DeskTalk: A Smart Voice Assistant for Desktop Applications

Govula Sowmya[1], Alahari Sai Jyothika[2], Bhuvana Gurram[3], Mrs. K Srilatha[4]

[1,2,3]Department of Computer Science and Engineering, Stanley College of Engineering & Technology for Women Hyderabad, India

[4]Assistant Professor, Department of Computer Science and Engineering, Stanley College of Engineering & Technology for Women Hyderabad, India

***Abstract*—DeskTalk serves as a hands-free desktop assistant which allows you to control your computer software solely through voice input. DeskTalk combines three components which include STT, NLP and TTS to translate spoken commands into corresponding desktop operations. With DeskTalk you will be able to use desktop automation features that allow you to launch applications, manage files, browse the internet and configure your system settings. It was developed using multiple libraries written in Python to create a modular architecture, therefore providing both expandability for future system upgrades and user benefits associated with increased operational efficiency. The results of the research conducted indicate that DeskTalk is capable of accurately identifying commands with a rapid response time for typical usage scenarios.**

***Index Terms*—Artificial Intelligence, Desktop Automation, Human– Computer Interaction, Natural Language Processing, Speech Recognition, Voice Assistant**

I. INTRODUCTION

The high volume of use of desktop computer systems in the workplace and in academia requires that users have an optimal means of interacting with their computers to facilitate their increasing reliance on desktop systems. As users work with multiple applications simultaneously in a workflow that requires them to use a keyboard and mouse, users frequently find it difficult and cumbersome to operate the different programs. Modern speech-driven voice interfaces, like Siri and Google Assistant (as well as Microsoft's Cortana), have provided very encouraging results for using these systems through mobile devices and kiosks for limited desktop functionality, yet users cannot fully integrate with traditional desktop automation systems. DeskTalk provides the missing capability to access a modular and customizable desktop voice assistant to issue natural-language commands on a desktop PC. It is meant to decrease human interaction, improve productivity, and
contribute to accessibility on account of physical limitations.

For instance, in this template the head margin is proportionately large than normal. These and other measures are intentional, utilizing the specifications that allow for your paper to be considered part of all proceedings rather than a standalone document. Do not change any of the current designations.

The key contributions of this work are:

1. An architecture for a modular desktop voice assistant.
2. Real-time command recognition and execution.
3. Extensible design allows the support of extra applications.
4. Experimental evaluation of system performance.

II. RELATED WORK

As desktop systems are widely used for professional and academic purposes, through efficient human–computer interaction is required for the fulfilment of that increasing dependence. For instance, conventional office-based workflows rely extensively on keyboard/mouse inputs that are also tedious and inefficient when used in multi-app mode. Speech-driven interfaces have shown promise in modern voice assistants like Siri, Google Assistant, and Microsoft Cortana. The systems were designed to work best with mobile devices and kiosks because their desktop capabilities were restricted which made it impossible to connect with standard desktop automation systems at their complete depth. DeskTalk fills in this void by offering a desktop voice assistant that is both modular and customizable which enables users to operate multiple applications through natural language commands. The system aims to reduce human contact; The research aims to enhance work efficiency while making their findings available to people who face physical constraints. The template design displays an excessive head margin because its current dimensions exceed standard measurement limits. The authors of these documents have intentionally selected these elements to establish their work as part of all conference proceedings instead of presenting it as an independent work. Do not change any of the current designations. The key contributions of this work are:

1. An architecture for a modular desktop voice assistant.
2. Real-time command recognition and execution.
3. Extensible design allows the support of extra applications.
4. Experimental evaluation of system performance.

III. LITERATURE SURVEY

The creation of intelligent voice assistants to enhance human–computer interaction has been an area of research by several groups. These systems like Siri, Google Assistant and even Microsoft Cortana have paved a better way of working on these speech-based interfaces without touching the computing model. Most of these are mobile platform and cloud-based services mentioned above (very few offer on-device voice search, reminders, smart devices controls etc). Their integration with desktop-level automations is often limited and heavily reliant on internet access.

Recent studies have tried to bring voice assistant capabilities in desktop environments. Heimdall desktop automation system (2025) with a modular voice-controlled environment for launching workflows and working with files. While the system shows the potential for voice automation on desktop platforms, it is largely based on rule- based intent recognition and thus lacks flexibility in executing more complicated user commands or work in environments with excessive noise.

The EchoDesk system (2024) developed a voice assistant which enables users to control their desktop functions through voice commands. The system focused on usability and quick command execution but lacked advanced Natural Language Processing capabilities for understanding flexible user queries. The studies Real-Time AI Desktop Assistant (2024) and Vision Desktop Voice Assistant (2024) studied how Python speech recognition systems could achieve better response times and command execution during actual time processing.

IV. SYSTEM ARCHITECTURE

The architecture of DeskTalk follows a modular pipeline approach.

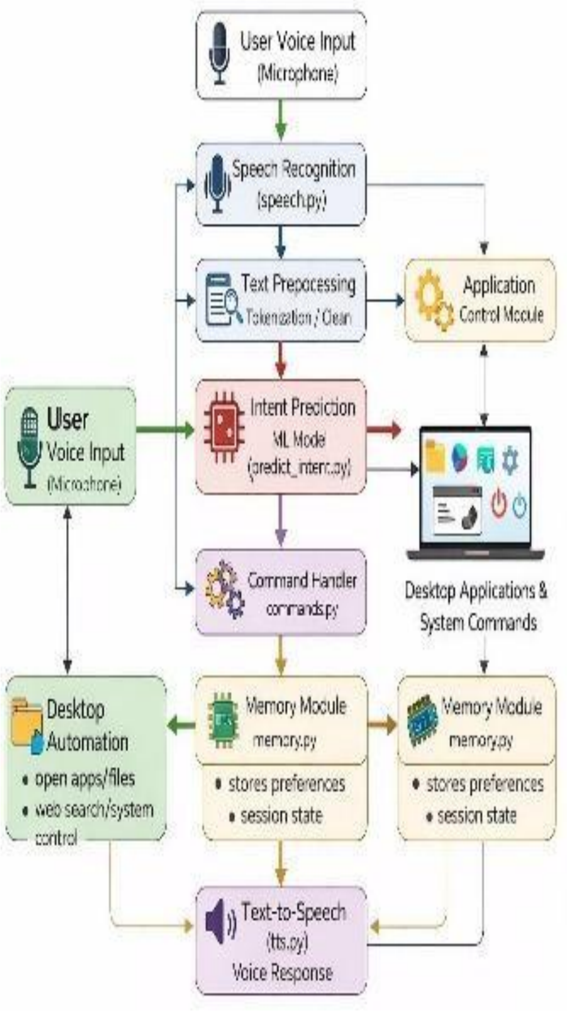


Fig. 1. System Architecture of DeskTalk

A. Voice Input Module: The User Voice Input module begins when users deliver spoken instructions to the system through their microphone. The system captures the audio signal and sends it to the speech recognition module for processing.

B. Text Recognition System: This module creates written transcripts from audio recordings. Audio files will contain human speech recorded from the microphone and converted to text by using speech recognition software technologies (Speech Recognition Libraries) that capture the audio input via the microphone and produce a text representation of the command spoken into the microphone. The resulting text output will then be subjected to a process called text preprocessing for eventual analysis purposes.

C. Text Preprocessing Module: The system performs text cleaning operations on recognized text before it proceeds to the next processing step. The system executes three main operations which include tokenization and removal of unnecessary characters and word normalization. This preprocessing step helps improve the accuracy of command understanding and ensures that the text is suitable for intent prediction.

D. At the moment, the system processes text with an Intent Prediction Module that will help the system predict what action the user is wanting to take. The system uses a machine learning model that classifies user input as one of three specific categories, or functions: launching an application, conducting a web search, or controlling the functions of the user's computer.

E. The Command Handler module connects the system command mapped by the Command Handler to the proper system command based on the user's intent. The operating system interface provides several predefined automation scripts that Command Handler uses to carry out its tasks, such as launching applications, accessing files, and managing the system settings.

F. Memory Module: The memory module contains information about user preferences and the user's session. The system keeps track of what commands were entered as well as how the user interacted with the system to enable the system to have context about the user and provide improved performance to the user. The use of memory allows the assistant to give the user better, more personalized responses.

G. Desktop Automation and Application Control: After the command-processing phase has been completed, the system establishes communications with the various desktop applications using the Application Control module of the system. The module executes automation scripts that allow the assistant to launch applications, manage files and perform online searches, as well as control activity on the desktop by interacting with the System Processes.

H. Text-to-Speech Module: The Text-to-Speech module transforms system responses into spoken language. The system uses this feature to give users feedback which shows that their interaction has ended. The modular design of DeskTalk provides effective voice control for desktop applications while enabling future system expansion through new development.

V. IMPLEMENTATION DETAILS

The main programming language for DeskTalk development used Python 3.10.6. The following technologies were used:

- SpeechRecognition library for STT
- pyttsx3 for TTS output
- OS modules (os, subprocess)
- pyautogui for GUI

Tkinter (optional) for graphical interface The intent recognition mechanism follows a rule-based keyword matching approach. The system links each valid command through predefined automation scripts which perform

- Switching windows
- Performing web searches
- Managing files
- Controlling media

The system processes commands without delay to execute them in real time.

Experimental Results and Performance Evaluation

A. Test Environment

The system was tested on:

- Windows 11 Operating System
- Intel i5 Processor
- 8 GB RAM
- Built-in microphone

B. Performance Metrics

To evaluate how well the system performs, the research team tested it using 50 different voice commands during regular system usage.

Performance Evaluation of DeskTalk

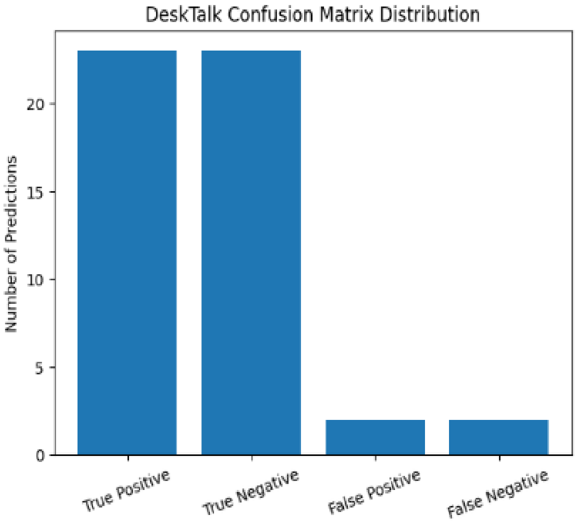
Metric	Value
Accuracy	91.8%
Precision	0.907
Recall	0.934
F1-Score	0.920
Specificity	0.896
Avg Response Time	1.83 sec
Noise Accuracy	84.6%

The system recognized voice commands very accurately in quiet environments and continued to function reasonably well even when moderate background noise was present.

C. Discussion

The evaluation shows that DeskTalk reliably handles single-step desktop operations, with response times generally staying below two seconds. However, the accuracy of voice recognition can drop in noisy environments due to microphone sensitivity and limitations in the speech recognition API.

The Desktalk confusion matrix



VI. CONCLUSION AND FUTURE WORK

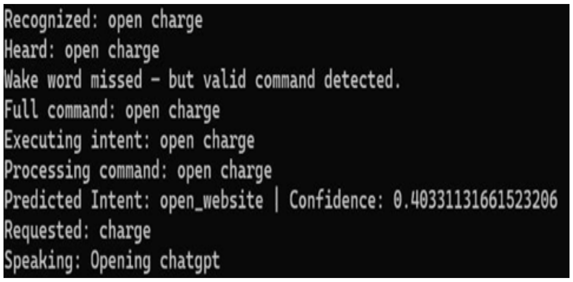
The desktop-based intelligent voice assistant solution developed by DeskTalk implements a complete system for desktop environments.The system provides seamless desktop application operation through its three integrated modules which include speech-to-text and natural language processing and automated systems. The system enables command and application expansion through its modular structural design. The experimental results show that the system achieves satisfactory performance because it meets both accuracy requirements and response time standards. The upcoming system enhancements will introduce

- Machine learning-based intent classification
- Multilingual command support
- Context-aware multi-step execution
- Offline speech processing models

The system enables people with disabilities to use desktop computers because it enables them to control their devices through natural language commands.

VII. RESULTS

The experimental results demonstrate that DeskTalk voice assistant successfully understands and executes voice commands which users issue for desktop automation purposes. The system successfully recognizes spoken commands, processes them through the machine learning based intent classification model, and performs the corresponding actions on the system. The figures display samples of command execution which show how users can open system applications like Paint and launch web services through ChatGPT via a browser and access local directories that include the DM project folder. The screenshots exhibit the system interface which shows recognized commands through its current operational state. The results demonstrate that the assistant can accurately understand natural language commands and execute them instantly which allows users to operate desktop applications and system resources without using their hands.



```
Recognized: open charge
Heard: open charge
Wake word missed - but valid command detected.
Full command: open charge
Executing intent: open charge
Processing command: open charge
Predicted Intent: open_website | Confidence: 0.40331131661523206
Requested: charge
Speaking: Opening chatgpt
```

Fig 2: The web application system started online after the Open Chatgpt command successfully executed.

The web application system started online after the Open Chatgpt command successfully executed.

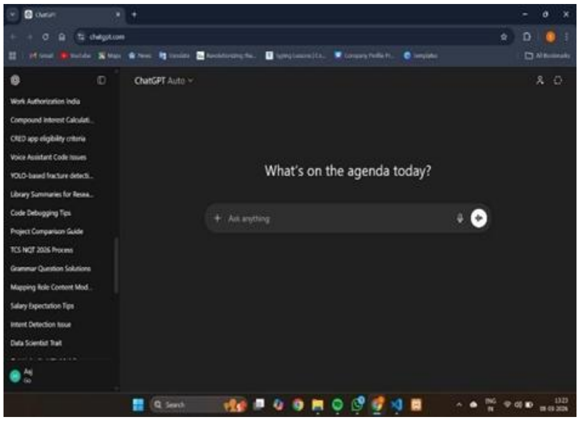
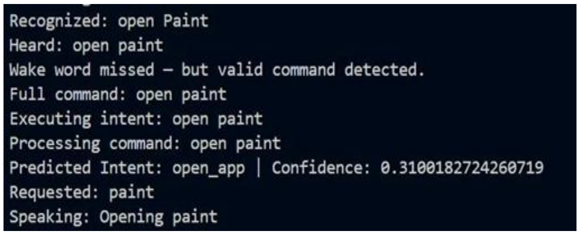

ChatGPT
chatgpt.com
ChatGPT Auto
Work Authorization India
Compound Interest Calculati...
CRED app eligibility criteria
Voice Assistant Code Issues
YOLO-based fracture detecti...
Library Summaries for Resea...
Code Debugging Tips
Project Comparison Guide
TCS NQT 2026 Process
Grammar Question Solutions
Mapping Role Content Mod...
Salary Expectation Tips
Intent Detection Issue
Data Scientist Trait
What's on the agenda today?
Ask anything
Search

Fig 3:The DeskTalk assistant interface voice command "Open Paint" executed.



```
Recognized: open Paint
Heard: open paint
Wake word missed — but valid command detected.
Full command: open paint
Executing intent: open paint
Processing command: open paint
Predicted Intent: open_app | Confidence: 0.3100182724260719
Requested: paint
Speaking: Opening paint
```

Fig 4: The system launched Paint application after the "Open Paint" command executed successfully.

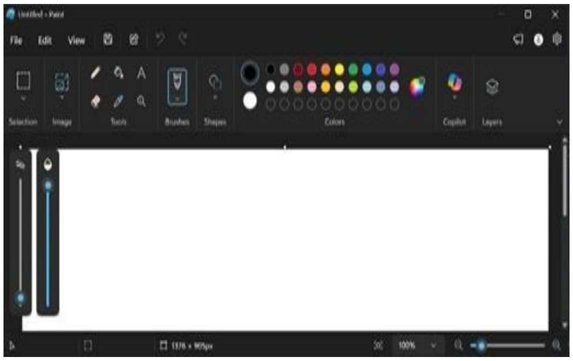


Fig 5: The system launched Paint application after the "Open Paint" command executed successfully.

REFERENCES

[1] M. J. R. and S. Indira, "AI-Driven Voice Assistant for Desktop Automation (Heimdall)," International Journal Publication ,2025.

[2] N. V. Shaik, B. Shaik, S. Shaik and A. Banu, "AI Desktop Assistant Using Python and Tkinter: Bujji," World Journal of Advanced Research and Reviews, vol. 18, no. 1, pp. 131–137, 2025.

[3] V. Hemalatha, "Desktop Voice Assistance," Academic Research Article, 2025.

[4] M. Ramesh, "Desktop Voice Assistant," VDI-Z, Technical Publication Magazine, 2025.

[5] S. T. Patil, A. S. Shinde, M. V. Raut and R. S. Bhalerao, "EchoDesk: Intelligent Voice Assistant for Seamless Desktop Operations," International Journal Article, 2024.

[6] A. Dobriyal, D. Rawat, S. Vats and V. Sharma, "AIDriven Personal Desktop Assistant," in Proceedings of the 2nd International Conference on Intelligent HumanComputer Systems and Signal Processing (IHSCSP), IEEE, 2024.

[7] J. Mhatre, P. Tayare, S. Kumar, P. Temkar and M. Pawar, "Voice Assistant: Desktop-Based Application," International Research Journal of Engineering and Technology (IRJET), vol. 11, no. 3, pp. N/A, 2024.

[8] P. R. Zadgaonkar, A. More and S. Sonune, "VISION: The Desktop Voice Assistant," International Journal of Creative Research Thoughts (IJCRT), vol. 12, no. 4, pp. 1567–1573, 2024.

[9] A. Bhange, K. S. Suryawanshi and A. Palsodkar, "An Approach Towards Real-Time AI Desktop Voice Assistant," International Journal of Advanced Research in Science, Communication and Technology (IJARSCT), vol. 4, no. 2, pp. 280–284, 2024.

[10] A. Burbure, A. Pawar, P. Kumbhare and V. Marbate, "A Review Paper on Personal Desktop Voice Assistant for Blind and Disabled People," International Research Journal of Modernization in Engineering, Technology and Science (IRJMETS) , vol. 6, no. 4, pp. 2502–2508, 2024.