LG AI Research & KAIST at EHRSQL 2024: Self-Training Large Language Models with Pseudo-Labeled Unanswerable Questions for a Reliable Text-to-SQL System on EHRs

Yongrae Jo[1]* **Seongyun Lee**[2]* **Minju Seo**[2]* **Sung Ju Hwang**[2] **Moontae Lee**[1]

[1]LG AI Research, [2]KAIST

{yongrae.jo, moontae.lee}@lgresearch.ai sjhwang82@kaist.ac.kr

Abstract

Text-to-SQL models are pivotal for making Electronic Health Records (EHRs) accessible to healthcare professionals without SQL knowledge. With the advancements in large language models, these systems have become more adept at translating complex questions into SQL queries. Nonetheless, the critical need for reliability in healthcare necessitates these models to accurately identify unanswerable questions or uncertain predictions, preventing misinformation. To address this problem, we present a self-training strategy using pseudo-labeled unanswerable questions to enhance the reliability of text-to-SQL models for EHRs. This approach includes a two-stage training process followed by a filtering method based on the token entropy and query execution. Our methodology's effectiveness is validated by our top performance in the EHRSQL 2024 shared task, showcasing the potential to improve healthcare decision-making through more reliable text-to-SQL systems.

1 Introduction

Electronic Health Records (EHRs) are relational databases storing patients' medical histories within hospitals, covering details from admission to discharge. Common challenges with EHRs include difficulties in documenting and tracking health information, supporting team coordination, and sharing data (Cifuentes et al., 2015). Although ensuring the accurate capture of relevant information is crucial for addressing these challenges, accessing and querying these records often requires knowledge of SQL, making it challenging for healthcare provides in practical settings without technical expertise. A solution to this problem is developing a text-to-SQL model that can translate natural language questions into SQL queries to retrieve information from EHRs.

Recent advancements in Large Language Models (LLMs) have expanded their utility beyond natural language processing to include code generation, enabling them to interpret text for table manipulation and translate descriptions into code effectively (Lee et al., 2024b). These capabilities showcased by code-generating LLMs (Li et al., 2023; Roziere et al., 2023; Guo et al., 2024) demonstrate their potential in text-to-SQL applications. These developments suggest a promising horizon for leveraging LLMs to make EHR data more accessible to healthcare professionals, eliminating the prerequisite of SQL knowledge and significantly simplifying information retrieval (Hwang et al., 2019a; Lyu et al., 2020; Wang et al., 2020b; Park et al., 2021).

However, in the healthcare domain, the reliability of text-to-SQL models is crucial compared to other areas of NLP application. These models must not only generate precise SQL queries from natural language but also identify unanswerable questions—queries that cannot be solved with the available database—to avoid potentially harmful outcomes. The risk of providing answers to such questions underscores the need for these models to err on the side of caution, by preferring not to provide an answer rather than risking the provision of incorrect information (Lee et al., 2023). This approach underlines the unique challenges faced in healthcare NLP, emphasizing the need for accuracy and the ability to recognize when it cannot provide a reliable answer.

Our work introduces PLUQ, an approach leveraging the self-training paradigm to improve the reliability of text-to-SQL models for EHRs through training with **P**seudo-**L**abeled **U**nanswerable **Q**uestions. Self-training, a semi-supervised learning technique, involves re-training a model using its own predictions on unlabeled data to boost its performance. Our method adopts a two-stage version of self-training process, where we initially fine-tune a seed model using a given training

*These authors contributed equally to this work.

Proceedings of the 6th Clinical Natural Language Processing Workshop, pages 635–643

dataset. We then augment the training dataset by incorporating unanswerable questions that the fine-tuned model identifies from an unlabeled dataset. Subsequently, we fine-tune the model once more using this augmented training data to produce the final model.

Self-training is commonly used in scenarios where unlabeled data is abundant but obtaining labeled data is costly. Text-to-SQL for EHRs exemplifies such a scenario. In this context, a real-world service can collect users' natural language queries without much effort, but determining the correct SQL statement or verifying its answerability with the given database is time-consuming. We adopt the self-training approach to effectively address the issue of class imbalance between answerable and unanswerable questions, thus enhancing the robustness and reliability of the model's performance.

After the two-stage self-training process, we apply a filtering strategy to eliminate uncertain predictions. This strategy employs two types of filtering: one based on the maximum entropy of tokens and the other on the execution results of queries. Specifically, we assess the entropy in each token generated by the language model, designating the prediction's entropy as that of the token with the highest entropy. If a prediction's entropy exceeds a certain threshold, we consider it an unanswerable question, reflecting the model's lack of confidence in providing a correct answer. Additionally, we remove SQL queries that either produce errors or fail to retrieve valid values from the MIMIC-IV[1] dataset.

This approach was validated by our performance in the EHRSQL 2024 shared task (Lee et al., 2024a), where we achieved the top ranking, demonstrating the effectiveness of our method in improving the reliability of text-to-SQL systems in healthcare.

In summary, our study contributes a method that enhances the reliability of text-to-SQL systems for EHRs, addressing the shared task of handling unanswerable questions. This work supports better access to and utilization of EHRs, aiding in informed healthcare decision-making.

The main contributions of our paper are:

1. We propose a self-training method that uses pseudo-labeled unanswerable questions to train text-to-SQL models. This approach helps improve the model's ability to identify queries it cannot answer accurately, thereby increasing reliability.

2. We detail the comprehensive strategy employed, from the initial prompting of the model to the filtering steps, to ensure the research can be reproduced. This clarity in methodology allows for the approach to be validated and applied by others in the field, enhancing text-to-SQL systems in healthcare.

3. Our method won the EHRSQL 2024 shared task, demonstrating its practical effectiveness in a competitive setting. This success showcases its potential to contribute to the healthcare field by improving access to EHRS through reliable text-to-SQL systems.

2 Related Work

In the field of Natural Language Processing (NLP), recent research has focused on text-to-SQL and applying large language models (LLMs) to Electronic Health Records (EHRs). These studies have advanced the handling of complex queries and the processing of healthcare data, setting the stage for our research on test-time data sample labeling and augmentation in EHRs.

Text-to-SQL In the evolving field of natural language processing, the development of text-to-SQL technologies represents a significant advancement. Pioneering efforts in this area, Hwang et al. (2019b) and Lyu et al. (2020), harnessed the power of BERT for column classification to tackle the Wiki SQL(Zhong et al., 2017) dataset, which is characterized by its simplistic select/where queries. For more complex scenarios, the SPIDER dataset(Yu et al., 2018), comprising Multi-Table questions, necessitated a understanding of relationship between different tables. Wang et al. (2020a) employed graph-based methods to integrating information, while Lin et al. (2020) introduced schema linking as an input. Moreover, fine-tuning pre-trained language models such as T5 (Raffel et al., 2019) has yielded substantial performance improvements in this field.

Large Language Models in Text-to-SQL The emergence of LLMs has inspired novel approaches to text-to-SQL tasks. Dong et al. (2023) introduced efficient zero-shot framework, which capitalize on the robust understanding capabilities of

[1]https://physionet.org/content/mimic-iv-demo/2.2/

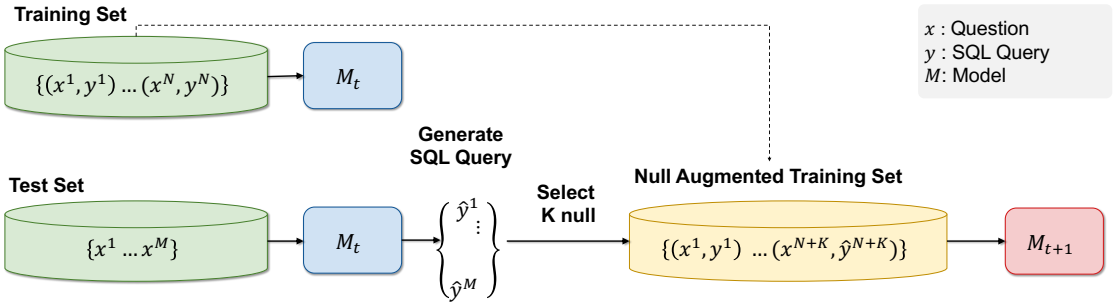

Figure 1: Training Process and SQL Query Generation. The model is initially trained using the training set. Then, a SQL query (or null) is generated for each sample in the test set using the trained model. Subsequently, we select K null samples and add them to the training set, resulting in a null-augmented training set. This augmented dataset is then used to train the final model, denoted as M_{t+1}.

LLMs, with a particular emphasis on prompt-based techniques that demonstrates remarkable efficiency. Tai et al. (2023); Nan et al. (2023); Gao et al. (2024) explore optimal demonstrations based on methodologies like text dense similarity or query similarity selection. Pourreza and Rafiei (2023) enhances the integrity of generated SQL through decomposition of queries and self-correction strategies. Shi et al. (2024) proposed LLMs as an agent for generating code and executing it, leverage the few-shot learning capabilities of LLMs for solving the multi-tabular health record datasets.

Enhancement of LLMs Through Data Augmentation and Self-Training Amini et al. (2022) presents an extensive review of self-training methods, including consistency-based approaches and transductive learning. Post the rise of LLMs, the field of self-training methods has garnered considerable attention. To enhance the capabilities of LLMs, some studies have focused on autonomous data generation. Wang et al. (2023) stands out by generating synthetic data from a pool consisting not only of seed data but also data generated through an iterative process. Seo et al. (2024) employs a few-shot learning approach, drawing samples from external sources to align the seed data in low-resource settings. The line of autonomously augmenting data broadens and enhances the model's capabilities. Yuan et al. (2024) introduces using their own outputs to continuously improve both their instruction-following and reward-modeling abilities, demonstrating significant performance enhancements over traditional training methods.

NLP in EHRs The application of NLP techniques in EHRs has been extensively explored, utilizing texts and structured knowledge. Pampari et al. (2018) proposed a question-answering system based on unstructured clinical notes. More recently, many works have been developed in the development of generation task based on structured EHRs. Wang et al. (2020b) construct the table-based QA datasets using MIMIC-III(Johnson et al., 2016) . Park et al. (2021) introduced a graph-based EHR QA system that leverages SPARQL queries from the MIMIC-SQL dataset(Wang et al., 2020b). Raghavan et al. (2021) focuses on QA tasks using the structured patient records in the MIMIC-III. Lee et al. (2023) datasets containing multi-table queries and those involving null values, reflecting real-world scenarios in healthcare domain.

In our research, we utilize a trained LLM on the training dataset for test-time data sample labeling and subsequent augmentation. This approach is particularly focused on addressing the imbalance in the 'null' class.

3 Method

We train a seed model using the original training dataset and then use this model for pseudo labeling on the test set. From this, we select only the samples labeled as unanswerable and augment them to the original training dataset to create the final dataset for self-training. Our self-trained model, PLUQ generates SQL queries. We apply post-processing and two stages of filtering to these queries to ensure their reliability and produce the final answers.

3.1 Seed model fine-tuning

In developing a model specialized for the text-to-SQL task, we initially fine-tuned seed model on the given training data. Because it is widely recognized that there exists a performance gap between open-source LLMs and proprietary LLMs in many benchmarks. In section 4.3, the results re-

garding performance corresponding to changes in the model substantiate it. Therefore, we utilized the Finetuning API provided by OpenAI to fine-tune the GPT-3.5-Turbo-0125 model.

The training dataset comprises a total of 5,124 samples, including both answerable and unanswerable questions. We employed all of these data samples in our training. Furthermore, to ensure that PLUQ accurately references the correct column names when generating SQL queries, we converted the table schema of the provided MIMIC-IV demo database into text format and incorporated it into the input for training. Additionally, to enhance PLUQ capability in distinguishing between answerable and unanswerable questions, we incorporated information about unanswerable questions into the input. This strategic inclusion aimed at refining PLUQ discernment, thus improving its overall accuracy in classifying questions.

3.2 Self-training

Unanswerable Question Pseudo Labeling Unanswerable questions refer to queries that either do not align with the given table schema or require external knowledge, rendering them unsolvable using only the MIMIC-IV demo database for SQL query generation. In our training dataset, the number of answerable questions is considerable, reaching 5,124, whereas unanswerable questions are limited to just 450. This disparity highlights a data imbalance issue within our training dataset, which may impede the model's ability to correctly respond to unanswerable questions during testing.

Moreover, there is a low similarity between the queries in the training data and those in the development/test sets. We found that the average cosine similarity between query embeddings in the train and development sets is only 0.36, and between the train and test sets is 0.34, measured using OpenAI's text-embedding-3-large embedding model. Such a disparity in dataset distribution could lead to significant performance declines for the model at test time. To address these issues, we initially perform pseudo labeling on the development/test set using PLUQ, which was originally trained solely on the original training dataset.

Training With Augmented Data Pseudo-labeling is one of the techniques used in semi-supervised learning, serving as a powerful tool for addressing issues of data scarcity and label imbalance. Particularly with the EHRSQL dataset, a notable disparity exists: the quantity of unanswerable questions is significantly lower compared to answerable ones within the training data. Training a model with such data increases the likelihood of the model's inability to accurately respond to unanswerable questions. In tasks where reliability is crucial, especially compared to other domains, this could result in substantial penalties. Therefore, we choose to augment the original training set with those samples predicted as unanswerable. Finally, we fine-tune PLUQ using the augmented dataset.

3.3 Filtering

Despite the two-stage training process, including self-training, PLUQ still generates incorrect SQL queries. To enhance the reliability of our final predictions, we implemented a filtering process to sift out samples that were either inaccurately generated or produced with uncertainty by the model. This filtering stage plays a crucial role in ensuring the outputs of PLUQ are more dependable and accurate.

Maximum Token Entropy Based Filtering Tokens in a language model-generated output have higher entropy when the information is uncertain. Therefore, treating samples with high entropy as unanswerable questions aids in creating a more reliable system while incurring fewer penalties. We evaluate the entropy of each token produced by the language model, and define the entropy of the prediction based on the token exhibiting the maximum entropy. Then, in the entire set of predictions, samples exceeding a certain entropy level are considered as unanswerable questions and are filtered out. We have set a threshold for this filtering process, determined by the proportion of unanswerable questions in the dataset we aim to predict. This proportion of unanswerable questions is used as a hyperparameter to calibrate the threshold for filtering.

Execution Based Filtering Finally, we implement an additional process of filtering to ensure that the remaining SQL queries, after the initial filtering, can successfully access the MIMIC-IV demo database and retrieve valid values. Utilizing the sqlite3 library in Python, we test each SQL query. Queries that trigger errors, return empty values, or yield None are deemed unable to retrieve valid values. Consequently, we filter these queries as unanswerable questions. This step further en-

sures the accuracy and reliability of the system by only allowing queries that can effectively interact with the database.

4 Experiments

The experiments are conducted on the development and test sets provided by the EHRSQL 2024 shared tasks. All results presented are derived from runs on the official platform. Section 4.1 details the models, datasets, and metrics used for training and inference, while section 4.2 discusses the experimental results. Finally, in Section 4.3, we conduct ablation studies on various components of PLUQ to examine their individual contributions and impacts.

4.1 Settings

Dataset & Model We utilize the EHRSQL 2024 dataset for both training and evaluation. The dataset comprises 5,124 training, 1,163 development, and 1,167 test data entries. Notably, only the training dataset is accompanied by gold SQL queries and their corresponding executed gold answers. For questions deemed answerable, it's essential to generate the correct SQL query. For those classified as unanswerable, a null output is required. Database for SQL query generation is the MIMIC-IV demo database. We employ the GPT-3.5-Turbo-0125 model for fine-tuning purposes. Evaluation of our method is conducted on the codabench platform, where we submitted SQL queries predicted by PLUQ for the test set and obtained scores based on their performance.

$$\phi_c(x) = \begin{cases} 1 & \text{if } x \in \mathcal{Q}_{\text{ans}}; g(x) = 1; \text{Acc}(x) = 1 \\ 0 & \text{if } x \in \mathcal{Q}_{\text{ans}}; g(x) = 0, \\ -c & \text{if } x \in \mathcal{Q}_{\text{ans}}; g(x) = 1; \text{Acc}(x) = 0 \\ -c & \text{if } x \in \mathcal{Q}_{\text{una}}; g(x) = 1, \\ 1 & \text{if } x \in \mathcal{Q}_{\text{una}}; g(x) = 0. \end{cases}$$

Figure 2: **Formal Definition of RS** for a single data instance. Q_{una} denotes unanswerable question, Q_{ans} represents answerable question. $g(x) = 1$ means that model generates SQL query and $g(x) = 0$ denotes that model generates 'null'. $Acc(x) = 1$ signifies instances where the model's prediction is correct, while $Acc(x) = 0$ indicates cases where the prediction is incorrect. c represents the penalty.

Metrics We utilize the Reliability Score (RS) as our primary metric (Lee et al., 2023). In figure 2, the RS aims to accomplish two main objectives: firstly, it provides rewards for correctly generating SQL for answerable questions Q_{ans} and for not generating SQL for unanswerable questions Q_{una}; secondly, it imposes penalties for wrongly generating SQL for Q_{ans} and for any attempts to create SQL for Q_{una}. However, the RS neither rewards nor penalizes for choosing not to answer Q_{ans}. The penalties are structured as 0, 5, 10, or N, where N corresponds to the total number of entries in the dataset. The final score is calculated by adding 1 point for each correct sample and deducting points based on the penalty for incorrect ones, followed by averaging these scores. Importantly, in the EHRSQL 2024 shared task, the primary metric for determining rankings is RS(10).

4.2 Results

Development Set In the development set, PLUQ exhibits the highest performance in RS(10), the primary metric, which positions it at the top of the official leaderboard when compared with other models. A notable aspect of PLUQ is the minimal difference between its RS(0) and RS(10) scores compared to other models. This indicates that PLUQ effectively reduces penalties by categorizing uncertain outcomes in answerable questions and unanswerable questions as 'unanswerable.' This strategy underscores our model's superior reliability, as it avoids the risk of incorrect answers where uncertainty exists, a feature that sets it apart from its counterparts.

Test Set In the final ranking phase of the shared task, which utilized the test set, PLUQ experienced a slight overall decrease in scores compared to its performance in the development set. Despite this dip, it maintained a higher score across all RS, including the pivotal RS(10), when compared with other models. This consistent performance across all metrics, even amidst a minor decline, ultimately led PLUQ to win the EHRSQL 2024 shared task.

4.3 Ablation Studies

Model Ablation We observe the performances across difference models. A total of three models were used, namely Flan-T5-base, Tulu-7b, GPT-3.5-Turbo-0125, and GPT-4-Turbo-Preview. Flan-T5-base, Tulu-7b, and GPT-3.5-Turbo-0125 is fine-tuned, while GPT-4-Turbo-Preview is applied with

Table 1: **Results of the Development and Test Phases** on the Official Codabench Leaderboard. The best results are highlighted in bold. Pivotal metric is RS(10) in this shared task. Note that Ours score for the development phase differs from the official leaderboard because we didn't add it to the leaderboard.

	Development				Test			
Team	RS(0)	RS(5)	RS(10)	RS(N)	RS(0)	RS(5)	RS(10)	RS(N)
PLUQ (Ours)	90.37	**89.51**	**88.65**	**-109.6**	**88.17**	**84.75**	**81.32**	**-711.83**
PromptMind	66.38	59.5	52.62	-1533.62	82.6	78.75	74.89	-817.4
ProbGate	84.18	79.45	74.72	-1015.82	81.92	78.06	74.21	-818.08
KU-DMIS	**91.57**	82.98	74.38	-1908.43	72.07	65.64	59.21	-1427.93
oleg1996	47.03	34.14	21.24	-2952.97	68.89	56.47	44.04	-2831.11
LTRC-IIITH	N/A	N/A	N/A	N/A	66.84	55.27	43.7	-2633.16
Saama Technologies	57.78	50.47	43.16	-1642.22	53.21	44.64	36.08	-1946.79
TEAM_optimist	N/A	N/A	N/A	N/A	14.14	-349.61	-713.37	-84885.86

System prompt

You are helpful text-to-sql assistant.

User prompt

You are given SQL table schema and the question. Generate the SQL query for the following question. Note that you should generate 'null' if the question cannot be converted to SQL query given information.

[SQL Table Schema]
table admissions, columns = [row_id, subject_id, hadm_id, admittime, dischtime, admission_type, admission_location, discharge_location, insurance, language, marital_status, age]
table chartevents, columns = [row_id, subject_id, hadm_id, stay_id, itemid, charttime, valuenum, valueuom]
table cost, columns = [row_id, subject_id, hadm_id, event_type, event_id, chargetime, cost]
table d_icd_diagnoses, columns = [row_id, icd_code, long_title]
table d_icd_procedures, columns = [row_id, icd_code, long_title]
table d_items, columns = [row_id, itemid, label, abbreviation, linksto]
table d_labitems, columns = [row_id, itemid, label]
table diagnoses_icd, columns = [row_id, subject_id, hadm_id, icd_code, charttime]
table icustays, columns = [row_id, subject_id, hadm_id, stay_id, first_careunit, last_careunit, intime, outtime]
table inputevents, columns = [row_id, subject_id, hadm_id, stay_id, starttime, itemid, amount]
table labevents, columns = [row_id, subject_id, hadm_id, itemid, charttime, valuenum, valueuom]
table microbiologyevents, columns = [row_id, subject_id, hadm_id, charttime, spec_type_desc, test_name, org_name]
table outputevents, columns = [row_id, subject_id, hadm_id, stay_id, charttime, itemid, value]
table patients, columns = [row_id, subject_id, gender, dob, dod]
table prescriptions, columns = [row_id, subject_id, hadm_id, starttime, stoptime, drug, dose_val_rx, dose_unit_rx, route]
table procedures_icd, columns = [row_id, subject_id, hadm_id, icd_code, charttime]
table transfers, columns = [row_id, subject_id, hadm_id, transfer_id, eventtype, careunit, intime, outtime]
foreign_keys = [admissions.subject_id = patients.subject_id, diagnoses_icd.hadm_id = admissions.hadm_id, diagnoses_icd.icd_code = d_icd_diagnoses.icd_code, procedures_icd.hadm_id = admissions.hadm_id, procedures_icd.icd_code = d_icd_procedures.icd_code, labevents.hadm_id = admissions.hadm_id, labevents.itemid = d_labitems.itemid, prescriptions.hadm_id = admissions.hadm_id, cost.hadm_id = admissions.hadm_id, cost.event_id = diagnoses_icd.row_id, cost.event_id = procedures_icd.row_id, cost.event_id = labevents.row_id, cost.event_id = prescriptions.row_id, chartevents.hadm_id = admissions.hadm_id, chartevents.stay_id = icustays.stay_id, chartevents.itemid = d_items.itemid, inputevents.hadm_id = admissions.hadm_id, inputevents.stay_id = icustays.stay_id, inputevents.itemid = d_items.itemid, outputevents.hadm_id = admissions.hadm_id, outputevents.stay_id = icustays.stay_id, outputevents.itemid = d_items.itemid, microbiologyevents.hadm_id = admissions.hadm_id, icustays.hadm_id = admissions.hadm_id, transfers.hadm_id = admissions.hadm_id]

Question: {question}
SQL Query:

Figure 3: The system prompt and the user prompt template used in PLUQ. The prompt integrates instructions for handling unanswerable questions and the MIMIC-IV database schema.

in-context learning. All results are conducted on the development set.

Among the fine-tuned models, GPT-3.5-Turbo-0125 demonstrates the highest performance. This indicates that there is still a performance gap between proprietary and open-source models. Furthermore, despite having more parameters, Tulu-7b shows lower performance compared to Flan-T5-base. Additionally, it is observed that GPT-4-Turbo-Preview, known for its high performance in nu-

Table 2: **Model Ablation Study** of the Development Set across the finetuned open-source LLMs and in-context learning, finetuned proprietary LLMs. FT denotes the fine-tuning of the model, while ICL represents in-context learning (Wei et al., 2022). In this work, the number of few-shot examples used for in-context learning is fixed at 4.

Models	RS(0)	RS(5)	RS(10)	RS(N)
Flan-T5-base FT	82.11	76.53	70.94	-1217.8
Tulu-7b FT	10.23	-38.77	-87.9	-11389.7
GPT-4-Turbo-Preview ICL	63.52	-118.85	-301.22	-186836.4
GPT-3.5-Turbo-0125 FT (Ours)	**90.37**	**89.51**	**88.65**	**-109.6**

Table 3: **Prompt Ablation Study** of the Development Set including the integration of table schema and the incorporation of unanswerable information to evaluate the impact of various prompts on model performance. The base model of fine-tuning is GPT-3.5-Turbo-0125 model.

Models	RS(0)	RS(5)	RS(10)	RS(N)
Fine-Tuning	83.23	78.5	73.77	-1016.7
+ Table Schema	89.85	83.83	77.82	-1310.1
+ Unans Info (Ours)	**90.37**	**89.51**	**88.65**	**-109.6**

Table 4: **Filtering Ablation Study** of Development Set. For maximum token entropy based filtering, we filtered out SQL queries possessing high entropy within the top 7%, classifying them as unanswerable questions.

Models	RS(0)	RS(5)	RS(10)	RS(N)
No Filtering	80.82	5.58	-69.94	-17419.1
+ Exec Filtering	**93.98**	**89.68**	85.38	-906.01
+ Ent Filtering (Ours)	90.37	89.51	**88.65**	**-109.6**

merous benchmarks, scored lower than fine-tuned models when only in-context learning is applied.

Prompt Ablation In the study, we compare the performance of models based on the information included in the input prompts during training. When table schema information is incorporated into the prompts, the models perform better than without it. This suggests that providing table schema information, such as column names, offers a valuable learning signal to the models.

Additionally, explicitly including information about unanswerable questions results in higher scores than when such information is omitted. By providing criteria for answerable and unanswerable questions, the models are aided in avoiding questions they could not answer and focusing on providing accurate responses to those that are answerable.

In the final version of the prompt, we incorporated the database schema of MIMIC-IV as well as the instruction related to unanswerable questions. You can find the prompt in Figure 3.

Filtering Ablation In table 4, by applying execution filtering, which treats invalid SQL queries that either do not execute or retrieve empty values as unanswerable questions, a significant performance improvement is observed, particularly in scenarios with substantial penalties such as RS(10) and RS(N). Additionally, by implementing entropy-based filtering, which filters out SQL queries with higher entropy than a set threshold among those with high maximum token entropy, performance is further enhanced by effectively eliminating SQL queries that, even when executed, return incorrect values.

5 Conclusion

In our work, we develop a self-training strategy designed to enhance the reliability of text-to-SQL models for Electronic Health Records (EHRs) through the inclusion of pseudo-labeled unanswerable questions. This approach is particularly valuable in scenarios where there is an abundance of unlabeled data and labeling is costly, thus providing substantial clinical utility in real-world applications. Our approach employs a two-stage training process alongside a filtering mechanism based on token entropy and query execution outcomes to improve the model's precision and its ability to identify unanswerable questions. The performance is validated by our leading performance in the EHRSQL 2024 shared task. Our method contributes towards rendering EHRs more accessible to healthcare professionals without SQL knowledge, addressing a critical need for reliable information retrieval in healthcare. Future research could explore how large language models facilitate the integration of unstructured medical texts into specific schemas, enhancing interoperability in varied healthcare settings.

Limitations

Our method achieve the best score in this challenge, as we adopt various techniques to enhance reliability. However, there are some limitations to our approach. Since our model is fine-tuned using EHRs, its ability to generalize across the entire EHR dataset is limited. Additionally, the fine-tuning process requires training data, which poses a challenge due to the high costs and time associated with data collection. Furthermore, despite

achieving the highest score among all teams, our RS(N) score still remains negative, indicating that caution should be exercised when considering the application of our method in real-world scenarios.

Acknowledgements

We would like to express our gratitude to Professor Minjoon Seo for his invaluable contributions to this project. His guidance and insightful discussions significantly enhanced our research.

References

Massih-Reza Amini, Vasilii Feofanov, Loïc Pauletto, Emilie Devijver, and Yury Maximov. 2022. Self-training: A survey. *CoRR*, abs/2202.12040.

Maribel Cifuentes, Melinda Davis, Doug Fernald, Rose Gunn, Perry Dickinson, and Deborah J Cohen. 2015. Electronic health record challenges, workarounds, and solutions observed in practices integrating behavioral health and primary care. *The Journal of the American Board of Family Medicine*, 28(Supplement 1):S63–S72.

Xuemei Dong, Chao Zhang, Yuhang Ge, Yuren Mao, Yunjun Gao, Lu Chen, Jinshu Lin, and Dongfang Lou. 2023. C3: zero-shot text-to-sql with chatgpt. *CoRR*, abs/2307.07306.

Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. 2024. Text-to-sql empowered by large language models: A benchmark evaluation. *Proc. VLDB Endow.*, 17(5):1132–1145.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y Wu, YK Li, et al. 2024. Deepseek-coder: When the large language model meets programming–the rise of code intelligence. *arXiv preprint arXiv:2401.14196*.

Wonseok Hwang, Jinyeong Yim, Seunghyun Park, and Minjoon Seo. 2019a. A comprehensive exploration on wikisql with table-aware word contextualization. *arXiv preprint arXiv:1902.01069*.

Wonseok Hwang, Jinyeung Yim, Seunghyun Park, and Minjoon Seo. 2019b. A comprehensive exploration on wikisql with table-aware word contextualization. *CoRR*, abs/1902.01069.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

Gyubok Lee, Hyeonji Hwang, Seongsu Bae, Yeonsu Kwon, Woncheol Shin, Seongjun Yang, Minjoon Seo, Jong-Yeup Kim, and Edward Choi. 2023. EHRSQL: A practical text-to-sql benchmark for electronic health records. *CoRR*, abs/2301.07695.

Gyubok Lee, Sunjun Kweon, Seongsu Bae, and Edward Choi. 2024a. 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.

Younghun Lee, Sungchul Kim, Tong Yu, Ryan A Rossi, and Xiang Chen. 2024b. Learning to reduce: Optimal representations of structured data in prompting large language models. *arXiv preprint arXiv:2402.14195*.

Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, et al. 2023. Starcoder: may the source be with you! *arXiv preprint arXiv:2305.06161*.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2020. Bridging textual and tabular data for cross-domain text-to-sql semantic parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 4870–4888. Association for Computational Linguistics.

Qin Lyu, Kaushik Chakrabarti, Shobhit Hathi, Souvik Kundu, Jianwen Zhang, and Zheng Chen. 2020. Hybrid ranking network for text-to-sql. *CoRR*, abs/2008.04759.

Linyong Nan, Yilun Zhao, Weijin Zou, Narutatsu Ri, Jaesung Tae, Ellen Zhang, Arman Cohan, and Dragomir Radev. 2023. Enhancing few-shot text-to-sql capabilities of large language models: A study on prompt design strategies. *CoRR*, abs/2305.12586.

Anusri Pampari, Preethi Raghavan, Jennifer J. Liang, and Jian Peng. 2018. emrqa: A large corpus for question answering on electronic medical records. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2357–2368. Association for Computational Linguistics.

Junwoo Park, Youngwoo Cho, Haneol Lee, Jaegul Choo, and Edward Choi. 2021. Knowledge graph-based question answering with electronic health records. In *Proceedings of the Machine Learning for Healthcare Conference, MLHC 2021, 6-7 August 2021, Virtual Event*, volume 149 of *Proceedings of Machine Learning Research*, pages 36–53. PMLR.

Mohammadreza Pourreza and Davood Rafiei. 2023. DIN-SQL: decomposed in-context learning of text-to-sql with self-correction. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou,

Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Preethi Raghavan, Jennifer J. Liang, Diwakar Mahajan, Rachita Chandra, and Peter Szolovits. 2021. emrkbqa: A clinical knowledge-base question answering dataset. In *Proceedings of the 20th Workshop on Biomedical Language Processing, BioNLP@NAACL-HLT 2021, Online, June 11, 2021*, pages 64–73. Association for Computational Linguistics.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Minju Seo, Jinheon Baek, James Thorne, and Sung Ju Hwang. 2024. Retrieval-augmented data augmentation for low-resource domain tasks. *CoRR*, abs/2402.13482.

Wenqi Shi, Ran Xu, Yuchen Zhuang, Yue Yu, Jieyu Zhang, Hang Wu, Yuanda Zhu, Joyce C. Ho, Carl Yang, and May D. Wang. 2024. Ehragent: Code empowers large language models for complex tabular reasoning on electronic health records. *CoRR*, abs/2401.07128.

Chang-Yu Tai, Ziru Chen, Tianshu Zhang, Xiang Deng, and Huan Sun. 2023. Exploring chain of thought style prompting for text-to-sql. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 5376–5393. Association for Computational Linguistics.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020a. RAT-SQL: relation-aware schema encoding and linking for text-to-sql parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7567–7578. Association for Computational Linguistics.

Ping Wang, Tian Shi, and Chandan K. Reddy. 2020b. Text-to-sql generation for question answering on electronic medical records. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 350–361. ACM / IW3C2.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 13484–13508. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir R. Radev. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3911–3921. Association for Computational Linguistics.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. *CoRR*, abs/2401.10020.

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