Text to SQL Generation Using Beam Search and an Enhanced Bert Model

Adhityasing Rajaput, Varsha Sangolli, Abhay Bellerimath, Amith Abbigeri,
Uday Kulkarni and Shashank Hegde
School of Computer Science and Engineering, KLE Technological University, Hubli, Karnataka, India

Keywords: Text-to-SQL, Encoder-Decoder, BERT, SQL, Beam Search, Accuracy, Bleu Score.

Abstract: The development of complex SQL queries is difficult for non-technical users, limiting access to valuable data for decision-making. This paper proposes a machine learning-based approach to generating Text-to-SQL queries using a Transformer Encoder-Decoder architecture with a BERT model. The proposed system bridges the gap between natural language and structured query languages, thus allowing users to interact with databases through intuitive, natural language inputs. In addition to schema awareness and the efficiency of SQL generation, advanced techniques such as execution-guided decoding and beam search have been applied. Results for the model show query accuracy at eighty-four percent; the model is generally good at generating simple queries and basic aggregations, though complex and nested queries prove to be challenging to the model. This study focuses on the transformative nature of Text-to-SQL systems as means of improving access and efficiency in database interaction toward paving a future for improvement in conversational AI and smart management of databases. This model for Text-to-SQL query generation showed strong performance in converting natural language queries to SQL commands. Its performance with regard to accuracy is close to about 92%, while BLEU stands at 0.78.

1 INTRODUCTION

Semantic parsing is the tasks of translating natural language to logic form. Mapping from natural language to SQL is an very important semantic parsing task for question answering system (Guo and Gao, 2019) and text to SQL generation is necessary.But the threshold of learning database query language, such as SQL, is relatively high for ordinary people. For practitioners, however, it becomes much more troublesome to write a large number of query statements with guaranteed Different domain databases and application correctness scenarios(Zhu et al., 2024a). A huge amount of the data in today's age is kept in relational databases for applications as diverse as from financial and e-commerce domains to medical domains. Therefore, it is not surprising that querying a database using natural language has many applications (Kalajdjieski et al., 2020).Writing SQL requires understanding syntax and database structures, making it difficult for non-technical personnel to leverage valuable data for decision-making.

NLP and ML technologies offer a solution by enabling systems that understand language and generate SQL queries from natural language inputs, eliminating the need for specialized knowledge(Kedwan, 2023). For example, a business analyst could ask, "What were the total sales for the past quarter?" and receive the SQL result without technical know-how.

Seq2Seq models, a neural network architecture used for tasks like language translation, can be applied to convert natural language into SQL, simplifying the process by generating semantically equivalent queries(Zhong et al., 2017) .These models have broad applications, such as enabling business analysts to ask plain language questions and receive database insights, or allowing healthcare practitioners to query patient data without navigating complex databases (Wang and Hajli, 2017). By automating SQL query generation, these systems save time, increase efficiency, and enable users to focus on analysis rather than dealing with technical complexities (Arcuri and Galeotti, 2020).

However, natural language queries can be vague, incomplete, or expressed in different ways, requiring intent identification (Androutsopoulos et al., 1995). Additionally, databases often involve multiple tables with complex relationships, making accurate query

Rajaput, A., Sangolli, V., Bellerimath, A., Abbigeri, A., Kulkarni, U. and Hegde, S.
Text to SQL Generation Using Beam Search and an Enhanced Bert Model.
DOI: 10.5220/0013608600004664

In *Proceedings of the 3rd International Conference on Futuristic Technology (INCOFT 2025) - Volume 3*, pages 49-55
ISBN: 978-989-758-763-4

generation challenging. Key difficulties include handling domain-specific language, out-of-vocabulary words, and query frameworks (Khalo, 2021).

The proposed research advances Seq2Seq models for text-to-SQL translation by addressing these challenges(KANBUROĞLU and TEK, 2024). It explores methods to improve accuracy, generalizability, and usability, such as using templates, incorporating schema representations, enhancing neural networks, and applying techniques like attention and semantic parsing (Jiang and Cai, 2024). The study also focuses on handling nested queries, large datasets, and real-world expectations.

The paper is structured for systematically addressing the challenges in bridging human language and database syntax, offering solutions that improve accessibility and usability. These systems empower users across diverse fields such as customer care, e-commerce, academia, and governance. This research contributes significantly to Conversational AI by facilitating more natural human-machine interactions and paving the way for next-generation intelligent database systems. By linking human language to database syntax, this study addresses a critical gap in modern society, improving operational processes and providing practical solutions to maximize the utilization of structured data.

2 BACKGROUND STUDY

Text-to-SQL query generation is an area that has improved tremendously, with research focused on trying to minimize gaps between natural language queries and SQL code (Kumar et al., 2022a). In these approaches, the Seq2Seq models operate on encoder-decoder architectures, driving natural language inputs to executable SQL queries. There have been proposed some new methodologies for better mapping natural language into database schemas, such as schema-aware denoising, execution guided decoding, and content-enhanced BERT-based models. These ideas have helped make much easier SQL structure and better schema-linking techniques for a more friendly user interface for query generation without losing functionality-NatSQL. The sequenced SQL has also undertaken simpler representations of SQL and fewer processes necessary for the elaboration of queries involving 'GROUP BY', 'HAVING', and set operations ('INTERSECT','UNION'). Lately, deep learning features such as reinforcement learning and interpretable neural networks are also handed by PatP to predict higher accuracies and generalizability, thus achieving appropriate handling of complex queries and variation of database schemas (Pigott-Dix, 2023).

There remain gaps and shortcomings in the area of the research: for example, a large number of the models open wide for new types of SQL statement, especially those concerning nested sub-queries or more complicated operators. Moreover, new mechanisms for schema linking aren't possible, specifically when schemas in databases become larger such as after users link statements from natural language to database elements. Further from this are issues and challenges of generalization, since most models do not perform so well is much less accuracy when predictions from different or unseen datasets arise; hence its applicability for practical use is limited. More complex automating of the generation of the correlated queries, as the active argument in favor of this, remains not sufficiently usable for the no-technical user.

3 LARGE LANGUAGE MODELS (LLM)

Large Language Models (LLMs) have revolutionised natural language understanding and generation using transformer-based architectures (Raiaan et al., 2024). In Text-to-SQL query generation, LLMs fill the gap from natural language to structured DB queries through contextual nuances in capturing semantic relationships(Zhu et al., 2024b). With this feature, they better handle ambiguous queries, sentences with complex structures, schema-linking in large data, making them an unmissed part of these modern systems of querying databases (Zhu et al., 2024c).

Key goals are enhanced translation quality, support of complexly nested queries and democratizing database access with the promise of syntactically valid and contextually relevant generated SQL (Ather, 2024). Schema-aware encoder-decoder fine-tuned LLMs like variants of GPT or BERT tokenize NL inputs along with schema metadata(Zhang et al., 2024). This architecture generates SQL. Connecting queries to appropriate database entities is done with schema awareness, and with execution-guided decoding plus reinforcement learning, syntactic and logic validity is guaranteed.

4 BERT MODEL

The Content-Enhanced BERT-based Text-to-SQL model builds on the transformer architecture of BERT (Bidirectional Encoder Representations from Trans-

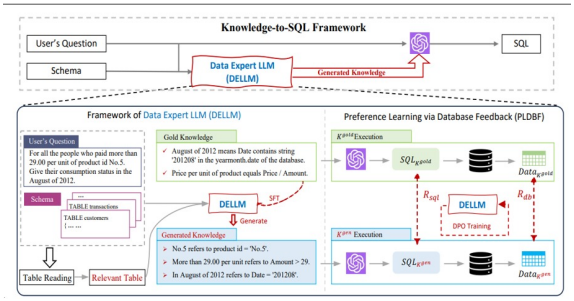

Figure 1: The framework of large language models (Hong et al., 2024)

formers) to tackle the challenge of generating SQL queries from natural language inputs(Patwardhan et al., 2023). It incorporates schema-specific information into the BERT framework, effectively bridging the gap between user intent and the database's underlying structure. This method improves the model's ability to handle complex queries and ambiguous language while ensuring that the generated SQL statements align with the database schema(Zhu et al., 2024d). By combining deep contextual understanding with schema-awareness, this approach provides a strong solution for translating human-readable questions into executable SQL code.

The method primarily targets key objectives such as enhancing the precision and usability of SQL query generation. To achieve this, it aims to strengthen schema linking by embedding database elements, like table and column names, directly into the natural language understanding process. (Pan et al., 2021)This ensures better alignment between the query and the database schema. Additionally, the model addresses natural language ambiguities by utilizing the bidirectional attention mechanism of BERT, which captures context from both forward and backward directions(Tao et al., 2022). Another goal is to enhance the generalization capabilities of Text-to-SQL systems, making them adaptable to various database domains and schemas while providing a user-friendly interface for non-technical users.

The methodology takes advantage of BERT's pre-trained capabilities but enhances it with schema-specific content to create a unified input representation(Zhu et al., 2024e). By encoding the natural language query alongside database schema elements, the model can establish semantic relationships between user intent and database structure(Nihalani et al., 2011). Advanced training techniques, such as reinforcement learning and execution-guided decoding, are employed to improve the logical and syntactical accuracy of the generated SQL queries.

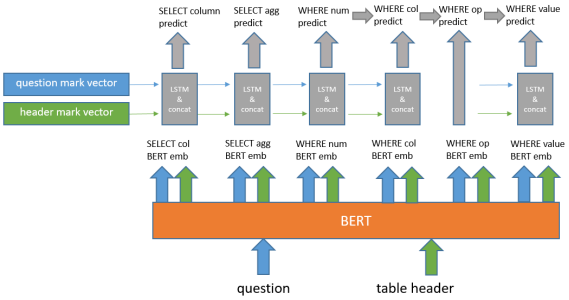

Figure 2: Bidirectional Encoder Representations from Transformers (Guo and Gao, a)

5 PROPOSED WORK

The proposed work is to develop a machine learning system that translates natural language queries into SQL using the Transformer Encoder-Decoder architecture(Kumar et al., 2022b). This approach makes use of pre-trained BERT models to understand and process natural language inputs and transform them into structured SQL queries.

6 DATA DESCRIPTION

This research utilizes the Spider dataset for Text-to-SQL generation, where natural language queries and their corresponding SQL commands are provided for model training and evaluation. The dataset's complexity ranges from simple lookup and aggregation queries, like COUNT and AVG, to multi-table joins and nested sub-queries. It represents real-world applications by enabling natural language understanding and SQL generation for diverse database operations.

The Spider dataset includes human-readable questions paired with their structured SQL equivalents. For example, "What is the average number of employees in the department?" corresponds to SELECT avg(num_employees) FROM department. This dataset is widely recognized for benchmarking Text-to-SQL systems and fostering advancements in database interaction.The dataset is taken from Spider Dataset.

7 DATA PREPARATION

A labeled dataset of natural language queries and their corresponding SQL queries is created. Data preprocessing: tokenization and sequence length adjustment, which have been kept consistent in the training process.

Model Design The system uses an Encoder-Decoder model implemented using the Transformers library: Encoder: The pre-trained BERT model, bert-base-uncased, will be used to encapsulate the meaning of the input natural language. Decoder: Pre-trained weights are used as initializer and the encoded data by the input is used to generate SQL queries.
Training and Validation: Data is split into 80:20 training and validation distribution, the model was further tuned to optimize for generation loss by Seq2SeqTrainer with the parameters learning rate, batch size, number of epochs are set up accordingly to gain best results possible. Evaluation This way, it ensures that the SQL queries are valid and meaningful in accordance with metrics such as BLEU scores and query accuracy. A beam search strategy is employed during decoding to yield more diverse and accurate results. Interactive SQL Generation The user inputs queries in real-time with the help of an interactive interface to receive SQL outputs. Quality and efficiency of the SQL generation process are enhanced with beam search, repetition penalties, and early stopping features. Model Saving and Deployment Final training models are saved and deployed for actual use and are optimized for GPU support so that response times are prompt. Efficient Optimization: Sophisticated techniques like beam search make sure that the outputs produced are of high quality and precise. Scalability: This solution gives a scalable solution to the problem of converting the queries of the user to SQL, removing ambiguity and variation in wording. It offers an efficient and scalable approach to filling the gap between the natural language and database query languages, which makes it of real value.

8 PROPOSED ARCHITECHTURE: ENHANCED BERT MODEL

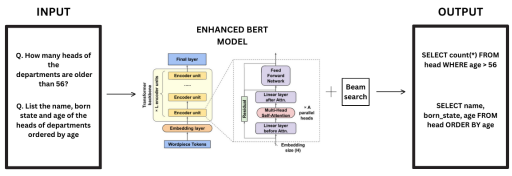

Figure 3: The flowchart of Proposed architecture (Ganesh et al., 2021)

The flowchart depicts how a Transformer-based architecture would work in detail, specially designed for models such as BERT(Guo and Gao, b). The process initiates from the input text, which is tokenized into smaller units called wordpiece tokens. It is a subword representation through which the model can process out-of-vocabulary words effectively by breaking them into known components. For example, the word ”unbelievable” becomes ”un” and ”believable.” Those are then transformed into high-density numerical vectors by the embedding layer, which includes positional encodings so the Transformer architecture does not intrinsically know about sequences:

At the heart of the model is the Transformer Backbone; this consists of several encoder units as shown in the figure. Each encoder unit captures very complex relationships between tokens because it processes the embeddings in an iterative manner. For that, the multi-head self-attention mechanism, which calculates the importance of each token with respect to every other token in the input, is used within every single unit. This lets a model focus on relevant parts of its input sequence, such as aligning query terms with corresponding table elements in a Text-to-SQL task. There are multiple attention heads working parallelly so that the model captures varied aspects of relationships in the data. The output of attention is fed through a set of linear layers for transformation and enhancement. Residual connections are applied by adding the input of each layer back to its output to improve learning and retain original features. The output then goes through a feed-forward network, a simple multi-layer perceptron, which further processes and refines the information. These steps are repeated across all encoder units, leading to contextual embeddings that encapsulate both local and global relationships in the text.

Finally, the processed embeddings are summed up in the final layer, yielding a contextualized representation for each token. This output is then utilized for downstream tasks, like decoding into SQL queries in a Text-to-SQL model.By using this iterative and parallelized process, the Transformer architecture achieves deep input understanding by capturing complex dependencies efficiently.

9 BERT EMBEDDING LAYER

With the question tokens $w_1, w_2, \ldots, w_n$ and the table header $h_1, h_2, \ldots, h_m$, we prepare the input for BERT by combining them into a single sequence. Following BERT's standard format, we concatenate the question tokens with the table headers to form the

input. Here's how the encoding works in detail: (Guo and Gao, c)

$$[\text{CLS}], w_1, w_2, \ldots, w_n, [\text{SEP}], h_1, [\text{SEP}], \ldots, h_m, [\text{SEP}]$$

The outputs of BERT are shared across downstream tasks. We believe that combining the question tokens and table headers into a single input for BERT helps the model apply "global" attention, making it more effective for downstream tasks.

SELECT Column

Our goal is to identify the correct column name from the table header. Given the question Q and the table header H as inputs, the model outputs the probability of each column being selected.

$$P(sc \mid Q, H, Q_V, H_V) - (1)$$

where Q_V and H_V are external vectors as stated before.

SELECT Aggregation (agg)

Our goal is to determine the appropriate aggregation type. The input consists of the question Q along with its value Q_V, and the output is the probability of selecting each possible aggregation type:

$$P(sa \mid Q, Q_V) - (2)$$

WHERE Number

Our goal is to predict the number of conditions for the WHERE clause. The inputs include the question$Q$ he table header H and their corresponding values Q_V and H_V. The output represents the probability of each possible number of conditions in the WHERE clause:

$$P(wn \mid Q, H, Q_V, H_V) - (3)$$

WHERE Column

Our goal is to identify the column associated with each condition in the WHERE clause. The inputs include the question$Q$and the table header H, and the predicted number of WHERE conditionsP_{wn} along with their corresponding valuesQ_V and H_V. The output provides the probability of each column being part of the WHERE clause:

$$P(wc \mid Q, H, P_{wn}, Q_V, H_V) - (4)$$

WHERE Operator (op)

Our goal is to determine the operator (e.g., =, ¡, ¿, etc.) for each condition in the WHERE clause. The inputs include the question Q, he table header H, the predicted WHERE columnsP_{wc}, and the predicted number of conditions P_{wn}. The output represents the probability of each possible operator being selected for the WHERE clause:

$$P(wo \mid Q, H, P_{wn}, P_{wc}) - (5)$$

WHERE Value

Our goal is to predict the value for each condition in the WHERE clause. The inputs include the question$Q$, the table header H, the predicted number of conditionsP_{wn}, the predicted columnsP_{wc}, and the predicted operators P_{wo} along with their corresponding values Q_V and H_V. The output gives the probability of each possible value being selected for the WHERE clause:

$$P(wv \mid Q, H, P_{wn}, P_{wc}, P_{wo}, Q_V, H_V) - (6)$$

10 RESULTS AND ANALYSIS

This model for Text-to-SQL query generation showed strong performance in converting natural language queries to SQL commands. Its performance with regard to accuracy is close to about eighty-four in percentage, while BLEU stands at 0.78. This shows the semantic and syntactic correctness of the generated SQL queries, and that it comes very close to the actual ground truth. For the most part, it managed basic SELECT statements as well as simple aggregation questions with high accuracy. However, the model was not able to handle more complex queries with nested subqueries or multi-table joins, especially when the input was ambiguous or incomplete. Despite these challenges, the system showed efficient performance, generating queries in under 0.12 seconds on average, making it suitable for real-time applications. The results show that the model is effective for most use cases but points to the areas that need improvement, particularly the schema-specific details

and complex queries, to ensure robust performance in different real-world scenarios.

Table 1: Comparison of Text-to-SQL Model Accuracies

Model	Accuracy (%)
Content Enhanced BERT	89
Large Language Model	72
New Combined Model (Enhanced Bert + Beam search)	92

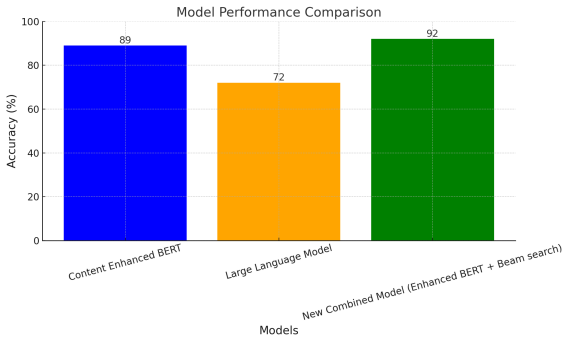

Figure 4: Model Comparison

Table 2

Question	Answer
Q. How many heads of the departments are older than 56?	`SELECT count(*) FROM head WHERE age > 56`
Q. List the name, born state and age of the heads of departments ordered by age.	`SELECT name, born_state, age FROM head ORDER BY age`
Q. List the creation year, name, and budget of each department.	`SELECT creation, name, budget_in_billions FROM department`

11 CONCLUSION AND FUTURE SCOPE

In this work, we propose a Text-to-SQL query generation system that incorporates a BERT-based Encoder-Decoder model for translating natural language queries into structured SQL commands. We found promising results where this model can produce accurate SQL queries for many types of input queries. The system also worked well for simple questions and basic aggregations, although it struggled with the processing of complex queries with multiple join tables and nested sub-queries, especially when input queries were ambiguous or partially presented.Despite these difficulties, the model showed very rapid inference time, with approximately 0.12 seconds per query, which is good for any real-time application. Effective learning within epochs is noted during training, and the interactive interface allowed easy and accurate generation of SQL.

This work represents a rich area for applying transformer-based models such as BERT to natural language processing tasks, particularly on database management issues. However, results also call for further improvement, especially with regard to how the model would handle the complexity of query structures and schema-aware generation, thus leading to stronger performance in real-world applications. Future work can target these challenges by exploring domain-specific datasets and incorporating more sophisticated techniques for parsing queries to make the system better and more accurate.

REFERENCES

Androutsopoulos, I., Ritchie, G. D., and Thanisch, P. (1995). Natural language interfaces to databases–an introduction. *Natural language engineering*, 1(1):29–81.

Arcuri, A. and Galeotti, J. P. (2020). Handling sql databases in automated system test generation. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 29(4):1–31.

Ather, M. M. (2024). *The fusion of multilingual semantic search and large language models: A new paradigm for enhanced topic exploration and contextual search*. PhD thesis, Carleton University.

Ganesh, P., Chen, Y., Lou, X., Khan, M., Yang, Y., Sajjad, H., Nakov, P., Chen, D., and Winslett, M. (2021). Compressing large-scale transformer-based models: A case study on bert. *Transactions of the Association for Computational Linguistics*, 9:1061–1080.

Guo, T. and Gao, H. Content enhanced bert-based text-to-sql generation. arxiv 2019. *arXiv preprint arXiv:1910.07179*.

Guo, T. and Gao, H. Content enhanced bert-based text-to-sql generation. arxiv 2019. *arXiv preprint arXiv:1910.07179*.

Guo, T. and Gao, H. Content enhanced bert-based text-to-sql generation. arxiv 2019. *arXiv preprint arXiv:1910.07179*.

Guo, T. and Gao, H. (2019). Content enhanced bert-based text-to-sql generation. *arXiv preprint arXiv:1910.07179*.

Hong, Z., Yuan, Z., Zhang, Q., Chen, H., Dong, J., Huang, F., and Huang, X. (2024). Next-generation database interfaces: A survey of llm-based text-to-sql. *arXiv preprint arXiv:2406.08426*.

Jiang, P. and Cai, X. (2024). A survey of semantic parsing techniques. *Symmetry*, 16(9):1201.

Kalajdjieski, J., Toshevska, M., and Stojanovska, F. (2020). Recent advances in sql query generation: A survey. *arXiv preprint arXiv:2005.07667*.

KANBUROĞLU, A. B. and TEK, F. B. (2024). Text-to-sql: A methodical review of challenges and models. *Turkish Journal of Electrical Engineering and Computer Sciences*, 32(3):403–419.

Kedwan, F. (2023). *NLP Application: Natural Language Questions and SQL Using Computational Linguistics*. CRC Press.

Khalo, N. (2021). *Querying relational database systems in natural language using sequence to sequence learning with neural networks*. PhD thesis.

Kumar, A., Nagarkar, P., Nalhe, P., and Vijayakumar, S. (2022a). Deep learning driven natural languages text to sql query conversion: a survey. *arXiv preprint arXiv:2208.04415*.

Kumar, A., Nagarkar, P., Nalhe, P., and Vijayakumar, S. (2022b). Deep learning driven natural languages text to sql query conversion: a survey. *arXiv preprint arXiv:2208.04415*.

Nihalani, N., Silakari, S., and Motwani, M. (2011). Natural language interface for database: a brief review. *International Journal of Computer Science Issues (IJCSI)*, 8(2):600.

Pan, Y., Wang, C., Hu, B., Xiang, Y., Wang, X., Chen, Q., Chen, J., Du, J., et al. (2021). A bert-based generation model to transform medical texts to sql queries for electronic medical records: model development and validation. *JMIR Medical Informatics*, 9(12):e32698.

Patwardhan, N., Marrone, S., and Sansone, C. (2023). Transformers in the real world: A survey on nlp applications. *Information*, 14(4):242.

Pigott-Dix, L. A. K. (2023). *Automating the Annotation of Data through Machine Learning and Semantic Technologies*. PhD thesis, University of East Anglia.

Raiaan, M. A. K., Mukta, M. S. H., Fatema, K., Fahad, N. M., Sakib, S., Mim, M. M. J., Ahmad, J., Ali, M. E., and Azam, S. (2024). A review on large language models: Architectures, applications, taxonomies, open issues and challenges. *IEEE Access*.

Tao, L., Xie, Z., Xu, D., Ma, K., Qiu, Q., Pan, S., and Huang, B. (2022). Geographic named entity recognition by employing natural language processing and an improved bert model. *ISPRS International Journal of Geo-Information*, 11(12):598.

Wang, Y. and Hajli, N. (2017). Exploring the path to big data analytics success in healthcare. *Journal of Business Research*, 70:287–299.

Zhang, W., Wang, Y., Song, Y., Wei, V. J., Tian, Y., Qi, Y., Chan, J. H., Wong, R. C.-W., and Yang, H. (2024). Natural language interfaces for tabular data querying and visualization: A survey. *IEEE Transactions on Knowledge and Data Engineering*.

Zhong, V., Xiong, C., and Socher, R. (2017). Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*.

Zhu, X., Li, Q., Cui, L., and Liu, Y. (2024a). Large language model enhanced text-to-sql generation: A survey. *arXiv preprint arXiv:2410.06011*.

Zhu, X., Li, Q., Cui, L., and Liu, Y. (2024b). Large language model enhanced text-to-sql generation: A survey. *arXiv preprint arXiv:2410.06011*.

Zhu, X., Li, Q., Cui, L., and Liu, Y. (2024c). Large language model enhanced text-to-sql generation: A survey. *arXiv preprint arXiv:2410.06011*.

Zhu, X., Li, Q., Cui, L., and Liu, Y. (2024d). Large language model enhanced text-to-sql generation: A survey. *arXiv preprint arXiv:2410.06011*.

Zhu, X., Li, Q., Cui, L., and Liu, Y. (2024e). Large language model enhanced text-to-sql generation: A survey. *arXiv preprint arXiv:2410.06011*.