ACCSCIENCE PUBLISHING

Artificial Intelligence in Health

ORIGINAL RESEARCH ARTICLE

# Efficient schema-less text-to-SQL conversion using large language models

**Youssef Mellah*, Veysel Kocaman, Hasham Ul Haq, and David Talby**

John Snow Labs, Coastal Highway, Lewes, Delaware, United States of America

## Abstract

Large language models (LLMs) are increasingly being applied to several tasks including text-to-SQL (the process of converting natural language to SQL queries). While most studies revolve around training LLMs on large SQL corpora for better generalization and then perform prompt engineering during inference, we investigate the notion of training LLMs for schema-less prompting. In particular, our approach uses simple natural language questions as input without any additional knowledge about the database schema. By doing so, we demonstrate that smaller models paired with simpler prompts result in considerable performance improvement while generating SQL queries. Our model, based on the Flan-T5 architecture, achieves logical form accuracy (LFA) of 0.85 on the MIMICSQL dataset, significantly outperforming current state-of-the-art models such as Defog-SQL-Coder, GPT-3.5-Turbo, LLaMA-2-7B and GPT-4. This approach reduces the model size, lessening the amount of data and infrastructure cost required for training and serving, and improves the performance to enable the generation of much complex SQL queries.

***Keywords:*** Large language models; MIMICSQL; Schema-less; Logical form accuracy; Defog-SQL-Coder; GPT-3.5-Turbo; LLaMA-2-7B; GPT-4

***Corresponding author:**
Youssef Mellah
(youssef@johnsnowlabs.com)

**Citation:** Mellah Y, Kocaman V, Haq HU, Talby D. Efficient schema-less text-to-SQL conversion using large language models. *Artif Intell Health.* 2024;1(2): 96-106. doi: 10.36922/aih.2661

**Received:** January 6, 2024

**Accepted:** February 23, 2024

**Published Online:** April 4, 2024



## 1. Introduction

Text-to-SQL technology has gained considerable attention in recent years, emerging as a transformative tool for database interaction. Its key advantage lies in enabling users, particularly those with limited SQL knowledge, to use a fine-tuned large language model (LLM) to interact with databases using natural language. This innovation significantly reduces the necessity to learn SQL for data retrieval and analytics from tabular datasets. The effectiveness of such systems hinges on two main aspects: the intuitiveness of usage and the accuracy of the generated queries. Essentially, this means that user prompts should be straightforward and the corresponding SQL queries must accurately address the user's query with high precision.

The growing abundance of structured and semi-structured data in various domains, ranging from e-commerce to healthcare, highlights the importance of the text-to-SQL task. This task gains relevance as the demand for more intuitive interfaces to query and extract information from these databases increases. Traditional SQL queries, which require understanding of both database schema and query syntax, are often challenging for users lacking technical expertise. Text-to-SQL aims to mitigate this challenge by

enabling users to formulate queries in natural language, thereby lowering the barriers to data access and analysis.

In the past decade, the field of natural language processing (NLP), especially through the development of LLMs, has seen remarkable progress, substantially enhancing text-to-SQL systems' performance.[1,2] Models such as T5, LLaMA, GPT-3, GPT-3.5, and GPT-4 have been pivotal in advancing natural language understanding and generation, displaying a profound ability to process and produce human-like text. Despite these advancements, adapting these versatile models for specific applications, such as generating SQL queries for structured data, remains a significant challenge.

In this research, we aim to tackle the dual challenges of simplifying input prompts and elevating accuracy in the generation of SQL queries, with a specific focus on the intricate landscape of the medical domain. Given the critical importance of precision in data retrieval within healthcare contexts, our primary goal is to fine-tune Flan-T5-based models using text-to-SQL query pairs meticulously tailored for the medical MIMICSQL dataset.[3] The decision to utilize a medical dataset in our research is driven by the distinctive challenges and precision requirements inherent in health-care data retrieval. The choice of the MIMICSQL dataset, derived from the widely-used MIMIC-III database, provides a realistic and clinically relevant context, allowing us to address the complexities of real-world medical scenarios. Focusing on the medical domain enables us to tailor our approach to the unique intricacies of healthcare data, contributing directly to advancements in medical data management. By enhancing the accuracy of SQL query generation in this specific context, our research seeks to deliver a meaningful impact on the efficiency of data retrieval in medical databases, benefiting health-care professionals, researchers and decision-makers.

To guide our investigation effectively, we pose the following research questions:

(i) How can we optimize the formulation of input prompts to simplify the querying process while maintaining the necessary specificity required for medical data retrieval?
(ii) What adjustments and enhancements can be made to Flan-T5-based models to improve their accuracy in generating SQL queries tailored to the nuances of the medical MIMICSQL dataset?
(iii) How do schema-less questions contribute to streamlining input prompt complexity, and what impact does this simplification have on the overall performance of SQL query generation in the medical domain?

As we delve into these research questions, our methodology strategically leverages schema-less questions, with a deliberate focus on mitigating the challenges posed by complex and lengthy input prompts. While acknowledging that this approach may potentially limit generalization across diverse database schemas, we anticipate that the pronounced enhancement in overall performance will substantiate this deliberate trade-off.

The organization of this paper is as follows: Section 2 presents a thorough review of the existing literature in the text-to-SQL field. Section 3 describes our methodology, including details about the MIMICSQL dataset, preprocessing steps, and the fine-tuning process. Section 4 discusses the experimental setup, covering evaluation metrics, comparison methods, experimental results, and their analysis. The final section concludes the paper, summarizing our contributions and highlighting the significance of applying LLMs to the text to-SQL task, with a special emphasis on schema-less querying.

## 2. Related works

The task of text-to-SQL is to convert natural utterances into SQL queries. This field has attracted researchers in the NLP and the database community for decades.[4-9] The methodologies currently in use to handle this task can be broadly divided into three categories: rule-based methods, fine-tuning methods, and in-context learning (ICL) methods. Rule-based approaches, as highlighted in other studies,[7,10] utilize predefined templates to generate SQL queries. These methods show proficiency in certain scenarios but are limited by the necessity for manual rule formulation, which restricts their versatility across diverse domains.

Addressing the limitations of rule-based methods, recent research has ventured into more flexible approaches. The utilization of bi-directional long-short-term memory and convolutional networks[11] in Seq2Seq models has enhanced adaptability and effectiveness, though integrating structural database information remains a persistent challenge. Graph neural networks have emerged as a solution to this, with approaches that treat the database schema as a graph, as seen in other works.[12,13] Furthermore, the introduction of the MIMICSQL dataset and a model-based system by Translate-Edit Model for Question-to-SQL (TREQS) marked a significant advancement in text-to-SQL, particularly in the medical domain. Their model sets a robust baseline for subsequent evaluations. In our study, we use the MIMICSQL dataset and the TREQS model as benchmarks to evaluate and compare the effectiveness of our proposed method. In addition, fine-tuning pretrained language models like T5 have demonstrated improved

performance in the text-to-SQL domain.[14-16] However, these fine-tuning methods typically require an extensive amount of labeled training data tailored to the specific task, and they are often susceptible to over-fitting. This limitation raises concerns about their versatility and efficiency in practical applications.

The advent of LLMs like GPT has opened new avenues in text-to-SQL tasks, particularly due to their ICL capabilities. These models often outperform fine-tuning methods in various NLP downstream tasks, especially in scenarios requiring few-shot or zero-shot learning. Nevertheless, the effectiveness of LLMs heavily relies on the design of input prompts, a factor that significantly influences the output quality.[17-19] The ICL performance of LLMs in text-to-SQL tasks, especially the impact of different prompts, has also been examined.

While basic prompting serves as a benchmark for assessing the fundamental capabilities of LLMs, more sophisticated prompt designs have shown to significantly enhance performance. Notably, a few-shot learning approach employing GPT-4 recently set a new benchmark in text-to-SQL tasks, achieving state-of-the-art results. However, this method necessitates manual input for demonstrations and tends to use a large number of tokens, requiring more time and resources.

This study extends the current advancements in LLMs within the Text-to SQL domain. Specifically, we fine-tune Flan-T5-based models on the MIMICSQL dataset. Each of these models is a sequence-to-sequence LLM that can be also used commercially. The model was published by Google researchers in late 2022 and has been fine-tuned on multiple tasks. It reframes various tasks into a text-to-text format, such as translation, linguistic acceptability, sentence similarity, and document summarization. Similarly, the architecture of the Flan-T5 model closely aligns with the encoder-decoder structure utilized in the original Transformer paper. The primary distinction lies in the size and nature of the training data; Flan-T5 was trained on an extensive 750 GB corpus of text known as the Colossal Clean Crawled Corpus (C4), and it comes with five variations: flan-t5-small (80M parameters, requiring 300 MB in memory), flan-t5-base (250M parameters, requiring 990 MB in memory), flan-t5-large (780M parameters, requiring 1 GB in memory), flan-t5-xl (3B parameters, requiring 12 GB in memory), and flan-t5-xxl (11B parameters, requiring 80 GB in memory). These models can be used for various NLP tasks out-of-the-box (with zero or few shot); however, to leverage its full potential and ensure optimal performance for specific applications, fine-tuning is a crucial step. Below are the main points and reasons highlighting the choice of fine-tuning FLAN-T5 for the specific text-to-SQL task:

(i) Fine-tuning FLAN-T5 is important to adapt the model to specific tasks and improve its performance on those tasks.
(ii) Fine-tuning allows for customization of the model to better suit the user's needs and data.
(iii) The ability to fine-tune FLAN-T5 on local workstations with CPUs makes it accessible to a wider range of users.
(iv) This accessibility is beneficial for smaller organizations or individual researchers who may not have access to GPU resources.
(v) Overall, fine-tuning FLAN-T5 is a valuable step in optimizing the model for specific use cases and maximizing its potential benefits.

Our emphasis on exploring schema-less approaches led us to investigate the viability and advantages of implementing text-to-SQL systems that depend less on explicit knowledge of database schema.

## 3. Data and methods

This section delineates the comprehensive methodology of our study, encompassing a detailed description of the dataset utilized, the architecture of the model employed, and the specifics of both the training and evaluation processes.

### 3.1. Dataset

The MIMICSQL dataset is a significant resource for question-to-SQL generation in the healthcare domain, comprising 10,000 question-SQL pairs. This large-scale dataset is based on the Medical Information Mart for Intensive Care III (MIMIC III) dataset, a widely used electronic medical records (EMR) database. It is divided into two subsets: one containing template questions (machine-generated) and the other featuring natural language questions (human-annotated).

#### *3.1.1. Diversity and complexity of the dataset*

The MIMICSQL dataset covers a wide range of patient information categories, including demographics, laboratory tests, diagnosis, procedures, and prescriptions. Those categories are embedded as a schema structure that outlines the database's tables, columns, and interrelationships, serving as a crucial guide for the models to comprehend the database structure and accurately formulate SQL queries. Table 1 illustrates the DEMOGRAPHIC table, while Table 2 presents the PROCEDURES table from the MIMICSQL dataset. This diversity reflects the complexity and multidimensionality of healthcare-related queries,

as the SQL queries generated from these questions often involve multiple tables and columns.

#### 3.1.2. Size and partitioning

The MIMICSQL dataset comprises approximately 10,000 examples, strategically partitioned into training and development (train and dev) sets, constituting 80% (8000 question-sql pairs), and a test set accounting for the remaining 20% (2000 question-sql pairs). This division facilitates both training and evaluation phases. Insights and statistical distributions from the MIMICSQL dataset are illustrated in Figure 1, and an illustrative example from the dataset is shown in Figure 2. Specifically, Figure 1A depicts the distribution of natural language questions, while Figure 1B focuses on the distribution of SQL query lengths. The presentation of natural language (NL) question and SQL query length distributions in the MIMICSQL dataset serves to reveal the dataset's inherent characteristics, aiding in the design of models capable of handling diverse language structures. In addition, it provides a basis for evaluating model performance by highlighting potential challenges associated with varying lengths. Understanding these distributions is crucial for both effective selections of the model training hyperparameters, especially the input and the output length, as well as assessment of the generalizability of the developed models to real-world applications.

**Table 1. Example of the DEMOGRAPHIC table from theMIMICSQL database**

| SUBJECT_ID | HADM_ID | Gender | ADMISSION_TYPE | … |
|---|---|---|---|---|
| 990 | 184231 | F | EMERGENCY | … |
| 17772 | 122127 | M | NEWBORN | … |
| … | … | … | … | … |
| 66411 | 178264 | F | EMERGENCY | … |

**Table 2. EExample of the PROCEDURES table from the MIMICSQL database**

| SUBJECT_ID | HADM_ID | SHORT_TITLE | … |
|---|---|---|---|
| 9258 | 183354 | Procedure-one vessel | … |
| 28588 | 141664 | Insert endot. tube | … |
| … | … | … | … |
| 66411 | 178264 | Abdomen artery inc. | … |

#### 3.1.3. Challenges addressed

One of the key challenges addressed by the MIMICSQL dataset is the prevalence of abbreviations and potential typos in healthcare-related questions. This poses a significant obstacle to accurately generating the corresponding SQL queries, as the keywords provided in the questions may not precisely match those used in the EMR data. Consequently, the dataset presents a real-world scenario that requires models to effectively handle the nuances and complexities of healthcare-related queries.

### 3.2. Problem formulation

The SQL query generation task can be formulated as follows: Let D = {(Qi, SQL(Qi))} for i = 1, 2,…, N represents the dataset, where Qi represents the i-th natural language question, and SQL(Qi) is the corresponding ground-truth SQL query. The objective is to learn a mapping function F(Q; θ) parameterized by θ using the LLMs Flan-T5 Base and Flan-T5 Large:

$$SQL(Q)=F(Q;\theta) \quad (I)$$

The training process involves minimizing the cross-entropy loss between the predicted SQL queries and the ground-truth queries:

$$Loss(\theta)= -\sum \log P\,(SQL(Q_i)|Q(_i;\theta) \quad (II)$$

where $P(SQL(Q_i)|Q_i;\theta)$ denotes the probability of the model predicting the correct SQL query for the *i*-th natural language question ($Q_i$), given the model parameters θ. Our schema-less approach entails using only the input question Q as context during inference, without explicit database schema information.

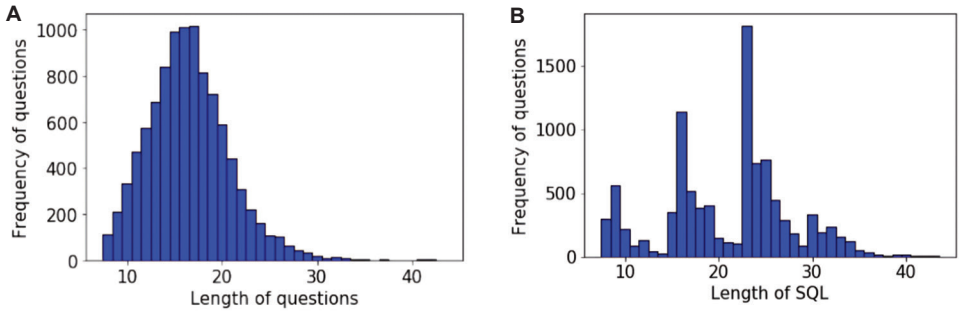


**Figure 1.** Distribution of the input and the output sequences in the MIMICSQL dataset. (A) The distribution of length on the questions (the input). (B) The distribution of length of SQL queries.

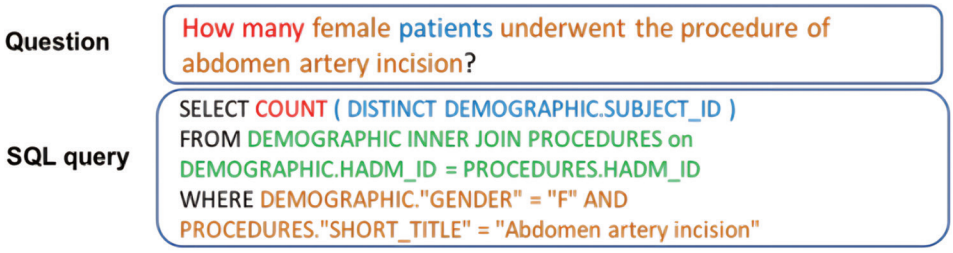


**Figure 2.** Illustrative example demonstrating how the MIMICSQL dataset utilizes the DEMOGRAPHICS and PROCEDURES tables to construct a response to a given question. This example employs color coding to distinctly indicate the correlations between components of the source question, the corresponding SQL query, and the SQL query template. Such examples highlight the dataset's structure and the complexity of mapping natural language questions to SQL queries.

### 3.3. Preprocessing

To effectively utilize the MIMICSQL dataset within the constraints of LLMs like Flan-T5, specific preprocessing steps are essential. These steps are designed to address the complexities of SQL syntax and the particular decoding capabilities of such models. A key consideration in this process is the model's vocabulary, as an excessive number of special tokens can detrimentally affect performance. The following outlines the detailed preprocessing steps undertaken:

#### *3.3.1. Enclosing JSON objects in an array*

Individual JSON objects in the MIMICSQL dataset were enclosed within an array to ensure a consistent JSON array structure. This step was essential for data manipulation and loading during subsequent processing.

#### *3.3.2. Replacing SQL characters*

Replacing symbols with their corresponding words in text preprocessing is a common practice in NLP and offers several advantages. One of the reasons for this practice is to address the absence of certain special characters, such as "<," "<=," and "<>," in the vocabulary of models like Flan-T5. On the other hand, this replacement enhances model understanding as words are typically more interpretable and easier for the model to learn. It can lead to improved generalization, as models have an easier time working with words, which are part of natural language, compared to arbitrary symbols. In addition, using words reduces ambiguity, as symbols can be context-dependent and unclear.

#### *3.3.3. Converting JSON to CSV*

After the preprocessing steps, the JSON data were transformed into CSV format to facilitate compatibility with data analysis and modeling libraries. The CSV format with "text" and "sql" columns allowed seamless data integration into the training and evaluation processes.

#### *3.3.4. Data cleaning*

The majority of the data cleaning process was conducted beforehand on the MIMIC III database, while building the MIMICSQL dataset, involving correcting errors in patient demographics, standardizing the format of clinical notes, and filtering out irrelevant data. On the MIMICSQL dataset, we only removed some duplicated statements and corrected some typos in the SQL queries.

#### *3.3.5. Tokenization and text prefixing*

In the preprocessing phase, we employed the Flan-T5 tokenizer to tokenize the input and the output texts. Flan-T5 utilizes a sub-word tokenizer to break down the input texts into smaller units, capturing both word-level and sub-word information. It is based on SentencePiece, a popular unsupervised text tokenizer and detokenizer, that employs a segmentation algorithm to divide the input texts into sub-word units, allowing the model to handle a wide range of vocabulary and linguistic nuances. By leveraging Flan-T5 Large's tokenization approach, we aim to capture the contextual information present in both complete words and sub-word units, enhancing the model's ability to comprehend and generate meaningful sequences during the subsequent stages of our text-to-SQL task. Moreover, we added a prefix "transform:" to each natural language question. The prefix is specific to the T5-based model, allowing it to recognize the text as a task to be transformed into SQL queries. For the target, a padding token ID is set to −100, an adjustment designed to disregard padding tokens during loss calculation.

Table 3 shows a running example for the preprocessing of two entries from the MIMICSQL dataset.

## 4. Experiments and results

In this section, we will describe the experimental setup, infrastructure details and evaluation metrics for the experiment, as well as a comparison with other models.

**Table 3. A running example for the preprocessing steps**

| Original text | Preprocessed text | Tokenized text |
|---|---|---|
| Input: Get the number of patients who died in or before 2132 and were less than 72 years of age. Output: SELECT COUNT (DISTINCT DEMOGRAPHIC.'SUBJECT_ID') FROM DEMOGRAPHIC WHERE DEMOGRAPHIC.'AGE' < '72' AND DEMOGRAPHIC.'DOD_YEAR' <= '2132.0' | Input: transform: get the number of patients who died in or before 2132 and were less than 72 years of age. Output: select count (distinct demographic.'subject_id') from demographic where demographic.'age' less than '72' and demographic.'dod_year' less than or equal to '2132.0' | Input: ['_transform', ':', '_Get', '_the', '_number', '_of', '_patients', '_who', '_died', '_in', '_or', '_before', '_21', '32', '_and', '_were', '_less', '_than', '_72', '_years', '_of', '_age', '.'] Output: ['_select', '_count', '_(', '_distinct', '_demographic', '.', '"', 'sub', 'ject', '_', 'i', 'd', '"', '_', ')', '_from', '_demographic', '_where', '_demographic', '.', '"', 'age', '"', '_less', '_than', '"', '72', '"', '_and', '_demographic', '.', '"', 'd', 'o', 'd', '_', 'year', '"', '_less', '_than', '_or', '_equal', '_to', '_', '"', '2', '13', '2.0', '"'] |
| Input: calculate the minimum days for which patients aged 20 years or older were hospitalized. Output: SELECT MIN (DEMOGRAPHIC.'DAYS_STAY') FROM DEMOGRAPHIC WHERE DEMOGRAPHIC.'AGE' >= '20' | Input: transform: calculate the minimum days for which patients aged 20 years or older were hospitalized. Output: select min (demographic.'days_stay') from demographic where demographic.'age' greater than or equal to '20' | Input: ['_transform', ':', '_calculate', '_the', '_minimum', '_days', '_for', '_which', '_patients', '_aged', '_20', '_years', '_or', '_older', '_were', '_hospital', 'ized', '.'] Output: ['_select', '_min', '_(', '_demographic', '.', '"', 'day', 's', '_', 'stay', '"', '_', ')', '_from', '_demographic', '_where', '_demographic', '.', '"', 'age', '"', '_greater', '_than', '_or', '_equal', '_to', '_', '"', '20', '"'] |

### 4.1. Experimental setup

A key objective of this study is to evaluate the practicality of training and inference processes. For this purpose, we employed a standalone machine equipped with a single Nvidia V100 GPU (16 GB vRAM) and 32 GB of system memory.

In our implementation, we utilized both Flan-T5 Base and Large versions of models, based on the original T5 encoder-decoder architecture, augmented with an instruction-finetune mechanism. This architecture consists of multiple layers of transformer blocks, including self-attention mechanisms and feed-forward neural networks. These transformer blocks enable the model to capture long-range dependencies and contextual information from input sequences.

The input and output sequence lengths were standardized to 1024 tokens. Sequences exceeding this length were truncated, while shorter sequences were padded using a pad token. This sequence length configuration enabled a maximum training batch size of two on our GPU setup. Various learning rates and optimizers were tested, ultimately leading to the selection of the Adam optimizer with a learning rate of 5e-5.

The fine-tuning phase for each model spanned five epochs, a decision based on empirical observations of convergence and generalization in preliminary trials. This epoch count offered an optimal balance between model performance and training duration, ensuring the models adequately learned from the MIMICSQL dataset patterns without overfitting. In terms of time, completing five epochs of training took approximately 4 h for the Flan-T5 Base version and 7 h for the Flan-T5 Large version. Table 4 provides in details the tuning scenario.

While evaluating other approaches, significant attention was given to the construction of prompts for LLaMA-2-7B, GPT-3.5-Turbo, GPT-4, and DeFog-SQLCoder to effectively generate SQL queries. For these models, prompts were meticulously designed to include schema information, facilitating the generation of accurate SQL queries. These prompts are essential for guiding the models through the task, leveraging their inherent language understanding capabilities. Detailed examples of these prompts are provided in Figure 3. Notably, our approach with the Flan-T5 models is distinct from this conventional method, removing the need for any schema information in the prompts. In this approach, the Flan-T5 models are fine-tuned in a way that the questions are the only inputs, and the database schema can be captured automatically in a more efficient fashion. This distinction underscores the uniqueness and efficiency of our methodology. It is important to mention that TREQS models, not being categorized as LLMs, did not necessitate such prompt-based approaches, but used the schema as input with the question, further differentiating our method from traditional practices.

### 4.2. Evaluation metric

The most common and used evaluation metric for the text-to-SQL task are logical form accuracy (LFA, or exact matching) and execution accuracy,[20] and since the MIMIQSQL community has not shared databases for computing the execution accuracy metric, the primary metric we used for evaluating the models performance in this study is LFA. In fact, it measures the percentage of exact string match between the generated SQL queries and the ground truth SQL queries. It is quantified as the percentage of instances where the model's predicted SQL

**Table 4. Details of the tuning scenario**

| Parameter | Value | Description |
|---|---|---|
| output_dir | /logs_long | Directory where the trained model and logs will be saved |
| per_device_train_batch_size | 1 | Number of training samples per device (GPU) in each batch |
| per_device_eval_batch_size | 1 | Number of evaluation samples per device (GPU) in each batch |
| predict_with_generate | True | Whether to use generation during evaluation |
| fp16 | False | Whether to use mixed precision training with FP16 |
| learning_rate | 5e-5 | Learning rate for training |
| num_train_epochs | 5 | Number of training epochs |
| logging_dir | ./logs_long | Directory for logging training metrics and logs |
| logging_strategy | steps | Strategy for logging training metrics (steps or epoch) |
| logging_steps | 500 | Interval for logging training metrics |
| evaluation_strategy | epoch | Strategy for evaluation during training (steps or epoch) |
| save_strategy | epoch | Strategy for saving checkpoints during training (steps or epoch) |
| save_total_limit | 2 | Maximum number of checkpoints to keep |

```
Prompt for GPT3.5-turbo and GPT4:
"""You are a helpful assistant that generates SQL queries. \n
Given the schema {schema}, generate an SQL query to answer the following question:\n\n{question}\n\n
 SQL:"""

Prompt for Defog-SQLCoder:
"""### Task
Generate a SQL query to answer the following question and database schema:
`{question}`
`{schema}`"""

Prompt for LLAMA2-7B (finetuned on text2sql):
"""Below is an context that describes a sql query, paired with an question that provides further
information. Write an answer that appropriately completes the request.
### Context:
{schema}
### Question:
{question}
### Answer:"""

Prompt for our FlanT5 models, which includes just the question as input, prefixed with "transform":
"Transform: {question}"
```

**Figure 3.** Comparison of prompt construction for SQL query generation. This figure illustrates the detailed prompts used for LLaMA-2-7B, GPT-3.5-Turbo, GPT-4, and DeFog-SQLCoder, highlighting the inclusion of schema information in each, except our Flan-T5 models in compliant with the schema-less approach. Since TREQS is not regarded as an LLM, no prompt generated for that approach.

query exactly aligns with the ground-truth as described in formula (III).

$$\mathrm{LFA} = \frac{\mathrm{NCLF}}{\mathrm{TNI}} \qquad \text{(III)}$$

Where NCLF is the number of correct logical forms, and TNI is the total number of instances in the test set. A higher LFA score signifies a model's enhanced capability in accurately translating natural language questions into their corresponding SQL queries, reflecting a deeper understanding of the logic and relationships inherent in these representations. This makes LFA a particularly pertinent measure for the text-to-SQL task, as it directly gauges the fine-tuned models' effectiveness in interpreting natural language and generating precise SQL queries.

### 4.3. Results and discussion

The evaluation of various text-to-SQL models on the MIMICSQL test set has provided significant insights. The baseline TREQS model recorded an LFA of 0.48, which marginally increased to 0.55 with the incorporation of a recovery technique (TREQS + Recover). The current state-of-the-art model, Defog-SQLCoder, achieved an LFA of 0.65. In comparison, the LLMs GPT 3.5-Turbo and GPT-4 demonstrated robust performance with LFA scores of 0.60 and 0.70, respectively, highlighting their applicability. In addition, the LLaMA-2-7B model, which was fine-tuned for text-to-SQL tasks, attained an LFA of 0.60. Remarkably, our custom fine-tuned model, Flan-T5 Large, surpassed all these models with an LFA of 0.85.

Figure 4 presents a clear illustration of a sample natural language query, the ground truth SQL query that would accurately respond to this query, and the SQL queries generated by the LLMs used in our experiments, namely, LLaMA 2-7B, GPT-3.5-Turbo, GPT-4, and DeFog-SQLCoder, along with our Flan-T5 models. This comparison vividly highlights the differences in the query generation capabilities of each model, offering a tangible demonstration of their respective performances in the text-to-SQL context.

This outcome indicates that while existing models such as GPT-3.5-Turbo (20B parameters), LLaMA-2-7B (7B parameters), and Defog-SQLCoder (15B parameters) show commendable proficiency, our approach using the schema-less text-to-SQL with Flan-T5 Large, which has only 780M parameters, notably outperforms others. This demonstrates not only superior performance but also remarkable efficiency, offering transformative potential in both specific domains and broader applications. The detailed results are tabulated in Table 5.

The results from our comprehensive evaluation shed light on the text-to-SQL domain, underscoring the significance of language model-based models (LLMs) and the promising potential of schema-less approaches in healthcare. It is crucial to note that the LLMs under scrutiny, specifically LLAMA-2-7B and DeFog-SQLCoder, were fine-tuned on the text-to-SQL task, encompassing datasets such as MIMICSQL, thereby directly incorporating knowledge pertinent to this domain. On the other hand, the GPT models (GPT-3.5-Turbo and GPT-4) are renowned for their versatility in evaluating various NLP tasks, including text-to-SQL, due to their extensive pre-training on diverse corpora. While these models were not specifically fine-tuned on the MIMICSQL dataset, their broad exposure during pre-training to a wide array of textual and structured data may have contributed to their performance on the MIMICSQL test set. This factor is important to consider when interpreting the comparative performance of these


**Figure 4.** Sample SQL query generation. This figure illustrates a sample natural language query alongside the corresponding ground truth SQL query and the SQL queries generated by the evaluated LLMs (LLaMA-2-7B, GPT-3.5-Turbo, GPT-4, and DeFog-SQLCoder) and our Flan-T5 models. In addition, an augmented version of the ground truth query is presented, serving as an example of how we enriched the training data during the fine-tuning of our FlanT5 models. It is important to note that this augmentation was exclusively for training purposes; no data in the test set were altered or augmented in any manner.

models against our fine-tuned Flan-T5 models, which were explicitly trained on the MIMICSQL dataset.

Comparison of our fine-tuned Flan-T5 models with baseline TREQS, as well as other LLMs such as Defog-SQLCoder, LLaMA-2-7B, GPT-3.5, and GPT-4, highlights the significant impact of compact LLMs and illustrates the effectiveness of schema-less querying. A key finding is the ability of smaller models, such as Flan-T5, to generate SQL queries directly from natural language questions without relying on explicit database schema information. This approach challenges the traditional schema-dependent paradigm. Furthermore, as illustrated in the Table 6, our study emphasizes the importance of model competence. The Flan-T5 models demonstrated proficiency in interpreting complex medical terminology, even when certain terms were absent from the model's vocabulary.

**Table 5. Comparison with current state-of-the-art models in the test set**

| Model | Parameter size | LFA |
|---|---|---|
| TREQS | 2.8 M | 0.48 |
| TREQS+Recover | 2.8 M | 0.55 |
| GPT-3.5-Turbo | 20 B | 0.60 |
| LLAMA-2-7B | 7 B | 0.60 |
| Defog-SQLCoder | 15 B | 0.65 |
| GPT-4 | - | 0.70 |
| Flan-T5 Base (Ours) | 220 M | 0.56 |
| Flan-T5 Large (Ours) | **770 M** | **0.85** |

Notes: Flan-T5 Large outperforms all other models. The parameter size of GPT-4 has not been disclosed by OpenAI. The boldface indicates the best model with the highest score achieved compared to other models, indicating its superior performance in the LFA metric.
Abbreviations: M: Million; B: Billion.

**Table 6. Examples demonstrating proficiency in interpreting complex medical terminology using our fine-tuned Flan-T5 Large model**

| Input question | SQL query generated by Flan-T5 Large model |
|---|---|
| How many patients have diagnoses icd9 code 28803? | SELECT COUNT (DISTINCT DEMOGRAPHIC."SUBJECT_ID") FROM DEMOGRAPHIC INNER JOIN DIAGNOSES ON DEMOGRAPHIC. HADM_ID=DIAGNOSES.HADM_ID WHERE DIAGNOSES."ICD9_CODE" = '28803'; |
| Which patients have a neolevo5 drug code? | SELECT COUNT (DISTINCT DEMOGRAPHIC."SUBJECT_ID") FROM DEMOGRAPHIC INNER JOIN PRESCRIPTIONS ON DEMOGRAPHIC.HADM_ID=PRESCRIPTIONS. HADM_ID WHERE PRESCRIPTIONS."FORMULARY_ DRUG_CD" = 'NEOLEVO5'; |

In the first example, the natural language query is accurately transformed into SQL, selecting the count of distinct patient IDs from the DEMOGRAPHIC table where the corresponding HADM_ID in the DIAGNOSES table matches the specified ICD9 code "28803." This showcases Flan-T5's ability to understand and incorporate complex medical terminology, such as ICD9 codes, into SQL queries effectively. While in the second example, the model accurately generates an SQL query that selects the count of distinct patient IDs from the DEMOGRAPHIC table based on the presence of the specific drug code "NEOLEVO5" in the PRESCRIPTIONS table. Despite the unfamiliar term "neolevo5," Flan-T5 effectively incorporates it into the SQL query, showcasing its ability to comprehend and handle diverse medical and pharmaceutical terminology. This capability is vital in healthcare, where accuracy and domain-specific language are paramount. Our findings encourage further exploration into strategies that reduce model and data complexity, focusing on task-oriented models suitable for commodity hardware.

## 5. Conclusion

This study investigates the efficacy of smaller, task-specific language models compared to more complex LLMs in the Text-to-SQL task, with a focus on the healthcare domain using the MIMICSQL dataset. Our findings reveal the remarkable performance of the fine-tuned Flan-T5 models, particularly Flan-T5 Large, which achieved an LFA score of 0.85. This score surpasses the current state-of-the-art model, Defog-SQLCoder, as well as other advanced LLMs such as LLaMA-2-7B, GPT-3.5-Turbo, and GPT-4. Our approach, advocating for the removal of schema definitions from input prompts and training separate models for distinct schemas, has proven effective, requiring less hardware resources and data for training. These findings underscore the potential of tailored compact language models for domain-specific applications, opening avenues for more efficient and effective natural language understanding in specialized contexts.

## Acknowledgments

None.

## Funding

None.

## Conflict of interest

The authors declare no competing interests.

## Author contributions

*Conceptualization:* Youssef Mellah, Veysel Kocaman
*Investigation:* All authors

*Methodology:* Youssef Mellah, Veysel Kocaman
*Formal analysis:* Youssef Mellah, Veysel Kocaman, Hasham UI Haq
*Writing – original draft:* Youssef Mellah, Veysel Kocaman, Hasham UI Haq
*Writing – review & editing:* Veysel Kocaman, David Talby

## Ethics approval and consent to participate

Not applicable.

## Consent for publication

Not applicable.

## Availability of data

Data used in this study can be found at: https://github.com/wangpinggl/TREQS/tree/master/mimicsql_data/mimicsql_natural_v2

## References


1. Deng N, Chen Y, Zhang Y. Recent Advances in Text-to-SQL: A Survey of What We Have and What We Expect. In: *Proceedings of the 29th International Conference on Computational Linguistics, Gyeongju, Republic of Korea. International Committee on Computational Linguistics*; 2022. p. 2166-2187.
2. Katsogiannis-Meimarakis G, Koutrika G. A survey on deep learning approaches for text-to-SQL. *VLDB J*. 2023;32:905-936.

   doi: 10.1007/s00778-022-00776-8
3. Wang P, Shi T, Reddy CK. Text-to-SQL Generation for Question Answering on Electronic Medical Records. In: *Proceedings of the Web Conference 2020* (*WWW '20*). New York, NY, USA: Association for Computing Machinery. p. 350-361.

   doi: 10.1145/3366423.3380120
4. Codd EF. A relational model of data for large shared data banks. *Commun ACM*. 1970;13(6):377-387.

   doi: 10.1145/362384.362685
5. Hemphill CT, Godfrey JJ, Doddington GR. The ATIS Spoken Language Systems Pilot Corpus. In: *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania*; 1990.

   doi: 10.3115/116580.116613
6. Dahl DA, Bates M, Brown M, et al. Expanding the Scope of the ATIS Task: The ATIS-3 Corpus. In: *Human Language Technology: Proceedings of a Workshop Held at Plainsboro*, New Jersey; 1994.

   doi: 10.3115/1075812.1075823
7. Zelle JM, Mooney RJ. Learning to Parse Database Queries Using Inductive Logic Programming. *In: Proceedings of the National Conference on Artificial Intelligence*; 1996. p. 1050-1055.
8. Popescu AM, Etzioni O, Kautz H. Towards a Theory of Natural Language Interfaces to Databases. In: *Proceedings of the 8th International Conference on Intelligent user Interfaces* (*IUI '03*). New York, NY, USA: Association for Computing Machinery; 2003. p. 149-157.

   doi: 10.1145/604045.604070
9. Bertomeu N, Uszkoreit H, Frank A, Krieger HU, Jörg B. Contextual Phenomena and Thematic Relations in Database QA Dialogues: Results from a Wizard-of-Oz experiment. In: *Proceedings of the Interactive Question Answering Workshop at HLT-NAACL*. New York, NY, USA: Association for Computational Linguistics; 2006. p. 1-8.
10. Saha D, Floratou A, Sankaranarayanan K, Minhas UF, Mittal AR, Özcan F. ATHENA: An ontology-driven system for natural language querying over relational data stores. *Proc VLDB Endow*. 2016;9(12):1209-1220.

    doi: 10.14778/2994509.2994536
11. Choi DH, Shin MC, Kim EG, Shin DR. RYANSQL: Recursively applying sketch-based slot fillings for complex text-to-SQL in cross-domain databases. *Comput Linguistics*. 2021;47(2):309-332.

    doi: 10.1162/coli_a_00403
12. Wang B, Shin R, Liu X, Polozov O, Richardson M. 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*. United States: Association for Computational Linguistics; 2020. p. 7567-7578.

    doi: 10.18653/v1/2020.acl-main.677
13. Cao R, Chen L, Chen Z, Zhao Y, Zhu S, Yu K. LGESQL: Line Graph Enhanced Text-to-SQL Model with Mixed Local and Non-Local Relations. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*. Vol. 1 (Long Papers); 2021. p. 2541-2555.

    doi: 10.18653/v1/2021.acl-long.198
14. Raffel C, Shazeer N, Roberts A, *et al.* Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv.* Preprint posted online 2019.

    doi: 10.48550/arXiv.1910.10683
15. Scholak T, Schucher N, Bahdanau D. PICARD: Parsing Incrementally for Constrained Auto-Regressive Decoding from Language Models. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Punta Cana, Dominican Republic. Association for Computational Linguistics; 2021. p. 9895-9901.

    doi: 10.18653/v1/2021.emnlp-main.779

16. Li H, Zhang J, Li C, Chen H. RESDSQL: Decoupling Schema Linking and Skeleton Parsing for Text-to-SQL. In: *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence (AAAI' 23/IAAI' 23/EAAI '23*), Vol. 37. Washington, DC, U.S: AAAI Press, 2023. p. 13067-13075.

doi: 10.1609/aaai.v37i11.26535

17. Min S, Lyu X, Holtzman A, *et al*. Rethinking the role of demonstrations: What makes in-context learning work? In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. p. 11048-11064.

doi: 10.18653/v1/2022.emnlp-main.759

18. Liu J, Shen D, Zhang Y, Dolan B, Carin L, Chen W. What Makes Good In-Context Examples for GPT-3? In: *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*. Dublin, Ireland: Association for Computational Linguistics; 2022. p. 100-114.

doi: 10.18653/v1/2022.deelio-1.10

19. Wei J, Wang X, Schuurmans D, *et al.* Chain-of-thought prompting elicits reasoning in large language models. *arXiv.* Preprint posted online 2022.

doi: 10.48550/arXiv.2201.11903

20. Yu T, Zhang R, Yang K, *et al*. 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: Association for Computational Linguistics; 2018. p. 3911-3921.

doi: 10.18653/v1/D18-1425