IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified ⁜ Impact Factor 7.105 ⁜ Vol. 9, Issue 5, May 2022

DOI: 10.17148/IARJSET.2022.9557

Voice Based Database Analyzer

Ayan Rajput[1], Ms. Meenu Garg[2]

Department of Information Technology, Maharaja Agrasen Institute of Technology, Delhi, India[1]

Assistant Professor, Department of Information Technology, Maharaja Agrasen Institute of Technology, Delhi, India[2]

Abstract: In recent few years, Artificial Intelligence has shown significant improvement and its user cases are growing rapidly. An application area of Artificial Intelligence is Natural Language Understanding (NLU). Voice assistants can have a conversation with the users in natural language. Voice assistants are quite easy to use and can handle many difficult tasks easy. But currently there are no voice assistants or chatbot to analyze database data and also it hard to analyze database data for people with non-tech background.

Almost all Artificial Intelligence programs are reasoning programs. And, to the extent that they reason well about a problem, all are somewhat good at problem solving. Expert systems have a knowledge-base containing extracted experience and an inference engine which is essentially a set of rules for applying the knowledge base to each particular situation that is described to the program. The chatbot's capabilities can be made better with modifications to the knowledge base or to the inference rules.

Traditionally chatbots were made on a decision tree-based logic. They contain many if else logics and these logic controls the flow of the tree-based logic. This tool is also based on the same logic where we try to extract user intent based on words present in the query.

In this project we are trying to make a voice-based database analyzer which will be an expert system. The expert system will have a knowledge base which will be used for database processing. It will take a database connection string as input then users will speak their queries, the queries will be converted from voice to text and it will perform the necessary calculations with the database data, knowledge base and queries and will show the text, tables or charts accordingly.

INTRODUCTION

In this new era of technology and research, the use of artificial intelligence, machine learning and natural language understanding has made our lives much easier. These technologies have been shown to benefit different people in various fields such as education, tech/non-tech industry, e-commerce, etc. and one of the most prominent is voice and chats. Intelligent Virtual Assistant is a professionally developed software with a powerful speech to text system focused on audio data processing in the system, converting it into readable text and doing the optimal tasks. Nowadays these types of audio and chat assistants are available for performing multiple tasks and automating different workflows.

An application that is playing a major role in a people's daily lives is an expert system. For example, there are Expert systems used to diagnose different diseases, profit-related predictions, and different studies of motor-based vehicles individually. Some types of expert systems are proposed to replace human intervention, while others are designed to help them. Professional framework functions are part of a standard class of PC framework applications known as Artificial Intelligence.

Planning a pro-structure, one needs a data creator, one who focuses on how human experts solve decisions and makes sense models in terms that the computer framework can get.

We are developing an intelligent virtual assistant which can easily analyze database data using voice as input. It is not possible to design an intelligent system to solve this problem as it requires lots of sample data. An intelligent system learns from past data and solves the new problems based on that learning. Here we don't have any previous data available as it is a new type of technology so we are using expert systems which relies on predefined set of rules to solve a particular problem.

The column names in the databases are usually present in the computer naming conventions such as snake case(my_cloumn), pascal case (MyColumn) or camel case(myColumn) due to which it is not possible to exactly match the column which comes from user input, this problem is solved using fuzzy string matching

LITERATURE REVIEW

Speech to text translation has a long history and several modifications of great new things. Speech recognition, search, and voice commands have become quite popular on computers, phones and smart devices.

A database has always been a tool for making history. Decades ago, it was introduced as business software, helping people save time on efficient way of storing data, from budgeting to data handling. Nearly 50 years later, the database enters a new era, one in which we can replenish and renew data using the OLTP and OLAP systems.

Nowadays there are various tools present for the automation of data pipelines such as Hadoop and Apache Spark but they require skilled professionals to correctly use them. There have been no studies around making a chatbot which even people having no or very less experience in data analytics and database engineering can easily use to analyze database while asking simple queries to get relevant insights from the data.

METHODOLOGY

To make a tool which can easily process database data we require a coding language which is capable of handling and processing different type of data very easily. Here python programming language comes into play, python nowadays is heavily used for data analytics and visualization. Python also supports application and API development through frameworks like Django etc. which are currently being used to create backends APIs (Application Programming Interfaces). Here we are using python in our project for both of its uses. The frontend of the project is developed using the standard HTML, CSS and Vanilla JavaScript and backend using Django (A Python Web Development Framework). Currently we are using MongoDB Atlas as the database and it can be easily integrated with the python using the pymongo library. MongoDB is a no-sql document-based database which contains data as key value pairs in the document.

The frontend of the project is developed using HTML, CSS, JavaScript and jQuery and fuse.js libraries of JavaScript. jQuery is primarily a DOM (Document Object Model) manipulation library for the JavaScript which is also used for making and handling asynchronous AJAX (Asynchronous JavaScript and XML) requests to the server. JavaScript fuse.js library is a library for fuzzy string matching which gets us the most correct selected option from a list of options using a single text for matching.

To get the audio input and its text conversion we use the Web Speech API. The Web Speech API is used to add voice data into web apps. The Web Speech API has two parts: Speech Synthesis (Text-to-Speech), and Speech Recognition (Asynchronous Speech Recognition).

The frontend gets the connection string, connects to database using it and gets all data to the frontend and then converts it into JSON object and stores it on the frontend and displays the data in as some HTML tables to the user to left side of their screen. Then user clicks the speak button and web speech API uses its text-to-speech engine to translate it into readable text format. Then the text along with the database data (JSON format) is sent to the backend for processing. If the backend processes the data successfully and responses with status code of 200, then we get the JSON object from the backend and use it to display the data to the user accordingly. If type of data is text, then it is shown to the user as it is, but if its type of the data is table then we create a HTML table using it and will display the results to the user on the right side of the screen. The user can again press the speak button again to generate a new query and the get the relevant results. If user wishes to join the tables, he should say join/left join and then name of the table to be joined. It will open a new screen that will ask user for the common columns to join. After that it replaces the old table with the new table which is formed after joining the two tables and user can perform queries on it. If users say reset, they will be redirected to the table selection page. If user asks to create a chart and specifies two columns then a graph is created using the google charts library

The backend of the project is made using Django, which is a backend framework for python, it handles the request to display the page to the client using Views and URLs and return a proper response. When the user makes a request to the /analyzer endpoint it returns the webpage to the user which is essentially the frontend of the project. When the user makes ajax call to server using the database data (JSON format) and the voice input which is has been converted into text it processes it using pandas and returns the results in JSON format.

The queries are divided into 6 categories:

- Sorting Queries
- Max/Min Queries
- Filtering Queries
- Graph Drawing Queries
- Table Joining Queries
- Reset Query

Sorting Queries: If the message contains words like arrange, sort, etc. then is will be considered as a sorting query. First, we convert the queries and data columns into lowercase, then we iterate through each column to check if it is present in the query, if no match is found then we return unable to process the query. But if a column if matched, the data is sorted using that column using pandas, we also check if query contains the word descending, if it is found then sorting is done in reverse order. Then it sends response as type table as a JSON response.

Max/Min Queries: If the query contains words like maximum, minimum, greatest etc. then it is considered as a max/min query. First, we convert the queries and data columns into lowercase, then we iterate through each column to check if it is present in the query, if no match is found then we return unable to process the query. But if a column if matched we fetch the row containing max/min of the number and return it as type table as a JSON response.

Filtering Queries: If the query contains works like where etc. then it is considered as a filtering query. First, we convert the queries and data columns into lowercase, then only use the query on the right side of the filtering keyword (e.g., where) then we iterate through each column to check if it is present in the query, if no match is found then we return unable to process the query. But if a column if matched we look for words like greater than, less than, equal to etc. if these words are found then we take the data to the right side of the word and match it against the found column in each row of data. Then we return matched rows as type table as a JSON response.

Graph Drawing Queries: If the message contains the words graph or chart. then is will be considered as a graph query. First, we convert the queries and data columns into lowercase, then we iterate through each column to check if it is present in the query, if 2 columns are not found then we return unable to process the query. But if a matched, the data is changed accordingly using those column using pandas. Then it sends response as type graph as a JSON response and frontend shows a chart.

Table Joining Queries: If the query contains works like join etc. then it is considered as a table joining query. First, we convert the queries and data columns into lowercase, then only use the query on the right side of the joining keyword (e.g., left join) then we iterate through each table name to check if it is present in the query, if no match is found then we return unable to process the query. But if a column if matched we ask user for the common column to join. Then we replace the old table with the new table formed using the join.

Reset Query: If the query is reset then we take the user back to the table selection screen.

RESULTS

The project was able to answer simple queries of the user of type Sorting Queries, Max/Min Queries, Filtering Queries, Chart Drawing Queries, Table Joining Queries and Reset Query. It is very quick to join the tables into JSON format because it is processed on the frontend without any help of the server. The voice to text engine is working as expected. Suppose the table contains the column name, age, category, marks, year etc. then the project was able to answer queries like “give me the maximum of age”, “sort the data by name”, “arrange the data by marks descending”, “get the data where year is equal to 4”, “join with another table”, ”draw graph of age vs marks” etc. It was unable to answer complex queries like “give me the top 3 students by marks”, “what is age of student with name Ayan”, and some queries where query is asked in a different format like “get student whose age is more than 23” which is not incorporated in the inference rules. It currently on supports only one way communication from the database and does not return anything back to the database.

CONCLUSION AND FUTURE SCOPES

This is an attempt to make a voice-based database analyzer to ease the work of the user in which user can give a database connection string as input and ask queries and it will answer those queries using the data from databases. It is able to answer simple queries but unable to answer complex queries. It works when column names of database are in a format

which can't be spoken like column names which contains an underscore, due to the fuzzy logic but when tables with a lot of columns are there it is unable to extract the specified columns.
It has great potential to become a fully viable product if we increase the knowledgebase and add more rules in the processing of query and add more types of databases to connect. In the future we can add more types of queries, ability to process more than one type of databases and combine multiple databases to answer the queries and be able to return data in different formats rather than showing a HTML table.

REFERENCES

[1]. Research on expert systems, *Jan 1982, Bruce G Buchanan, University of Pittsburgh*

[2]. A Voice Based Assistant Using Google Dialogflow and Machine Learning *May 2021 International Journal of Scientific Research in Science and Technology DOI:10.32628/IJSRST218311*

[3]. Neiffer, J. P. (2018). Intelligent personal assistants in the classroom: Impact on student engagement. *Ph.D. Dissertation. College of Education and Human Sciences, University of Montana. Nilsen, J. H*

[4]. Røyneland, K. (2019). «It knows how to not understand us! » *A study on what the concept robustness entails in design of conversational agents for preschool children (Master's thesis).*

[5]. Voice Assistants and Smart Speakers in Everyday Life and in Education George TERZOPOULOS, *Maya SATRATZEMI Department of Applied Informatics, University of Macedonia, Thessaloniki, Greece Email: gterzopoulos@uom.edu.gr, maya@uom.edu.gr*

[6]. Automated Data Analysis Using Excel June 2007 DOI:10.13140/2.1.2381.2803 Edition: 1 Publisher: Chapman and Hall CRC Editor: Randi Cohen ISBN: 978-1584888857 Brian Bissett Department of the Treasury