A conversational Data Agent for Bridging Gap between Data Scientists and Data users

Ashik Elahi   
*Data Science Fellow*  
*Center for Data Research and Analytics*Bethesda, Maryland  
ashik.elahi@cfdra.com

Yushuf Sharker  
*Director*  
*Center for Data Research and Analytics*Bethesda, Maryland, USA  
yushuf.sharker@cfdra.com

*Abstract*— This paper presents a conversational data agent designed to enhance data accessibility for non-technical users re-usability by data scientists, particularly within ALS clinical research. It addresses the challenges of data comprehension and the disconnect between data scientists and users, leading to inefficient processes and fragmented documentation. The agent integrates Large Language Models (LLMs) to facilitate natural language interactions, allowing users to query data and receive analytical insights without requiring technical expertise. Developed as a Shiny app in Python, this system combines data, codebooks, and metadata while offering on-demand analytical results and R/Python code to ensure reproducibility. While currently effective for structured data, it lacks support for unstructured data types, with future enhancements aimed at incorporating machine learning for adaptive learning based on user interactions. This innovative approach promotes data-driven decision-making and bridges communication gaps between data scientists and end-users.

Keywords— Conversational Data Agent, Data Accessibility, Non-Technical Users, ALS Clinical Research, Large Language Models (LLMs), Natural Language Interaction, Shiny App, Analytical Insights, Researchers, Monitoring and evaluation, Data-Driven Decision-Making.

# Introduction

Understanding data is a difficult and resource-intensive process that requires technical expertise. While data scientists create tables and figures to interpret data for a general audience, the full potential of this information often goes unrealized. When a data scientist receives new data, they usually lack background knowledge about its origin or metadata. They spend a significant amount of time on an exploratory process to understand the data generation protocol and its context, a crucial but often poorly documented effort. As this understanding evolves, the lack of proper documentation means that when a team changes, new members must repeat the entire learning process from scratch. It is therefore essential to have well-documented and aligned information about domain knowledge, data generation processes, and metadata, ensuring both technical and non-technical users can benefit from the data [1].

Data scientists often face the challenge of making complex analytical reports understandable to non-technical audiences. The reports frequently require additional support and interpretation from the data scientist, as the audience is not equipped to navigate the underlying data.

Integrated data documentation and reporting is a proven solution but is often too difficult for small projects to implement. The required infrastructure and specialized personnel are typically not available. This leads to crucial background information being left in fragmented locations after finalizing analyzing the data by a data scientist, making it nearly impossible for others to reproduce the reports or reuse the data. The problem becomes even more severe with complex databases.

A powerful solution is a system that directly embeds all relevant data, metadata, and background information within the analytical report itself, as seen in some advanced dashboards. However, the final hurdle is ensuring this integrated information is presented in a way that is easily digestible and fully understandable to the end user.

## Integrating Large Language Models

Large Language Models (LLMs) are a type of AI that can process, understand, and generate human-like text. They are a powerful new tool capable of interpreting analytical results with high accuracy. When properly prompted, an LLM can even analyze data, perform complex queries, and return the necessary code or results [2].

Integrating an LLM into data analytics outputs, such as a data dashboard or web page, presents a smart solution for enhancing data accessibility. LLMs can enhance user engagement by providing more natural, personalized, and efficient interactions, responding to complex queries in a conversational manner.

The integration of chatbots with traditional data dashboards is a major step forward in how users interact with business intelligence. While traditional dashboards are static and require manual navigation, an AI chatbot transforms this passive experience into an active, conversational one. This creates a "conversational dashboard" where the chatbot acts as a virtual data analyst, providing tailored, on-demand insights in response to a user's questions. This eliminates the need for users to sift through complex menus or possess technical expertise.

## The Conversational Dashboard

This integration facilitates a more dynamic and personalized user experience by enabling real-time data interaction through natural language. For example, instead of manually applying filters to see regional sales data, a user can simply type or speak a request like, "What were the sales figures for the Northeast region last quarter?" The chatbot processes this query, translates it, and either presents the answer conversationally or automatically updates the dashboard's visualizations. This not only speeds up the time to insight but also democratizes data access, empowering non-technical users to ask ad-hoc questions and receive immediate, data-driven answers.

## A Streamlined System for Data Analysis

A streamlined system of analysis and representation can reduce statisticians' coding burden and make results reusable over time. It would help researchers and statisticians access and build on previous work easily, saving time, improving consistency, and enhancing collaboration. Integrating large language models (LLMs) into the system would enable researchers to get information about the data and interpret it easily.

Therefore, we aimed to develop a data agent that integrates LLMs in analytics. The smart data agent contains the basic analytics of data, which are interpreted on the pages. The LLM’s behavior is programmed so that it can respond to all queries that might arise from the data.

This would be an initiative to bridge the gap between data scientists and users. The primary audience would be the researchers who are involved with a large database that contains longitudinal data from multiple completed Phase II/III clinical trials for ALS. PRO-ACT is a large database includes multiple data tables and >13K ALS subjects [3]. The data collection protocol is a mix of different study protocols, and the study population is diverse. Therefore, a better understanding of the data and metadata is required to get useful information.

The project intends to develop an LLM-supported R/Python Shiny app for the PRO-ACT database, the largest database for ALS trial data, to support TAK-138 in early and late-phase development.

# Methodology

A Shiny app was developed in Python and R, integrating data, codebooks, and metadata into a single application. To facilitate team learning and knowledge transfer, a dynamic log file was incorporated, allowing for a journal of learning about the data. The application features a dashboard with various pages and tabs that display summary analytics, including tables, figures, and their general description and interpretation.

A dedicated page within the dashboard was created for the chatbot, which is powered by Large Language Models (LLMs) accessed via an Application Programming Interface (API). We integrated both Google's Gemini and OpenAI's GPT models, giving the user the flexibility to choose their preferred model for interpreting results.

To guide the LLM's behavior, we developed a generic prompt that directs its responses to user queries. A custom function was created to automatically generate a schema from the integrated data and codebook, which is then dynamically included in the generic prompt. This process provides the LLM with a comprehensive understanding of the database's content and structure without exposing sensitive data. We also appended relevant analytical results to the prompt to further inform the model's responses.

The chat interface includes a query box for user input and displays the LLM's responses directly on the screen. All conversations are stored in a log file, which serves as a valuable resource for developers and analysts to improve the user experience over time.

The application's backend was developed using Python 3 and is designed to be published on a dedicated server. It leverages various Python libraries including *shiny* [4], *plotly*, *chatlas* for analyzing and visualizing data, as well as for processing and understanding natural language queries. We also used backend services like *DuckDB* for efficient data querying and retrieval.

We used the Pooled Resource Open-Access Amyotrophic Lateral Sclerosis Clinical Trials (PRO-ACT) database, the largest database for ALS trial data, to demonstrate our application. The PRO-ACT database contains longitudinal data from multiple completed Phase II/III clinical trials, with over 13,000 subjects and diverse study protocols. We reorganized the data tables to simplify integration and performed descriptive analytics to showcase key clinical outcomes and patient survival metrics within the dashboard.

Recognizing that LLM-supported analytics is a novel field, our approach remains flexible to accommodate future improvements and enhanced user experiences. To ensure data confidentiality, we implemented a system where only the database schema is shared with the LLM, not the raw data itself, thereby eliminating the risk of a data breach.

# Results and Dashboard Functionality

The dashboard presents summary results across various tabs and pages. The initial pages provide a detailed overview of the data and summary statistics, including patient demographics, clinical characteristics, genetic mutations, and major endpoints such as the ALSFRS score. It also visualizes the longitudinal distribution of the score for both treated and control subjects across different trials.

The LLM chat feature is located on the final page. Users can ask questions about the data and receive clear and concise responses. The system is designed to respond to questions related to both the data itself and the analytical results displayed on the dashboard. For analytical queries, the LLM will provide not only the result and its interpretation but also the underlying R/Python code. This code can then be compiled by a local interpreter to reproduce the results. The conversational interface also supports follow-up questions, allowing users to delve deeper into the data in a natural iterative manner.

## Enhancing User Interaction

The chat interface handles a wide range of natural language queries. For example, a user can ask "What are the average ALSFRS scores for male patients?" to receive a direct answer, or pose a more complex question like "Show me the relationship between patient age at diagnosis and disease progression rate." In response, the LLM generates the necessary R/Python code to perform the analysis. This code is then executed, and the system returns the visualization, analytical output, and a clear interpretation of the findings.

## Performance and Logging

All chat conversations are logged to monitor the system's performance and continuously improve the LLM's behavior. This log file provides valuable insights into common user queries, helps identify areas where the LLM's responses can be refined, and informs future updates to the prompt engineering and backend services. This feedback loop ensures the tool remains a relevant and effective asset for researchers and analysts.

# Discussion

We found the app to be an effective tool for exploring data, especially for non-technical researchers like clinicians who may not be familiar with database management or coding. The conversational nature of the LLM is simple and intuitive, allowing them to extract information and formulate more meaningful research questions for data scientists to address.

Analyzing new data is a resource-intensive process. Clinical researchers often ask questions, and statisticians analyze the data to provide answers. Initially, neither party may fully understand the data contents or collection process. Once clarified, statisticians provide insights that prompt further questions from researchers, creating an ongoing cycle of analysis and requests. This keeps statisticians occupied with coding and data interpretation.

Furthermore, data and analysis results are often left disorganized, leading to the repetition of analysis for later needs. Code and results are frequently scattered across platforms like SharePoint, making them difficult to reproduce. This is particularly evident with a complex database like PRO-ACT [1]. The data analytics process often repeats itself for new data or when different researchers and statisticians are involved, wasting time and leading to underutilized outcomes. A better system is needed to effectively preserve and reuse data and analysis outputs. A streamlined system of analysis and representation can reduce the coding burden on statisticians and make results reusable over time, helping researchers and statisticians build on previous work, saving time, improving consistency, and enhancing collaboration. Integrating LLMs into the system enables researchers to easily get information about and interpret the data.

This app model is in its developing stage, and we have collected user experiences, primarily in an unstructured manner, from researchers. Their feedback confirms the utility and popularity of the tool. We strongly believe that this kind of smart data dashboard can also benefit other applications, such as monitoring and evaluation, or following stock prices. By making data more useful to users, we can promote data-driven decision-making.

Regarding data confidentiality, we designed the app to share only the schema with the LLM, not the raw data. This approach eliminates the risk of data breaches through LLM.

## Limitations and Future Work

The app is reproducible for databases that includes structured data tables. However, it currently has limited support for unstructured data, such as images and audio, due to the high cost of processing and interpretation by LLMs. Despite this limitation, we believe this app will be a great support tool for data users across various research, business, and non-profit sectors that require a data monitoring system. A potential future development would be to incorporate a machine learning model that learns from chat conversations. This would allow the system to develop a recommendation engine for data-driven decision-making.

# Conclusion

This app represents a novel approach to presenting and sharing data, creating a system that is sustainable, reusable, and easy for non-technical individuals to understand. By providing a conversational interface with an LLM, the app empowers non-technical users to independently extract information from complex data. This innovative approach helps bridge the communication gap between data scientists and end-users, promoting a more collaborative and efficient data-driven decision-making process.

# Acknowledgments

I would like to express my sincere gratitude to Jaynal Abedin for providing insightful comments and feedback throughout the development of this app and contributed to enhancing the quality and clarity of this paper.

# References

1. A. Y. Wang et al., “Documentation matters: Human-centered AI system to assist data science code documentation in computational notebooks,” ACM Transactions on Computer-Human Interaction, vol. 29, no. 2, pp. 1–33, 2022.
2. Baker, N., & Radha, S. (2024). “Advancements in Large Language Models: Enhancing User Interaction with AI.” Journal of Artificial Intelligence Research, 68, 1-15. Available at: https://www.jair.org/index.php/jair/article/view/16358.
3. Pooled Resource Open-Access ALS Clinical Trials Consortium. (2024), <https://ncri1.partners.org/proact/document/displaylatest/6>, Access date: 08/23/2025
4. Posit, PBC. (2024). Shiny for Python: A web framework for Python. https://shiny.posit.co/py/