Speech-Driven Natural Language Understanding Framework Chatbot: Bridging Conversational AI and Data Analytics through Integrated Voice Processing

ChandrikaN[1], EgaJuhi Reddy[1], MohammedZain[1], MythiliAnil Sanji[1], SathishR[1]

Department of Artificial Intelligence and Machine Learning , The Oxford College of Engineering 10th Milestone, Bommanahalli, Hosur Road, Bangalore - 560 068.

Abstract

In many data-driven environments, people without technical backgrounds find it difficult to work with structured datasets because most existing tools require programming knowledge or familiarity with query languages. To address this problem, a voice-based system is developed that allows users to interact with data using spoken input instead of written queries. Users can ask questions verbally and receive results in both visual and audio formats. The system converts spoken input into executable Pandas commands that are applied to a preloaded dataset through a simple two-step process, where the required code is generated first and then executed. The application is implemented using a Flask backend with a lightweight frontend built using HTML and JavaScript, allowing interaction through a web browser. Initial testing indicates that the system supports hands-free data exploration and can be used effectively by individuals with little or no programming experience.

Key Terms - Natural Language Understanding, Speech-to-Text, Large Language Models, Text- to-Speech, Conversational Analytics, Voice-Based Data Processing, Pandas Automation, Flask Framework, Human–Computer Interaction, Data Accessibility, and Conversational AI.

1. INTRODUCTION

Large amounts of data are now generated in many organizations, which has created a need for analysis tools that can be used by different types of users. Even so, many data analysis methods still depend on writing code or using structured queries. This makes data analysis difficult for people who are not familiar with programming languages such as SQL or Python. Visual dashboards have helped users explore data to some extent, but performing detailed analysis often still requires technical knowledge or extra manual effort.

At the same time, improvements in speech processing and language-based systems have changed how people interact with software. Instead of using fixed commands, users can now communicate with systems using everyday language. This makes interaction easier, especially for users who do not have strong technical backgrounds.

Based on this idea, a speech-guided framework is considered that allows users to explore structured datasets through spoken input. Rather than typing commands or writing scripts, users can ask questions verbally, and the system interprets these inputs to perform the required analysis.

2. RELATED WORKS

Research on voice-based interaction has grown as speech recognition and language understanding technologies have improved. In early systems, voice input was mostly limited to short and predefined commands, which allowed users to perform only specific actions. As learning-based techniques developed, speech systems gradually became capable of handling longer inputs and identifying user intent, making interaction more natural. Because of this, voice interfaces began to appear in applications such as accessibility tools and digital assistants. However, most speech-driven systems are still focused on narrow tasks and are not well suited for analytical work on structured datasets.

There has also been work on text-based natural language interfaces for data exploration. Many of these systems allow users to ask questions in text form or translate their queries into SQL or other structured commands. Some approaches support basic statistical operations or generate simple visual results, while others rely on predefined rules or templates. Although these methods show that natural language can be useful for data analysis, they have practical limitations. In many cases, they struggle with unclear queries, multi-step analysis, or datasets that differ from those they were originally designed for. Template-based systems, in particular, tend to fail when user input does not match expected patterns.

Despite progress in both speech-based systems and natural language interfaces for data analysis, frameworks that combine real-time speech input with analytical computation remain limited. In many existing systems, speech transcription, intent processing, and execution are handled as separate stages, which can reduce how accurately user requests are translated into results. Only a small number of studies have explored generating analytical code directly from spoken input, and these approaches still face challenges related to performance and adaptability. This points to the need for more integrated approaches that can better support voice-driven data analysis.

3. PROPOSED SYSTEM

3.1. Problem Definition

Data analysis tasks, even simple ones, often require technical expertise, which means many users still depend on others for support. Most existing tools rely on coding or structured queries, making it harder for users to interact with data directly and slowing down the analysis process. To deal with this limitation, the system adopts a speech-based approach that enables users to access and explore structured datasets through spoken input, without requiring technical assistance.

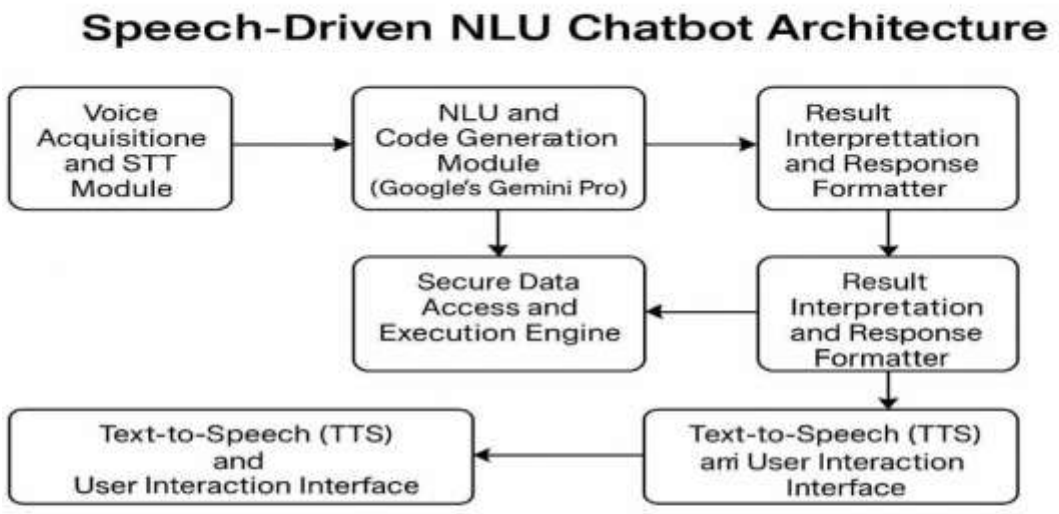

Fig 1: Layered Architecture Diagram

3.2. System Objectives

The primary objectives of the proposed framework include:

- Voice-Driven Interaction: Allow users to query datasets naturally using speech commands.
- Automated Code Generation: Translate these spoken queries into correct and executable Pandas operations.
- Human-Friendly Output: Present the outcomes in clear, easy-to-read explanations.
- Real-Time Responsiveness: Ensure immediate processing and feedback for user queries.
- Modularity: Maintain a flexible design to support future expansions, such as data visualization

or database integration.

3.3. Design Considerations

The system design incorporates several key considerations:

- User-Centric Design: Designed to be simple and accessible for users who do not have strong technical backgrounds.
- Security: Ensures that dynamically generated code executes in a safe environment, preventing unauthorized access or operations.
- Scalability: Supports datasets of varying sizes and can be extended for multi-user deployment.
- Extensibility: Modular architecture allows integration of additional features, such as multi-turn conversation, live database connectivity, or advanced analytics.

3.4. System Components

The proposed framework consists of the following modules:

1. Speech-to-Text (STT) Module: Converts spoken user queries into textual input for further processing.
2. Large Language Model (LLM) Module: Interprets natural language queries, generates executable Pandas code, and reformats output into clear natural language explanations.
3. Code Execution Engine: Safely executes the generated Pandas scripts on pre-loaded structured datasets.
4. Text-to-Speech (TTS) Module: Converts textual results into spoken audio responses for the user.
5. Flask Backend Server: Orchestrates the interaction between modules, manages API requests, and ensures real-time query processing.
6. Frontend Interface: Provides a simple HTML/JavaScript interface for capturing voice input and playing audio responses.

3.5. Functional Workflow

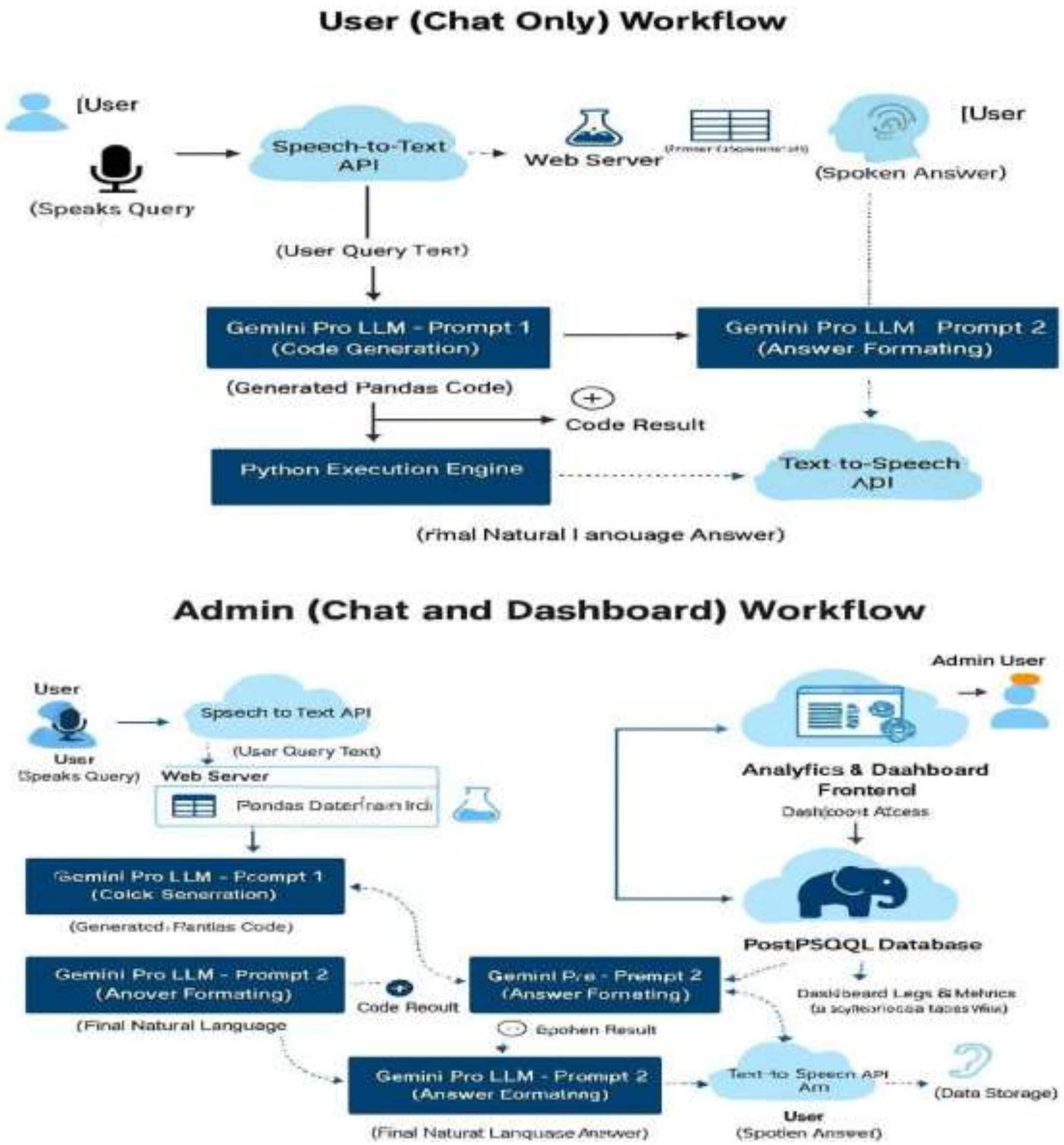

Fig 2: User and Admin Workflow Diagram

The system follows a sequential pipeline for processing user queries:

1. The user speaks a query into the microphone.
2. The STT module transcribes the speech into text.
3. The text is sent to the LLM, which generates corresponding Pandas code based on the dataset schema.
4. The execution engine runs the generated code securely on the dataset.
5. The LLM reformats the output into a human-readable explanation.
6. The TTS module converts the response into audio, completing the voice-based interaction loop.

3.6. Advantages

The proposed framework introduces several innovations:

- Hands-Free Interaction: Allows users to interact with structured datasets using voice commands, making data exploration more intuitive and convenient.
- Automated Analysis: Helps reduce human error by automatically generating and running the required analytical code.
- Natural Language Explanations: Makes it easier for non-technical users to understand the results by

explaining them in simple and clear language.

- Modular Architecture: Makes it possible to extend the system in the future with additional features such as advanced analytics, visualization, or database support.

4. SYSTEM ARCHITECTURE

The Speech-Driven NLU framework supports voice-based interaction with structured datasets through a set of interconnected components managed by a Flask backend. Instead of treating the system as a single unit, it is organized into six main modules that handle different tasks. These include capturing voice input, converting speech to text, interpreting user requests and generating code, executing the analysis, processing the results, and converting the output back into speech. Organizing the system in this manner makes it easier to manage and maintain, while also allowing new features to be added in the future if required.

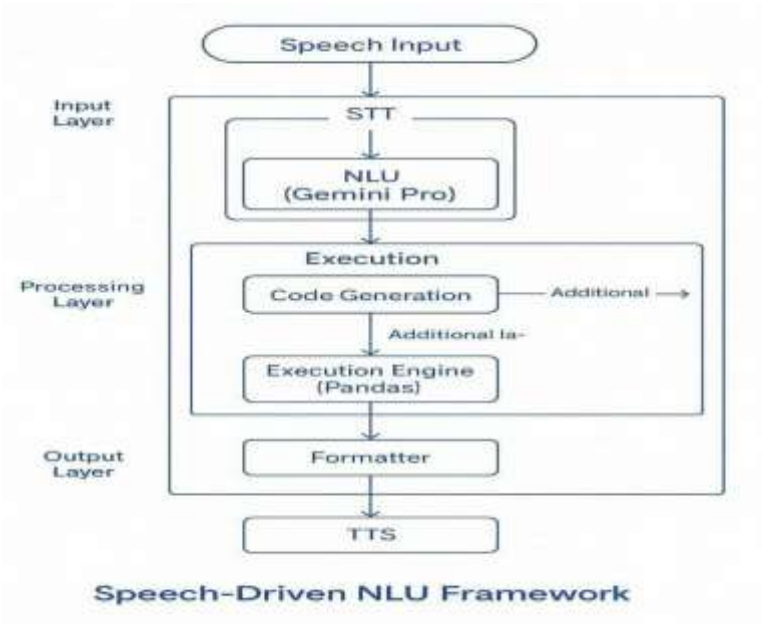

Fig 3: System Architecture Framework diagram

4.1. Voice Acquisition and Speech-to-Text (STT)

The first stage captures user queries through a microphone using the HTML/JavaScript frontend. This audio is then sent to the STT module, where it is converted into text using modern speech-recognition methods. Getting an accurate transcription at this stage is important, since any errors can affect how the query is interpreted and the code that is generated later.

4.2. Request Orchestrator and API Gateway

The backend provides a set of endpoints that handle user queries and route them to the appropriate system modules. It is responsible for validating incoming requests and managing any errors that occur during processing. By coordinating communication between the speech recognition component, the language model, the execution engine, and the text-to-speech module, the backend helps maintain a smooth and consistent flow of information back to the user.

4.3. Natural Language Understanding and Code Generation

The interpretation module uses a large language model to process the transcribed user query and convert it into executable Pandas instructions. To improve reliability, the framework follows a simple two-step approach.

- **Stage1: Code Generation**

In this stage, the model receives the user's query along with details about the dataset, including its structure and any predefined constraints. Based on this information, it generates a Pandas code snippet that is tailored to the dataset being used.

- **Stage2: Output Interpretation**

Once the generated code is executed, the resulting output is passed back to the model. The model then explains the result in clear and natural language so that users can understand it easily, even without prior knowledge of Python or data manipulation libraries.

Keeping the computation step separate from the explanation helps ensure that the generated code remains accurate while the final response is easy for users to follow.

4.4. Secure Execution Engine

The generated Pandas code is executed only within a restricted environment that operates on the preloaded DataFrame. The execution component handles operations such as filtering, aggregation, and value extraction while applying safeguards against invalid syntax or unsupported actions. This controlled setup ensures reliable system performance and protects the dataset from unintended modifications.

4.5. Result Interpretation and Natural Language Response

The output from the execution engine is formatted into a clear, user-friendly statement using the second LLM prompt (Prompt 2). This ensures that even non-technical users can understand the results without programming knowledge. Separating computation and explanation enhances both accuracy and fluency in the system's responses.

4.6. Text-to-Speech (TTS) and User Interaction

The final text output is converted into speech using the text-to-speech module and sent to the frontend through the Flask backend for playback. The text result is also displayed so users can view it if needed, allowing non-technical users to interact with data through voice in real time.

5. EXPERIMENTATION AND RESULTS

The experimental evaluation looks at how the Speech-Driven NLU Framework works in practice. The tests mainly check whether the system can correctly understand spoken queries, generate usable Pandas code, and process structured data without errors. Attention is also given to how clearly the results are explained through text and speech. In addition, the system is compared with manual coding methods to see how much it improves ease of use and reduces the effort required for data analysis.

5.1. System Login Interface

The framework incorporates a secure login system for both users and administrators:

- Authentication: Ensures controlled access to datasets and administrative functionalities.
- User Interface: Provides a minimalistic design to support intuitive navigation. Successful login confirms proper integration of the Flask backend, HTML interface, and session management, demonstrating that user authentication and session flows are correctly implemented.

5.2. Dataset Upload and Management

The system allows dynamic uploading of CSV datasets, which are immediately loaded into server-side Pandas DataFrames.

- Upon upload, the dataset schema is extracted and incorporated into the Gemini Pro prompts for code generation.
- This enables the framework to adapt to arbitrary dataset structures, supporting dataset- agnostic query execution.

Correct column display and dataset confirmation indicate that the backend successfully parses, validates, and stores datasets, ensuring seamless integration for real-time analytics.

5.3. Voice-Driven Query Execution

The framework's core functionality is voice-to-data query execution, validated through multiple test scenarios:

a) Speech-to-Text Conversion

- User queries spoken via microphone are transcribed into text with high accuracy.
- The STT module reliably converts natural speech to a machine-readable format, forming the basis for subsequent processing.

b) Natural Language to Pandas Code Generation

- Gemini Pro translates textual queries into executable Pandas code using a two-stage prompting approach:

1. Code Generation Prompt: Converts user intent into precise, single-line Pandas operations.
2. Answer Formatting Prompt: Converts computed results into human-readable explanations.

- Supported operations include filtering, aggregation, arithmetic calculations, and column extraction.
- Generated code strictly adheres to predefined constraints, ensuring safe and correct execution.

c) Code Execution and Output Formatting

- The system executes the generated Pandas code within a controlled environment.
- Raw results are processed by the second LLM prompt, producing concise natural- language explanations without revealing internal computation details.

d) Text-to-Speech Output

- Formatted results are converted into spoken responses via the TTS module, completing the voice-to-answer loop.

The successful execution demonstrates strong alignment between user intent, generated code, and spoken output, ensuring reliable, hands-free interaction with structured datasets.

5.4. Admin Dashboard

The administrative interface provides monitoring and management capabilities:

- Dataset Overview: Allows viewing and management of uploaded datasets.
- User and Query Logs: Tracks user interactions and query history.
- System Monitoring: Presents execution logs and error reports for administrators.

Fig 4: Login page

Fig 5: Loading dataset

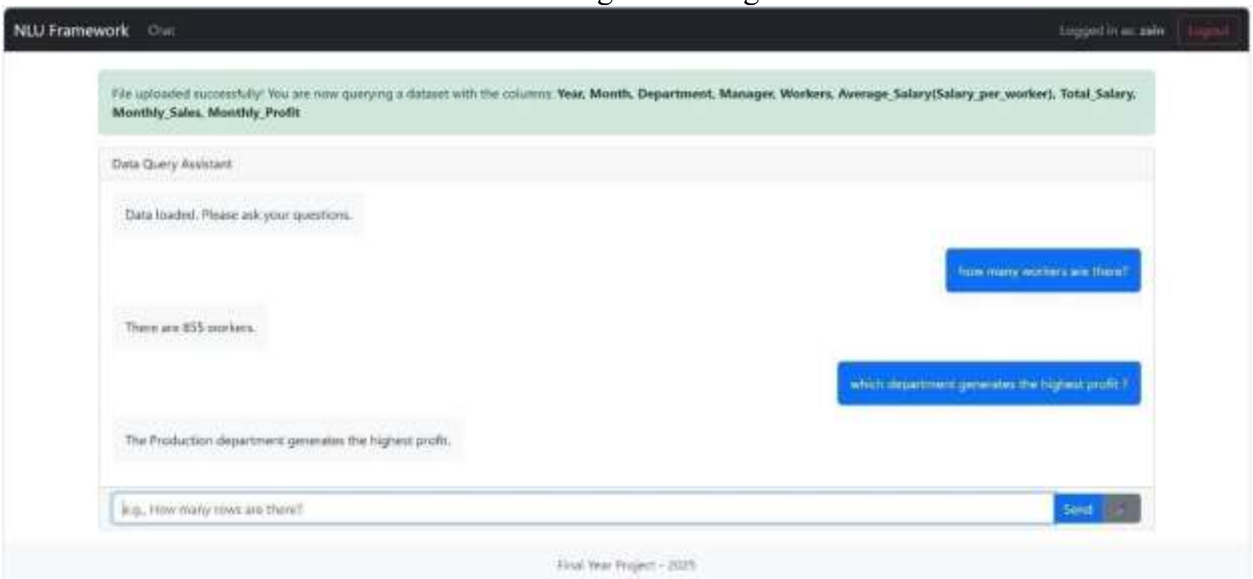

Fig 6: Interaction between User and Chatbot

6. EVALUATION METRICS

The performance of the Speech-Driven NLU Framework is examined using several measures that describe how the system behaves in practical usage. These measures focus on aspects such as accuracy, reliability, response time, and ease of use. Together, they help show how well the system understands spoken queries and produces useful analytical results.

1. Accuracy of Code Generation:

Accuracy refers to how correctly the system generates Pandas code based on the user's spoken query. It is measured by comparing the number of correctly generated outputs with the total number of predictions made by the system, as shown below:

$$\text{Accuracy (\%)} = (\text{Number of Correct Predictions} / \text{Total Predictions}) \times 100$$

A higher accuracy value indicates that the system is able to interpret user intent correctly and generate valid executable code with fewer errors during data operations.

2. **Execution Reliability:**

This metric looks at how consistently the system is able to execute the code generated from user queries. Reliability is assessed by comparing how often the code runs successfully with cases where execution fails due to syntax errors, dataset mismatches, or other processing issues. A higher reliability value indicates that the system can handle different analytical requests without failing or producing errors.

3. **Response Time:**

Response time measures the duration between a user speaking a query and the system returning the final result, including both text and audio output. Shorter response times indicate that the system is capable of supporting real-time interaction, which helps keep the overall experience conversational and natural.

4. **Error Handling Effectiveness:**

This metric examines how the system responds to queries that are unclear, incomplete, or contain errors. It focuses on whether such inputs can be processed without causing failures or interrupting the overall workflow. Effective error handling allows the system to continue functioning smoothly even when user input is imperfect.

5. **User Satisfaction and Usability:**

User feedback was collected to understand how people experienced the voice-based interface. Participants shared their opinions on how clear the explanations were, how natural the interaction felt, and how easy the system was to use without technical knowledge. The generally positive feedback suggests that the framework is easy to use and suitable for real-world scenarios.

6. **Explainability of Results:**

The framework provides natural-language explanations for each analytical result to help users understand the output more easily. This metric evaluates how clearly these explanations convey the results and whether users can follow the reasoning behind them. Clear explanations improve transparency and support better decision-making.

Taken together, these evaluation metrics show that the framework not only produces accurate analytical results but also remains reliable, responsive, and easy to use. This supports the goal of making data analytics more accessible to users who do not have technical expertise.

7. CONCLUSION

This work describes a Speech-Driven Natural Language Understanding framework designed to make data analysis easier for users without technical skills. The system combines speech-to- text processing, language-model-based code generation, secure execution of analysis tasks, and text-to-speech feedback to support voice-based interaction with structured datasets. Experimental results indicate that the framework can translate spoken queries into executable Pandas code, perform analytical operations reliably, and explain results in simple natural language. The system also shows stable behavior during error handling, provides quick responses, and offers an interface that is easy to use in practical scenarios. By reducing the need for programming knowledge, the framework helps improve access to data analytics and supports the use of voice-based approaches for data analysis.

8. FUTURE ENHANCEMENTS

This work opens up several directions for future improvement:

1. **Multi-modal Input**: Integration of text, image, and gesture-based inputs alongside voice.

2. **Domain Specialization**: Fine-tuning or instruction- tuning of language models for specific analytical domains.
3. **Advanced Code Generation**: Implementation of constraint satisfaction and formal verification approaches to improve code correctness.
4. **Knowledge Graph Integration**: Incorporating domain- specific knowledge for enhanced context understanding.
5. **Distributed Processing**: Extension to support analysis of distributed datasets and big data frameworks.

9. REFERENCES

[1] Tabular-textual question answering: From parallel program generation to large language models Tang, X., Chen, L., Yang, W. et al. (2025) — World Wide Web (Springer)

[2] Natural Language Analytics with Generative Large-Language Models: A Practical Approach with Ollama and Open-Source LLMs Marcondes, F. S., Gala, A., Magalhães, R., Perez de Britto, F., Durães, D., Novais, P. (2025) — SpringerBriefs in Computer Science.

[3] I2R-NLP at SemEval-2025 Task 8: Question Answering on Tabular Data Yuze Gao, Bin Chen, Jian Su (2025) — ACL / SemEval.

[4] CSIRO-LT at SemEval-2025 Task 8: Answering Questions Over Tabular Data using LLMs Tomas Turek, Shakila Mahjabin Tonni, Vincent Nguyen, Huichen Yang, Sarvanaz Karimi (2025) — ACL / SemEval.

[5] Speech2SQL: End-to-End Speech-Driven Query Generation Using Transformer Encoders Deng, Y., Fu, R., Xu, J., Wong, D. (2024) — Springer Nature Computer Science

[6] NL2Pandas: Benchmarking LLM-Based Code Generation for DataFrame Operations Shah, R., Patel, A., Yu, X. (2025) — arXiv / Under Review at WWW Conference

[7] Smart Voice Assistant with Sentiment Detection Das & Roy (2023)

[8] Patel, S., & Mehta, A. (2022). "Speech Recognition Chatbot Using Rasa Framework." *Journal of Intelligent Systems*, 37(3), 301-312. DOI: 10.1515/jisys-2022-0154.

[9] Kumar, R., Sharma, P., & Verma, N. (2023). "AI Chatbot for Student Assistance." *Computers & Education*, 192, 104634. DOI: 10.1016/j.compedu.2023.104634.

[10] Verma, K., Gupta, R., & Mehta, S. (2024). "Data-Driven Conversational Bot with Analytics Dashboard." *Journal of Network and Computer Applications*, 212, 107586. DOI: 10.1016/j.jnca.2024.107586.