Natural Language Processing Engine For On-The-Go Application

Muhammad Hadi Asyrafi Bin Abdul Halim

Faculty of Electrical Engineering

Universiti Teknologi Malaysia

Johor Bahru, Malaysia*.*

hadiasyrafi94@gmail.com

Dr. Musa bin Mohd Mokji

Department of Electronic and Computer Engineering

Faculty of Electrical Engineering

Universiti Teknologi Malaysia

Johor Bahru, Malaysia.

*musa@fke.utm.my*

*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.

Keywords—provide 3-5 keywords separated by semicolons (;) eg: power system; fault analysis; Linear programming

# Introduction

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 combine 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.

# Ease of Use

## Selecting a Template (Heading 2)

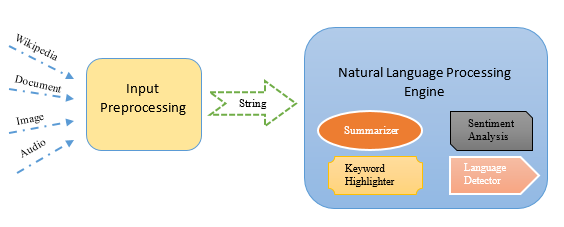
First, confirm that you have the correct template for your paper size. This template has been tailored for output on the A4 paper size. If you are using US letter-sized paper, please close this file and download the file “MSW\_USltr\_format”.

## Maintaining the Integrity of the Specifications

The template is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities. For example, the head margin in this template measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

# project methodology

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

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.

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. 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 |

## 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.

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 (1)” [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)

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 word 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.

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.

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.

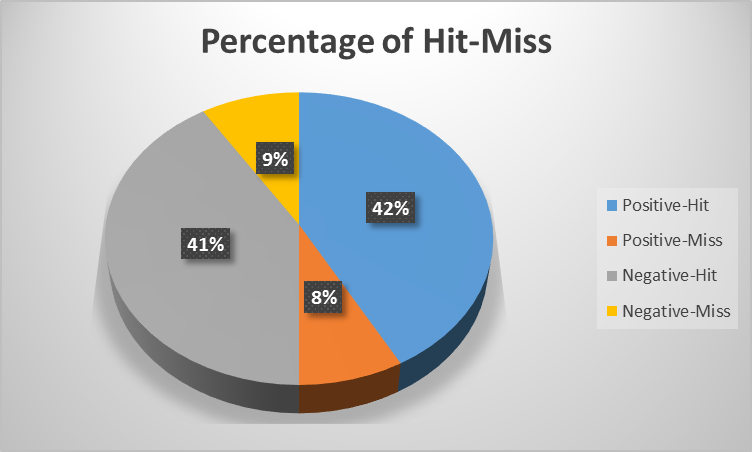
# result and analysis

## 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.

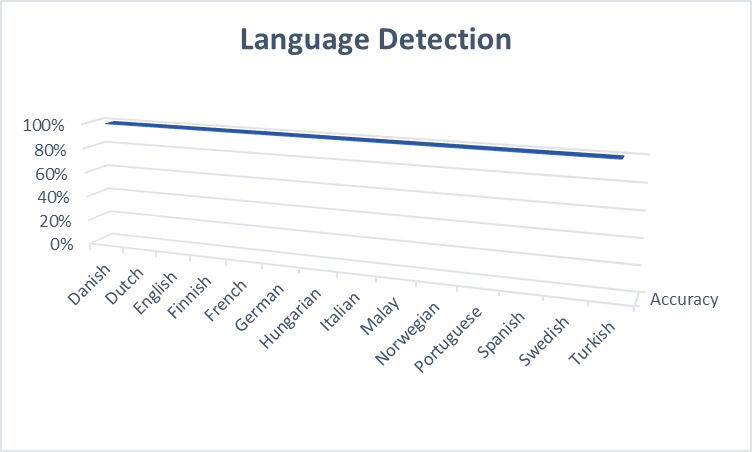
## Semantic Analysis

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.



## 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.

# 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 pave the way how we deal with this vast volume of information called as Big Data in the near future.

##### References

1. [1] H. Brody, 08 December 2012. [Online]. Available: https://blog.hartleybrody.com/web-scraping/.
2. [2] S. Ray, 22 October 2015. [Online]. Available: https://www.analyticsvidhya.com/blog/2015/10/beginner-guide-web-scraping-beautiful-soup-python/.
3. [3] S. Bird, E. Klein and E. Loper, Natural Language Processing with Python, O'Reilly Media, 2014.
4. [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. [5] T. Joachims, Learning To Classify Text Using Support Vector Machines, Kluwer Academic, 2001.
6. [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. [7] A. Dragland, 22 May 2013. [Online]. Available: http://www.sintef.no/en/latest-news/big-data--for-better-or-worse/.
8. [8] D. Gutierrez, 5 June 2015. [Online]. Available: http://insidebigdata.com/2015/06/05/text-analytics-the-next-generation-of-big-data/.
9. [9] J. Victor, 26 October 2011. [Online]. Available: http://www.sharemarketschool.com/how-does-news-affect-stock-prices/.
10. [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. [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. [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. [13] C. D. Manning, P. Raghavan and H. Schütze, Introduction to Information Retrieval, Cambridge University Press, 2008.
14. [14] 31 August 2004. [Online]. Available: http://jmlr.csail.mit.edu/papers/volume5/lewis04a/a11-smart-stop-list/english.stop.
15. [15] [Online]. Available: http://scikit-learn.org/stable/modules/svm.html.
16. [16] S. P. Ponzetto, Knowledge acquisition from a collaboratively generated encyclopedia, Heidelberg: Akademische Verlagsgesellschaft, 2009. 2005]