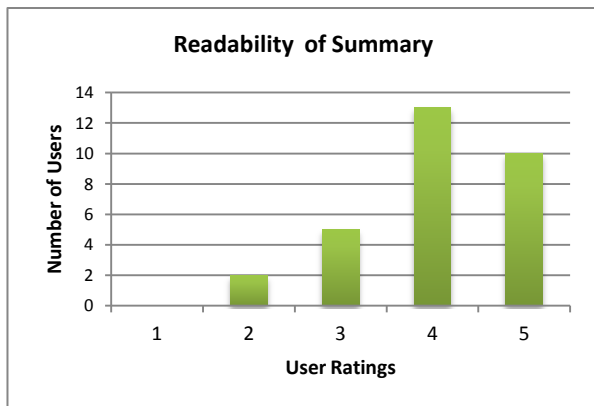


FIGURE 12: Evaluation Results for Readability of Summary

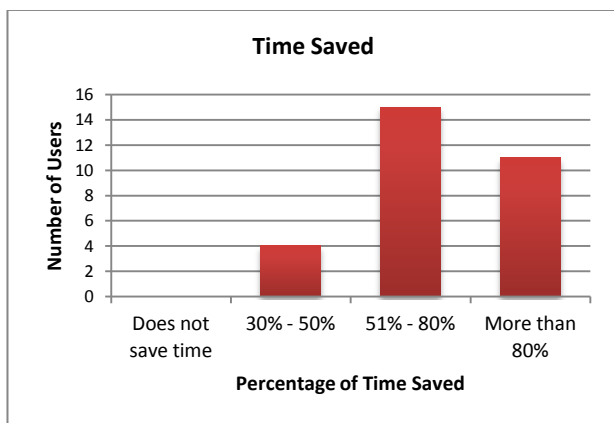


FIGURE 13: Evaluation Results for the Percentage of Time Saved by Using the Application

## V. CONCLUSION AND FUTURE WORK

In this research, semantic extraction using WordNet is used to enhance the summarization process which gives a better outcome of summaries. It has also enhanced the preprocessing of text in terms of its stemming algorithm and additional part-of-speech tagging. An improved features weight calculation is also implemented to give a higher accuracy in selecting sentences for output summaries. This research will be a stepping stone for future mobile web summarization. However, it is recommended to run the back end processes on a server to reduce the heavy processing on the mobile device. Text classification can be extended to the mobile application to categorized documents into themes.

## ACKNOWLEDGEMENT

The author would like to express her sincere appreciation to her supervisor, Ms. Amy Foong Oi Mean who guided and assisted her patiently as well as her friends for their continuous support throughout the development of the application and algorithm.

## REFERENCES

- [1] I. Mani, "Recent Developments in Text Summarization," in *Proceedings of the tenth international conference on Information and knowledge management, CIKM*, New York, USA, 2001.
- [2] E. Hovy and C.-Y. Lin, "Automated Text Summarization and the SUMMARIST system," 1999.
- [3] H. P. Luhn, "The Automatic Creation of Literature Abstracts," *IBM Journal*, pp. 159-165, 1958.
- [4] P. Baxendale, "Machine-made Index for Technical Literature - An Experiment," *IBM Journal of Research Development*, vol. II, no. 4, 1958.
- [5] M. S. Binwahlan, S. Naomie and L. Suanmali, "Fuzzy Swarm Diversity Hybrid Model for Text Summarization," *Information Processing and Management*, vol. 46, pp. 571-588, April 2010.
- [6] O. M. Foong, A. Oxley and S. Sulaiman, "Challenges and Trends of Automatic Text Summarization," *International Journal of Information and Telecommunication Technology*, vol. I, no. 1, pp. 34-39, 2010.
- [7] K. S. Jones, "Automatic Summarising: factors and directions," in *MIT Press*, England, UK, 1998.
- [8] V. Gupta, "A Survey of Text Summarization Extractive Techniques," *Journal of Emerging Technologies in Web Intelligence*, vol. II, no. 3, pp. 258-268, 2010.
- [9] S. Jusoh and H. M. Al Fawareh, "Semantic Extraction from Texts," in *Proceedings of International Conference on Computer Engineering and Applications IPCSIT*, Singapore, 2011.
- [10] R. K. Wong and C. Sia, "An Efficient WordNet-Based Summarizer for Large Text Documents," New South Wales, Australia, 2003.
- [11] Princeton University, "WordNet: A lexical database for English," 6 January 2012. [Online]. Available: <http://wordnet.princeton.edu/>. [Accessed 31 July 2012].
- [12] "WordNet," Wikipedia: the free encyclopedia, [Online]. Available: <http://en.wikipedia.org/wiki/WordNet>. [Accessed 1 August 2012].
- [13] A. Developers, "Platform Versions," Android, [Online]. Available: <http://developer.android.com/about/dashboards/index.html>. [Accessed 2 October 2012].
- [14] "Lessons from Software Development," DrTdaxp, [Online]. Available: <http://www.tdaxp.com/archive/2005/07/20/dreaming-5th-generation-war.html>. [Accessed 20 June 2012].
- [15] J. Larocca Neto, A. A. Freitas and C. A. A. Kaestner, "Automatic Text Summarization Using a Machine Learning Approach," in *Advances in Artificial Intelligence*, Brazil, Springer Berlin Heidelberg, 2002, pp. pp 205-215.
- [16] O. M. Foong and A. Oxley, "A Hybrid PSO Model in Extractive Text Summarizer," in *IEEE Symposium on Computers & Informatics*, 2011.
- [17] A. Barrera and R. Verma, "Combining Syntax and Semantics for Automatic Extractive Single-Document Summarization," *Proceeding CICLing'12 Proceedings of the 13th international conference on Computational Linguistics and Intelligent Text Processing*, vol. Part II, pp. 366-377, 2012.
- [18] C.-y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," in *Proceedings of Workshop on Text Summarization Post-Conference Workshop (ACL 2004)*, Barcelona, Spain, 2004.
- [19] "Automatically summarize a document," Microsoft Corporation, [Online]. Available: <http://office.microsoft.com/en-us/word-help/automatically-summarize-a-document-HA010255206.aspx>. [Accessed 15 November 2012].