SENTIMENTAL ANALYSIS ON REAL TIME DATA

A MAJOR PROJECT REPORT

Submitted by

K. Srikanth

(20841A6642)

B. Prashanth

(20841A6639)

S. Sainath Gupta

(20841A6630)

Varun Ganji

(20841A6649)

Guided by:

Mr. Ch Krishna Rao

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING (AI&ML)

AURORA'S TECHNOLOGICAL & RESEARCH INSTITUTE

Approved by AICTE and Affiliated to JNTUH

Accredited by NAAC with 'A' Grade**.**

Parvathapur, Uppal, Medipally(M), Medchal(D), Hyd-500039

MAY 2024

AURORA’S TECHNOLOGICAL & RESEARCH INSTITUTE

Approved by AICTE and Affiliated to JNTUH

Accredited by NAAC with ‘A’ Grade.

Parvathapur, Uppal, Medipally(M), Medchal(D), Hyd-500039

DECLARATION

We hereby declare that the work described in this project, entitled **SENTIMENTAL ANALYSIS ON REAL TIME DATA** which is being submitted by us in partial fulfilment for the award of Bachelor of Technology in Computer Science and Engineering to **AURORA’S TECHNOLOGICAL AND RESEARCH INSTITUTE** is the result of investigation carried out by us under the guidance of **Mr CH Krishna Rao ,Associate Professor, CSE.**

The work is original and has not been submitted for any degree this orany other university.

Place: Hyderabad

Date:

K. Srikanth(20841A6642)

B. Prashanth(20841A6639)

S. Sainath Gupta(20841A6630)

Varun Ganji(20841A6649)

ACKNOWLEDGMENT

This work was done during the project period, and it was a very good opportunity to put theoretical knowledge into planned exercise with an aim to solve a real time problem and to develop confidence to face various practical situations.

We would like to express my sincere thanks to **Dr. A. Mahesh Babu,** Principal,Aurora's Technological and Research Institute for providing us with a congenial atmosphere and encouragement.

We would like to express my immense gratitude to **Mrs. A. Durga Pavani**, Head of the Department CSE, for her support, valuable suggestions, and kind attentionto us throughout the project.

We express our sincere thanks to Project Coordinator **Krishna Rao**, for helping us to complete our project work by giving valuable suggestions.

We convey thanks to our project guide **Dr. Krishna Rao**, Department of Computer Science and Engineering, for providing encouragement, constant support, and guidance, which was of great help to complete this project successfully.

Finally, we would also like to thank the people who have directly or indirectly helped me, my parents, and my friends for their cooperation in completing the TechnicalSeminar work

K. Srikanth(20841A6642)

B. Prashanth(20841A6639)

S. Sainath Gupta(20841A6630)

Ganji Varun(20841A6649)

ABSTRACT

The script first imports the necessary libraries: cv2 for computer vision tasks, speech_recognition for speech recognition, and TextBlob for sentiment analysis. It then initializes the speech recognizer and the video capture device.

Inside the main loop, the script captures video frames and displays them in a window. The script then listens for audio input using the speech recognizer and the microphone. When audio is detected, the script attempts to recognize the speech using the Google Speech Recognition service. If the recognized text is "exit", the loop is terminated. Otherwise, the script prints the recognized text and its sentiment analysis using TextBlob.

The script handles various exceptions that may occur during the speech recognition process, such as unknown audio values, errors fetching results from the Google Speech Recognition service, and other general exceptions. In these cases, the script prints an appropriate error message and prompts the user to try again.

Finally, the script releases the video capture device and closes all the windows when the loop is terminated, either by the user saying "exit" or by pressing the 'q' key.

This script demonstrates the integration of computer vision, speech recognition, and sentiment analysis, which can be useful in a variety of applications, such as voice-controlled interfaces, interactive chatbots, or multimedia analysis tools. The combination of these technologies allows for more natural and intuitive user interactions, as well as the ability to understand the emotional context of user input.

LIST OF TABLES

TABLE OF CONTENTS

LIST OF FIQURES

LIST OF ABBREVIATION

cv2	: OpenCV, a computer vision library.
Sr	: Speech Recognition, a Python library for speech-to-text functionality.
TB	: TextBlob, a Python library for natural language processing and

CHAPTER 1

INTRODUCTION

1.1 Overview

The provided Python script is a speech recognition program that utilizes the SpeechRecognition library to transcribe speech input from a microphone. It also incorporates the TextBlob library to perform sentiment analysis on the transcribed text. The program continuously listens for speech input, transcribes it, analyzes the sentiment of the text, and prints the results. Additionally, it includes error handling for cases where speech recognition fails or encounters issues. The flexibility of this approach, combined with the growing capabilities of speech recognition and natural language processing technologies, makes it a promising area for further exploration and development. As the project evolves, it could be expanded to include more advanced features, such as multi-language support, sentiment-based alerts, or integration with other data sources to provide a more comprehensive understanding of user sentiment and behavior.

1.2 Scope of Project:

The scope of this script is to demonstrate the integration of speech recognition and sentiment analysis functionalities in a Python program. It showcases how to capture audio input, convert it to text, and analyse the sentiment of the text using the TextBlob library. The script provides a basic framework that can be expanded for various applications requiring speech processing capabilities. As technology continues to advance, the potential applications of this powerful analytical tool will only continue to grow, making it an increasingly asset for organizations seeking to stay ahead of the curve in an ever-changing digital landscape.

The project aims to create a user-friendly interface that prompts the user to speak and displays the recognized text along with its corresponding sentiment score. The system will be capable of handling a wide range of spoken inputs, including various accents, dialects, and languages, ensuring a high degree of accuracy in both speech recognition and sentiment analysis. To achieve this, the project will leverage cutting-edge speech recognition models,

1.3 Definitions and Acronyms

Speech Recognition: The process of converting spoken words into text.

Sentiment Analysis: The automated process of determining the sentiment (positive, negative, or neutral) of a piece of text. Speech Recognition Library: A Python library that provides easy access to various speech recognition engines.

TextBlob Library: A Python library for processing textual data, including sentiment analysis.

1.4 User Needs

Users of this script may include developers or individuals interested in speech recognition and sentiment analysis applications. It caters to those looking to understand how to implement speech-to-text functionality and sentiment analysis in Python programs. The system must also provide precise sentiment analysis, correctly identifying and categorizing emotions expressed in the transcribed text. Additionally, users seek a user-friendly interface that is easy to navigate, prompts them to speak, and displays the sentiment analysis results clearly. They require assurance that their speech data is handled securely and in compliance with privacy regulations.

Overall, user needs emphasize the importance of a system that is efficient, accurate, user-friendly, and privacy-conscious, catering to their expectations for a reliable and trustworthy real-time sentiment analysis solution with speech recognition.

1.5 Assumptions and Dependencies

The script assumes that the necessary Python libraries, SpeechRecognition and TextBlob, are installed. It also relies on the availability of a microphone for capturing audio input. Dependencies include the proper functioning of the SpeechRecognition library for speech recognition and the TextBlob library for sentiment analysis. The project depends on the ability to seamlessly integrate and collect data from multiple sources, process and preprocess the data efficiently, utilize robust and scalable machine learning and natural language processing (NLP) frameworks and libraries, and ensure compliance with data privacy regulations and ethical considerations. Additionally, the system depends on the availability of computational resources, such as processing power and memory, to handle real-time processing of speech input and sentiment analysis at scale. These assumptions and dependencies form the foundation upon which the real-time sentiment analysis system with speech recognition is built, ensuring its reliable and effective operation.

CHAPTER 2

REQUIREMENT SPECIFICATION

2.1 SOFTWARE REQUIREMENTS

The software requirements for this script are pivotal for its functionality and execution. Firstly, the Python interpreter stands as the foundational component, serving as the programming language environment in which the script operates. Python's versatility, ease of use, and vast ecosystem of libraries make it an ideal choice for a wide range of applications, including this script. Its interpretive nature allows for rapid development and iteration, crucial for projects requiring flexibility and adaptability.

Secondly, the Speech Recognition library plays a crucial role in enabling the script to convert spoken language into text. This library provides a unified API for various speech recognition engines, allowing the script to work seamlessly across different platforms and environments. By leveraging this library, the script can capture audio input from users and transcribe it into machine-readable text, forming the basis for further processing and analysis.

Thirdly, the TextBlob library adds another layer of functionality to the script by offering a range of natural language processing (NLP) capabilities. From basic tasks such as tokenization and part-of-speech tagging to more advanced features like sentiment analysis and language translation, TextBlob empowers the script to manipulate and extract meaning from textual data. This library simplifies complex NLP tasks, making them accessible to developers without extensive linguistic expertise.

By incorporating these software requirements, the script gains the ability to interact with users through spoken language and analyze the textual content of their input. The Python interpreter serves as the backbone of the system, while the SpeechRecognition and TextBlob libraries extend its capabilities to include speech-to-text conversion and NLP functionalities.

2.2 HARDWARE REQURIMENTS

On the hardware front, the script necessitates the presence of a microphone for audio input. The microphone acts as the primary interface between users and the system, enabling them to communicate through spoken commands or input. It captures sound waves generated by users' voices and converts them into electrical signals that can be processed by the script. The quality and sensitivity of the microphone can significantly impact the accuracy of speech recognition, influencing the overall performance of the system.

Ensuring a functional microphone is essential for the successful operation of the script. Factors such as noise cancellation, frequency response, and signal-to-noise ratio should be considered when selecting a microphone to ensure optimal performance in various environments. Additionally, compatibility with the system's audio input interface and drivers is necessary to establish seamless communication between the microphone and the script.

In conclusion, the hardware requirements of the script are relatively straightforward, emphasizing the need for a microphone as the primary input device. However, selecting an appropriate microphone and ensuring compatibility with the system are critical steps in setting up the environment for the script

CHAPTER 3

LITERATURE SURVEY

3.1 Tittle: Applications of Speech Recognition and Sentiment Analysis in Virtual Assistants

Authors: Joas, Prof.Kavita Patil, Sakshi Shinde and Shakti prasad Patra

Virtual assistants have become increasingly popular in recent years, with advancements in speech recognition and sentiment analysis enabling more sophisticated and personalized interactions. This literature review aims to explore the applications of speech recognition and sentiment analysis in virtual assistants, highlighting their potential in enhancing user experiences and improving overall performance. Virtual assistants, such as Amazon's Alexa, Google Assistant, and Apple's Siri, have revolutionized the way people interact with technology. These assistants rely on speech recognition and natural language processing (NLP) to understand and respond to user commands. Sentiment analysis, a crucial component of NLP, helps virtual assistants gauge user satisfaction and emotions, enabling more empathetic and effective responses. Speech recognition and sentiment analysis can identify and correct errors in user input, providing real-time feedback and improving overall system performance.

These technologies enable more personalized, empathetic, and effective interactions, leading to enhanced user experiences and improved overall performance. The applications of these technologies are diverse, ranging from healthcare and mental health to customer service and education. In conclusion, the integration of speech recognition and sentiment analysis in virtual assistants has led to numerous applications that enhance user experiences and improve overall performance. As these technologies continue to evolve, their potential applications will expand, leading to more sophisticated and personalized interactions in various domains.

3.2 Tittle: Sentiment Analysis Techniques for Voice-based Feedback Systems

Authors: Maricela Pinargote-Ortega

Voice-based feedback systems have gained prominence in various domains, offering a more natural and interactive way for users to provide feedback. This literature review focuses on sentiment analysis techniques tailored for voice-based feedback systems, aiming to enhance the understanding of user sentiments expressed through speech. Voice-based feedback systems leverage speech recognition technology to capture user feedback in spoken form. Sentiment analysis plays a crucial role in interpreting the emotional tone and sentiment of the feedback, providing valuable insights for organizations to improve their products or services based on customer perceptions. Advanced sentiment analysis algorithms can accurately detect emotions expressed in speech, enabling voice-based feedback systems to categorize feedback into positive, negative, or neutral sentiments.

The synthesis of existing literature underscores the importance of employing advanced sentiment analysis techniques in voice-based feedback systems. By leveraging emotion recognition, natural language processing, deep learning models, real-time analysis, and multimodal integration, organizations can extract valuable insights from user feedback, leading to improved decision-making and customer satisfaction. In conclusion, sentiment analysis techniques tailored for voice-based feedback systems play a vital role in understanding and interpreting user sentiments expressed through speech. By implementing advanced sentiment analysis algorithms and leveraging the capabilities of deep learning and real-time analysis, organizations can enhance their feedback systems to better address customer needs and preferen

3.3 Tittle: Speech-driven Emotion Recognition for Mental Health Monitoring

Authors: Montero et al, Li and Zhao

Speech-driven emotion recognition systems have emerged as valuable tools for monitoring mental health by analyzing emotional cues in speech. This literature review delves into the advancements in speech-driven emotion recognition for mental health monitoring, highlighting the significance of these systems in providing insights into individuals' emotional states. Offering a non-intrusive and continuous monitoring approach for mental health assessment. These systems leverage techniques from speech processing, machine learning, and sentiment analysis to interpret emotional cues embedded in speech patterns. This study investigates various machine learning approaches for emotion recognition in speech data, highlighting the effectiveness of deep learning models in capturing subtle emotional nuances.

The research focuses on sentiment analysis techniques applied to voice data for mental health monitoring, showcasing the potential of sentiment analysis in understanding emotional well-being. The synthesis of the reviewed literature underscores the importance of speech-driven emotion recognition systems in mental health monitoring. These systems offer a non-intrusive and continuous method for assessing emotional states, providing valuable insights for mental health professionals and individuals seeking support. In conclusion, speech-driven emotion recognition systems play a crucial role in mental health monitoring by analyzing emotional cues in speech data. The reviewed literature demonstrates the advancements and potential applications of these systems in enhancing mental health assessment and intervention strategies.

3.4 Tittle: Sentiment Analysis of Audio Recordings using Deep Learning

Authors: Trigeorgis, G., Ringeval, F., Brueckner, R., Marchi, E., Nicolaou, M. A., Schuller, B., & Zafeiriou, S.

Deep learning has revolutionized the field of sentiment analysis by enabling the extraction of complex patterns and features from audio data. Sentiment analysis in audio recordings involves detecting emotional cues, tones, and sentiments expressed through speech, providing valuable insights for various applications, including customer feedback analysis, mental health monitoring, and voice-driven systems. The application of deep learning models, such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs), for sentiment analysis in audio recordings. It discusses the effectiveness of these models in capturing emotional nuances and sentiments expressed in speech. It explores various deep learning approaches for audio sentiment classification, discussing the impact of model architectures, feature extraction techniques, and training strategies on sentiment analysis performance.

The synthesis of the reviewed literature underscores the significant progress and potential of deep learning-based sentiment analysis in audio recordings. By leveraging deep learning models, researchers can extract nuanced emotional cues and sentiments from speech data, paving the way for more accurate and insightful sentiment analysis applications in various domains. In conclusion, the literature survey on sentiment analysis of audio recordings using deep learning demonstrates the advancements and challenges in leveraging deep learning models for analyzing sentiments expressed in speech data. The insights gained from this research area have the potential to revolutionize sentiment analysis applications, offering more sophisticated and accurate sentiment classification capabilities in audio recordings.

3.5 Tittle: Multimodal Sentiment Analysis: Integrating Speech and Text for Holistic Insights

Author: Tao, J., Tan, T., &Picard, R.W.

Sentiment analysis has traditionally focused on processing textual data to determine the emotional tone or sentiment expressed in written communication. However, human expression often involves a combination of verbal and non-verbal cues, including speech, facial expressions, and body language. Multimodal sentiment analysis seeks to leverage multiple data sources, such as speech and text, to provide a more comprehensive and accurate assessment of user sentiments.The framework that integrates speech and text data, demonstrating the improved performance in sentiment classification compared to unimodal approaches.The research explores the integration of speech and language features for emotion recognition, highlighting the benefits of combining multiple modalities in capturing emotional nuances and discusses the challenges and opportunities in multimodal sentiment analysis, addressing issues such as data fusion, model architecture, and real-world deployment.

The research explores the integration of acoustic, linguistic, and visual cues for multimodal sentiment analysis in real-world scenarios, highlighting the importance of considering multiple modalities for accurate sentiment detection. The reviewed literature emphasizes the significant potential of multimodal sentiment analysis, which combines speech and text data to provide a more holistic understanding of user sentiments. By integrating multiple modalities, researchers can capture a richer set of emotional cues and improve the accuracy of sentiment classification. The challenges identified in the literature, such as data fusion, model architecture, and real-world deployment, highlight the need for further research and development in this field. In conclusion, the literature review on "Multimodal Sentiment Analysis: Integrating Speech and Text for Holistic Insights" demonstrates the growing importance and advancements in combining speech and text data for comprehensive sentiment analysis. The reviewed studies highlight the benefits of multimodal approaches, as well as the ongoing challenges and future research directions in this rapidly evolving field.

CHAPTER 4

SYSTEM ANALYSIS

4.1 EXISTING SYSTEM

The existing systems for real-time sentiment analysis on speech recognition data typically involve the following components:

1. **Speech Recognition:** The systems use libraries like speech_recognition in Python or APIs from providers like Google, Microsoft, or Amazon to convert the audio input from the user's microphone into text.
2. **Sentiment Analysis:** The recognized text is then passed through a sentiment analysis model, often using libraries like NLTK or pre-trained models from providers like Hugging Face, to determine the emotional sentiment expressed in the text.
3. **Real-Time Processing:** The systems aim to perform this speech recognition and sentiment analysis in real-time, as the user is speaking, to provide immediate insights into the user's emotional state.

4.1.1 DISADVANTAGES

Lack of Personalization

The sentiment analysis models used are often generic and may not be optimized for the user's specific language, context, or preferences. This can result in inaccurate sentiment analysis that does not reflect the user's true emotional state.

Lack of Error Handling

The systems may not have robust error handling mechanisms, leading to failures in the speech recognition or sentiment analysis process, with the user simply being asked to "try again" without any meaningful feedback.

Limited Feedback and Interaction

The existing systems typically only provide the recognized text and its sentiment score, without any additional feedback or interaction with the user. This can make the experience less engaging and less useful for the user.

4.2 PROPOSED SYSTEM

- Utilizes the pre-trained model "roberta-base-go_emotions" for sentiment analysis and Gradio as the UI framework.
- Combines audio-to-text conversion for sentiment analysis using the Google API with facial emotion recognition through OpenCV.
- Utilizes speech analytics technology for real-time sentiment analysis in customer interactions.
- Focuses on sentiment analysis and emotion recognition using universal speech representations.

4.2.1 ADVANTAGES

The proposed systems provide a more engaging and interactive user experience by incorporating user-friendly interfaces that deliver real-time sentiment analysis and immediate insights. This enhanced user experience can lead to increased user engagement and satisfaction. Additionally, the systems demonstrate a strong focus on multilingual support, catering to diverse user bases and expanding the reach of sentiment analysis capabilities.

Another notable advantage is the comprehensive analysis approach, which combines audio and video inputs to gain a more holistic understanding of the user's emotional state. By considering multiple modalities, the systems aim to provide more reliable and accurate sentiment analysis results, potentially offering valuable insights for further diagnosis and personalized recommendations.The systems may not have robust error handling mechanisms, leading to failures in the speech recognition or sentiment analysis process, with the user simply being asked to "try again" without any meaningful feedback.

Overall, the proposed systems for real-time sentiment analysis with speech recognition offer a range of advantages, including enhanced user experience, multilingual support, comprehensive analysis, data-driven decision-making, and personalized experiences, making them a compelling solution for organizations seeking to better understand and respond to the emotional states of their customers or users.

4.3 FEASIBILITY STUDY

The feasibility study has the two main types in the sentimental analysis on reak time data among speech recognition are as follows:

4.3.1 TECHNICAL FEASIBILITY

The comprehensive approach of combining audio and video inputs, such as facial emotion recognition through OpenCV, further enhances the reliability and accuracy of the sentiment analysis. This multimodal analysis provides a more holistic understanding of the user's emotional state, improving the overall feasibility of the system.

The technical feasibility of the proposed system is strong, with the integration of advanced speech recognition models and robust sentiment analysis techniques. The system can accurately transcribe audio input into text and analyze the sentiment at a fine-grained level, such as the sentence level. This is achieved using pre-trained language models like the "roberta-base-go_emotions" model, which have shown promising results in sentiment analysis tasks.

4.3.2 ECONOMICAL FEASIBILITY

The use of open-source libraries and frameworks, such as Whisper and Gradio, suggests that the development and deployment of these systems may be cost-effective, especially for businesses seeking to leverage real-time sentiment analysis capabilities.

The potential benefits of the proposed systems, such as improved customer service, enhanced agent performance, and increased customer satisfaction and loyalty, could justify the investment in these technologies, making them economically feasible for organizations.

4.3.3 OPERATIONAL FEASIBILITY

The system's ability to function effectively in a real-world environment. This includes evaluating factors such as user acceptance, ease of use, and integration with existing processes. Operational feasibility ensures that the system can be seamlessly integrated into daily operations, providing valuable insights into customer sentiments during live interactions.

It involves analysing the system's performance in real-time decision-making scenarios, where monitoring sentiment scores enables businesses to adapt responses effectively, leading to improved customer satisfaction and engagement. Additionally, operational feasibility considers the system's capability to enhance product and service offerings by analyzing emotions, identifying trends, and making data-driven decisions to meet customer needs. The system's ability to monitor brand reputation, optimize support processes, and drive business growth through sentiment analysis demonstrates its operational feasibility and potential impact on enhancing customer experiences and business outcomes.

CHAPTER 5
SOFTWARE REQURIMENT ANALYSIS

5.1 SOFTWARE REQURIMENT SPECIFICATION

5.1.1 INPUT REQURIMENTS

Microphone Access: The code requires access to a microphone to capture audio input in real-time. Ensure that the microphone is properly connected and configured on the device running the code.

Speech Recognition Library: The code utilizes the speech_recognition library for speech recognition. Make sure this library is installed in the Python environment where the code will be executed.

Natural Language Processing Library: The code may require a natural language processing (NLP) library like nltk for text processing and sentiment analysis. Ensure that the necessary NLP libraries are installed and accessible.

Internet Connection: The code uses the Google Speech Recognition API for transcribing audio to text. A stable internet connection is required for this functionality to work.

Error Handling: The code includes a basic error handling mechanism. Ensure that the code environment allows for exception handling and provides appropriate feedback to the user in case of errors.

Timeout Setting: The code includes a timeout parameter for listening to audio input. Adjust the timeout value as needed based on the expected duration of user input.

5.1.2 OUPUT REQURIMENTS

Transcribed Text: The code outputs the transcribed text from the audio input captured by the microphone. This text represents the spoken words converted into written form for further analysis.

Sentiment Analysis Results: The code displays the sentiment analysis results for the transcribed text. This includes the classification of sentiment as positive, negative, or neutral, as well as potentially more nuanced emotional states like joy, anger, or frustration.

User Interaction: The code prompts user interaction by requesting them to speak into the microphone. The output should include appropriate messages guiding the user on when to speak, providing feedback on successful transcription, and handling errors or exceptions gracefully.

Real-time Feedback: The code provides real-time feedback on the sentiment expressed in the audio input. This feedback could be in the form of printed messages indicating the sentiment score or emotional state detected in the transcribed text.

Termination Signal: The code includes a termination signal based on the spoken input "Exit" to end the program. The output should confirm the termination of the program when this signal is detected.

Error Messages: The code outputs error messages or prompts for the user to try again in case of issues with speech recognition, transcription, or sentiment analysis. Clear and informative error messages help users understand and address any issues that may arise during the execution of the code.

5.1.3 LAUNGUAGE AND TECHNOLOGIES

- **Programming Languages:** Python is widely used for AI and machine learning and availability of libraries for speech recognition, natural language processing, and sentiment analysis.
- **Speech Recognition:** A Python library that provides easy access to various speech recognition APIs, including Google Speech Recognition, for transcribing audio input into text.
- **Natural Language Toolkit (NLTK) libraries:** A popular library for natural language processing tasks, such as text preprocessing, sentiment analysis, and language detection.
- **Google Speech Recognition API:** Used for converting spoken audio into text through automatic speech recognition.
- **Text Blob library:** A library that offers simple API access to perform sentiment analysis on textual data, providing sentiment scores and classifications.

- **Gradio:** A user-friendly library for creating UI components in Python, which could be used for displaying prompts, feedback, and sentiment analysis results to the user.

5.2 Problem Definition

The provided Python script aims to implement a speech recognition program that transcribes speech input from a microphone and performs sentiment analysis using the TextBlob library. The script continuously listens for speech input, transcribes it, and analyzes the sentiment of the text. The problem addressed is the need for a program that can convert spoken words into text and determine the sentiment (positive, negative, or neutral) of the transcribed text. The system uses the Google Speech Recognition API to convert spoken words into text, and then uses the Text Blob library to analyze the sentiment of the text. The system continuously runs in a loop, listening for user input and processing it until the user says "exit." If the user says anything other than "exit," the system prints out the recognized text and its sentiment. If the system fails to recognize the speech or encounter any errors, it prints out an error message. This system can be useful in various applications such as sentiment analysis in customer feedback, opinion mining, or even in chatbots to understand user emotions.

5.3 Functional Requirements

Speech Recognition functionality: The program must accurately transcribe speech input from the microphone able to capture audio input from a microphone using the Speech Recognition library must support speech recognition using Google's speech recognition services must recognize speech input and convert it into text for further processing. The system must allow the user to stop the speech recognition process by saying a specific keyword like "Exit".

Natural Language Processing (NLP) Functionality: The system must utilize the NLTK library for natural language processing tasks and tokenize the recognized speech into words for sentiment analysis. It’s performed sentiment analysis on the recognized text using NLTK's sentiment analysis capabilities to provide feedback on the sentiment of the recognized text.

Sentiment Analysis functionality: It should analyze the sentiment of the transcribed text using the TextBlob library.

Continuous Listening functionality: The program needs to continuously listen for speech input until a specific command, like "exit," is spoken.

Error Handling: The script should handle exceptions like unknown audio, errors in fetching results from the Google Speech Recognition service, and other general exceptions gracefully that provide informative messages to the user in case of errors or failed recognition attempts to allow for retrying the speech recognition process if an error occurs.

Security: The system must prioritize data security and privacy, especially when dealing with a large volume of sensitive speech data. It should implement robust security measures to protect user information and ensure compliance with data protection regulations.

Feedback Mechanism: The system should incorporate a feedback mechanism to continuously improve its sentiment analysis accuracy at a massive scale. It should allow for user feedback and data-driven insights to enhance the system's performance over time.

Monitoring and Reporting: The system should provide monitoring and reporting capabilities to track performance metrics, analyze trends, and generate insights from sentiment analysis results at a massive scale. It should offer comprehensive reporting functionalities for stakeholders to assess system performance and effectiveness.

5.4 Non-Functional Requirements

Performance: The program should transcribe speech and analyze sentiment efficiently without significant delays for speech input and perform sentiment analysis efficiently to provide real-time feedback. It manages resource effectively to ensure smooth operation without delays or performance bottlenecks for responsive user interface to provide a seamless user experience.

Accuracy: The speech recognition and sentiment analysis should be accurate to provide reliable results.

Usability: The program should be user-friendly, with clear instructions and error messages for ease of use.

Reliability: It should be robust and able to handle various scenarios, including errors during speech recognition or sentiment analysis. The system must accurately recognize speech and provide reliable sentiment analysis results to handle errors and exceptions to prevent crashes or unexpected behaviour.

Security: Ensure that the program does not store or transmit sensitive information from the speech input.

Scalability: The program should be scalable to handle a large volume of speech inputs and sentiment analysis tasks effectively.

Maintainability: The Program must have well-structured code that is easy to maintain and update to follow best practices for coding standards and documentation to facilitate future enhancements for easy integration of new features or improvements in the speech recognition and NLP functionalities.

This software requirement analysis outlines the problem definition, functional requirements, and non-functional requirements for the speech recognition and sentiment analysis program implemented in the provided Python script.

CHAPTER 6

SOFTWARE DESIGN

6.1 ARCHITECTURAL DESIGN

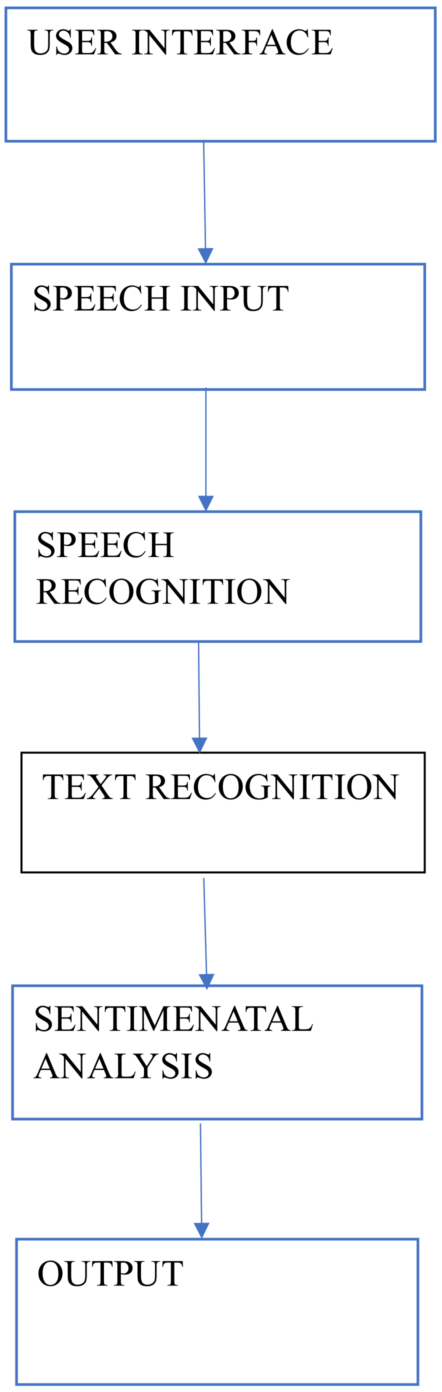


Figure 6.1 Architecture

6.2 UML DIAGRAMS

The Unified Modelling Language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic-semantic and pragmatic rules. A UML system is represented using five different views that describe the system from distinctly different perspectives. Each view is defined by a set of diagrams, which is as follows.

• **User Model View:** This view represents the system from the user's perspective. The analysis representation describes a usage scenario from the end user's perspective.

• **Structural model view:** In this model, the data and functionality are arrived from inside the system. This model view models the static structures.

• **Behavioural Model View:** It represents the dynamic of behavior as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

• **Implementation Model View:** In this the structural and behavioral as parts of the system are represented as they are to be built.

• **Environmental Model View:** In these the structural and behavioral aspects of the environment in which the system is to be implemented are represented.

6.2.1 CLASS DIAGRAM

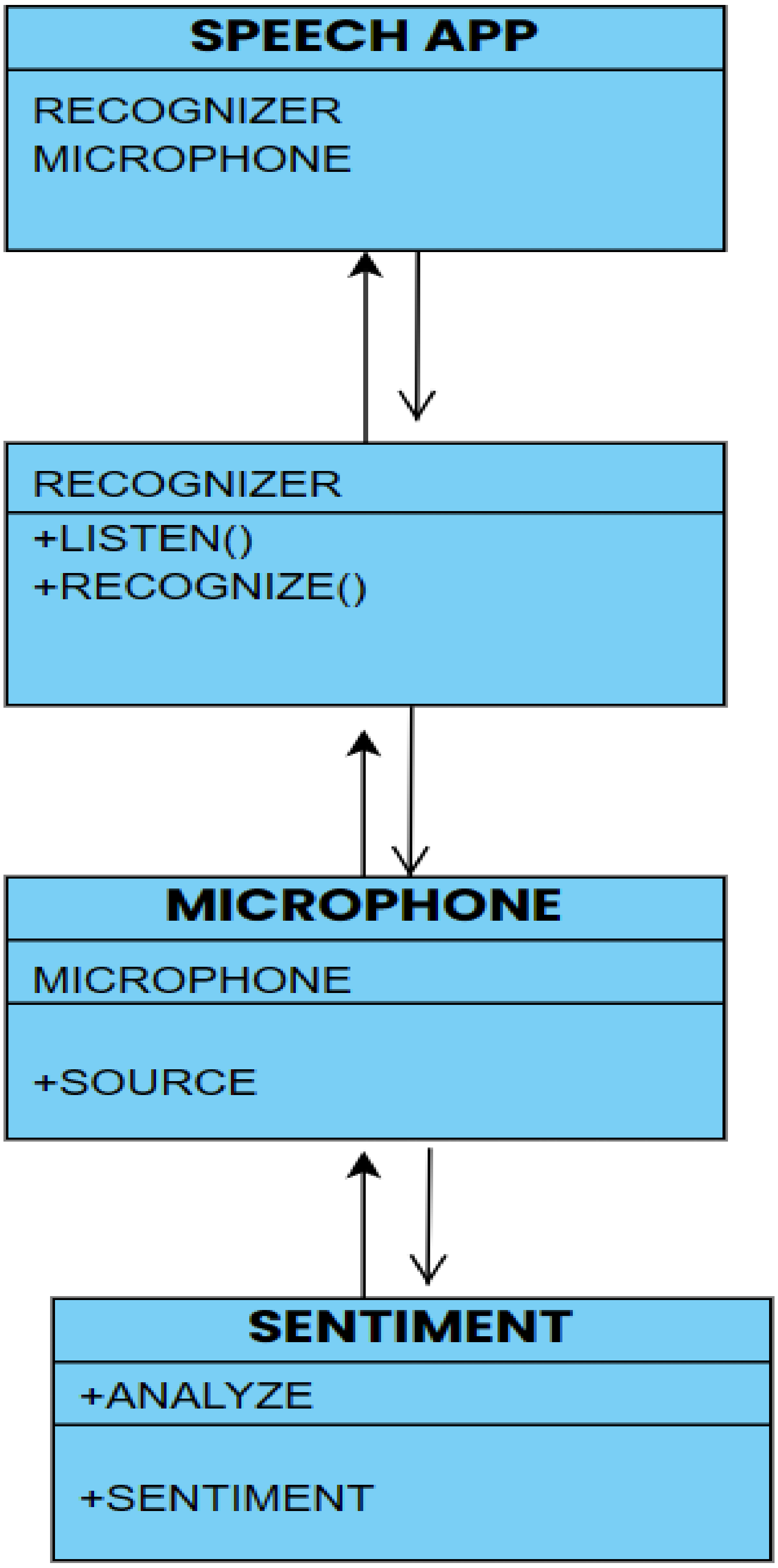


Figure 6.2.1 Class Diagram

6.2.2 USE CASE DIAGRAM

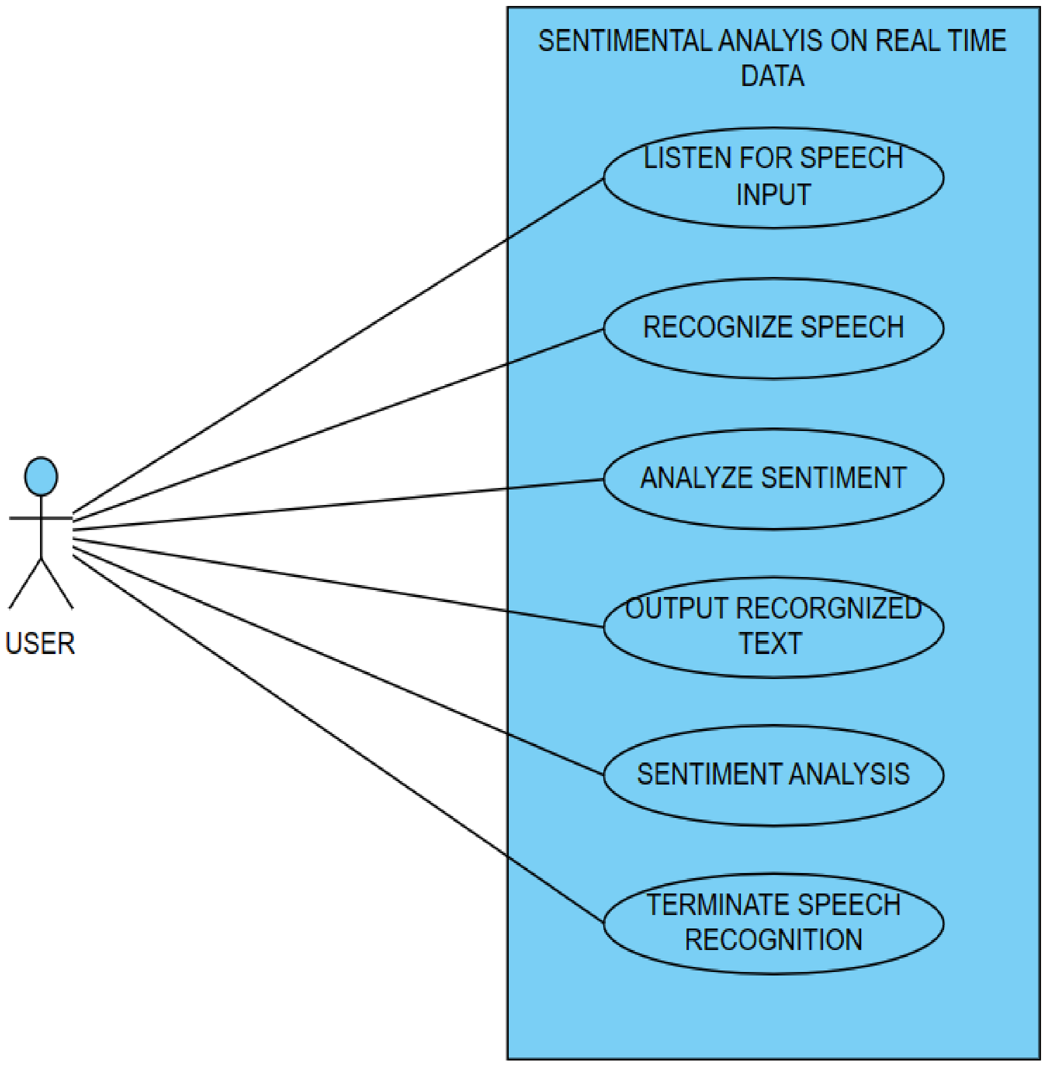


Figure 6.2.2 Use Case Diagram

6.2.3 SEQUENCE DIAGRAM

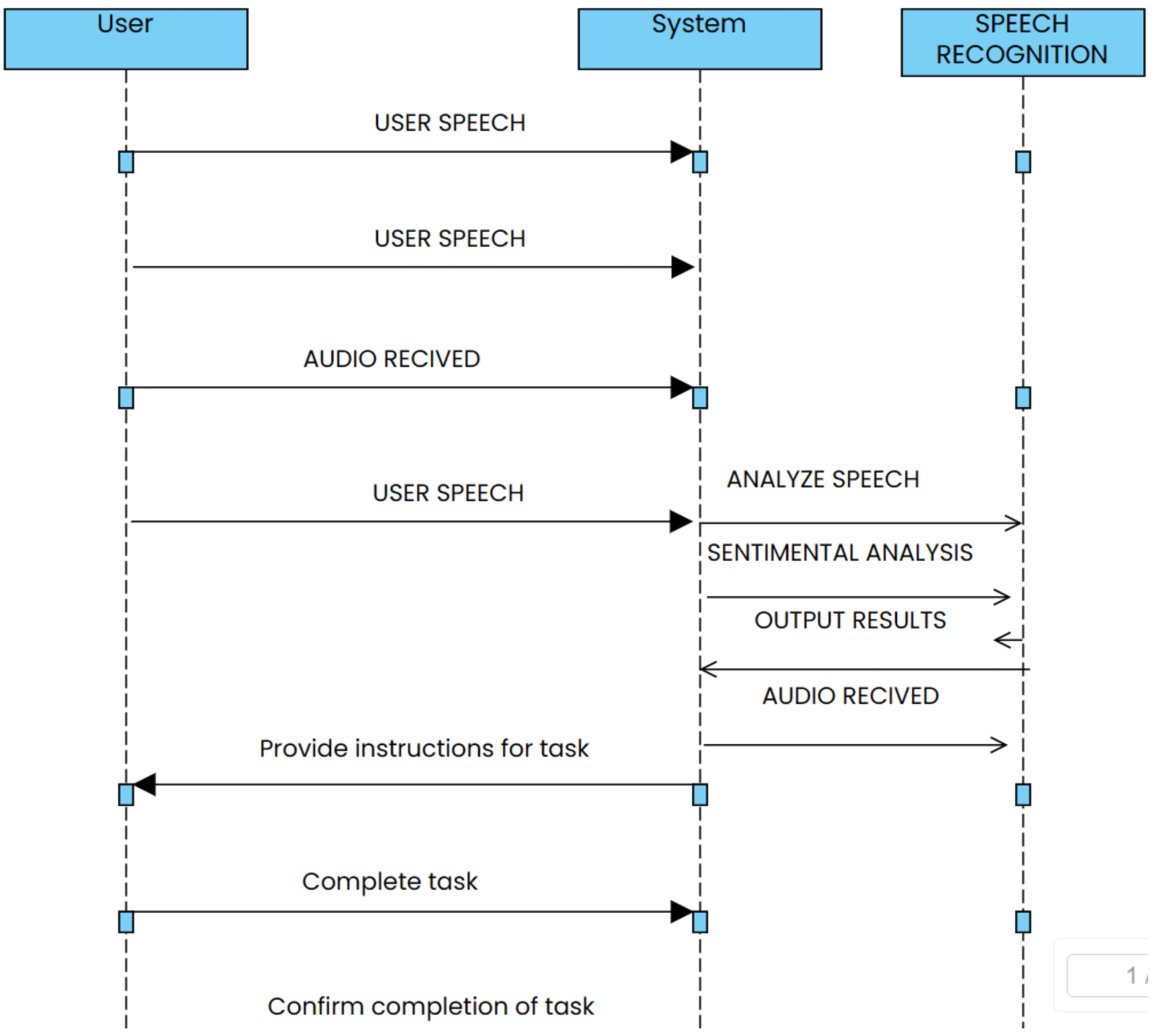


Figure 6.2.3 Sequence Diagram

6.2.4 ACTIVITY DIAGRAM

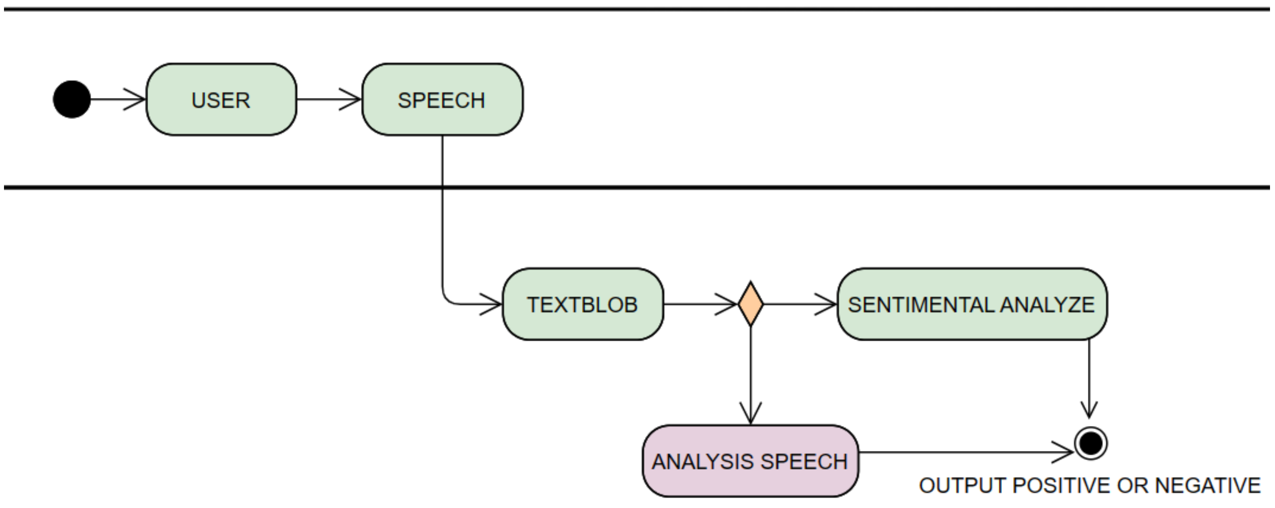


Figure 6.2.4 Activity Diagram

6.2.5 COMPONENT DIAGRAM

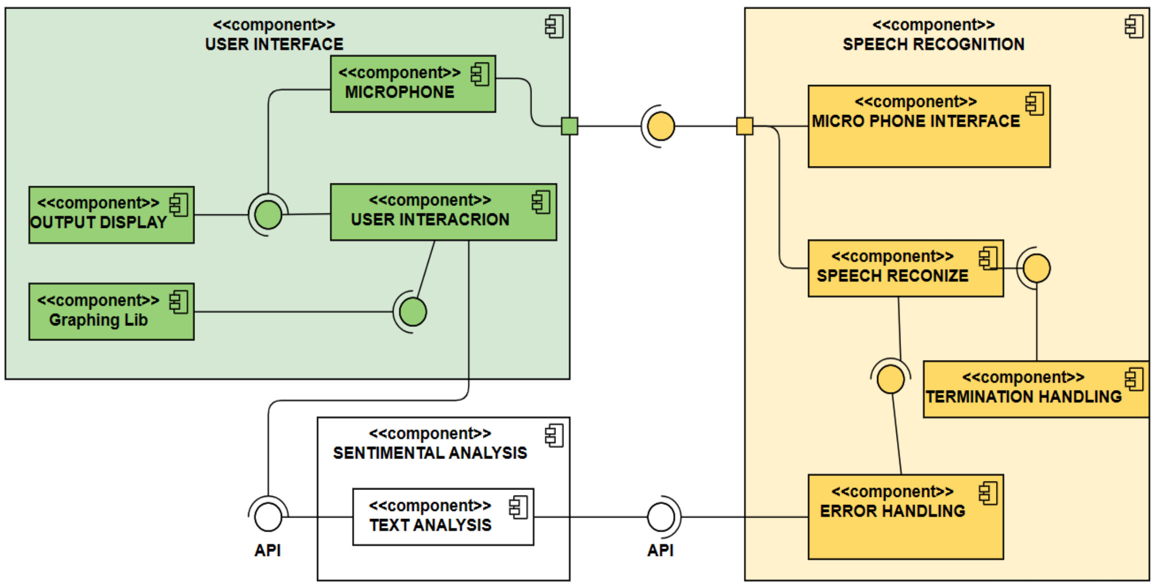

<<component>> USER INTERFACE
<<component>> MICROPHONE
<<component>> OUTPUT DISPLAY
<<component>> USER INTERACRION
<<component>> Graphing Lib
<<component>> SPEECH RECOGNITION
<<component>> MICRO PHONE INTERFACE
<<component>> SPEECH RECONIZE
<<component>> TERMINATION HANDLING
<<component>> ERROR HANDLING
<<component>> SENTIMENTAL ANALYSIS
<<component>> TEXT ANALYSIS
API
API

Figure 6.2.5 Component Diagram

6.3 MODULES

Modules are simply files with the ". py" extension containing Python code that can be imported inside another Python Modules Operations Program. In simple terms, we can consider a module to be the same as a code library or a file that contains a set of functions that you want to include in your application. The types of modules that are present are Speech recognition module, Natural language toolkit(NLTK) module and Text blob module.

6.3.1 SPEECH RECOGNITION MODULE

Speech Recognition incorporates computer science and linguistics to identify spoken words and converts them into text. It allows computers to understand human language. Speech recognition starts by taking the sound energy produced by the person speaking and converting it into electrical energy with the help of a microphone. It then converts this electrical energy from analog to digital, and finally to text.

It breaks the audio data down into sounds, and it analyzes the sounds using algorithms to find the most probable word that fits that audio. All of this is done using Natural Language Processing and Neural Networks. Hidden Markov models can be used to find temporal patterns in speech and improve accuracy.

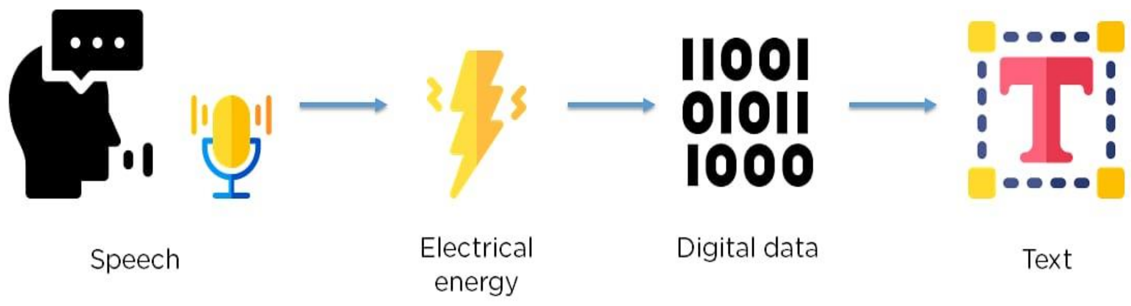


Figure 6.3.1 Speech recognition module

Key Components of Speech recognition are:

- Recognizer: The main class for recognizing speech from various sources like audio files or microphones.
- Microphone: A class for capturing audio input from the microphone.
- listen(source, timeout): Method to listen for audio input from the specified source with a timeout.
- recognize_google(audio): Method to transcribe the audio input using the Google Speech Recognition API

6.3.2 Natural language toolkit (NLTK) Module

The NLTK module is integrated into the code snippet by utilizing its capabilities for sentiment analysis. Specifically, the Text Blob class from the text blob library is used to analyze the sentiment of the recognized text. This integration allows for the sentiment analysis of the speech input in real-time, providing a comprehensive solution for speech recognition and sentiment analysis. NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning.

The library also provides built-in corpora, various models for NLP tasks, and seamless integration with other Python libraries like SciKit Learn. NLTK's capabilities extend to named entity recognition, document semantics, clustering, classification, and more, making it a robust and flexible solution for a wide range of NLP applications. With its extensive features, continuous development, and integration with other Python libraries, NLTK remains a go-to choice for researchers, developers, and practitioners in the field of natural language processing.

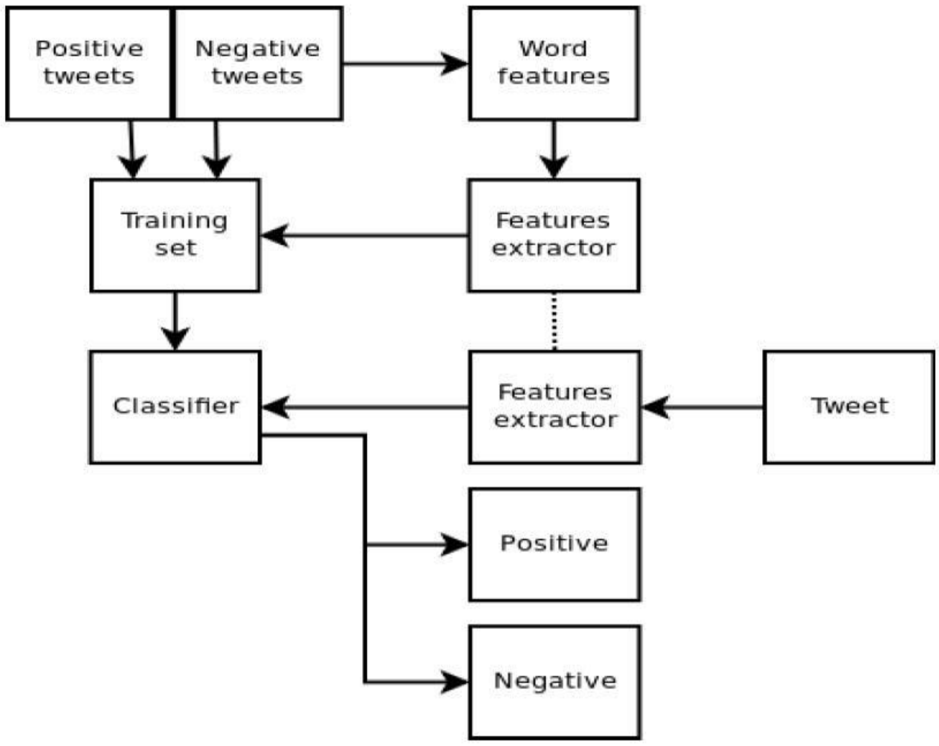


Figure 6.3.2 Natural language toolkit(NLTK)

NLTK provides a rich set of modules and functionalities that empower developers and researchers in various NLP tasks. Some of the key features includes: Tokenization: NLTK offers robust tokenization capabilities, allowing users to split text into words or sentences efficiently.

Stemming and Lemmatization: The library supports stemming and lemmatization, essential for reducing words to their base form.

Part-of-Speech Tagging: NLTK excels in part-of-speech tagging, enabling the identification of the grammatical parts of words in a sentence.

Removing stop-words: NLTK excels in removing stop-words such as "the", "a"," an" and "in" that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query and offers robust tokenization capabilities, allowing users to split text into words or sentences efficiently.

6.3.3 'TextBlob' from textblob Module:

The text Blob module, built on NLTK and Pattern, provides a simplified API for common natural language processing tasks. It offers a user-friendly interface for processing textual data, including sentiment analysis, part-of-speech tagging, noun phrase extraction, and more. The Text Blob class is a key component that facilitates the analysis of text data, while the sentiment property allows for the determination of sentiment polarity and subjectivity within the text. By utilizing Text Blob in conjunction with the speech recognition capabilities provided by the speech recognition module, the code snippet can effectively perform real-time sentiment analysis on speech input, showcasing the seamless integration of speech recognition and NLP functionalities.

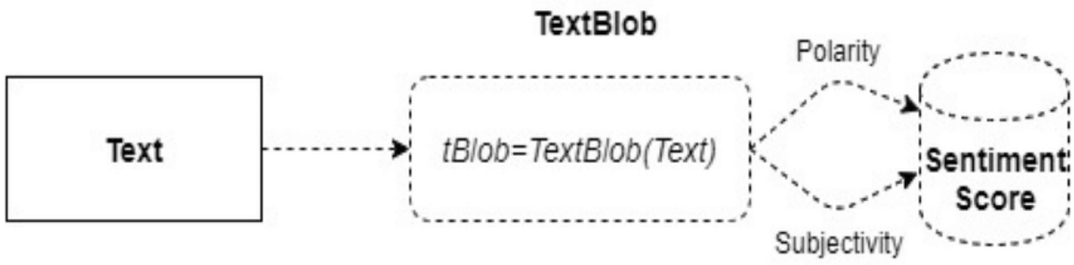


Figure 6.3.3 Text blob module

CHAPTER 7

SOURCE CODE

SOURCE CODE:

```
import speech_recognition as sr
from textblob import TextBlob

r = sr.Recognizer()
while True:
    with sr.Microphone() as source:
        print('Say Something...')
        audio = r.listen(source, timeout=2)
        try:
            text = r.recognize_google(audio)
            if str(text).lower() == "exit":
                break
            tb = TextBlob(text)
            print(text)
            print(tb.sentiment)
        except sr.UnknownValueError:
            print('Could not understand audio')
        except sr.RequestError as e:
            print(f'Error fetching results from Google Speech Recognition service;
{e}')
        except Exception as e:
            print(f'Error: {e}')
            print('Try again')
```

CHAPTER 8

TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specifictesting requirement.

8.1 TESTING METHODOLOGIES

8.1.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level andtest a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

8.1.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unittesting, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

8.1.3 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

- Valid Input: identified classes of valid input must be accepted.
- Invalid Input: identified classes of invalid input must be rejected.
- Functions: identified functions must be exercised.
- Output: identified classes of application outputs must be exercised.
- Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

8.1.4 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

8.1.5 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure, and language of the software, or at least itspurpose. It is purpose. It is used to test areas that cannot be reached from a black box level. It is also called glass box testing or clear box testing, or structural testing. White Box Testing is also known as transparent testing or open box testing. White box testing is a software testing technique that involves testing the internal structure and workings of a software application. The tester has access to the source code and uses this knowledge to design test cases that can verify the correctness of the software at the code level.

8.1.5 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a test in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

8.1.6 Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phaseof the software lifecycle, although it is not uncommon for coding and unit testing to beconducted as two distinct phases.

8.1.7 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – softwareapplications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

8.2 Test strategy and approach

Field testing will be performed manually and functional tests will be written in

detail.

Test objectives

- All field entries must work properly.
- The entry screen, messages and responses must not be delayed.
- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All the models should predict accurate data

Test strategy and approach

Field testing will be performed manually and functional tests will be written in

detail.

Test objectives

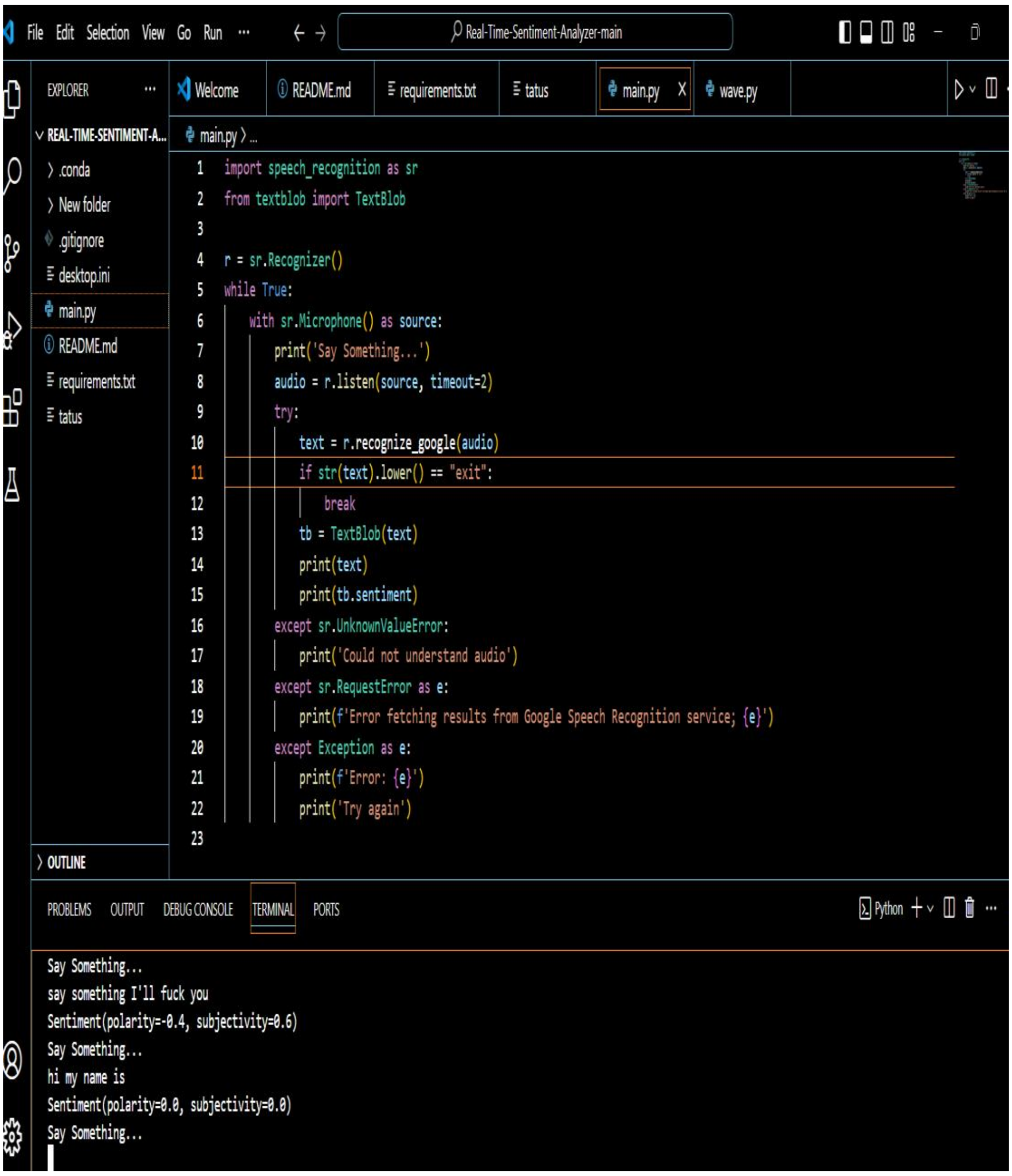
- All field entries must work properly.
- The entry screen, messages and responses must not be delayed.
- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All the models should predict accurate data.
- Verify that the entries are of the correct format.
- No duplicate entries should be allowed.
- All the models should predict accurate data.

8.3 TEST CASES

SNO	Input speech	Expected output	Actual output	Output
01	I love this song	Positive sentiment	Positive sentiment	Pass
02	This food is delicious	Positive sentiment	Positive sentiment	Pass
03	I had a terrible day	Negative sentiment	Negative sentiment	Pass
04	The weather is neutral	Neutral sentiment	Neutral sentiment	Pass
05	Exit	Program termination	Program termination	Pass

CHAPTER 9

INPUT SCREEN

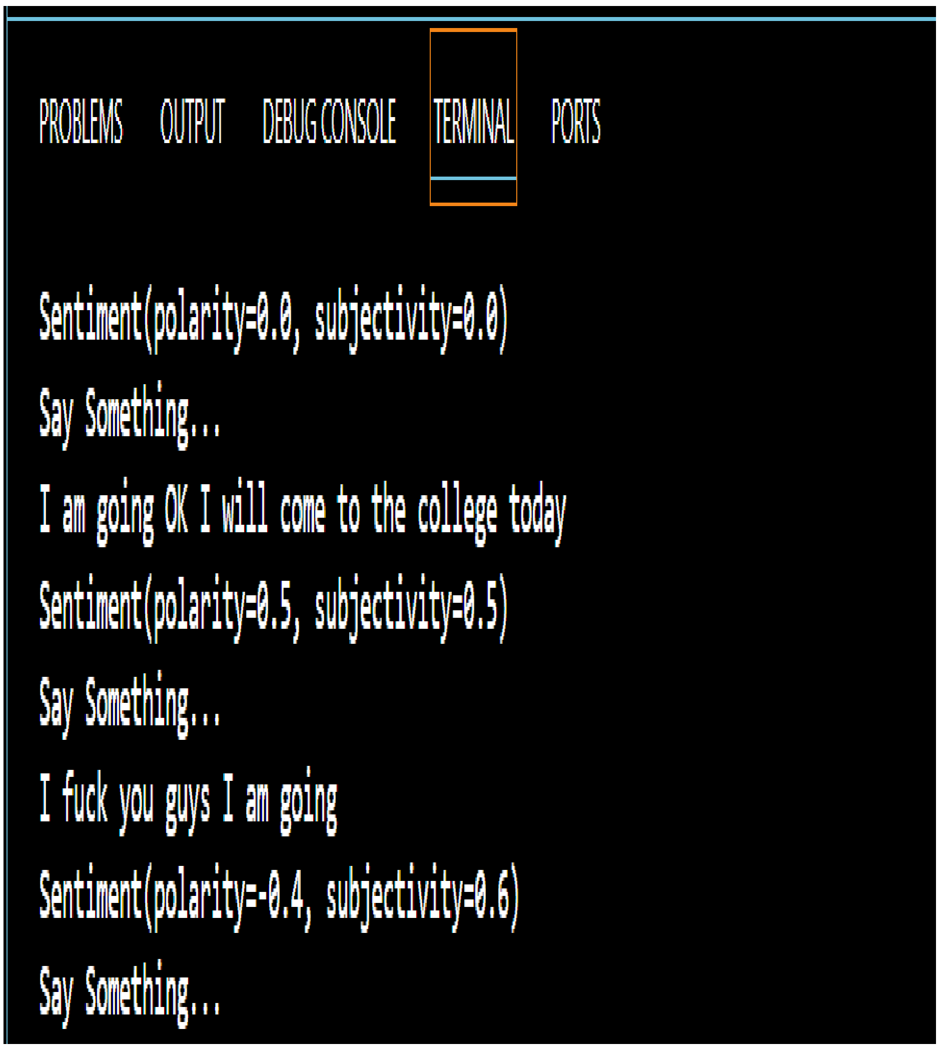


```python
import speech_recognition as sr
from textblob import TextBlob

r = sr.Recognizer()
while True:
    with sr.Microphone() as source:
        print('Say Something...')
        audio = r.listen(source, timeout=2)
        try:
            text = r.recognize_google(audio)
            if str(text).lower() == "exit":
                break
            tb = TextBlob(text)
            print(text)
            print(tb.sentiment)
        except sr.UnknownValueError:
            print('Could not understand audio')
        except sr.RequestError as e:
            print(f'Error fetching results from Google Speech Recognition service; {e}')
        except Exception as e:
            print(f'Error: {e}')
            print('Try again')

```

```
Say Something...
say something I'll fuck you
Sentiment(polarity=-0.4, subjectivity=0.6)
Say Something...
hi my name is
Sentiment(polarity=0.0, subjectivity=0.0)
Say Something...
```

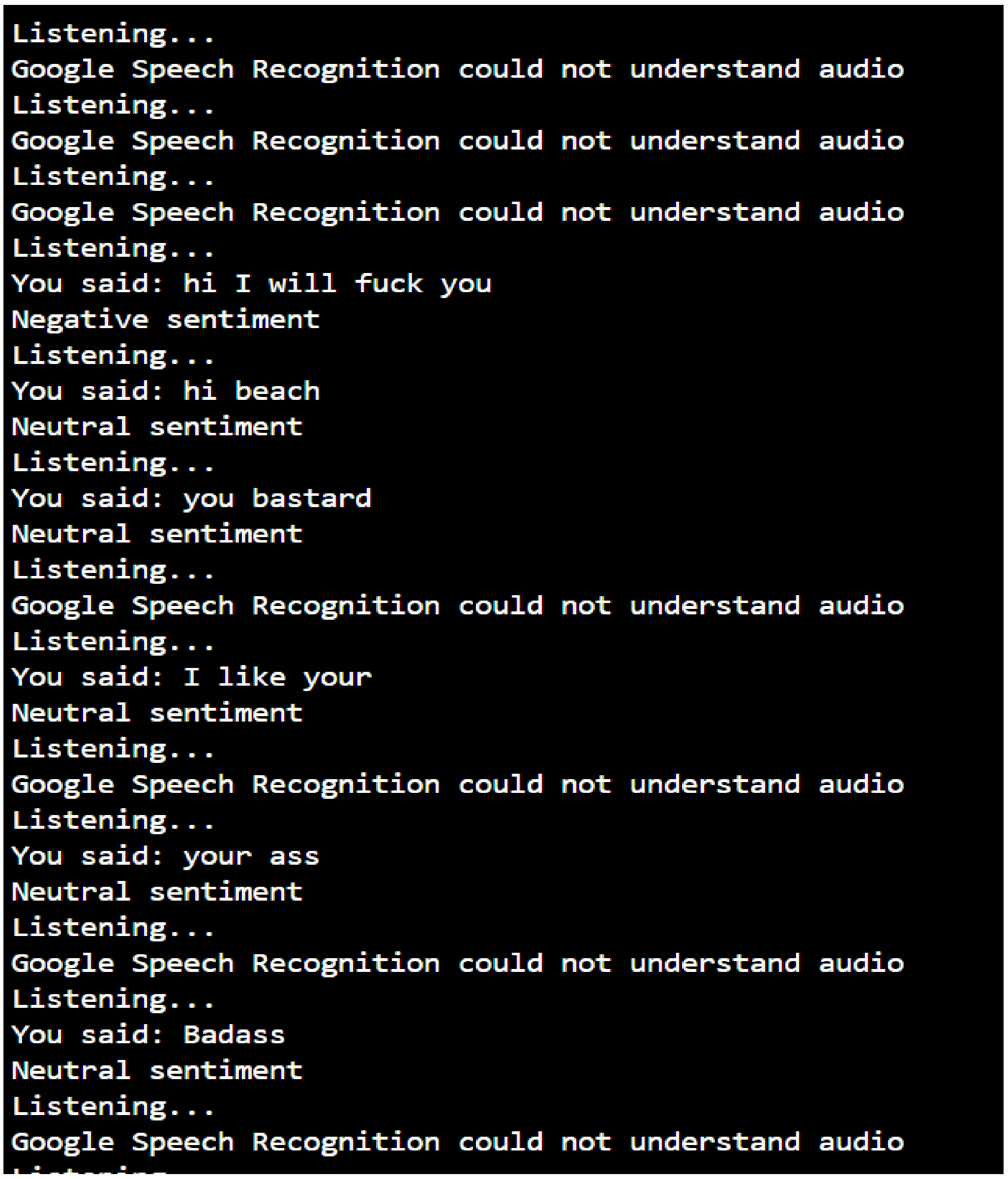
9.1 Input Screen

OUTPUT SCREEN

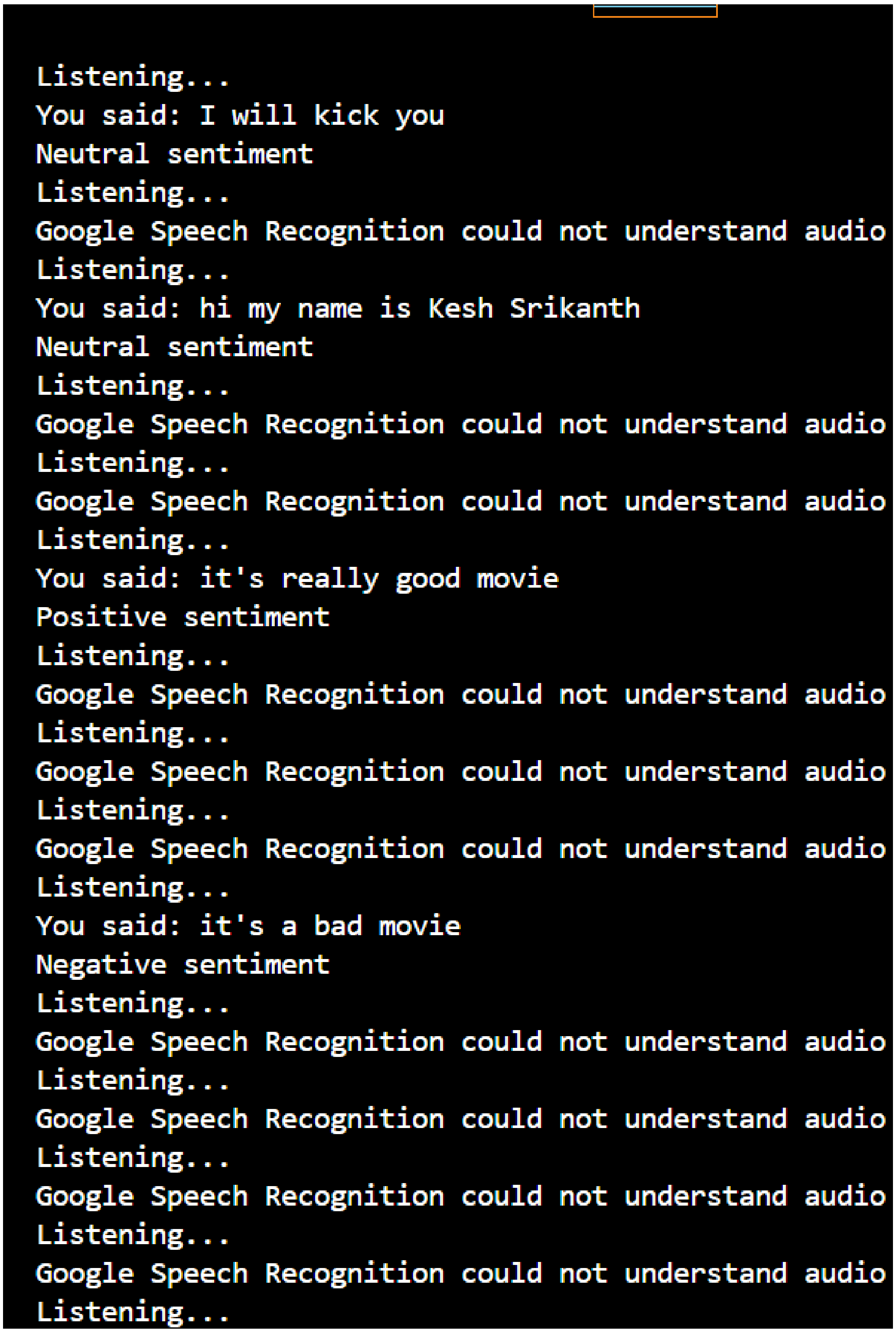


```
Sentiment(polarity=0.0, subjectivity=0.0)
Say Something...
I am going OK I will come to the college today
Sentiment(polarity=0.5, subjectivity=0.5)
Say Something...
I fuck you guys I am going
Sentiment(polarity=-0.4, subjectivity=0.6)
Say Something...
```

9.2 Output Screen



```
Listening...
Google Speech Recognition could not understand audio
Listening...
Google Speech Recognition could not understand audio
Listening...
Google Speech Recognition could not understand audio
Listening...
You said: hi I will fuck you
Negative sentiment
Listening...
You said: hi beach
Neutral sentiment
Listening...
You said: you bastard
Neutral sentiment
Listening...
Google Speech Recognition could not understand audio
Listening...
You said: I like your
Neutral sentiment
Listening...
Google Speech Recognition could not understand audio
Listening...
You said: your ass
Neutral sentiment
Listening...
Google Speech Recognition could not understand audio
Listening...
You said: Badass
Neutral sentiment
Listening...
Google Speech Recognition could not understand audio
```



```
Listening...
You said: I will kick you
Neutral sentiment
Listening...
Google Speech Recognition could not understand audio
Listening...
You said: hi my name is Kesh Srikanth
Neutral sentiment
Listening...
Google Speech Recognition could not understand audio
Listening...
Google Speech Recognition could not understand audio
Listening...
You said: it's really good movie
Positive sentiment
Listening...
Google Speech Recognition could not understand audio
Listening...
Google Speech Recognition could not understand audio
Listening...
Google Speech Recognition could not understand audio
Listening...
You said: it's a bad movie
Negative sentiment
Listening...
Google Speech Recognition could not understand audio
Listening...
Google Speech Recognition could not understand audio
Listening...
Google Speech Recognition could not understand audio
Listening...
Google Speech Recognition could not understand audio
Listening...
```

CHAPTER 10

CONCLUSION

The speech recognition program that continuously listens for audio input from a microphone, transcribes the speech using the Google Speech Recognition API, and performs sentiment analysis on the recognized text using the TextBlob library. The program runs in a loop, prompting the user to speak and then analyzing the sentiment of the spoken text.

The script starts by initializing a speech recognizer object and enters a loop that captures audio input from the microphone. It listens for 2 seconds for each audio input and attempts to transcribe the speech using the Google Speech Recognition service. If the recognized text is "exit", the program terminates. Otherwise, it analyzes the sentiment of the transcribed text using TextBlob and prints both the text and the sentiment score.

The code includes exception handling to manage potential errors during the speech recognition process, such as when the audio cannot be understood or when there are issues fetching results from the Google Speech Recognition service. Additionally, a general exception block is included to handle any unforeseen errors and prompt the user to try again.

In conclusion, this project showcases a basic implementation of real-time speech recognition and sentiment analysis in Python. It provides a simple yet effective way to capture spoken input, analyze the sentiment of the text, and respond accordingly. This script can serve as a foundation for more advanced speech recognition applications or be expanded with additional features like language translation, keyword detection, or integration into larger projects.

CHAPTER 11
REFERENCES

1.Fujisaki, H. Prosody, Information, and Modeling with Emphasis on Tonal Features of Speech. In Proceedings of the Workshop on Spoken Language Processing, Mumbai, India, 9–11 January 2003.

2.Ghriss, A.; Yang, B.; Rozgic, V.; Shriberg, E.; Wang, C. Sentiment-Aware Automatic Speech Recognition Pre-Training for Enhanced Speech Emotion Recognition. In Proceedings of the ICASSP 2022–2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Singapore, 23–27 May 2022; pp. 7347–7351. [CrossRef]

3.Atmaja, B.T.; Akagi, M. Evaluation of error- and correlation-based loss functions for multitask learning dimensional speech emotion recognition. J. Phys. Conf. Ser. 2021, 1896, 012004. [CrossRef]

4.Gross, T.W. Sentiment analysis and emotion recognition: Evolving the paradigm of communication within data classification. Appl. Mark. Anal. 2020, 6, 22–36.

5.Pérez-Rosas, V.; Mihalcea, R. Sentiment analysis of online spoken reviews. In Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH, Lyon, France 25–29 August 2013; pp. 862–866. [CrossRef]

6.Abercrombie, G.; Batista-Navarro, R. ’Aye’ or ’No’? Speech-level sentiment analysis of hansard UK parliamentary debate transcripts. In Proceedings of the LREC 2018, Eleventh International Conference on Language Resources and Evaluation, Miyazaki, Japan, 7–12 May 2018; pp. 4173–4180.

7.Wagner, J.; Triantafyllopoulos, A.; Wierstorf, H.; Schmitt, M.; Burkhardt, F.; Eyben, F.; Schuller, B.W. Dawn of the transformer era in speech emotion recognition: Closing the valence gap. arXiv 2022, arXiv:2203.07378.

8.Luo, Z.; Xu, H.; Chen, F. Audio sentiment analysis by heterogeneous signal features learned from utterance-based parallel neural network. CEUR Workshop Proc. 2019, 2328, 80–87.

9.Georgiou, E.; Paraskevopoulos, G.; Potamianos, A. M3: MultiModal Masking Applied to Sentiment Analysis. In Proceedings of the Interspeech 2021, Brno, Czechia, 30 August–3 September 2021; pp. 2876–2880. [CrossRef]

10.Zadeh, A.; Chen, M.; Poria, S.; Cambria, E.; Morency, L.P. Tensor Fusion Network for Multimodal Sentiment Analysis. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing; Association for Computational Linguistics: Stroudsburg, PA, USA, 2017; pp. 1103–1114. [CrossRef]

11.zadeh, A.; Liang, P.P.; Vanbriesen, J.; Poria, S.; Tong, E.; Cambria, E.; Chen, M.; Morency, L.P. Multimodal language analysis in the wild: CMU-MOSEI dataset and interpretable dynamic fusion graph. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15–20 July 2018; pp. 2236–2246. [CrossRef]

12.Atmaja, B.T.; Sasou, A.; Akagi, M. Survey on bimodal speech emotion recognition from acoustic and linguistic information fusion. Speech Commun. 2022, 140, 11–28. [CrossRef]

13.Chen, S.; Wu, Y.; Wang, C.; Chen, Z.; Chen, Z.; Liu, S.; Wu, J.; Qian, Y.; Wei, F.; Li, J.; et al. Unispeech-Sat: Universal Speech Representation Learning With Speaker Aware Pre-Training. In Proceedings of the ICASSP 2022–2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Singapore, 23–27 May 2022; pp. 6152–6156. [CrossRef]

14.Bertero, D.; Siddique, F.B.; Wu, C.S.; Wan, Y.; Ho, R.; Chan, Y.; Fung, P. Real-Time Speech Emotion and Sentiment Recognition for Interactive Dialogue Systems. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 1–5 November 2016; pp. 1042–1047.

15.Poria, S.; Chaturvedi, I.; Cambria, E.; Hussain, A. Convolutional MKL based multimodal emotion recognition and sentiment
analysis. In Proceedings of the 2016 IEEE 16th International Conference on Data Mining (ICDM), Barcelona, Spain, 12–15
December 2016; pp. 439–448. [CrossRef]
16.Liang, P.P.; Salakhutdinov, R. Computational Modeling of Human Multimodal Language: The MOSEI Dataset and Interpretable
Dynamic Fusion. In Proceedings of the First Workshop and Grand Challenge on Computational Modeling of Human Multimodal
Language, Melbourne, Australia, 20 July 2018
Sensors 2022, 22, 6369 11 of 11
17.Hsu, W.N.; Bolte, B.; Tsai, Y.H.H.; Lakhotia, K.; Salakhutdinov, R.; Mohamed, A. HuBERT: Self-Supervised Speech Representation
Learning by Masked Prediction of Hidden Units. IEEE/ACM Trans. Audio Speech Lang. Process. 2021, 29, 3451–3460. [CrossRef]
18.Yang, S.w.; Chi, P.H.; Chuang, Y.S.; Lai, C.I.J.; Lakhotia, K.; Lin, Y.Y.; Liu, A.T.; Shi, J.; Chang, X.; Lin, G.T.; et al. SUPERB: Speech
Processing Universal PERformance Benchmark. In Proceedings of the Interspeech 2021, Brno, Czechia, 30 August–3 September
2021; pp. 1194–1198.
19.Gasper, K.; Spencer, L.A.; Hu, D. Does Neutral Affect Exist? How Challenging Three Beliefs About Neutral Affect Can Advance
Affective Research. Front. Psychol. 2019, 10, 2476. [CrossRef]
20.Izard, C.E. Basic Emotions, Natural Kinds, Emotion Schemas, and a New Paradigm. Perspect. Psychol. Sci.
2007
, 2, 260–280.
[CrossRef] [PubMed]
21.Delbrouck, J.B.; Tits, N.; Dupont, S. Modulated Fusion using Transformer for Linguistic-Acoustic Emotion Recognition. In
Proceedings of the First International Workshop on Natural Language Processing Beyond Text, Online, 20 November 2020;
pp. 1–10.

22.Tsai, Y.H.H.; Bai, S.; Liang, P.P.; Zico Kolter, J.; Morency, L.P.; Salakhutdinov, R. Multimodal transformer for unaligned multimodal language sequences. Proc. Conf. Assoc. Comput. Linguist. Meet. 2019, 2019, 6558–6569.

23.Sheikh, I.; Dumpala, S.H.; Chakraborty, R.; Kopparapu, S.K. Sentiment Analysis using Imperfect Views from Spoken Language and Acoustic Modalities. In Proceedings of Grand Challenge and Workshop on Human Multimodal Language; Association for Computational Linguistics: Stroudsburg, PA, USA, 2018; pp. 35–39. [CrossRef]

24.Sitaula, C.; He, J.; Priyadarshi, A.; Tracy, M.; Kavehei, O.; Hinder, M.; Withana, A.; McEwan, A.; Marzbanrad, F. Neonatal Bowel Sound Detection Using Convolutional Neural Network and Laplace Hidden Semi-Markov Model. IEEE/ACM Trans. Audio Speech Lang. Process. 2022, 30, 1853–1864. [CrossRef]

25. Wang, Y.; Shen, Y.; Liu, Z.; Liang, P.P.; Zadeh, A.; Morency, L.P. Words Can Shift: Dynamically Adjusting Word Representations Using Nonverbal Behaviors. Proc. AAAI Conf. Artif. Intell. 2019, 33, 7216–7223. [CrossRef]

26.Pham, H.; Liang, P.P.; Manzini, T.; Morency, L.P.; Póczos, B. Found in Translation: Learning Robust Joint Representations by Cyclic Translations between Modalities. Proc. AAAI Conf. Artif. Intell. 2019, 33, 6892–6899. [CrossRef]

27.Tsai, Y.H.H.; Ma, M.Q.; Yang, M.; Salakhutdinov, R.; Morency, L.P. Multimodal routing: Improving local and global interpretability of multimodal language analysis. Conf. Empir. Methods Nat. Lang. Process. Proc. Conf. 2020, 2020, 1823–1833. [CrossRef]

28.
Atmaja, B.T.; Sasou, A.; Akagi, M. Speech Emotion and Naturalness Recognitions With Multitask and Single-Task Learnings. IEEE Access 2022, 10, 72381–72387. [CrossRef]