LTRC-IIITH at EHRSQL 2024: Enhancing Reliability of Text-to-SQL Systems through Abstention and Confidence Thresholding

Jerrin John Thomas, Pruthwik Mishra, Dipti Sharma, Parameswari Krishnamurthy
LTRC, International Institute of Information Technology, Hyderabad, India
`{jerrin.thomas, pruthwik.mishra}@research.iiit.ac.in`
`{dipti, param.krishna}@iiit.ac.in`

Abstract

In this paper, we present our work in the EHRSQL 2024 shared task which tackles reliable text-to-SQL modeling on Electronic Health Records. Our proposed system tackles the task with three modules - abstention module, text-to-SQL generation module, and reliability module. The abstention module identifies whether the question is answerable given the database schema. If the question is answerable, the text-to-SQL generation module generates the SQL query and associated confidence score. The reliability module has two key components - confidence score thresholding, which rejects generations with confidence below a predefined level, and error filtering, which identifies and excludes SQL queries that result in execution errors. In the official leaderboard for the task, our system ranks 6th. We have also made the source code public[1].

1 Introduction

Electronic Health Records (EHRs) have revolutionized healthcare by serving as comprehensive digital repositories of medical histories of patients. They capture every step, from initial admission and diagnosis to treatment plans and discharge summaries. While EHRs are invaluable for clinical data storage and retrieval, unlocking their full potential goes beyond basic searches. Traditional methods often necessitate proficiency in Structured Query Language (SQL), a complex hurdle for many healthcare providers, especially those pressed for time. To bridge this gap and make EHR data more accessible, researchers are exploring the development of question-answering systems that leverage text-to-SQL models. These systems empower users to ask questions in plain natural language and receive answers directly retrieved from the EHR data, streamlining the process of extracting valuable insights from patient data.

[1]`https://github.com/jr-john/ehrsql-2024`

In this task (Lee et al., 2024), we tackle the problem of developing a reliable text-to-SQL model tailored for an EHR database, ensuring accurate responses while abstaining from providing incorrect answers. This model must handle a diverse range of topics relevant to clinical settings, such as patient demographics, vital signs, and disease survival rates. The model should accurately generate SQL queries for answerable questions, abstain from providing erroneous answers, and recognize and abstain from addressing unanswerable questions, whether they extend beyond the database schema or are impossible to solve using SQL alone. The spectrum of unanswerable questions also encompasses adversarially crafted queries designed to mislead text-to-SQL models. Successfully tackling this task will yield a robust and scalable question-answering system for EHRs, significantly enhancing how clinicians leverage clinical knowledge.

2 Related Work

With the advent of deep learning models, there has been a renewed interest in generating text-to-SQL models in the medical domain. Wang et al. (2020) tackle the problem by developing a deep learning-based approach that adapts a sequence-to-sequence architecture to directly generate SQL queries for a given question. The model further performs the necessary edits using an attentive copying mechanism and task-specific lookup tables. Additionally, they release a large-scale dataset called MIMIC-SQL that generates SQL queries from questions in this domain.

Several papers utilize pre-trained BERT (Devlin et al., 2019) models, as their foundation blocks for their text-to-SQL systems. These models leverage large amounts of text data to learn effective representations of language in their pre-training stage, which are then fine-tuned for the specific task of EHR-based question answering. Pan et al. (2021)

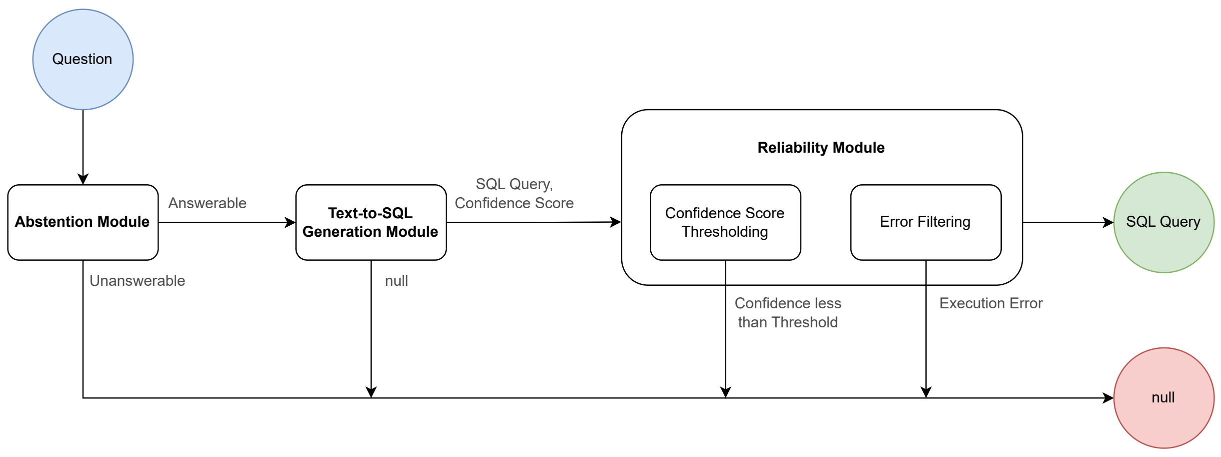
Question
Abstention Module
Answerable
Unanswerable
Text-to-SQL Generation Module
null
SQL Query, Confidence Score
Reliability Module
Confidence Score Thresholding
Error Filtering
Confidence less than Threshold
Execution Error
SQL Query
null

Figure 1: System Workflow

develop a BERT-based model to convert medical text into an intermediate representation that can then be translated into SQL. Gao et al. (2023) explore using open-source Large Language Models (LLMs) and supervised fine-tuning methods for text-to-SQL tasks. Tarbell et al. (2023) investigate the generalizability of the text-to-SQL models across different EHR systems and data formats. They also introduce a data augmentation approach to improve generalizability.

There has been limited prior work in improving the reliability of text-to-SQL systems, and this task aims to address this gap.

3 Dataset

The dataset provided for the shared task is the Medical Information Mart for Intensive Care IV (MIMIC-IV) demo version of EHRSQL with additional unanswerable questions (Lee et al., 2022). MIMIC-IV (Johnson et al., 2023, 2021; Goldberger et al., 2000) is a large database containing de-identified patient information from Beth Israel Deaconess Medical Center. It is a relational database consisting of twenty-six tables, which stores the data collected during routine clinical care, including patient demographics, vital signs, diagnoses, medications, and procedures.

EHRSQL (Lee et al., 2022) is a dataset designed to evaluate and enhance text-to-SQL systems specifically tailored for EHRs. It contains real-world questions gathered from various hospital staff, including doctors, nurses, insurance specialists, and record-keeping personnel. This ensures that the dataset incorporates practical queries healthcare professionals ask daily. The questions range from simple data lookups to complex calculations, such as determining patient survival rates. Recognizing that not all questions have answers within the EHR data, EHRSQL empowers text-to-SQL systems to identify cases when low confidence or missing information necessitates abstaining from a response.

The given dataset consists of 5,124 samples in the training set, 1,163 samples for validation, and 1,167 samples for testing. The database encompasses eighteen different tables adapted from MIMIC-IV. The training data is exclusively used for model training, while the validation data is not utilized during training. No external data sources are incorporated and no data augmentation techniques are employed.

4 System Description

Our system contains three modules - abstention module, text-to-SQL generation module, and reliability module.

4.1 Pre-Trained Model Description

We choose SQLCoder-7b-2, a fine-tuned implementation of CodeLlama-7b (Rozière et al., 2024), as the pre-trained text-to-SQL model.[2] This model outperforms GPT-4 (as of Feb 5, 2024) and GPT-4-Turbo (as of Feb 5, 2024) in text-to-SQL benchmarks.

The model is fine-tuned on a comprehensive SQL curriculum, ranging from basic clauses to underrepresented categories like date functions and advanced operations like window functions. It

[2] https://huggingface.co/defog/sqlcoder-7b-2

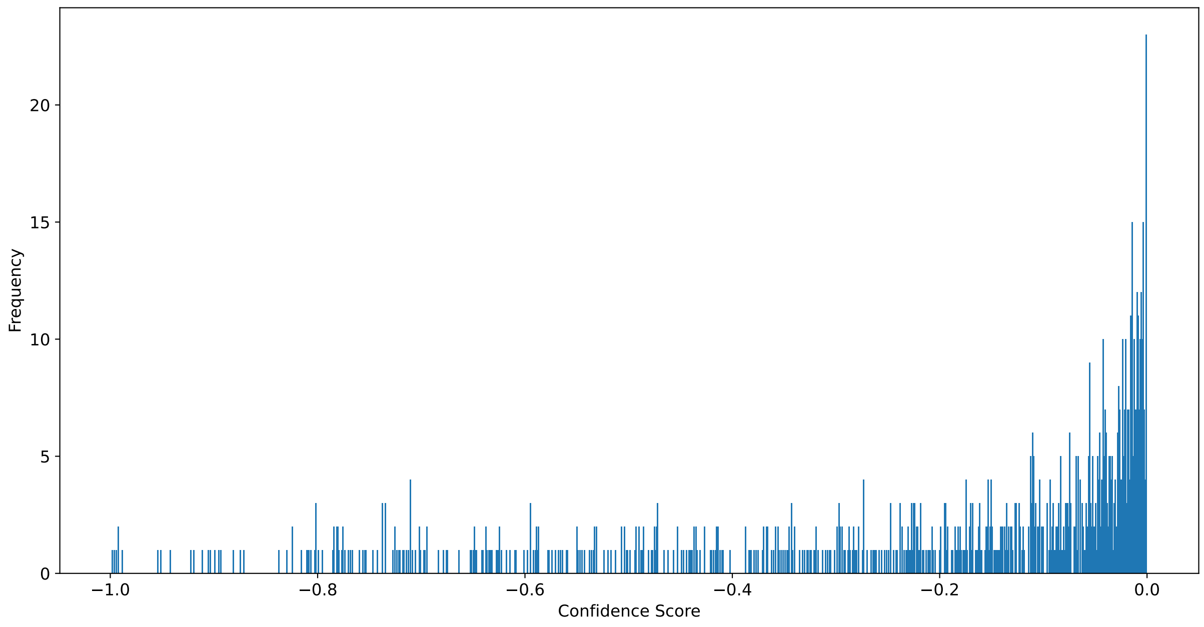

Figure 2: Distribution of confidence scores for test predictions, with scores below -1 excluded

contains hand-crafted SQL queries as well as augmented data from WikiSQL (Zhong et al., 2017). It has thirteen different schemas and questions of varying levels of difficulty. The schemas are quite complex with four to twenty tables. Each question in the dataset has been classified into “easy", “medium", “hard", and “extra-hard" categories. The model is fine-tuned in two stages. First, the base CodeLlama model is fine-tuned on easy and medium questions. Then, the resulting model is fine-tuned on hard and extra-hard questions.

4.2 Text-to-SQL Generation Module

The text-to-SQL generation module has a pre-trained text-to-SQL model that is fine-tuned on the training data for EHR domain adaptation. We use the prompt template of the pre-trained model for fine-tuning (the prompt template is provided). Following the template, we provide the table metadata as DDL (Data Definition Language) commands for creating the tables. For each table in the database, a "CREATE" statement is generated that includes all the fields and information about the primary key. Each field in the database has a descriptive comment explaining its purpose. The comments at the end of the metadata provide information about the foreign key relationships between the tables. These foreign key constraints define the dependencies between the tables.

The training data is processed and prepared for fine-tuning the model. We train this model for two epochs till convergence. If the model cannot answer the question with the available database schema, the system returns "null".

Prompt for Text-to-SQL Generation

Task
Generate a SQL query to answer [QUESTION]{user_question}[/QUESTION]
Instructions
If you cannot answer the question with the available database schema, return ’I do not know’
Database Schema
The query will run on a database with the following schema:
{table_metadata_string}
Answer
Given the database schema, here is the SQL query that answers [QUESTION]{user_question}[/QUESTION]
[SQL]{answer}[/SQL]

4.2.1 Prompt Engineering

We experiment with the prompt by providing the table metadata without any comments. The foreign keys in each table are declared using the "FOREIGN KEY" constraint.

Experiment	RS(0)	RS(5)	RS(10)	RS(N)
Text-to-SQL Generation (Initial Prompt)	33.59	-295.46	-624.51	-76766.41
Text-to-SQL Generation	78.58	-25.96	-130.51	-24321.42
Text-to-SQL Generation + Error Filtering	82.60	13.20	-56.21	-16117.40
Error Filtering				
+ Abstention + Text-to-SQL Generation	**83.55**	25.28	-32.99	-13516.45
+ Abstention (Multi-Task) + Text-to-SQL Generation (Multi-Task)	71.47	-9.94	-91.35	-18928.53
Abstention (Multi-Task) + Text-to-SQL Generation + Error Filtering				
+ Confidence Thresholding (Threshold = -1)	78.66	54.24	29.82	-5621.34
+ Confidence Thresholding (Threshold = -0.5)	69.92	**55.78**	41.65	-3230.07
+ Confidence Thresholding (Threshold = -0.4)	66.84	55.27	**43.70**	-2633.16
+ Confidence Thresholding (Threshold = -0.35)	65.38	54.24	43.10	-2534.61
+ Confidence Thresholding (Threshold = -0.3)	63.92	53.21	42.50	-2436.08
+ Confidence Thresholding (Threshold = -0.2)	58.44	51.17	43.87	**-1641.55**

Table 1: Evaluation Results for Different Experiments (Best Results in Bold)

4.3 Abstention Module

Prompt for Abstention

```
### Task
Classify whether the question is answerable or unanswerable - [QUESTION]{user_question}[/QUESTION]
### Instructions
- Remember that answerable question is one that can be answered with the given database
- Remember that unanswerable question is one that cannot be answered with the given database
### Database Schema
The query will run on a database with the following schema: {table_metadata_string}
### Answer
Given the database schema, here is the class of [QUESTION]{user_question}[/QUESTION]
[CLASS]{answer}[/CLASS]
```

The abstention module has a pre-trained text-to-SQL model (SQLCoder-7b-2) that is fine-tuned to classify whether a question is answerable given the database schema (the prompt template is provided). We generate the data for fine-tuning the abstention model by taking all the unanswerable questions and randomly sampling the same number of answerable questions, thus preventing class imbalance. The model is fine-tuned for six epochs till convergence. If this model classifies the question as unanswerable, it returns "null".

4.3.1 Multi-task Training

We further experiment on the abstention module by training a multi-task model on both text-to-SQL and abstention tasks. Multi-task models are those which are trained to perform multiple related tasks. The training data of both of the tasks are combined and the model is fine-tuned for one epoch till convergence.

4.4 Reliability Module

The reliability module has two checks - confidence score thresholding and error filtering. The SQL query is returned if both conditions are met.

4.4.1 Confidence Score Thresholding

The confidence score is calculated by summing up the log probabilities of the generated tokens. It checks whether the confidence score of the generated SQL query is above a certain threshold, returning "null" if it does not satisfy this criterion. The confidence score distribution is plotted for the test dataset and the threshold is chosen heuristically.

4.4.2 Error Filtering

The generated SQL query is executed on the database and returns "null" if there is an error in execution.

5 Experiments

The system is developed incrementally, allowing us to evaluate each module after its introduction. We begin with a baseline system consisting solely of the text-to-generation module and an experiment is conducted with different representations of the table metadata. Error filtering is introduced through the execution of the query. Next, we integrate the abstention module and compare its performance to the multi-task model trained on both text-to-SQL generation and abstention tasks. Finally, we incorporate the reliability check of confidence score thresholding, experimenting with different threshold values to optimize performance.

5.1 Experimental Setup

We perform each fine-tuning using 4-bit Quantized Low-Rank Adaptation (QLoRA) (Dettmers et al., 2023). QLoRA is applied to all the linear layers of the model and the LoRA rank and alpha are chosen as 32 and 64 respectively. A paged 8-bit Adam optimizer with weight decay (Loshchilov and Hutter, 2019) is used with a learning rate of 2.5e-5 on a linear scheduler. We fine-tune the model with a batch size of 8 and 2 gradient accumulation steps. The inference is optimized using vLLM (library for LLM inference and serving) (Kwon et al., 2023). We use greedy decoding for inference and the tokens generated are limited to 4096.

5.2 Evaluation Metrics

The system is evaluated using reliability score (RS). RS metric rewards accurate SQL generation for answerable questions and abstaining from answering unanswerable questions. It penalizes incorrect generation or attempts to answer unanswerable questions. The aggregate RS is the mean of individual scores represented as a percentage.

The severity of the penalty can be adjusted by a parameter c. A higher value of c leads to stricter evaluation. RS(0) does not penalize any mistakes ($c = 0$). In RS(5), every accurate prediction earns a +1 reward, while each mistake results in a -5 penalty. This means every 5 correct predictions weigh the same as one incorrect prediction.

6 Results

The evaluation results of the different experiments can be seen in Table-1. We achieve the best performance with the system of abstention module + text-to-SQL generation module + reliability module, with a confidence score threshold of -0.4 (See Fig 2 for confidence score distribution). Our submission ranks 6th on the leaderboard.

6.1 Limitations

The system suffers from the effects of cascading errors. Each module has its own intricacies and potential points of failure. If any of the modules makes an incorrect prediction, the subsequent modules will likely propagate and amplify the error.

The reliability module's performance may heavily depend on the chosen confidence score threshold. Setting the threshold requires careful consideration. A high threshold might reject good queries, while a low threshold might allow unreliable ones.

Despite the reliability checks, there is still a possibility of false positives (accepting unreliable queries) or false negatives (rejecting reliable queries). Balancing between these two extremes is crucial for the system's overall reliability and performance.

7 Conclusion

In this work, we present a system consisting of several layers that contribute to reliable text-to-SQL modeling. By filtering out unanswerable questions based on the provided database schema, the system avoids generating incorrect SQL queries and focuses on its strengths. This makes the system robust to unanswerable questions. The reliability module ensures a certain level of confidence in the generated query before using it. Error filtering avoids errors during execution.

Fine-tuning the pre-trained models on EHR data specifically helps the system understand the medical language and schema, leading to more accurate SQL generation for EHR-related queries in comparison to models trained on generic text-to-SQL tasks. Using pre-trained text-to-SQL models as a starting point helps the system leverage existing knowledge and reduces the amount of training data required for fine-tuning. This has led to resource efficiency as no external training data is used.

The modular design allows for easier development, maintenance, and potential future improvements to each specific module. This facilitates adaptation to evolving requirements or changes in the dataset or task. Overall, this system presents a promising approach for reliable text-to-SQL generation in the EHR domain. However, the potential limitations need to be managed to ensure the system's robust and reliable performance in real-world applications.

References

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. 2023. Text-to-sql empowered by large language models: A benchmark evaluation. *CoRR*, abs/2308.15363.

Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. 2000. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220.

Alistair Johnson, Lucas Bulgarelli, Tom Pollard, Steven Horng, Leo Anthony Celi, and Roger Mark. 2021. Mimic-iv (version 1.0).

Alistair Johnson, Lucas Bulgarelli, Tom Pollard, Steven Horng, Leo Anthony Celi, and Roger Mark. 2023. Mimic-iv clinical database demo (version 2.2).

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Gyubok Lee, Hyeonji Hwang, Seongsu Bae, Yeonsu Kwon, Woncheol Shin, Seongjun Yang, Minjoon Seo, Jong-Yeup Kim, and Edward Choi. 2022. Ehrsql: A practical text-to-sql benchmark for electronic health records. *Advances in Neural Information Processing Systems*, 35:15589–15601.

Gyubok Lee, Sunjun Kweon, Seongsu Bae, and Edward Choi. 2024. Overview of the ehrsql 2024 shared task on reliable text-to-sql modeling on electronic health records. In *Proceedings of the 6th Clinical Natural Language Processing Workshop*, Mexico City, Mexico. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Youcheng Pan, Chenghao Wang, Baotian Hu, Yang Xiang, Xiaolong Wang, Qingcai Chen, Junjie Chen, Jingcheng Du, 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.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2024. Code llama: Open foundation models for code. *Preprint*, arXiv:2308.12950.

Richard Tarbell, Kim-Kwang Raymond Choo, Glenn Dietrich, and Anthony Rios. 2023. Towards understanding the generalization of medical text-to-SQL models and datasets. *AMIA Annual Symposium Proceedings*, 2023:669–678.

Ping Wang, Tian Shi, and Chandan K. Reddy. 2020. Text-to-sql generation for question answering on electronic medical records. In *Proceedings of The Web Conference 2020*, WWW '20, page 350–361, New York, NY, USA. Association for Computing Machinery.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103.