

A Review on Speech-to-Text

Aman Raj Singh

Computer Science Engineering
Chandigarh University
Mohali, India
amanrajsinghsingh04@gmail.com

Prabhat Kumar

Computer Science Engineering
Chandigarh University
Mohali, India
prabhatkumar.cse@gmail.com

Ishani Rana

Computer Science Engineering
Chandigarh University
Mohali, India
Ishani.e14552@cumail.in

Sakshi Bhagat

Computer Science Engineering
Chandigarh University
Mohali, India
sakshibhagatofficial@gmail.com

Taniya Mukhija

Computer Science Engineering
Chandigarh University
Mohali, India
mutaniya114@gmail.com

Abstract— The current era represents the global apex of groundbreaking advances in artificial intelligence (AI) technology, especially in the field of speech-to-text (STT). This review article focuses on the development of human skills through smooth, natural language interaction between people and robots, providing a thorough overview and exploration of the impressive advancements made in recent years. The study outlines a model intended to reinvent human-computer interaction, highlighting its ability to translate spoken language into text and carry out commands via a conversational, dynamic interface. Real-world applications examine neural network designs, deep learning, natural language processing, and multimodal approaches; Python acts as the engine of execution, utilizing large-scale libraries like pyttsx3 and speech recognition. The model's development is highlighted by the investigation of methods, such as neural network topologies, which handle issues with various speech patterns, accents, and background noise. In the current AI scene, the study wants to contribute to existing discussions on the revolutionary influence of STT technology on human-computer interaction, despite current challenges including processing nuanced language and minimizing the impact of background noise on recognition accuracy.

Keywords—Speech-to-Text(STT), Natural language processing, Python-driven execution and libraries, Multimodal Approaches, Deep Learning, Recognition accuracy, Human-computer Interaction, Desktop Voice Assistant

1. INTRODUCTION

Consider you have a disorganized desktop; you recall that you have an important email to deliver. What if you could just speak your thoughts and they magically converted into a nicely prepared message instead of struggling with your keyboard? This is Speech-to-Text (STT) as it is currently evolving.

STT goes beyond traditional mouse and keyboard operations. It's about changing the way we talk to our electronics. Imagine using natural, spoken language to issue commands, write documents, or navigate your desktop environment. Your desktop experience will become more engaging and natural as a result of this paradigm shift from mechanical input to communication.

This looks simple interaction hidden a world of amazing technological capabilities. Imagine neural networks that function similarly to the human brain, deep learning algorithms that are able to recognize complexity in language, and Python, a programming language, smoothly guiding the process. Using initiatives like PyTtsx3 and speech recognition, these concepts are put into practice rather than remaining purely theoretical.

However, in this conversation between humans and machines, perfection takes time. The enemies are background noise and accents. But this review doesn't back down from a challenge—rather, it welcomes it. By doing this, it imagines a time when STT easily becomes part of desktop interactions and changes to fit the varied and dynamic layouts of our work environments.

Let's imagine a desktop where your words have the same power as the keys you use as we explore each aspect of STT. It's about creating a more organic and peaceful relationship between you and your desktop environment, not just about effective communication.

2. LITERATURE REVIEW

The prospective use of Speech-to-Text (STT) technology to improve human-computer interaction—especially with desktop assistants—has drawn a lot of attention in recent years. With a focus on applications, difficulties, and new trends, this review attempts to give an in-depth account of the state of the art in Speech-to-Text Desktop Assistant research and development.

Speech-to-Text desktop assistant development has been encouraged by advancements in speech recognition technology. Early systems had issues with accuracy and vocabulary limits. But with the development of deep

learning and machine learning algorithms, real-time processing speed and accuracy have significantly increased.

| Solution | Published Year | Key Features | Drawbacks |
|---------------------|----------------|---|---|
| Smith et al | 2018 | Improved workflow efficiency, hands-free operation. | Limited vocabulary, occasional accuracy issues. |
| Chen et al. | 2020 | Emphasized natural language understanding g, context-awareness. | Requires stable internet connection, learning curve. |
| Gao et al. | 2021 | Addressed noise cancellation challenges, privacy-preserving STT models. | Resource-intensive, may impact real-time processing. |
| Liang et al. | 2022 | Explored real-time learning, integration of contextual information. | Complexity in implementation n n, potential over-reliance on context. |
| Wang and Liu | 2019 | Explored ethical considerations, user privacy, data security, trust. | Potential user resistance due to privacy concerns. |

3. METHODOLOGY

Though speech-to-text technology has developed tremendously and is being used in more applications, in many parts of our life and work, but there remains obstacles and challenges due to its limitations or lack of use. The problems arise from either not having speech-to-text technology or using it insufficiently are Accessibility Barriers, Productivity Limitations, Exclusion of speech-dependent applications, challenges in multilingual environments, etc. These problems can only be addressed or solve by having an enhanced user experience, accessibility, and productivity of the methods. The main objective of this paper is to create a methodology which not only solve the problems also provide us the best solution of the problems which we are facing in the adaption or building of this environment. Keeping this in mind the methodology of the technology is introduced whose primary objective is to Integrate and Advanced Speech-To-Text Systems into the environments where you want to in build or improve your accuracy, language support, and real-time

transcription capabilities. By Natural Language Understanding including the deep learning we could lead to better understanding of conversational nuances, which makes better interactions with the assistant more natural and intuitive for any platforms. By having or bettering the features of offline capabilities ensures users to perform speech-to-text tasks without a constant internet connection, providing more flexibility. It contains different python libraries for the development and improving the accuracy, language support and for the real-time transcription also. The basic idea under the development of this Speech-to-text, or automatic speech recognition with the help of different types of machine learning algorithms and python libraries is that to convert spoken language into written text and written text into speech that involves processing audio signals, extracting features, training models for language understanding, decoding likely word sequences, and generating a final transcription. The goal is to enable computers to understand and transcribe human speech accurately, finding applications in voice assistants, transcription services, and other voice-activated systems. With the advance in machine learning and natural language processing we can improve the accuracy and performance of speech-to-text technology.[1]

The proposed methodology works on to the well-structured and planned step by step procedure to analyze and build a one of the crucial Speech-To-Text models which have the all the features of all kind of work that leads to the development of the automating the desktop tasks.

Algorithm for Speech-to-Text Model Development:

Step 1: Information Gathering:

- Get a wide collection of audio recordings that span the gamut of accents and voices the model is supposed to be able to handle.
- Assign ground truth transcriptions to each dataset entry.[2]

Step 2: Preprocessing the Data:

- Transform audio files into a format that is appropriate for handling.
- Create training, validation, and testing sets from the dataset.
- Take out pertinent information from the audio recording, usually by applying methods such as Mel-frequency cepstral coefficients (MFCCs).

Step 3: Model Architecture Selection:

- Select an appropriate architecture for the model of speech-to-text. Typical selections consist of Recurrent Neural Networks (RNNs), Long Short-Term Memory networks (LSTMs), or Convolutional Neural Networks (CNNs).
- Using pre-trained models or architectures for optimizing the speech recognition, like Deep Speech.[3]

Step 4: Model Training:

- Provide the data that has been preprocessed into the chosen model for model training.
- Utilizing the training set, train the model, and then use the validation set to assess its performance.
- Use algorithms like CTC (Connectionist Temporal Classification) loss to deal with training data that contains variable-length sequences.[4]

Step 5: Hyperparameter Tuning:

- Modify model hyperparameters to correlate with validation results, such as learning rate, batch size, and network architecture.
- Try various configurations to maximize the accuracy of the model.

Step 6: Evaluation:

- To guarantee that the model can be applied to fresh, untested data, evaluate its performance on the testing set.

deployment.[5]

Step 8: Integration with Desktop Assistant:

- Integration with Desktop Assistant: Apply the speech-to-text model that has been trained to the desktop assistant program.
- Provide a system that allows user input to be transcribed in real-time or almost in real-time.

Step 9: Continuous Improvement:

- Observe how the model performs in practical situations and get user input.
- Update the model on a regular basis to adjust to evolving user behavior and increase accuracy over time.[6]

Step 10: User Interaction:

- Use logic in the desktop assistant to get the speech-to-text module's transcriptions.[7]
- Create interactions based on the instructions and questions that users ask via the text that has been transcribed.[8]

Step 11: Error Handling:

- Put in place procedures to deal with mistakes politely, offering comments or requesting clarification when there are problems with identification.

The basic idea and basic work that can be used to work properly is shown in Fig: 1.

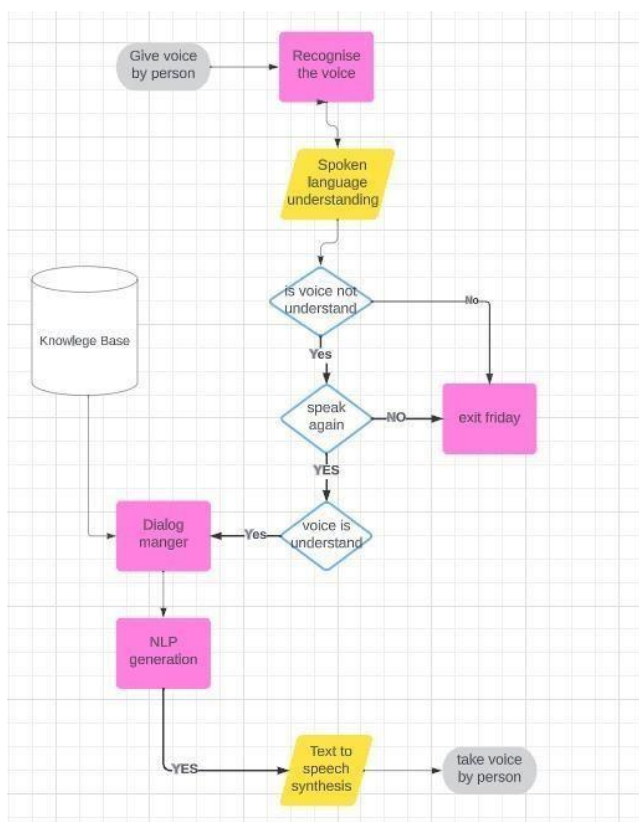


Fig. 1

- To measure the accuracy of transcription, consider important metrics such as Word Error Rate (WER) or Character Error Rate (CER).

Step 7: Model Optimization:

- Optimize the trained model for deployment, by considering the factors like model size, inference speed, and resource efficiency for desktop

4. OBJECTIVE

As you know Speech-to-Text is a very vast field of development and we can do tremendous things which will help to ease our work. In this paper I am highlighting one of the uses of Speech-to-Text technology where we use one of the recent and most dynamic and popular categories of evolution is Artificial Intelligence so I am highlighting the objectives of the project:

- Open any website while speaking to your made assistant by giving the verbal command on your desktop screen.
- Provides speech-based information on any news which are highlighting and which you want.
- Capable of doing any sort of calculations and speaking out the outcome.
- Respond orally to inquiries.
- Use speech to inform the user of the time and date.
- Use speech to inform the user of the city's weather.
- Send emails to the recipient you want by vocal.
- Launch the different programs and applications of the desktop.
- Select the desired videos from YouTube.
- Go to Wikipedia and search for information on that specific subject.
- Play music on your command.[9]

There are many more things that have been achieved by this technology which can't be listed in one paper like everyone wants to have an extra smart AI voice which can do any types of work for you for picking up the calls to handling your business, that one which look each and every necessary actions or things for you in every possible time in every aspect of life. So, this thing is now obtainable because

of this Speech-to-Text technology.[10]

5. PROGRAM SETUP AND INTEGRATION

In the development of any speech-To-Text technology and environment we are required to adopt and use one of any programming language and different types of algorithms of machine learning that helps to achieve the outcome of our desired work. So, in this paper we are highlighting the use of python language in the making of one of the famous Speech-To-Text technology Desktop Assistant where it offers a versatile and powerful programming environment and features for setting and automating our tasks.

So, we use Natural Language Processing(NLP) machine learning algorithm to set an artificial Intelligence that means intelligence used by machines rather than humans and for that it requires a large dataset. NLP contains 4 components that are ASR(Automatic Speech Recognition, Natural language understanding, Natural Language Generation, Text-to-Speech.

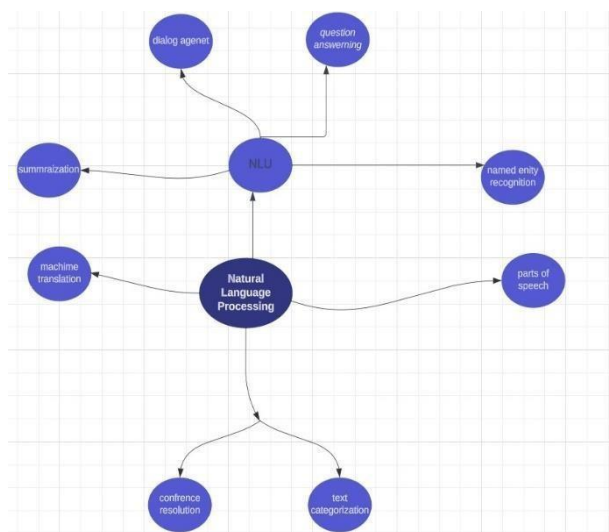


Fig. 2

All of the speech signals are translated into the appropriate word or string of words using the ASR model. The vocabulary is growing broader and broader everyday [6]. A single word is easier to recognize than a speech that is spoken continuously. The rate of speech errors can be influenced by an individual's accent [5]. According to evaluations, the error rate in Japanese and Spanish-accented English is three to four times higher than that of English spoken normally [7].

The most crucial stage after using ASR to convert speech to text is to comprehend what the text means. Ambiguity and variability are NLU's problems [8].

Natural language Generation (NLG) is the process of verbalizing the procedure. It is the creation of a language that is somewhat logical and human-like [9]. What to say

and how to say are the two phases that make up the NLG process [10].

Another name for text-to-speech is speech synthesis. This completes the process of creating voice assistants.

Waveform synthesis and text synthesis are the two steps in this process. Following the translation of words into speech, a desired sentence is created using waveform synthesis, which is based on previously recorded speech samples [5].

The four layers that make up our Desktop Voice Assistant are as follows:

1. Text to speech.
2. Analysis of Text.
3. Know instructions.
4. Speech to text.

Deep learning automatically learns complex features from unprocessed audio data, improving speech-to-text (STT) in desktop assistants. It makes it possible to create end-to-end models that can efficiently match audio to transcriptions and capture temporal connections, such as recurrent neural networks (RNNs) with attention mechanisms. For processing variable-length sequences, methods like Connectionist Temporal Classification (CTC) are useful. Accuracy and robustness are further enhanced via hybrid models, data augmentation, and transfer learning. The flexibility of deep learning facilitates ongoing learning, which is essential for changing user interactions. All things considered, it transforms STT by providing complex language models, opportunities for multimodal integration, and enhanced performance under various acoustic circumstances.

6. PROCEDURE AND PROGRAMS USED

In the development of this technology; we use a set of procedure and accordingly we move towards it while all the applications are run under the upcoming set of defined procedure. All the python programming and the different libraries used in this to achieve our objective are highlighted accordingly to our set of defines procedures:

FIRST STEP (Speech to Text):

This is a software that translates spoken words into written language.

- We employed Speech Recognition for this.
- It cannot comprehend every word you speak.

SECOND STEP (TEXT ANALYZING):

Text that has been converted is just letters for computers.

- Text is converted by software so that it can be understood by computers.
- The computer interprets this text as a command since it can understand commands.
- A computer command is made up of functions and their corresponding arguments.

THIRD STEP (INTERPRET COMMANDS):

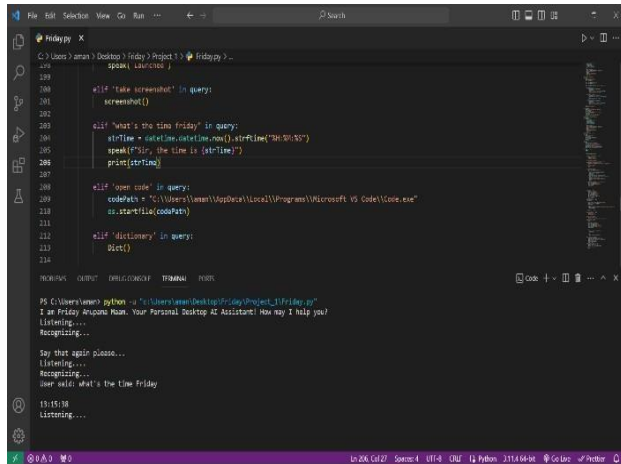
- Not much happens at this level.
- Through the internet, the mapped computer commands are sent to the server.
- The speech is assessed locally concurrently.

FORTH STEP (TEXT TO SPEECH LAYER):

- This is completed at the desktop level.
- The desktop virtual assistant translates text to voice and provides.

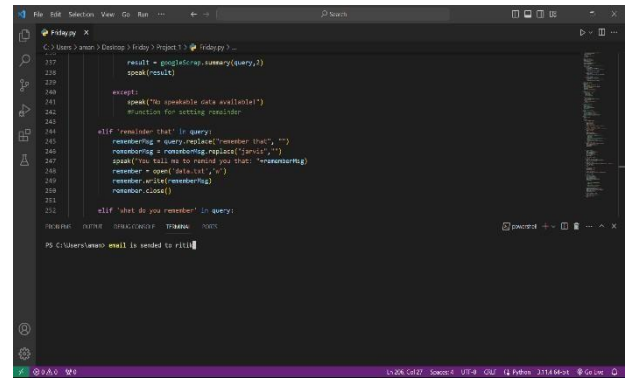
The main libraries which are necessary for the development of this projects and highlighted in the various papers are used to make this project working if we are using the python as a development tools.

| Library | Functionality | Installation |
|---------------------------|--|-----------------------------------|
| Speech Recognition | Captures audio from the microphone and provides speech recognition functionality. | pip install Speech Recognition |
| Pytsx3 | Implements text-to-speech functionality, allowing the assistant to respond audibly. | pip install pytsx3 |
| Wikipedia | Enables the assistant to search and provide summaries from Wikipedia based on user queries. | pip install Wikipedia |
| Web Browser | Allows the assistant to open specified websites based on user requests. | Built-in (No installation needed) |
| datetime | Facilitates the retrieval of the current date and time for personalized greetings. | Built-in (No installation needed) |
| NumPy | Useful for numerical operations and array manipulations. | pip install NumPy |
| NLTK | Provides tools for working with human language data, including tokenization and other NLP tasks. | pip install nltk |



```

199         speak('Launched')
200     elif 'take screenshot' in query:
201         screenshot()
202     elif 'what's the time Friday' in query:
203         strTime = datetime.datetime.now().strftime("%H:%M:%S")
204         speak(f"Sir, the time is {strTime}")
205         print(strTime)
206     elif 'open code' in query:
207         codePath = "%\\Users\\%user\\AppData\\Local\\Program\\Microsoft VS Code\\code.exe"
208         os.startfile(codePath)
209     elif 'dictionary' in query:
210         dict()
211
212 #PROGRAM OUTPUT
213 PS C:\Users\%user> python -u "C:\Users\%user\Desktop\Friday\Project 1\Friday.py"
214 I am Friday Anyname Huan. Your Personal Desktop AI Assistant! How may I help you?
215 Listening...
216 Recognizing...
217 Say that again please...
218 Listening...
219 Recognizing...
220 User said: what's the time Friday
221 13:15:18
222 Listening...
  
```



```

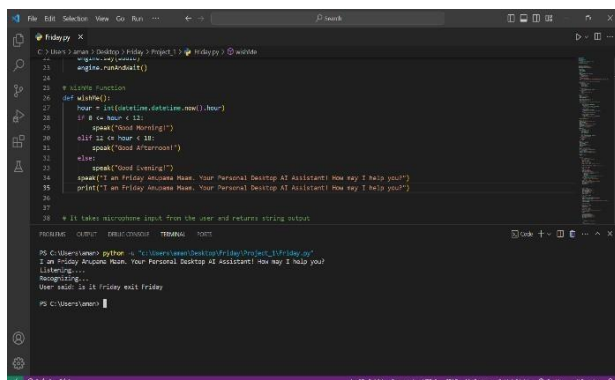
197         result = googleSearch.summary(query,2)
198         speak(result)
199     except:
200         speak("No readable data available!")
201         print("Error for getting answer")
202     elif 'remember that' in query:
203         remember = query.replace("remember that", "")
204         remember = remember.replace(":", "")
205         email = open("data.txt", "w")
206         remember = open("data.txt", "w")
207         remember.write(remember)
208         remember.close()
209     elif 'what do you remember' in query:
210         #PROGRAM OUTPUT
211         PS C:\Users\%user> email is send to rishi
  
```

These libraries cover a range of functionalities required for speech recognition, text-to-speech, web interaction, natural language processing and more in the development of speech-to-text desktop assistant.

7. RESULTS AND ANALYSIS:

GREETINGS:

The system's voice assistant will first ask for your name before saying "hello, your name." After that, extend greetings of "Good Morning," "Good Afternoon," and "Good Evening," depending on the time of day, and ask, "What can I do for you?".



```

23         engine.runAndWait()
24
25 # Greeting Function
26 def wishMe():
27     hour = int(datetime.datetime.now().hour)
28     if 0 <= hour < 12:
29         speak("Good Morning!")
30     elif 12 <= hour < 18:
31         speak("Good Afternoon!")
32     else:
33         speak("Good Evening!")
34
35 #speak("I am Friday Anyname Huan. Your Personal Desktop AI Assistant! How may I help you!")
36 print("I am Friday Anyname Huan. Your Personal Desktop AI Assistant! How may I help you!")
37
38 # It takes microphone input from the user and returns string output
39
40 #PROGRAM OUTPUT
41 PS C:\Users\%user> python -u "C:\Users\%user\Desktop\Friday\Project 1\Friday.py"
42 I am Friday Anyname Huan. Your Personal Desktop AI Assistant! How may I help you?
43 Listening...
44 Recognizing...
45 User said: hi Friday what's Friday
46 PS C:\Users\%user>
  
```

TELLS CURRENT DATE AND TIME

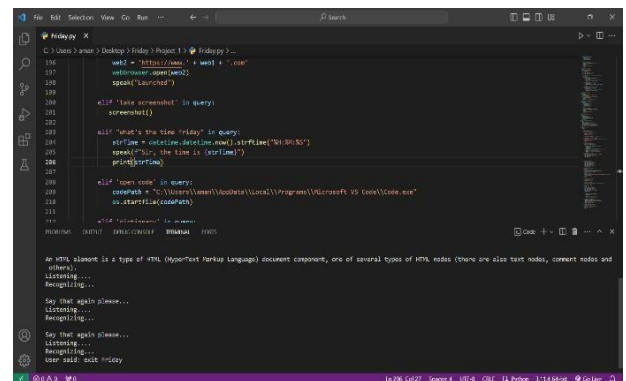
After greetings it will automatically tell you the time if you ask like What's the time, then it will respond with the current time and date also.

they provide it, application asks, "To whom you want to send mail, please type email id for security reasons." The user is then required to write the email address. Next, the assistant asks, "Send email to this mail yes/no." If the answer is "Yes," it then asks for the subject of the email and the body of the message before responding with "Email sent."

SEND EMAIL

You may send mail with this application. When a user says, "I want to send email," application asks for their password in the form of "Tell your email password." Once SEARCH ANYTHING ON WEB

If you want to search anything on google and on the YouTube, it has the feature like you say "open google, YouTube then you will have that platform with acknowledgement.



```

196         web = "https://www." + web + ".com"
197         webbrowser.open(web)
198         speak('Launched')
199     elif 'take screenshot' in query:
200         screenshot()
201     elif 'what's the time Friday' in query:
202         strTime = datetime.datetime.now().strftime("%H:%M:%S")
203         speak(f"Sir, the time is {strTime}")
204         print(strTime)
205     elif 'open code' in query:
206         codePath = "%\\Users\\%user\\AppData\\Local\\Program\\Microsoft VS Code\\code.exe"
207         os.startfile(codePath)
208     elif 'dictionary' in query:
209         #PROGRAM OUTPUT
210         PS C:\Users\%user> email is send to rishi
  
```

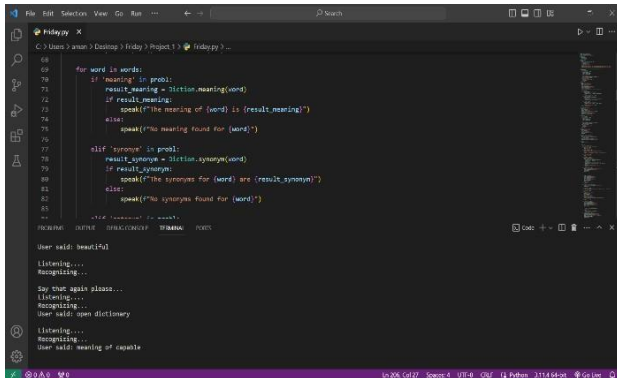
REMOTE ACCESS THROUGH PHONE

If you having a problem in voice recognition while speaking to any speech to the machine so we came across through a solution in which we control our voice command with our phone to reduce the latency in that all the command or speech you spoke to your phone that directly access you machine where you want to in build this feature.

TELLS THE MEANING OF ANYWORD

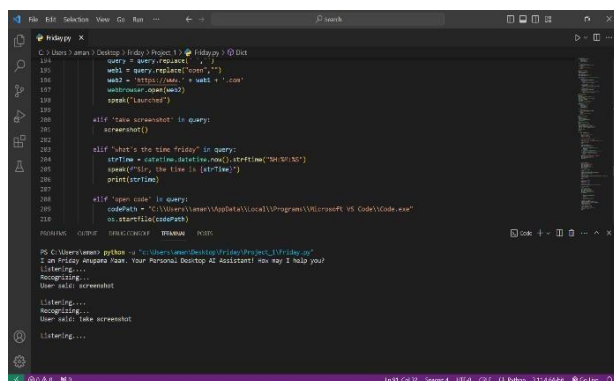
If you want to know the meaning of any words then by giving the command of "open dictionary" then it

proceed and ask the query “Tell me the problem!” after that you ask your word which you want to know then it will give the meaning of that particular word.



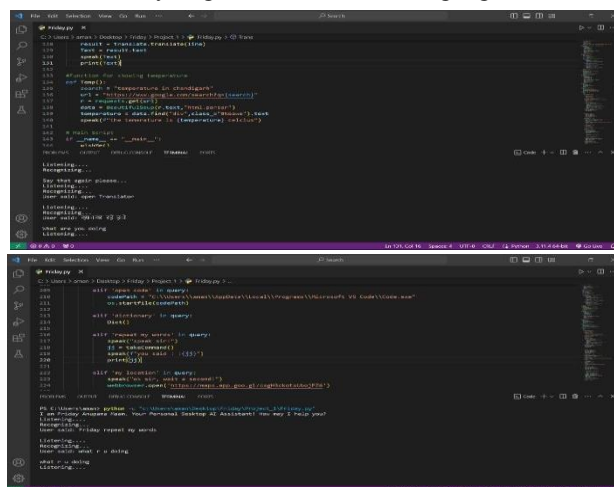
TAKE SCREENSHOT BY GIVING THE COMMAND

You will also have the feature to access the screenshot part by giving the command “Take Screenshot” then after that it ask about the file where you want to save that screenshot and in that particular path you have that screenshot.



LANGUAGE CONVERTER

You may convert language from this application. When a user said “Open Translator” and after that it goes into that function of that translator in which it starts listening the user preferred sentence that are executed to change and after that you got that translated language.

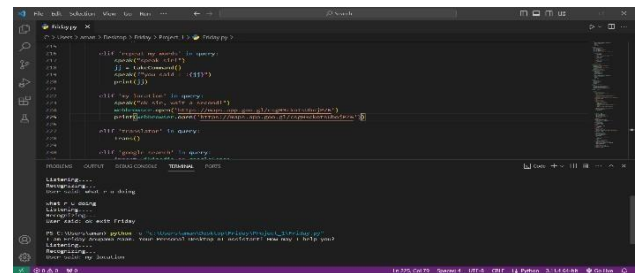


REPEATING THE WORD

In this application you came across the feature of repeating of the sentences when user enter a command of “repeat my words” then it repeats according to the flow of the data.

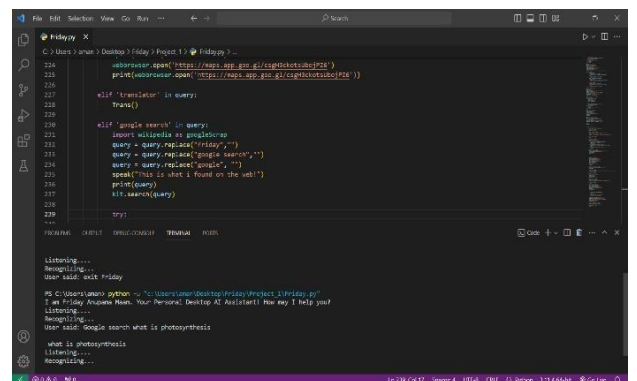
LOCATION TRACER

In this application we have the feature of location tracer in that we can trace the location of any person by just saying the query “my location” after that it give your current location.



SCRAPPING GOOGLE SEARCH

You may scrap the data with the help of the query “google search” and after that what do you want to search speak that then you will get the interface of the google where your search result is called and scrapping is also done like it automatically speaks the result which comes on the screen.



REMAINDER

You may have the feature in which you set the remainder by speaking the query “remainder that” after that the sentence or things that you want to remember it reminds you by saying the query “You tell me to remind you that”.

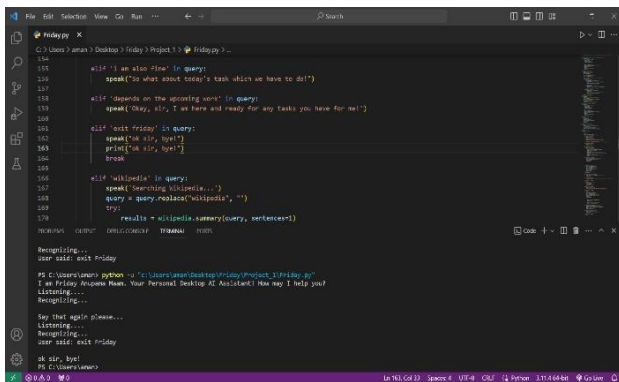


LEAVE

Say anything like "Bye," "Exit," or "Sleep" to end the software, and it will respond with "Okay, bye, your name." Enjoy your day!

We need to use artificial intelligence and natural language processing to build voice assistants before we can have smart assistants.

This software is intended to manage the system, including opening all desktop applications, gathering data from nearly anything on the Internet through Web Scraping, executing mathematical operations, sending emails, obtaining meteorological data. People with busyschedules and the deaf and dumb can both use our application.



```
134 #if 'I am also fine' in query:
135     speak("So what about today's task which we have to do?")
136
137 #if 'depends on the upcoming work' in query:
138     speak("Okay, sir, I am here and ready for any tasks you have for me!")
139
140 #if 'exit Friday' in query:
141     speak("Oh sir, bye!")
142     print("Oh sir, bye!")
143     break
144
145 #if 'wikipedia' in query:
146     speak("Searching Wikipedia...")
147     query = query.replace('wikipedia', "")
148     results = wikipedia.summary(query, sentences=1)
149     print(results)
```

7. CHALLENGES AND FUTURE CONSIDERATIONS

Through our overall analysis we came to know that this project is working very good in terms of all the related methodology and objectives, we also came across some of the challenges or issues like:

- **Handling Noise:**

Challenge: Accurate speech recognition in noisy settings, such as offices, is difficult.

- **Recognizing the Background:**

Challenge: Because spoken language has so much context, understanding what someone is saying can be difficult.

- **Managing Various Accents:**

Challenge: People talk in a variety of accents, and our systems need to be able to comprehend everyone.

- **Responding Quickly:**

Challenge: In Realtime interactions in particular, we want our computers to respond to speech rapidly.

Challenge: When our systems handle your voice, we need to make sure your information stays private and secure.

- **Mixing Different Ways of Communicating:**

Challenge: People use not just voice but also gestures and another ways to communicate. We need to make it all work together smoothly.

By tackling or improving these types of challenges, we aimed or have that Speech-to-Text systems that are easing our tasks.

8. FUTURE SCOPE

Emotion Recognition:

- **Future Touch:** STT systems might soon understand and respond to our emotions, making interactions more empathetic and context-aware.

Multilingual Support:

Language Expansion: Expect STT systems to speak and understand even more languages, making communication more inclusive and accessible globally.

Context-Aware Conversations:

- **Natural Flow:** Future STT systems could grasp ongoing conversations better, responding in a way that feels more like talking to a person.

Customizable Personalization:

- **Tailored Interaction:** Look forward to STT assistants that you can customize to match your unique way of speaking and preferences.

Real-Time Translation:

- **Breaking Language Barriers:** STT systems might soon translate languages in real-time, making global communication smoother and more natural.

Advanced Security:

- **Privacy First:** Expect tighter security measures, with STT systems implementing even stronger encryption to safeguard your private information.

Collaboration with AR/VR:

- **Immersive Experiences:** STT technology could team up with Augmented and Virtual Reality, creating more engaging and hands-free interactions.

Continuous Learning:

- **Getting Smarter:** STT systems will keep learning and adapting, staying sharp and accurate as language evolves over time.

Connection of Chatbot:

- **Innovation:** STT systems will connect with the conversational artificial intelligence model developed by OpenAI like ChatGPT and Google Bards. It can basically convert the conversation into speech by using Speech-to-Text methods.

ACKNOWLEDGMENT

We would like to thank the IEEE Society for giving me the chance to write this article.

REFERENCES

- [1] Voice Assistant Application, Deny Nancy, Anushri Sai, M. Ganga, R.S. Abisree, Sumithra Praveen, college website.
- [2] Abhay Dekate, Chaitanya Kulkarni, and Rohan Killedar's study of a voice-controlled personal assistant
- [3] At ITD.Y. Patil College of Engineering and Technology, Dr. Kshama V. Kulhalli is the Dean.
- [4] Android Voice Assistant with Intelligence: A Need for the Future Ms. Krina, Ms. Yesha, Ms. Ayushi, and CSE Madhuben, Department, and Bhanubhai Patel
Women's Institute of Technology
- [5] Martin, J. H., and D. Jurafsky (2009). Chapter 13: Syntactic Interpretation. An Overview of Natural Language Processing through Speech and Language Processing. Speech recognition and computational linguistics (2nd ed.). Prentice Hall, Pearson.
- (6) Dutta, A., Sen, S., and Dey, N. (2019). Speech Processing and Recognition System. In Audio Processing and Speech Recognition (pp. 13-43). Springer, Singapore.
- [7] Tomokiyo, L. M. (2001). Recognizing non-native speech: characterizing and adapting to non-native usage in LVCSR. Unpublished doctoral dissertation, Carnegie Mellon University, Pittsburgh, PA.
- [8] Karshabi, D. (2019) [8]. Natural Language Understanding through Reasoning-Based Question Answering. preprint arXiv:1908.04926 arXiv.
- [9] Shaikh, S., and Santhanam, S. (2019). An overview of natural language generation methods with a particular emphasis on dialogue systems—their history, current state, and future prospects. The preprint arXiv is arXiv:1906.00500.
- 10] Speech and language processing, Daniel Jurafsky and James H. Martin, draft third edition, 2018.