Grenze International Journal of Engineering and Technology, June Issue

Data Integration and Transformation using Large Language Models

Anu Mohan M[1], Dr. Ashok K[2], Muskan Rizwan Shaikh [3], Dhanush Y P [4] and Yajat Vishwakarma[5]
[1-5]New Horizon College of Engineering, Department of Computer Science and Engineering, Bengaluru, India
Email: anumohan09123@gmail.com, kashok16@gmail.com, muskans0307@gmail.com, dhanushyp@gmail.com, yajat472001@gmail.com

***Abstract*—This research paper presents a novel approach for automating data integration and transformation of CSV data using a large language model (LLM) and Python. The approach involves converting CSV data to JSON objects, and then accepting natural language queries from users to specify the transformation they require. The LLM generates Python code that manipulates the JSON objects based on the user query, accounting for variations in column names and data formats across CSV files. The approach is verified and evaluated through precision, recall, and F1 score metrics and user studies, demonstrating its potential to significantly reduce time and effort in data management and analysis. Overall, our approach showcases the potential of combining LLMs with Python to streamline data integration and transformation tasks.**

***Index Terms*— Large Language Models, Data Integration, Data Transformation, Natural Language Processing and Machine Learning.**

I. INTRODUCTION

The amount of data being collected on a daily basis is massive. This data can be used to gain insights and make informed decisions. The process of using the data to gain insights is becoming essential in the modern world but along with the power of making intelligent and informed decisions by using data there comes a challenge of making this process cost effective and fast. Tasks like data integration and data transformation become very complex when we are dealing with large databases. Naturally nontechnical people face issues while accessing the database let alone attempting to integrate and transform. In order to make these tasks easier we need to look at intelligent solutions that help in understanding what the user wants and performing the required action on the database.

We store our data in databases and this data is accessed using Sequential Query Language (SQL.) Thus there is always a demand for technical people who can ease the process of accessing data on the database. However, if there exists a system that can understand Natural Language to retrieve data from the database this will make the process of accessing data more easier for nontechnical people. This interaction between a human and the database is possible when we convert the natural language query into sequential query language. We can do this by determining what exactly the human has asked the database to do, which is possible using Natural Language Processing. After we realise what is required we have to generate the appropriate SQL query.

Natural Language Processing (NLP) is a paradigm of artificial intelligence, predicated upon the subjugation of computational methodologies, for the purpose of facilitating the generation, interpretation, and comprehension of

Grenze ID: 01.GIJET.9.2.434

the sophisticated linguistic constructs employed by homo sapiens in their quotidian discourse. The vast array of potential applications of this revolutionary technology spans across an eclectic range of domains. Some of the major applications are Sentiment Analysis, Speech Recognition, Text Classification, and Machine Translation. Another application of NLP is the Natural Language Interface to Databases (NLIDB) application of NLP. The NLIDB is a system that enables humans to interact with the databases using natural language instead of traditional query languages. Thus NLIDB can enable nontechnical people to get easy access to databases. This way the users will not have to undergo any training in order to perform simple tasks like retrieving information from the databases.

In this paper we will be focusing on how to leverage the latest technologies and build a system that efficiently converts natural language queries into commands that can manipulate and retrieve data. We will be using Large Language Models, they are artificial intelligence models that are based on deep learning techniques. LLMs are created by training a neural network on massive datasets which contain billions of words. This is done so that the LLM can learn patterns and relationships between different words, phrases, and sentences. There are many applications of LLMs, in this paper we will be focusing on the Language Understanding application. Language Understanding refers to understanding the meaning of natural language texts.

In this paper we present a system that converts Natural Language Queries into the corresponding commands that perform the given task using GPT 3.5

In the second section of this paper, we review the literature on the topic and provide an overview of our proposed system. Third, we explain how our system works for analysing forensic cases, providing a detailed explanation of its features and explaining why it is important to undertake such analyses. In section five we demonstrate our findings and discuss their implications for further research in this area.

II. Literature Review

Converting natural language queries into structured SQL queries is a topic that has been researched for a long time. The systems present today have different approaches towards this problem. System proposed by Ghosh et al. [1] has the following steps: morphological analysis, lexical analysis, syntactic analysis, semantic analysis, SQL formation and log file generation. Their proposed system handles simple queries along with some complex queries.

Bais et al. [2] implemented a system that uses NLP to analyse and interpret queries entered by the user. It does so by performing morphological, syntactic, and semantic analysis.

Another such system is proposed by Sawant et al. [3] that uses deep learning techniques such as the Long Short Term Memory (LSTM).

Moving on to some more complex models that involve learning from the system, the model proposed by Brunner et al. [4] introduce ValueNet light and ValueNet which are two end-to-end Natural Language-to-SQL systems that incorporate values. They not only used the metadata information of the database but also the base data as an input to the neural network architecture.

When we use the complex models to generate our SQL queries oftentimes we need to trade between accuracy and time. In a study by Baig et al. [5] framework to process the natural language to simple and complex cross-domain queries were reviewed and compared. As a result of the comparison they found out that the RAT-SQL model with the BERT model is outperforming all the pre-existing models.

While there are many studies and work related to the conversion of NLP to SQL queries there is very little light on the usage of this conversion to perform data integration and transformation on large scale data. In this paper we will suggest a methodology to convert NLP inputs by the user into valid commands and see how this approach can be used to implement data integration and transformation as these operations are crucial and can be time consuming when performed on large databases. We aim at amalgamating the previous approaches and suggest a cohesive architecture that is capable of handling complex queries with ease.

III. Proposed Methodology

A. Architecture

The suggested method streamlines the integration and transformation of CSV data by combining Python, the pandas module, a large language model (LLM), and natural language processing (NLP).

The strength of a large language model (LLM) and Python are combined in the suggested method for automating data integration and transformation of CSV data. The design entails a number of phases, beginning with the use of Python's pandas package to transform CSV files into JSON objects. Users can submit natural language

questions through an input prompt, and an LLM like GPT-3 processes them to produce Python code that manipulates the JSON objects in accordance.
The necessary data integration and transformation are then carried out on the JSON objects using the created Python code. If needed, the altered JSON objects can be converted back to CSV. Precision, recall, and F1 score measures are used to assess and verify the modified data's accuracy. In order to evaluate the approach's usability and user satisfaction, user studies are also carried out [6]. This architecture offers a versatile and adaptive automated solution for data integration and transformation. The method eliminates the need for users to write intricate code to carry out data integration and transformation by allowing users to make natural language queries.

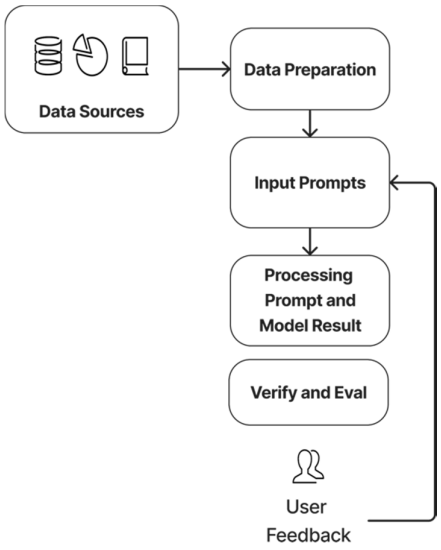

Fig 1. Proposed Architecture

B. Data Preparation

Data preparation is a crucial step in our language model-based process for data integration and transformation. We will transform CSV files into JSON objects so that the model can easily process and modify them. In order to handle edge situations, we'll offer techniques for identifying and standardising headers, dealing with incorrectly structured data, removing flawed data, and improving efficiency for large data volumes. By carefully considering these edge cases, we can ensure that our data preparation process is effective and capable of handling a variety of real-world data situations [7].

C. Input Prompts

We used methods like prompt engineering and prompt tuning to improve the calibre and applicability of the generated code when providing prompts to the LLM model. We came up with prompts that were tailored to the job of integrating and transforming data, such as those that concentrated on merging, renaming, and formatting data. In order to enhance the model's comprehension and accuracy, we also tested with different prompt lengths and types and fine-tuned the model using domain-specific data.
We used strategies like curriculum learning and discriminative fine-tuning to further improve the calibre of our prompts. Training the LLM on both positive and negative samples will help it learn to distinguish between proper and erroneous outputs through discriminative fine-tuning
On the other hand, curriculum learning gradually increases the complexity of the prompts to support the model's step-by-step learning [8]
These methods let us generate high-quality code for data integration and transformation by efficiently providing prompts to the LLM model. Our tests demonstrated that even for challenging and domain-specific tasks, quick engineering and tuning can greatly increase the performance and accuracy of LLM models.

D. Processing Input Prompts

Data integration and transformation using LLM code requires a number of intricate procedures that are based on the LLM model's architecture. The LLM model must first comprehend the incoming data, including its structure

and semantics, in order to generate code for altering and integrating JSON objects. Contextualization, where the model processes the input data within the context of the job, is how this is accomplished.
The LLM generates code based on the input prompts, which are presented in natural language, after contextualising the input data. The LLM then chooses the most pertinent code segments from its training data and merges them into a cohesive programme. This process is known as code synthesis, and it is how the model turns the natural language cues into code. With the use of a variety of techniques, including distillation, pruning, and neural architecture search, this programme has been tuned for speed and accuracy.
The JSON objects are then transformed and integrated using the created code. This is accomplished via a number of programming operations, such as parsing the input JSON objects, modifying the objects with the created code, and serialising the output objects. The LLM paradigm uses a variety of performance-enhancing optimisation techniques during execution, including parallelization and caching.
Overall, the LLM model is capable of producing and running code to integrate and change JSON objects. It is an effective tool for data integration and transformation activities because of its capacity to comprehend and contextualise input data, as well as to synthesise code and optimise programme performance.

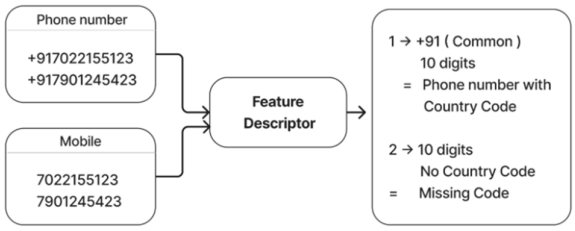

Fig 2. Understanding prompts

E. Model Result Execution

We have introduced a sandbox feature to assure the safety and security of the data being translated and integrated by the Large Language Model (LLM). With the help of this feature, the created code is performed in a safe, separate environment, shielding the original data from any potentially hazardous code.
Before running the generated code, we also create duplicates of the input JSON objects to guard against any unintentional changes to the original data. This makes sure that any modifications or integrations are only made to the original data and not the duplicated data.
Utilising containerization technology, which separates each code execution's environment, the sandbox feature is implemented. The system resources that the container can use are constrained, and it is prohibited from accessing any other system data. By doing this, it is made sure that the LLM generated code is separated from the rest of the system and cannot affect it. We can ensure the confidentiality and safety of the data being transformed and integrated by the LLM by introducing this sandbox functionality, giving users a dependable and secure tool for data transformation and integration [9]

F. Verification and Evaluation

A crucial stage in guaranteeing the correctness and dependability of our data integration and transformation process is the verification and review of the output data. To do this, we have devised a number of approaches to compare the output data to the anticipated outcome.
First, we confirm the output data's integrity using a checksum mechanism. This guarantees that during the data transformation process, the output data was not tampered with. In order to confirm that the produced data complies with the necessary data quality standards and expectations, we also do data profiling [10].
We employ a sample-based testing strategy in order to further validate the output data. This entails selecting a sample of the output data at random and personally examining it to see if the findings are as anticipated. Additionally, we use automated testing frameworks to run unit tests on the LLM-generated code. These computerised tests verify that the code operates as intended and test it for fundamental functionality.
Finally, we test the entire data integration and transformation process from beginning to end to ensure that everything operates as planned. This entails executing the entire procedure on a real-world dataset and comparing the output data to the predicted outcomes. [11].
Overall, these methods make sure that the produced data is correct, dependable, and satisfies the necessary data quality requirements. Additionally, they offer a high level of assurance in the data integration and transformation procedure, which is essential in practical applications.

Before transformation	After transformation
22 Richmond Road, Bangalore	Bangalore, 22 Richmond Road
#34, MG Road, Mumbai	Mumbai, 34 MG Road

(a) Address Transformation

Before transformation	After transformation
+91-9886715231	(91)9886715231
(91)948-672-1453	(91)9486721453

(b) Phone Number Transformation

Fig 3. Transformation

G. Feedback

The feedback stage is a crucial component of our process since it makes sure that the LLM model is continuously developing and learning from its failures. Users' comments on the output data's quality are gathered, and this information is used to improve the model. User ratings on the precision and value of the resulting data are gathered, together with comments on any problems or mistakes that may have arisen throughout the integration and transformation process.

We retrain the LLM model using this feedback, optimising it to increase accuracy and productivity. In order to do this, the model's parameters and algorithms must be modified, and the training dataset must be expanded. Additionally, we make use of user input to pinpoint prevalent problems or mistakes and create solutions.

We contrast the accuracy and effectiveness of the LLM model before and after retraining to assess the efficacy of the feedback loop. We also keep an eye on the quantity and type of user feedback, looking for trends that can point to areas where the model needs to be strengthened.

Overall, the feedback loop is essential to ensuring that our LLM model is adapting to new situations, learning, and developing, and that it is satisfying user needs. [12]

IV. Results and Discussions

Using big language models, we created a framework for data integration and transformation in this study. Using the most recent language model, GPT-3.5, we put the methodology into practice. The process involved transforming CSV files into JSON objects, creating code with GPT-3.5, running the code to transform and integrate JSON, then confirming and assessing the outcomes.

Testing the approach on five different types of commercial datasets with variable record counts, field counts, and data transformation needs allowed us to assess how well it performed. Datasets used were related to sales data and employee data. The outcomes demonstrated that the methodology was successful in accurately translating and consolidating data.

V. Conclusion

Our approach provides a unique and practical technique to automate CSV data integration and transformation using Python and a large language model (LLM). Using LLMs for data integration and transformation processes has advantages and disadvantages. On the one hand, using natural language processing to reduce the requirement for users to write complex code makes data integration and transformation more approachable for people without a lot of programming knowledge. Because it is simple to adapt to variations in column names and data formats among CSV files, our solution is adaptable and efficient. However, manual intervention may be required to correct errors because the precision and efficacy of LLM-generated code may not always be optimum, especially for more complex queries.

Furthermore, LLM-generated code could not always be as effective as hand-written code, which might cause processing times for bigger datasets to take longer.

Overall, we think that these methods will develop into more potent tools for data integration and transformation as LLM technology continues to advance. LLMs have a lot of promise for use in data integration and transformation tasks, including lowering the requirement for programming knowledge and boosting productivity. Future work may involve extending our method to handle data types other than CSV files and enhancing the precision and effectiveness of the Python code generated by LLM.

REFERENCES

[1] Ghosh, P.K., Dey, S., Sengupta, S.: Automatic sql query formation from natural language query. International Journal of Computer Applications 975, 8887 (2014)

[2] Bais, H., Machkour, M., Koutti, L.: A model of a generic natural language interface for querying databases. international journal of intelligent systems and applications 8(2), 35 (2016)

[3] Sawant, A., Raina, R., Patil, A., Pardeshi, A.: Ai model to generate sql queries from natural language instructions through voice. In: Journal of Physics: Conference Series, vol. 2273, p. 012014 (2022). IOP Publishing

[4] U. Brunner and K. Stockinger, "ValueNet: A Natural Language-to-SQL System that Learns from Database Information," *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, Chania, Greece, 2021, pp. 2177-2182, doi: 10.1109/ICDE51399.2021.00220.

[5] Baig, Muhammad & Imran, Azhar & Yasin, Aman & Butt, Abdul & Khan, Muhammad & Imran, Imran. (2022). Natural Language to SQL Queries: A Review. International Journal of Innovations in Science and Technology. 4. 147-162. 10.33411/IJIST/2022040111.

[6] Hättasch, B., Truong-Ngoc, M., Schmidt, A., & Binnig, C. (2022). It's AI Match: A Two-Step Approach for Schema Matching Using Embeddings. *arXiv preprint arXiv:2203.04366*.

[7] Wu, Bo, Pedro Szekely, and Craig A. Knoblock. "Learning data transformation rules through examples: Preliminary results." Proceedings of the Ninth International Workshop on Information Integration on the Web. 2012.

[8] Kumar, S. M., Majumder, D., Naragunam, A. S., & Ashoka, D. V. A Symmetrically Diminished Interconnected Database Segmentation Framework Using Data Mining. In *Journal of Physics: Conference Series* (Vol. 1964, No. 4, p. 042071). IOP Publishing,2021

[9] Sabarmathi, G., and R. Chinnaiyan. "Investigations on big data features research challenges and applications." 2017 International Conference on Intelligent Computing and Control Systems (ICICCS). IEEE, 2017.

[10] Ilango, V., R. Subramanian, and V. Vasudevan. "A five step procedure for outlier analysis in data mining." European Journal of Scientific Research 75.3 (2012): 327-339.

[11] Calvanese, Diego, et al. "Data integration in data warehousing." International Journal of Cooperative Information Systems 10.03 (2001): 237-271

[12] Balakrishna, Sivadi, M. Thirumaran, and Vijender Kumar Solanki. "IoT sensor data integration in healthcare using semantics and machine learning approaches." *A handbook of internet of things in biomedical and cyber physical system*. Springer, Cham, 2020. 275-300.