NATURAL LANGUAGE PROCESSING ENGINE FOR ON-THE-GO APPLICATION

MUHAMMAD HADI ASYRAFI BIN ABDUL HALIM

A report submitted in partial fulfilment of

the requirements for the award of the

degree of Bachelor of Electrical – Electronics Engineering

Faculty of Electrical Engineering

Universiti Teknologi Malaysia

December 2016

I declare that this thesis entitled "Automated Real-Time News Classification System" is the result of my own research except as cited in the references. The thesis has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature : ....................................................

Name : ....................................................

Date : ....................................................

**ACKNOWLEDGEMENT**

As for the completion of this report, there are many people that helped me all the way through which I would personally want to give my dearest thanks to. Clearly, without their help and influential advice I would have not been able to complete this project successfully as planned. In particular, I would like to express my sincere appreciation to Dr. Musa bin Mohd Mokji, my final year project’s supervisor for allowing me to undertake this project and shared his insight and expertise tirelessly.

I would also like to give a special thanks to Universiti Teknologi Malaysia (UTM) for providing this platform and giving me a chance to become part of this institution. It was a really great pleasure to be here. Continuous support and warm gesture by my fellow undergraduates helps me to go through my study until the completion of this report.

And lastly, I would like to also further my gratitude to all my beloved lecturers in UTM and also to everyone and anyone who have helped me in one way or another. My family also played an important part in supporting me and encouraging me. I am grateful to each and every one of them.

**ABSTRACT**

A popular goal of researchers in the field of data mining and artificial intelligence today is the natural language processing by computers. As we all know, computers generally didn’t have the capability to individually process natural language, the language that human used every-day. This project aims to apply Natural Language Processing (NLP) in creating a processing module targeted towards mobile devices in on-the-go applications. In this fast moving world today, time is gold. Hence, this module will introduce features that will be beneficial for everyday usage, travelers, students and even professional workers. It hopes to increase in user productivity while reducing time consume skimming through text traditionally. This processing engine accepts commonly used information files ranging from documents, images, audio files and even Wikipedia articles. Text processing features available are Automatic Text Summarization, Semantic Analysis, Language Detector and Keyword Highlighting. The coding will be solely written in Python programming language as they are simple yet powerful while also support related libraries for NLP and text extraction.

**TABLE OF CONTENTS**

**CHAPTER TITLE PAGE**

**DECLARATION** ii

**DEDICATION**  iii

**ACKNOWLEDGEMENT**  iv

**ABSTRACT**  v

**ABSTRAK**

**TABLE OF CONTENTS** vi

**LIST OF TABLES**

**LIST OF FIGURES**

**LIST OF ABBREVIATIONS**

**LIST OF SYMBOLS**

1. **INTRODUCTION 1**
   1. PROJECT BACKGROUND 1
   2. PROBLEM STATEMENT 3
   3. OBJECTIVES 4
   4. PROJECT SCOPE 5
   5. PROJECT GANTT CHART 5
   6. REPORT OUTLINE 6
2. **LITERATURE REVIEW 8**
   1. INTRODUCTION 8
   2. NEWS GATHERING 9
   3. PRE-PROCESSING 9
   4. INDEXING 10
   5. FEATURE SELECTION 11
   6. CLASSIFICATION 14
   7. FINDINGS 14
3. **METHODOLOGY 16**
   1. INTRODUCTION 16
   2. PROJECT WORKFLOW 17
   3. SOFTWARE IMPLEMENTATION 20
      1. WEB SCRAPER 20
      2. TEXT PREPROCESSING 21
      3. CLASSIFICATION 21
4. **RESULT AND DISCUSSION 23**
   1. INTRODUCTION 23
   2. EXPECTED OUTCOME 24
5. **CONCLUSION 25**

**REFERENCES 26**

**CHAPTER 1**

**INTRODUCTION**

1. **Project Background**

This project aims to build a dedicated natural language processing engine for mobile devices targeted for on-the-go application. It will consist of 2 parts which is input preprocessing, natural language processing engine which is the main part of this project. These 2 parts will be combined together and wrapped by a graphical user interface (GUI) to create an intuitive and fully functional text processing engine for on-the-go application in mobile devices. The program will be coded in Python together with several related libraries already available for usage.

The preprocessing module are mainly responsible for converting input from several sources into Python’s string to be fed to the main processing engine afterwards for further processing. A powerful library called ‘textract’ is used for this. Besides browsing local documents, preprocessing module also supports text extraction from Wikipedia. Wikipedia’s article scraping are chosen to be integrated into the program due to its popularity and usability as information source nowadays. In-order to extract news articles from websites, the use of APIs and RSS feeds are indeed the simplest approach available. Data accessed through this manner comes in structured form making it easier to process. However, not all websites provide these features and sometimes even if they are available they are not maintained regularly [1]. This is where web scraping comes in. Web scraping is a technique of extracting information from websites and transforms them from unstructured form into a structured format [2]. A Python library named ‘BeautifulSoup’ will be used to assist in HTML parsing. The document later will undergo HTML tags stripping which will only leave text document for further processing.

The second module is the text preprocessing engine for Natural Language. “In contrast to artificial languages such as programming languages and mathematical notations, natural languages have evolved as they pass from generation to generation, and are hard to pin down with explicit rules” [3]. These natural language text which are in the form of unstructured data cannot be processed directly by computers [4]. Hence, Natural Language Toolkit, or NLTK in short which provides rich libraries for solving the complex nature of natural language related programming in Python are used to help with the processing tasks. Some of the most important features of this libraries that are used within this program are Tokenization, Part-of-Speech (POS) tagging and stop-words identification. Processing tasks that are handled by this module includes language detector, keywords highlighting, sentiment analysis and text summarization.

These three parts will be combined together into a running program which will automatically classify newly published news from several assigned websites immediately in real time. Web scraper will pass extracted news for preprocessing using NLTK library and later will be classified by SVM classifiers. The resulting classification will be cross-referenced with human based classification of the exact material for accuracy evaluation. Time taken for each classification will also be taken as one of the performance indicators of the system.

1. **Problem Statement**

We have long pass the era of analog and today we can clearly see how digital the world around us is. Almost every part of our lives today is managed and handled electronically be it in communication, education, financial, military and even as far as our daily social lives. We have indeed come upon the great era of computing and this is only the beginning.

Millions of information in the form of digital data are pass around in networks spreading worldwide, reaching from one part of the world to another opposite part almost instantly. Hence, not to our surprise 90% of the world’s data has only been created during these past 2 years [7]. This abundance of data coined as Big Data will be growing exponentially in years to come. This vast amount of data is seemingly impossible for humans to handle manually.

These huge data collections are both blessings and a curse according to the way it is handled. Hence, the world we live today are in need of certain way to automate knowledge extraction from textual documents and significantly simplifies the process without losing the essence of it . Traditional ways of reading, and skimming may very well be considered obselete and time consuming. By using computerized processing, textual documents are approached differently than before increasing in productivity and efficiency.

Businesses, governments, militaries and industries have all took part in this race. The race of extracting this large volume of data and convert them into a meaningful format. Computers are indeed exceptional in those jobs however only if they came in the form of structured data, the form of data with high degree of organizations. On the other hand, unstructured data such as text and voice are essentially the polar opposite. These unstructured data made up around 80% of the all data available [8]. These unstructured data call for Natural Language Processing.

1. **Objectives**
2. Use text analytics to process unstructured data from news articles
3. Create a program that can classify news as close to human based classification
4. Provides an immediate and automated classification for news from websites
5. Provides a running program that can classify news in real time
6. News classification of at least 80% accuracy
7. **Project Scope**
8. Extract news from only a handful of websites
9. Use only bag of words model to convert unstructured data into structured data
10. Classify news into 5 categories of business, world, sport, lifestyle and others
11. **Gantt Chart**

Work progression plan for FYP 1 and FYP 2 have been planned carefully in the form of Gantt Chart as depicted in table 1 and 2.

Table 1: Gantt Chart FYP 1

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Task | 2016 | | | | | | | | | | | | | | | | |
| September | | | | October | | | | November | | | | | December | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 5 | 1 | 2 | 3 | 4 |
| 1 | Literature review |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 | Big Data |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 | Natural Language Processing |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 | Machine Learning Algorithm |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5 | Web content extraction |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Table 2: Gant Chart FYP 2

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Task | 2017 | | | | | | | | | | | | | | | | | | | |
| February | | | | March | | | | April | | | | May | | | | Jun | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| 6 | Web scraper |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 7 | Text preprocessing |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 8 | Support Vector Machine |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 9 | Application Program |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 10 | Testing and tuning |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 11 | Report writing |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 12 | Viva preparation |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

1. **Report Outline**

This project report is divided into 5 different chapters. The first chapter is the introduction to this project. It consists of explanations on project background, problem statements, objectives, project scopes and detailed Gant Chart.

The 2nd chapter focuses on reviewing previous works and researches done related to my project. It serves as a guideline to the reader on this topic. Chapter 3 on the other hand emphasizes on methodology and approach to be taken including project workflow to complete this project.

Expected outcome of this project will be discussed briefly on chapter 5. Lastly, chapter 5 serves to conclude of this project. Suggestion and recommendation will also be discussed here.

**CHAPTER 2**

**LITERATURE REVIEW**

1. **Introduction**

Text classification has been widely researched in recent years. It is fueled by the advanced of information and communications technologies of this 20th century which makes acquiring relevant information and sorting textual data a daunting task. This topic had been approached using various methods many times before in the attempt to replace labor workforce for the task. Thankfully the advent of practical artificial intelligence algorithms had paved the path for achieving an autonomous text classification. This review will focus on recently published papers on machine learning approach for this topic.

1. **News Gathering**

There are many methods available for gathering textual data for classifiers training. However, machine learning training needs a large set of data for training to get an accurate model. Some researcher relies on corpora which contains thousands of documents readily prepared for text analytics problems. Chen at el. (2016) used corpora from 20Newsgroup and RCV1-V2 while Dadgar et al. (2016) also use dataset from 20Newsgroup combined with BBC for training. The other reviewed researchers also downloaded corpus from another party such as Chinese text classification corpora of Fudan University which has 1880 classified documents [10] and New China News Agency (NCNA) 2013 issues of domestic news [11].

1. **Pre-processing**

Text data are unstructured data as it comes in the natural free form [12]. Before these unstructured data can be processed by a computer, they need to be processed beforehand. The extent of preprocessing differs between each research methodology and relies on training algorithm, feature selection and training algorithms to be used. Its objective is mainly to clean up the text from noises and useless information which may distract our classifiers. They also help in extracting only the useful information from the given text [12].

Some of the most common preprocessing stages are the removal of punctuation marks, semi colons, quotes, comma, exclamation marks, date and other irrelevant characters. These characters which are defined differently across languages are known as Diacritics [4]. Some researcher also remove numbers for convenience as they also didn’t contribute to text classification. Next, remaining texts are commonly broken down using tokenization into smaller segments. Tokenized texts can be represented in several levels. Unigram, bigram and trigram which are part of n-grams are the examples of sub-word level representation.

In every language, there exists stop words which are common and carries little to no meaning and are not useful for training purposes. Some example of these words in English vocabulary are ‘a’, ‘an’, ‘and’, ‘are’, ‘it’, and ‘of’ [13]. These words are best removed to reduce training time. Fortunately, there are also prepared stop words readily available to be used provided by Journal of Machine Learning Research called SMART [14]. SMART contains 571 stop words commonly found in English documents [6]. In addition, there are some other preprocessing methods used such as stemming and character transformation that may be used to improve accuracy of the classifiers.

1. **Indexing**

Bag of words models are one of the simplest and most commonly used indexing methods for text classification. It treats the text document as a bag of words and convert them into term frequency representation in vector space. The downturn of this method is that it only counts the frequency of term repetition in a given text, hence syntactic and semantic are disregarded. However, this method is efficient enough in most text classification problems and gives a good accuracy.

1. **Feature Selection**

Feature selection is the process of reducing the dimensionality of feature space by selecting subsets of features for model construction [6]. There are so many types of feature selection studied before even in the scope of text classification such as Term Frequency – Inverse Document Frequency (TF-IDF), Boolean Weighting, Information Gain, Document Frequency and Mutual Information [4]. All of these studied methods shows promising results and have their own pro and cons. However, for this project feature selection are omitted. It has been reported that is indeed increases accuracy and reduces training time, but using SVM approach the differences are small.

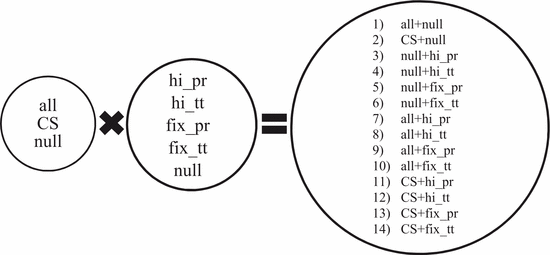


Figure 1: 14 Combination of feature sets [6]

As mentioned in the paper by Chen et al. they conducted experiments to compare the accuracy of each combinations to find out whether implementing LDA and feature selection into SVM classification improves the result. The results are shown in figure 2 below.

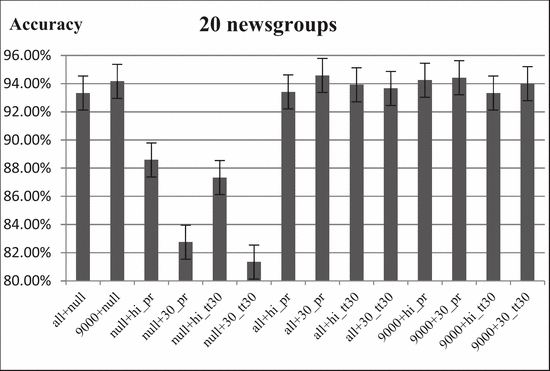


Figure 2: Accuracy of feature sets combination on 20 Newsgroups [6]

Figure 2 above shows the differences of accuracy of classifiers trained using a combination of feature sets. ‘9000’ refers to feature selected sets using chi-square test a feature selection method while ‘all’ is the experimental result of using all term without any feature selection. The term ‘null’ after addition symbol refers to the classifiers trained without using Latent Dirichlet Allocation (LDA). From this experimental result, we can see that even by omitting LDA and feature selection in the project we can still achieve a good accuracy by comparing ‘all+null’ and ‘9000+null’.

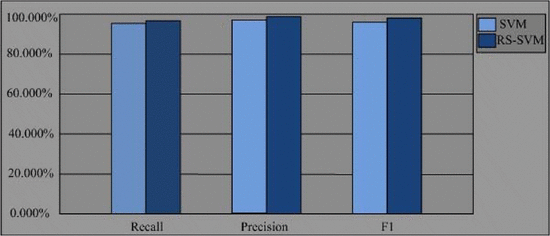


Figure 3: Recall and Precision accuracy of SVM and RS-SVM [10]

Another research tries to compare the accuracy of text classification by using SVM versus SVM with feature selection using Rough Set as depicted in figure 3 also showed similar findings as mentioned before. Incorporating feature selection and LDA certainly increases the accuracy by some extent. However, this project omits both of them in trading off accuracy with simplicity.

1. **Classification**

The most important aspects for any text classification is the classification methods adopted. This plays an important role and directly affect the accuracy, precision, training and recall time. There exist many classification methods and algorithm and each has been researched before. Some examples of them are k-Nearest Neighbor, Naïve Bayes, SVM, ANN and Decision Tree [4]. From literature reviews conducted, SVM seems to be the most used algorithm in the field of text classification.

This project later will also use SVM algorithm to train classifiers as it shows promising results in previous researches. The accuracy of SVM reaches approximately 90% in all researches reviewed before. There are some researches that incorporating LDA as mentioned before. Some combines them in hierarchical methods creating 2 layers of classification [11]. First classifies into several most likely group using topic modelling which later will be classified into one specific category using SVM. This method of combining LDA and SVM are also used by Chen et al. in their report.

1. **Findings**

Below are the findings from 4 separate papers that have been reviewed summarized. Summary from paper authored by Rana et al. are not included as they didn’t carry out any specific experiments instead focused on reviewing the most common methodologies in text classification.

Table 3: Findings from literature reviews

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | [11] | [6] | [12] | [10] |
| Input Corpora | New China News Agency | 20Newsgroup and RCV-V2 | BBC and 20Newsgroup | Chinese Text Classification Corpora of Fudan University |
| Preprocessing Method | Parsing, stop-word removal | Punctuation, Numbers, and SMART stop words removal. Stemming | Diachritics and stop words removal, character transforming, and tokenization | N/A |
| Indexing | Word sequence | Bag of Words | N/A | N/A |
| Feature Selection | TF-IDF | TF and or Topic Modelling | TF-IDF | Rough Set |
| Classification | LDA and SVM | LDA and or SVM | SVM | SVM |
| SVM Kernel | RBF | N/A | RBF | N/A |
| Accuracy of SVM classifier (approx..) | 94.55% | 93.00% (20 Newsgroup), 96.90% (RCV-V1) | 97.84% (BBC), 94.93% (20 Newsgroup) | 95.00 % |

**CHAPTER 3**

**RESEARCH METHODOLOGY**

1. **Introduction**

This chapter will focus on methods and approach taken to complete this project. This project focuses mainly in software components written entirely in Python programming language. We will explore extensively the idea and steps to preprocessing several sources of input into Python’s string and the main engine’s processing features. However, this project will not delve deep into algorithm parts of these topics. Readily available open source libraries for Python will be used to help complete this project.

1. **Project Workflow**

Figure 4 shows project workflow of the Automated Real-Time News Classification System from the beginning of FYP 1 until the end of FYP 2 in general. It started with finding and determining the problem statement for this project. This will be the back bone of later phase in design and analysis.

The next step is reviewing past papers for solutions suggested by other researchers. Information gathering stage is important as it involves a lot or reading and understanding project related materials ranging from web scraping, natural language processing, artificial intelligence and machine learning algorithms that will help throughout this project.

Listing objective, aim and scope will act as a good guideline and mile stones for project design and analysis. The next stage is the development of software. The software later will be test for accuracy. If the project achieved the desired result, then the project will end with interface improvement for presentation.

Problem Statement

Literature Review

Information Gathering

Objective, Aim and Scope

Design and Analysis

Software Development

Performance Testing

Accuracy > 80%

Interface Improvement

No

Yes

Figure 4: Project Workflow

Web Scraper

Text Preprocessing

Support Vector Machine Classifier Training

Programs Integration

Figure 5: Software Development Flow

Figure 5 shows the development cycle of software components which are the core of this project. This project adopts the waterfall model as it is the easiest to implement and sufficient for small scale project such as this. The development part starts off with writing code for web scraper. Then, the text preprocessing part will make use of library for natural language processing. The 3rd part of this development cycle will focus entirely on training classifiers as specified in the objective. Upon completing all these 3 part, they will all combine together into a complete running program. The program development cycle will be reiterated if the accuracy of the classifiers falls below expected percentage.

1. **Software Implementation**

Software implementation is the biggest and the most important part of this project. The software development cycle I expected to take up around 65% of the total time spent on this project. As mentioned before, this project consists of 2 parts or modules which is input preprocessing and natural language processing. Each part needs to work harmoniously with each other for achieving the desired outcome.

Input Preprocessing

Natural Language Processing Engine

Wikipedia

String

Summarizer

Sentiment Analysis

Audio

Document

Image

Language Detector

Keyword Highlighter

1. **Input Preprocessing**

The main role of this module is processing and converting any supported input file fed into the system into Python’s string data type. This is because the main module of NLP processing accepts only string type of data for processing. The library used for this feature is called textract. Textract provides a single interface for extracting data from multiple file types by having dependencies on several related established libraries for Python such as python-docx, tesseract-ocr, pdftotext, python-pptx, and SpeechRecognition. Below are listed some of the supported data types available.

|  |  |  |
| --- | --- | --- |
| Category | File Type | File Extension |
| Document | Text File | .txt |
| Power Point Presentation | .pptx |
| Portable Document Format | .pdf |
| Word Document | .doc .docx |
| Open Document Text | .odt |
| Image | Portable Network Graphic | .png |
| Graphic Interchange Format | .gif |
| Joint Photographic Expert Group | .jpg .jpeg |
| Audio | MPEG Audio Layer-3 | .mp3 |
| WAVE Audio Format | .wav |
| OGG Vorbis File Format | .ogg |

In addition to local text extraction using textract, this preprocessing module also integrated Beautiful Soup into the module. Beautiful Soup is a powerful open source HTML parsing library. Urllib2 is another important library that is used to fetch HTML documents online. Combining these 2 libraries together, a web scraper dedicated for Wikipedia text extraction is employed. “Web scraping is a computer software technique of extracting information from websites” [2]. It deploys an autonomous bot or program code that goes inside any given website refers by its URL as endpoint and extract relevant information according to the programmer specification. The information from websites are usually in HTML format. In this project, the web scraper will be programmed to extract text only from Wikipedia articles. Other junk data that may compromises NLP processing unit such as table, image and charts will also be filtered out.

1. **Natural Language Processing Engine**

This NLP module will rely mostly on Natural Language Toolkit (NLTK) an open source library for language processing in Python. Using this library, features including Tokenization, Stop Words Identification and Part-of-Speech Tagging are the one will be most commonly used. Tokenization is the process of splitting words individually in the whole text and part-of-speech such as nouns, verbs, and adjective will later be assigned by POS-tagger to the individual words. In every language, there exists stop words which are common and carries little to no meaning and are not useful for processing. Some example of these words in English vocabulary are ‘a’, ‘an’, ‘and’, ‘are’, ‘it’, and ‘of’ [13]. These basic features will be the foundation of more complex features available inside the module. Depending on the application, these features combining with several algorithms will produce Language processing features which are Summarization Algorithm, Keyword Highlighting, Semantic Analysis and Language Detection.

1. Summarization Algorithm

The first feature included in the module is Text Summarization Algorithm. There are so many approaches and techniques introduced on this particular topic. For this project, a technique published by H.P Luhn in IBM Journal April 1958 is used. It is one of the simplest technique available. According to Russel (2013), the paper by H.P Luhn talks on a summarization technique which filters out sentences containing frequently appearing words that appear near one another. In another sense, this method is based principally upon sentence detection and frequency analysis within sentences [13].

The idea is that important sentences are the one that contains frequently occurring words. This is true except for stop-words which are frequently appearing but carries less significant. Hence, stop-words are filtered out. “In order to score each sentence, the algorithm in score\_sentences applies a simple distance threshold to cluster tokens, and scores each cluster according to the formula: ” [13]. Each sentence will be scored using the formula and later be filtered out using statistical threshold by computing mean and standard deviation for the scores obtained. Even-though this method is simple and performed well on most cases, its downturn is that it doesn’t consider sentences on a deeper semantic level.

1. Keyword Highlighting

Reading plain text is rather daunting and boring task for most human. In-order to counter this, NLP processing module also features a simple keyword highlighting algorithm. There are so many other methods available for this feature, each with their own set of rules of determining what are the important words with respect to individual sentences or even the whole text. However, for this module one of the simplest approach are employed. After tokenization, each individual words are tagged with POS. Words that are tagged with either NN (noun, singular), NNS (noun, plural), NNP (noun, proper singular) and NNPS (noun, proper plural) are considered as important. These tags are usually associated to the main topic in a specific sentence. After finish tagging, the text will be re-written with selected keywords bolded.

1. Semantic Analysis

This next feature aims to guess the sentiment of a text document as a whole determining whether it is positive or negative. This feature is useful when dealing with a bunch of long reviews to gain insight on something. This method relies on several dictionaries which are negative, positive, inverse, decrementing, and increasing list of words. Negative and positive dictionaries contain hundreds of respective tone words which are retrieved from a paper by Bing Liu et.al (2005). Inverse dictionaries contain words that invert a certain word sentiment for example ‘not’ and ‘would not’. Decrementing and increasing dictionaries contain words that lowers or increases semantic value of the words preceding it. Text input then will be tokenized to their individual words and compared with the available dictionaries and scored according to their positivity and negativity. The final value will be added up, positive value shows a positive sentiment and vice versa.

1. Language Detection

The last feature available inside this processing module is the language detection. The algorithm for this feature relies on the use of pre-defined stop-words inside NLTK library. The method behind this feature is that each tokenized word will be cross referenced with stop-words of each language available inside the library. The language with the most number of similarities is chosen as the natural language of the text. There are 13 available stop-word’s languages available which are Danish, Dutch, English, Finnish, French, German, Hungarian, Italian, Norwegian, Portuguese, Spanish, Swedish and Turkish. To account for Malay, I have created my own set of stop-words for Malay language. So, there are a total of 14 languages that can be detected by this feature.

**CHAPTER 4**

**RESULT AND DISCUSSION**

1. **Introduction**

This chapter will discuss what is the expectation by the end of this project. What is expected to be achieved and how to determine the success of this project. Performance evaluation will also be measured to gauge the usability and robustness of this system.

1. **Result and Analysis**
2. **Summarization Analysis**

For analyzing the summarization of textual document using this module, 10 BBC news articles are retrieved from their websites. The graph below shows percentage of summarization achieved by each article respectively. To conclude, the summarization percentage ranges between 65.78% to 82.01% with the average of 75.76%. Approximately ¼ of the total text remained for each summarization.

1. **Expected Outcomes**
2. Semantic Analysis

Semantic analysis deals with determining positive or negative tone of a text. Samples are taken from movie rating website called imdb. These samples are chosen from such websites because movie reviews are generally strongly bias towards one sided. In addition, star rating in such websites helps to define the sentiment for comparison. Review with rating below 4 are considered as negative while rating of above 7 are considered as positive. 50 negative reviews and 50 positive reviews are extracted randomly from several movies for sampling. From the results, we can clearly see that hit (accurate analysis) make up 83% of the total analysis. The percentage error of this feature for this particular case are 17% as shown.

1. Language Detection

This subtopic will be discussing the result of analysis of language detection feature in this NLP processing module. For each language, 5 random text containing mainly short stories are extracted online for analysis accounting for a total of 70 textual document. These texts are ran through the module and the resulting language detection are compared with the original language of the text. The result clearly shows a 100% accuracy for all languages detection.

The system should be able to automatically extract news articles directly after published on given websites and later classify the news into its respective category. This program is expected to be running in background all the time as to do classification in real time. This project can be considered as successful only if the accuracy of news classification rise above 80% of the time. In addition, the running program should not eat up too much computer resources such as RAM and CPU while also didn’t burden the internet traffic by much. This is also the indicator that the program is written and executed successfully.

**CHAPTER 5**

**CONCLUSION**

Natural Language Processing is not an entirely new field to be studied. The rapid growth of Artificial Intelligence had indeed gives us new perspective on solving problems that we deal with daily. NLP processing features introduced in this project only are only some of the possible applications in intelligent information management. This project is hoped to later paved the way how we deal with this vast volume of information called as Big Data in the near future.

# **REFERENCES**

|  |  |
| --- | --- |
| [1] | H. Brody, 08 December 2012. [Online]. Available: https://blog.hartleybrody.com/web-scraping/. |
| [2] | S. Ray, 22 October 2015. [Online]. Available: https://www.analyticsvidhya.com/blog/2015/10/beginner-guide-web-scraping-beautiful-soup-python/. |
| [3] | S. Bird, E. Klein and E. Loper, Natural Language Processing with Python, O'Reilly Media, 2014. |
| [4] | M. I. Rana, S. Khalid and M. U. Akbar, "News classification based on their headlines: A review," in *Multi-Topic Conference (INMIC)*, Karachi, 2014. |
| [5] | T. Joachims, Learning To Classify Text Using Support Vector Machines, Kluwer Academic, 2001. |
| [6] | Y.-H. Chen and S.-F. Li, "Using latent Dirichlet allocation to improve text classification performance of support vector machine," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, Vancouver, BC, 2016. |
| [7] | A. Dragland, 22 May 2013. [Online]. Available: http://www.sintef.no/en/latest-news/big-data--for-better-or-worse/. |
| [8] | D. Gutierrez, 5 June 2015. [Online]. Available: http://insidebigdata.com/2015/06/05/text-analytics-the-next-generation-of-big-data/. |
| [9] | J. Victor, 26 October 2011. [Online]. Available: http://www.sharemarketschool.com/how-does-news-affect-stock-prices/. |
| [10] | W. Zhou and C. Lili, "The algorithm of text classification based on rough set and support vector machine," in *Future Computer and Communication (ICFCC)*, Wuhan, 2010. |
| [11] | L. Cui, F. Meng and Y. Shi, "A Hierarchy Method Based on LDA and SVM for News Classification," in *Data Mining Workshop (ICDMW)*, Shenzen, 2014. |
| [12] | S. M. H. Dadgar, M. S. Araghi and M. M. Farahani, "A novel text mining approach based on TF-IDF and Support Vector Machine for news classification," in *Engineering and Technology (ICETECH)*, Coimbatore, 2016. |
| [13] | C. D. Manning, P. Raghavan and H. Schütze, Introduction to Information Retrieval, Cambridge University Press, 2008. |
| [14] | 31 August 2004. [Online]. Available: http://jmlr.csail.mit.edu/papers/volume5/lewis04a/a11-smart-stop-list/english.stop. |
| [15] | [Online]. Available: http://scikit-learn.org/stable/modules/svm.html. |
| [16] | S. P. Ponzetto, Knowledge acquisition from a collaboratively generated encyclopedia, Heidelberg: Akademische Verlagsgesellschaft, 2009. |