International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 10 | Oct - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

Desktop Based Voice Assistant

Prem Panchal[1] , Parth Barot[2] , Arpit Patel[3] , Asst. Prof. Ms. Manisha Vasava [4]

[123]*Research Scholar, Institute of Information Technology, Krishna School Of Emerging Technology & Applied Research,* KPGU University ,*Varnama, Vadodara, Gujarat, India*

[4] *Assistant Professor, Department of Information Technology and Engineering, Krishna School Of Emerging Technology & Applied Research,* KPGU University. *Varnama, Vadodara, Gujarat, India*

Abstract -The advancement of voice-based technologies has led to the development of user-friendly systems that integrate voice recognition with task automation. This paper presents a desktop-based voice assistant capable of interpreting voice commands, processing them through Python, and executing various tasks autonomously. By leveraging speech-to-text conversion, the assistant can understand user input, engage in contextual task execution, and provide real-time outputs, enhancing productivity and accessibility. Unlike many commercial voice assistants, this solution is tailored for local desktop environments, ensuring data privacy and allowing for greater customization. The paper reviews the architecture, key modules such as Haar Cascades, LBPH, DNN and the underlying Python algorithms that power the assistant, along with its real-world applications.

Key Words: Haar Cascades, LBPH, DNN

1.INTRODUCTION

The Python-developed voice assistant offers an intelligent and interactive system that responds to voice commands while prioritizing user security. To ensure secure access, the project integrates advanced facial recognition technology, making it versatile for various desktop applications.

User authentication is achieved using the OpenCV (cv2) library, which employs several algorithms for reliable facial recognition. Haar Cascades is utilized for real-time face detection, while the LBPH (Local Binary Patterns Histograms) algorithm identifies facial patterns by comparing grayscale values against a stored database of authorized users. Additionally, for more advanced face detection, the system can leverage DNN (Deep Neural Networks) with pre-trained models such as SSD (Single Shot Multibox Detector) or YOLO (You Only Look Once).

Once authenticated, users can interact with the voice assistant through various methods. The first method involves clicking a microphone button to issue commands. Alternatively, users can speak a hot word detected by the pvporcupine library's keyword spotting algorithm. The assistant also accepts commands typed directly into a chat interface using the Eel library. Lastly, users can activate the assistant with the Windows+J hotkey combination for quick access.

The voice assistant is capable of executing a wide range of tasks, utilizing specific Python libraries for different functionalities. The pyautogui and subprocess libraries are employed for opening desktop applications, while the webbrowser library allows the assistant to access websites seamlessly. To enable messaging and calling features, the pywhatkit library is utilized. Furthermore, for natural language processing and conversational tasks, the assistant integrates with hugchat, which uses transformer-based language models like GPT.

For effective audio processing and voice feedback, the assistant relies on the pyaudio library to capture voice input from the microphone. The playsound library is used to provide audio feedback, allowing the assistant to respond to user commands audibly.

User data and preferences are securely stored using the sqlite3 library, which facilitates a local database to manage authentication data and command history. This allows for a personalized experience based on the user's past interactions, enhancing usability and convenience.

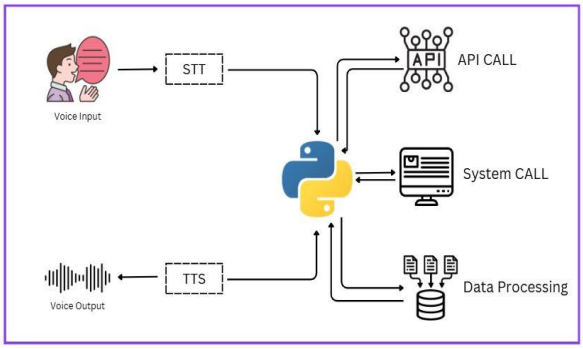
Lastly, the assistant incorporates image and video processing capabilities through additional libraries. The Pillow (PIL) library is employed for basic image processing tasks, while numpy is used for handling arrays and matrices, especially during facial recognition. Together, these technologies create a powerful, secure, and user-friendly voice assistant designed to enhance productivity through hands-free operation.

2 Literature Summary

Sr No	Paper Author/ Title/Year Of Publication	Method	Data set	Limitations	Future Scope
1	Jash Vora1, Deepak Yadav2, Ronak Jain3 & Jaya Gupta4 JARVIS: A PC Voice Assistant [July 2021]	Hidden Markov Model (HMM), Regex	Wolfram Alpha API , Wikipedia API	limited to predefined tasks like weather information and Wikipedia searches	Voice assistant retrieves weather, plays music, reads news.
2	Vishal Bisht Desktop Voice Assistant System with the Help of Natural Language Processing [2022]	Google Cloud's API for Natural Language Processing (NLP) , pyttsx3 , speech_recognition	real-time user input (voice commands)	better device integration and a more sophisticated graphical user interface (GUI)	Interoperability and Integration , Removal of Wake Words , Enhanced User Interface
3	Vedant Titarmare, Dr. Pankaj H. Chandankhede, Dr. Minakshi Wanjari Zira Voice Assistant- A Personalized Interactive Desktop Application [2022]	pyttsx3 , speech_recognition , webbrowser	Zira operates based on real-time user inputs	designed for low-end devices which limits its functionality compared to commercial voice assistants like Siri.	device compatibility, adding more sophisticated AI capabilities
4	Satyam Singh, Aryan Mahale, Ishan Kumar Enhancing Desktop Productivity: An Approach of Virtual Assistance for Desktop Workflow [2023]	pyttsx3, speech_recognition, Tkinter for GUI creation	A MySQL database stores user information, primarily for user login and authentication purposes	The system's accuracy can be affected by environmental noise and voice clarity	Future development could focus on improving speech recognition accuracy
5	Rajdip Paul1, Nirmalya Mukhopadhyay2 A Novel Python-based Voice Assistance System for reducing the Hardware Dependency of Modern Age Physical Servers [May 2021]	pyttsx3, smtplib, speech_recognition	system processes real-time voice inputs	lacking machine learning capabilities and advanced natural language processing (NLP)	machine learning for better decision-making, improving NLP for multi-language support

6	Aaditya Bhardwaj1, Deepankar Pratap Singh2, Hritik Sahu3, Udaysingh Kushwaha Automating Desktop Tasks with a Voice-Controlled AI Assistant using Python [May 2024]	pyttsx3 , speech_recognition	it processes real-time voice commands	Privacy concerns about data handling , improving the accuracy of voice recognition	improving security features, user personalization , integrating more APIs for greater functionality
7	Dr. N.Thirupathi Rao, M.V.Sushriya, M. Navya Manaswi, .V.N.S.Aishwarya,P Vikas, P Surya Teja Voice Based Virtual Personal Assistant [2024]	Natural Language Processing (NLP) , Text-to-Speech (TTS) , Action execution , Knowledge base and information retrieval	speech data from users for voice recognition and response generation	Limited Functionality , Privacy Concerns , Language Barriers	Incorporating New Capabilities , Enhancing Current Functions , Diverse Usage Cases
8	Asadullah Shaikh1, Sharada Patil2 ORCA Desktop and Web Assistant [2021]	pyttsx3 , PyPDF2	system relies on live user inputs through voice commands	The assistant requires registration and accounts for certain services like Google Assistant.	Improving automation for complex and repetitive tasks involved in managing servers
9	Siddhish Nirgude, Vandana Jagtap, Pallavi Parlewar Revolutionizing Interaction: A Desktop Voice Assistant Powered by Artificial Intelligence [2024]	Pyttsx3 , Speech-to-Text	Uses audio data (speech samples) converted into spectrograms for processing.	The assistant is mainly made for Windows users, Accuracy depends on the quality of speech recognition	Incorporating support for more languages , expanding capabilities to handle complex tasks like ticket booking
10	Hardik Muley1, Jaydeep Ghosh2, Ankit Lal Sinha3, Prof. Padmavati Sarode 4 SKYE : Voice Based AI Desktop Assistant [2023]	pyttsx3 , OS , PyWhatKit	it uses real-time voice commands from the user	Designed primarily for desktop applications with limited mobile support	Plans to introduce support for multiple languages , interact with web content for visually impaired users

3. Materials and Methods



The following materials were used in the development of the desktop-based voice assistant:

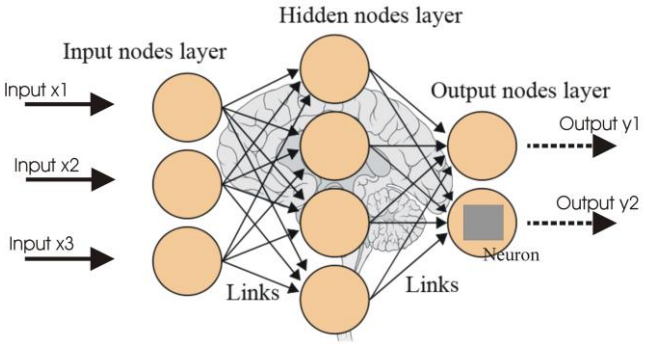
- **Programming Language**: Python 3.x
- **Development Environment**: Anaconda or Jupyter Notebook (for experimentation and initial development)
- **Libraries**:
- **OpenCV (cv2)**: For facial recognition and image processing tasks.
- **pvporcupine**: For implementing wake word detection via keyword spotting.
- **Eel**: For creating the web-based user interface that connects the backend with the frontend.
- **pyautogui**: For automating GUI interactions and controlling mouse and keyboard actions.
- **subprocess**: For launching desktop applications and performing system tasks.
- **webbrowser**: For opening web pages in the default browser.
- **pywhatkit**: For sending messages and making calls through WhatsApp.
- **hugchat**: To integrate AI capabilities with conversational tasks using transformer-based language models.
- **pyaudio**: For capturing voice input from the microphone.
- **playsound**: For providing audio feedback to users.
- **sqlite3**: For storing user data and preferences in a local database.
- **Pillow (PIL)**: For basic image processing.
- **numpy**: For handling numerical data and arrays during image processing.

Methods

3.1 User Authentication via Facial Recognition

The voice assistant employs facial recognition technology to authenticate users before granting access. The process begins by capturing live video feed from the webcam using the OpenCV library.

- **Face Detection**: The Haar Cascades algorithm is utilized for real-time face detection. This algorithm identifies and locates faces within the video frames by analyzing the features of each detected face.
- **Facial Recognition**: After detecting a face, the system uses the LBPH (Local Binary Patterns Histograms) algorithm to analyze and compare facial patterns against a pre-stored database of authorized users. This method ensures reliable and secure authentication.
- **Advanced Detection**: For enhanced detection capabilities, the system can implement DNN (Deep Neural Networks) techniques using pre-trained models such as SSD (Single Shot Multibox Detector) or YOLO (You Only Look Once). These models improve the accuracy of face detection in various lighting and background conditions.



3.2 Voice Assistant Operation

Once user authentication is confirmed, the assistant can be activated and controlled through various methods:

- **Microphone Button**: Users can click a designated microphone button on the interface to issue voice commands. The assistant then processes the audio input in real-time.
- **Hot Word Detection**: The system employs the pvporcupine library for continuous listening for a specific hot word (e.g., "Hey Assistant"). Upon detecting the hot word, the assistant activates

voice recognition to interpret subsequent commands.

- **Typing Commands**: Users have the option to type their commands into a chat interface developed using the Eel library. This provides an alternative method for interacting with the assistant without requiring voice input.
- **Keyboard Activation**: The assistant can also be triggered by pressing the Windows+J key combination, enabling quick access to its features for users who prefer keyboard shortcuts.

3.3 Task Execution

Upon receiving a command, the assistant processes the request and executes various tasks through the following methods:

- **Opening Applications**: The assistant can launch desktop applications such as web browsers, media players, and text editors using the pyautogui and subprocess libraries. These libraries automate GUI interactions by performing mouse clicks and keyboard actions as instructed.
- **Web Navigation**: To open websites, the webbrowser library is utilized. The assistant seamlessly opens specified URLs in the user's default web browser, enhancing hands-free browsing capabilities.
- **WhatsApp Messaging and Calls**: The pywhatkit library facilitates automated messaging and calling features, allowing users to send WhatsApp messages or initiate calls using voice commands.
- **AI Integration**: For natural language processing tasks, the assistant employs the hugchat library, which integrates transformer-based language models like GPT. This allows the assistant to interpret user queries and provide intelligent, contextually relevant responses.

3.4 Data Storage and Management

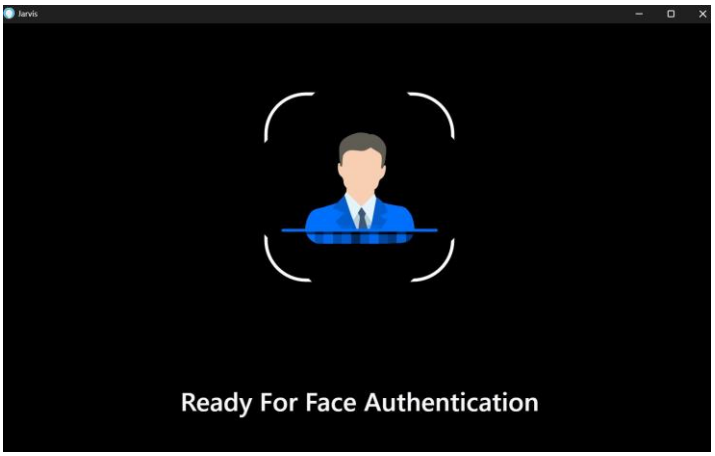
User data, preferences, and command history are managed using the sqlite3 library, facilitating a local database that personalizes the user experience based on previous interactions.

3.5 GUI Interaction and Image Processing

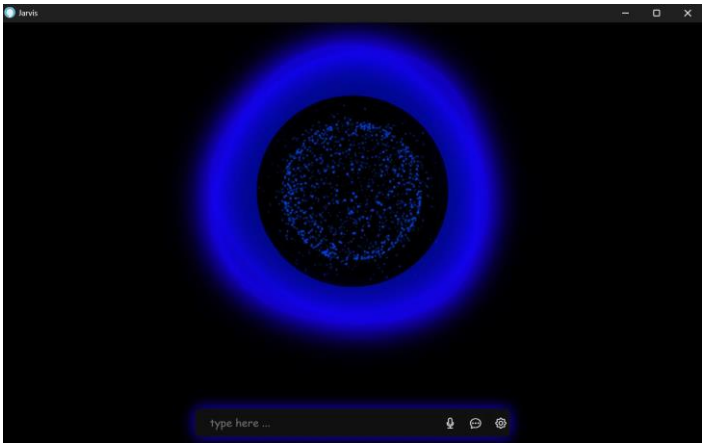
The graphical user interface is developed using the Eel library, allowing for smooth interaction between the frontend and backend. Image processing tasks, such as resizing and converting images for facial recognition, are accomplished using the Pillow library, while numpy aids in handling numerical data and arrays throughout the processing stages.

4. Experimental Results

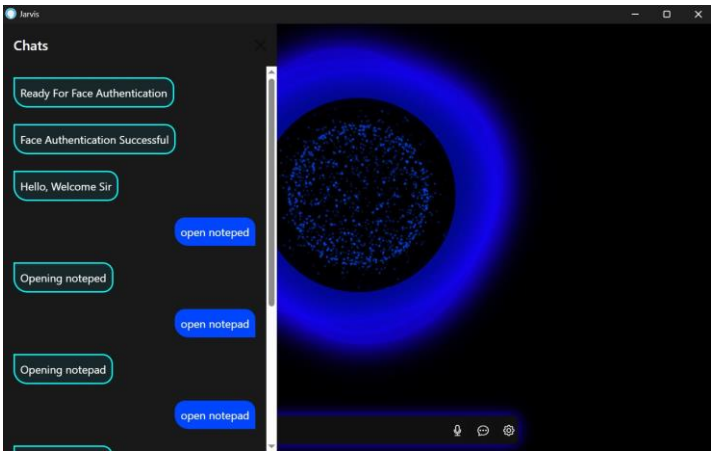
Face Detection :-



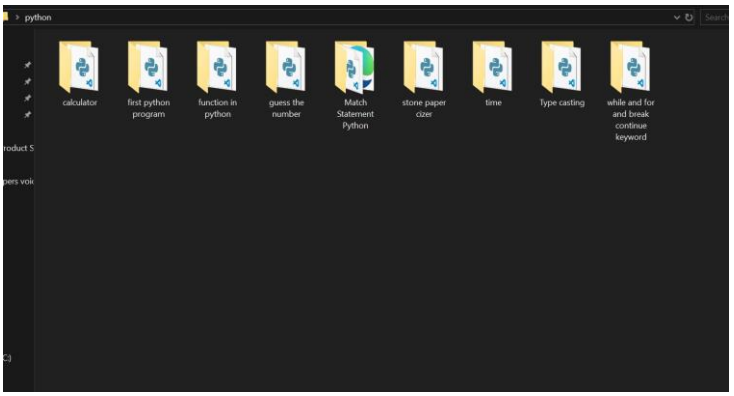
GUI :-



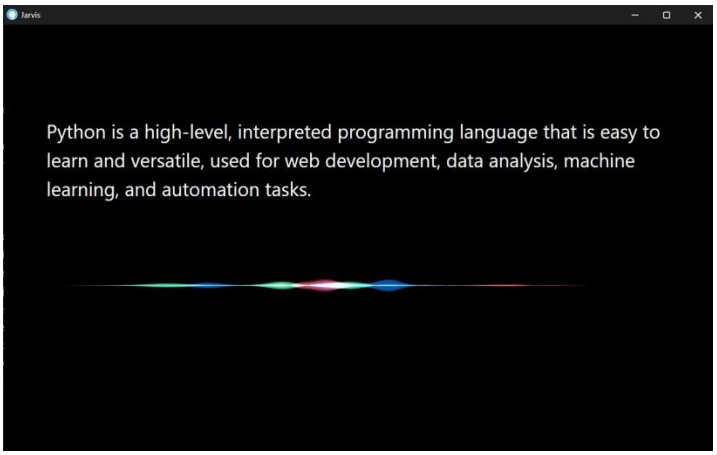
Chat History :-



Open Files And Folders :-



Response from hugchat :-



Open Web Applications :-

Confusion Matrix

Task/ Command	Correct Response	Incorrect Response	No Response
Open Notepad	9	1	0
Open Youtube	10	0	0
Send Message to [“Person name”]	7	2	1
Play [Video Name] on youtube	8	1	1
Make Video Call to [“Person Name”]	9	1	0

Performance Metrics:

Accuracy: 86%
Precision: 89.6%
Recall: 95.6%
F1-Score: 92.4%

5. CONCLUSIONS

In conclusion, the developed desktop-based voice assistant demonstrates a significant advancement in user interaction and security through its integration of facial recognition and diverse command methods. By utilizing advanced libraries and algorithms, the assistant effectively automates various desktop tasks, enhancing

productivity and providing a user-friendly experience. Its capability to process natural language queries through AI integration further enriches interactions, making it a versatile tool for everyday use. Overall, this project not only showcases the potential of combining voice processing with robust security features but also sets the foundation for future enhancements, making it a valuable addition to desktop applications.

REFERENCES

1 Jash Vora1, Deepak Yadav2, Ronak Jain3 & Jaya Gupta4
JARVIS: A PC Voice Assistant
[July 2021]

2 Vishal Bisht Desktop Voice Assistant System with the Help of Natural Language Processing [2022]

3 Vedant Titarmare, Dr. Pankaj H. Chandankhede, Dr. Minakshi Wanjari Zira Voice Assistant- A Personalized Interactive Desktop Application [2022]

4 Satyam Singh, Aryan Mahale, Ishan Kumar Enhancing Desktop Productivity: An Approach of Virtual Assistance for Desktop Workflow [2023]

5 Rajdip Paul1, Nirmalya Mukhopadhyay2 A Novel Python-based Voice Assistance System for reducing the Hardware Dependency of Modern Age Physical Servers [May 2021]

6 Aaditya Bhardwaj1, Deepankar Pratap Singh2, Hritik Sahu3, Udaysingh Kushwaha Automating Desktop Tasks with a Voice-Controlled AI Assistant using Python [May 2024]

7 Dr. N.Thirupathi Rao, M.V.Sushriya, M. Navya Manaswi, .V.N.S.Aishwarya,P Vikas, P Surya Teja Voice Based Virtual Personal Assistant [2024]

8 Asadullah Shaikh1, Sharada Patil2 ORCA Desktop and Web Assistant [2021]

9 Siddhish Nirgude, Vandana Jagtap, Pallavi Parlewar Revolutionizing Interaction: A Desktop Voice Assistant Powered by Artificial Intelligence [2024]

10 Hardik Muley1, Jaydeep Ghosh2, Ankit Lal Sinha3, Prof. Padmavati Sarode 4 SKYE : Voice Based AI Desktop Assistant [2023]