Title: DSCI 551 Final Project Midterm Progress Report

Team Details (ChatDB 17)

Members: Qingyi Feng & Chuqi(Angel) Jin
Background and Skills:
***Qingyi Feng* (USC ID:8401362266 Email:qingyife@usc.edu)**
This is Qingyi Feng. During my undergraduate studies, I gained experience with basic programming languages, including R and Python. I also participated in several group projects that strengthened my problem-solving and collaboration skills. One of the most significant projects I worked on involved developing an interpreter for a subset of the Scheme language. Throughout this project, my team carefully considered various challenges in programming language design, ensuring that our implementation decisions for both the interpreter and compiler directly influenced the language's behavior and properties. While I have not learned SQL or NoSQL yet, I am eager to explore database management systems. Before now, I had no experience with MongoDB or Firebase. But, I am always open to learning new technologies that will enhance my programming and data-handling capabilities. Last semester, I took DSCI 552, where I studied machine learning techniques, including classification methods and tree-based algorithms. This course provided me with hands-on experience in implementing models and analyzing data, strengthening my analytical and computational thinking skills. Additionally, I worked on projects that required applying these concepts to real-world datasets, which helped me develop a more intuitive understanding of machine learning principles.

Chuqi (Angel) Jin (*USC ID: 9740105151 Email: chuqijin@usc.edu*)
I am a first-year applied data science master's student at USC. I have a bachelor's degree in statistics with a computational concentration and a minor in accounting from UCR. Last summer, I graduated with a variety of project experience and research skills in statistical analysis, data analysis, and data science. Some of the classes I've taken in recent years include probability and statistical theories, regression analysis, statistical computing, applied optimization, numerical analysis, introduction to data structure and algorithms, and data science using R, SAS, Python, and C++. My expertise is in R for massive dataset analysis and regression model building. The majority of projects include data cleaning, combining, organizing, exploratory data analysis, exploration of interaction terms, variable transformation, model building, model comparison and residual diagnosis. Furthermore, I took DSCI 552 last semester, which taught me more about fundamental machine learning methods. However, prior to taking this class, I had very little understanding of database systems and management, and I only knew the very basics of SQL and knew nothing about NoSQL. So, I believe this project will be a little challenging for me, but I'm prepared to learn throughout the semester and overcome each challenge one by one in order to lay the solid foundation for my future studies.

Implementation Questions:

- **Tech Stack Used:**
  - Pymysql - works with MySQL databases
  - MySQL - creates database queries.
  - Jupyter notebook - used for dataset preprocessing, such as cleaning, organizing, and subsetting.
  - Re: extracting keywords and patterns from user input.
  - Pandas: for efficiently handling, processing, and manipulating data.
  - Google-genai: Google Gemini LLM API for natural language processing.
  - OpenAI API: GPT for natural language processing.
- **Query Syntax Implementation Plan:**
  - NLP processing natural language input from the user & keywords extraction
    - Regex & NLP
      - Ex: Show me apartments in Los Angeles.
    - When the query structure is complex or includes a number of logical combinations. Then, we'll utilize large LLM models (ChatGPT and Google Gemini) to "understand" and generate precise queries.
      - Ex: Search for all two-bedroom apartments with pools in Los Angeles priced between $1,000 and $2,000 that are pet friendly.
    - Keywords: amenities, pets, bedroom, bathrooms, locations, price, etc.
      - Ex: “apartment in LA” → city = “LA”; “price above 1000” → price > 1000.
  - Convert to SQL/MongoDB queries
    - If it is a simple query, splice the SQL or MongoDB query directly based on the rules (e.g. SELECT ...). WHERE ... or db.collection.find(...))
    - If it is a complex query, use LLM to process user intent into database queries.
  - Run the queries and return the results from the databases
    - The generated query is executed in the database to retrieve data on the requests. Return the results to the user.
- **Database Selection:**
  - SQL database: MySQL is a SQL database that stores reasonably structured data and allows for many table associations.
  - NoSQL Database: MongoDB is a NoSQL database that is ideal for storing unstructured data with flexible schemas.

Planned Implementation:

Our approach begins with NLP processing natural language input from the user, followed by keyword extraction using regex and LLM (use GPT and Google Gemini to parse more complicated natural language needs). We're using keywords that relate to specific attributes in each table, then turn those keywords into SQL and MongoDB queries that we write and execute to access the database and retrieve results. Finally, give the results back to the users.

The database structure we designed now is slightly different from our original plan. The original plan was to organize our dataset into three tables:

- Apartments basic information: id, location, size, price, category 2

- Apartments facilities: amenities, bathrooms, bedrooms 3.
- Additional information about the apartments: has_photo, pets_allowed

However, after cleaning our data and looking through each feature value, we decided to remove some features from the original dataset due to massive missing values in the ‘address’ feature. Redundancy in 'category', 'body', 'fee', 'price_display', 'price_type', as well as the irrelevance of the 'time' feature. We redesigned our database into these three databases:

- Rental (2 tables)
    - General_info: this table includes id, title, square_feet, cityname, state, latitude and longitude
    - Property_details: this table includes bathrooms, bedrooms, amenities, pets_allowed
- Price (1 table)
    - Pricing: this table includes id, price, currency
- Additional (2 tables)
    - Media: this table contains id and has_photo
    - Sources: this table contains id and source

The change of the database structure helped us to have a more clean structure by reducing useless features from our dataset and improved queries efficiency.

Project Status:
We’re on track. Data preprocessing and data structure design have been completed. We have already set up the LLM and written the LLM test prompt we will use for this project. We are almost finished designing our SQL schema, writing queries to construct our SQL database. We are organizing data into correct tables with their attributes. The next step is to further improve our queries, finalize our SQL implementation, and move on to the MongoDB data structure. Then, we will begin NLP processing, using Regex & LLM models to process the natural languages from our test prompt. At the same time, we also need to conduct performance tests to ensure everything works properly.

Challenges Faced:
- Data preprocessing: The original dataset is large and incomplete, so it needs to be cleaned, de-weighted and reasonably split into multiple tables.
- Natural language processing: It is difficult to handle complex expressions with simple regularization, and we need to do prompt engineering when introducing NLP/LLM.
- Query accuracy: Sometimes the automatically generated query requires multiple verifications, we are optimizing the parsing logic through sample libraries and test cases.

Timeline:

Date	**Phase**	**Checkpoints**
~~Due 02/14~~	~~Data Cleaning~~	~~● Collect, check, and clean data.~~

		~~● Remove observations with empty values.~~ ~~● Reorganize the dataset, ensuring that each variable value is in its appropriate column.~~
~~Due 02/21~~	~~Data Cleaning~~	~~● Remove unimportant variables.~~ ~~● Select and filter a subset before proceeding with the next phases.~~
~~Due 02/28~~	~~LLM Setup~~ ~~Data Structure (mySQL)~~	~~● Start writing the LLM test prompt.~~ ~~● Learn how to turn text into SQL and NoSQL queries.~~ ~~● Design MySQL schema~~ ~~● Organize the data into tables.~~ ~~● Code out the queries command.~~
~~Due 03/07~~	~~Submission~~	~~Midterm Progress Report~~
Due 03/14	Data Structure (MongoDB)	● Define the NoSQL schema ● Optimize queries
Due 03/19	Data Structure (mySQL & MongoDB)	● Complete the code for data entry in both databases. ● Examine the produced database structures in both databases. ● Test data retrieval from both databases.

Due 03/21	LLMs	• Use the LLM API to process natural language queries. • Fine-tune prompt engineering
Due 03/28	LLMs	• Improve our written code based on the test results from the previous phase. • In both databases, Test LLM generated SQL and NoSQL queries.
Due 04/04	LLMs	• Edit the code to obtain more improvements. • Test the entire system.
Due 04/11	Implementation	• Unit Testing • Debugging
Due 04/18	Performance optimization	• Optimize query execution time for large datasets • Improve database
Due 04/20	Implementation	• Run the code to see its overall performance
Due 04/21	Presentation	Demo (In-class)
Due 04/23	Presentation	Demo (In-class)
Due 04/25	Final Report	• Report writing
Due 05/02	Final Report	• Test system • Provide link to the code repository and necessary documentation
Due 05/09	Submission	Final report submission