Proceedings of the 3rd International Conference on Software Engineering and Machine Learning
DOI: 10.54254/2755-2721/2025.21094

Application of a Large Language Model Incorporating Semantic Information of Data in the SQL Generation Task

Nianlong Zhang

School of Computer Science and Technology, Zhejiang Normal University, Jinhua, China
znl_0322@163.com

Abstract: In the era of big data, data analysis is crucial to enterprise decision-making, but the traditional SQL query writing poses a challenge for non-professionals. With the rapid development of NLP technology, large language models (LLM) such as BERT and GPT have shown strong capabilities. In this paper, the template filling method is used to study the NL2SQL task in the single table scenario and design query templates containing multiple slot bits. The experimental results show that the constructed model achieves high logical form accuracy (LX) and execution accuracy (EX) in both the validation set and the test set, and improves the fault tolerance through fuzzy matching. In addition, the model also performs well in each sub-task, and the introduction of word and table field similarity (sim) further improves the accuracy of conditional value prediction.

Keywords: large language model, SQL generation, data semantic information, natural language processing

1. Foreword

In the era of big data, data analysis has become an important support for enterprise decision-making. However, traditional data analysis processes often require professionals to write complex SQL query statements [1], which is a huge challenge for non-professionals. With the rapid development of natural language processing (NLP) technology, large language models (Large Language Model, LLM) such as BERT, GPT and so on have shown strong capabilities in many fields [2]. This paper will explore the application of large language model of data semantic information in SQL generation task, and reveal how it can change the pattern of data analysis industry.

2. Introduction of the large language model

Large Language Models (LLMs) are deep learning-based natural language processing models that learn syntax and semantics to generate human-readable text [3]. Trained on large-scale corpora (e.g., BERT, GPT), they excel in natural language understanding, generation, and translation. The core technologies include deep learning, neural networks (e.g., RNN, LSTM, Transformer), and the self-attention mechanism. These technologies enable feature extraction, pattern recognition, and contextual understanding by processing sequential data and capturing long-range dependencies [4].

3. And the core method and experimental dataset

3.1. The core method

Template filling is a common method for NL2SQL tasks in single-table scenarios, relying on predefined templates and filling rules to split SQL generation into subtasks. This method clarifies query language and structure, avoiding complexities of multi-table scenarios. We adopt this method for single-table NL2SQL tasks. The template captures key information with slots for SQL components (e.g., table names, fields, operators). The template form is:

SELECT [Column 1, Column 2, ...] FROM [Table name] WHERE [condition 1, condition 2, ...].

Candidate values for fields and operators are predefined based on the database structure, including aggregation functions (AGG), columns (COL), operators (OP), conditional values (VALUE), and connectors($CONN). We use the Chinese-BERT-WWM model to encode natural language queries and database fields into context-dependent vector representations for semantic resolution in SQL generation.

3.2. The experimental dataset

The Spider dataset, a highly representative SQL dataset, was used in this experiment. It contains over 200 databases across various fields, with around 11,000 natural language queries and corresponding SQL statements. These queries cover complex scenarios like single-table queries, multi-table joins, subqueries, and aggregations, providing robust data for model training and evaluation.

Additionally, a specialized test set was constructed to assess the model's adaptability to different database environments. This test set includes variations in relational databases such as MySQL and PostgreSQL, covering differences in table naming, column data types (e.g., DATETIME vs. TIMESTAMP), and primary/foreign key settings. Experiments on this test set evaluate the model's flexibility in diverse real-world database scenarios.

4. Comparative model analysis

4.1. Comparison model

To comprehensively evaluate the performance of our constructed models, several representative traditional SQL generation methods and large language models without fused semantic information of the data were selected for comparison.

Rule-based SQL generation method: This method generates SQL statements using predefined syntax and semantic parsing rules. For example, "Find an employee older than 30" maps "find" to SELECT and "older than 30" to a WHERE clause condition. However, it struggles with complex natural language and diverse database schemas due to rule limitations.

Template-based SQL generation method: This method uses predefined SQL templates and fills them based on query keywords and syntax. For example, "Count the workforce for each department" matches a template like SELECT Department, COUNT (*) FROM Employee Table GROUP BY Department. Its performance depends on the completeness of the template library and struggles with special queries or new database schemas.

GPT-3 model without data and semantic information fusion: Although GPT-3 excels in natural language processing, it faces challenges in accurately mapping natural language queries to database tables and columns in SQL generation tasks due to the lack of data semantic fusion. This is particularly evident in complex queries involving multiple tables.

4.2. Evaluation indicators

Logical form accuracy (LX) and execution accuracy (EX) are used as the key evaluation indicators.

Logical form accuracy (LX): This metric measures the matching degree of the generated SQL statements with the real SQL statements in terms of grammatical structure and logical expressions. For example, errors in the spelling of column names in the SELECT clause or incorrect use of conditional operators in the WHERE clause will reduce LX accuracy. It focuses solely on the syntax and logical correctness of the SQL statement, without considering the execution results in the actual database environment.

Execution accuracy (EX): This index focuses on the consistency between the results obtained by the generated SQL statement and the real results in the actual database. Even if the SQL syntax is correct, EX accuracy is affected if the table or column selection is wrong, leading to incorrect execution results. For example, when querying "Find out the employee in the sales department," if the table name is incorrect, the execution result will be wrong, thus reducing EX accuracy.

5. Experimental results

It is clearly seen from the above experimental results that the large language model integrating the semantic information of data constructed in this paper is significantly higher in both logical form accuracy and execution accuracy than other comparison models in the validation set and test set. In the validation set, the accuracy of the logic form reached 0.85, which increased by 25 percentage points compared with the rule-based method, and the execution accuracy reached 0.80, an increase of 25 percentage points. It also performs well on the test set, which fully proves that the semantic information of the fusion data and the template filling method can effectively enhance the performance of large language models in SQL generation tasks.

Table 1: Large language models incorporating semantic information of the data in the validation set and test set analysis

model	validation set LX	validation set EX	test set LX	test set EX
Rule-based approach	0.60	0.55	0.58	0.53
Template-based approach	0.65	0.60	0.63	0.58
GPT-3 without fused semantic information of the data	0.70	0.65	0.68	0.63
The model constructed in this paper	0.85	0.80	0.83	0.78

Still generating efficient SQL statements in the face of the imprecision of the natural language input, we introduce a fuzzy matching mechanism. When matching keywords in natural language with database table fields, a fuzzy matching algorithm based on edit distance (such as Levenshtein distance) is used. The algorithm measures the similarity of two strings by calculating the minimum number of editing operations (insertion, delete, replace). For example, when an Employee Name appears in natural language and the field name is "employee_name", the fuzzy matching mechanism calculates the edit distance between the Employee Name and all the field names in the database. Assuming that the edit distance between "employee name" and "employee_name" is 3, and the edit distance from other field names is greater than 3, then the model can identify the association of the two. In this way, the model can match according to the similarity even if the keywords are not exactly consistent with the fields. In practical tests, when entering some natural language queries with misspelling or irregular representation, the proportion of models that introduce fuzzy matching mechanism generating correct or approximately correct SQL statements is increased by about 15%, compared with those that were not introduced, which greatly enhances the practicability of the model in practical application.

The performance of the model on each subtask is deeply analyzed, including column selection, table selection, condition generation, and conditional value prediction.

Table 2: Performance on each subtask

subtask	precision
Column selection	0.90
table select	0.95
Conditional generation	0.88
Prediction of conditional values	0.85 (before sim); 0.90 (after sim)

The model achieved high accuracy in column selection (0.90) and table selection (0.95) subtasks, attributed to its effective integration of data semantics and deep natural language understanding. For instance, given the query "Get product names with prices greater than 100 in the product table," it accurately identified the "Product Table," "Product Name," and "Price" columns. In conditional generation, the model reached an accuracy of 0.88, capable of generating SQL conditional expressions from natural language descriptions, though occasional errors occurred with complex logical relationships.

For conditional value prediction, accuracy improved from 0.85 to 0.90 after introducing a semantic similarity metric (sim) to match natural language values with database values [5]. For example, when the query mentioned "red product" and the database had "Red" in the color field, the similarity calculation enabled accurate matching. This demonstrates that precise alignment of natural language and database fields significantly enhances the model's ability to generate complex conditions.

6. Conclusion

In conclusion, large language models incorporating semantic information of data show significant advantages in SQL generation tasks. Through the template filling method, the model can effectively convert natural language queries into SQL statements, meeting the needs of non-professionals for data analysis. The experimental results show that the constructed model achieves satisfactory results in both logical form accuracy and execution accuracy, and improves the fault tolerance through fuzzy matching. Moreover, the high accuracy of the model on each subtask also demonstrates its powerful natural language understanding and generation capabilities.

References

[1] Zhang Q , Dong J , Chen H ,et al.Structure Guided Large Language Model for SQL Generation[J]. 2024.

[2] Wu Z .Large Language Model Based Semantic Parsing for Intelligent Database Query Engine[J].Journal of Computer and Communications, 2024, 12(10):13.DOI:10.4236/jcc.2024.1210001.

[3] Di Martino B , Graziano M , Colucci Cante L .Semantic, Business Process andNatural Language Processing foreBuilding[C]//International Conference on Complex, Intelligent, and Software Intensive Systems.Springer, Cham, 2024.DOI:10.1007/978-3-031-70011-8_35.

[4] Peng H .Chinese Natural Language Processing: From Text Categorization to Machine Translation[J].Applied Mathematics and Nonlinear Sciences, 2024, 9(1).DOI:10.2478/amns-2024-1860.

[5] Nakpih C I .A modified Vector Space Model for semantic information retrieval[J].Natural Language Processing Journal, 2024, 8.DOI:10.1016/j.nlp.2024.100081.