BRUCE: A Next Generation Voice Assistant for Enhanced User Interaction and Task Execution

Ms. Maryam Farooqui[1], Mrs. Novera[2], N. Vinay Yadav[3], GM. Vamshi[4], Jashwanth[5], Manish[6]

[1] Assistant Professor of Department of AIML, J.B. Institute of Engineering and Technology, Hyderabad
[2] Associate Professor of Department of AIML, J.B. Institute of Engineering and Technology, Hyderabad
[3] Department of AIML, J.B. Institute of Engineering and Technology, Hyderabad
[4] Department of AIML, J.B. Institute of Engineering and Technology, Hyderabad
[5] Department of AIML, J.B. Institute of Engineering and Technology, Hyderabad
[6] Department of AIML, J.B. Institute of Engineering and Technology, Hyderabad

---***---

Abstract -
The demand for voice-activated technologies has been increasing rapidly since the early 2000's particularly for personalized assistant systems, this highlights the need for solutions that are tailored for different user needs. Bruce-A desktop voice Assistant addresses this challenge by offering two distinct modes a basic mode for simple tasks and an Advanced mode for complex tasks and interactions, Bruce is designed to be easy to use for people of all ages. With a simple toggle, users can switch between basic and advanced modes depending on what they need. The basic mode handles simple tasks like managing files, playing music, and setting reminders, all while using very little system power and working offline. The advanced mode takes things a step further, offering smart language processing that helps Bruce hold conversations and respond appropriately to what's happening. The toggle feature makes it simple for users to change modes as needed, so that Bruce can fit different preferences and levels of interaction.

Key Words: Desktop Assistant, Automatic Speech Recognition, OpenAI, Natural Language Processing (NLP), Python, API Integration, Offline Capability.

1. INTRODUCTION

The global shift towards sustainable transportation has gained momentum in recent years, driven by the urgent need to reduce greenhouse gas emissions and combat climate change. Among the various solutions being explored, electric city buses have emerged as a promising alternative to traditional diesel-powered vehicles, offering significant environmental benefits and operational efficiencies. However, the successful integration of electric buses into urban transit systems requires a comprehensive understanding of their energy consumption patterns and operational dynamics.

As cities continue to expand and evolve, the complexity of urban transportation systems increases, necessitating innovative approaches to energy management. Traditional methods of estimating energy needs often fall short in capturing the multifaceted interactions between various factors such as route characteristics, passenger demand, and environmental conditions. This is where machine learning (ML) comes into play, providing powerful tools to analyze large datasets and uncover hidden patterns that can inform decision-making.

In this context, we will explore various machine learning algorithms, assessing their effectiveness in predicting energy consumption and identifying key factors that influence operational performance. Additionally, we will evaluate the role of real-time data analytics in optimizing scheduling, battery usage, and overall fleet efficiency. The insights gained from this research will contribute to the development of a data-driven energy economy, ultimately supporting the transition to cleaner, more efficient public transportation systems.

Through this work, we aim to provide a valuable resource for policymakers, transit agencies, and researchers committed to advancing sustainable urban mobility while ensuring cost-effective and reliable transit solutions.

2. LITERATURE REVIEW

To start off the development of voice assistants has been on boom for almost a decade. With the advancements in AI, NLP, and ML, there are numerous assistants like Siri for IOS, Google Assistant for Android, and Alexa as a personal day-to-day smart home assistant. And also, for desktop there are assistants like Microsoft Cortana. However, these mainstream voice assistants mostly operate in cloud environments, which raises concern about data privacy, and it requires internet connectivity even for a basic command like what's the time; basically, they have very limited offline functionality. Based on our research, the insights that we got about existing voice assistants, there are 3 types of assistants Cloud-Based Assistants, Offline Voice Assistants, Python-Based-Assistants/LLM-powered-assistants, and the Hybrid Approaches. The description and their gaps are: *2.1 Cloud-Based Assistants:*
Most of the popular voice assistants use cloud computing for functioning. For example, Google Assistant and Siri take the

International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 03 | March - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

user voice record and send it to remote servers using the internet, and the NLP algorithm interprets the user input and returns the suitable output. This method delivers highly accurate results due to the computational ability of cloud servers. However complete dependency on external servers can cause delays, server issues, and mainly privacy concerns as voice data will be stored and processed by third parties. For users worried about privacy or with poor connectivity, this approach may not be a good fit.

2.2 Offline Voice Assistants:

For the above limitations several offline assistants like Mycroft are developed. In this system, the processing, storing, and functioning are done locally without internet. However, the tasks and functioning are of a basic level, and they lack advanced capabilities like cloud-based.

2.3 Python-Based-Assistants/LLM-powered-assistants:

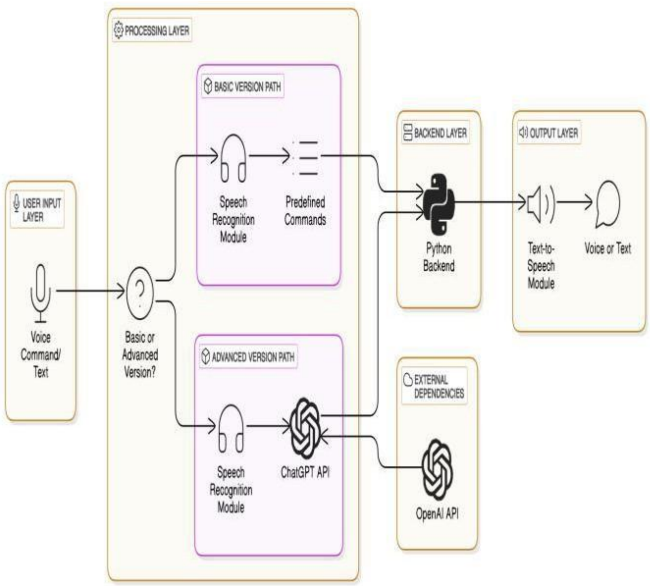
The example for this is Jarvis and Dragonfly, they provide desktop control and task automation through voice commands. Mainly using prebuilt Python libraries and API's of LLM models. The problem with them is they may have API downtime sometimes.

2.4 Hybrid Approaches:

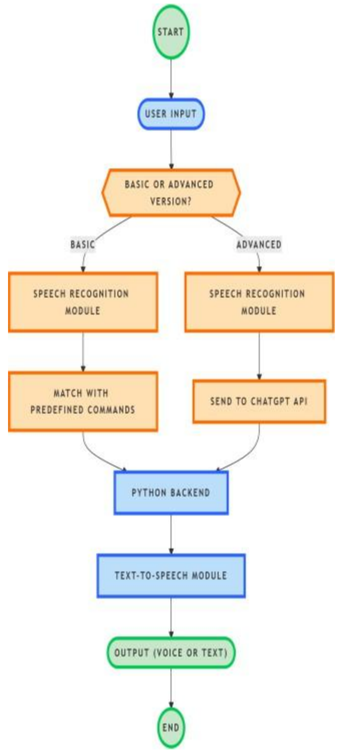
This is the latest approach overcomes the gap between offline and cloud-based systems, hybrid-type assistants are developed that use local processing for simple commands and cloud computing for advanced queries. It is more secure than pure cloud-based, but for advanced queries it uses cloud-based resources, Bruce follows this hybrid approach.

2.5 Gaps in Existing Solutions:

Despite the advancements in voice assistant technologies the major gaps in current voice assistants are privacy concerns, internet dependency, limited local capabilities, restricted flexibility.



3. METHODOLOGY



3.1 DEVELOPMENT:

The development process of Bruce was planned thoughtfully ensuring it was suitable to work for the needs of all the age groups. The process is divided into 3 key phases: Development, Implementation, and Evaluation. Python programming language is used to build Bruce due to its powerful in-built libraries which serve as a base for voice processing, Natural language processing (NLP), and API integration. Key libraries like Speech-Recognition, Pyaudio, pyttsx3 are used to recognize speech and convert text to speech. The Advanced version integrates OpenAI (ChatGPT) API for conversational responses while the basic version uses a rule-based command parser for task execution.

3.2 IMPLEMENTATION:

To ensure a seamless user experience the system is designed with modularity in mind, where core components like speech Recognition, NLP, task execution, text-to-speech work independently but they also exchange data through well-defined APIs. The idea of implementation is to offer two versions which cater different user needs:

3.2.1 Basic Version:

The basic version can operate offline and utilizes only minimal resources. It uses a predefined dictionary to recognize the commands and to execute them like file management, reminders, music playback, time & date, shutting down PC etc.

3.2.2 Advanced Version:

The advanced version provides dynamic interactions and conversational capabilities. It processes user queries using OpenAI's GPT-4 API to generate context-aware responses and resolves more complex queries through API integrations.

4. RESULTS

The evolution of Bruce is focused upon its performance across different metrics such as speech recognition accuracy, task execution efficiency, response time and user Satisfaction. The test is conducted with diverse age groups across different professions.

4.1 Speech Recognition Accuracy:

The Basic version got 92.5% accuracy for predefined commands showcasing its ability to work with limited resources. And on the other side advanced version which is powered by GPT-4 has secured 97.3% accuracy for conversational queries.

4.2 Task Execution Efficiency:

The Task completion in the basic version took an average of 1.8 seconds per task, while the Advanced Version averaged 2.3 seconds due to network dependencies.

4.3 Response Time Analysis:

The Basic version responded within 2 seconds for 90% of queries while the advanced version maintained around 2.5 seconds response time for internet-reliant tasks. (this may change according to processor and internet speed)

4.4 Usability Feedback:

The Bruce was accessed by 50+ students and got a rating of 4.7 out of 5. Where the basic version was praised for its simplicity and offline functionality, And the advanced version was impressed with its dynamic interactions.

4.5 Comparative Evaluation:

The Basic version is optimized for resource efficiency and offline use, suitable for carrying out straightforward tasks with limited internet connectivity. The Advanced version offers NLP capabilities and contextual assistance ideal for complex user needs. These results confirm Bruce's ability to offer a flexible, efficient, and user-friendly desktop voice assistant adaptable to diverse user needs.

5. CONCLUSION

Bruce-The Desktop Voice Assistant successfully fills the gap between simplicity and complexity by offering two operating modes: a basic mode optimized to work under limited internet and resources and an advanced version that is powered by NLP for conversational and context-aware responses. With modular design, a seamless toggle function, and a focus on user-centered adaptability makes Bruce cater to all age groups with varied technological proficiency. The above evaluation results demonstrate its efficiency, accuracy, and usability showcasing its potential as a versatile voice assistant.

For future work, several enhancements are envisioned to further elevate Bruce's Capabilities like:

5.1 Machine Learning Integration:

Training and creating Custom models, especially for speech recognition to increase speech recognition accuracy over the offline environment.

5.2 Multilingual Support:

Adding a broad range of languages to improve Bruce's accessibility for non-English speakers. So that Bruce can cater to users of different languages.

5.3 Integrating with IoT Devices:

To expand Bruce's capabilities to work with more devices like home appliances.

5.4 Accessibility Features:

Adding visual aid options and adaptive voice modulation to assist users with disabilities.

6. REFERENCES

1 Sayyed, A., Shaikh, A., Sancheti, A., Sangamnere, S., & Bhangale, J. H. (2021). Desktop Assistant AI Using Python. UG Scholar, Department of Mechanical Engineering, Matoshri College of Engineering and Research Centre, Eklahare, Nashik. SOURCE: https://ijarsct.co.in/Paper1567.pdf

2 Rabin, J., Kar, S., Wondimu Bamud, A., & T R, M. (2023). Personal A.I. Desktop Assistant. International Journal of Innovative Technology and Research Applications, 2(2), 54-60. ISSN: 2583 5343. DOI: 10.59461/ijitra.v2i2.58. SOURCE: https://www.ijitra.com/index.php/ijitra/issue/archive

3 OpenAI (2023). ChatGPT API Documentation. SOURCE: https://platform.openai.com/docs

4 Fard, A (2023). How to Use ChatGPT API. SOURCE: https://adamfard.com/blog/how-to-use-chatgpt-api

5 Patel, N., & Chandra, R (2023). Voice Recognition System for Desktop Assistant. ResearchGate. SOURCE: https://www.researchgate.net/publication/369895948_Voice_Recognition_System_for_Desktop_Assistant

6 Python TTS (2023). Implementing Google Text-to-Speech API in Python. SOURCE: https://pyttsx3.readthedocs.io/en/latest/

7 Nielsen Norman Group (2023). Toggle Switch Guidelines. SOURCE: https://www.nngroup.com/articles/toggle-switch-guidelines/