Abstract

POS and text summarization are the core concepts of every Natural Language processing Techniques, Text Summarization reduces the size of the content and give the brief summary of the whole content. we have implemented Kannada text summarizer using NLP. this model will allow the user to upload the files which is in Kannada. Once the file is uploaded the text rank algorithm is processed in a few minutes get back to the user and prompt the summary size from the whole document. the idea behind the algorithm is ranking the sentences using the graph method, vector, cosine similarity. Higher the cosine similarity is the top sentence. The page rank algorithm will take the numerical values which obtain by the cosine similarity and arrange them in accordingly. The end of the process, the model will generate top ranking sentences in the user interface to perform print or generate the pdf. The process of generating summary involves various sub methods or sub stages to generate the final summary. All the steps from data cleaning to generate summary have done with specification and handled all the exception.

TABLE OF CONTENTS

CHsAPTER 1

INTRODUCTION

1.1 General Introduction

In modern times efficiency is the core to all work. Any form of information is reducing to only what's necessary to understand the topic. For this one of the tools implemented is the text summarization software. This application summarizes a kannada document to a more concise form all the while maintaining the same meaning without losing any crucial information. But text summarization is mainly present for only a minute range of dialects such as English. there is little effort put into this field for the Kannada language. In this project we try to implement NLP text summarization for the Kannada language. For text summarization there are fundamentally two methods of implementation are present they are- extractive approach and abstractive approach. The former extrapolates words and phrases from the original text to create a summary. In this paper we implement extractive method of text summarization. In extractive method the words and word phrases are extracted from the Kannada document the documents are first cleaned and undergo pre-processing and word embedding before being ready to be summarized using the text rank algorithm to initiate the extractive method of summarization. The authors implement text summarization for Kannada language in this method and verify each step by testing it in a controlled environment.

1.2 Problem Statement

In modern times with the massive amounts of data available online/offline it becomes necessary to compress the data to efficiently process it. For this one of the tools implemented is the text summarization software but this application is not widely available for some regional dialects. People who require summarizations for documents or articles in Kannada language need's to perform it manually as there is no advancement for Kannada language in this field. This in turn adversely effects the popularity and spread of Kannada language.

1.3 Objective Of The Project

The main objective of the project is to construct a text summarization software that specifically summarizes documents and articles in Kannada language while maintain its original. It achieves this by

- The proposed system first takes the given the given Kannada document cleans and formats it
- Then the Kannada word embedding's found for the cleaned document.
- The processed document undergoes cosine similarity to find similarity matrix is found
- Finally, the summarized form of the input document is obtained using the text rank algorithm.

By this process, we aim to obtain a summary any Kannada document.

1.4 Project Deliverables

The end user can use the software by taking document in Kannada language and passing it through the system. The software first pre-processes the document by cleaning the document and performs word embedding on it then the processed data is summarized using the text rank algorithm to obtain a summary of the input document. It will give the output in required format as user want.

1.5 Current Scope

Presently text summarization is mostly available for the English language. There is for implementation in this field for different regional languages such as Kannada. This software can effectively take documents which are in Kannada language and produce a summarized output, which contains the same meaning as the input but in a much more conscience manner and reduces end user important time and patience instead of reading entire document.

1.6 Future Scope

There is much scope for our project for enhancement, such as increasing the regional dialects the software can summarize by adding more languages such as Tamil, Telegu, Spanish etc. We can also include software which can perform translation of languages and perform POS (parts of speech) tagging

CHAPTER 2

PROJECT ORGANIZATION

2.1 Software Process Model

For this project we have implemented the agile software process model. Agile method is a method that allows organizations to think up and quickly respond to changing demand, while minimising risk, It allows the designers to quickly adapt to any change in situations and also to make necessary changes to the system. Agile software development method is usually operated in quick and minute steps. This let's us gain in more frequent increasing releases with each one being the foundation for functionality of the one before it. By the way of analysing the end result the software quality is maintained.

Figure 2.1

2.2 Roles And Responsibility

RAVI KUMAR – PROJECT MANAGER , LEAD DEVELOPER – In charge of team and responsible for developing the software

PRATHEEK S N – TEST ANALYST, DEVOLPER – Responsible for analysing and conducting tests also part of the development team.

RAJATH – RESOURCE MANAGER , DEVELOPER – Responsible for managing resources of project such as software hardware, also part of development team

KRUTHIKA – PROJECT ANALYST,DEVELOPER – Responsible for analysing project progress as well as performance, also part of development team.

CHAPTER 3

LITERATURE SURVEY

3.1 Introduction

Part of the Speech tagging is the process of marking up a word in a corpus as corresponding to the correct parts of speech. Text summarization in Kannada is the process of reducing the set of data to create a subset of that data to represents the most important and meaning full information with original content. This survey mainly focuses on summarizing the methodologies and the techniques used by different researchers.

3.2 Related Works with the citation of the References

[1] In this paper the author introduces tools which can represents text in the form of graphs and in turn which helps in manually creating summaries. Text summarization with the help of graphs gives a more comprehensive understanding of the original text to the viewer, it helps the user in getting a better understanding of the text before the actual summarization. Text mining tools are being implemented with respect to the various field in education as an application for processing data, which have shown positive results. An experiment pertaining to text mining tools where conducted where students were given the tool and asked to create summaries of texts with its help their interaction with the computers were recoded. The experiment showed that the students could write summaries with greater ease as the text mining tools helped them by identifying the main theme of the original text. The graphs produced by the students were determined to be close to what was considered to be close to what was considered important information. The observations of the students using the tools will provide crucial information on the future use of text mining tools in the various fields. The author has concluded that the use of graphs and other such tools has proven to be an effective method in creating summaries by allowing the users to get a better understanding of the original text.

[2] In this paper the author demonstrates a comprehensive abstraction on the Indian dialects more specifically for the kannada language in relation to guided summarization. Abstractive summarization is the method of implementing a smaller version of a given

text document by collating only the important information in it. It also formats the text in such a way that it is much more simple to comprehend. The nonlinear format of the Indian language makes an already difficult process more complicated. This works aims to custom create IE process for Kannada language by implementing tagging rule such as NER. To prove the validity of this concept a case study consisting of about 50 documents in Kannada were taken, these papers were pertaining to miscellaneous aspects. Irrespective of the overall size of the document which are considered as inputs the total number of concepts for a particular category which are taken into account remains constant as they are custom made to suit the exact requirements. Evaluation is performed by comparing the summary to a human made summary for the same inputs. The results of the study indicates a proper representation of key aspects. When compared to the human made summaries few aspects where missed due to restriction in length but most of the key aspects where present. The monotony in the summaries is a consequence of this method. The system can be expanded to include a more properties. The summarized sentences can be represented in a more universal language or the text can be converted into speech.

[3] In this paper the authors perform research on abstractive summarization for Indian languages. The authors study the different methods of abstractive summarization while comparing to previous works in this field. Abstractive summarization is of two types semantic and structural. In semantic approach the process is divided into three steps in first step the text is fed into the system and a model is created making use of the different concepts. In the second step from the model important concepts are obtained in the final step a summary is generated. In structural approach important points are gleamed and create a predefined structure which is crucial to obtain the summary without losing its original meaning. This paper provides the reader with a séance of absence pertaining to the research conducted on this specific topic. It also provides a séance of motivation for research pertaining to abstractive summarization using previously unemployed methods.

[4] In this paper the authors proposes the creation of a parts of speech tagger for the kannada language which in turn assists in the analyzing and processing of the texts in kannada language . POS tagging is a necessary application in a myriad of different processes which can be performed to gain crucial information on any given language. In analyzing any dialect the ambiguity of the dialect should be properly processed to create an effective tagger. Due to the complexity of the kannada language it makes it difficult to

perform POS the authors hence proposes a method by which they create a tag set with 30 tags and created a tagger for the language with the assistance of SVM. The svm shows a limited count of lexicons at the start due to which the accuracy of the process is negatively affected. The process is repeated with a larger dataset and by using manual correction high accuracy is obtained. A group of texts which were taken from kannada documents and papers were analyzed and tagged with the mentioned technique. The results showed a high accuracy and increased effectiveness when compared with previous methods of tagging for the kannada language. The author concludes that the tagger can be used to create different applications in natural language processing for the Kannada language.

[5] In this paper the authors have explored abstractive summarization for the kannada language as research in this field has only been done for foreign languages. sArAmsha system which was designed involves processing the given documents in Kannada dialect and performing various language processing operation such as data analysis and processing information taken from the texts which helps in the creation of summaries. The abstraction methodology which has been implemented in this paper is much more effective in the creation of concise summaries than the extraction process. The process in this paper relies on extracting data and templates to create an abstractive summary. A list of different templates are created and the best one of them is chosen. The sentences created have to be short and have the most amount of information pertaining to the input data. The sArAmsha system segregates the dialects into processing and domain specific IE. This enables the system to be modified to include various other features. To validate the result the summaries from the system were compared to summaries made manually for this a group of 5 people were asked to submit summaries of certain documents . The same documents were fed to the system and both the summaries from the system and the summaries from the people were compared. The summaries obtained from the system had covered all key aspects and they provided high customer satisfaction. Lack of originality can become pane of the major side effects of this process. Future work needs to be done in order to increase the range of aspects provided by the system. Comparison needs to be done between different summarization systems to better improve the systems.

[6] In this paper the authors proposes POS tagging process based on the linguistic rules of dialects for the kannada language. POS tagging is the process of tagging words with grammatical titles . The authors proposes a method using CRF which is a machine learning process. The main goal in this paper is to design a flexible pos tagger which

provide high degree of accuracy and can be implemented on a large data set of words without having any negative drawbacks. A hierarchical tag set called AUKBC is implemented all forms syntactical labels .To validate the effectiveness of this method a dataset of 1000 words from Kannada newspapers magazines etc. were used and the data taken was divided into separate words by using a java program before the application was implemented. The result showed a high accuracy of slightly above 99 percent and 55% accuracy while AUKBC hierarchical tag set was used. For even better results certain features need to be initiated to analyses the suffixes . In future the author proposes study to implement name recognition for the Kannada language

[7]In this paper main objective is to summarize the single document. They have used Extractive approach in this approach pre-processing and processing steps are used and LSA tool is used. statistical and algebraic are the two methods they have used in the LSA tool and these methods shows the unseen structure of words. LSA tool extracts the source text and converts it into term sentence matrix. SVD algorithm is used to reduce the noise and illustrate the relationship between the sentence and the text. And then they calculate the human machine similarity scores this is also important when summarize the text.

[8] In this paper abstractive method is used to summarize the text document and different statistical approaches are used and 3 main steps have to be done i.e. pre-processing, summarizer and post-processing. The output summary should be structured and accurate. Here they are used 3 main steps and these steps have different methods, in pre-processing stemming, removal of stop words and POS tagging like this way they are used different methodology to get the proper text summarization. And also they focused on the size of the text, forecast the content then easily identify the text summarization.

[9] In this paper clustering and extraction are the main methods to summarize the text document. information retrieval and text classification, gives a correct solution to the information overload problem. Every cluster calculates the aggregation sentence based on the similar text. This clustering and extraction can improve the performance of text summarization.

[10]In this paper the web caused a humongous growth in the amount of growth of knowledge available to common man. Summaries of the inputs provide assistance to the required data and provide greater assistance when the input file is large. Important phrases are related to the document and content is shown

[11]This paper represents full hiding for dialects of India to be specific kannada in the form of summaries .The method produces abstractive summaries delving specifically on presentation model alongside IE (Information Extraction). TF/IDF rules implemented are divided into profiles. Implementation of templates makes the output informative.

[12] In this paper the need of an automatic summarizer capable of summarizing textual data in the original document without losing it's originality is explained . Summarization is applicable to all Indian Languages including Kannada as depicted by the researchers we can effectively summarize the contents of the given document using this abstractive summarizer.

3.3 Comparison of the results

Serial NO	Method (Extractive or Abstractive, CRF, Lexical, etc.)	Algorithm/ Summarizer tool	Result (may include Time Complexity and Accuracy)
1	Extractive method	Latent semantic analysis(LSA) tool is used to extract the source code and converted into term matrix this process algorithm called SVD.	In Single document there are some categories. Each of the categories have some scores and average is given in the last column based on this to indicates that domain knowledge and generate the summary
2	Abstractive method	Pre-processing, summarizer and post-processing	Predicted summary they got with control size which could be identify the relation between original text and summarized text
3	Extractive method	Supervised learning algorithm	Here summary is generated but lack of efficient summary is not getting it depends on the human.
4	Extractive method	Crawling, indexing, summarizer	Manual evaluation of the result is given. Different people may choose different sentence like way.
5	Abstractive method	TF, IE method is used for statistical	Computed for each automated summary to generate a summary.

Comparing above results with our implemented algorithm which is based on the TextRank. It's also a unsupervised learning methodology and it's a Extractive type of summarization. Our algorithm give efficient result than above mentioned. TextRank

algorithm provide automatic text summarization and customizing the user input and gives the desired result with efficient time . it will run efficiently both for CPU and GPU

3.4 Conclusion

We have observed that several techniques like SVM, CRF, Neural network model, Kernel based, latent semantic analysis yields good accuracy for POS tagging in English language but it's very limited to Kannada language (techniques used for Kannada). paper [6] proposed well fitted model(CRF) applied on online news articles and paper content as their dataset. CRF for the POS tagging gives the accuracy 99%. For the text summarization, two main techniques abstractive and extractive are popular one. With the significant analysis we found that an abstractive text summarization [2], gives the significant improvement and high accuracy when compared with extractive summarization. Accuracy result in the range of 80-85%; this methods reduce the redundant content in the summary of the document.

CHAPTER 4

PROJECT MANAGEMENT PLAN

4.1 Schedule of Project

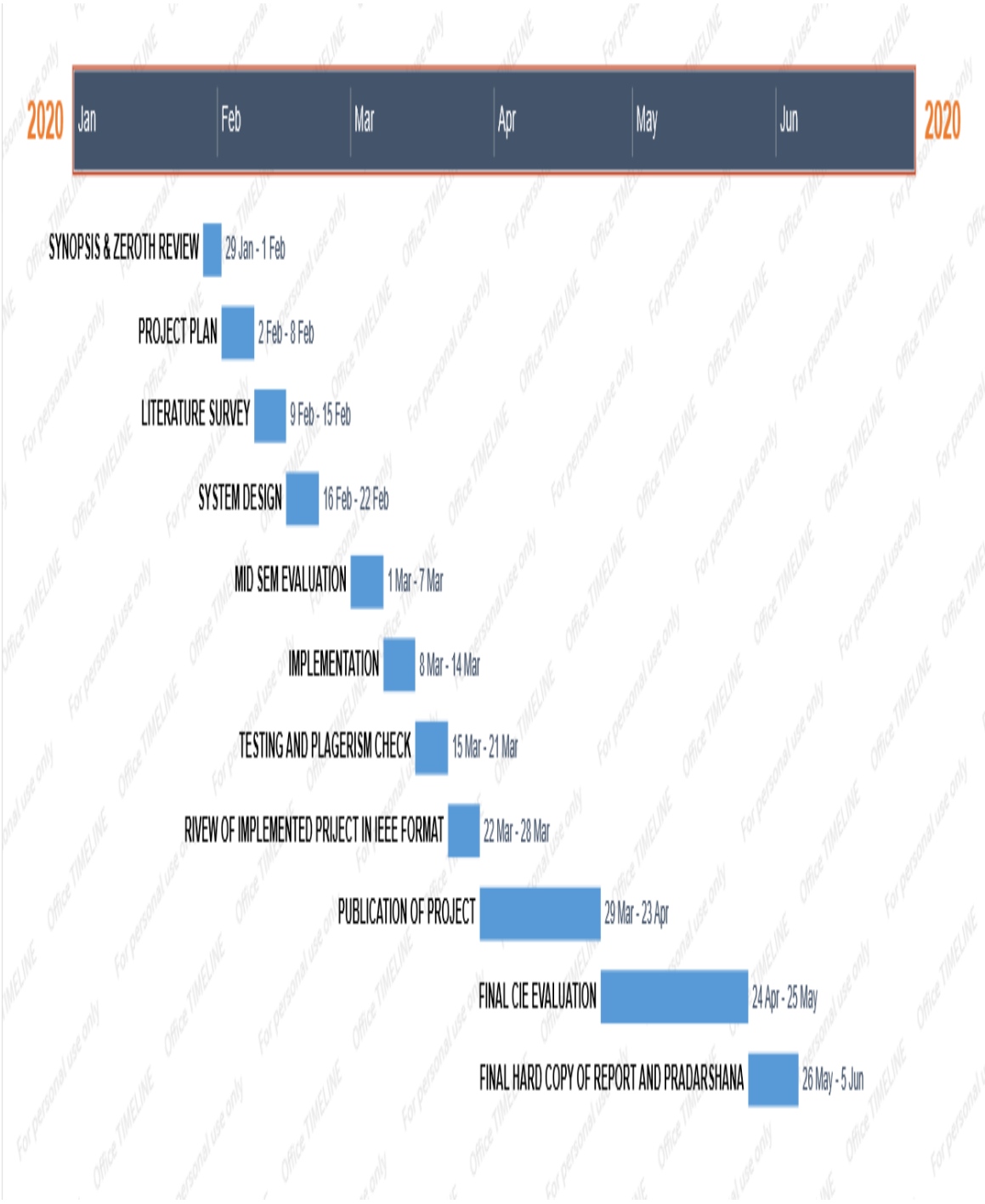


Chart 4.1

In chart 4.1 the schedule followed by us the authors of this report is displayed. It represents all the time taken for us to finish the project and what part of the project was finished when is display.

4.2 Risk Identification

Risk identification is the method of identifying threats to the project. In this paper we have created a text summarization device. Any number of variables may adversely affect the system and the output may not be the one desired. Each risk is assigned a value from 1-10 representing the likelihood of occurrence and impact. Some of the possible risks to this project are:

BUDGET RISK: In this type of risk the cost of implementing the project out ways the initial budget assessment. This in turn prevents resources from being attained and may lead to shutting down of the project.

RESOURCE RISK: In this type of risk the present resources may not last till the completion of the project this may be due to lack of budget or lack of proper management of project resources. This leads to halting of project or improper implementation of project.

QUALITY RISK: In this type of risk the quality of the end product may not be up to par with the initial estimation. This leads to customer dissatisfaction.

PROGRAM RISK: In this type of risk the program may not function due to error in coding. Which results in the project not giving the desired result.

LANGUAGE RISK: In this type of risk the input language may not match the language which the program was designed to summarize. This leads to error in output.

OUTPUT RISK: In this type of risk the output attained may not match the desired output for that specific input. This leads to faulty results.

CHAPTER 5

SOFTWARE REQUIREMENT SPECIFICATIONS

5.1 Product Overview

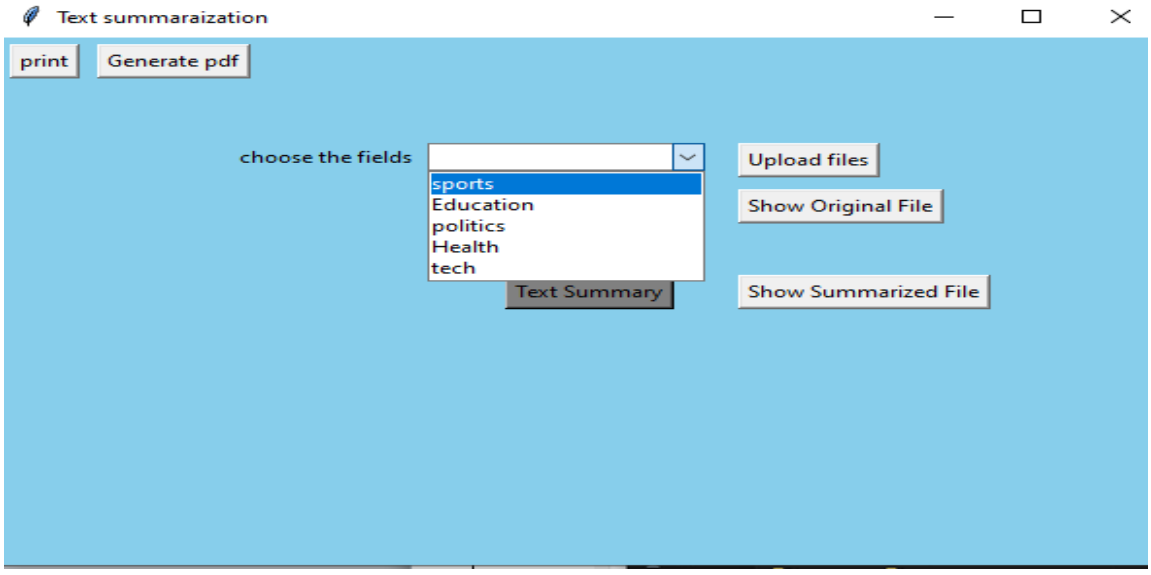
The proposed paper presents the facts and investigates the improvement of data of grammatical forms tagger more, content summarizer for one of the Indian territorial language for example Kannada. POS labelling is the way toward stamping up a word in a corpus as relating to the right grammatical features in Kannada. POS tagger will give the genuine which means of the word in an alternate situation with the goal that we can make the assessment examination of the specific report. POS gives the tagger as well as it will pass judgment on definition and setting of the word for Kannada this bode well in Kannada language, taking care of ambiguities in Kannada lexical things is likewise testing these days. Content outline in Kannada is the procedure of the decreasing the arrangement of information to make a subset of that information to speaks to the most significant and importance full data within the first substance this model will permit the client or per user to concentrate on the primary substance of the doc/content . we have chipped away at some examination papers which are for the most part cantered around the POS and Text summarizer in Kannada utilizing the NLP toolbox. Each paper gave us one of a kind technique and thought to assemble the effective tagger and furthermore summarizer. Tagger and summarizer both are very extraordinary however have inside relationship. Sentence can be broken into words and dissect the each word with morphological and semantic structure to make the tagger, This tagger will assist us with building the summarizer model. Regulated content rundown is a lot of like managed key phrase extraction. Essentially, on the off chance that you have an assortment of reports and human-created outlines for them, you can learn highlights of sentences that make them great possibility for consideration in the rundown. Highlights may remember the situation for the record (i.e., the initial barely any sentences are most likely significant), the quantity of words in the sentence, and so on. The fundamental trouble in directed extractive rundown is that the realized synopses must be physically made by separating sentences so the sentences in a unique preparing record can be marked.

5.2 External Interface Requirements

Since we have proposed to develop a web application the application is compatible on any device and can run on that particular devices specification , programming necessities creates familiarity and dialogue between the customers and conceptual producers or inform on the intended performance of the creation(In a capita enterprise these works might be performed by outside agencies).Programming necessities is complete mark up of prerequisites predating the explicit framework steps and its goal is to lessen future update. It to similarly provide a sensible premise to analyse timeline ; threats and product price Utilized correctly programming necessities information may be of assistance in forestalling disappointment in the system. The product prerequisites determination report notes down satisfactory and important necessities for advancement. To guess prerequisites the creator needs clear and exhaustive understanding of the products. This is achieved by the way of point by point and consistent interchanges with the programmers and customers all through advancement process

5.2.1 User Interfaces

User interface –it's GUI based Desktop Application which runs on the Windows and Linux based system, it will provide required feature as developer mentioned their guide. After installation a executable file will present in desktop and it will run.



5.2.2 Hardware Interfaces :

Not specific , but requires for the storage purposes and process the required functions. Our product is completely a software based one we don't need any additional hardware to support our application it can run on the current underlying hardware that is available. Equipment middleware is present in a considerable lot of segments to illustrate the multitudes of transport, collecting and storing technology and various other machines used for input of data and output etc. A machine panel to exchange information is depicted through the signs at the interface to illustrate SCSI separate's the future thought from the presentation of registering machines. Machine interface may correspond to a minute number of associations delivering portions of data meanwhile, or sequential where data are sent slightest bit at a single time. UI's, are consoles mice, orders and list's utilized due to correspondence among the user and PC. Models are order lines in OS and the GUI in window mac and Linux PC. Models are the order lines in DOS and Unix, and the graphical interfaces in Windows, Mac and Linux.

5.2.3 Software Interfaces

Operating system : Ubuntu , Windows

- Technology Used : NLTK , NumPy , Pandas , Python Tkinter
- IDE : PyCharm
- Back-End : Python 3
- Front-End : Python 3

5.2.4 Communication Interfaces

when user click on the desktop application it will run and open as show above image if everything fine then it will allow the end user to upload files. The communication starts here by uploading the files for the back end algorithms to process and generate the summary. This may include the direct of to the software and user don't know about the back end process and no need worry about the communication between user interface and backend algorithms

5.3 Functional Requirements

In the process of design of the system many components have been implemented from hard ware such as devices for storage to software such as PyCharm and Tkinter. We have used a GUI to display our inputs and our outputs, and text rank algorithm for summary generation. This functional Requirements includes the use case diagrams and the use case scenarios of our text summarization software. Below mentioned Functional requirements are more specific for the use cases.

5.3.1 Text Summarizer Requirements:

- The system will accept the whole Text document and separate it into Paragraph, sentences, and the words as tokens.
- After separating remove unwanted words like stop words in the kannada document file.
- Creating well trained word embedding's and dictionary creation
- Well usage cosine similarity function to get the similarly between pair sentences

5.3.2 Summarize text in user interface

- In the user interface we have provide some buttons to upload the kannada text files and show the summary of that document in separate windows.
- Process button to run the backend processes to extract the summary from the original document and standing for the show summary button clicks

5.3.3 TextRank Requirements:

- This is very essential requirements in functional part of the systems or program.
- This includes the vector creation, assigns the vector value to each word in sentence, and takes the average of the words, keep as major values.
- Define the cosine similarity to gather require sentences.
- Creating a graphs using the cosine score by the networks
- Graph as input to the Text Rank algorithms, process the text rank algorithm using the network libraries and generate all sentences with ranks.

CHAPTER 6

DESIGN

6.1 Introduction

In the designing section we have covered both analysis and development of system. We have represent the all requirement use case in diagrammatic representation which more helpful and suitable implementation part such coding. It include the system architecture, user interfaces and use case scenarios, and class and objects. Major representation show blow the labelling.

6.2 Architecture Diagram

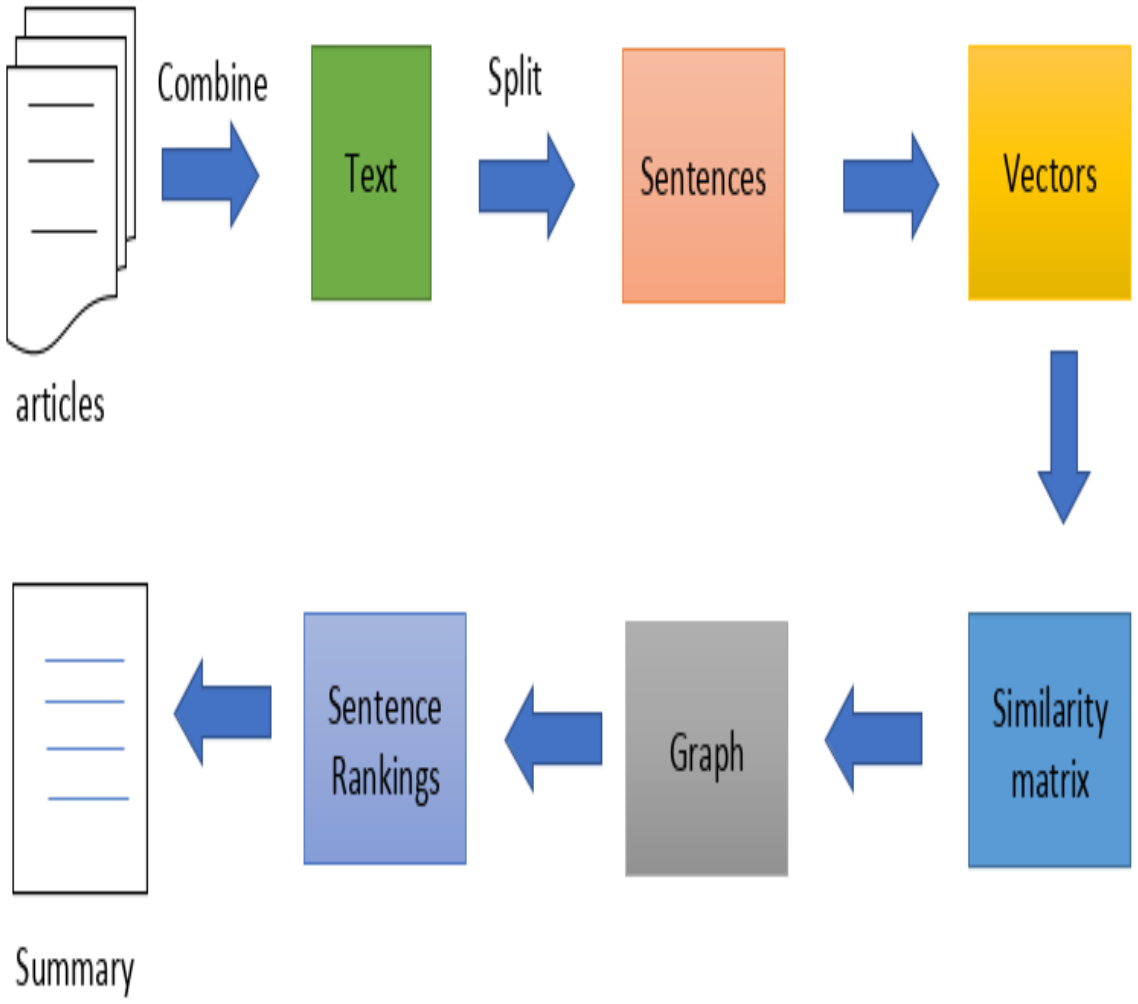
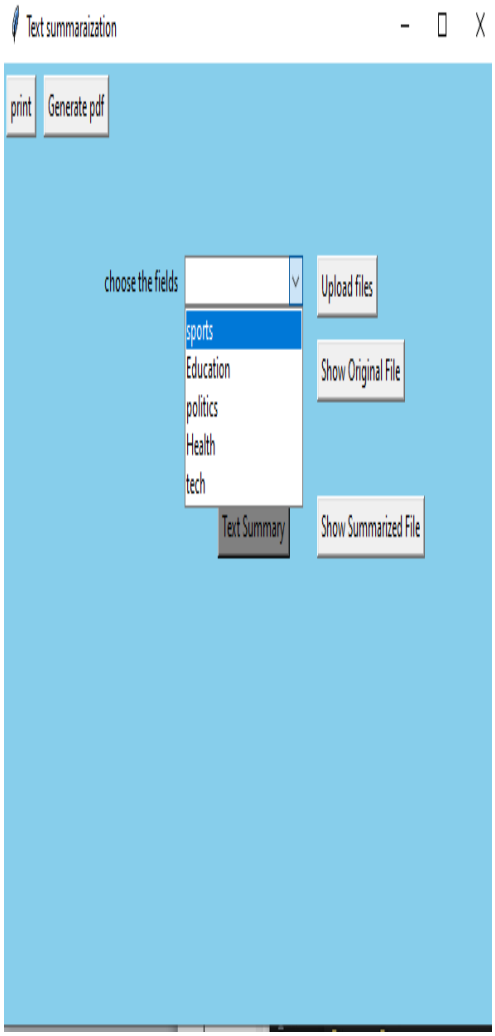


Figure 6.2.1

6.3 GUI(Graphical User interface):



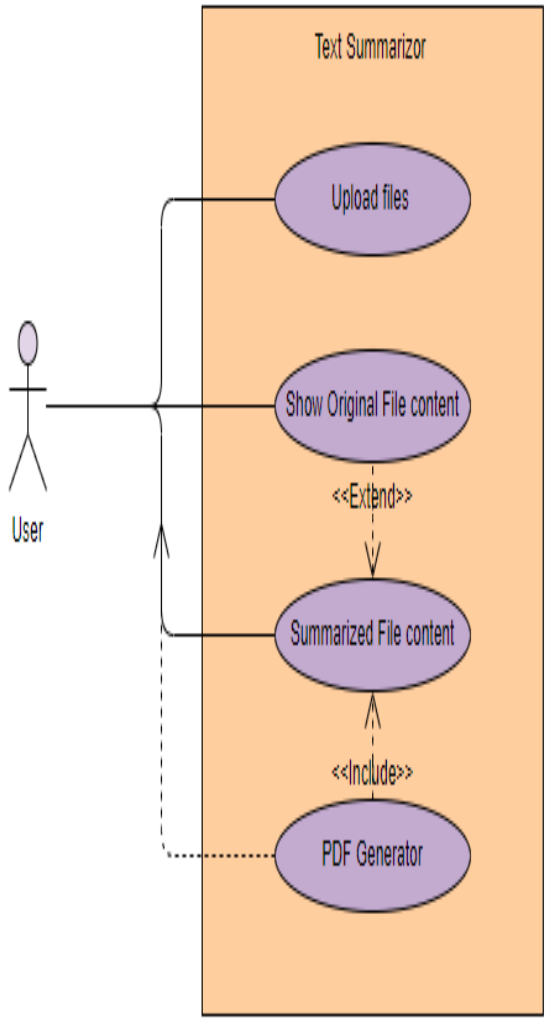


Figure 6.3.1 figure 6.3.2

figure 6.3.1 shows the user interface in the GUI. The pictorial representation also attached here in the figure 6.3.2. it shows the technical implementation of the GUI

6.4 Class Diagram and Classes

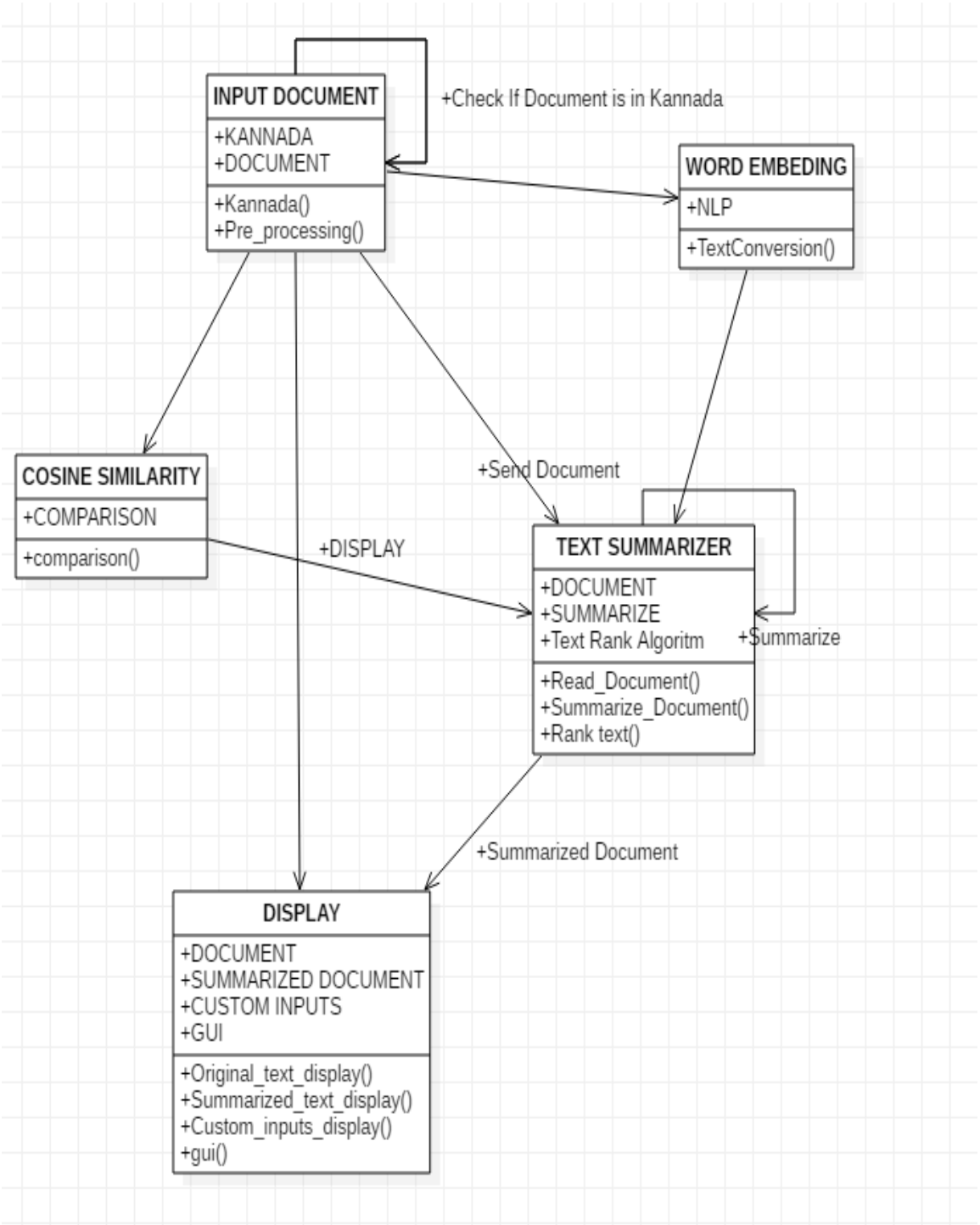


Figure 6.4.1

In Figure 6.4.1 the class diagram of the summarization software is displayed along with the classes of the software. The classes are in order of implementation input in which the kannada document is pre-processed and cleaned, word embedding where all possible words are embedded , cosine similarity where the similarity of the document sentences

are compared , text summarizer where the summarized form of the document is obtained and finally the summarized document is displayed in the display class.

6.5 Sequence Diagram

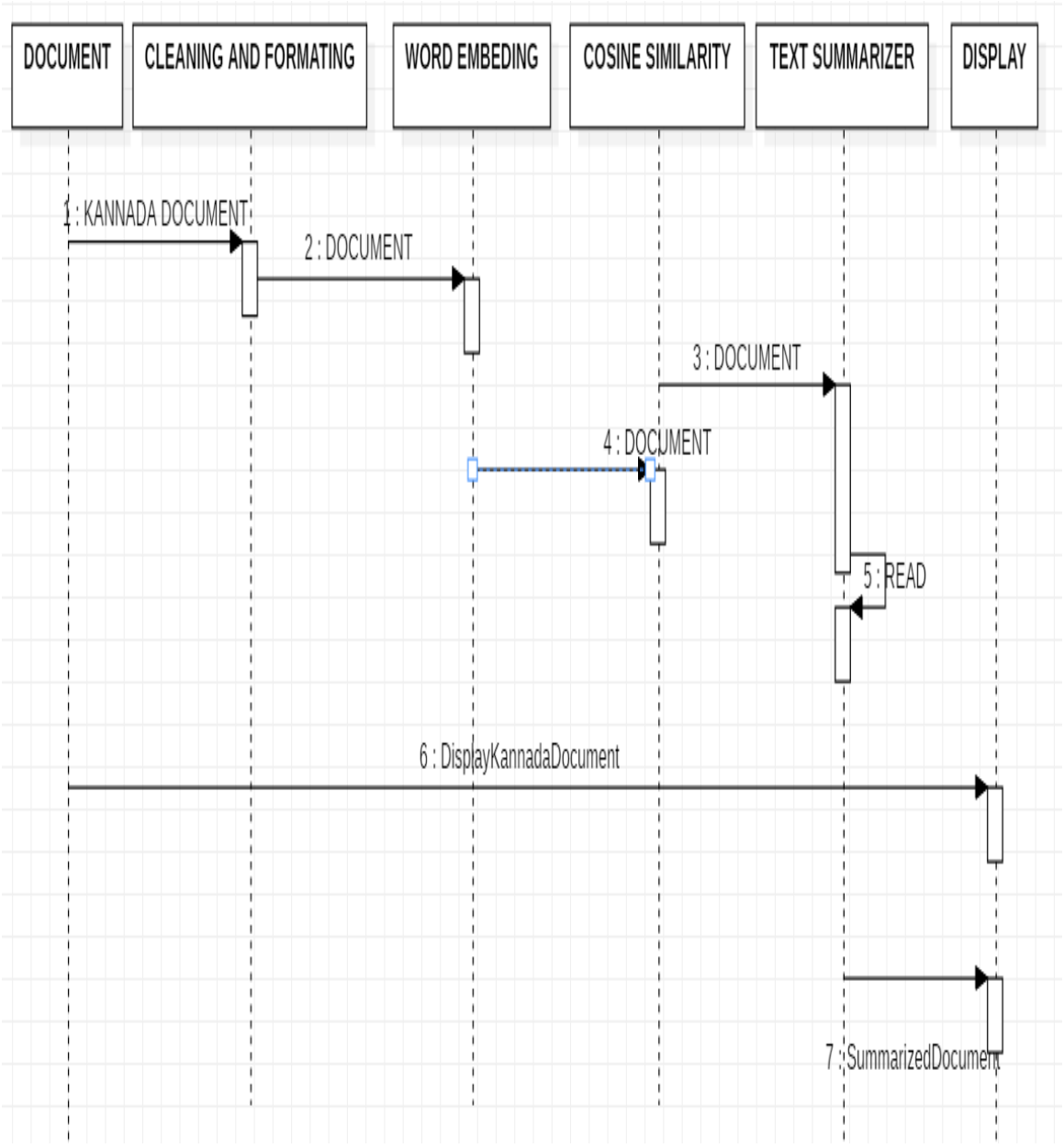


Figure 6.5.1

The above sequence diagram (Figure 6.5.1) shows the step by step illustration of how the software operates and its time intervals. The kannada document first is cleaned and formatted the stop words are removed the new document undergoes a similarity matrix and the it is summarized the summarized document is displayed along with the original kannada document.

6.6 Data Flow Diagram

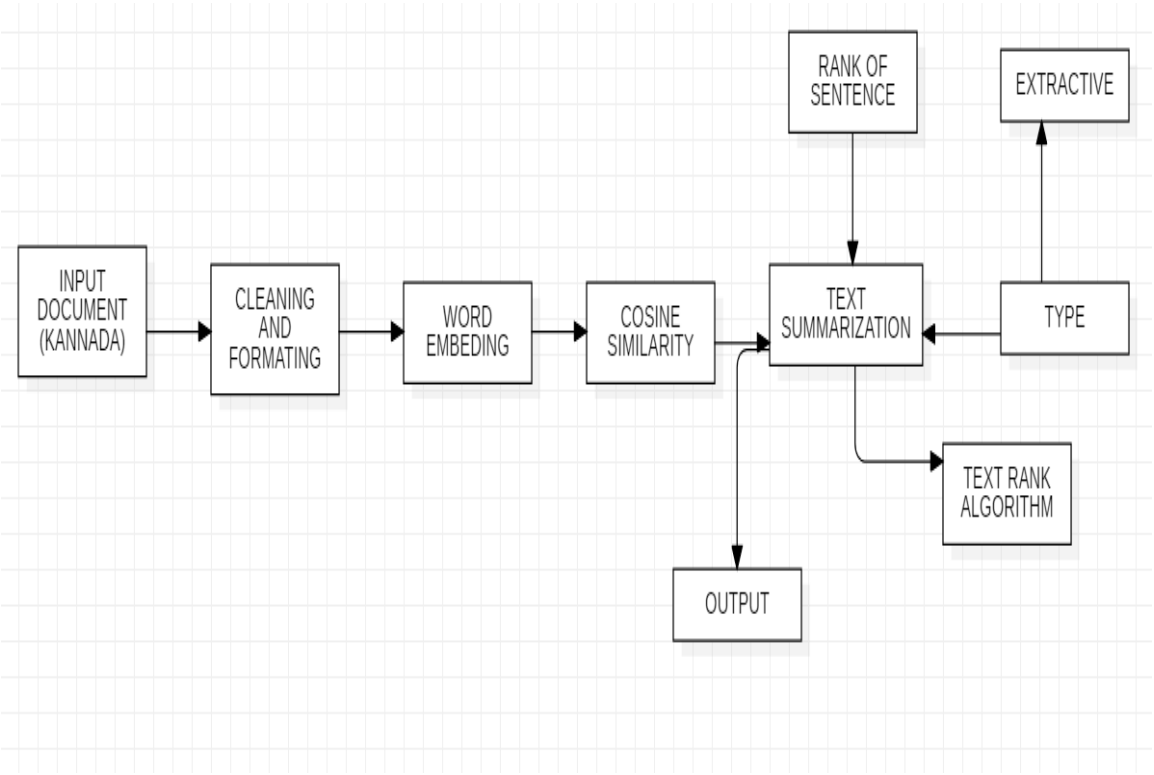


Figure 6.6.1

The data flow diagram represents the transfer of data which takes place in the software in the form of a block diagram Figure 6.6.1 . The data from the input document is cleaned and word embedding's are found the similarity matrix is compared and the document is summarized. The text summarizer uses extractive type and text rank algorithm which is mentioned in the dataflow diagram and finally the output is obtained .

6.7 Conclusion

Software design is the process by which the design specification for the program are represented it may be subject to change before implementation. It gives designers a method to represent all aspects of the software for implementation. The designs so far shown for text summarization show the step by step procedure followed by the program during implementation. First the document is cleaned and formatted the cleaned document undergoes word embedding .The embedded document is used to create a similarity matrix through which the summarized document is obtained by using text rank algorithm.

CHAPTER 7

IMPLEMENTATION

7.1 Tools Introduction

- **PYCHARM IDE**

Pycharm is an integrated development environment (IDE) used in programming , it provides a space whose sole purpose for coding using python. Pycharm is a cross platform software which can be implemented on mac window and Linux. Pycharm provides help in coding analyzing the code and fixing mistakes in a program. It provides easy access to files. It provides debugging for codes. It helps in unit testing the software code. It provides it's own API so that the programmers can implement their own plugins.

- **NLP**

NLP stands for natural language processing is a subdivision of linguistics, computer science, information engineering, and artificial intelligence It covers the relation between human language and computer language mainly on how to get the computer to process data which is in the form of natural data. The major difficulties of implementing NLP includes voice identification, comprehension of the spoken tongue, and natural language creation.

- **NUMPY**

NUMPY is a library used in python programming language adding support for huge, multi-dimensional arrays and matrices, accompanying massive groups of mathematical operations of a much more higher caliber to work on these arrays. Numpy pursuits the c-python reference implementation of Python, which is a non-evolving machine used to interpret bytecode language. Mathematical operations created for the mentioned specific model of Python regularly operate a great deal slower than compiled equivalents. NumPy solves this problem partially through presenting multidimensional arrays and features and operators that function effectively on arrays, requiring recreating some programs, often internal loops the use of Numpy. This array storage and interaction with data is greatly implemented by the python features of computer vision library Opencv. The NumPy array Is prevalent statistics shape in OpenCV for images, removed characteristic scores,

filter kernels and many additions greatly reduces the programming work done and the removing of errors and bugs from the program.

- **PANDAS**

In software engineering Pandas is a software library written for the Python coding language for the use of testing and controlling the information. To specify, it gives data structures and functions for controlling arithmetic values in tabular form and chronological series. Some of the features are

- Data frame for data manipulation,
- reading and writing data
- data alignment
- inputting data,
- deleting data
- filtering of data etc.

- **TKINTER**

Tkinter is the standard GUI library for Python. Python when blended with Tkinter presents a quick and effortless way to create GUI applications. It gives an effective object-oriented interface to the Tk GUI toolkit. Tkinter gives a chance for the combined operation of Python and Tcl in a single application by interpreting all Tkinter calls in to Tcl commands which are fed to an embedded Tcl -interpreter. In python Tkinter is the easiest and quickest method of creating a GUI.

7.2 Technology Introduction

- **PYTHON 3**

Python is an all-purpose comprehensive, sociable, object-oriented, and has a greater degree of programming language. PYTHON3 is a version of python which was released in 2008. Python is simple and readable it has fewer syntactical constructions than any other language. Python is easy to learn and easy to read and maintain a few of the important characteristics of python are

- It supports both functional and structural implementations as well as OOP
- It can be used as a scripting language

- It helps in debugging codes
- It can be easily integrated with other programming languages such as c,c++ etc.

➢ TKINTER GUI

Tkinter is the standard GUI library for Python. . In python Tkinter is the easiest and quickest method of creating a GUI. It provides widgets which can be used by the user to interact with the computer such as to input data or display data.

Machine learning

Machine Learning is the process of ingraining knowledge of computer programs that increases the quality on its own through previously learnt information. It is mostly considered as a subdivision of augmented intelligence. Ml programs construct an numerical mannequin primarily implemented by the way of pattern data, considered generally as data used for training, to achieve the goal of making forecasts or choices except being specifically created to perform so. These types of programs are implemented in a huge range of implementations such as language processing or image processing, where creating algorithms to perform the tasks become complex due to ever changing attributes it is also familiarly related to computer statistics because it can be used in crating prediction algorithms. For non-complex tasks assigned to computers, it is viable for the software programs commanding the system how to operate all cycles necessary to resolve the difficulty at present on the system without the need for learning. There are Three approaches for machine learning they are

- supervised learning - An example input is given with desired output
- unsupervised learning- for the given input no desired output is given the system needs to fin the structure of the inputs on its own.
- reinforcement learning - A program socializes with it's ever changing surroundings where it is required to reach a certain goal as it navigates its problem space Aside from these methods other methods have also been initiated such as modelling, metal earning etc.

7.3 Overall View of the Project In Terms Of Implementation

We have used python as our implementation language, it's very powerful in data science and statistical leaning. Implemented on windows 10, 64bit operating system. We have used couple of libraries and tools to implement this. Python is our choice to implement as the language perspective, below mentioned libraries have used here- Numpy, pandas,nltk, nltk.tokenize, os, networkx, time, sklearn.metrics.pairwise. these libraries make our implementation quite simple and easy.

Extractive text summarization – A high intensive text summarization method which summarization the kannada text without modifying the actual meaning or word in the given text, it will extract the important sentence from the paragraph of kannada word document and produce a desired summary.

TextRank Algorithm- in Natural language processing every task involves algorithm or method to get the desired results. In our Extractive text summarization we will be using TextRank algorithm, because it is rank based algorithm which is more preferable , suitable for the Extractive text summarization. It will produce the list of ranked sentences so that we can grab the high ranked sentence as our summary part. Before going to actual implementation we have done some woks as pre-training – Reading the input data from the user input and handling the kannada document.

Pandas: we have used pandas as our data handling library for the analysis. Reading the text file and convert this to a csv file for the further modifications and analysis;
Break the paragraph into individual sentences using the sent_tokenize(sentences) by passing the text data as argument which is a data frame in pandas.

Core Areas and usage for Kannada text summarization

- **Word embedding** : word embedding is a numerical data in NLP. It's

representation of a particular word-vector format. We can't analyze the given text data as it is in the analysis part, we will convert the given text data into numerical values so that we can perform various tasks on this data. Kannada word embedding gives us a context of word , semantic, Syntactic similarity, relation with other words. Actually this word embedding will gives us a dimension of kannada word, which means a how can we use a single word in different situation. ಕರಿ -0.13563 -0.17479 0.27192 -0.54677 -0.093725 -0.21872 0.040615 -0.40623 -0.28811 -0.14832 -0.65536 -0.58332 -0.016178 -0.41617 -0.00015106 0.21612 0.46569 -0.56685 0.0035116 0.30482 -0.19963 -0.49293 0.59506 -

0.32115 0.35145 0.6541 0.046347, here ‘kari’ is word which has different meaning in different scenarios. These numerical show the different meaning of the that word in different scenarios. It can be a verb, adverb, noun, etc. we have used pre trained word embeddings to get better results and cover all the dataset which nearly 400MB of the kannada words in 300 dimensions, source of the vector is here- https://fasttext.cc/docs/en/pretrained-vectors.html

- **Text preprocessing:** kannada is quiet difficult to handle in the morphological

perspective. It’s structure is total different from English like language. In this part we will be cleaning the data to make noise –free and efficient to use in the processing time. Here we will be removing the punctuations, numbers and special characters form the kannada text. stopword removal- ಮತ್ತು, ಹಾಗೂ, ಅವರು, ಅವರ, ಬಗ್ಗೆ ಎಂಬ, ಆದರೆ, ಅವರನ್ನು, ಆದರೆ, ತಮ್ಮ ,ಒಂದು, ಎಂದರು, ಮೇಲೆ, ಹೇಳಿದರು, ಸೇರಿದಂತೆ, ಬಳಿಕ...etc. these words can break the paragraph and make the individual sentences more meaningful manner. We have created a kannada stop word list and used here, to get the desired result.

- **Vector representation for kannada sentences:** kannada word embedding is like

a dictionary which means weightage or value of word in sentence. Using the word embedding, we have created numerical value to each word in sentence so that we can get complete numerical sentence. We will take vector of size our convenient for the constituents words in a sentence and then take the mean or average of those words to come a consolidated vector for the sentence. Each vector of required size will contain the 0.0964534 0.03232259 -0.0406382 0.07368313 0.03389427 - 0.01219876…values. Suppose [“ಪಂದ್ಯ”, “ ಮುಗಿಯುತ್ತಿದ್ದಂತೆ”, “ಸ್ಟೇಡಿಯಂನಲ್ಲಿ”,” ಹೊಡೆದಾಡಿಕೊಂಡ”, “ಅಭಿಮಾನಿಗಳು”], is our sentence, then , initially [0,0,0,0,0,0,0,0,0,0] which is our required size.”ಮುಗಿಯುತ್ತಿದ್ದಂತೆ”- value of this word in sentence is depends on the size of the sentence and occurrence in the sentence. We just take the numerical data from the word embedding and represented in the required format as vector. [0, 0.0354256, 0 ,0,0,0,0,0,0,0], at the end we have taken the average of these values and put into a vector so that we can make the analysis clear for the text summarization.

- **Similarity Matrix Preparation for the kannada sentence:**

till here we have got vector for each sentence which shows the numerical value of the sentence. Now we have to check the similarity between the sentence, this means we will be removing the duplicate and similar meaning sentence because to generate the summary we have to reduce the size of the text data. To achieve the similarity between the sentence we have used cosine similarity matrix to get the similarity. Below library - from sklearn.metrics.pairwise import cosine_similarity is taken for the finding the similarity between pair of sentences. Before using this library we have to prepare zero matrix of size "n x n" n represents each sentence size. After this step we will obtain matrix. We have passed the bunch of kannada vectors which means numerical data to this cosine function and then it will generate a matrix. We can observe the similarity in sentences.

- **Similarity Matrix to graph:** at this step we obtained sentences, vectors,similarity

scores. Graph is consist of nodes and edges. In our era let's consider sentences as nodes and similarity scores between sentence as edges . now create graph for this data. We have created graph reason behind this is that TextRank algorithm will work on the graphs and it's a graph based algorithm. So we have come text to numeric data and numeric data to graph. Graph=network.from_numpy_array(simialarity_matrix)

- **Algorithm Apply**

We have imported network library which contain several network based graph algorithm and functionalities. Now generate a ranking for each sentence so that we will know the importance of the sentence in the kannada text data.
Rank_score= network.pagerank(Graph)

- **Collecting Summary:** after applying the algorithm we have sorted the entire

Rank_socre array. Sorted in reverse order because higher value of the ranked sentence will have more importance and more weightage. Now in the front end we have given the option to user that they can put the size of the summary. After the taking the size we will fetch the high ranked sentences in top to bottom.

- **Result Analysis:** getting summary after the algorithm stops to generate the

Summary. We have compared with original document and summary document. In the result part top important sentences have gathered and combined in meaningful manner. There is no extra or new words will added in the result. It's clear and completely depends on the kannada input text. By this overview we have achieved the extractive based summarization. Increase the input data size will increase the time consuming on the CPU based execution. We have tested this system on CPU for the
space and time complexity. It's quite different from all other methods. Algorithm execution will take less time when compare to data preprocessing and word vector generation.

7.4 Explanation of Algorithm- TextRank Algorithm

TextRank algorithm is based on the page rank algorithm which more popular in web pages. pageRank algorithm is used to rank the web pages when people search particular content in browser it will display as the rank will it assign to that particular page, in the same manner textRank will use the following era-

- In place of web pages, we use sentences
- Similarity between any two sentences is used as an equivalent to the web page transition probability
- The similarity scores are stored in a square matrix, similar to the matrix M used for PageRank
- TextRank is particular depends on the text data which nothing but the kannada sentences and it will make more sense towards the relationship between the sentences.

For the implementation purpose, we have design the algorithm based on the problem statement as given below-

TextRank is an extractive and unsupervised text summarization technique.

Step 1: Start

Step 2: import required libraries –NLP and Machine leaning libraries

Step 3: Read the data which is kannada document file (text format)

Step 4: inspect and check the data

Step 5: split the text into sentences-tokenization

Step 6: Data preprocessing- for x in input data and clean each sentence

Keep required data items

Step 7: Word embedding's –create the word embedding's with specific size

For each word create a numerical value

Step 8: Vector Representation of sentences

Step 9: Similarity Matrix preparation

Step 10: applying page rank algorithm and give the rank to the sentences

Step 11: Summary Generation with custom range in user interface

Step 12. END

The textRank algorithm is a derivative of page rank algorithm in which the web pages are ranked by the number times visited and the number of pages which are linked to them. To explain text rank algorithm take a graph in which the vertices of the graph are sentences, and the weight of the edges between sentences denotes the similarity between sentences. Using the following steps this important sentences are extracted:

- The first step would be to concatenate all the text contained in the articles
- Then split the text into individual sentences
- In the next step, we will find vector representation (word embeddings) for each and every sentence
- Similarities between sentence vectors are then calculated and stored in a matrix
- The similarity matrix is then converted into a graph, with sentences as vertices and similarity scores as edges, for sentence rank calculation
- Finally, a certain number of top-ranked sentences form the final summary

CHAPTER 8

TESTING

8.1 Introduction

Testing is the process of evaluating whether the created software works as per requirement or not. In this project we have created an application that summarizes given document in Kannada language. We have tested the efficiency and working of this by comparing the achieved summarized documents using the software with the manually summarization of the same document. We have compared the two summarizations to check if the achieved summarization through the software covers all necessary information without lapsing any relevant data while using as little amount of lines as possible.

8.2 Testing Tools and Environment

For testing, the tools used are the summarization algorithm and software IDE Pycharm and documents extracted from various sources for summarization. The environment used is Window or Linux and other OS as the software is applicable on any OS

8.3 Test Cases

Serial no	**Test Scenario**	**Test Steps**	**Inputs**	**Expected Results**	**Obtained Results**	**Pass Or Fail**
1	NLTK ,numpy, network, and pandas library testing	1) Install the required nltk and pandas library 2) Import csv file and run the pandas as	Custom inputs to pandas as pd Observin: Nltk	Correctly printing required format of dataset Features	Correctly printing required format of dataset Features seen	PASS

		import pandas as pd 3) Install numpy 4) Check all the nltk feature	features Creating: Arrays Inputs to network	seen with nltk Created arrays and obtained desired results	with nltk Created arrays and obtained desired results	
2	Test with custom inputs to the library functionality on Pycharm IDE	Input the csv file and press with valid interpreter	CSV file	Reading file and displayed correctly Pycharm working with all the installed library	Reading file and displayed correctly Pycharm working with all the installed library	PASS
3	Reading kannada text data	Import the kannada	Input text file With encoding 'utf8'	Prints required contents in correct text	Prints required contents in correct text	PASS
4	Tokenizing	1.Convert given text file to csv 2.Analyse text data 3.With nltk tokenize generate tokens	Input kannada document	Kannada word tokens are generated	Kannada word tokens are generated	PASS
5	Remove stop words	Read the tokenize file	Input kannada	Successfully Removed	Successfully Removed	PASS

		and import kannada stop words Iterate each line and remove stop words	text data	Stop word to generate summary	Stop word to generate summary	
6	Word embedding	With cleaned sentences apply word embedding which obtained for kannada	Cleaned sentences	Obtained all the possible word embedding for the given document Appended in vector	Obtained all the possible word embedding for the given document Appended in vector	PASS
7	Similarity matrix i.e. cosine similarity	Process word embedding to generate similarity matrix	Vectors	Similarity between the sentences	Similarity between the sentences	PASS
8	Summary generation	Apply the cosine similarity to page rank algorithm Input the required data	Similarity matrix	Successfully generates the summary	Successfully generates the summary Summary depends on the user input size	PASS

9	summary generation in different language from kannada	Apply the cosine similarity to page rank algorithm Input the required data	similarity matrix	successfully generate in kannada	failure to generate summary	FAIL
10	Summary generation	Apply the cosine similarity to page rank algorithm Input the required data	similarity matrix	Successfully Generate Summary	summary generated does not cover all necessary information	FAIL

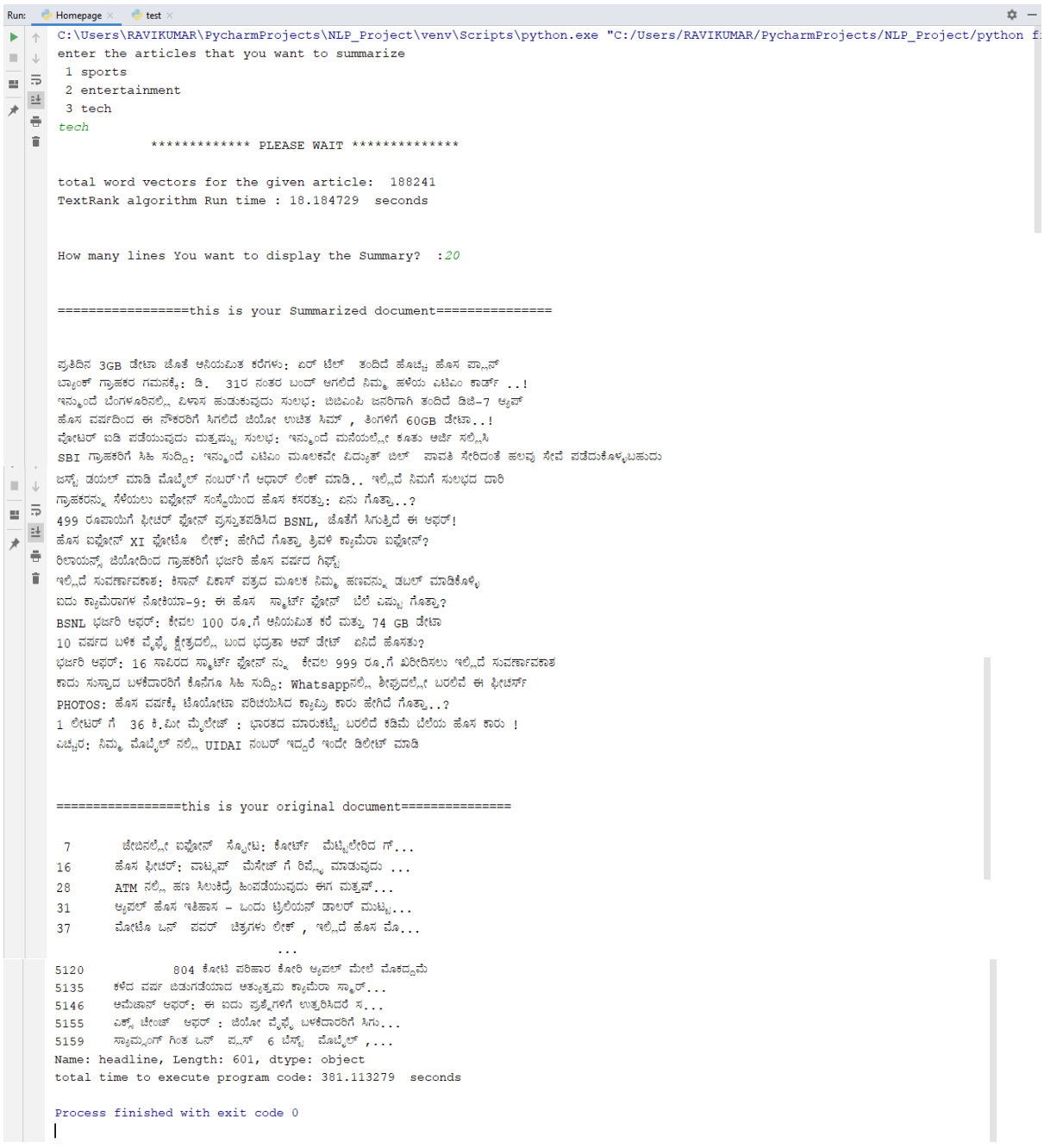
CHAPTER 9

RESULTS & PERFORMANCE ANALYSIS

9.1 Result Snapshots

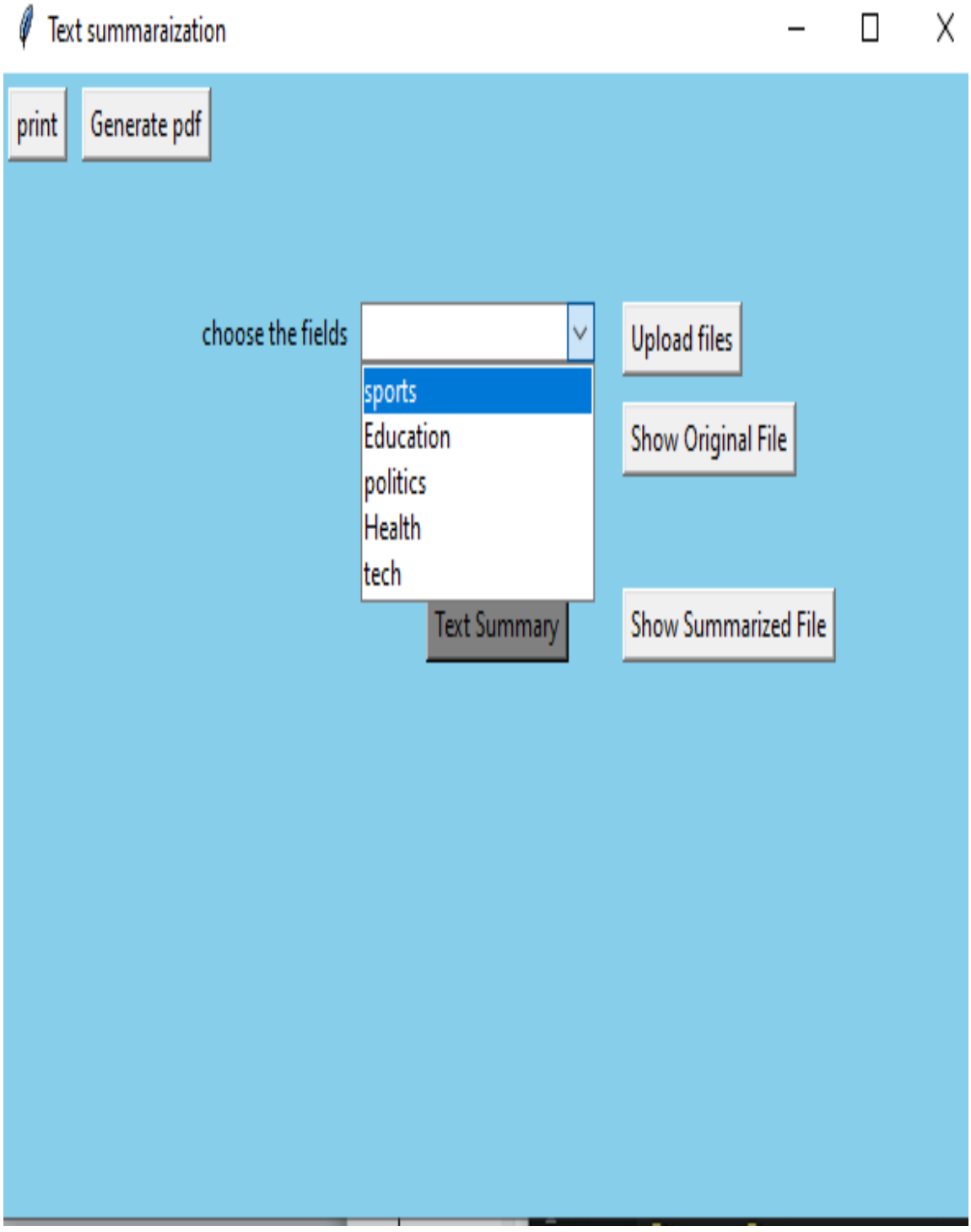
These snapshots contain the required output and input and showing the desire results:

- **Algorithm perspective**:



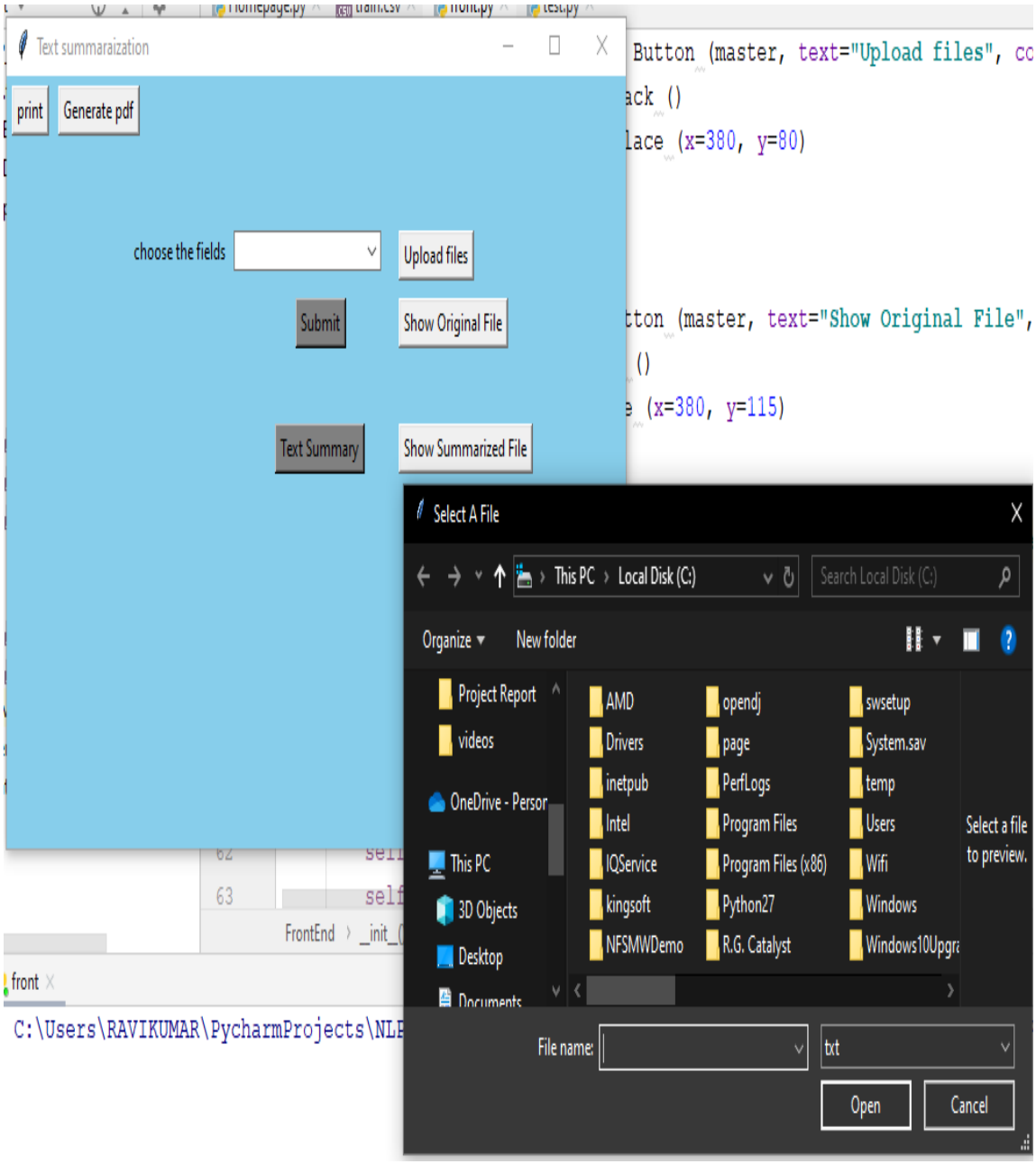
Screen shot 9.1.1

✸ **User interface working:**



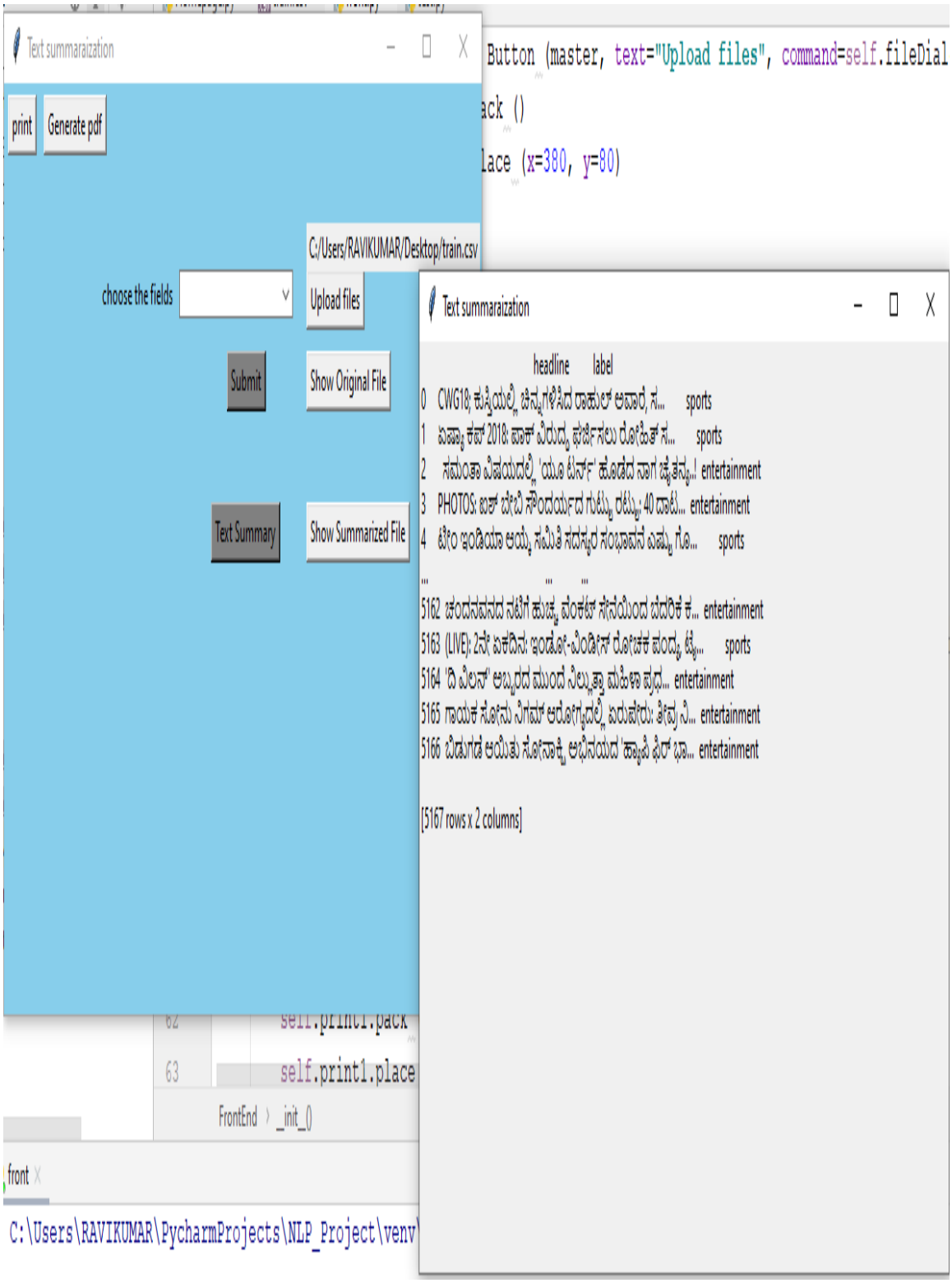
Screen shot 9.1.2

✸ Upload Files



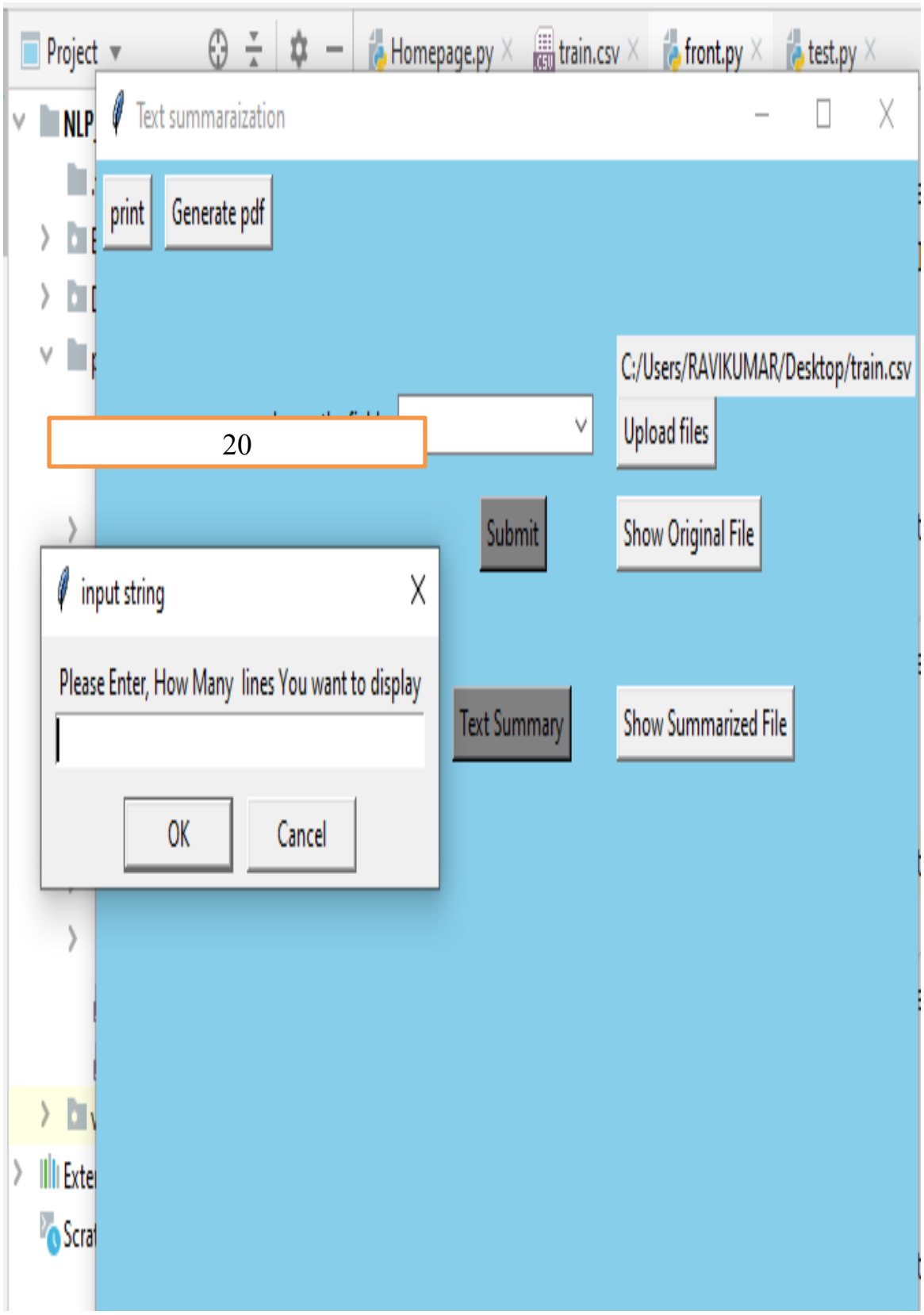
Screen shot 9.1.3

✸ Original File Display and Summary



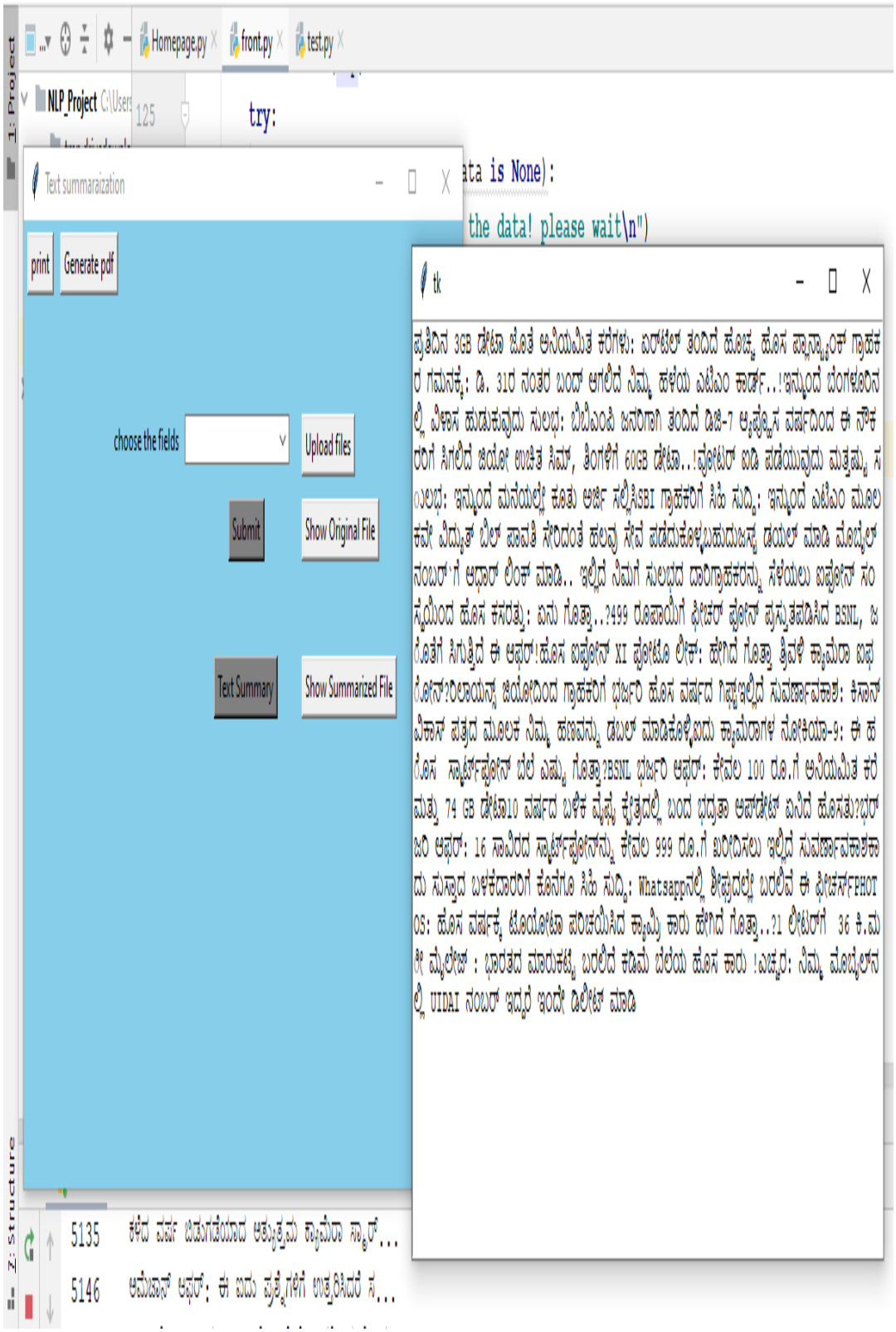
Screen shot 9.1.4

✸ **Number of lines displaying –custom input**



Screen shot 9.1.5

✸ **Required Number of lines –Summary Generate**



Screen shot 9.1.6

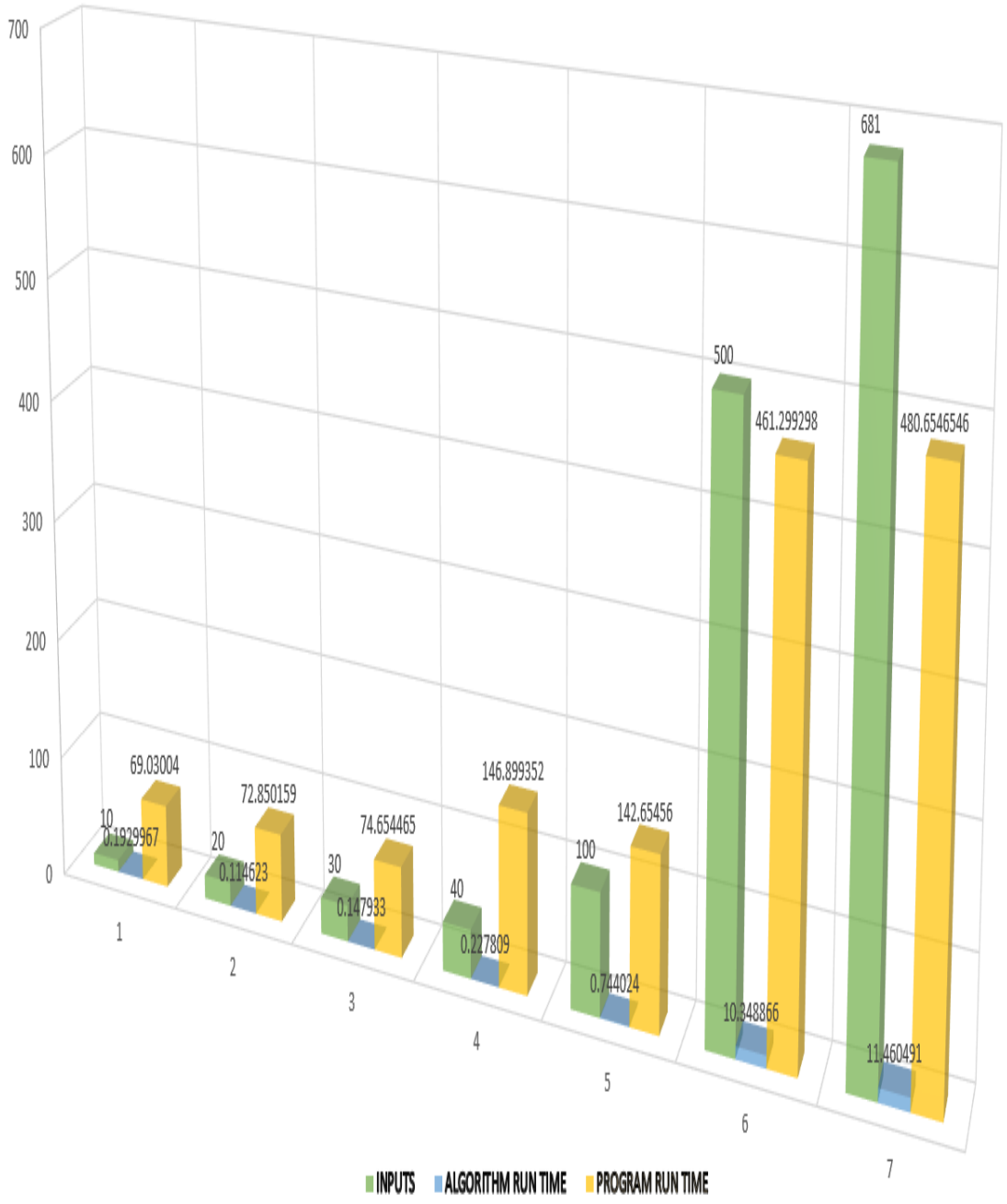
9.2 Comparison Results Tables –from survey papers results

Serial NO	Method (Extractive or Abstractive, CRF, Lexical, etc.)	Algorithm/ Summarizer tool	Result (may include Time Complexity and Accuracy)
1	Extractive method	Latent semantic analysis(LSA) tool is used to extract the source code and converted into term matrix this process algorithm called SVD.	In Single document there are some categories. Each of the categories have some scores and average is given in the last column based on this to indicates that domain knowledge and generate the summary
2	Abstractive method	Pre-processing, summarizer and post-processing	Predicted summary they got with control size which could be identify the relation between original text and summarized text
3	Extractive method	Supervised learning algorithm	Here summary is generated but lack of efficient summary is not getting it depends on the human.
4	Extractive method	Crawling, indexing, summarizer	Manual evaluation of the result is given. Different people may choose different sentence like way.
5	Abstractive method	TF, IE method is used for statistical	Computed for each automated summary to generate a summary.

Comparing above results with our implemented algorithm which is based on the TextRank. It's also a unsupervised learning methodology and it's a Extractive type of summarization. Our algorithm give efficient result than above mentioned. TextRank

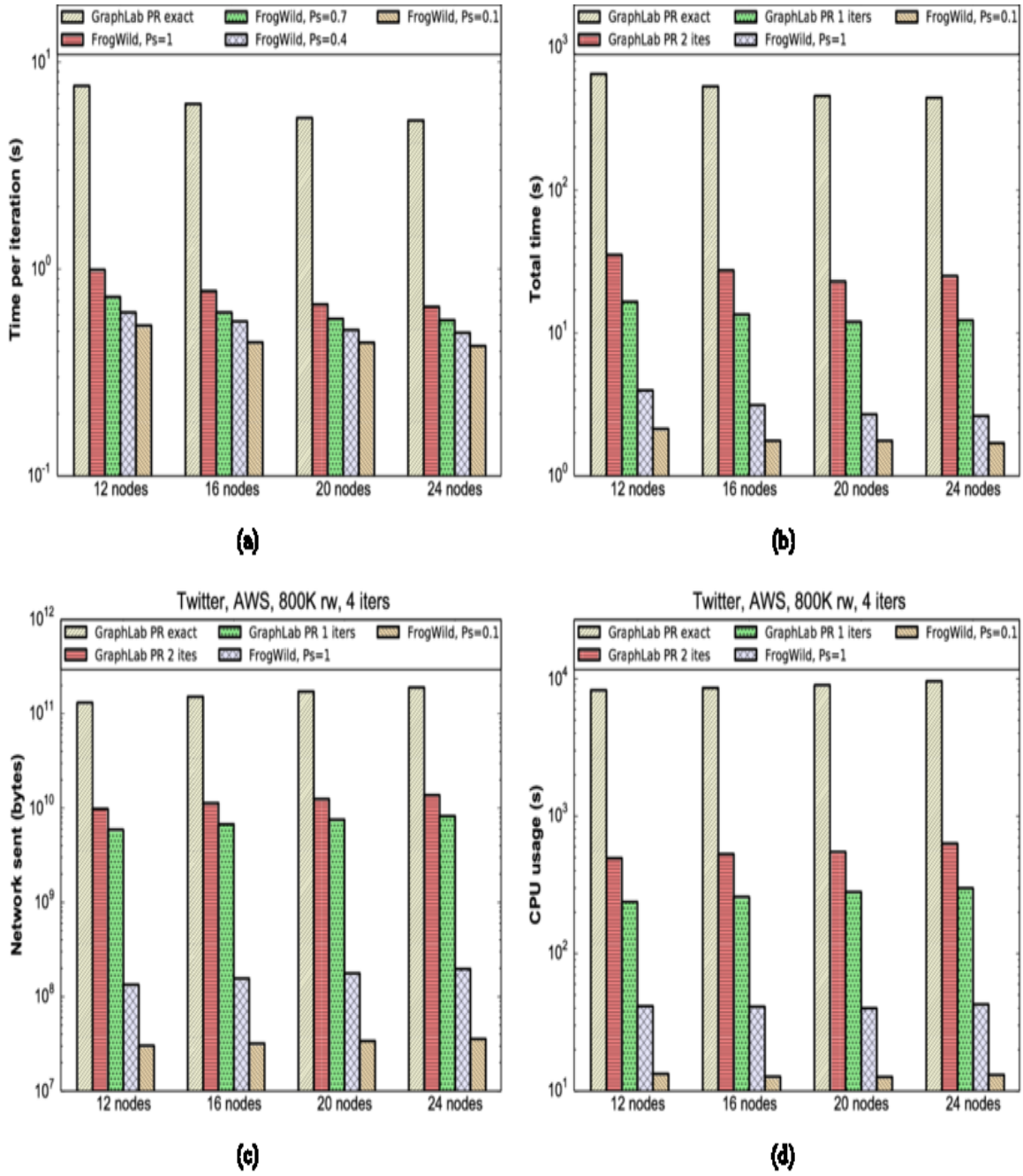
algorithm provide automatic text summarization and customizing the user input and gives the desired result with efficient time . it will run efficiently both for CPU and GPU

9.3 Performance analysis – graphs, tables etc..



Graph 9.3.1

In the above graphs contain the information about the user input and the time required to run-CPU



Graph 9.3.2

Graph 9.3.2 shows the Page rank performance for various number of nodes Pager rank algorithms is backbone for or text rank algorithm, exactly same process happenings in the both algorithm, instead of web page we are taking the sentence

CHAPTER 10

CONCLUSION & SCOPE FOR FUTURE WORK

Conclusion

Extractive Text summarization is gathering important sentences from a given document. We have concentrated more on implementation part of this project here. TextRank is graph-based algorithm which is similar to google PageRank algorithm. Here we are referring each sentence as our page link in PageRank. This paper conclude the complete implementation of Extractive Text summarizer. We have implemented front end in the python using tkinter which is GUI for handling our results and providing.

GUI application will allow the end user to give the input as text file in kannada and upload it. Rest of things handled in the backend algorithm. After sometimes when the processing done, it will generate the summary of that document in human readable format in front end. And also original document. You can compare the original and summarized document. We also provided some extra features like after you done the thing you can print the summary in portable format like pdf.

Limitation of this Research Work

- It limits to the document size – which nearly 6000 lines of text.

If text is larger than this size it will give a time out to end user.

- It accepts only csv and text files, it will give the warning , if you upload pdf or other kind of documents type
- Minimum hardware, 4GB of RAM- if not it will take large amount of time.
- It applicable for Laptop, PC, Tablet, any other desktop – because of GUI
- It can’t run on Mobile devices.
- It contains word embedding in 300 dimensions.
- Similarity calculation is based on the cos function and it’s built in NLP, don’t compare the similarity with other tools
- Execution time of the TextRank algorithm depends on the size of the input data.

Directions for the Future works

We have tried as our level best to gather all the required information and requirement to meet our analysis part. It meets our majority goals and objective of this work. Suppose if you find any mistakes or any missing feature or incomplete always you can extend our existing work. It as a major advantage to south Indian like Telagu, Tamil, Malayalam, etc. our direction is to meet our required objectives such as easy user interface and efficient Text summarization. In order to achieve these goals , must be review the existing works which we have shown our results. Future works can solve our limitation which we have mentioned earlier. And this can be as suitable reference to each of the Extractive Text summarization work for the regional languages. For our suggestion it needs a lot more basics of NLP and Data science. So our theme for the direction is to build strong basics including he coding and analysis to move forward in Text summarizations. You extent this research to build Translator for your regional Language.
This project contains very basic of major tools required for NLP processes. So make sure all explained tools and concepts to clear for further works. Further works may include , efficiency, accuracy , different methods, techniques etc..

REFERENCES APPENDICES:

[1] E. Reategui, M. Klemann and M. D. Finco, Using a Text Mining Tool to Support Text Summarization, IEEE 12th International Conference on Advanced Learning Technologies, DOI: 10.1109/ICALT.2012.51, 2012, pp. 607-609.

[2] Shilpa G V, Shashi Kumar D R, Abs-Sum-Kan An Abstractive Text Summarization Technique for an India Regional Language by Induction of Tagging Rules, International Journal of Recent Technology and Engineering (IJRTE), ISSN: 2277-3878, 2019

[3] Sunita C, A Jaya, Amal Ganesh, A study on abstractive summarization in Indian language, Fourth Recent Trends in Computer Science and Engineering, DOI:https://doi.org/10.1016/j.procs.2016.05.121, 2016, pp. 25-31.

[4] P. J. Antony and Dr. Soman K. P., Kernel based part of speech tagger for kannada, International Conference on Machine Learning and Cybernetics (ICMLC), DOI:10.1109/ICMLC.2010.5580488, 2010, vol. 4, pp. 2139-2144.

[5] V. R. Embar, S. R. Deshpande, A. K. Vaishnavi, V. Jain and J. S. Kallimani, sArAmsha - A Kannada abstractive summarizer, International Conference on Advances in Computing, Communications and Informatics, DOI: 10.1109/ICACCI.2013.6637229, 2013, pp. 540-544.

[6] Pallavi, Anitha S Pillai, Parts Of Speech (POS) Tagger for Kannada Using Conditional Random Fields (CRFs), National Conference on Indian Language Computing, 2014

[7] Geetha J K and Deepamala N, Kannada text summarization using Latent Semantic Analysis, International Conference on Advances in Computing, Communications and Informatics, DOI: 10.1109/ICACCI.2015.7275826, 2015, pp. 1508-1512.

[8] Kallimani, Jagadish S, K G Srinivasa, and Eswara Reddy B, Information extraction by an abstractive text summarization for an Indian regional language, International

Conference on Natural Language Processing and Knowledge Engineering, DOI: 10.1109/NLPKE.2011.6138217, 2011, pp. 319-322.

[9] Pei-ying Zhang, Cun-he Li, Automatic text summarization based on sentences clustering and extraction, 2nd IEEE International Conference on Computer Science and Information Technology, DOI: 10.1109/ICCSIT.2009.5234971, 2009, pp. 167-170.

[10] Jayashree.R, Srikanta Murthy.K and Sunny.K, Document Summarization in Kannada Using Keyword Extraction, First International Conference on Artificial Intelligence, Soft Computing and Applications, DOI: 10.5121/csit.2011.1311, 2011, pp. 121–127.

[11] Santosh Kumar Bharti , KorraSathya Babu , and Sanjay Kumar Jena, Automatic Keyword Extraction for Text Summarization: A Survey, ArXiv,2017, Vol-abs/1704.03242, Corpus ID: 23384543.

1. Data Set- https://kannada.webdunia.com/
2. Word Embedding's - https://fasttext.cc/docs/en/pretrained-vectors.html
3. Publications- 2020 5th IEEE International Conference on Computing, Communication and Automation (ICCCA-2020) Jointly organized by Aurel Vlaicu University of Arad, Romania & Galgotias University, India Conference Website: www.iccca.in Conference Email Id: iccca2020@gmail.com ISSN for ICCCA (Online): 2642-7354, ISBN: 978-1-7281-6324-6 (Xplore), 978-1-7281-6323-9 (USB) An IEEE Industry Applications Society Financial Sponsored (100%)