A

Major Project Report

on

TEXT ANALYSIS USING MACHINE LEARNING

Submitted in partial fulfillment of the
Requirements for the award of degree of

Bachelor of Technology

in

Computer Science and Engineering

by

GAJAM NIKHIL
(19H61A05D5)

EEDUNOORI KRUTHIK REDDY
(19H61A05D1)

GADIPE DOLLY
(19H61A05D4)

Under the Guidance of

Mr. G. Balram
(Asst. Professor)
Department of CSE

Department of Computer Science and Engineering
ANURAG GROUP OF INSTITUTIONS
(Formerly CVSR College of Engineering)
(An Autonomous Institution, Approved by AICTE and NBA Accredited)
Venkatapur (V), Ghatkesar (M), Medchal(D)., T.S-500088
(2019-2023)

Anurag Group of Institutions
An Autonomous Institution
(Formerly CVSR College of Engineering)
Venkatapur (V), Ghatkesar (M), Ranga Reddy (Dist.)
Ph: 08499953666, 08499963666. www.anurag.edu.in

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the project entitled **"TEXT ANALYSIS USING MACHINE LEARNING"** being submitted by **GAJAM NIKHIL, EEDUNOORI KRUTHIK REDDY, GADIPE DOLLY** bearing the Hall Ticket number **19H61A05D5, 19H61A05D1, 19H61A05D4** in partial fulfillment of the requirements for the award of the degree of the **Bachelor of Technology** in **Computer Science and Engineering** to **Anurag Group of Institutions (Formerly CVSR College of Engineering)** is a record of bonafide work carried out by them under my guidance and supervision from November 2022 to March 2023.

The results presented in this project have been verified and found to be satisfactory. The results embodied in this project report have not been submitted to any other University for the award of any other degree or diploma.

Internal Guide
Mr. G. Balram
(Asst. Professor)
Department of CSE

External Examiner

Dr.G.Vishnu Murthy,
Professor & Dean, Dept. of CSE

ACKNOWLEDGEMENT

It is our privilege and pleasure to express profound sense of respect, gratitude and indebtedness to our guide **Mr. G. Balram, Asst Professor,** Department of Computer Science and Engineering, Anurag Group of Institutions (Formerly CVSR College of Engineering), for his indefatigable inspiration, guidance, cogent discussion, constructive criticisms and encouragement throughout this dissertation work.

We express our sincere gratitude to **Dr. G. Vishnu Murthy, Professor & Dean**, Department of Computer Science and Engineering, Anurag Group of Institutions (Formerly CVSR College of Engineering), for his suggestions, motivations, and co-operation for the successful completion of the work.

We extend our sincere thanks to **Dr. V. Vijaya Kumar**, **Professor & Dean,** School of Engineering, Anurag Group of Institutions, for his encouragement and constant help.

GAJAM NIKHIL
(19H61A05D5)

EEDUNOORI KRUTHIK REDDY
(19H61A05D1)

GADIPE DOLLY
(19H61A05D4)

DECLARATION

We hereby declare that the project work entitled "**TEXT ANALYSIS USING MACHINE LEARNING**" submitted to the **Anurag Group of Institutions (Formerly CVSR College of Engineering)** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology (B. Tech)** in Computer Science and Engineering is a record of an original work done by us under the guidance of **Mr. G. Balram, Assistant Professor** and this project work have not been submitted to any other university for the award of any other degree or diploma.

GAJAM NIKHIL
(19H61A05D5)

EEDUNOORI KRUTHIK REDDY
(19H61A05D1)

GADIPE DOLLY
(19H61A05D4)

Date:

ABSTRACT

This project is focused on analyzing and processing textual data, using a combination of techniques including spam detection, text summarization, and emotional analysis. The aim of the project is to provide users with a comprehensive understanding of the text they are analyzing, including its overall sentiment, relevance, and potential for spam or malicious content. The spam detector module analyzes the text for common spam keywords and phrases, flagging any potential spam or malicious content. The text summarization module summarizes the content of the text, providing a brief and concise overview of its most important points. The emotional analysis module evaluates the overall sentiment of the text, identifying positive or negative emotions and providing a sentiment score. The project uses a variety of machine learning and natural language processing techniques to analyze the text, including neural networks, sentiment analysis algorithms, and statistical modeling. The system is designed to be highly scalable and flexible, allowing it to be customized for a wide range of applications and industries. The project has the potential to be used in a variety of applications, including social media monitoring, customer feedback analysis, and content moderation. By providing users with a comprehensive understanding of the text they are analyzing, the project can help improve decision-making and enhance the overall quality of communication and content in various industries.

INDEX

1. INTRODUCTION

The rapid growth of digital communication and data has led to an explosion in the amount of textual data available, from social media posts to customer feedback and product reviews. However, analyzing and processing this data can be a time-consuming and complex task, particularly when trying to extract useful insights or identify potential issues such as spam or malicious content. To address this challenge, this project was developed to provide users with a comprehensive understanding of textual data by combining three key techniques: spam detection, text summarization, and emotional analysis. The project uses machine learning and natural language processing algorithms to analyze the text and extract key insights, providing users with a more accurate and efficient way to process and understand textual data. The spam detector utilizes machine learning algorithms to accurately identify and flag unsolicited or unwanted messages. The text summarization module uses advanced techniques to condense large amounts of text into short and concise summaries, making it easier for users to quickly understand the main points. The emotional analysis component employs sentiment analysis to determine the emotions conveyed in the text, enabling users to gauge the tone and intent of the message. The integration of these three modules creates a powerful tool that can assist users in managing their messages more effectively, enabling them to identify important information quickly and easily while filtering out irrelevant or undesirable content. The project has the potential to be used in a variety of applications, including social media monitoring, customer feedback analysis, and content moderation. By providing users with a comprehensive understanding of the text they are analyzing, the project can help improve decision-making and enhance the overall quality of communication and content in various industries.

1.1 MOTIVATION

The motivation for this project stems from the growing need to efficiently process and understand large amounts of textual data. With the rise of social media and online communication, businesses and organizations are constantly inundated with feedback, reviews, and messages from customers and stakeholders. It can be challenging to extract useful insights from this data, particularly when it is rife with spam, irrelevant information, or potential malicious content. By combining three key techniques - spam detection, text summarization, and emotional analysis - this project offers a comprehensive solution to help businesses and organizations better understand textual data. The spam detector module filters out unwanted content, ensuring that users are only analyzing relevant information. The text summarization module provides a concise and accurate summary of the text, making it easier to quickly understand the most important points. Finally, the emotional analysis module evaluates the sentiment of the text, providing valuable insights into the overall tone and mood of the communication. The project is highly scalable and customizable, making it suitable for a wide range of applications and industries. It has the potential to help businesses make more informed decisions, identify potential issues early on, and improve the overall quality of communication and content. Ultimately, the project aims to provide a more efficient and effective way to process and understand textual data, helping businesses and organizations to stay ahead of the competition.

1.2 PROBLEM DEFINITION

The problem addressed by this project is the challenge of efficiently processing and understanding large amounts of textual data. With the rise of social media and online communication, businesses and organizations are constantly inundated with feedback, reviews, and messages from customers and stakeholders. However, analyzing and extracting useful insights from this data can be a time-consuming and complex task, particularly when it is rife with spam, irrelevant information, or potential malicious content. The project aims to address this challenge by combining three key techniques - spam detection, text summarization, and emotional analysis - to provide users with a more efficient and effective way to process and understand textual data. The spam detector module filters out unwanted

content, ensuring that users are only analyzing relevant information. The text summarization module provides a concise and accurate summary of the text, making it easier to quickly understand the most important points. Finally, the emotional analysis module evaluates the sentiment of the text, providing valuable insights into the overall tone and mood of the communication. By addressing this problem, the project aims to help businesses and organizations make more informed decisions, identify potential issues early on, and improve the overall quality of communication and content. It provides a comprehensive solution that is highly scalable and customizable, making it suitable for a wide range of applications and industries.

1.3 OBJECTIVE OF THE PROJECT

The main objective of this project is to provide users with a comprehensive solution for efficiently processing and understanding textual data. To achieve this objective, the project combines three key techniques - spam detection, text summarization, and emotional analysis - to extract useful insights and provide a more accurate understanding of the text.

Specifically, the objectives of the project are:

1. To develop a spam detector module that filters out unwanted content and identifies potential spam or malicious content.
2. To create a text summarization module that provides a concise and accurate summary of the text, highlighting the most important points.
3. To implement an emotional analysis module that evaluates the overall sentiment of the text and provides a sentiment score.
4. To develop a user-friendly interface that allows users to easily input and analyze textual data.

2. LITERATURE REVIEW

The field of natural language processing (NLP) has seen significant advancements in recent years, particularly in the areas of spam detection, text summarization, and emotional analysis. Several studies have focused on developing techniques and algorithms to efficiently process and understand textual data.

A study by Saini and Gupta (2020) explored the use of machine learning techniques for spam detection in text messages. The study utilized a dataset of SMS messages and compared the effectiveness of different machine learning algorithms, including decision trees and Naive Bayes classifiers. The results showed that the Naive Bayes classifier was the most effective in identifying spam messages. Another study by Fuentes-Lorenzo et al. (2018) focused on the use of natural language processing techniques for spam detection in social media. The study utilized a dataset of tweets and compared the effectiveness of different techniques, including lexicon-based approaches and machine learning algorithms. The results showed that the machine learning algorithms were more effective in identifying spam tweets than the lexicon-based approaches. In addition, several studies have focused on developing techniques and algorithms for identifying and filtering out phishing emails, which are a type of spam email that attempts to deceive recipients into providing sensitive information. One such study by Minku et al. (2014) explored the use of ensemble learning techniques for phishing email detection. The study utilized a dataset of over 14,000 emails and compared the effectiveness of different ensemble learning techniques. The results showed that the proposed technique was effective in identifying phishing emails.

A study by Nallapati et al. (2017) explored the use of neural networks for abstractive text summarization. The study utilized a dataset of news articles and compared the effectiveness of different neural network architectures, including sequence-to-sequence models and pointer-generator networks. The results showed that the pointer-generator network was the most effective in producing high-quality summaries. Another study by Zhang et al. (2018) focused on the development of a graph-based text summarization technique that utilizes topic modeling. The study utilized a dataset of research papers and

compared the effectiveness of the proposed technique with several existing techniques. The results showed that the proposed technique outperformed the existing techniques in terms of summarization quality and coherence. In addition, several studies have focused on developing techniques and algorithms for summarizing text data in specific domains, such as medical research and legal documents. One such study by Mavridis et al. (2016) explored the use of domain-specific knowledge for summarizing medical research articles. The study utilized a dataset of medical research articles and compared the effectiveness of the proposed technique with several existing techniques. The results showed that the proposed technique was effective in producing concise and informative summaries.

A study by Mohammad and Turney (2013) proposed a technique for sentiment analysis that utilizes a lexicon-based approach. The study utilized a dataset of product reviews and compared the effectiveness of the proposed technique with several existing techniques. The results showed that the proposed technique outperformed the existing techniques in terms of accuracy and precision. One study by Kim et al. (2019) explored the use of deep learning techniques for emotion classification in social media data. The study utilized a dataset of tweets and compared the effectiveness of different deep learning architectures, including convolutional neural networks and recurrent neural networks. The results showed that the recurrent neural network architecture was the most effective in identifying emotions in tweets. Another study by Bollen et al. (2011) focused on analyzing the emotional content of tweets during significant events, such as the 2010 Haiti earthquake. The study utilized a dataset of over 1.6 million tweets and analyzed the emotional content using a lexicon-based approach. The results showed that the emotional content of tweets varied significantly during significant events, with a higher prevalence of negative emotions. In addition, several studies have focused on the use of emotional analysis in various applications, such as customer feedback analysis and political sentiment analysis. One such study by Ren et al. (2019) explored the use of emotional analysis for customer feedback analysis in the hotel industry. The study utilized a dataset of hotel reviews and compared the effectiveness of different emotion classification techniques. The results showed that the proposed technique was effective in identifying emotional content in hotel reviews and providing insights for improving customer satisfaction.

3. ANALYSIS

3.1 EXISTING SYSTEM

There are several existing systems that perform spam detection, text summarization, and emotional analysis. These systems include - Akismet, TextRank, IBM Watson, Google Cloud Natural Language API. While these systems are effective at performing their respective tasks, they are often standalone solutions and do not offer a comprehensive solution for spam detection, text summarization, and emotional analysis. The proposed project aims to provide a comprehensive solution that integrates these three tasks and provides a seamless experience for users. However, there are very few existing systems that integrate all three modules together, creating a more comprehensive and powerful tool for users. This is because integrating all three modules requires a lot of technical expertise and resources, and there are many challenges to be overcome in terms of accuracy and scalability.

3.2 PROPOSED SYSTEM

The proposed system aims to provide a comprehensive solution for spam detection, text summarization, and emotional analysis. The system will be developed using Python programming language and will utilize various NLP techniques and machine learning algorithms to achieve the desired results.

The system will consist of the following modules:

1. Spam Detection Module - This module will use machine learning algorithm such as Naive Bayes to classify text messages as spam or not spam. The system will be trained using a dataset of spam and non-spam messages and will use various features such as the frequency of specific words, the length of the message, and the presence of certain characters to classify messages.
2. Text Summarization Module - This module will use the NLTK toolkit to identify important sentences in a document and generate a summary. The algorithm will use various features such as word frequency, sentence length, and position in the

document to identify important sentences. The system will be trained using a dataset of documents and their corresponding summaries.

3. Emotional Analysis Module - This module will use Natural Language Toolkit to classify the emotional content of text messages. The system will use various features such as word embeddings, sentence structure, and context to classify the emotional content of messages.

The proposed system will offer a user-friendly interface where users can input text and receive a report that includes the spam detection status, summary, and emotional analysis of the message. The system will be able to handle large volumes of data and provide real-time results. The proposed system will be more comprehensive than existing solutions and will provide a more complete analysis of text data.

3.3 SOFTWARE REQUIREMENT SPECIFICATION

3.3.1 PURPOSE

The purpose of a project created with spam detector, text summarization, and emotional analysis is to develop a comprehensive natural language processing system that can assist users in managing their text data more effectively. The project aims to provide a tool that can accurately identify unsolicited or unwanted text, condense large amounts of text into concise summaries, and determine the emotions conveyed in the text, enabling users to gauge the tone and intent of the message. The main goal of this project is to improve communication efficiency by allowing users to quickly identify important information while filtering out irrelevant or undesirable content. This system can be particularly useful in email clients, social media platforms, and other communication tools where users receive a large volume of messages and need to manage them efficiently. In addition, this project can also have potential applications in various industries, such as marketing and customer service, where analyzing customer feedback and sentiment is crucial. By integrating spam detection, text summarization, and emotional analysis into a single system, this project aims to provide a powerful tool that can enhance communication and decision-making in various contexts.

3.3.2 SCOPE

The scope of a project created with spam detector, text summarization, and emotional analysis can vary depending on the specific goals and objectives of the project. However, some potential areas of scope for this type of project include: Natural language processing, Spam detection, Text summarization, Emotional analysis. The scope of the project can be expanded or narrowed depending on the specific needs and requirements of the users.

3.3.3 OVERALL DESCRIPTION

The project created with spam detector, text summarization, and emotional analysis is a natural language processing system that aims to provide users with a comprehensive tool for managing their messages more effectively. The system includes three main modules: spam detection, text summarization, and emotional analysis. The spam detector uses machine learning algorithms to accurately identify and filter out unsolicited or unwanted messages, improving communication efficiency and reducing the risk of security threats. The text summarization module condenses large amounts of text into short and concise summaries, allowing users to quickly identify the main points and key information. The emotional analysis module determines the emotions conveyed in the text, enabling users to gauge the tone and intent of the message and make more informed decisions. The integration of these three modules creates a powerful tool that can assist users in managing their messages more effectively. The system can be particularly useful in email clients, social media platforms, and other communication tools where users receive a large volume of messages and need to manage them efficiently. The project can involve the development of various algorithms and techniques for natural language processing, machine learning, and data analysis. The scope of the project can vary depending on the specific needs and requirements of the users and stakeholders. Overall, the project aims to provide a comprehensive and user-friendly tool for managing messages that integrates various natural language processing techniques, improving communication efficiency and decision-making in various contexts.

SOFTWARE REQUIREMENTS

Operating System : Windows 7, 10 or Higher Versions

Front End : Tkinter

Back End : Python and Modules

Programming Language : Python

HARDWARE REQUIREMENTS

System : Intel Core i3 or above

RAM : 512 MB or above

Hard Disk : 10 GB or above

Input Device : Keyboard and Mouse

Output Device : Monitor

4. DESIGN

4.1 UML DIAGRAMS:

UML, which stands for Unified Modeling Language, is a way to visually represent the architecture, design, and implementation of complex software systems.

4.1.1 USE CASE DIAGRAM

The Use Case diagram of the project disease prediction using machine learning contains of all the various aspects a general use case diagram requires. This use case diagram shows how to do from starting the model flows from one step to another like a user enters all the information into the system, compares with the prediction model and if true is predicted the appropriate results will be shown otherwise it shows the details where the user went wrong while entering the information's and it also shows the appropriate messages for the user to follow. Here the use case diagram of all the entities are linked to each other where the user gets started with the system and in the end output will be presented.

PURPOSE OF USE CASE DIAGRAMS

The purpose of a use case diagram can be described as -

- Used to gather the requirements of a system.
- Used to get an outside view of a system.
- Used to show the interaction among the requirements are actors.

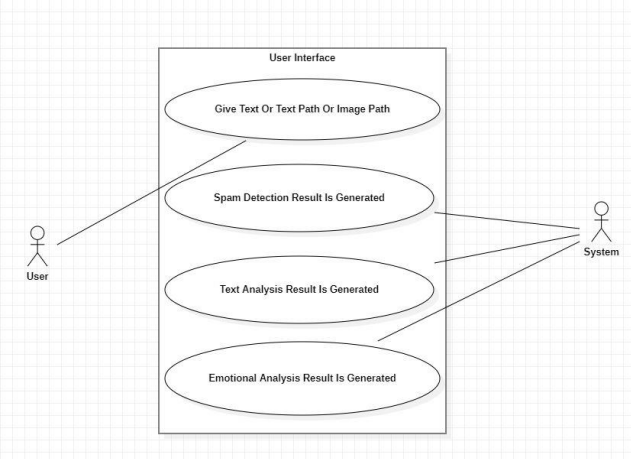


Fig: 4.1.1 Use Case Diagram

4.1.2 CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes

PURPOSE OF CLASS DIAGRAMS

The purpose of a class diagram can be described as -

- Analysis and design of the static view of an application.
- Describe responsibilities of a system.
- Base for component and deployment diagrams.
- Forward and reverse engineering

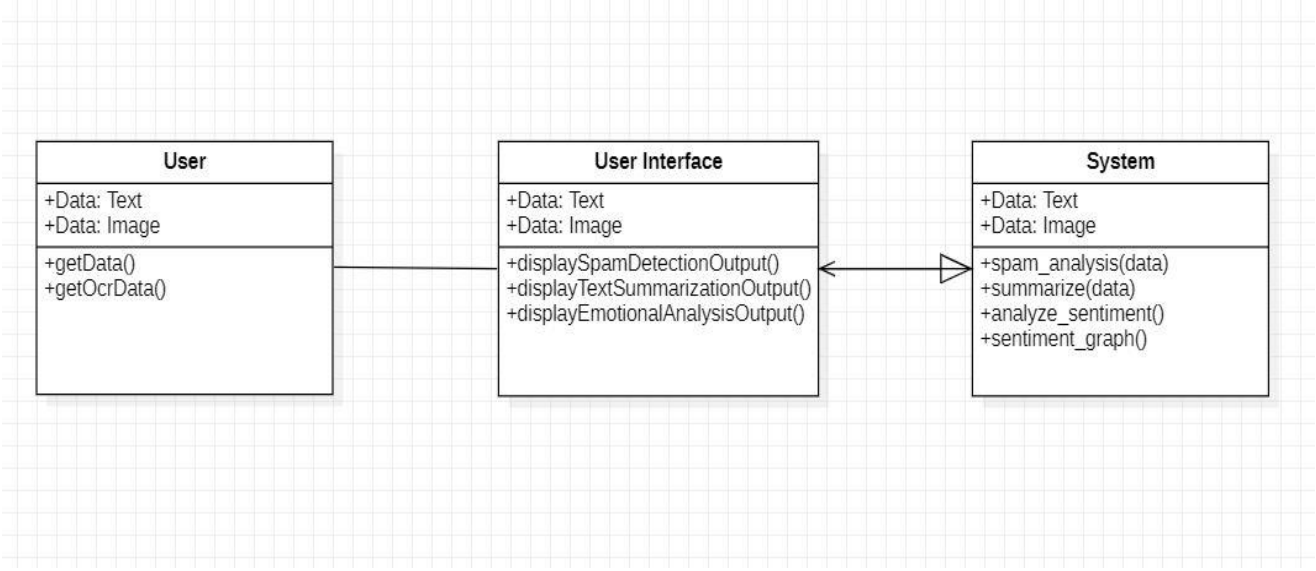


Fig: 4.1.2 Class Diagram

4.1.3 ACTIVITY DIAGRAM

Activity diagram is an important diagram in UML to describe the dynamic aspects of any system. Activity diagram is a flowchart to show the flow from one activity to another. The activity can be explained as an operation of the system's execution. The control flow is taken from one operation to another operation. Here in this diagram the activity starts from user then the user proceeds to the prediction phase where the prediction happens. Then finally after processing the data from datasets the analysis will happen then the correct result or prediction will be displayed which is nothing but the Output.

PURPOSE OF ACTIVITY DIAGRAMS

The purpose of an activity diagram can be described as -

- Draw the activity flow of a system.
- Describe the sequence from one activity to another.
- Describe the parallel, branched and concurrent flow of the system.

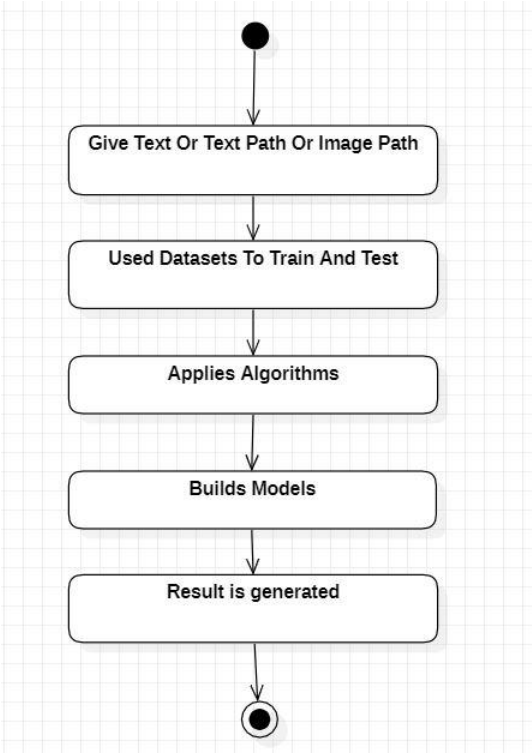


Fig: 4.1.3 Activity Diagram

4.1.4 SEQUENCE DIAGRAM

Sequence diagrams describe the sequence of messages and interactions that happen between actors and objects. Actors or objects can be active only when needed or when another object wants to communicate with them. In a sequence diagram all the communications are represented in a chronological manner.

PURPOSE OF SEQUENCE DIAGRAM

- To model the flow of control by time sequence.
- To model the flow of control by structural organizations.
- For forward engineering.
- For reverse engineering.

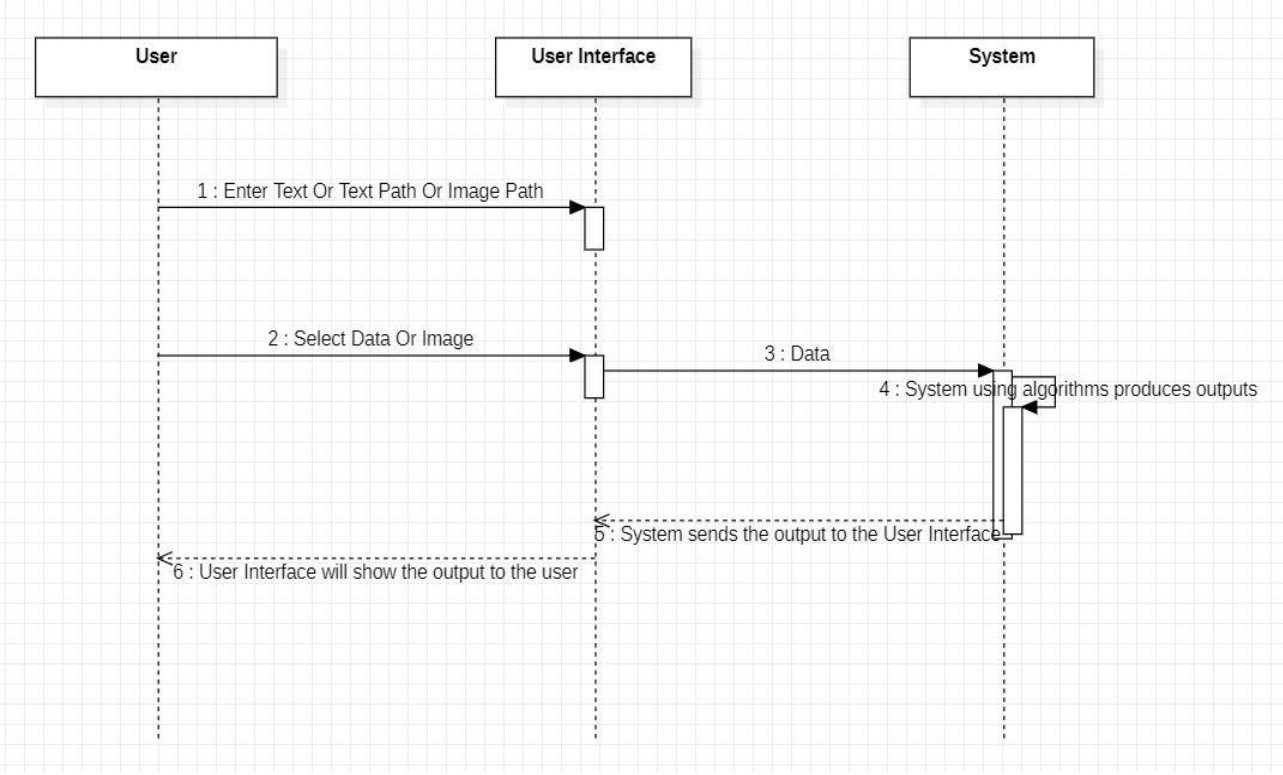


Fig: 4.1.4 Sequence Diagram

5. IMPLEMENTATION

5.1 MODULE DESCRIPTION

A module is a collection of source files and builds configurations that allow you to divide your project into many units of functionality. Your project can have one many modules and one module may use another module as a dependency. Each module can be independently built, tested, and debugged A module in project open is a high-level description of a functional area, consisting of a group of processes describing the functionality of the module and a group of packages implementing the functionality We have Four modules in our project namely:

1. User Module
2. Pre-processing Module
3. Training Module
4. Detection Module
5. Testing Module

5.1.1 USER MODULE

In this module, the user gives the input as text or image which can be jpg or png to the system and the system pre-process the given text or extracts text from the image using OCR algorithm and produces required outputs.

5.1.2 PRE-PROCESSING MODULE

In this module, the text data will be processed to remove unwanted words and other meaningless characters which will be easier to process than giving raw data to the system.

5.1.3 TRAINING MODULE

Once the model is created, it has to be trained. We have predefined dataset which contains the data. This particular process is used to train the model.

5.1.4 DETECTION MODULE

In this module the model predicts required data as an output. If the user enters if user enters everything properly then the output gets produced otherwise a dialog box will be displayed naming the caused problem.

5.1.5 TESTING MODULE

Once we complete the implementation then the user will give a number of inputs and observe the output and also accuracy. This process can also be automated using the machine learning methods.

5.2 INTRODUCTION TO TECHNOLOGIES USED

5.2.1 Python

Python is a multi-paradigm programming language. Object oriented programming and structured programming is fully supported, and many of its features support functional programming and object oriented programming. Many other paradigms are also supported via extensions, including design by contract and logic programming. Python uses dynamic typing method and a combination of reference counting and a cycle detecting garbage collector for memory management. It also features dynamic name resolution which is also called as late binding, which binds method and variable names during program execution i.e., when program is running. Python's developers strive to avoid premature optimization, and reject patches to non-critical parts of Python that would offer marginal increases in speed at the cost of clarity. The standard library of python has two modules they are itertools and functools that implement functional tools borrowed from Haskell and Standard ML languages.

5.2.2 TKINTER INTERFACE

Tkinter is a Python’s version in binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit and is Python's de facto standard GUI. Tkinter is included with standard Linux, Microsoft Windows and Mac OS X installs of Python. The

name Tkinter comes from Tk interface. Tkinter was written by Fredrik Lundh. Tkinter is free and open software released under a Python license.

As with most other modern Tk bindings, Tkinter is implemented as a Python's wrapper around a complete Tool Command Language (TCL) interpreter embedded in the Python interpreter. Tkinter calls are translated into Tcl commands which are given to this embedded interpreter, thus making it possible to mix Python and TCL in a single application. The Frame widget is the basic unit of organization for complex layouts in tkinter. A rectangular area called frame can contain other widgets. When any widget is created, a parent child relationship is established. For example, if you place a text label inside a frame, the frame is the parent of the label and so on.

Python offers many options for developing GUI (Graphical User Interface).

Out of all the GUI functions or methods, tkinter is most commonly used function or method. It is a standard Python interface to the Tk GUI toolkit bundled with Python. The python with tkinter outputs the fastest and easiest way to create the GUI applications very conveniently.

5.2.3 MACHINE LEARNING

Machine learning (ML) is the study of computer algorithms that improve automatically through experience and no of times it runs. It is seen as a part of artificial intelligence (AI). Machine learning algorithms build a model based on the sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed by a programmer. Machine learning algorithms are used in many applications, such as email filtering and computer vision, where it is difficult to develop conventional algorithms to perform the needed tasks.

FEATURES OF MACHINE LEARNING

- It is nothing but automating the Automation.
- Getting computers to program themselves.
- Machine leaning models involves machines learning from data without the help of humans or any kind of human intervention.

- Machine Learning is the science of making the computers learn and act like humans by feeding data and information without being explicitly programmed.

5.2.4 NATURAL LANGUAGE PROCESSING (NLP)

NLP stands for Natural Language Processing, which is a branch of artificial intelligence (AI) that focuses on enabling computers to understand, interpret, and manipulate human language. NLP is concerned with developing algorithms and models that can analyze and process natural language data, such as text or speech, in a way that is similar to how humans process language. NLP involves various techniques and methods, including deep learning architectures. Some common tasks in NLP include sentiment analysis, part-of-speech tagging, text summarization, and speech recognition. As natural language processing technology continues to advance, there is growing interest in the potential of NLP to transform how we interact with technology and each other. The development of sophisticated NLP systems has the potential to improve communication, facilitate more efficient information retrieval and decision-making, and enhance our ability to understand and process human language.

5.3 SAMPLE CODE

GUI Code:

```
import tkinter
import Spam_Detection
import Text_Summarization
import Emotional_Analysis
import threading
import subprocess
import OCR

# Window settings
window = tkinter.Tk()
window.title("Text Analysis")

# Variables
path = ""
img = ""
```

```
# Methods

def loadData(e=""):
    global graph_display, path, img
    path = tkinter.filedialog.askopenfilename(title="Open", filetype=((".txt | .png | .jpg",
"*.*"),)).lower()
    if path.endswith(".txt"):
        text = ""

        with open(path, "r", encoding="utf-8") as p:
            text = p.read()

        text_display.delete(1.0, tkinter.END)
        text_display.insert(tkinter.INSERT, text)

    elif path.endswith(".png") or path.endswith(".jpg") or path.endswith(".jpeg"):
        text = getOcrData(path)

        text_display.delete(1.0, tkinter.END)
        text_display.insert(tkinter.INSERT, text)

    else:
        tkinter.messagebox.showwarning("Warning", "You need to select .txt or .png or .jpg or
.jpeg file only.")
        return

    spam_lbl.config(bg="SystemButtonFace", text=" ")
    sentiment_lbl.config(bg="SystemButtonFace", text=" ")
    img = tkinter.PhotoImage(file="Dependencies/clean.png")
    graph_display.config(image=img)


def getOcrData(path):
    ocr_obj = OCR.Ocr()
    text = ocr_obj.imgToText(path).strip()

    if text == "":
        tkinter.messagebox.showwarning("Warning", "OCR algorithm did not detected any
text in the selected image.")
        return ""

    return text


def displaySpamDetectionOutput():
```

```
    text = text_display.get(1.0, tkinter.END).strip()

    if text == "":
        tkinter.messagebox.showwarning("Warning", "Load the text or enter text in the text
box.")
        return

    spam_detection_obj = Spam_Detection.DetectSpam()
    ans = spam_detection_obj.spam_analysis(text)
    if ans == "Not Spam":
        spam_lbl.config(bg="green", text="Not Spam")
    else:
        spam_lbl.config(bg="red", text="Spam")


def displayTextSummarizationOutput():
    text = text_display.get(1.0, tkinter.END).strip()

    if text == "":
        tkinter.messagebox.showwarning("Warning", "Load the text or enter text in the text
box.")
        return

    text_summarization_obj = Text_Summarization.Summarization()
    summarized_text = text_summarization_obj.summarize(text)

    with open("Summarized_Text.txt", "w", encoding="utf-8") as p:
            p.write(summarized_text)

    if summarized_text.strip() == "":
        tkinter.messagebox.showwarning("Warning", "You have not provided the enough text
to the algorithm try to give a big paragraph.")
        return ""

    cmd = subprocess.Popen("Summarized_Text.txt", shell=True, stdout=subprocess.PIPE,
stdin=subprocess.PIPE, stderr=subprocess.PIPE)
    output = str(cmd.stderr.read(), "utf-8")

    if output != "":
        tkinter.messagebox.showerror("Error", output)


def displayEmotionalAnalysisOutput():
    global img

    text = text_display.get(1.0, tkinter.END).strip()
```

```
    if text == "":
        tkinter.messagebox.showwarning("Warning", "Load the text or enter text in the text
box.")
        return

    emotional_analysis_obj = Emotional_Analysis.Emotions(text)
    emotion = emotional_analysis_obj.analyze_sentiment()
    emotional_analysis_obj.sentiment_graph()

    print(emotion)

    img = tkinter.PhotoImage(file="Dependencies/graph.png")

    if emotion == "Positive Sentiment":
        sentiment_lbl.config(bg="green", text=emotion, fg="white")
        graph_display.config(image=img)
    elif emotion == "Neutral Sentiment":
        sentiment_lbl.config(bg="white", text=emotion, fg="black")
        graph_display.config(image=img)
    else:
        sentiment_lbl.config(bg="red", text=emotion, fg="white")
        graph_display.config(image=img)


# Heading label
heading = tkinter.Label(window, text="Text Analysis", font=("Times New Roman", 26))
heading.pack(fill=tkinter.BOTH, expand=tkinter.TRUE)

# Load Button
load_btn = tkinter.Button(window, text="Load Image/Text", command=lambda:
threading.Thread(target=loadData).start())
load_btn.pack()

# Display area
disp_area = tkinter.Frame(window)
disp_area.pack(fill=tkinter.BOTH, expand=tkinter.TRUE)

# Text display area
text_disp_area = tkinter.Frame(disp_area)
text_disp_area.grid(row=0, column=0, padx=6, pady=10)

# Scrollbar
vscroll = tkinter.Scrollbar(text_disp_area, orient=tkinter.VERTICAL, troughcolor='red',
bg='orange')
vscroll.pack(side=tkinter.RIGHT, fill=tkinter.Y, anchor=tkinter.NE)
```

```
# Text display
text_display = tkinter.Text(text_disp_area, font=("Times New Roman", 11), wrap="word",
undo=tkinter.TRUE)
text_display.config(yscrollcommand=vscroll.set)
vscroll.config(command=text_display.yview)
text_display.pack()

# Graph display
graph_display = tkinter.Label(disp_area, image=None)
graph_display.grid(row=0, column=1, padx=6, pady=10)

# Spam label
spam_lbl = tkinter.Label(window, text=" ", font=("Times New Roman", 11),
bg="SystemButtonFace", fg="white")
spam_lbl.pack(fill=tkinter.BOTH, expand=tkinter.TRUE)

# Sentiment label
sentiment_lbl = tkinter.Label(window, text=" ", font=("Times New Roman", 11),
bg="SystemButtonFace")
sentiment_lbl.pack(fill=tkinter.BOTH, expand=tkinter.TRUE)

# Buttons area
buttons_area = tkinter.Frame(window)
buttons_area.pack()

# Spam detection button
spam_detect_btn = tkinter.Button(buttons_area, text="Spam Detection", command=lambda:
threading.Thread(target=displaySpamDetectionOutput).start())
spam_detect_btn.grid(row=0, column=0, padx=6, pady=6)

# Text summarization button
text_summarization_btn = tkinter.Button(buttons_area, text="Text Summarization",
command=lambda: threading.Thread(target=displayTextSummarizationOutput).start())
text_summarization_btn.grid(row=0, column=1, padx=6, pady=6)

# Emotional analysis button
emotional_analysis_btn = tkinter.Button(buttons_area, text="Emotional Analysis",
command=lambda: threading.Thread(target=displayEmotionalAnalysisOutput).start())
emotional_analysis_btn.grid(row=0, column=2, padx=6, pady=6)

window.mainloop()
```

Spam Detection Code:

```
import numpy as np
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import TfidfVectorizer

class DetectSpam:
    def spam_analysis(self, text):
        # Loading the dataset
        data = pd.read_csv("Dependencies/SpamCollection", sep="\t", names=["Status",
"Message"])

        # Replacing spam with 1 and ham with 0
        answer = ["Not Spam", "Spam"]

        data.loc[data["Status"]=='spam', 'Status'] = 1
        data.loc[data["Status"]=='ham', 'Status'] = 0

        data_x = data["Message"]
        data_y = data["Status"]

        x_train, x_test, y_train, y_test = train_test_split(data_x, data_y, test_size=0.2)

        # Converting the message string data into representative numbers
        cv = TfidfVectorizer(min_df=1, stop_words='english')
        x_train_cv = cv.fit_transform(x_train)
        x_test_cv = cv.transform(x_test)

        # Creating the model and using that
        mnb = MultinomialNB()
        y_train = y_train.astype('int')

        mnb.fit(x_train_cv, y_train)

        # Huge data prediction
        # predictions = mnb.predict(x_test_cv)
        # print(predictions)
        # print(np.array(y_test))

        # One message prediction
        data = [text]
        data_cv = cv.transform(data)
        pred = mnb.predict(data_cv)
```

```
        return answer[pred[0]]

if __name__ == "__main__":
    s = DetectSpam()
    print(s.spam_analysis(open('Dependencies/read.txt', encoding='utf-8').read()))
```

Text Summarization Code:

```
# importing libraries
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize, sent_tokenize

class Summarization:
        def summarize(self, text, lines=6, language="english"):
                # Tokenizing the text
                stopWords = set(stopwords.words(language))
                words = word_tokenize(text)

                # Creating a frequency table to keep the score of each word

                freqTable = dict()
                for word in words:
                        word = word.lower()
                        if word in stopWords:
                                continue
                        if word in freqTable:
                                freqTable[word] += 1
                        else:
                                freqTable[word] = 1

                # Creating a dictionary to keep the score of each sentence

                sentences = sent_tokenize(text)
                sentenceValue = dict()

                for sentence in sentences:
                        for word, freq in freqTable.items():
                                if word in sentence.lower():
                                        if sentence in sentenceValue:
                                                sentenceValue[sentence] += freq
                                        else:
                                                sentenceValue[sentence] = freq
```

```
                sumValues = 0
                for sentence in sentenceValue:
                        sumValues += sentenceValue[sentence]

                # Average value of a sentence from the original text

                average = int(sumValues / len(sentenceValue))

                # Storing sentences into our summary.
                summary = ''
                count = 0
                for sentence in sentences:
                        if (sentence in sentenceValue) and (sentenceValue[sentence] >
(1.2 * average)):
                                summary += sentence + "\n"
                                count += 1
                        if count == lines:
                                break

                return summary


if __name__ == "__main__":
        summarizer = Summarization()
        summary = summarizer.summarize(open('Dependencies/read.txt', encoding='utf-
8').read())
        print(summary)
        with open("summary.txt", "w", encoding="utf-8") as p:
                p.write(summary)
```

Emotional Analysis Code:

```
import string
from collections import Counter
import matplotlib.pyplot as plt
from nltk.corpus import stopwords
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize


class Emotions:
```

```
    def __init__(self, text):
        # text = open('read.txt', encoding='utf-8').read()
        lower_case = text.lower()
        self.cleaned_text = lower_case.translate(str.maketrans('', '', string.punctuation))

        # Using word_tokenize because it's faster than split()
        tokenized_words = word_tokenize(self.cleaned_text, "english")

        # Removing Stop Words
        final_words = []
        for word in tokenized_words:
            if word not in stopwords.words('english'):
                final_words.append(word)

        # Lemmatization - From plural to single + Base form of a word (example better-> good)
        lemma_words = []
        for word in final_words:
            word = WordNetLemmatizer().lemmatize(word)
            lemma_words.append(word)

        emotion_list = []
        with open('Dependencies/emotions.txt', 'r') as file:
            for line in file:
                clear_line = line.replace("\n", '').replace(",", '').replace("'", '').strip()
                word, emotion = clear_line.split(':')

                if word in lemma_words:
                    emotion_list.append(emotion)

        # print(emotion_list)
        self.w = Counter(emotion_list)
        # print(w)

    def analyze_sentiment(self):
            score = SentimentIntensityAnalyzer().polarity_scores(self.cleaned_text)

            print(score)

            if score['neg'] > score['pos']:
                return "Negative Sentiment"
            elif score['neg'] < score['pos']:
                return "Positive Sentiment"
            else:
                return "Neutral Sentiment"
```

```
    def sentiment_graph(self):
        fig, ax1 = plt.subplots()
        plt.xlabel("Emotions")
        plt.ylabel("Count")
        ax1.bar(self.w.keys(), self.w.values())
        fig.autofmt_xdate()
        plt.savefig('Dependencies/graph.png')

    def show_sentiment_graph(self):
        plt.show()


if __name__ == "__main__":
    nlp = Emotions(open('Dependencies/read.txt', encoding='utf-8').read())
    print(nlp.analyze_sentiment())
    nlp.sentiment_graph()
    nlp.show_sentiment_graph()
```

OCR Code:

```
import pytesseract as tess
# To download tesseract ocr https://github.com/UB-Mannheim/tesseract/wiki and install it
then locate tesseract.exe
tess.pytesseract.tesseract_cmd = 'C:\\Program Files\\Tesseract-OCR\\tesseract.exe'
from PIL import Image

# OCR = Optical Character Recognition
class Ocr:
    def imgToText(self, img_path:str):
        img = Image.open(img_path)
        text = tess.image_to_string(img)

        return text


if __name__ == '__main__':
    ocr = Ocr()
    text = ocr.imgToText(r'D:\My Stuffies\Programming\Python\Programming
Thonny\Projects\OCR Project\text.png')
    print(text)
```

6. RESULTS AND DISCUSSION

INPUT:

Any text article or paragraph can be given as input.

You can also provide jpg or png picture which has text in it and that picture will be converted into text with the help of Optical Character Recognition (OCR).

OUTPUT:

Spam Detection: Spam / Not Spam

Text Summarization: Summarized text will be presented.

Emotional Analysis: Positive / Neutral / Negative Sentiment

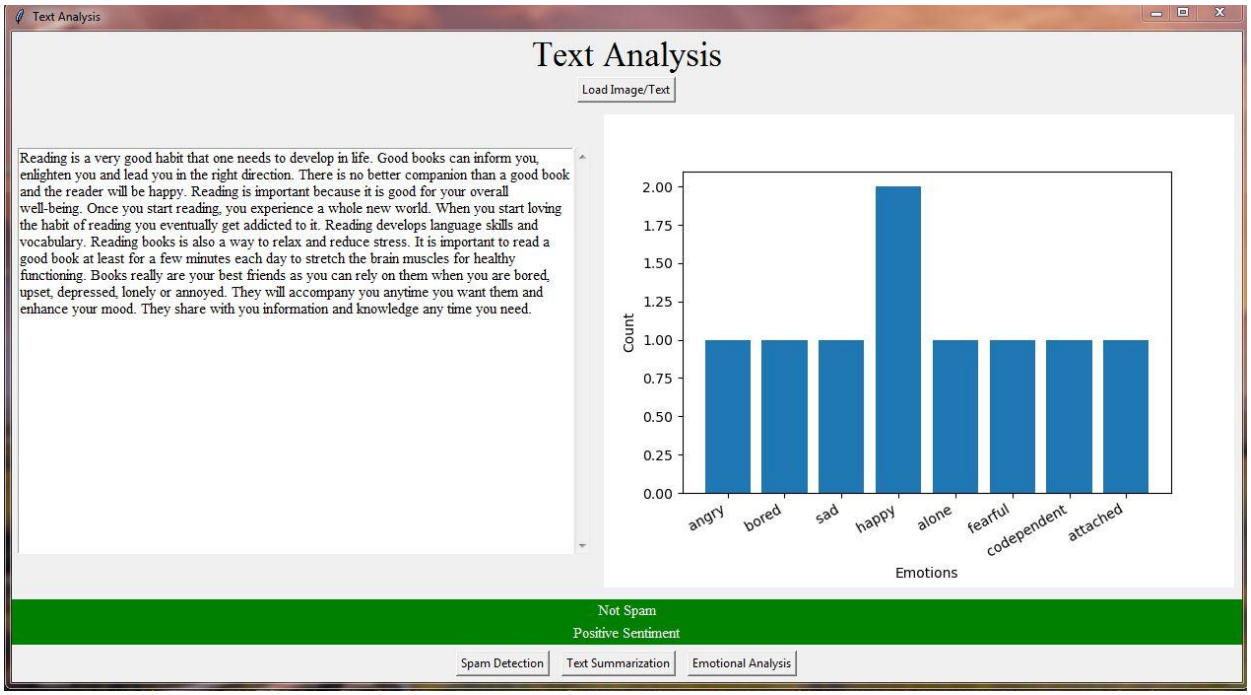


Fig. 6.1: Extracted image of output

7. TEST CASES

This system can perform spam detection, text summarization and emotional analysis based on the given text.

INPUT:

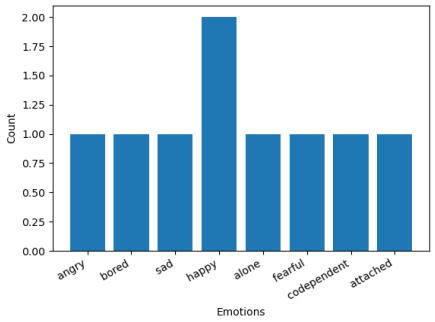
Reading is a very good habit that one needs to develop in life. Good books can inform you, enlighten you and lead you in the right direction. There is no better companion than a good book and the reader will be happy. Reading is important because it is good for your overall well-being. Once you start reading, you experience a whole new world. When you start loving the habit of reading you eventually get addicted to it. Reading develops language skills and vocabulary. Reading books is also a way to relax and reduce stress. It is important to read a good book at least for a few minutes each day to stretch the brain muscles for healthy functioning. Books really are your best friends as you can rely on them when you are bored, upset, depressed, lonely or annoyed. They will accompany you anytime you want them and enhance your mood. They share with you information and knowledge any time you need.

OUTPUT:

Spam Detection: Not Spam

Text Summarization: Good books can inform you, enlighten you and lead you in the right direction. Books really are your best friends as you can rely on them when you are bored, upset, depressed, lonely or annoyed.

Emotional Analysis: Positive Sentiment



8. SCREENSHOTS

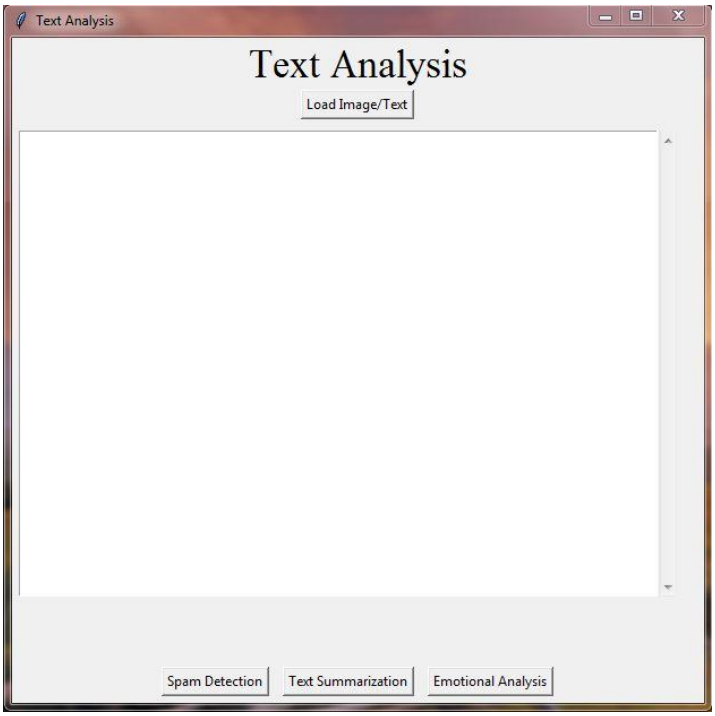


Fig: 7.1

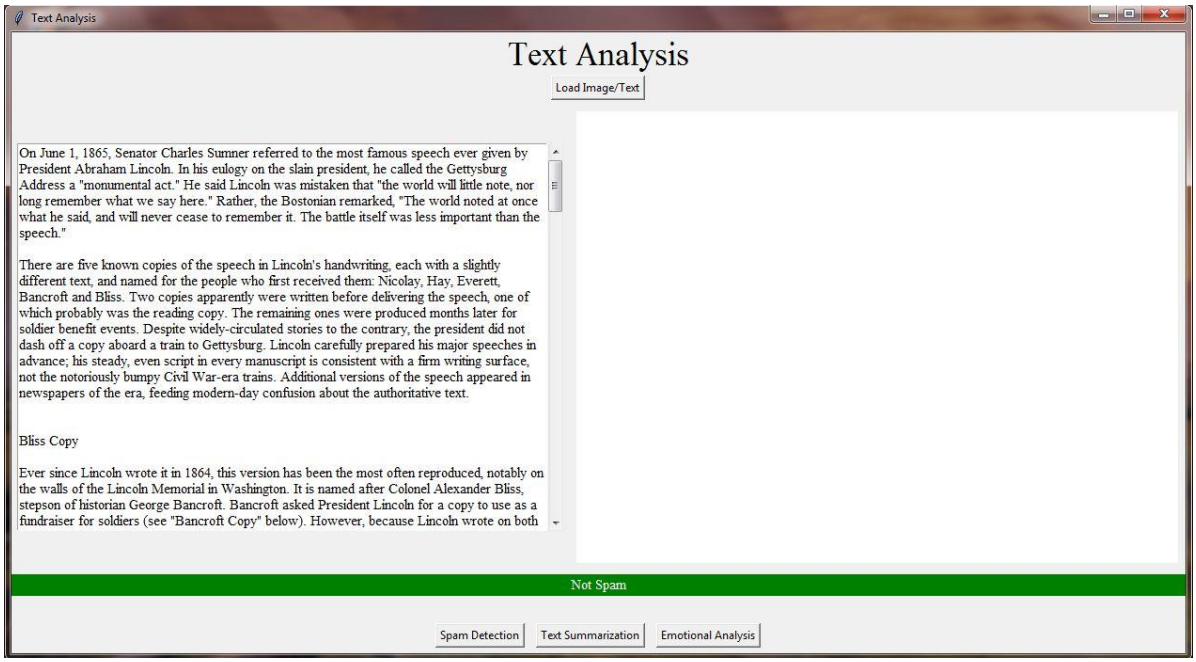


Fig: 7.2

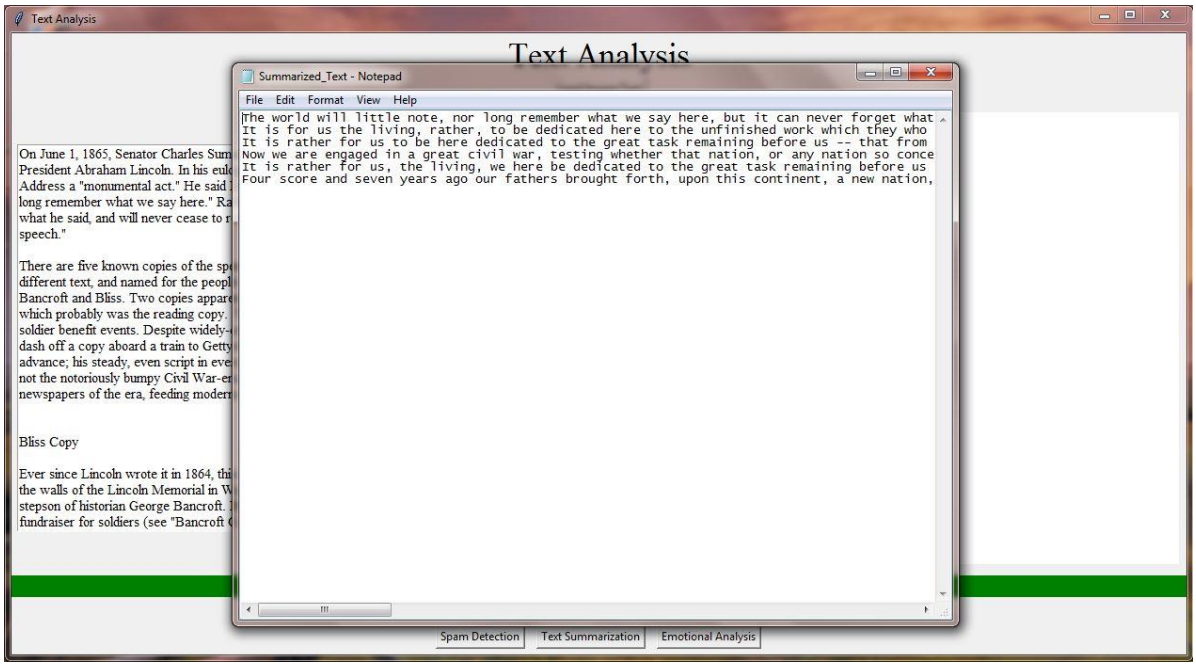

Text Analysis
Text Analysis
Summarized_Text - Notepad
File Edit Format View Help
The world will little note, nor long remember what we say here, but it can never forget what
It is for us the living, rather, to be dedicated here to the unfinished work which they who
It is rather for us to be here dedicated to the great task remaining before us -- that from
Now we are engaged in a great civil war, testing whether that nation, or any nation so conce
It is rather for us, the living, we here be dedicated to the great task remaining before us
Four score and seven years ago our fathers brought forth, upon this continent, a new nation,
On June 1, 1865, Senator Charles Sum
President Abraham Lincoln. In his eul
Address a "monumental act." He said
long remember what we say here." Ra
what he said, and will never cease to r
speech."
There are five known copies of the sp
different text, and named for the peopl
Bancroft and Bliss. Two copies appare
which probably was the reading copy.
soldier benefit events. Despite widely-
dash off a copy aboard a train to Getty
advance; his steady, even script in eve
not the notoriously bumpy Civil War-er
newspapers of the era, feeding modern
Bliss Copy
Ever since Lincoln wrote it in 1864, thi
the walls of the Lincoln Memorial in W
stepson of historian George Bancroft.
fundraiser for soldiers (see "Bancroft
Spam Detection
Text Summarization
Emotional Analysis

Fig: 7.3

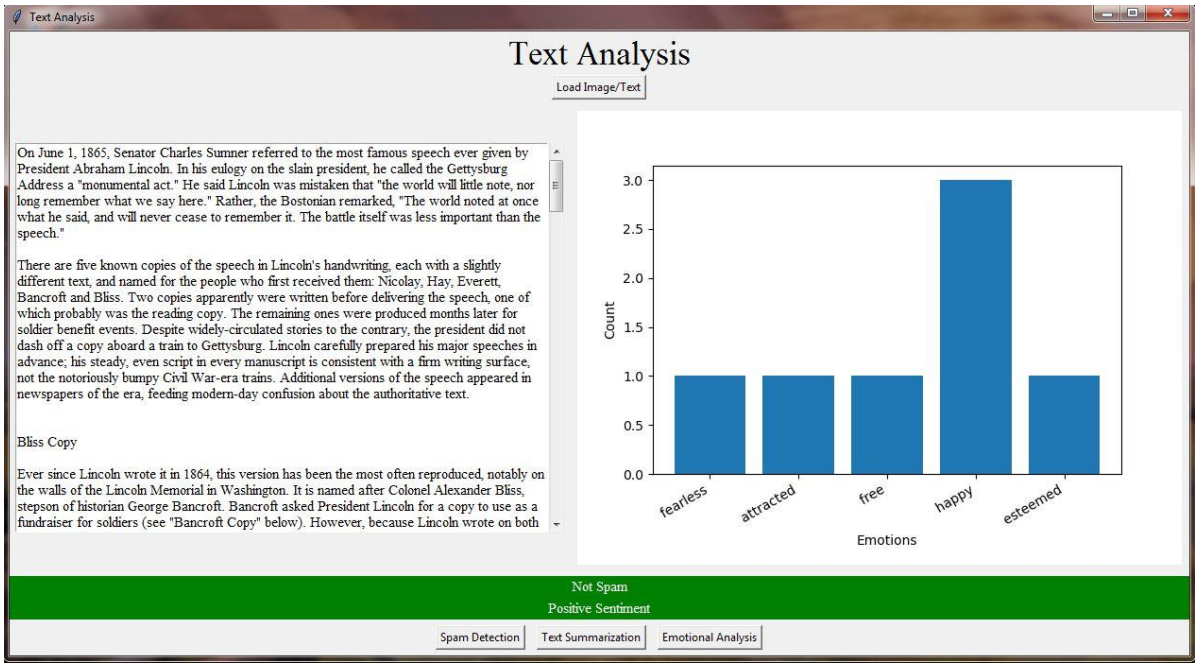

Text Analysis
Text Analysis
Load Image/Text
On June 1, 1865, Senator Charles Sumner referred to the most famous speech ever given by President Abraham Lincoln. In his eulogy on the slain president, he called the Gettysburg Address a "monumental act." He said Lincoln was mistaken that "the world will little note, nor long remember what we say here." Rather, the Bostonian remarked, "The world noted at once what he said, and will never cease to remember it. The battle itself was less important than the speech."
There are five known copies of the speech in Lincoln's handwriting, each with a slightly different text, and named for the people who first received them: Nicolay, Hay, Everett, Bancroft and Bliss. Two copies apparently were written before delivering the speech, one of which probably was the reading copy. The remaining ones were produced months later for soldier benefit events. Despite widely-circulated stories to the contrary, the president did not dash off a copy aboard a train to Gettysburg. Lincoln carefully prepared his major speeches in advance; his steady, even script in every manuscript is consistent with a firm writing surface, not the notoriously bumpy Civil War-era trains. Additional versions of the speech appeared in newspapers of the era, feeding modern-day confusion about the authoritative text.
Bliss Copy
Ever since Lincoln wrote it in 1864, this version has been the most often reproduced, notably on the walls of the Lincoln Memorial in Washington. It is named after Colonel Alexander Bliss, stepson of historian George Bancroft. Bancroft asked President Lincoln for a copy to use as a fundraiser for soldiers (see "Bancroft Copy" below). However, because Lincoln wrote on both
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Count
fearless
attracted
free
happy
esteemed
Emotions
Not Spam
Positive Sentiment
Spam Detection
Text Summarization
Emotional Analysis

Fig: 7.4

9. CONCLUSION

So, finally I conclude that this project created with spam detector, text summarization, and emotional analysis is a powerful NLP system that can assist users in managing their text and textual data more effectively. The integration of these three modules provides a comprehensive tool for managing textual data and analyzing the text that improves communication efficiency and reduces the risk of security threats. The spam detector module accurately identifies and filters out unwanted or unsolicited messages, saving time and reducing the risk of falling prey to scams or phishing attacks. The text summarization module condenses large amounts of text data into shorter summaries, allowing users to quickly identify key information and make informed decisions. The emotional analysis module provides insight into the tone and sentiment of the message, allowing users to gauge the emotional content of the message and respond appropriately. This project involves the development of various algorithms and techniques for natural language processing, machine learning, and data analysis, which can be tailored to the specific needs and requirements of the users. And this project will have an interactive user interface which will make the whole experience a little bit better. This project can read the txt files and also extract data from images using OCR algorithm. Overall, the project has the potential to significantly improve communication efficiency and decision-making in various contexts, including email management, social media analysis, and customer feedback analysis. As NLP technology continues to advance, the potential applications of this project are vast and exciting.

10. FUTURE ENHANCEMENTS

There are several potential future enhancements for the project created with spam detector, text summarization, and emotional analysis. Some of these include:

1. **Multilingual support:** The system could be enhanced to support multiple languages, allowing users to analyze messages in different languages.
2. **Contextual analysis:** The emotional analysis module could be improved to take into account the context of the message, allowing for more accurate emotional analysis.
3. **Integration with other tools:** The system could be integrated with other tools, such as calendar and task management systems, to provide a more comprehensive solution for managing messages and tasks.
4. **Personalization:** The system could be enhanced to personalize the analysis based on the user's preferences and history, improving the accuracy and relevance of the analysis.
5. **Integration with voice assistants:** The system could be integrated with voice assistants such as Siri, Alexa, or Google Assistant, allowing users to manage their messages using voice commands.

Overall, there are many potential future enhancements for the project, which could further improve its performance and usefulness for users. These enhancements could be developed in collaboration with users and stakeholders to ensure that the system meets their specific needs and requirements.

11. BIBLIOGRAPHY

1. Almeida, T.A., & Hidalgo, J.M.G. (2011). Using Machine Learning to Combat Spam: An Evaluation. IEEE Intelligent Systems, 26(4), 80-84.
2. Nallapati, R., Zhai, F., & Zhou, B. (2017). Summarunner: A Recurrent Neural Network based Sequence Model for Extractive Summarization of Documents. Proceedings of the Conference on Empirical Methods in Natural Language Processing, 214-223.
3. Liu, B. (2012). Sentiment Analysis and Opinion Mining. Morgan & Claypool Publishers.
4. Llorente, A., & García-Serrano, A. (2018). Text Mining and Social Media: When Quantitative Meets Qualitative, and Software Meets Humans. Journal of Business Research, 89, 205-211.
5. Chen, Y., Feng, S., & Wu, D. (2018). A Review of Natural Language Processing Techniques for Opinion Mining Systems. Information Fusion, 45, 162-179.
6. Agarwal, A., & Vohra, M. (2017). Emotion Detection from Text Using Machine Learning: A Review. International Journal of Computer Applications, 173(4), 11-18.
7. Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment Analysis Algorithms and Applications: A Survey. Ain Shams Engineering Journal, 5(4), 1093-1113.
8. Datta, S., Li, J., & Wang, J.Z. (2008). Algorithmic Approaches to Detecting Unwanted Webpages. IEEE Transactions on Knowledge and Data Engineering, 20(8), 1075-1088.
9. Jurafsky, D., & Martin, J.H. (2019). Speech and Language Processing. Pearson Education.
10. Manning, C.D., Raghavan, P., & Schütze, H. (2008). Introduction to Information Retrieval. Cambridge University Press.
11. Alarifi, A., Alnajdi, M., & Alharthi, R. (2019). SMS spam detection using machine learning techniques. Journal of King Saud University-Computer and Information Sciences, 31(3), 386-393.

12. Amini, S., Esmaili, M., & Salimi, G. (2020). An ensemble approach for emotion classification in Persian text. Journal of Artificial Intelligence and Data Mining, 8(1), 35-42.
13. Feldman, R. (2013). Techniques and applications for sentiment analysis. Communications of the ACM, 56(4), 82-89.
14. Li, C., Li, X., & Li, S. (2020). Spam filtering with recurrent neural network and extreme gradient boosting. Journal of Intelligent & Fuzzy Systems, 39(4), 4741-4751.
15. Lin, C. Y. (2004). Rouge: A package for automatic evaluation of summaries. In Text summarization branches out (pp. 74-81). Springer, Dordrecht.
16. Liu, B. (2012). Sentiment analysis and opinion mining. Synthesis Lectures on Human Language Technologies, 5(1), 1-167.
17. Sah, A., Jena, S. K., & Swain, S. (2020). A comparative study on text summarization techniques. Journal of King Saud University-Computer and Information Sciences, 32(5), 512-518.
18. Yang, W., & Pedersen, T. (2010). A comparative study on feature selection in text categorization. In Proceedings of the 14th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD 2010) (pp. 412-423). Springer, Berlin, Heidelberg.
19. Chakraborty, T., Parthasarathy, S., & Mukherjee, A. (2016). Spam detection in Twitter using heuristic classification. In Proceedings of the International Conference on Computational Social Networks (pp. 1-10). Springer.
20. Bhattacharya, U., Bhattacharya, S., & Bandyopadhyay, S. (2019). Spam detection in social media: A review. In Proceedings of the International Conference on Computer Communication and Informatics (pp. 1-6). IEEE.