# TEXT SUMMARIZATION FOR TAMIL ONLINE SPORTS NEWS USING NLP

Thevatheepan Priyadharshan

168290F

Degree of Master of Science in Artificial Intelligence

Department of Computational Mathematics

University of Moratuwa
Sri Lanka

April 2019

# TEXT SUMMARIZATION FOR TAMIL ONLINE SPORTS NEWS USING NLP

Thevatheepan Priyadharshan

168290F

Thesis submitted in partial fulfilment of the requirement for the degree Master of Science

Department of Computational Mathematics

University of Moratuwa
Sri Lanka

April 2019

# Declaration

“I declare that this is my own work and this dissertation does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any university or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where due the acknowledgement is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my dissertation, in whole or part in print, electronic or other medium. I retain the right to use this content in whole or part in future works.”

Signature: Date:

The above candidate has carried out research for the Masters dissertation under my supervision.

Signature of the supervisor: Date:

## Abstract

Text summarization plays an important role in natural language understanding and information retrieval. Presently automatic text summarization getting much more attention by people because it is efficiently and effectively serving time in decision making process even in day to day life. Many approaches such as statistical based, machine learning based approaches have been presented by researchers where statistical based approaches are less semantic consideration in terms of forming summary and most machine learning approaches are language independent. Presently neural network models get more attention than the traditional approaches. There are few statistical based approaches that are presented for Tamil text summarization with less natural language processing.  The primary objective of this research work is to propose a methodology to address the problem of summarization for Tamil sports news which can automatically create extractive summary for the news data with the use of Natural Language Processing (NLP) and a generic stochastic artificial neural network.

The sports news gathered from different resources has been given as input to the system where most relevant sentences will be extracted from the text and presented as an extractive summary to the input text. The input will go through six sub process such as pre-processing, feature extraction, feature vector matrix, feature enhancement, sentence extraction and summary generation. Where in the pre-processing the sentences will be initially tokenized. After this set of stop words will be removed from the tokenized output, finally named entities available within the text will be tagged such as person's name, location name, date, numeral. After pre-processing, feature extraction will be executed where features such as sentence position, sentence position related to paragraph, number of named entities, term frequency and inverse document frequency and number of numerals are employed to generate a score against each sentence available in the text.

By using these scores in feature vector matrix sub process, feature matrix will be generated for the whole text where each feature score values for each sentence available in the text is arranged in the row of the matrix. This feature vector matrix will be given as an input to the Restricted Boltzmann Machine which is embedded in the feature enhancement sub process to generate the enhanced feature matrix. After obtaining the Enhanced feature matrix in the sentence extraction process, row values are summed where it gives the summed enhanced feature values for each sentence, after this high score sentence will be extracted as the most relevant sentence to form the summary where it is considered as a sub set of sentences in the summary. And at last the summary generation process will be executed where most relevant sentence selected in the sentence extraction module will be used for cosine similarity measures with the other existing sentences in the text and another sub set of sentences will be extracted from the text to form the summary, likewise the process will be done. Finally, the sentences will be ordered as in the order in the text and extracted summary of the text will be presented. This summary generation will happen on real time by using different resources.

A comparative evaluation has been done for the text summarization systems’ result. For evaluation purpose, 30 news data set has been used, where each summary regarding to each news data set, has been evaluated by 3 Tamil speaking human assessors. Each news has been distributed among those evaluators and they have to read the news data and they have to select the sentences which will form the summary, likewise the responses for each news data set has been gathered. In the experiment, each and every summary generated by the system has been evaluated against the human generated summary and the average F-measure of this text summarization system is 76.6% which is higher than the existing approaches for the Tamil text summarization approaches.

Keywords—Extractive summarization, Natural Language Processing, Neural network, Restricted Boltzmann Machine, Feature matrix, F-measure.

## Acknowledgements

First and foremost, I would like to express my sincere gratitude to my research supervisor Dr. Sagara Sumathipala for his valuable guidance extended throughout the research. This research would not have been completed to success without his immense support and guidance. Further, I would like to thank Prof. Asoka Karunananda who gave valuable guidance to the documentation of the work during the lectures. And also, I would like to thank all academic staff of the Department of Computational Mathematics who gave sufficient knowledge to complete this research.

Last but not least special thanks should be given to my family members and all of my batchmates for extending their supportive hands of friendship towards the successful drive of the research.

# Table of Contents

# LIST OF FIGURES

page

# LIST OF ABBREVIATION

| Abbreviation | Description |
|---|---|
| NLP | Natural Language Processing |
| RBM | Restricted Boltzmann Machine |
| PC | Personal Computer |
| Tf-IDF | Term frequency and Invers Document Frequency |
| DUC | Document Understanding Conferences |
| ROUGE | Recall Oriented Understudy for Gisting Evaluation |
| DNN | Deep Neural Networks |
| LNS | Language Neutral Syntax |
| SOP | Subject Object Predicate |
| SVM | Support Vector Machine |
| FFNN | Feed Forward Neural Network |
| MR | Mathematical Regression |
| PNN | Probabilistic Neural Network |
| DE | Differential Evolution |
| MRF | Markov Random Field |
| BM | Boltzmann Machine |

# LIST OF APPENDICES

# CHAPTER 1
# INTRODUCTION

## 1.1. Prolegomena

Document summarization is an important problem that has many applications with the evolution of modern information technologies [1], we have huge amount of electronic documents on the web which are generated day by day. We have enough data on the web but the problem is knowledge starving from data. For example, in business most of the companies are now struggling to deal with billions of past business records in an efficient way to forecast future business trends. Not only that, nowadays people are using web to gather others opinions before making decision or buying anything. At that time, people are very much interested to read summaries rather than reading whole documents as they get bored and finding useful and favourite documents from huge text repositories creates significant challenges for users [2].

Summarization of a document becomes as a key problem in many applications in information retrieval and natural language understanding[1]. These summarization techniques are primarily categorized into two categories such as extractive based summarization and abstractive based summarization [3]. By using one or more text a summary can be generated as an abstract view of the main text while keeping the main points which are addressed by the main text without much information loss. When this job is done automatically, it can be said as automatic text summarization which also a solution for information overload problem.

## 1.2. Aims & Objectives

The aim of this study is to develop a computer-based solution to address the problem of summarization for Tamil sports news which can automatically create extractive summary for the sports news readers.

In addition, the following listed are the major objectives related to the research:

- Critical study on text summarization systems, and how it’s applied to create extractive summaries.
- Critical study of the literature on how to use different technical approaches to solve the text summarization problem.
- Development of a fully functional solution using NLP and artificial neural network as the principal technology of the system.
- Critical study on how text similarity measures are used while creating a summary for a give text regarding to similar information.
- Critical evaluation of the implemented system using appropriate measures and comparative analysis of this approach with traditional approaches.
- Writing a research paper and a conference paper on the project.
- Producing the final documentation.

## 1.3. Background & Motivation

Summary is considered as the most prominent information in the document, simply we can say that is the abstract view of the document. Majority of the literature on document summarization is dedicated to extractive summarization [1] and relatively less attention has been paid to abstractive summarization because it needs more language related knowledge. Traditional extractive summarization can be broadly classified into greedy approaches [4], graph-based approaches [5], and constraint optimization-based approaches [6]. In recent times, neural network-based methods have become popular for this extractive summarization [7]. In traditional extractive summarization methods used keywords to extract the summary from the text, where longer sentences selected as the summary and extractive summary is fully dependent on the keyword extraction and also duplication of the keywords give away to make erroneous outputs. Because of this important information where the text is carrying is omitted.

Generating extractive summaries without omitting important information for a text is very much useful for the user. Recent extractive summarization approaches give more insight towards semantics of the text, machine learning approaches also used in this regard. Presently neural network-based approaches have become popular.

## 1.4. Problem in Brief

In English, like language neural network based extractive summarization gives promising results in extractive summarization with paid attention to the semantics. Efforts has been given towards semantics using semantic graph-based methods [8] in Tamil language but less effort have been given in Tamil language in terms of using neural network. This research is paying attention towards on neural network and NLP technologies to create extractive summaries for Tamil sports news providing the gist of the text to the user.

## 1.5. Proposed Solution

### 1.5.1. Hypothesis

In order to address the text summarization problem using the NLP and neural network-based solution it will efficiently handles the challenges and the opportunities in generating extractive summaries comparing to other technologies.

Following discussed are the identified inputs, outputs and the features of the system:

### 1.5.2. Inputs

- Tamil sports news from different resources

### 1.5.3. Outputs

- Extractive summary of the sports news

### 1.5.4. Process

The main process of this system can be divided into 6 phases such as pre-processing phase, feature extraction phase, feature vector matrix generation phase, feature enhancement phase, sentence extraction phase and finally the summary generation phase.

- Pre-processing phase- Raw sports news data will be fed as input into this phase, tokenization, stop word filtering will be done here and furthermore named entity recognition will be applied to the text and the NER tagged text will be as the output of this phase.
- Feature extraction phase- Features such as sentence position, number of named entities, sentence position regarding to paragraph, term frequency and inverse document frequency and number of numerals are considered regarding the sentence which is in the text and this gives the way to generate the feature vector matrix for each sentence available in the text.
- Feature vector matrix- A vector matrix regarding to each sentence in the text, will be generated with the consideration of features considered in the feature extraction phase, therefore every sentence available in the text will have a vector matrix.
- Feature enhancement phase- $5\times$ n (n= number of sentences) feature vector will be given as the input to the Restricted Boltzmann Machine (RBM) [9] which is a generative stochastic artificial neural network to improve the weighted values of the feature vector of each sentence. Enhanced feature matrix will be the output of this phase.
- Sentence extraction phase- Based on the output of the feature enhancement phase each sentences' score will be calculated by adding the five feature values which is obtained from the feature enhancement phase. Based on this score a sentence will be extracted from the text as to form the summary of the text.
- Summary generation phase- Then this extracted sentence in the previous phase will be used to compare the similarity of other sentence available in the text to from the extractive summary. Finally, an extractive summary will be generated for the sports news.

### **1.5.5.** Features

- Extractive summaries will be created for each sports news.
- Summaries will be created in real-time.

- Using different resources to create extractive summaries regarding to sports news.

### 1.5.6. Users

- Sports news readers.
- Text summarization researchers.

## 1.6. Resource Requirements

Following are the resource requirements that would be necessary for continuing work on this project.

### 1.6.1. Computer hardware requirements

- PC/Laptop with Intel i5 or i7 Processor, minimum 8GB of RAM

### 1.6.2. Software requirements

- Software is expected to run on platforms above Microsoft Windows 7
- PyCharm Community Edition
- Python
- JAVA with jdk

### 1.6.3. Other requirements

- Sports news and their relevant extractive summaries.

To do an experiment on the system, sports news was used to generate extractive summaries. So considerable amount of sports news will be gathered.

## 1.7. Structure of the Thesis

The structure of the thesis is as follows: Chapter 1 provides the brief introduction to the proposed solution of the study. Chapter 2, provides a critical review of the developments and issues in the area of text summarization by defining the research problem and identification of technologies. Chapter 3, the technology chapter elaborates on the different technologies identified and utilized in the proposed system and the chapter 4, the approach chapter will provide an overall image of the proposed solution for the text summarization system.

The Design and Implementation chapters providing the advance details on how the system has been designed and implemented using the proper architectures and technologies according to the approach discussed in the previous chapter. Finally, the Evaluation chapter compare and contrast the performance of the system regarding to the other approaches exists and finally the Conclusion chapter provides the final outcome on the success of the project and provides its advantages as well as the technical issues and opportunities.

## 1.8. Summary

In this chapter, introduction has been given to the entire project. We have presented our research problem, aims & objectives, technology adopted, proposed solution and resource requirements. The next chapter provides a detailed critical review of text summarization problem.

# CHAPTER 2
# TEXT SUMMARIZATION – DEVELOPMENTS & CHALLENGES

## 2.1. Introduction

A text summarization system generates summary, where the gist of the information regarding the text document has been communicated by the compressed version of the document. The purpose of this kind of summarization system facilitate quick and accurate identification of the gist of information, because of this it makes easy to find useful information and also it saves the time and effort in finding useful information in a given text document. The first text summarization approach has been presented in late fifties by Luhu[10]. Up to now there is a great improvement in this field. Many approaches have been developed by using different technologies such as static approaches, machine learning approaches, ect. in this field [11]. Summaries generated by these text summarizers should contain the most appropriate information in a text and simultaneously, it should have less space than the original text.

With the growth of recent information technologies, a number of electronic documents on the web which are generated day by day are gradually increasing and users are struggling to find relevant information with minimal usage of resources, because of this need for the text summarization has emerged. And also, with the large number of documents available with redundancy in the texts, users are getting bored and they omit reading useful and interesting information in the document and this also makes a strong need for the text summarizer.

Depending on the documents that has to be summarized, output (if extract or abstract is required), regarding on the purpose, availability of training data and language of the document, the summarization technique will be classified as single and multi-document summarization, extractive and abstractive summarization, generic and query focused summarization, supervised and unsupervised classification and mono, multi and cross-lingual summarization.

However this summarization is still challenging [12] because of redundancy in information, cohesion in text, order of the sentence, coverage of information, covering the gist of information, etc. This makes this summarization task more complex.

## 2.2. Classification of Text Summarization

Mainly the text summarization techniques are classified into two categories such as extractive based summarization and abstractive based summarization [3].

Based on the number of documents to be summarized, summarization technique can be classified as single and multi-document summarizations [13]. If the summary is generated from a single document this technique is called as single document summarization whereas in multi-document summarization, many documents are used for summarization purpose. Summarization technique can also clustered as generic or query-focused summarization technique[14]. If the summarization technique is included the query related content it is called as query-focused summarization technique and in the generic summarization technique uses the general information available in the document.

Summarization task can be classified either supervised or unsupervised [15]. For supervised task training data is needed. Large amount of labelled or annotated data is needed for learning techniques. But unsupervised summarization task doesn’t require any training data, by accessing the target document it generates summaries. Clustering technique has been employed by these unsupervised systems. If the summaries tell about what the document is about then it can be classified as indicative summarization technique, whereas informative summarization shares the information about the topic of the document.

Depending on the language of the document the text summarization technique can be classified as mono-lingual, multi-lingual and cross-lingual summarization techniques. Where the document and its summary contain only one language it is called as mono-lingual, if the document and its summary contain more than on language it is called multi-lingual and if the document is in different language and the summary is in another language it is called cross-lingual text summarization technique.

Furthermore, the summarization can be cluster [16] as web-based summarization[17], E-mail based summarization, Personalized summaries, opinion summaries and survey summaries.

## 2.3. Extractive Summarization Approaches

An extractive summarization technique is done by selecting important sentences from the original source document and presented as the summary for that particular document where the length of the summary of that document depends on the compression rate. Compared to abstractive summarization it is a simple and robust method for text summarization.

### 2.3.1 Statistical based approaches

A hybrid approach has been presented[18] using statistical sentence extraction approaches combining with contextual information with the bi-gram pseudo sentence for single and multi-document summarization, where contextual information has been used to handle feature sparseness problem. Because of the proposed methodology, it is independent from the language of the text document where it is applicable for many languages but it is less meaningful in terms of information redundancy and cohesion.

Another statistical approach has been presented by Fattah [13] where statistical features such as position of sentence, positive keyword, negative keyword, centrality of sentence, resemblance of sentence to the title, relative length of the sentence, presence of numerical data in the sentence, presence of name entity in the sentence, sentence’s bushy path, aggregated similarity, etc. For each sentence in the document, a score is calculated and highly scored sentences are extracted for forming the summary. Positive, negative keyword calculation and done based on Tf-Idf (term frequency and inverse document frequency). Scores are assigned to the sentences based on this feature weights, where highest scored sentences are extracted to generate the summary.

The common architecture diagram of the statistical based text summarization is depicted below in figure 2.1.

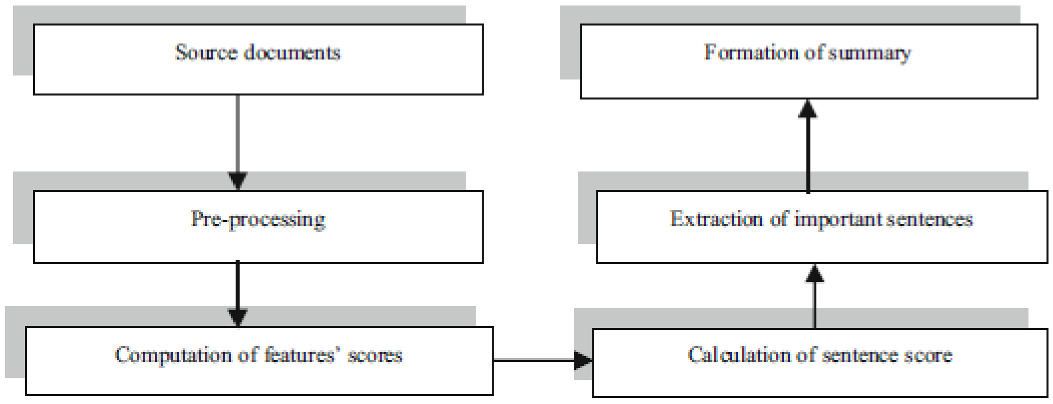

Source documents
Pre-processing
Computation of features' scores
Calculation of sentence score
Extraction of important sentences
Formation of summary

*Figure 2.1 Common architecture of statistical based approach (Source Gambhir 2016)*

### 2.3.2 Graph based approaches

An extractive, graph-based unsupervised technique for summarizing single documents has been proposed by Praveen[19] where it considers three important properties of summarization, such as importance, non-redundancy and local coherence. Input text document is represented by means of a bipartite graph consisting of sentences and entity nodes. A graph based ranking approach has been utilized on this graph to calculate the rank of the sentences based on their importance. Finally, by optimization process summary is made non-redundant and locally coherent. This approach is an unsupervised technique no training data doesn't depend on any parameter and good coherent is achieved, but it is capable in generating single document summarization.

Tamil document summarization using semantic graph [8] has been presented, where semantic features of the text are identified using Logical Form Parser, named entity and feature extraction. Document Graph is drawn for the nodes representing noun clause and main clause. Advantage of this system is that Language-Neutral Syntax (LNS) is the system used for performing semantic analysis of the input here. Subject-Object-Predicate triples created here for each sentence and using that a semantic graph will be generated. After applying semantic normalization (for reducing noises and redundancies) a support vector machine (SVM) is used for extracting relevant sentences as summary.

### 2.3.3 Support-Vector- Machine

The paper presented by M. S. Patil [20] , suggested a summarization system based on several extractive text summarization approaches, and on the Support-Vector-Machine(SVM). This system implements a machine learning algorithm to the summarizing system which trains the system every time a document is given to it, so that the summary is better each time and the system tries to improve the performance and quality of the summary generated by the clustering technique by cascading it with SVM.

### 2.3.4 Neural network-based text summarization

Nallapati [1] presented a two-layered recurrent neural network-based sequence model to address the extractive summarization problem for the DUC 2002 corpus. A binary decision is made regarding each sentence in the original document whether to include in the summary or not. Abstractive features such as absolute position, relative position, salience, content, novelty were used in the evaluation to predict the probability of each sentence in the document. This model performs better than the other deep learning models. ROUGE-1, ROUGE-2 and ROUGE-L=0.343 measure has been used [ROUGE-1=0.422, ROUGE-2=0.173 and ROUGE-L=0.343] in the evaluation and it performs better than other existing models.

Another paper proposed a new unsupervised frame work using Deep Neural Networks (DNN) [21] for document summarization. In this model, the word count vectors were created and those were fed as inputs in the input layers of the DNN. In this work the vectors weights were enhanced by using RBM in the Fine-tuning the weight stage which is processed after the Pre-training the weights stage. ROUGE toolkit has been used in this research work for the evaluation purpose.

### 2.3.5 Combination of genetic algorithm and neural network

Automatic text summarization using genetic algorithm and neural network model [13] has been presented where statistical feature like Position of Sentence, positive, negative keyword, Resemblance of sentence to the Title, Centrality of Sentence, Presence of Name Entity in sentence, Presence of Numbers in sentence, Bushy Path of sentence, Relative Length of sentence, and Aggregate Similarity are considered in the selection process. A weighted score function corresponding to a sentence is computed with regarding the above mentioned feature. Ranking is done after this and high score sentences are used to generate the summary with the 10,20,30% compression rates. Genetic algorithm and Mathematical Regression (MR) models are trained for getting an appropriate mix of feature weights. For sentence classification Feed Forward Neural Network (FFNN) and Probabilistic Neural Network (PNN) are used. The evaluation shows that different features are vital in generating summaries in different kinds of documents considered.

The proposed automatic summarization model is presented below in Fig. 2.2.

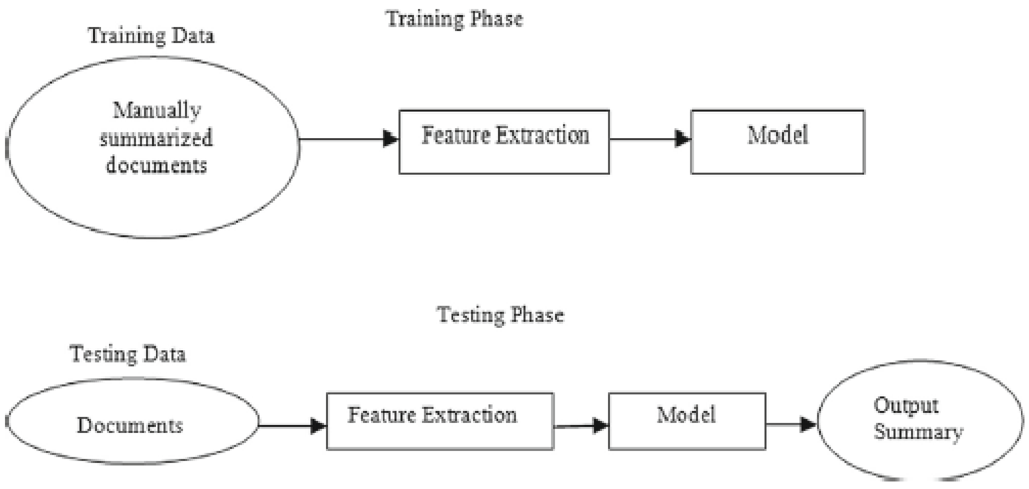


*Figure 2.2 Proposed automatic summarization model by Fattah (Source Fattah 2009)*

### 2.3.6 Evolutionary algorithm

OCDsum-SaDE [22] a  multi document summarization technique which uses evolutionary optimization algorithm that deals simultaneously with content coverage and redundancy. By using an algorithm named as self-adaptive differential evolution (DE), the optimization problem has been handled. This method focuses on all the three features of summarization such as content coverage, diversity and length. This approaches the redundancy in the summaries generated and includes desirable content form the original document but the implementation of this approach is disturbed by the runtime complexity of the differential evolution algorithm.

A memetic algorithm population-based search of evolutionary algorithm with directed local search strategy for the single document extractive summarization technique has been (MA-SingleDocSum) [23] proposed where the extractive summary generation problem has been addressed as a binary optimization problem. Features such as position of sentence, resemblance of sentence with the title, sentence length, cohesion and coverage are considered in this approach. The features considered are domain independent, where multi-bit mutation encourages information diversity. But information redundancy exits in the summary.

### 2.3.7 Phrase-based compressive cross-language summarization

Phrase-based compressive cross-language document summarization [24] has been presented. In this approach source document language and the summary generation language are different. Phrase-based machine translation model has been used with the scoring function. Greedy algorithm has been used to handle the scoring function which has been used in the summarization.

## 2.4. Abstractive Based Approaches

Another major classification of text summarization is abstractive text summarization where it produces an abstract summary which includes words and phrases different from the original source document. Therefore, the main ideas or concepts in the original source document will be taken into consideration in abstractive summary and the idea will be reinterpreted by other words in the languages and shown in a different form. So natural language processing will be extensively used. And also, it is much more complex than extractive summarization. Below, few papers are discussed.

A paper presented for Opinosis  [25], an abstractive summarization system where it used graphs for producing brief abstractive summaries using shallow NLP without any domain knowledge for highly redundant opinions. For evaluation reviews regarding on hotels, cars and various other products have been used where the results show that output generated by Opinosis have reasonable agreement with human summaries.

A data-driven approach for generating abstractive summaries has been presented [3] by adopting neural machine translation technique where the neural attention-model is combined with a contextual input encoder. Based on the input sentence each word for the summary has been generated. Evaluation has been done on the DUC 2004 data set using ROUGE-1, ROUGE-2 and ROUGE-L measures.

A sentence-based abstraction technique has been proposed[26] where the model uses linguistic extraction techniques to generates summary which extract the sentences from the relevant portion of the document. Textual continuity has been represented by the discourse. A smallest discourse segment such as a discourse network has been created and it is used to combine segments together. This discourse network considers the sentence boundaries and also considers text which is constructed by interrelated parts as a single unit. Cohesion factors such as referential cohesion, lexical cohesion and verb cohesion are considered in this technique and also to identify the coherence relation in the text Rhetorical structure theory has been employed. Where these cohesion and coherence are used to measure the discourse continuity. And also, connection between sentences in the nearest segments is represented by cohesion.  This technique improves the information retrieval where it correlates the semantically relevant sentence.

Figure 2.3 shows the high-level architecture of the model presented in the above paper,

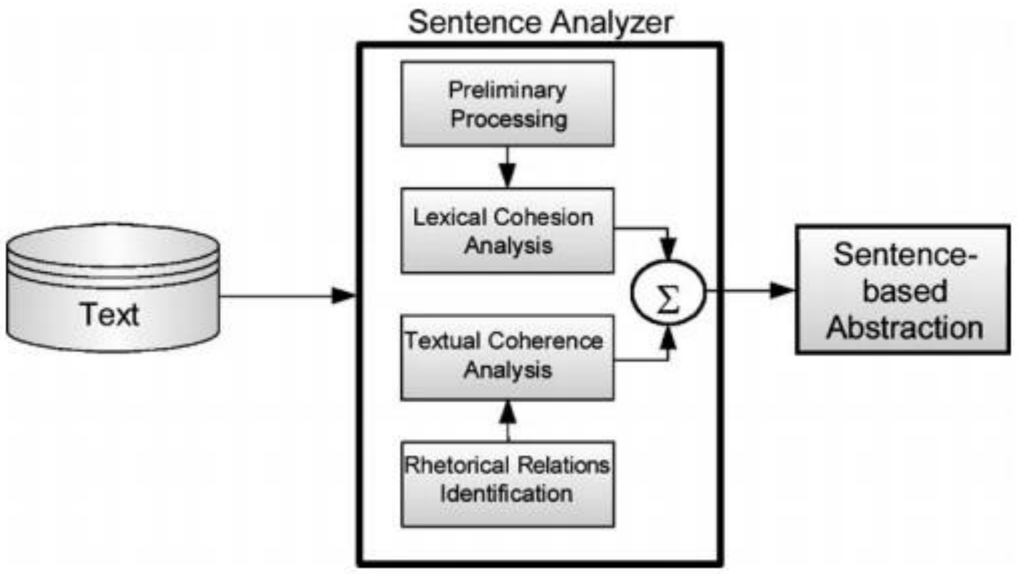

Sentence Analyzer
Preliminary Processing
Lexical Cohesion Analysis
Textual Coherence Analysis
Rhetorical Relations Identification
Text
Σ
Sentence-based Abstraction

*Figure 2.3 Schematic diagram of the model (Source Chan 2006)*

Another abstractive summarization paper has been presented for generating summaries of multi documents using integer linear programming [27], that maximizes content regarding to information and linguistic quality and reducing redundancy in the final summary. Initially most important document from the multi-document set has been selected and each sentence from those documents is used to generate separate clusters. Similarity measures have been used to assign sentences in other documents for different clusters and K-shortest paths are generated for sentences of each cluster by utilizing word-graph structure. At last, sentence selection have been done based on the set of shortest paths. Experiments have been done on DUC 2004 and DUC 2005 datasets. The ROUGE-2, ROUGE-L, ROUGE-SU4 measures are used for evaluation.

## 2.5. Future Directions in Text Summarization

Because of the evolution in the technology many kinds of documents such as multi-language based, multimedia content-based documents are created day by day. Therefore, the text summarization system has to put effort in this regard. This future direction section gives the details in this regard.

Text summarization starts with single document summarization, later on multi document summarization came into to existence because of large number of documents in different formats available in the web. Because of the fast development of the technology multi language, documents that contain multimedia have come into existence, So the text summarization has to consider these in their summarization process. And also, the features that most of the text summarization consider is based on term frequency, position etc.[16]. Therefore, some new features need to be discovered which can help to extract sentences which are semantically important from the text document.

Because of the availability of multi-language documents, text summarization can also consider these kinds of document in the summarization process. Because of less attention paid by the researchers for different languages regarding on linguistic this text summarization is not effective in those scenarios, so a text summarization based statistical approach will be given as a way to help in this to provide a good summary that matches with the human summary.

And also, there is a need for the combination of abstractive as well as extractive summaries, to address this issue. Hybrid approaches can be utilized to generate a good quality summary by concatenating extractive and abstractive approaches together.

## 2.6. Summary

In this chapter, it has been deliberated on the various topics related to the technologies and literature of this research domain. Initially, the introduction part covers the history of text summarization approaches used in greater detail, and subsequently, the need for text summarization covers application of this text summarization. Later on, under the third, topic various types of text summarization approaches such as extractive and abstractive approaches have been discussed. Finally, future directions of the text summarization approaches have been discussed. In the next chapter, the technology that is used in this system will be discussed in greater detail.

CHAPTER 3

# TECHNOLOGY

## 3.1 Introduction

The main technology used in the proposed system is Restricted Boltzmann Machine which is a generative stochastic artificial neural network model based on latent variables that can learn probabilities over its set of inputs. Its applicability propagates to a wide variety of problems and locales in the past few years.

From binary inputs, RBM have been prolonged to model various types of input distributions [28] [29]. Conditional versions of RBMs have also been developed for combined filtering [30] and to prototypical motion capture data [31], topic modelling and video sequences [32]. RBM have been particularly successful in classification problems either as feature extractors for text and image data [33] or as a good initial training step phase for deep neural network classifiers [29]. Furthermore RBM can be used as stand-alone non-linear classifier together with other standard and more popular classifiers, instead of simply being considered as modest feature extractors [34].

## 3.2 Restricted Boltzmann Machines (RBM)

RBMs are light, two-layer neural nets that constitute the building blocks of deep-belief networks. First layer of the RBM is called the visible layer, or input layer, and the second is the hidden layer. Neuron like nodes are depicted as circles in the below mentioned figure where the calculation are take place. Nodes are connected to each other across layers but nodes in the same layer are not connected, because of this there is no communication between the same layers, this is the restriction in Restricted Boltzmann Machine. Each node is a locus of computation that processes input, and begins by making stochastic decisions about whether to transmit that input or not. Where each visible node takes a low-level feature from an item in the dataset to be learned.

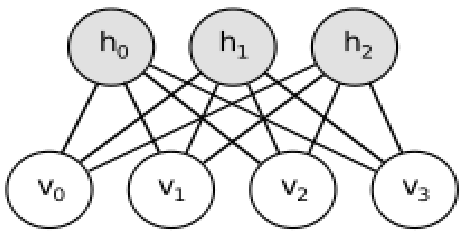


*Figure 3.1 Visible and hidden layers of Restricted Boltzmann Machine*

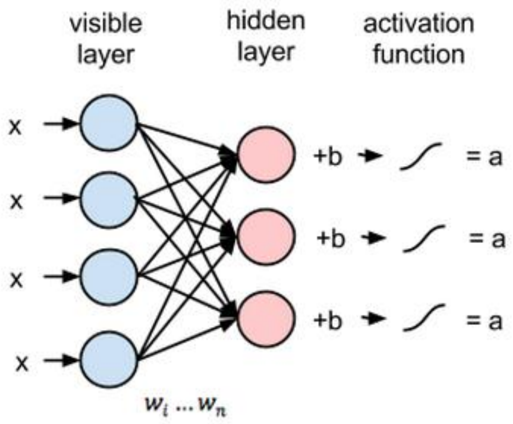


*Figure 3.2 RBM Activation Function*

Each hidden node receives the inputs X multiplied by their respective weights W. The sum of those products is again added to a bias and the result is passed through the activation function producing one output for each hidden node.

*Activation f ((weight w * input x) + bias b) = output a.*

Boltzmann Machines (BMs) are a particular form of log-linear Markov Random Field (MRF), i.e., for which the energy function is linear in its free parameters. To make them powerful enough to represent complex distributions (i.e., go from the limited parametric setting to a non-parametric one), it can be considered that some of the variables (hidden variables) are never observed. By having more hidden variables or units, it can increase the modelling capacity of the Boltzmann Machine (BM).

### 3.2.1 Energy based models with hidden units

Energy- based probabilistic models define a probability distribution through an energy function, as follows:

$$p(x) = \frac{e^{-E(x)}}{Z} \qquad (1)$$

Z is called the partition function by analogy with physical systems.

$$z = \sum_x e^{-E(x)} \qquad (2)$$

Introducing some non-observable variables to increase the expressive power of the model, where $x$ denoted as observable part and $h$ denoted as hidden part. The above equation 1 will be as follows:

$$p(x) = \sum_h p(x,h) = \sum_h \frac{e^{-E(x,h)}}{z} \quad (3)$$

Samples used to estimate the negative phase gradient are referred to as negative particles, which are denoted as $\mathcal{N}$ . The gradient written as follows:

$$-\frac{\partial \log p(x)}{\partial \theta} \approx \frac{\partial \mathcal{F}(x)}{\partial \theta} - \frac{1}{|\mathcal{N}|} \sum_{\tilde{x} \in \mathcal{N}} \frac{\partial \mathcal{F}(\tilde{x})}{\partial \theta}. \quad (4)$$

The energy function ***E(v,h)*** of an RBM is defined as:

$$\boldsymbol{E(v,h) = -b'v - c'h - h'Wv} \quad (5)$$

Where visible layer denoted by v, hidden layer denoted by h, $W$ represents the weights connecting hidden and visible units and $b$, $c$ are the offsets of the visible and hidden layers respectively.

This interprets directly to the following free energy formula:

$$F(v) = -b'v - \sum_i log \sum_{h_i} e^{h_i(c_i + W_i v)} \quad (6)$$

As of the particular structure of RBMs, hidden and visible entities are conditionally independent given one-another. Using this property, we can write:

$$p(h|v) = \prod_i p(h_i|v) \quad (7)$$

$$p(v|h) = \prod_j p(v_j|h) \quad (8)$$

### 3.2.2 RBMs with binary units

In the commonly studied case of using binary units (where $v_j$ and $h_i \boldsymbol{\epsilon}$ {0, 1}), we obtain from Eq. (5) and (3), a probabilistic version of the usual neuron activation function:

$$p(h_i = 1|v) = sigm(c_i + W_i\, v) \tag{9}$$

$$p(v_j = 1|h) = sigm(b_j + W'_j\, h) \tag{10}$$

The free energy of an RBM with binary units further simplifies to:

$$F(v) = -b'v - \sum_i \log(1 + e^{(c_i + W_i v)}) \tag{11}$$

### 3.2.3 Update equations with binary units

Combining Equations. (4) with (11), we obtain the following log-likelihood gradients for an RBM with binary units:

$$\begin{aligned}
-\frac{\partial \log p(v)}{\partial W_{ij}} &= E_v[p(h_i|v) \cdot v_j] - v_j^{(i)} \cdot sigm(W_i \cdot v^{(i)} + c_i) \\
-\frac{\partial \log p(v)}{\partial c_i} &= E_v[p(h_i|v)] - sigm(W_i \cdot v^{(i)}) \\
-\frac{\partial \log p(v)}{\partial b_j} &= E_v[p(v_j|h)] - v_j^{(i)}
\end{aligned} \tag{12}$$

## 3.3 Natural Language Processing

Natural language processing techniques used in this system in (pre-processing) development process, where semantics is important in text summarization tasks. This natural language processing has been grabbing the importance in many such areas like speech recognition, natural language understanding and natural language generation. NLP has been used in major evaluation tasks such as syntax identification, lexical semantics identification, named entity recognition, speech recognition, etc…

In this research named entity recognition has been used to identify the named entities like person, organization, location and numerals. Stanford named entity tagger has been modified and used for this named entity recognition purpose. In the pre-processing phase named entity recognition tagger has been used to identify named entities where it is used for feature extraction purpose.

## 3.4 Summary

In this chapter, technologies used in this research have been explained. A greater detail of Restricted Boltzmann Machine has been explained, where the graphical description of RBM  and detailed description of energy-based models are presented in this chapter and description of natural language processing technique part is also presented. In the next chapter, the approach of the research will be discussed in greater detail.

CHAPTER 4

# APPROACH

## 4.1 Introduction

Previous chapter explained the detailed description of the technology used in this work and this chapter gives the description of the approach of this research work, furthermore it gives the details of hypothesis, input, output, process, features and users of this research work. Finally, it concludes with the summary.

## 4.2 Hypothesis

The main hypothesis of this project is, implementing a neural network with natural language processing based solution. It will efficiently handle the text summarization problem in Tamil sports news domain where the gist of the news will be effectively extracted from the text, and it overcomes this issue in the traditional approaches in text summarization problem, where the traditional approaches were inefficiency in extracting the semantics of the text.

Following are the identified inputs, outputs and the features of the system which are discussed below:

## 4.3 Inputs

- Tamil sports news from different resources gathered will be used as the input to the system.

## 4.4 Outputs

- System will generate extractive summaries for the sports news where it is given as the input to the system.

## 4.5 Process

The main process of the text summarization system includes the following steps:

- Raw sports news will be fed as input into this step, where the text is tokenized into sentences and words for sentence/ word boundary disambiguation, furthermore regular expressions will be used to detect the character structures such as punctuations, time, date, numbers, etc. After that stop word filtering will be done here. For this purpose a set of stop words like "ஒரு" (oru), "என்று"(enru), "மற்றும்" (marrum), etc has been used, furthermore named entity recognition tagger has been applied to the sentence which were tokenized. The NER tagged text will be as the output of the first phase such as pre-processing phase.

- The second phase such as feature extraction phase where features such as sentence position, sentence position regarding paragraph, number of named entities, term frequency and inverse document frequency and number of numerals are considered regarding to the sentence which is in the text and this gives the way to generate the feature vector matrix for each sentence available in the text. Feature and their relevant calculation are mentioned below.
  - Sentence position has been used because it plays an important role in the text summarization [2]. Usually first few sentences carrie the main idea of the text, last few sentences carries the conclusion regarding the text and rest carries the supportive details for the main idea of the whole text.

    *Score* (Sentence Position) = 1; if first or last sentence position
    cos ((Sent_Position - min) ((1/max) - min)); else (*01*)

    Where, *Sent_Position* = position of sentence in the text, "*N"* is the total number of sentences in the document, "*th"* is the thresh hold, where it is equal to 0.2 x N, *min* = *th* x *N* and *max* = *th* x *2* x *N*.

  - Sentence position regarding the paragraph[13] is planned to use here because at the beginning of each paragraph it carries a new piece of information and at the end of the paragraph it carries a conclusion.

    *Score (Para Position)* = 1; if the first or last sentence of a paragraph
    0; else. (*02*)

- Number of named entities, sentences which are locating named entities like person names, organizations, locations, time expressions, etc... are mostly used to draw the main point in a sentence which is used to give the meaning of the entire text. Here, the total number of named entities in the sentence have been used.

  *Score (Named entities)* = # No. of named entities (*03*)

- Term frequency and inverse document frequency has been used here to elaborate the frequency of word that is intended to reflect how important a word is to a document in a collection or corpus. This measure is calculated for each sentence by using the equation 04.

  $$Score\ (tf.idf\ (t, d)) = tf\ (t, d) \times \log(N/df\ (t)) \qquad (04)$$

  Where, the term frequency, *tf (t, d)* is the number of occurrences of "*t" (term) in* "*d" (document),* the document frequency *df (t)* is the number of documents in the corpus containing at least one occurrence of "*t"* and "*N"* is the number of documents in the corpus.

- Number of numerals [13] in most of the text documents figures are always taking a key part to present important facts, which gives more meaning to the text. Here, the calculation has been done through the ration of numerals to total number of words in the sentence. This feature value has been calculated using the equation 05.

  $$Score\ (Sentence\ Numerals) = N/T \qquad (05)$$

  Where, "N is the number of numerals in the sentence and "T" is the total words in the sentence.

- In feature vector matrix step, a 5×n (n= number of sentences) matrix called feature vector matrix has been generated with the consideration of features considered in the feature extraction stage, therefore every sentence available in the text will have a vector matrix where the feature vector matrix includes the individual matrix regarding to the sentence available in the text. The final output of this stage is the 5×n (n= number of sentences) feature vector matrix.

- In feature enhancement phase 5×n, (n= number of sentences) feature vector matrix will be given as the input to the Restricted Boltzmann Machine (RBM) [9] to improve the weighted values of the feature vector matrix. Recalculation is done on this feature vector matrix. Each feature vector values are multiplied by the learned weights and a bias value and is added to all feature vector values where the learned weights and bias values are learned by the RBM. After this a refined enhanced matrix has been generated. In this step we use RBM because to omit the reduction of dimensionality and we build a complex feature out of simple ones [9].

- At the sentence extraction step, enhanced feature vector values are summed together to produce a score against each sentence. Based on the score the highest score sentence has been chosen as the most relevant sentence, which is a part of subset of sentences which will form the summary.

- After this, in summary generation step sentences available in the existing text will be compared for similarity measure, where the highest similarity value contain sentence has been extracted as another sentence, where it is considered as a part of subset of sentences which will form the summary. Likewise, this process will be executed. Then the extracted sentences will be arranged in the order that they appear in the text. Finally, an extractive summary will be generated for the sports news.

### 4.5.1 Pre-processing

- **Text segmentation**

Text segmentation has been done in the pre-processing step,s where the text available in the news are divided into meaningful units such as sentences and words. Furthermore, regular expressions were used to identify the boundaries between two sentences as well as words.

- **Stop words Removal**

Stop word like “ஒரு” (oru), “என்று”(enru), “மற்றும்” (marrum), etc… are eliminated from the text where these words are used to connect words or sentences which contain less meaning in natural language processing. Using a stop word list which contains 125 stop words in Tamil language has been used for this stop word removal purpose.

- **Named Entity Recognition**

Stanford Named Entity Recognition tagger has been used in this system, where named entities such as person names, organizations, locations, time expressions like entities were identified through this tagger.

### 4.5.2 Feature enhancement

- **Restricted Boltzmann Machine**

In the feature enhancement stage, to improve the weighted values of the feature vector of each sentence Restricted Boltzmann Machine (RBM) has been used which is a generative stochastic artificial neural network that can learn a probability distribution over its set of inputs. The Restricted Boltzmann Machine is constituted by two layers, the first layer is the visible layer and the second layer is the hidden layer. Feature vectors of each sentence will be passed through the hidden layer, where each feature vector values are multiplied by weights learned by the process and a bias value is added to the feature vector values by the Restricted Boltzmann Machine. Finally, enhanced sentence-feature matrix will be obtained.

## 4.6 Features

- Extractive summaries will be created for each sports news.
- Summaries will be created in real-time.
- Using different resources to create extractive summaries regarding to sports news.

## 4.7 Users

- Sports news readers
- Text Summarization Researchers

## 4.8 Summary

In this section, the most important part of this thesis was discussed including the hypothesis, and the inputs, outputs, process and the features of the proposed text summarization system. In the next section, this will be elaborated and discussed in more detail.

# CHAPTER 5

# DESIGN

## 5.1. Introduction

In this chapter, the design of the proposed text summarization system will be discussed in detail. As it has been mentioned, the proposed text summarization system that utilizes Restricted Boltzmann Machine as the fundamental approach.

## 5.2. Architecture of the System

Following figure displaying the high-level architecture of the proposed system:

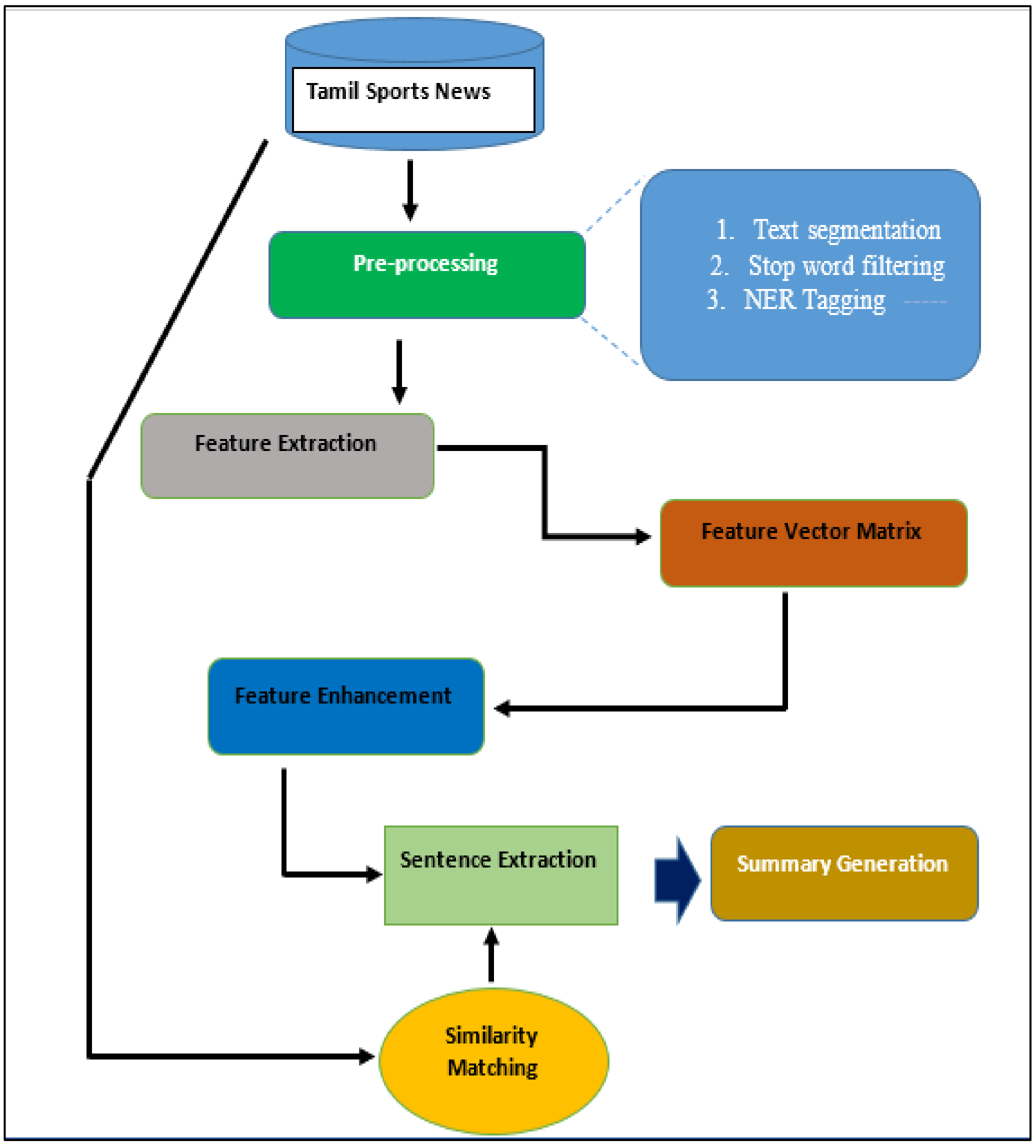


*Figure 5.1 High level architecture of the system*

Figure 5.1 illustrates the high-level architecture of the text summarization, where news data will be given as input to the system. Furthermore, the data will be processed through pre-processing, feature extraction, feature vector matrix, feature enhancement, sentence extraction modules and finally through the summary generation module.

In each module, output from the previous module has been used in the immediate next module. In this process, where output from the pre-processing module has been used in the feature extraction module, likewise the processing has been done. In this system, design each previous module is important to the immediate next module, without those processes a module cannot work independently, each module presented in this design jointly works to achieve the compression of the input data.

Pre-processing module is the initial module in this system where it does the work such as tokenizing the sentence, remove stop words and tag named entities for the next modules' operation. In feature extraction module features such as sentence position, sentence position regarding the paragraph, number of named entities, term frequency and inverse document frequency and number of numerals are considered to generate the feature vector in the feature vector matrix module. The relevant calculation will be done based on the features considered in this module. As previously mentioned in the approach part, the relevant calculation regarding on the features that we considered will be done. Output from this module will help to generate a matrix for each sentence regarding on the features considered in the next module.

In feature vector matrix module, feature matrix for each sentence available in the text will be generated while using the output from the feature extraction module. The output of this module is a vector matrix for the whole text. In the feature enhancement module, the vector matrix generated from the previous module has been trained using Restricted Boltzmann Machine (RBM) for enhancing the vector matrix generated from the feature vector matrix module. The RMB will learn the weights and bias values from the matrix and generate a matrix with heightening values for the feature values that is created in the feature extraction module. This module is the vital part in the text summarization system.

In the sentence, extraction module enhanced feature vector values are summed together to generate a score against each sentence. Based on the score, the sentence with the highest value has been chosen as the most pertinent sentence, which is a part of subset of sentences which will form the summary.

Finally, the most pertinent sentence has been used to do a similarity matching with the other sentence available in the exiting text, where the highest similarity value sentence will be extracted as the another sentence, likewise the rest of the process have been done and it is considered as a part of subset of sentences which will form the summary. At last the extracted sentences in the system have been arranged in the order that they appeared in the original text and this will be presented as the summary of the text.

### 5.2.1. Pre-processing

This module is responsible for tokenizing the sentences, stop word removal and named entity tagging. Input of this module is raw news data and the output for this is tokenized named entity tagged text without stop words.

**Properties**

- Tokenization
- Stop word removal
- Named Entity Recognition

**Functions**

- Text tokenized into sentence.
    - Using regular expressions, text have been tokenized into sentences.
- Sentences tokenized into words.
- Stop word removal.
    - Tokenized words have been compared with set of stop word and words in the stop words list have been removed for that output.

- Tagging named entities
    - Using the tokenized output from the above mentioned sub process (Modified Stanford NER tagger has been used to tag words) words have been tagged as person (Denoted as "P") or location (Denoted as "L"), numeral (Denoted as "N"), date (Denoted as "D") and other words are denoted as "O".

### 5.2.2. Feature extraction

The main purpose of this module is extracting the feature such as sentence position, sentence position regarding the paragraph, number of named entities within the sentence, term frequency and inverse document frequency of a word within the sentence and number of numerals within the sentence. Finally generate a score for the relevant features.

**Properties**

- Sentence position.
- Sentence position regarding to paragraph.
- Number of named entities.
- Term frequency and Inverse document frequency (Tf-Idf).
- Number of numerals.

**Functions**

- From the output of pre-processing module, sentence position have been considered and generate a score for each and every sentence available in the text. Sentence score have been calculated accordingly:
    - If the sentence is first or last sentence of the text sentence score value is 1
    - Score for other sentences available in the text have been calculate using equation 01.

- Generate score for sentence position regarding to paragraph.
    - If the sentence considered is in the first or last paragraph score will be 1.
    - If the sentence considered is not in the first or last paragraph score will be 0.
- Generate score for number of named entities. Score value will be equal to the number of named entities available in the corresponding sentence.
- Generate score for number of named entities. Score value will be equal to the number of named entities available in the corresponding sentence.
- Generate score for term frequency and inverse document frequency. For this purpose, each and every word within the tokenized sentence have been considered and the summation of the value of individual term frequencies have been considered. Calculation have been done using equation 04.
- Generate score for number of numerals for the corresponding sentence.
    - Ration between the number of numerals available and total number of words within the sentence has been used.

### 5.2.3. Feature vector matrix

This module is responsible for generating a matrix called feature matrix (a $5 \times n$, where $n$ is the number of sentences in the text) for a text considered for summarization purpose. Here each sentence in the text have been assigned five score values that have been generated by the pervious module regarding on the features that have been considered. So, output of this module will be a matrix, each sentences' feature score values will be represented in the row in the generated matrix.

**Functions**

- Generate feature vector matrix.

### 5.2.4. Feature enhancement

This module is responsible to generate the enhanced feature matrix. In this module, feature vector matrix generated from the previous module has been given as input to the Restricted Boltzmann Machine (RMB) which is embedded in this module. Using the feature vector matrix, the weights and bias values have been learned by the algorithm in the RBM and relevant weights have been multiplied to the feature scores and the bias value will be added to that score. Thereafter, an enhancement feature score has been established. Likewise, all the feature score values have been enhanced and finally an enhanced feature matrix have been generated.

**Properties**

- Restricted Boltzmann Machine

**Functions**

- Generate enhanced feature matrix.

### 5.2.5. Sentence extraction

In this module, enhanced feature matrix has been used to extract the relevant sentence in the text. Values regarding to each sentence have been summed and a sentence that is having highest value (feature score value) in the matrix has been extracted as the most relevant sentence to form the summary. This particular sentence will be considered as a sub set to form the final summary for the relevant text.

**Functions**

- Extract the relevant sentence form the text.

### 5.2.6. Summary generation

This summary generation module is the final module in the text summarization system and this module will use the output the sentence that has been extracted by the previous module. This particular sentence has been used for similarity matching with the existing sentences in the text. Cosine similarity has been used for this similarity matching purpose. And sentence which is having highest similarity value has been selected as another sentence to form the summary. This similarity matching has been done until considerable number of sentences have been extracted to form the summary. Finally, sentence in the sub set of forming the summary has been ordered in the original order in the particular text and that output will be shown as the summary for the particular text

**Properties**

- Similarity matching

**Functions**

- Do cosine similarity matching with the existing sentences available in the text.
- Extract a sentence that is having the best similarity value form the text.
- Order the extracted sentences as per the order in the original text.

## 5.3. Use Case Diagram

The figure mentioned below shows the use case diagram for the text summarization system.

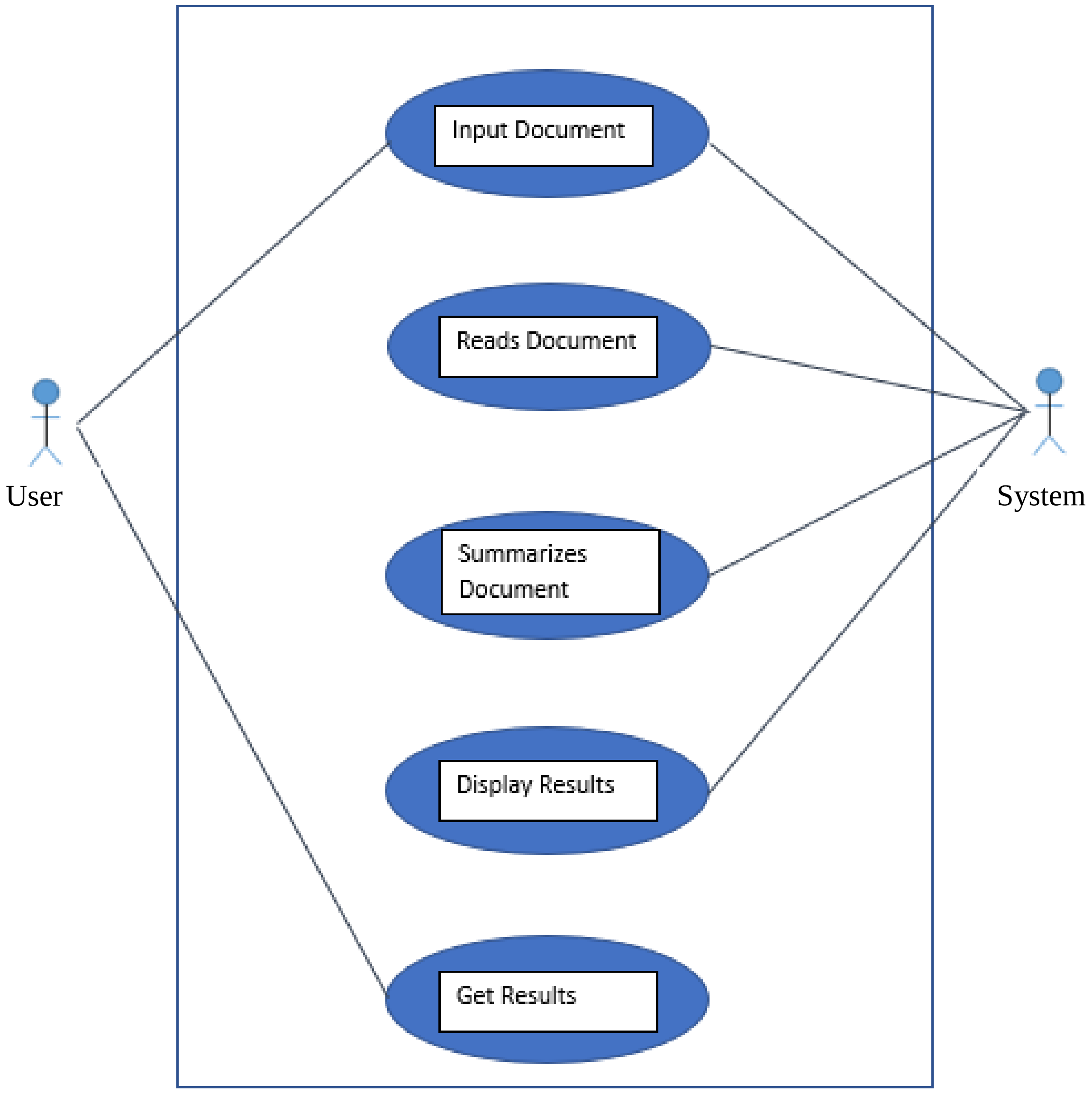


*Figure 5.2 Use case diagram of the system*

## 5.4. Summary

This chapter provides the design of the text summarization system and each and every module such as pre-processing module, feature extraction module, feature vector matrix module, feature enhancement module, sentence extraction module and summary generation module in this system has been presented with detail explanation. Next chapter provides the implementation of the text summarization system.

# CHAPTER 6
# IMPLEMENTATION

## 6.1. Introduction

In the previous chapter, the design of the text summarization system has been discussed and it gives an overview of the high level architecture of the system. In this chapter, the implementation of the system will be discussed elaborately, specially how the modules have been implemented which are discussed in the previous chapter. Firstly, this chapter gives an overview of the software used to develop the system. Furthermore, it discussed about the libraries that have been used to develop the modules in the text summarization system and the implementation codes for relevant modules also discussed.

## 6.2. Python for Text Summarization System Development

Text summarization is a subset of Natural Language Processing (NLP) that handles with extracting summaries from huge number of texts. Nowadays, numerous programming languages, tools are available for the development of text summarization systems. In this research, python programming language has been selected to develop the text summarization system since it’s a commonly used programming language in computational linguistic. Python supports multiple programming paradigms, such as object- oriented, imperative and procedural programming paradigms because of its’ dynamic type and automatic memory handling ability and also its’ text processing ability for Natural Language Processing.

To do the development PyCharm an integrated development environment (IDE), which is cross-platform with Windows, macOS and Linux versions has been used. PyCharm provides code analysis, graphical debugging and an integrated unit tester. In this system, development PyCharm version 2018.3.2 has been used.

## 6.3. Stanford NER Tagger

Named entity recognition (NER) is a subtask of information extraction where it is used to identify or classify named entities stated in unstructured text into well-defined groups such as names of persons, organizations, places, quantities, date, etc…. There are many named entity recognition tools such as GATE, OpenNLP, SpaCy. Among them Stanford NER with modification has been used in the text summarization system in the pre-processing stage to identify named entities.

This Stanford NER [35]is also known as CRF Classifier where it is developed in java programming language. There is also an extension available for various programming languages including python such as pyner, NLTK, scrapy-corenlp.

## 6.4. Pre-processing Module

Firstly, the user gives the text for summarization, where the system gets this as an input by using "readline" library the system reads the text that to be summarized. After that the system starts with the pre-processing module as described in the design chapter tokenization, stop word removal and named entity recognition has been done. For tokenization NLTK library has been utilized, where it splits the text into sentence further into words. Further prefixes such as திரு (Mr), திருமதி (Mrs) ect.. , regular expressions has been used for this tokenizing purpose. Used codes for this purpose has been presented below;

```
def remove_stop_words(sentences):

    tokenized_sentences = []
    for sentence in sentences:
        tokens = []
        split = sentence.split()
        for word in split:
            if word not in stop:
                try:
                    tokens.append(stemmer.stemWord(word).strip())
                except:
                    tokens.append(word)

        tokenized_sentences.append(tokens)
    return tokenized_sentences
```

```
caps = "([ஃ-])"
prefixes = "(திரு|புனித|திருமதி|செல்வி|கலாநிதி)[.]"
```

For stop word removal, a text file which contains set of stop words in Tamil language (Stop words attached in Appendix A) has been used where the stop words are removed from the tokenized output of the process of this module. Below mentioned image shows the code segments are used for stop word removal:

```
import ...

stop = set(stopwords.words('tamil.txt'))
```

For named entity recognition Stanford NER has been modified and used. Before using the NER tagger provides by the Stanford it should be trained to identify named entities in Tamil language. For this purpose, Tamil corpus [36] has been utilized to train the NER tagger. Following are some code segments for named entity recognition used in the system development:

```
java -cp "*" -mx4g edu.stanford.nlp.ie.crf.CRFClassifier -prop train/prop.txt

java -classpath stanford-postagger-3.5.1.jar edu.stanford.nlp.tagger.maxent.MaxentTagger -prop swedish- -
-tagger.props
```

```
trainFile = train/tamil/tamil-corpus.tsv

serializeTo = ner-model-tamil.ser.gz
map = word=0,answer=1

useClassFeature=true
useWord=true
useNGrams=true
noMidNGrams=true
maxNGramLeng=6
usePrev=true
useNext=true
useSequences=true
usePrevSequences=true
maxLeft=1
useTypeSeqs=true
useTypeSeqs2=true
useTypeySequences=true
wordShape=chris2useLC
useDisjunctive=true
```

```
import nltk

from nltk.tag.stanford import StanfordNERTagger

import os

def ner(sentence,cwd):
    jar = cwd+'/stanford-ner-tagger/stanford-ner.jar'
    model = cwd+'/stanford-ner-tagger/ner-model-tamil.ser.gz'
    ner_tagger = StanfordNERTagger(model, jar, encoding='utf8')
    words = nltk.word_tokenize(sentence)
    return(ner_tagger.tag(words))
```

Finally, tokenized sentences without stop words along with named entity tags will be given as an input to the feature extraction module.

## 6.5. Feature Extraction Module

After the pre-processing, feature extraction process have been done based on the features identified as per described in approach chapter. After this process, sentences and their relevant feature scores have been mapped. Here additionally "math" library has been used to develop this module.

Following are some of the code segments relevant to generating score of sentence position in the system development.

```
import math

def senPos(sentences):
    th = 0.2
    minv = th * len(sentences)
    maxv = th * 2 * len(sentences)
    pos = []
    for i in range(len(sentences)):
        if i == 0 or i == len((sentences)):
            pos.append(0)
        else:
            t = math.cos((i - minv) * ((1 / maxv) - minv))
            pos.append(t)
    return pos
```

Below mentioned code segments are relevant to generating score of sentence position regarding to paragraph used in the system development

```
import text_segmentation as tSeg

def paraPos(paragraphs):
   scores = []
   for para in paragraphs:
      sentences = tSeg.split_into_sentences(para)
      if len(sentences) == 1 :
         scores.append(1.0)
      elif len(sentences) == 2 :
         scores.append(1.0)
         scores.append(1.0)
      else :
         scores.append(1.0)
         for x in range(len(sentences)-2) :
            scores.append(0.0)
         scores.append(1.0)
   return scores
```

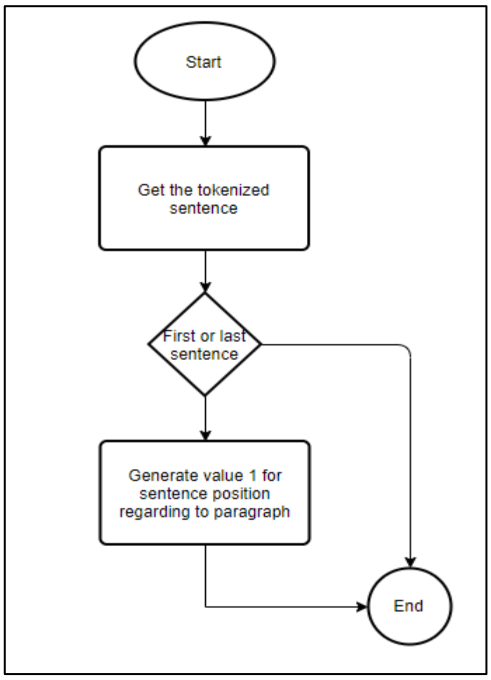

Start
Get the tokenized sentence
First or last sentence
Generate value 1 for sentence position regarding to paragraph
End

*Figure 6.1 Flow chart of decision making on sentence position score*

Code segments regarding to Tf-IDF calculation used in the system development has been presented below;

```
def tFiDF(tokenized_sentences):
   scores = []
   COUNTS = []
   for sentence in tokenized_sentences:
      counts = collections.Counter(sentence)
      isf = []
      score = 0
      for word in counts.keys():
         count_word = 1
         for sen in tokenized_sentences:
            for w in sen:
               if word == w:
                  count_word += 1
         score = score + counts[word] * math.log(count_word - 1)
      scores.append(score / len(sentence))
   return scores
```

Below mentioned code segments are relevant to generating score of number of numerals used in the system development.

```
def is_number(s):
    try:
        float(s)
        return True
    except ValueError:
        return False

def numericToken(tokenized_sentences):
    scores = []
    for sentence in tokenized_sentences :
        score = 0
        for word in sentence :
            if is_number(word) :
                score +=1
        scores.append(score/float(len(sentence)))
    return scores
```

Following are some of the code segments regarding to calculating score for number of named entity identified within the sentence in the system development.

```
import ner_tamil

def namedEntityRecog(sentences, cwd):
    counts = []
    for sentence in sentences:
        count=len(ner_tamil.ner(sentence,cwd))
        counts.append(count)
    return counts
```

## 6.6. Feature Vector Matrix Module

Generating a feature vector matrix is the objective of this particular module. Where it uses the output from the previous module, the relevant scores are mapped against the sentences and a matrix have been produced. The output of this module is attached in the appendix.

## 6.7. Feature Enhancement Module

In this module, the feature vector matrix have been sent through the Restricted Boltzmann Machine for training where each score gained in the feature extraction module have been multiplied by the weights and the bias value will be added to these scores. These weights and bias values are learned by the RBM.

The RBM that has been used in this development has 5 nodes in the visible layer with the learning rate of 0.1. Persistence Contrastive Divergence method has been used to sample the data further Gibbs chains have been used in this sampling process. To develop this module “numpy”, “theano”, “pandas” and “timeit” libraries have been utilized.

Some code segments of this module have been presented below,

```
import numpy

import theano
import theano.tensor as T
import os
import pandas as pd

from theano.tensor.shared_randomstreams import RandomStreams
from logistic_sgd import load_data
```

```
class RBM(object):
    """Restricted Boltzmann Machine (RBM)  """
    def __init__(
        self,
        input=None,
        n_visible=784,
        n_hidden=500,
        W=None,
        hbias=None,
        vbias=None,
        numpy_rng=None,
        theano_rng=None
    ):
```

## 6.8. Sentence Extraction Module

This module is responsible for extracting the sentence which has the highest enhanced feature matrix values. The enhanced feature matrix value will be obtained through the previous module that has been used for this purpose, where each enhanced feature matrix value regarding to each sentence have been summed and then according to summed values, the sentences will be arranged in the descending order. After this, the first sentence in this order will be selected as the most relevant sentence.

## 6.9. Summary Generation Module

In this module, the extracted sentence will be compared with the other sentences available in the text, here sentence similarity calculation has been done. Here cosine similarity measure has been used to measure the similarity between two sentences. After this according to the similarity values rest of the sentence will be arranged in the descending order, where the sentence having the highest value has been extracted as another subset of forming the summary.

Sentence that has been extracted from the sentence extraction module and other sentences extracted from this module have been used to form the summary of the particular text, where sentence extracted in the last two modules have been presented in the order as they are in the order of the original text. Finally, this output has been presented for the user as the summary of that particular text.

Codes for the similarity measure has been presented below,

```
def get_cosine(vec1, vec2):
    intersection = set(vec1.keys()) & set(vec2.keys())
    numerator = sum([vec1[x] * vec2[x] for x in intersection])

    sum1 = sum([vec1[x]**2 for x in vec1.keys()])
    sum2 = sum([vec2[x]**2 for x in vec2.keys()])
    denominator = math.sqrt(sum1) * math.sqrt(sum2)
    if not denominator:
      return 0.0
    else:
      return float(numerator) / denominator


def text_to_vector(text):
    words = text.split()
    return Counter(words)
```

## 6.10. Summary

This chapter provides the implementation of the text summarization system, where each and every module such as pre-processing module, feature extraction module, feature vector matrix module, feature enhancement module, sentence extraction module and summary generation module in this system has been presented with detailed explanation. This text summarization system will help the user to summarize the news articles and provide the gist of the news, as a result the user efficiently manages the resources. Next chapter provides the detailed description of the evaluation of the text summarization system, where it explains the evaluation strategy and experimental setup deeply.

CHAPTER 7

# EVALUATION

## 7.1 Introduction

Previous chapter presents the implementation details of the text summarization system where each module implementation details has been broadly discussed. In this chapter, evaluation of the text summarization system has been presented.

## 7.2 Evaluation Strategy

The text summarization system generated summaries, have been tested against the summaries generated by the human evaluator, where summaries are compared with the human generated summaries. Finally, f-measure has been used to get the performance of the system [1] where the system generated extractive summaries and human generated extractive summaries are considered.

Accuracy of the system will be presented through the precision value where it implies level of correctness of the results, where recall explains level of completeness of the results. Test’s accuracy measure is given through the F-measure, where it is defined as the weighted harmonic mean of the precision and recall of the test. Below mentioned equations show how to calculate the precision, recall and f-measure values.

Precision: $P = A/(A+B)$

Recall: $R= A/(A+C)$

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

Where “A” is, Where "A" is sentences where the human experts think that is relevant for the summary and the same time the system also thinks that those sentences are relevant for summary generation, "B" is sentences where the human experts think that is irrelevant for the summary and the same time the system also thinks that those sentences are relevant, for summary generation, "C" is sentences where the human experts think that is relevant for the summary and the same time the system also thinks that those sentences are irrelevant for summary generation.

Evaluation for a news data:

News:

கிவிடோவாவை வீழ்த்தி சம்பியனானார் ஒசாகா

2019 ஆம் ஆண்டுக்கான அவுஸ்திரேலிய பகிரங்க டென்னிஸ் தொடரின் பெண்களுக்கான ஒற்றையர் பிரிவின் இறுதிப் போட்டியில் ஜப்பான் வீராங்கனை ஒசாக செக்குடியரசின் கிவிடோவாவை வீழ்த்தி சம்பியன் ஆனானர்.
சுமார் 2 மணி 27 நிமிடங்கள் நீடித்த இந்த ஆட்டத்தில் 7–6 (7–2), 5–7, 6–4 என்ற செட் கணக்கில் கிவிடோவாவை சாய்த்து அவுஸ்திரேலிய பகிரங்க டென்னிஸ் சம்பியன் பட்டத்தை ஒசாக முதன் முறையாக கைப்பற்றியுள்ளார்.
இந்த வெற்றியின் மூலம், உலக தரவரிசையில் 4 ஆவது இடத்தில் இருந்த ஒசாகா ‘நம்பர் ஒன்’ இடத்தை பிடித்துள்ளார்.

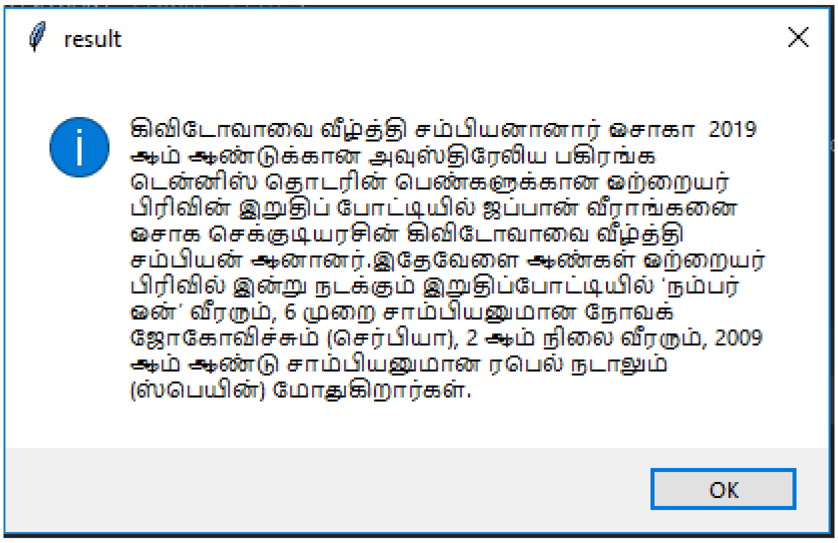
இதேவேளை ஆண்கள் ஒற்றையர் பிரிவில் இன்று நடக்கும் இறுதிப்போட்டியில் ‘நம்பர் ஒன்’ வீரரும், 6 முறை சாம்பியனுமான நோவக் ஜோகோவிச்சும் (செர்பியா), 2 ஆம் நிலை வீரரும், 2009 ஆம் ஆண்டு சாம்பியனுமான ரபெல் நடாலும் (ஸ்பெயின்) மோதுகிறார்கள்.

இவர்கள் இருவரும் இதுவரை 52 முறை நேருக்கு நேர் மோதியிருக்கிறார்கள். இதில் 27 இல் ஜோகோவிச்சும், 25 இல் நடாலும் வெற்றி பெற்றுள்ளமை குறிப்பிடத்தக்கது.

Human generated summary:

கிவிடோவாவை வீழ்த்தி சம்பியனானார் ஒசாகா 2019 ஆம் ஆண்டுக்கான அவுஸ்திரேலிய பகிரங்க டென்னிஸ் தொடரின் பெண்களுக்கான ஒற்றையர் பிரிவின் இறுதிப் போட்டியில் ஜப்பான் வீராங்கனை ஒசாக செக்குடியரசின் கிவிடோவாவை வீழ்த்தி சம்பியன் ஆனானர். இந்த வெற்றியின் மூலம், உலக தரவரிசையில் 4 ஆவது இடத்தில் இருந்த ஒசாகா ‘நம்பர் ஒன்’ இடத்தை பிடித்துள்ளார்.

System Generated summary:


result

கிவிடோவாவை வீழ்த்தி சம்பியனானார் ஒசாகா 2019
ஆம் ஆண்டுக்கான அவுஸ்திரேலிய பகிரங்க
டென்னிஸ் தொடரின் பெண்களுக்கான ஒற்றையர்
பிரிவின் இறுதிப் போட்டியில் ஜப்பான் வீராங்கனை
ஒசாக செக்குடியரசின் கிவிடோவாவை வீழ்த்தி
சம்பியன் ஆனானர்.இதேவேளை ஆண்கள் ஒற்றையர்
பிரிவில் இன்று நடக்கும் இறுதிப்போட்டியில் 'நம்பர்
ஒன்' வீரரும், 6 முறை சாம்பியனுமான நோவக்
ஜோகோவிச்சும் (செர்பியா), 2 ஆம் நிலை வீரரும், 2009
ஆம் ஆண்டு சாம்பியனுமான ரபெல் நடாலும்
(ஸ்பெயின்) மோதுகிறார்கள்.

*Figure 7.1 System generate summary result*

Calculation of F-measure:

A= 2 B= 1 C= 1

Precision: P= A/ (A+B) = 2/ (2 + 1) = 0.67

Recall: R = A/ (A+C) = 2/ (2+1) = 0.67

F-measure= 2PR/(P+R) = 2 * 0.67 *0.67 / (0.67+0.67) = 0.67

Likewise, each and every news data has been evaluated against 3 human generated summaries. Finally, average F-measure value has been considered.

## 7.3 Required Data

To test the performance of this text summarization system news regarding the sports domain gathered from various resources has been used. These new data set has been obtained through such as online sports, sports news articles, sports news reviews, etc…

## 7.4 Experimental Setup

A comparative evaluation has been done for the text summarization systems' result. For evaluation purpose, 30 news data have been used where each summary regarding to each news data has been evaluated against summary generated by 3 Tamil speaking human assessors or judges.

Each news has been distributed among those evaluators and they have to read the news data and they have to select the sentences which will form the summary, likewise the human generated summary responses for each news data set has been gathered.

## 7.5 Results

The below mentioned chart shows the precision, recall and F-measure comparison for a single news document regarding to each human evaluators,

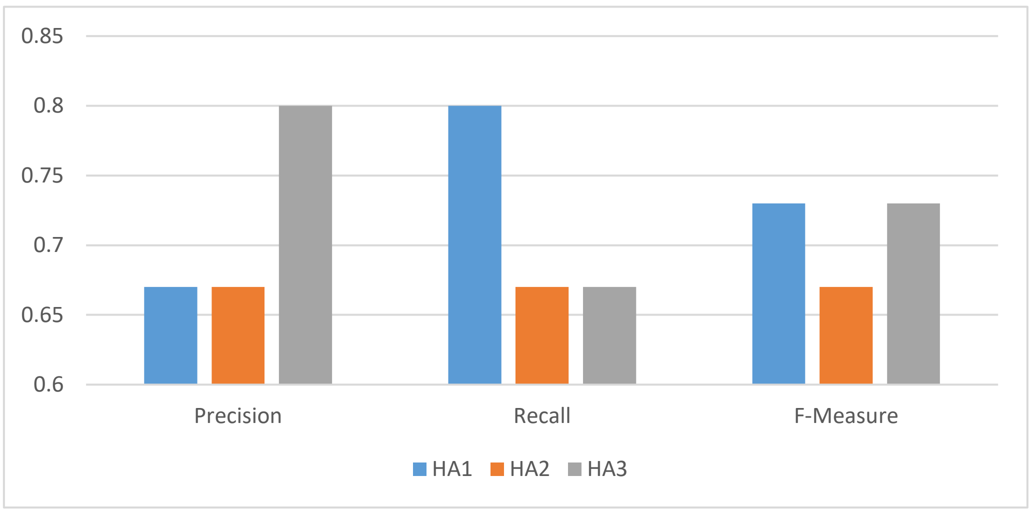


*Figure 7.2 Precision, recall & F-measure for single document*

Term frequency, Sentence score and Sentence paragraph score (according to the equation mentioned) against each sentence in a single text document has been presented below:

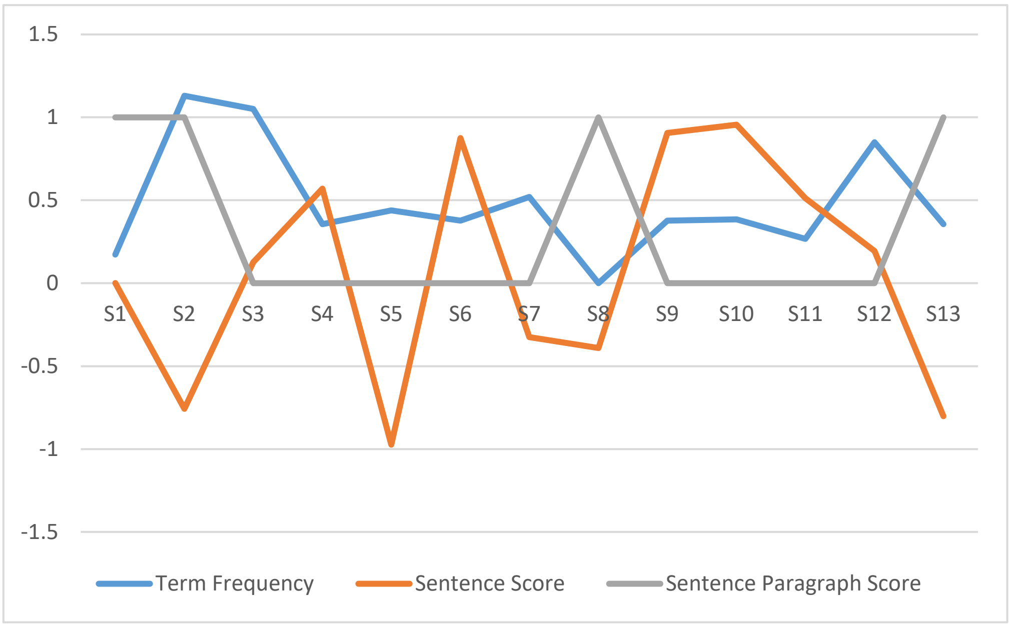


*Figure 7.3 Comparison of Feature scores*

Values of feature vector sums and their enhanced feature vector sums are mentioned against the sentence for one document is shown below:

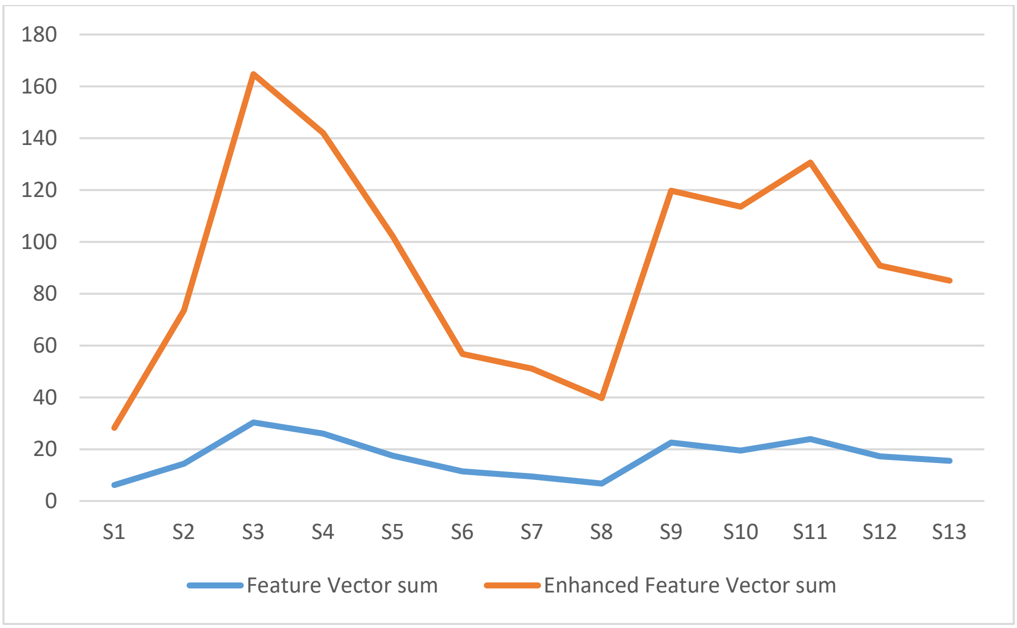
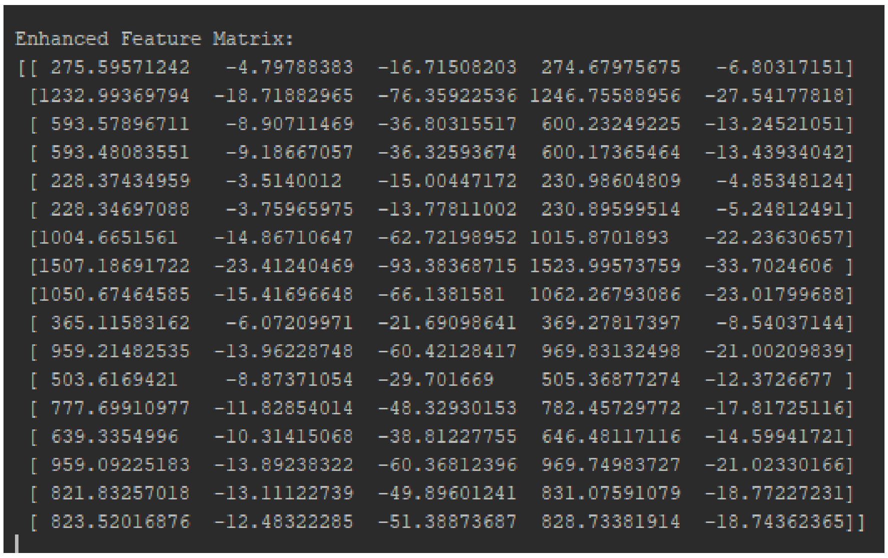


*Figure 7.4 Comparison of Feature vector sum against enhanced feature sum*

Below mentioned figure shows the enhanced feature matrix for a single document:



```
Enhanced Feature Matrix:
[[ 275.59571242   -4.79788383  -16.71508203  274.67975675   -6.80317151]
 [1232.99369794  -18.71882965  -76.35922536 1246.75588956  -27.54177818]
 [ 593.57896711   -8.90711469  -36.80315517  600.23249225  -13.24521051]
 [ 593.48083551   -9.18667057  -36.32593674  600.17365464  -13.43934042]
 [ 228.37434959   -3.5140012   -15.00447172  230.98604809   -4.85348124]
 [ 228.34697088   -3.75965975  -13.77811002  230.89599514   -5.24812491]
 [1004.6651561   -14.86710647  -62.72198952 1015.8701893   -22.23630657]
 [1507.18691722  -23.41240469  -93.38368715 1523.99573759  -33.7024606 ]
 [1050.67464585  -15.41696648  -66.1381581  1062.26793086  -23.01799688]
 [ 365.11583162   -6.07209971  -21.69098641  369.27817397   -8.54037144]
 [ 959.21482535  -13.96228748  -60.42128417  969.83132498  -21.00209839]
 [ 503.6169421    -8.87371054  -29.701669    505.36877274  -12.3726677 ]
 [ 777.69910977  -11.82854014  -48.32930153  782.45729772  -17.81725116]
 [ 639.3354996   -10.31415068  -38.81227755  646.48117116  -14.59941721]
 [ 959.09225183  -13.89238322  -60.36812396  969.74983727  -21.02330166]
 [ 821.83257018  -13.11122739  -49.89601241  831.07591079  -18.77227231]
 [ 823.52016876  -12.48322285  -51.38873687  828.73381914  -18.74362365]]
```

*Figure 7.5 Enhanced Feature Matrix*

## 7.6 Summary

This chapter provides the details of, how the evaluation has been done, experimental setup and the results obtained through the experiment. Furthermore, the interpretation of the results, also the limitation and the future work of this research work has been discussed in the next chapter.

# CHAPTER 8
# CONCLUSION & FURTHER WORK

In the previous chapter, it explains the overall evaluation of this research. The evaluation strategy explains how overall performance of the approach has been demonstrated, experimental setup gives the description of how the approach has been evaluated and results demonstrated the output using the required data, which give the way to come to a conclusion regarding this approach. This concluding chapter explains the final opinion about this approach mentioned in this research and furthermore it gives the details about further research.

Initially, different text summarization approaches have been critically studied regarding to develop an approach for this research, where extractive approaches presented previously by the research community and abstractive summarization approaches presented by the research community are also critically studied. By doing an investigation regarding different technologies used in the literature statistical based approaches also gives good performance independent from the language of the text that will be summarized where semantic of the text mostly covered by natural language-based text summarization approaches. Different machine learning languages are also identified in the literature and it also gives good performance in terms of cohesion. In language dependant text summarization approaches, linguistic measures are having a vital role where natural language processing is essential in this regard.

In the experiment each and every summary generated by the system has been evaluated against the human generated summary and the average Precision, Recall and F-measure of this text summarization system is 76.5%,76.9% and 76.6% respectively, which is higher than the existing approaches for Tamil text summarization approaches.

Because of the development of the web technology and other mass media different documents will contain information regarding the same thing. Summaries generated from these different documents are very useful for the user to save the time for reviewing different documents. This research can be extended to create summaries for different documents as a further work.

**Summary**

This chapter provides the conclusion of the research project, achievements of its objectives and further research work, where summaries can be generated from different documents.

## REFERENCES

[1] R. Nallapati, F. Zhai, and B. Zhou, “SummaRuNNer: A Recurrent Neural Network based Sequence Model for Extractive Summarization of Documents,” *ArXiv161104230 Cs*, Nov. 2016.

[2] Marta Vlainić and Nives Mikelić Preradović, “A Comparative Study of Automatic Text Summarization System Performance,” *Recent Adv. Inf. Sci.*, no. 13, Jun. 2013.

[3] A. M. Rush, S. Chopra, and J. Weston, “A Neural Attention Model for Abstractive Sentence Summarization,” *ArXiv150900685 Cs*, Sep. 2015.

[4] J. Carbonell and J. Goldstein, “The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries,” in *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA, 1998, pp. 335–336.

[5] G. Erkan and D. R. Radev, “LexRank: Graph-based Lexical Centrality As Salience in Text Summarization,” *J Artif Int Res*, vol. 22, no. 1, pp. 457–479, Dec. 2004.

[6] R. McDonald, “A Study of Global Inference Algorithms in Multi-document Summarization,” in *Proceedings of the 29th European Conference on IR Research*, Berlin, Heidelberg, 2007, pp. 557–564.

[7] M. Kågebäck, O. Mogren, N. Tahmasebi, and D. Dubhashi, “Extractive Summarization using Continuous Vector Space Models,” in *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*, Gothenburg, Sweden, 2014, pp. 31–39.

[8] M. Banu, C. Karthika, P. Sudarmani, and T. V. Geetha, “Tamil Document Summarization Using Semantic Graph Method,” in *International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007)*, 2007, vol. 2, pp. 128–134.

[9] A. Taherkhani, G. Cosma, and T. M. McGinnity, “Deep-FS: A feature selection algorithm for Deep Boltzmann Machines,” *Neurocomputing*, vol. 322, pp. 22–37, Dec. 2018.

[10] H. P. Luhn, “The Automatic Creation of Literature Abstracts,” *IBM J. Res. Dev.*, vol. 2, no. 2, pp. 159–165, Apr. 1958.

[11] K. Spärck Jones, “Automatic summarising: The state of the art,” *Inf. Process. Manag.*, vol. 43, no. 6, pp. 1449–1481, Nov. 2007.

[12] J. Goldstein, V. Mittal, J. Carbonell, and M. Kantrowitz, “Multi-document Summarization by Sentence Extraction,” in *Proceedings of the 2000 NAACL-ANLP Workshop on Automatic Summarization*, Stroudsburg, PA, USA, 2000, pp. 40–48.

[13] M. A. Fattah and F. Ren, “GA, MR, FFNN, PNN and GMM based models for automatic text summarization,” *Comput. Speech Lang.*, vol. 23, no. 1, pp. 126–144, Jan. 2009.

[14] Y. Gong and X. Liu, “Generic Text Summarization Using Relevance Measure and Latent Semantic Analysis,” in *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA, 2001, pp. 19–25.

[15] K. Riedhammer, B. Favre, and D. Hakkani-Tür, “Long Story Short - Global Unsupervised Models for Keyphrase Based Meeting Summarization,” *Speech Commun*, vol. 52, no. 10, pp. 801–815, Oct. 2010.

[16] M. Gambhir and V. Gupta, “Recent automatic text summarization techniques: a survey,” *Artif. Intell. Rev.*, vol. 47, no. 1, pp. 1–66, Mar. 2016.

[17] D. R. Radev, W. Zhang, and Z. Zhang, *WebInEssence: A Personalized Web-Based Multi-Document Summarization and Recommendation System*. 2001.

[18] Y. Ko and J. Seo, “An Effective Sentence-extraction Technique Using Contextual Information and Statistical Approaches for Text Summarization,” *Pattern Recogn Lett*, vol. 29, no. 9, pp. 1366–1371, Jul. 2008.

[19] D. Parveen and M. Strube, “Integrating Importance, Non-Redundancy and Coherence in Graph-Based Extractive Summarization,” in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[20] M. Patil, M. S. Bewoor, and S. H. Patil, “A Hybrid Approach for Extractive Document Summarization Using Machine Learning and Clustering Technique,” 2014.

[21] C. Yao, J. Shen, and G. Chen, “Automatic Document Summarization via Deep Neural Networks,” *2015 8th Int. Symp. Comput. Intell. Des. ISCID,* vol. 1, pp. 291–296, 2015.

[22] R. M. Alguliev, R. M. Aliguliyev, and N. R. Isazade, “Multiple documents summarization based on evolutionary optimization algorithm,” *Expert Syst. Appl.,* vol. 40, no. 5, pp. 1675–1689, Apr. 2013.

[23] M. Mendoza, S. Bonilla, C. Noguera, C. Cobos, and E. León, “Extractive single-document summarization based on genetic operators and guided local search,” *Expert Syst. Appl.,* vol. 41, no. 9, pp. 4158–4169, Jul. 2014.

[24] J. Yao, X. Wan, and J. Xiao, “Phrase-based Compressive Cross-Language Summarization,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing,* Lisbon, Portugal, 2015, pp. 118–127.

[25] K. Ganesan, C. Zhai, and J. Han, “Opinosis: a graphbased approach to abstractive summarization of highly redudant opinions,” in *In COLING,* 2010.

[26] S. W. K. Chan, “Beyond keyword and cue-phrase matching: A sentence-based abstraction technique for information extraction,” *Decis. Support Syst.,* vol. 42, no. 2, pp. 759–777, Nov. 2006.

[27] S. Banerjee, P. Mitra, and K. Sugiyama, “Multi-document abstractive summarization using ILP based multi-sentence compression,” *ArXiv160907034 Cs,* Sep. 2016.

[28] M. Welling, M. Rosen-Zvi, and G. Hinton, “Exponential Family Harmoniums with an Application to Information Retrieval,” in *Proceedings of the 17th International Conference on Neural Information Processing Systems,* Cambridge, MA, USA, 2004, pp. 1481–1488.

[29] Geoffrey Hinton, “To Recognize Shapes, First Learn to Generate Images,” *Comput. Neurosci. Theor. Insights Brain Funct. Elsevier,* Oct. 2006.

[30] R. Salakhutdinov, A. Mnih, and G. Hinton, “Restricted Boltzmann Machines for Collaborative Filtering,” in *Proceedings of the 24th International Conference on Machine Learning,* New York, NY, USA, 2007, pp. 791–798.

[31] G. W. Taylor, G. E. Hinton, and S. Roweis, “Modeling Human Motion Using Binary Latent Variables,” in *Proceedings of the 19th International Conference*

*on Neural Information Processing Systems*, Cambridge, MA, USA, 2006, pp. 1345–1352.

[32] I. Sutskever and G. Hinton, "Learning Multilevel Distributed Representations for High-Dimensional Sequences," in *Artificial Intelligence and Statistics*, 2007, pp. 548–555.

[33] P. V. Gehler, A. D. Holub, and M. Welling, "The Rate Adapting Poisson Model for Information Retrieval and Object Recognition," in *Proceedings of the 23rd International Conference on Machine Learning*, New York, NY, USA, 2006, pp. 337–344.

[34] H. Larochelle and Y. Bengio, "Classification using discriminative restricted Boltzmann machines," in *Proceedings of the 25th international conference on Machine learning - ICML '08*, Helsinki, Finland, 2008, pp. 536–543.

[35] Jenny Rose Finkel, Trond Grenager, and Christopher Manning, "Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling," *Proc. 43nd Annu. Meet. Assoc. Comput. Linguist. ACL 2005*, pp. 363–370, 2005.

[36] Sobha Lalitha Devi, Sindhuja Gopalan, L Gracy, N Padmapriya, A Gnanapriya and N H Parimala. (2016) " AUKBC Tamil Part-of Speech Corpus(AUKBC-TamilPOSCorpus2016v1)".Web Download. Computational Linguistics Research Group, AU-KBC Research Centre, Chennai, India, May 2016.

## APPENDIX A

## STOP WORD LIST

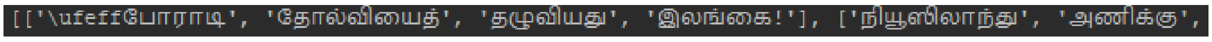
| | | | | |
|---|---|---|---|---|
| ஒரு | இதன் | இருந்த | அன்று | இடத்தில் |
| என்று | அது | மிகவும் | ஒரே | அதில் |
| மற்றும் | அவன் | இங்கு | மிக | நாம் |
| இந்த | தான் | மீது | அங்கு | அதற்கு |
| இது | பலரும் | ஓர் | பல்வேறு | எனவே |
| என்ற | என்னும் | இவை | விட்டு | பிற |
| கொண்டு | மேலும் | இந்தக் | பெரும் | சிறு |
| என்பது | பின்னர் | பற்றி | அதை | மற்ற |
| பல | கொண்ட | வரும் | பற்றிய | விட |
| ஆகும் | இருக்கும் | வேறு | உன் | எந்த |
| அல்லது | தனது | இரு | அதிக | எனவும் |
| அவர் | உள்ளது | இதில் | அந்தக் | எனப்படும் |
| நான் | போது | போல் | பேர் | எனினும் |
| உள்ள | என்றும் | இப்போது | இதனால் | அடுத்த |
| அந்த | அதன் | அவரது | அவை | இதனை |
| இவர் | தன் | மட்டும் | அதே | இதை |
| என | பிறகு | இந்தப் | ஏன் | கொள்ள |
| முதல் | அவர்கள் | எனும் | முறை | இந்தத் |
| என்ன | வரை | மேல் | யார் | இதற்கு |
| இருந்து | அவள் | பின் | என்பதை | அதனால் |
| சில | நீ | சேர்ந்த | எல்லாம் | தவிர |
| என் | ஆகிய | ஆகியோர் | மட்டுமே | போல |
| போன்ற | இருந்தது | எனக்கு | இங்கே | வரையில் |
| வேண்டும் | உள்ளன | இன்னும் | அங்கே | சற்று |

வந்து வந்த அந்தப் இடம் எனக்

# APPENDIX B

# EXPERIMENT OUTPUTS

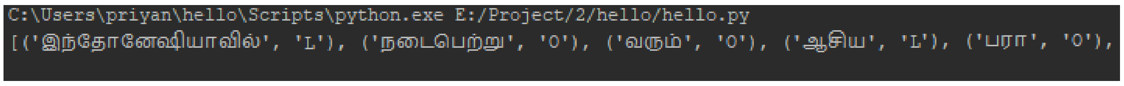
## Figure of Tokenization



```
[['\ufeffபோராடி', 'தோல்வியைத்', 'தழுவியது', 'இலங்கை!'], ['நியூஸிலாந்து', 'அணிக்கு',
```

*Figure A.1 Tokenized output form pre-processing*

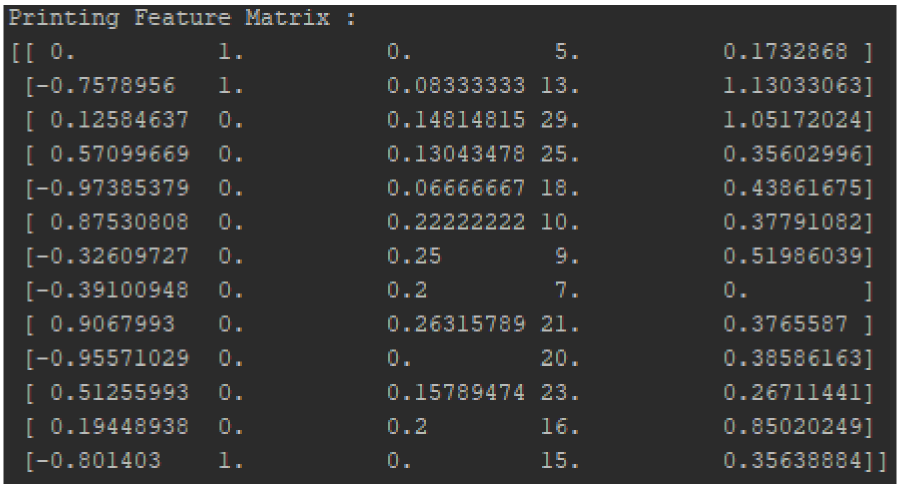
## Figure of Named Entity Tagging



```
C:\Users\priyan\hello\Scripts\python.exe E:/Project/2/hello/hello.py
[('இந்தோனேஷியாவில்', 'L'), ('நடைபெற்று', 'O'), ('வரும்', 'O'), ('ஆசிய', 'L'), ('பரா', 'O'),
```

*Figure A.2 NER tagger output*

## Feature Matrix Output



```
Printing Feature Matrix :
[[ 0.          1.          0.          5.          0.1732868 ]
 [-0.7578956   1.          0.08333333 13.          1.13033063]
 [ 0.12584637  0.          0.14814815 29.          1.05172024]
 [ 0.57099669  0.          0.13043478 25.          0.35602996]
 [-0.97385379  0.          0.06666667 18.          0.43861675]
 [ 0.87530808  0.          0.22222222 10.          0.37791082]
 [-0.32609727  0.          0.25        9.          0.51986039]
 [-0.39100948  0.          0.2         7.          0.        ]
 [ 0.9067993   0.          0.26315789 21.          0.3765587 ]
 [-0.95571029  0.          0.         20.          0.38586163]
 [ 0.51255993  0.          0.15789474 23.          0.26711441]
 [ 0.19448938  0.          0.2        16.          0.85020249]
 [-0.801403    1.          0.         15.          0.35638884]]
```

*Figure A.3 Feature matrix output*

## APPENDIX C

## SOME USEFUL LINKS

I. http://deeplearning.net/tutorial/rbm.html#rbm

II. https://github.com/AshokR/TamilNLP/wiki/Stopwords

III. https://nlp.stanford.edu/software/CRF-NER.shtml

# APPENDIX D

## INTERFACE OF THE SYSTEM

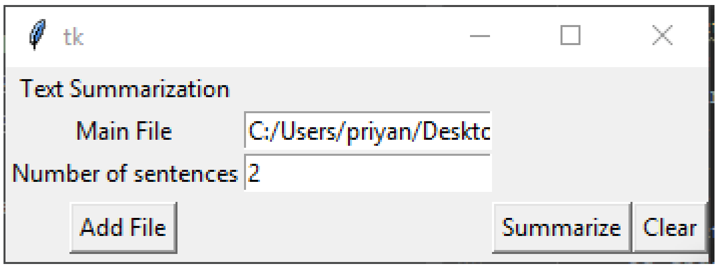
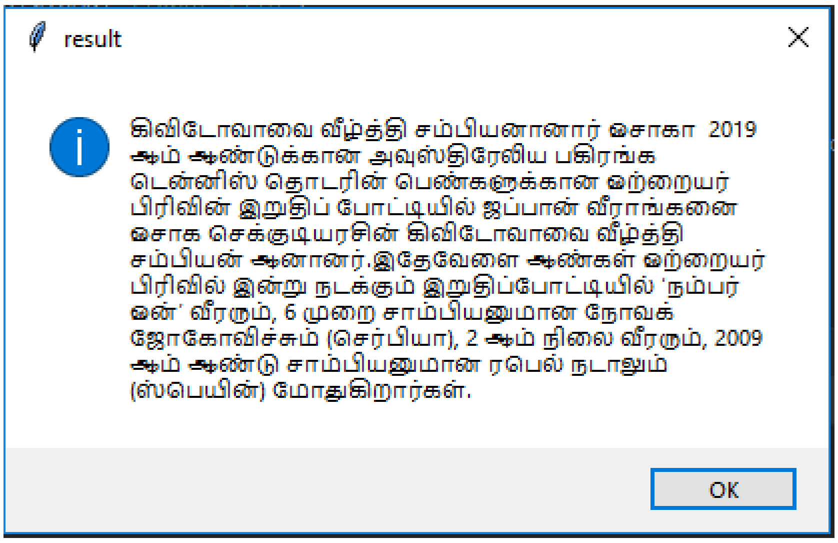


*Figure A.4 System Interface*

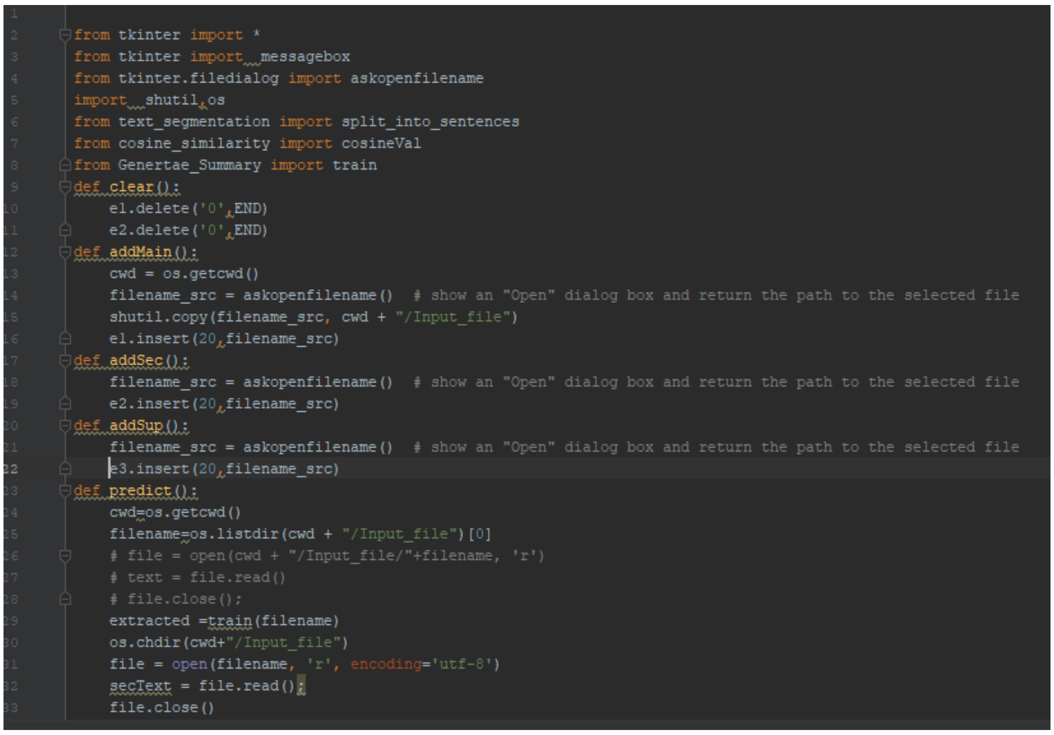


கிவிடோவாவை வீழ்த்தி சம்பியனானார் ஒசாகா 2019 ஆம் ஆண்டுக்கான அவுஸ்திரேலிய பகிரங்க டென்னிஸ் தொடரின் பெண்களுக்கான ஒற்றையர் பிரிவின் இறுதிப் போட்டியில் ஜப்பான் வீராங்கனை ஒசாக செக்குடியரசின் கிவிடோவாவை வீழ்த்தி சம்பியன் ஆனானர்.இதேவேளை ஆண்கள் ஒற்றையர் பிரிவில் இன்று நடக்கும் இறுதிப்போட்டியில் 'நம்பர் ஒன்' வீரரும், 6 முறை சாம்பியனுமான நோவக் ஜோகோவிச்சும் (செர்பியா), 2 ஆம் நிலை வீரரும், 2009 ஆம் ஆண்டு சாம்பியனுமான ரபெல் நடாலும் (ஸ்பெயின்) மோதுகிறார்கள்.

*Figure A.5 System Output Interface*

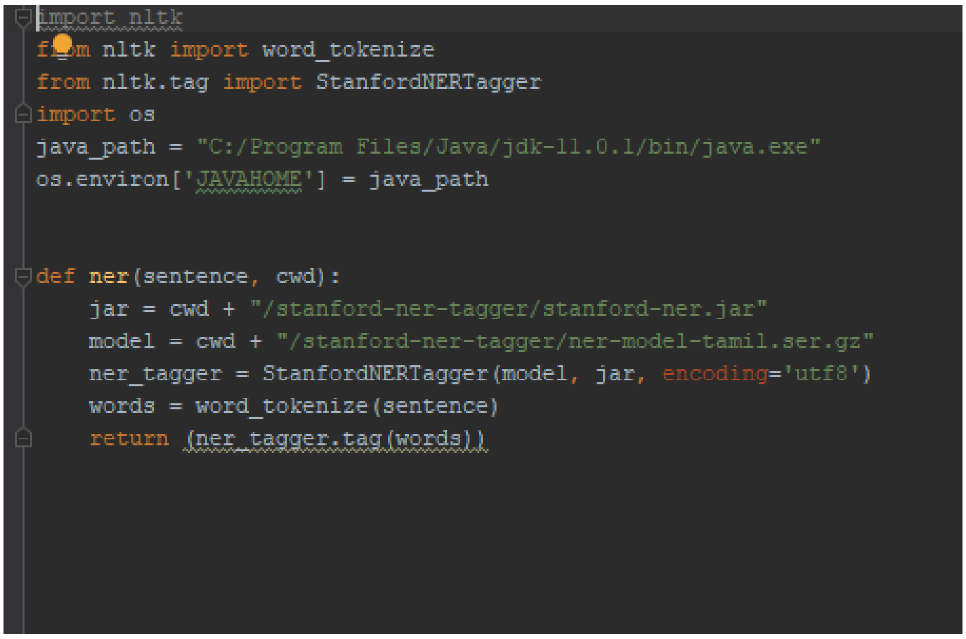
# APPENDIX E

## CODE SECTIONS OF THE SYSTEM DEVELOPMENT



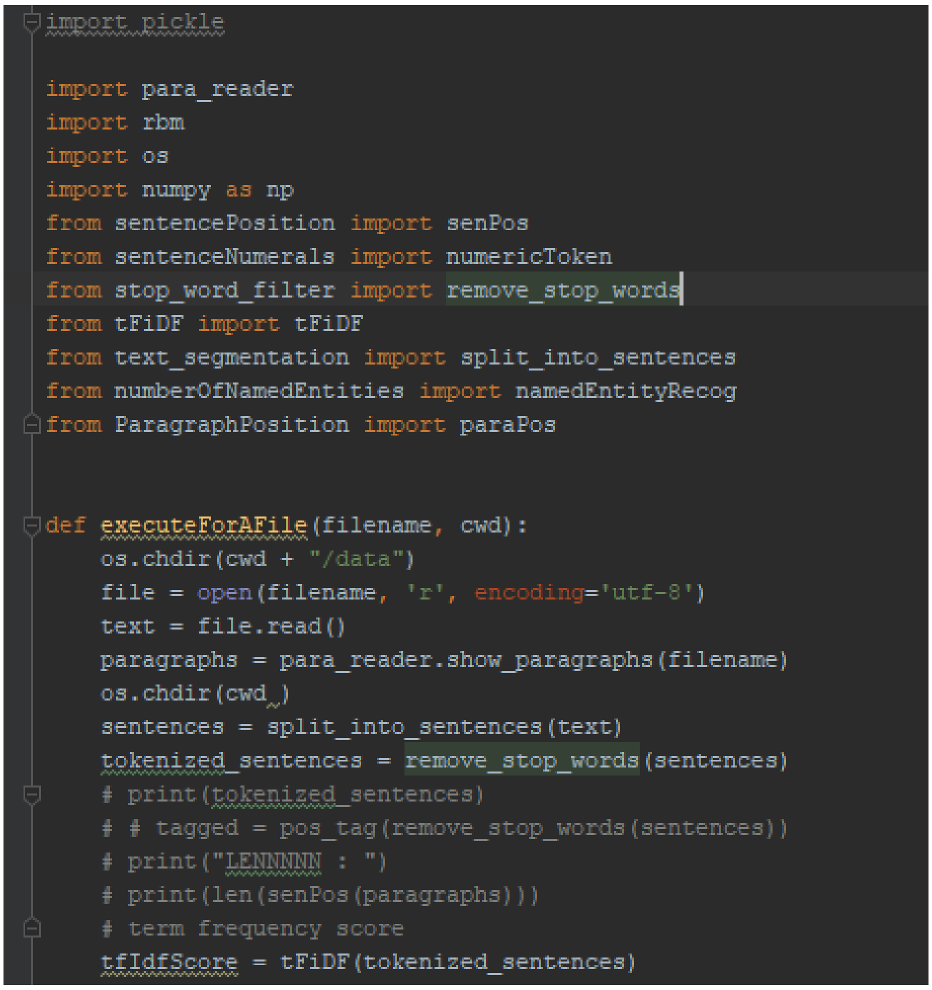
```python

from tkinter import *
from tkinter import  messagebox
from tkinter.filedialog import askopenfilename
import  shutil,os
from text_segmentation import split_into_sentences
from cosine_similarity import cosineVal
from Genertae_Summary import train
def clear():
    e1.delete('0',END)
    e2.delete('0',END)
def addMain():
    cwd = os.getcwd()
    filename_src = askopenfilename()  # show an "Open" dialog box and return the path to the selected file
    shutil.copy(filename_src, cwd + "/Input_file")
    e1.insert(20,filename_src)
def addSec():
    filename_src = askopenfilename()  # show an "Open" dialog box and return the path to the selected file
    e2.insert(20,filename_src)
def addSup():
    filename_src = askopenfilename()  # show an "Open" dialog box and return the path to the selected file
    e3.insert(20,filename_src)
def predict():
    cwd=os.getcwd()
    filename=os.listdir(cwd + "/Input_file")[0]
    # file = open(cwd + "/Input_file/"+filename, 'r')
    # text = file.read()
    # file.close();
    extracted =train(filename)
    os.chdir(cwd+"/Input_file")
    file = open(filename, 'r', encoding='utf-8')
    secText = file.read();
    file.close()
```

*Figure A.6 Code section of Interface Development*

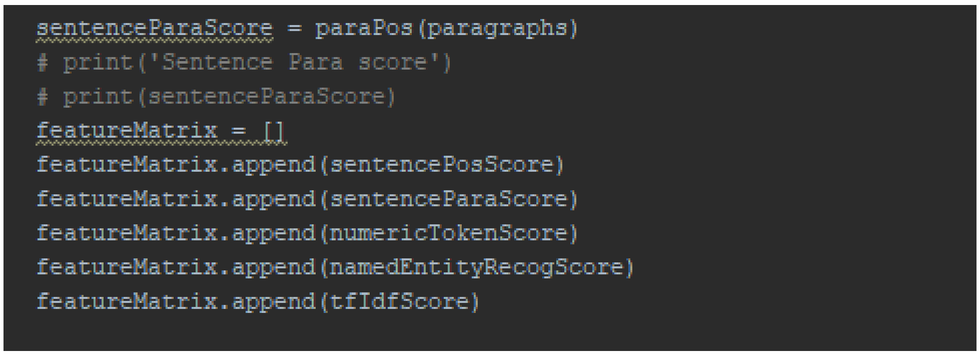


```python
import nltk
from nltk import word_tokenize
from nltk.tag import StanfordNERTagger
import os
java_path = "C:/Program Files/Java/jdk-11.0.1/bin/java.exe"
os.environ['JAVAHOME'] = java_path


def ner(sentence, cwd):
    jar = cwd + "/stanford-ner-tagger/stanford-ner.jar"
    model = cwd + "/stanford-ner-tagger/ner-model-tamil.ser.gz"
    ner_tagger = StanfordNERTagger(model, jar, encoding='utf8')
    words = word_tokenize(sentence)
    return  (ner_tagger.tag(words))
```

*Figure A.7 Code section of Named entity recognition*

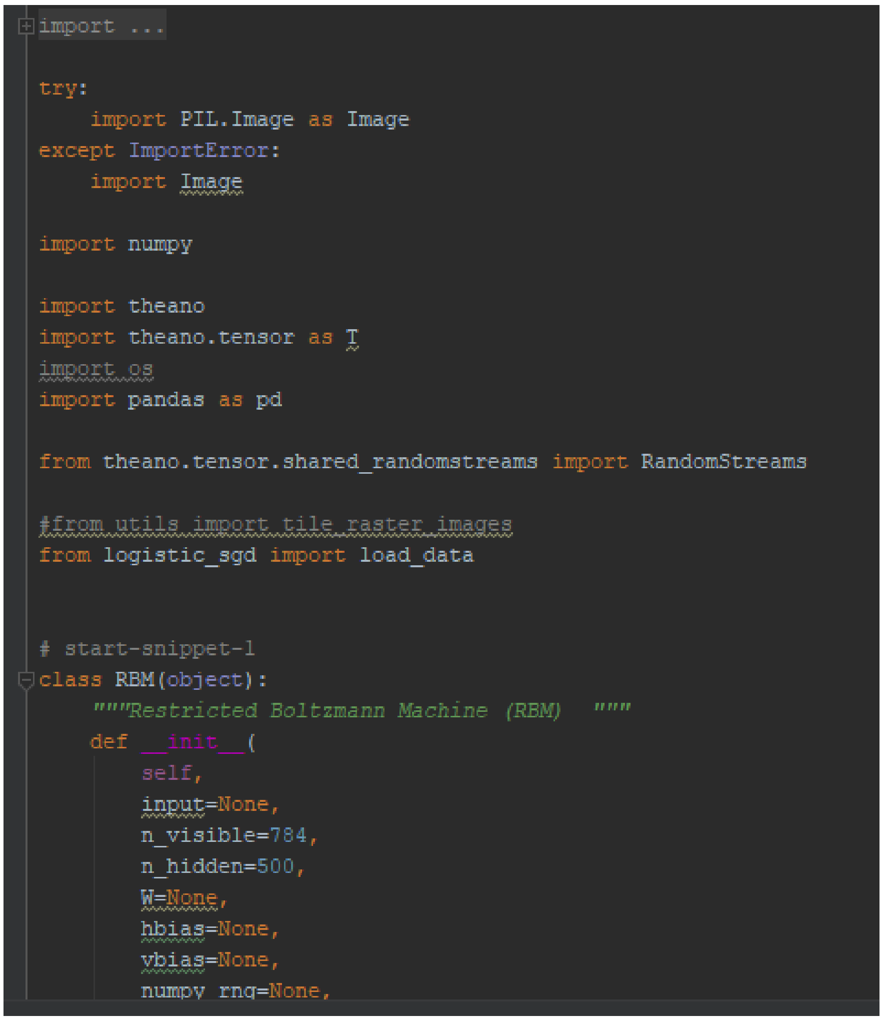


```
import pickle

import para_reader
import rbm
import os
import numpy as np
from sentencePosition import senPos
from sentenceNumerals import numericToken
from stop_word_filter import remove_stop_words
from tFiDF import tFiDF
from text_segmentation import split_into_sentences
from numberOfNamedEntities import namedEntityRecog
from ParagraphPosition import paraPos


def executeForAFile(filename, cwd):
    os.chdir(cwd + "/data")
    file = open(filename, 'r', encoding='utf-8')
    text = file.read()
    paragraphs = para_reader.show_paragraphs(filename)
    os.chdir(cwd )
    sentences = split_into_sentences(text)
    tokenized_sentences = remove_stop_words(sentences)
    # print(tokenized_sentences)
    # # tagged = pos_tag(remove_stop_words(sentences))
    # print("LENNNNN : ")
    # print(len(senPos(paragraphs)))
    # term frequency score
    tfIdfScore = tFiDF(tokenized_sentences)
```



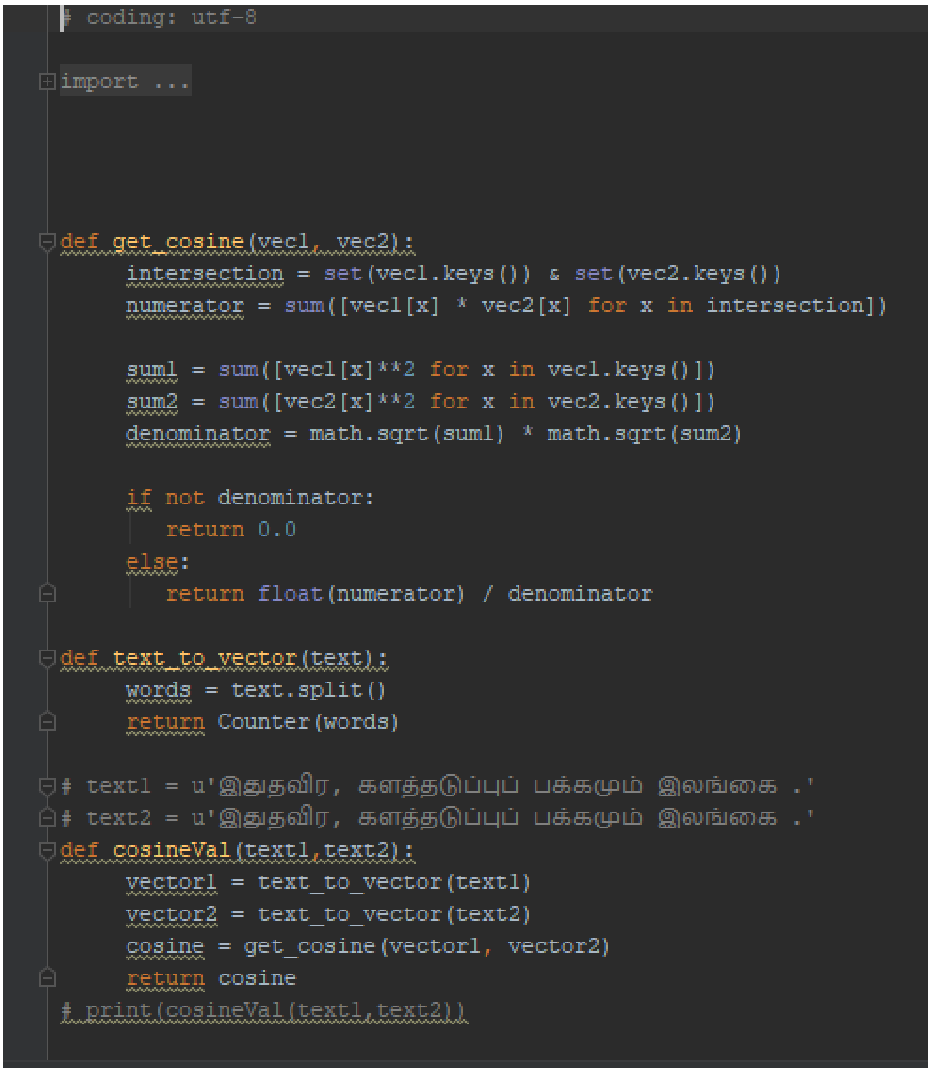
```
    sentenceParaScore = paraPos(paragraphs)
    # print('Sentence Para score')
    # print(sentenceParaScore)
    featureMatrix = []
    featureMatrix.append(sentencePosScore)
    featureMatrix.append(sentenceParaScore)
    featureMatrix.append(numericTokenScore)
    featureMatrix.append(namedEntityRecogScore)
    featureMatrix.append(tfIdfScore)
```

*Figure A.8 Code section of summary generation*



```
import ...

try:
    import PIL.Image as Image
except ImportError:
    import Image

import numpy

import theano
import theano.tensor as T
import os
import pandas as pd

from theano.tensor.shared_randomstreams import RandomStreams

#from utils import tile_raster_images
from logistic_sgd import load_data


# start-snippet-1
class RBM(object):
    """Restricted Boltzmann Machine (RBM)  """
    def __init__(
        self,
        input=None,
        n_visible=784,
        n_hidden=500,
        W=None,
        hbias=None,
        vbias=None,
        numpy_rng=None,
```

*Figure A.9 Code section of RBM*

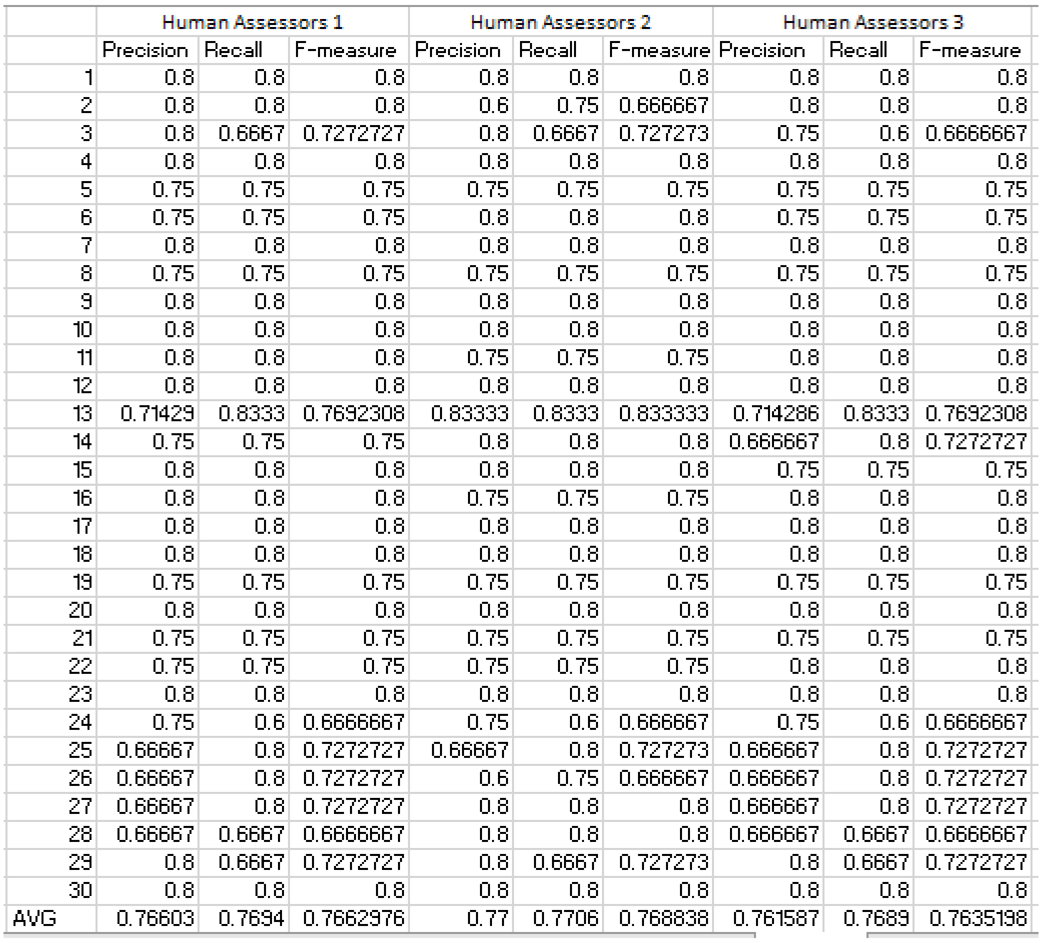


```python
# coding: utf-8

import ...



def get_cosine(vec1, vec2):
        intersection = set(vec1.keys()) & set(vec2.keys())
        numerator = sum([vec1[x] * vec2[x] for x in intersection])

        sum1 = sum([vec1[x]**2 for x in vec1.keys()])
        sum2 = sum([vec2[x]**2 for x in vec2.keys()])
        denominator = math.sqrt(sum1) * math.sqrt(sum2)

        if not denominator:
            return 0.0
        else:
            return float(numerator) / denominator

def text_to_vector(text):
        words = text.split()
        return Counter(words)

# text1 = u'இதுதவிர, களத்தடுப்புப் பக்கமும் இலங்கை .'
# text2 = u'இதுதவிர, களத்தடுப்புப் பக்கமும் இலங்கை .'
def cosineVal(text1,text2):
        vector1 = text_to_vector(text1)
        vector2 = text_to_vector(text2)
        cosine = get_cosine(vector1, vector2)
        return cosine
# print(cosineVal(text1,text2))
```

*Figure A.10 Code section of similarity matching*

# APPENDIX F

## DETAILED F-MEASURE, PRECISION AND RECALL



| | Human Assessors 1 | | | Human Assessors 2 | | | Human Assessors 3 | | |
|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | F-measure | Precision | Recall | F-measure | Precision | Recall | F-measure |
| 1 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 |
| 2 | 0.8 | 0.8 | 0.8 | 0.6 | 0.75 | 0.666667 | 0.8 | 0.8 | 0.8 |
| 3 | 0.8 | 0.6667 | 0.7272727 | 0.8 | 0.6667 | 0.727273 | 0.75 | 0.6 | 0.6666667 |
| 4 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 |
| 5 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 |
| 6 | 0.75 | 0.75 | 0.75 | 0.8 | 0.8 | 0.8 | 0.75 | 0.75 | 0.75 |
| 7 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 |
| 8 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 |
| 9 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 |
| 10 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 |
| 11 | 0.8 | 0.8 | 0.8 | 0.75 | 0.75 | 0.75 | 0.8 | 0.8 | 0.8 |
| 12 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 |
| 13 | 0.71429 | 0.8333 | 0.7692308 | 0.83333 | 0.8333 | 0.833333 | 0.714286 | 0.8333 | 0.7692308 |
| 14 | 0.75 | 0.75 | 0.75 | 0.8 | 0.8 | 0.8 | 0.666667 | 0.8 | 0.7272727 |
| 15 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.75 | 0.75 | 0.75 |
| 16 | 0.8 | 0.8 | 0.8 | 0.75 | 0.75 | 0.75 | 0.8 | 0.8 | 0.8 |
| 17 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 |
| 18 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 |
| 19 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 |
| 20 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 |
| 21 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 |
| 22 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.8 | 0.8 | 0.8 |
| 23 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 |
| 24 | 0.75 | 0.6 | 0.6666667 | 0.75 | 0.6 | 0.666667 | 0.75 | 0.6 | 0.6666667 |
| 25 | 0.66667 | 0.8 | 0.7272727 | 0.66667 | 0.8 | 0.727273 | 0.666667 | 0.8 | 0.7272727 |
| 26 | 0.66667 | 0.8 | 0.7272727 | 0.6 | 0.75 | 0.666667 | 0.666667 | 0.8 | 0.7272727 |
| 27 | 0.66667 | 0.8 | 0.7272727 | 0.8 | 0.8 | 0.8 | 0.666667 | 0.8 | 0.7272727 |
| 28 | 0.66667 | 0.6667 | 0.6666667 | 0.8 | 0.8 | 0.8 | 0.666667 | 0.6667 | 0.6666667 |
| 29 | 0.8 | 0.6667 | 0.7272727 | 0.8 | 0.6667 | 0.727273 | 0.8 | 0.6667 | 0.7272727 |
| 30 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 |
| AVG | 0.76603 | 0.7694 | 0.7662976 | 0.77 | 0.7706 | 0.768838 | 0.761587 | 0.7689 | 0.7635198 |

*Figure A.11 Precision, Recall and F-measure values*