PeerJ Computer Science

Enhancing text-to-structured query language translation for seamless electronic medical record access

Gomathi Bakthavatchalu[1], Saravana Balaji Balasubramanian[2], Omar Khattab[3], Mhd Omar Al-Kadri[4], Dhanushkumar Vadivel[1], Harish Prasath V. G.[1], Srinivasa Pradeep S[1] and Somaiyeh MahmoudZadeh[4]

[1] Department of Computer Science and Engineering, PSG Institute of Technology and Applied Research, Coimbatore, Tamil Nadu, India
[2] Department of Computing, De Montfort University Kazakhstan, Almaty, Kazakhstan
[3] Department of Computer Science and Engineering, Kuwait College of Science and Technology, Doha, Kuwait
[4] College of Computing and Information Technology, University of Doha for Science and Technology, Doha, Qatar

ABSTRACT

Traditional models for natural language-to-SQL translation in Electronic Medical Record (EMR) systems struggle with understanding medical terminology, handling complex queries, and bridging the syntax-semantics gap, leading to scalability and accuracy issues. Advanced solutions like Large Language Model (LLM) based approaches address these challenges by leveraging deep learning and domain-specific training to enhance performance and usability. Hence, this article introduces an advanced medical Text-to-Structured Query Language (SQL) paradigm that simplifies accessing EMRs by translating natural language queries into SQL commands. This model is built on the advanced Code-T5 (Text-to-Text Transfer Transformer) architecture, further enhanced with Low-Rank Adaptation (LoRA) and Quantized Low-Rank Adaptation (QLoRA) techniques; it effectively addresses the challenges posed by the complexity of traditional SQL queries enabling seamless access to critical healthcare data. The innovation of the proposed model lies in its exceptional performance across multiple evaluation metrics. It achieves a Bilingual Evaluation Understudy (BLEU) score of 81.68, significantly outperforming leading models like T5, Fine-Tuned Language Net (FLAN) T5, and Bidirectional and Auto-Regressive Transformers (BART) while excelling in Recall-Oriented Understudy for Gisting Evaluation (ROUGE) metrics, underscoring its proficiency in generating semantically accurate and coherent SQL queries. Furthermore, the proposed model attains a high token-level F1-score, ensuring a balanced precision and recall and a Jaccard similarity score of 0.83, surpassing T5, Flan T5, and BART. The proposed model excels in handling complex medical queries, bridging natural language and SQL to empower data-driven decisions and advance medical informatics.

Submitted 18 March 2025
Accepted 19 November 2025
Published 26 February 2026

Corresponding author
Somaiyeh MahmoudZadeh, somaiyeh.mahmoudzadeh@udst.edu.qa

Academic editor
Nicole Nogoy

Additional Information and Declarations can be found on page 25

DOI **10.7717/peerj-cs.3467**

Subjects Artificial Intelligence, Data Mining and Machine Learning, Databases, Natural Language and Speech, Programming Languages

Keywords Natural language interface to databases, Large language model, Query processing, Structured query language, Natural language processing

How to cite this article Bakthavatchalu G, Balasubramanian SB, Khattab O, Al-Kadri MO, Vadivel D, V. G. HP, S SP, MahmoudZadeh S. 2026. Enhancing text-to-structured query language translation for seamless electronic medical record access. **PeerJ Comput. Sci. 12:e3467 DOI 10.7717/peerj-cs.3467**

INTRODUCTION

Effective data utilization in modern healthcare is critical for improving patient care, clinical decision-making, and research. Healthcare systems generate massive amounts of data, frequently in organized formats like Comma-Separated Values (CSV) files, containing sensitive information such as patient records, medical histories, and clinical results. Despite the abundance of data accessible, deriving good insights remains a considerable barrier, especially for healthcare practitioners with limited technical experience in data analysis. This can lead to inefficiencies, delays in decision-making, and potential errors, all of which negatively influence patient outcomes (*Badawy, Ramadan & Hefny, 2024*; *Zhang & Liu, 2019*). Traditionally, retrieving data from structured formats like CSV requires expertise in Database Management Systems (DBMS) and query languages like structured query language (SQL). However, recent advances in Natural Language Processing (NLP) have permitted the conversion of plain text into executable SQL queries. Early solutions relied on essential techniques like tokenization and lexical analysis, but accuracy and efficiency have been significantly improved with more advanced methods. Large Language Models (LLMs) such as Bidirectional Encoder Representations from Transformers (BERT), Generative Pre-trained Transformer (GPT), and T5 have shown great potential in automating this process. These models can understand complex user inquiries and generate structured queries, making them ideal for healthcare applications (*Vaswani et al., 2017*; *Devlin et al., 2019*).

Despite its potential, the intricacy of these models poses substantial challenges in terms of memory consumption, training time, and resource utilization, especially in resource-constrained contexts. New strategies like Low-Rank Adaptation (LoRA) and Quantified LoRA (QLoRA) have emerged to solve these issues. LoRA is an efficient approach for fine-tuning big models by reducing the number of trainable parameters. At the same time, QLoRA improves on this by adding quantization to model weights, considerably reducing memory cost while maintaining performance (*Houlsby et al., 2019*). These limitations create a critical barrier for non-technical healthcare professionals who require quick, intuitive access to Electronic Medical Record (EMR) data without needing to master complex query languages. To directly address these issues, this article introduces the MedQuery, an advanced medical Text-to-SQL system built on the Code-aware Text-to-Text Transfer Transformer (CodeT5) architecture, enhanced with efficient techniques like LoRA and QLoRA. By simplifying the translation of natural language queries into SQL commands, MedQuery reduces computational burden and ensures seamless access to critical healthcare data, empowering clinicians and administrators to make data-driven decisions effortlessly. The technology allows users to query data in natural language and converts these searches into structured SQL commands for processing. This research aims to enhance the accessibility and efficiency of healthcare data by comparing the CodeT5 model to other models, including T5, Fine-Tuned Language Net (FLAN)-T5, and BART. Specifically, the system employs LoRA and QLoRA methodologies to ensure its functionality can be implemented efficiently in real-world scenarios.

To clearly define the focus of this research, the following research questions and objectives are formulated. **Research Questions (RQs):** RQ1: How effective are LoRA and

QLoRA in fine-tuning the CodeT5 model for EMR text-to-SQL translation? RQ2: What trade-offs exist between model efficiency (in terms of memory usage and processing speed) and query accuracy when using LoRA and QLoRA compared to conventional fine-tuning methods? RQ3: How does the proposed MedQuery system enhance accessibility and usability for non-technical healthcare professionals in real-world EMR scenarios? **Research Objectives:** (1) To build an NLP-based system that transforms natural language queries into SQL statements specific to medical datasets. (2) To assess the effectiveness of LoRA and QLoRA strategies in improving the efficiency of large language models for healthcare applications. (3) To determine the usability and accessibility of the proposed system for non-technical healthcare workers. (4) To evaluate the system's performance in terms of accuracy, execution speed, and resource consumption compared with baseline models such as T5, FLAN-T5, and BART.

To assess the performance of the proposed system, several evaluation metrics are employed, including BLEU Score for n-gram precision, Exact Match Accuracy for evaluating exact matches between system output and expected results, and ROUGE Scores for measuring the overlap between the generated text and reference summaries (*Papineni et al., 2002*; *Lin, 2004*). Additionally, Levenshtein Distance, Jaccard Similarity, and Token-Level F1-score are used to evaluate edit distance, token-level similarity, and precision-recall balance (*Levenshtein, 1966*; *Jaccard, 1912*). By abstracting the complexities of SQL syntax and database structure, the system empowers healthcare professionals to interact with data intuitively and efficiently, ultimately enhancing clinical decision-making and patient care.

RELATED WORK

Recent research in Natural Language to SQL (NL2SQL) systems has focused on leveraging advanced techniques to enhance accuracy, scalability, and adaptability in diverse querying scenarios. *Fan et al. (2024)* propose the Natural Language to Structured Query Language (ZNL2SQL) framework, which integrates Small Language Models (SLMs) and LLMs for zero-shot NL2SQL translation. It performs better by dividing NL2SQL tasks into sub-tasks like SQL sketch generation and SQL query completion. This approach enhances schema alignment and reasoning capabilities, outperforming state-of-the-art methods by 10–20% in execution accuracy on benchmarks such as Dr. Spider and Kaggle Database Question Answering (KaggleDBQA). Key innovations include database serialization and multi-level matching for schema and value alignment. However, limitations like input length constraints and complex SQL generation challenges persist.

Sivasubramaniam et al. (2024) designed a multi-model Synthetic Multi-Model Medical Text-to-Query framework (SM3-Test-to-Query framework) for medical contexts. Spanning relational, graph, and document database models, it supports SQL, SPARQL Protocol and RDF Query Language (SPARQL), Cypher, and MongoDB Query Language (MQL) queries, creating a comprehensive dataset with 40K text-query pairs based on the Systematized Nomenclature of Medicine–Clinical Terms (SNOMED-CT) taxonomy. The benchmark evaluates Text-to-Query systems across SQL, MQL, Cypher, and SPARQL using a 10,000 natural language query pair dataset. The study provides key insights for advancing efficient and accurate Text-to-Query systems in healthcare by analyzing

performance trade-offs between database models and query languages. The Purpose, Understanding, Requirements, Plan, Learn, Execute (PURPLE) framework (*Ren et al., 2024*) improves Natural Language to SQL (NL2SQL) translation by optimizing LLM prompts. PURPLE achieves an improvement of 11.8% accuracy, addresses challenges with complex logical operators, and sets new benchmarks with 80.5% exact-set match accuracy and 87.8% execution match accuracy on the Spider benchmark. It is robust, cost-effective, and includes an ablation study to analyze its components. Nevertheless, issues like buggy SQL generation, challenges in skeleton prediction, and diminishing returns for lengthy inputs remain areas for further improvement.

The Query-based Clinical Named Entity Recognition (QCNER) approach (*Lalitha et al., 2023*) simplifies SQL generation for non-technical business users by employing tokenization, lemmatization, and Synthetic Minority Oversampling Technique (SMOTE)-based techniques, achieving 92.31% accuracy. This work describes a natural language interface for databases that uses Semantic Grammar to transform user queries into Procedural Language/Structured Query Language (PL/SQL), allowing non-technical users to access information more effectively. The technology also generates SQL workouts by combining questions and answers, which addresses conventional inefficiencies. A workflow that includes preprocessing, Named Entity Recognition, and SQL production effectively demonstrates its capabilities. The SM3 framework (*Sivasubramaniam et al., 2024*) also emphasizes the complexity of querying heterogeneous medical data by evaluating text-to-query systems across four database models. It reveals trade-offs between query complexity and performance in systems leveraging SQL, MQL, SPARQL, and Cypher for Electronic Health Records (EHRs). Med-PaLM (*Singhal et al., 2023*) fine-tunes a large-scale LLM using medical QA datasets and expert human feedback, achieving state-of-the-art performance on the Medical Question Answering (MedQA) and MultiMedQA benchmarks. It applies instruction-tuning and alignment techniques to enhance clinical reasoning and factual accuracy. Additionally, it incorporates self-consistency prompting to reduce hallucination and ensure safe clinical outputs. BioGPT (*Luo et al., 2022*), pre-trained on PubMed abstracts and full texts, incorporates biomedical entity embeddings and uses masked language modeling with a domain-specific vocabulary to support downstream tasks such as medical text generation, QA, and relation extraction. It also demonstrates strong performance in biomedical relation extraction and question-answering benchmarks, such as Biomedical Language Understanding and Reasoning Benchmark (BLURB). Clinical-T5 (*Lu, Dou & Nguyen, 2022*), a domain-adapted variant of the T5 model, is fine-tuned on Medical Information Mart for Intensive Care III (MIMIC-III) notes for clinical Named Entity Recognition (NER), document classification, and summarization tasks. It leverages encoder-decoder fine-tuning and domain-specific pre-tokenization to handle the complexity and noise in clinical narratives. It introduces task-specific prompt templates to boost adaptability across multiple clinical NLP benchmarks. These contributions collectively push the boundaries of NL2SQL research, addressing challenges in generalizability, domain-specific applications, and database adaptability. However, accuracy, query optimization, and real-world deployment limitations highlight the need for ongoing innovation.

However, the proposed MedQuery system is an innovative medical text-to-SQL generation model that stands out for its unique design and efficiency. It utilises the CodeT5 architecture, specifically designed for code generation tasks, to generate accurate and context-aware SQL queries for complex medical questions. Furthermore, it utilises Low-Rank Adaptation (LoRA) and QLoRA approaches to enhance computational efficiency, making it well-suited for resource-constrained healthcare scenarios. These elements assure precision, relevance, and practicality while querying medical databases.

BASELINE MODELS

To facilitate comparison, the performance of our proposed system is evaluated against three baseline models: BART, T5, and FLAN-T5. These models are widely utilized in various natural language processing (NLP) tasks, including text-to-SQL generation, and serve as robust benchmarks for assessing the effectiveness of our approach (*Lewis et al., 2020*; *Raffel et al., 2020*). The bidirectional and Auto-Regressive Transformers (BART) Model is a denoising autoencoder-based model that combines bidirectional and autoregressive transformers (*Brown et al., 2020*). Developed by Facebook AI Research (FAIR), it excels in sequence-to-sequence tasks such as text generation, summarization, and translation (*Vaswani et al., 2017*). The model architecture comprises a bidirectional encoder (similar to BERT) and an autoregressive decoder (similar to GPT), enabling it to capture contextual information and sequential dependencies effectively. This work uses fine-tuning a pre-trained BART model on a task-specific dataset to evaluate its performance in text-to-SQL tasks.

Google Research introduced the Text-to-Text Transfer Transformer (T5), which is built around a unified text-to-text framework where every NLP task is formulated as a text generation problem (*Radford et al., 2018*). Pretraining with a span-based objective, where masked spans are predicted, allows T5 to generalise across a wide range of tasks (*Houlsby et al., 2019*). We employ the T5 base model for our experiments and fine-tune it on a text-to-SQL dataset to benchmark its capabilities against our proposed system (*Sanh et al., 2019*). FLAN-T5 extends the T5 architecture by incorporating instruction tuning. This technique fine-tunes the model using a mixture of datasets and tasks in an instruction-based format (*Chung et al., 2022*). This enables FLAN-T5 to generalise to unseen tasks and perform better on domains outside its training data (*Wei et al., 2022*). For our research, we fine-tune the FLAN-T5 model on our text-to-SQL dataset to compare its performance against other baselines. These baseline models serve as substantial comparative benchmarks, allowing us to validate the improvements offered by our system, which integrates advanced techniques such as LoRA, QLoRA, and medical-domain optimisation.

PROPOSED MEDQUERY SYSTEM FOR EHR ACCESS

The MedQuery is designed to efficiently translate natural language queries into SQL statements tailored for medical datasets, as depicted in Fig. 1. This process is constructed as a pipeline comprising several stages: It begins with users providing queries in natural language. For example, a query like "*What is the average length of stay for patients*

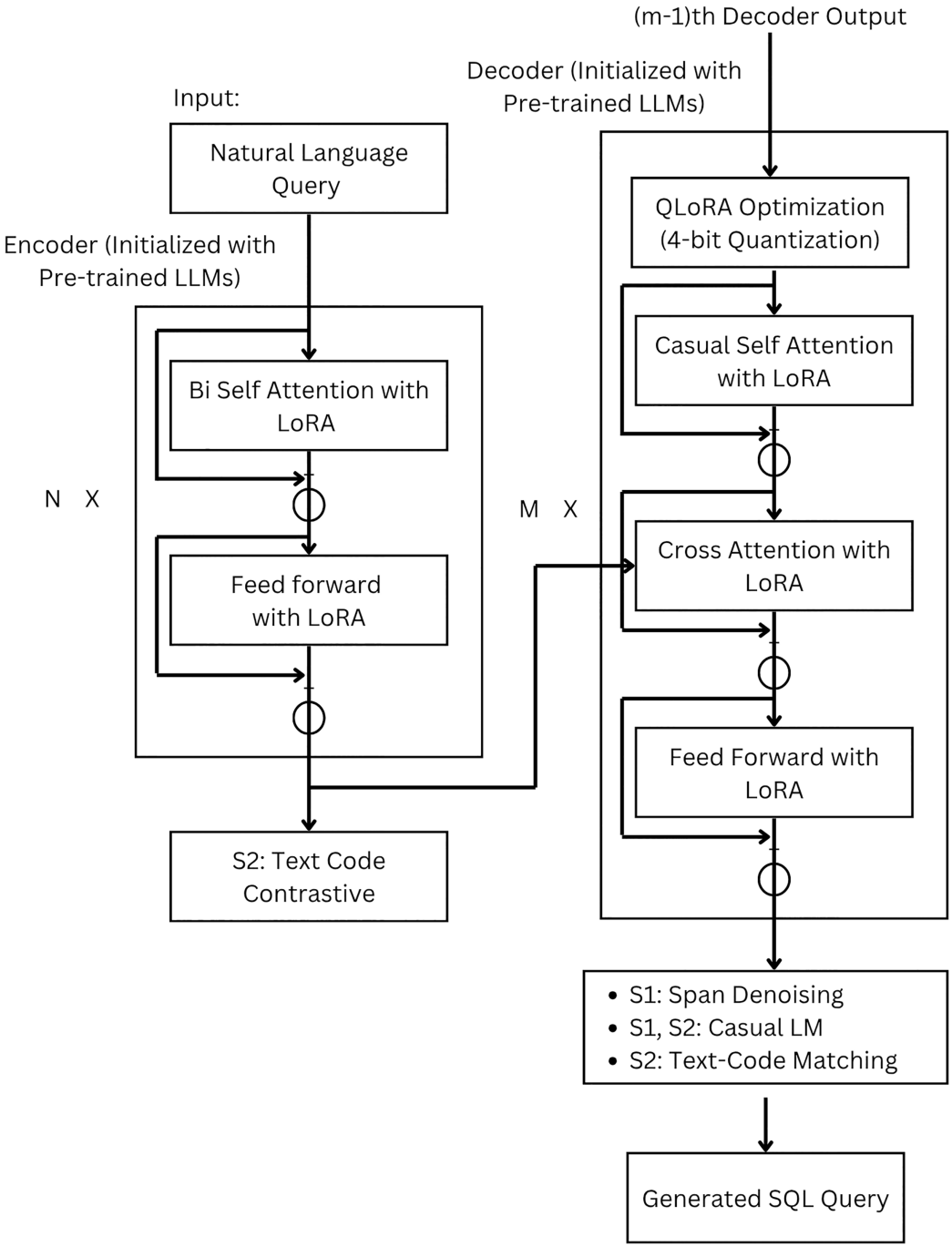


Figure 1 **Flow diagram for MedQuery process.** Full-size DOI: 10.7717/peerj-cs.3467/fig-1

diagnosed with diabetes?" is input into the system. This user-friendly interaction removes the need for technical knowledge of SQL syntax or database structure. Furthermore, the query proceeds to the NLP phase for tokenization, parsing, and preprocessing. A natural language query $T = (t_1, t_2, \ldots, t_n)$ where t_i represents the ith token in the query. A database schema $DS = (S_1, S_2, \ldots, S_m)$ where S_2 represents the ith table in the database,

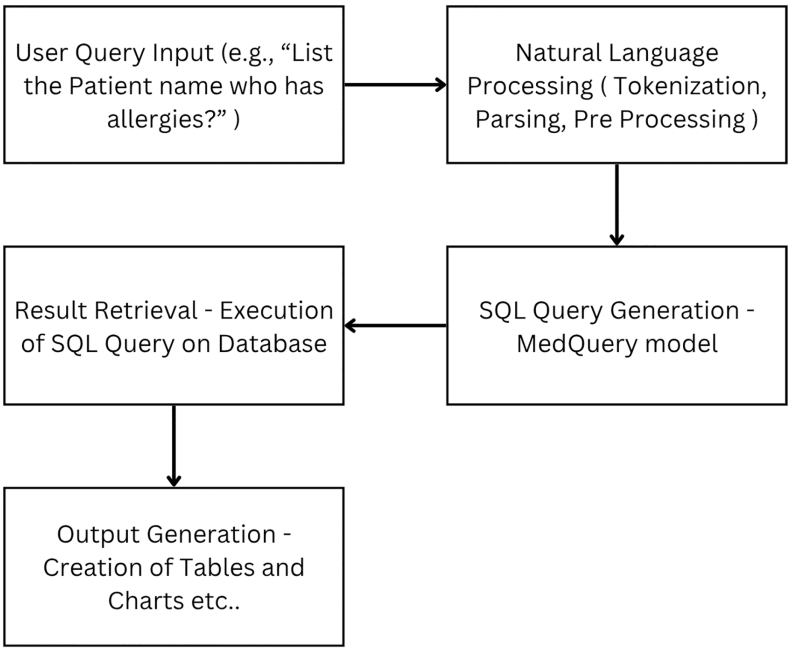


Figure 2 **Architecture for MedQuery with lightweight adaptation techniques.**
Full-size DOI: 10.7717/peerj-cs.3467/fig-2

and each table S_i includes columns C_{ij} like $S_i = \{C_{i1}, C_{i2}, \ldots, C_{ip}\}$. An SQL query can be represented as $\hat{Q} = (\hat{q}_1, \hat{q}_2, \ldots, \hat{q}_n)$, where $\hat{q}_l$ expressed as ith token in the query. The task is determining the SQL query $\hat{Q}$ that achieves the highest conditional probability given T and DS, as shown in Eq. (1):

$$\hat{Q} = arg \underset{Q}{Max}\ Prob(Q|T, DS). \tag{1}$$

Equation (1) indicates that the system selects the SQL query Q that is most likely, given the user's words (T) and the database layout (DS), similar to choosing the best translation for a sentence based on context. Tokenisation segments the query into meaningful tokens, parsing identifies syntactic structures, and preprocessing involves tasks such as normalization and stopword removal. These processes ensure the query is accurately interpreted for SQL generation. The SQL query generation component employs a robust encoder-decoder transformer architecture optimized with lightweight techniques known as Low-Rank Adaptation (LoRA) and Quantized Low-Rank Adaptation (QLoRA). These methods enhance the system's efficiency by reducing the computing resources required, which is crucial for practical use in healthcare settings with limited hardware resources. LoRA works by fine-tuning only a small set of additional parameters, rather than adjusting the entire model, much like updating just a few key parts of a large machine rather than rebuilding it entirely. This saves memory and speeds up the process without losing accuracy. QLoRA goes a step further by compressing the model's data into a smaller format (similar to shrinking a large file for storage), cutting down memory use even more down while still maintaining performance. Figure 2 illustrates the architecture, which

combines advanced model enhancements for efficient processing. This ensures that the translation from natural language queries to SQL is accurate and computationally efficient, making the system scalable and suitable for large medical datasets. In this architecture, the encoder transforms user inputs, such as the natural language query T and schema DS, into a contextualised representation H that captures the semantic meaning of the query, as shown in Eq. (2):

$$H = \text{Encoder}\,(T,\ DS), \tag{2}$$

H in Eq. (2) is a summary of the query's intent and the database's structure, created by the encoder to help build the right SQL. The encoder uses mechanisms like bi-directional self-attention, which looks at all words in the query from both directions (before and after) to understand context, calculated as given in Eq. (3). H is a hidden state that captures semantic and syntactic information from T and DS. It processes the query through the following mechanisms:

- Bi-directional Self-Attention with LoRA: This mechanism captures relationships between tokens in both directions, allowing the model to understand the context from both a token's left and right sides. The self-attention mechanism, a significant component of this process, can be mathematically described as shown in Eq. (3):

$$\text{Attention}(T, W, V) = \text{softmax}\left(\frac{\text{UW}^{\text{T}}}{\sqrt{\text{d}_{\text{w}}}}\right)V, \tag{3}$$

where U represents the query matrix, W represents the key matrix, V remains the value matrix, and d_W is the query/key vectors dimensionality. This formula enables the system to focus on the essential words in the query. LoRA enhances this by reducing the amount of data to process, making it faster and lighter. LoRA enhances this by reducing the number of parameters in the self-attention layer, making it more memory efficient. This is particularly useful in medical datasets, which often require processing complex queries with many tokens.

- Feed-Forward Networks with LoRA: These networks process each token independently, applying transformations while leveraging LoRA for lightweight adaptations. This enables the model to effectively capture complex features of the query, such as medical terminologies, without introducing excessive computational overhead.
- Text-Code Contrastive Objective: A text-code contrastive objective bridges the gap between natural language and SQL. This objective aligns the embeddings of semantically similar natural language queries and SQL statements. By training the model to map similar queries and their corresponding SQL code to similar vector spaces, the encoder gains a deeper understanding of the query's intent, which improves its ability to generate relevant SQL code.

Once the query is understood and encoded into an intermediate representation, the decoder generates the corresponding SQL query. The decoder uses the encoded

representation h to construct the SQL query $\hat{q}$, token by token. The probabilities of each token in the SQL query can be represented using Eq. (4):

$$P(Q|T, DS) = \prod_{i=1}^{k} P(\hat{q}_i|h, \hat{q}_{1:i-1}). \tag{4}$$

Equation (4) means the likelihood of the SQL query Q is the product of probabilities for each part qi, based on the summary H and previously built parts ($q\{1{:}i-1\}$), ensuring the query builds logically step by step. Tokens $\hat{q}_{1:i-1}$ are previously generated and $P(\hat{q}_i|h, \hat{q}_{1:i-1})$ is the probability of the ith token given context and previous tokens. The decoder's design incorporates the following key components:

- Causal Self-Attention with LoRA: The causal self-attention mechanism ensures that the SQL query is generated incrementally and logically, with each token relying only on previously generated tokens. This sequential generation ensures the SQL query is syntactically correct. LoRA is applied to the self-attention mechanism to maintain computational efficiency.
- Cross-Attention with LoRA: This mechanism allows the decoder to attend to the encoder's output, ensuring that the generated SQL query aligns with the user's input query. Cross-attention plays a critical role in ensuring that the SQL code reflects the exact meaning of the user's query. For example, if the query asks about the "average length of stay for patients diagnosed with diabetes," the decoder must ensure that it references the correct database tables and columns for "length of stay" and "diabetes."
- QLoRA Optimization: The decoder benefits from QLoRA (Quantized LoRA) optimization, which applies 4-bit quantization to the model's weights. This reduces the computational overhead, making it more efficient without sacrificing accuracy. In the context of medical datasets, which often involve large-scale data, QLoRA allows the system to handle high-volume queries while maintaining responsiveness and low latency.
- Feed-Forward Networks with LoRA: As in the encoder, feed-forward networks in the decoder apply transformations to tokens. These networks are optimized with LoRA for lightweight adaptations, ensuring that the system remains efficient even when generating complex SQL queries. This approach helps balance high model performance with minimal resource consumption, which is crucial for generating real-time medical queries.

As shown in Fig. 3, the LoRA mechanism optimizes both encoder and decoder components for lightweight adaptation, making the system highly efficient in terms of both computation and memory:

- Frozen Pretrained Weights (W): The pre-trained weights from the base model are retained, ensuring that the model can leverage the knowledge acquired during its initial training. This helps maintain high performance across various queries without requiring full retraining for each task.
- Low-Rank Matrices (A and B): LoRA introduces two low-rank matrices (A and B) representing the updates to the pre-trained weights. This reduces the number of

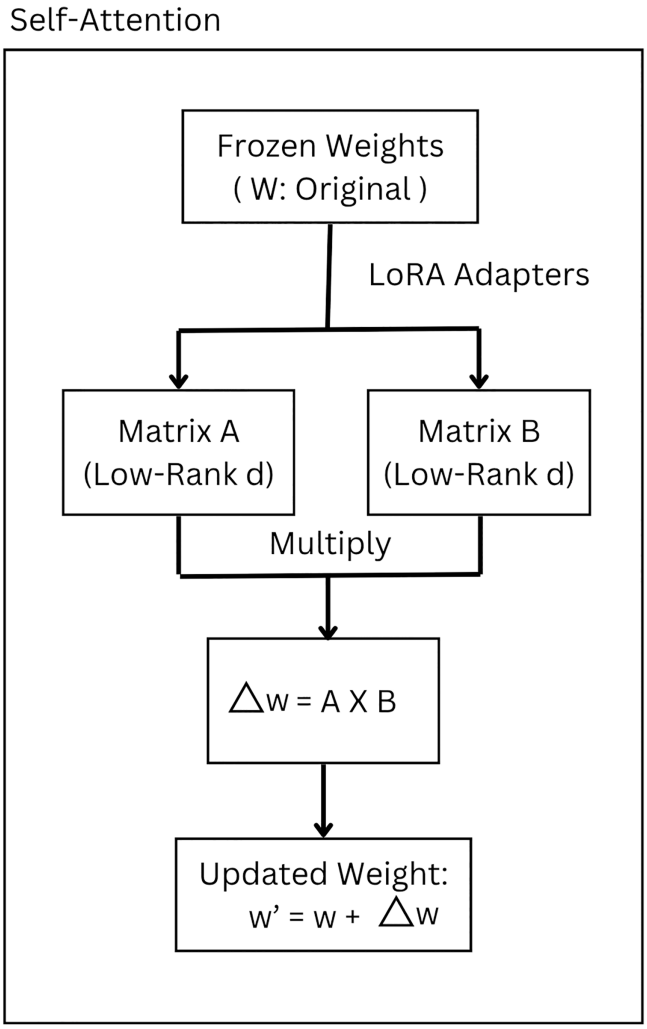


Figure 3 **Lightweight adoption process.** Full-size DOI: 10.7717/peerj-cs.3467/fig-3

trainable parameters, allowing for efficient fine-tuning of the model without significantly increasing resource requirements.

- Updated Weights (W′): The LoRA mechanism is integrated into attention layers to fine-tune the model efficiently. The final updated weights are computed as a combination of the frozen pre-trained weights and the low-rank updates. The updated weight matrix W′ is calculated as:

$$\delta' = \delta + \Delta\delta, \quad \Delta\delta = AB, \tag{5}$$

where δ is a pre-trained weight matrix, A and B are low-rank matrices. In Eq. (5), W is the original model setup, ΔW is a slight change made by multiplying two small parts A and B, and W′ is the new setup. This keeps updates efficient, like tweaking a recipe slightly instead of starting over.

- This approach drastically reduces the computational burden during fine-tuning while maintaining the model's accuracy and effectiveness.

Using LoRA and QLoRA, the proposed system optimizes the model's ability to generate SQL queries while minimizing resource usage, making it scalable to large datasets and ensuring it can run efficiently in real-world applications. This approach enables the system to handle a wide range of medical queries, from basic data retrieval to complex, multi-parameter analyses, without compromising speed or accuracy. The generated SQL query is executed in the result retrieval phase, which involves running the query against a structured medical database containing patient records, treatment histories, and other clinical data. This step ensures that the system retrieves accurate and relevant information while maintaining the integrity of the database schema.

Finally, the retrieved data is transformed into user-friendly outputs, including tabular formats for detailed analysis and visualizations like charts to provide clear insights. The output generation illustrates how the system presents results in an easily interpretable manner, enabling healthcare professionals to make informed decisions effectively and complex databases, empowering data-driven decision-making.

Dataset for MedQuery system

This work employs the Medical Information Mart for Intensive Care Structured Query Language (MIMICSQL) dataset (*Johnson et al., 2016*), a comprehensive and domain-specific resource tailored for text-to-SQL generation within the healthcare domain. It is constructed from the MIMIC-III database (*Yoon, Rumshisky & Szolovits, 2020*), which contains anonymized patient records from intensive care units, ensuring full compliance with privacy regulations and data protection standards. The dataset comprises 10,000 natural language question–SQL query pairs, designed to cover a wide range of SQL operations including aggregations, filtering, joins, and multi-table queries. These pairs are grounded in clinical data distributed across five normalized relational tables representing demographics, laboratory tests, diagnoses, procedures, and prescriptions.

Each instance is stored in a structured format, capturing both the linguistic and logical components of the query. The schema includes fields such as the database_id, which identifies the source database; interaction, which contains the natural language question and corresponding SQL query in structured form; query, which holds the raw SQL string; utterance, the natural language input; utterance_toks, a tokenized version of the utterance; and sql, a structured JavaScript Object Notation (JSON) representation that decomposes the query into components such as SELECT, WHERE, and GROUP BY clauses. The dataset is split into 80:10:10 proportions for training, validation, and testing, aligning with conventions in previous studies and ensuring reproducible evaluation.

In terms of query complexity, the dataset includes a wide range of SQL types. Approximately 42% of the queries are simple SELECT statements, while 21% include aggregation functions such as COUNT or AVG. Queries with GROUP BY clauses make up around 13% of the dataset, and about 15% involve joins between two or more tables. Nested or sub queries account for approximately 6%, and 31% of the queries include multiple WHERE conditions, often overlapping with other query types. This distribution reflects a balanced and realistic mix of query structures, supporting comprehensive evaluation of models across both complex and straightforward text-to-SQL generation scenarios (*Xie et al., 2021*).

Performance metrics to compare proposed model with baseline models

To evaluate the models' performance, we utilized a variety of well-established metrics that assess different aspects of text generation quality, including precision, recall, similarity, and structural accuracy (*Chen, Zhang & Zhang, 2021*). Below, we define each metric and its formula.

(1) ***BLEU score***

The BLEU score measures the precision of n-grams in the generated text compared to reference texts. It assesses how well the generated query matches reference queries in terms of exact phrasing and structure, which is critical in SQL syntax. Higher BLEU scores mean the generated output shares more exact word-sequence overlap with the reference. It evaluates how many n-grams in the generated output are present in the reference, considering multiple levels of n-grams as shown in Eq. (6).

$$BLEU = BP.\exp\left(\sum_{n=1}^{N} \omega_n \log P_n\right), \tag{6}$$

where:

- P_n is the precision of n-grams of size n.
- ω_n are the weights assigned to different n-gram sizes (typically uniform).
- BP (Brevity Penalty) is calculated as shown in Eq. (7):

$$BP = \begin{cases} 1 & if\ c > r \\ e^{(1-r/c)} & if\ c \leq r \end{cases} \tag{7}$$

Here, c is the length of the candidate text, and r is the length of the reference text. Higher BLEU scores indicate more remarkable similarity in n-gram structure between the generated and reference texts. However, BLEU does not account for recall or semantics.

(2) ***Exact Match (EM) accuracy***

This metric measures the percentage of generated output that matches the reference text, as shown in Eq. (8). It provides a strict correctness measure, ensuring that the generated SQL is identical to the expected one, which is essential for mission-critical use cases. In simple terms, it tells us how often the model gets the entire query perfectly correct.

$$M\ Accuracy = \frac{Number\ of\ Exact\ Matches}{Total\ Number\ of\ Samples} \times 100. \tag{8}$$

Exact Match Accuracy highlights how often the model's output is identical to the reference, emphasizing complete alignment in structure and content.

(3) ***ROUGE scores***

ROUGE metrics evaluate the overlap between the n-grams or subsequences in the generated and reference texts, as shown in Eq. (9). It emphasizes recall and helps identify

whether the generated query captures all relevant information from the reference, which is useful when multiple SQL representations are valid. These metrics are widely used in summarization tasks.

ROUGE-1 measures unigram (word-level) overlap.

ROUGE-2 measures bigram (two-word sequence) overlap.

$$\text{ROUGE} - \text{N} = \frac{\sum_{n-gram \in Reference} Count_{match}\ (n-gram)}{\sum_{n-gram \in Reference} Count_{ref}\ (n-gram)}, \tag{9}$$

where:

$Count_{match}\ (n-gram)$ is the number of n-grams common to both generated and reference texts.

$Count_{ref}(n-gram)$ is the total number of n-grams in the reference text.

ROUGE-L measures the longest common subsequence (LCS) between the generated and reference texts, as shown in Eq. (10).

$$ROUGE - L = \frac{LCS\ (Generated,\ Reference)}{Length\ of\ Reference} \tag{10}$$

ROUGE-Lsum is a variant of ROUGE-L, applied to summary-level analysis, which compares the LCS at the document level. ROUGE metrics emphasize recall, quantifying how much of the reference text is captured by the generated text.

(4) ***Levenshtein distance***

The Levenshtein distance measures the minimum number of single-character edits (insertions, deletions, or substitutions) needed to transform the generated text into the reference text. It captures minor syntactic differences and is suitable for measuring closeness even when the queries are not identical. In simple terms, it measures how many character edits are needed to correct the generated query.

Let $D(i,j)$ be the edit distance between the first i characters of the generated text and the first j characters of the reference text as shown in Eq. (11):

$$D(i,j) = \begin{cases} 0 & \text{if } i = 0 \text{ and } j = 0 \\ i & \text{if } j = 0 \\ j & \text{if } i = 0 \\ \min \begin{cases} D(i-1,\ j) + 1 \\ D(i, j-1) + 1 \\ D(i-1, j-1) + cost(i,j) \end{cases} & otherwise \end{cases}, \tag{11}$$

where $cost(i,j) = 0$ if the characters match; otherwise, it is 1.

(5) ***Jaccard similarity***

Jaccard similarity measures the similarity between the sets of unique tokens in the generated and reference texts. It evaluates semantic token overlap, helping to identify whether the key components (tables, columns, and operators) are used correctly. It checks

how much the two queries overlap on their key elements. Higher Jaccard similarity scores indicate a more significant overlap in content.

$$Jaccard\ Similarity = \frac{\left| T_{generated} \cap T_{reference} \right|}{\left| T_{generated} \cup T_{reference} \right|}, \tag{12}$$

where $T_{generated}$ and $T_{reference}$ are the sets of unique tokens in the text and reference texts, respectively, higher Jaccard similarity scores indicate a more significant overlap in content.

(6) ***Token-level F1-score***

The token-level F1-score evaluates the quality of generated queries by balancing precision and recall at the token level. **Precision measures the fraction of predicted tokens that are correct, while recall measures the fraction of reference tokens that are correctly predicted.** The F1-score is the harmonic mean of precision and recall, providing a single metric that reflects both accuracy and completeness. The formulas are given in Eqs. (13), (14) and (15).

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \tag{13}$$

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \tag{14}$$

$$F1\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall}. \tag{15}$$

The token-level F1-score provides a nuanced understanding of partial correctness, rewarding models that generate most of the correct tokens even when the entire query is not exact. In essence, it reflects how accurately and completely the generated query matches the reference at the token level.

RESULTS AND DISCUSSION

To enhance Text-to-SQL Translation for Seamless Electronic Medical Record Access, an advanced CodeT5 model is used in this proposed system. Furthermore, the Tesla T4 GPU (*NVIDIA, 2018*) provides an excellent computational performance for training deep learning models, such as To evaluate the computational, within the constraints of the available resources (*Houlsby et al., 2019*). The T4 GPU, with 16 GB of Graphics Double Data Rate 5 (GDDR5) memory, was chosen for its efficiency in handling large models and capability to accelerate training times significantly compared to standard CPUs. The system also utilized 16 GB of system RAM and a multi-core CPU, which supported the overall training environment by managing data processing, model parameter updates, and evaluation tasks. This hardware setup facilitated fine-tuning the models with large datasets while ensuring smooth execution throughout the training process.

In addition, we adopted the standard data partitioning strategy used in previous studies, such as Text Retrieval and Question Answering System (TREQS) (*Johnson et al., 2016*). The dataset was split into training, validation, and test sets in an 80:10:10 ratio, providing a balanced framework for performance evaluation and enabling fair comparison with

existing methodologies. The training dataset is structured with key columns such as database_id, interaction (containing SQL query and natural language utterance), query (raw SQL query), utterance (natural language version), utterance_toks (tokenized version of the utterance), and SQL (structured representation of the SQL query). An example query from the dataset illustrates how natural language questions are mapped to SQL queries. For instance, a query like "Count patients by gender for Asian ethnicity" is transformed into an SQL query that counts the number of patients grouped by gender, where ethnicity is 'Asian'. This detailed breakdown ensures a comprehensive understanding of the dataset's structure, statistics, and the task of generating SQL queries from natural language input.

Furthermore, the proposed and baseline models (*Loshchilov & Hutter, 2019*) were trained using the AdamW optimizer to handle sparse gradients and effective parameter updates efficiently. A learning rate 5e−5 was chosen to balance stability and convergence, with weight decay applied to prevent overfitting. Due to hardware constraints, a batch size of 4 per device was used, complemented by gradient accumulation over eight steps to simulate a larger adequate batch size. The training spanned 30 epochs to ensure convergence, with input and target sequences truncated and padded to a maximum length of 64 tokens for uniformity and efficient memory utilization.

To evaluate the computational efficiency and scalability of our approach, we conducted a comparative analysis between full Fine-Tuning and the parameter-efficient LoRA + QLoRA strategy based on key training parameters. Full fine-tuning requires updating all approximately 223 million model parameters, resulting in high GPU memory consumption of roughly 10–11 GB of VRAM. In contrast, LoRA + QLoRA drastically reduces the memory footprint to just 3.4–4.2 GB by quantizing the base model to 4-bit precision and fine-tuning only ~1.77M parameters (~1.15%) through low-rank adapters. The quantization configuration followed the standard QLoRA setup using NormalFloat4 (NF4) quantization with double quantization enabled, optimized using the PagedAdamW optimizer. The LoRA configuration was set with a rank (r) of 8, scaling factor (α) of 16, and dropout rate of 0.05, applied to the attention and feed-forward projection layers. All experiments were conducted for 30 epochs with a learning rate of 2e−4 and a fixed random seed of 42 to ensure reproducibility. This enables efficient training on modest hardware configurations without compromising model capacity or expressiveness. Additionally, the training time was significantly optimized: full fine-tuning required around 3.5–4.5 h over 30 epochs, whereas LoRA + QLoRA completed training in just ~2.5–3 h, benefiting from fewer back propagation steps and reduced parameter update overhead. During inference, LoRA + QLoRA also achieved a ~1.2× speedup, attributed to its lighter, quantized architecture. Overall, these results confirm the practicality and effectiveness of LoRA + QLoRA for deploying the MedQuery system in real-world clinical environments, particularly in settings with limited computational resources.

To assess the proposed model's performance, a comprehensive evaluation was conducted against three baseline models, T5, BART, and Flan T5, using six widely accepted metrics: BLEU Score, Exact Match Accuracy, ROUGE Scores, Levenshtein Distance, Jaccard Similarity, and Token-Level F1-score. These metrics were selected to provide a

Table 1 Performance comparison of text-to-SQL models across various evaluation metrics.

Metric	BART	T5	Flan T5	Proposed model
BLEU score	43.04	62.31	59.28	81.68
Exact match accuracy	15.50%	25.60%	20.60%	35.20%
ROUGE-1	0.579	0.851	0.826	0.901
ROUGE-2	0.413	0.831	0.780	0.879
ROUGE-L	0.572	0.848	0.821	0.897
ROUGE-Lsum	0.572	0.848	0.821	0.897
Average Levenshtein distance	66.81	42.80	46.35	26.99
Average Jaccard similarity	0.40	0.77	0.73	0.83
Token-level F1-score	0.59	0.98	0.96	0.96

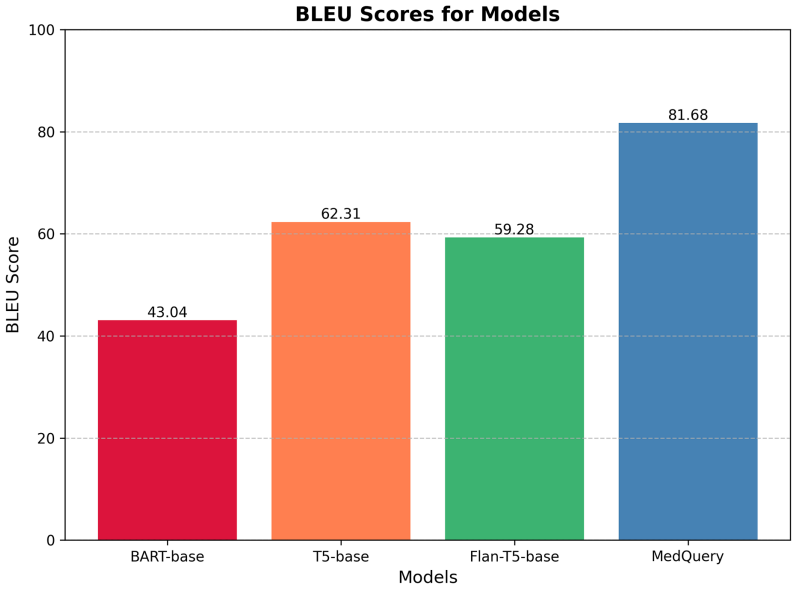


Figure 4 Comparison of BLEU score across different models.
Full-size DOI: 10.7717/peerj-cs.3467/fig-4

comprehensive view of the model's performance in terms of syntactic accuracy, semantic relevance, and output fidelity (*Sai, Mohankumar & Khapra, 2022*). The evaluation results are summarized in Table 1. These metrics underscore the proposed model's capability to handle the complexities of token alignment in natural language queries. Its architecture ensures high recall without compromising precision, enabling it to generate complete and accurate outputs. T5, while competitive, exhibited a slight imbalance between precision and recall, while BART's lower F1-score highlights its limitations in handling token-level details.

Figures 4, 5, 6, 7, 8, 9, and 10 visually depict the comparative performance across all metrics, highlighting the advantages of the proposed model's domain-specific optimizations and advanced processing techniques. These findings reinforce the model's

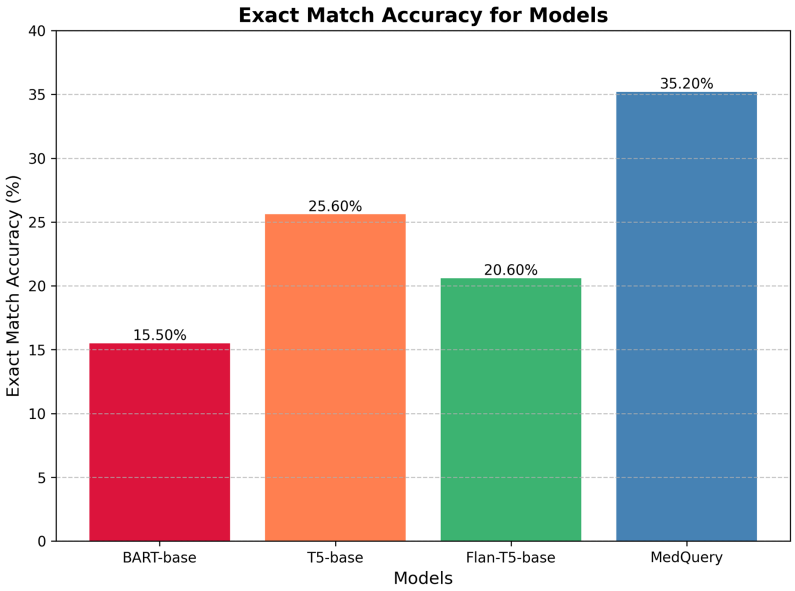


Figure 5 **Comparison of EM accuracy across different models.**
Full-size DOI: 10.7717/peerj-cs.3467/fig-5

capability to deliver high-quality, contextually accurate, and syntactically precise outputs for natural language database queries. The BLEU Score, a standard metric for evaluating n-gram overlap between the predicted and reference outputs, demonstrated MedQuery's strong capability in generating syntactically accurate responses. MedQuery achieved a score of 81.68, outperforming T5 (62.31), Flan T5 (59.28), and BART (43.04), as illustrated in Fig. 4.

The significant improvement in the BLEU score, particularly the **19.37% higher score** compared to T5, **22.4% higher score** compared to Flan T5, and **38.64% higher score** than BART, highlights the proposed model's ability to balance precision and recall in generating responses. This can be attributed to the proposed model's specialized architecture, which incorporates domain-specific embeddings and fine-tuned attention mechanisms for better capturing critical relationships in natural language queries. The relatively lower BLEU scores for T5, Flan T5, and BART suggest their challenges in maintaining n-gram coherence, mainly when applied to specialized queries, further emphasizing the advantages of a domain-adapted approach in the proposed model.

Exact Match Accuracy, which measures the percentage of outputs that exactly match the ground truth, revealed the limitations of general-purpose models like T5, Flan T5, and BART in handling specialized tasks. The proposed model achieved an exact match accuracy of **35.20%**, surpassing T5 (25.60%), Flan T5 (20.60%), and BART (15.50%). The results are visualized in Fig. 5. This significant margin of improvement indicates the proposed model's ability to accurately parse and understand complex queries. The proposed model's architecture is designed to handle the nuances of database-related queries, enabling it to generate responses that align more closely with the intended database representations. The lower performance of T5, Flan T5, and BART reflects their

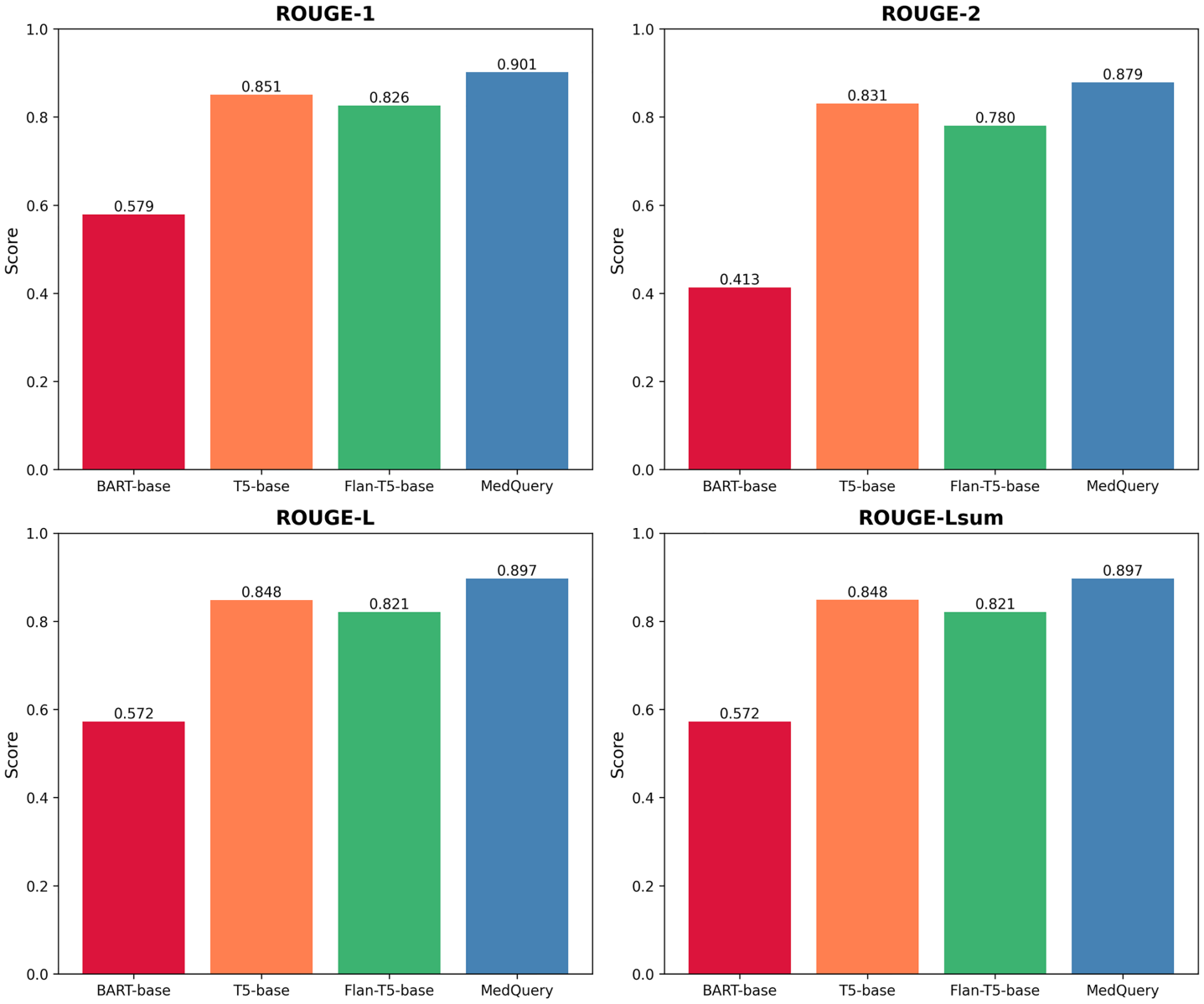


Figure 6 **Comparison of different ROUGE scores across different models.** Full-size DOI: 10.7717/peerj-cs.3467/fig-6

tendency to produce approximate results rather than precise mappings. This further supports the need for models tailored to domain-specific tasks, such as the proposed model.

ROUGE metrics, which measure the overlap of units such as n-grams, word sequences, and sentence structures between the generated and reference outputs, further validated the proposed model's superior performance. The values below summarize the ROUGE scores, while Fig. 6 illustrates the comparative results.

- ROUGE-1: The proposed model scored 0.901, outperforming T5 (0.851), Flan T5 (0.826), and BART (0.579).

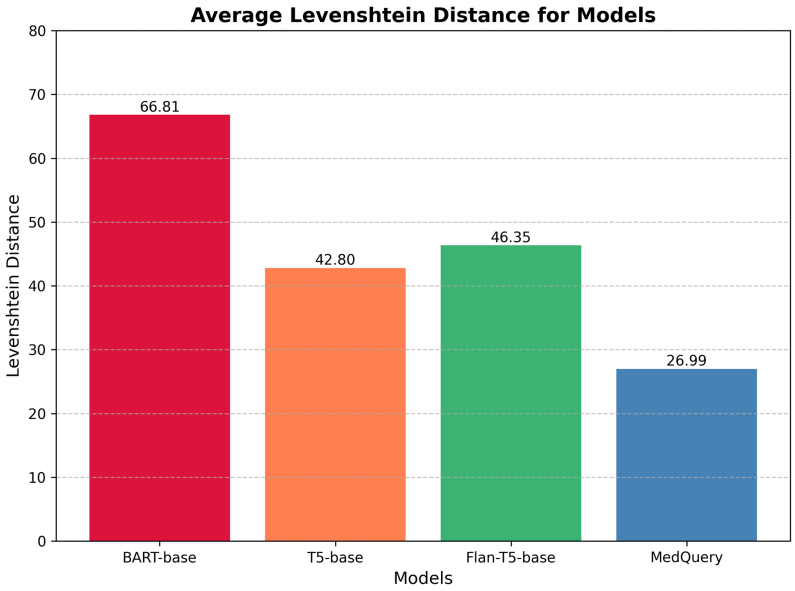


Figure 7 **Comparison of average Levenshtein distance across different models.**
Full-size DOI: 10.7717/peerj-cs.3467/fig-7

- ROUGE-2: The proposed model achieved 0.879, compared to T5 (0.831), Flan T5 (0.780), and BART (0.413).
- ROUGE-L: The proposed model scored 0.897, while T5 and Flan T5 achieved 0.848 and 0.821, respectively, and BART achieved 0.572.
- ROUGE-Lsum: The proposed model achieved a ROUGE-L performance score of 0.897.

These results emphasize the proposed model's proficiency in capturing both word-level and phrase-level relevance and preserving the structural coherence of responses. The model's specialized architecture, which includes enhanced token embeddings and a refined sequence alignment mechanism, allows it to generate outputs that are not only accurate but also contextually relevant. While competitive, T5, Flan T5, and BART exhibited slight reductions in recall and precision, likely due to their generic training objectives and lack of domain adaptation. Furthermore, the Levenshtein Distance, a metric that quantifies the minimum number of edits required to transform the predicted output into the ground truth, revealed MedQuery's efficiency in producing outputs that closely align with the reference. MedQuery achieved the lowest average distance of 26.99, compared to T5 (42.80), Flan T5 (46.35), and BART (66.81), as shown in Fig. 7.

This metric highlights the proposed model's ability to minimize errors at both lexical and structural levels. The significantly higher Levenshtein distance for BART indicates its challenges in maintaining syntactic accuracy and semantic relevance, often producing outputs with more significant deviations from the ground truth. T5 and Flan T5 performed better than BART but fell short of the proposed model, underlining the latter's tailored optimizations for database query generation. Jaccard Similarity, which evaluates the overlap between sets of words in the predicted and ground truth outputs, provided

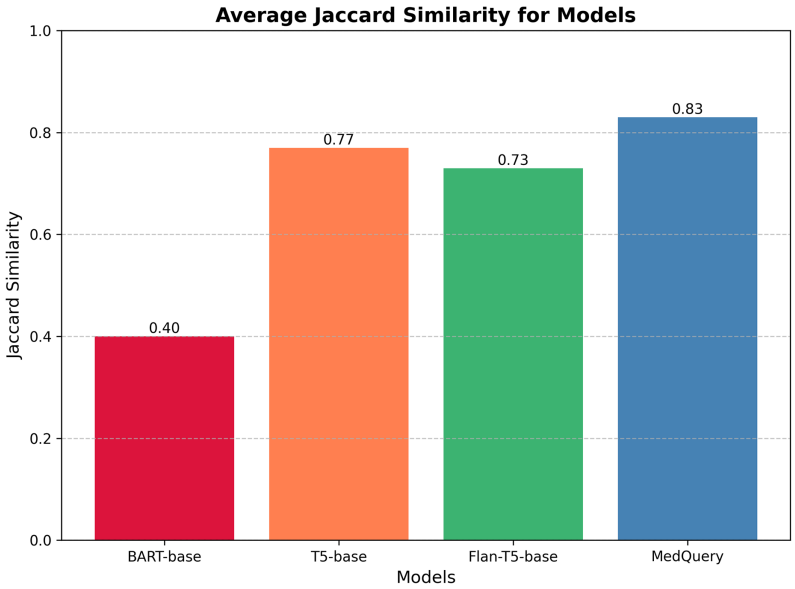


Figure 8 **Comparison of average Jaccard similarity across different models.**
Full-size DOI: 10.7717/peerj-cs.3467/fig-8

additional insights into model performance. The proposed model achieved the highest similarity score of 0.83, outperforming T5 (0.77), Flan T5 (0.73), and BART (0.40), as depicted in Fig. 8.

The proposed model's higher Jaccard similarity score reflects its ability to maintain high lexical overlap with the ground truth while avoiding unnecessary tokens. This outcome stems from its refined token selection and attention mechanisms, which focus on preserving key elements in the responses. The lower scores for T5, Flan T5, and BART indicate their tendency to generate extraneous or irrelevant tokens, reducing overall similarity to the ground truth.

The Token-Level F1-score, a combined token-level precision and recall measure, confirmed the proposed model's robust performance in capturing fine-grained token relationships. The proposed model achieved an F1-score of 0.96, surpassing T5 (0.98), Flan T5 (0.96), and BART (0.59) by notable margins. These results are visualized in Fig. 9. While the proposed model's F1-score is slightly lower than T5's (by 0.02), this can be attributed to the model's focus on achieving both precision and recall while maintaining contextual accuracy. T5's slightly higher F1-score indicates that its token-level precision and recall are balanced more optimally. However, the proposed model's advantage lies in its ability to generate more accurate and contextually relevant responses, which results in higher BLEU, ROUGE, and Jaccard scores. BART's relatively lower F1-score reflects its incredible difficulty handling token-level details, particularly in specialized domains.

Figure 10 illustrates the convergence of training loss for T5, BART, FLAN T5, and the proposed model over 10,000 training steps. While all models show a decrease in training loss, the proposed model (based on the code T5 architecture) outperforms the others in terms of both speed and stability. The proposed model significantly reduces loss from

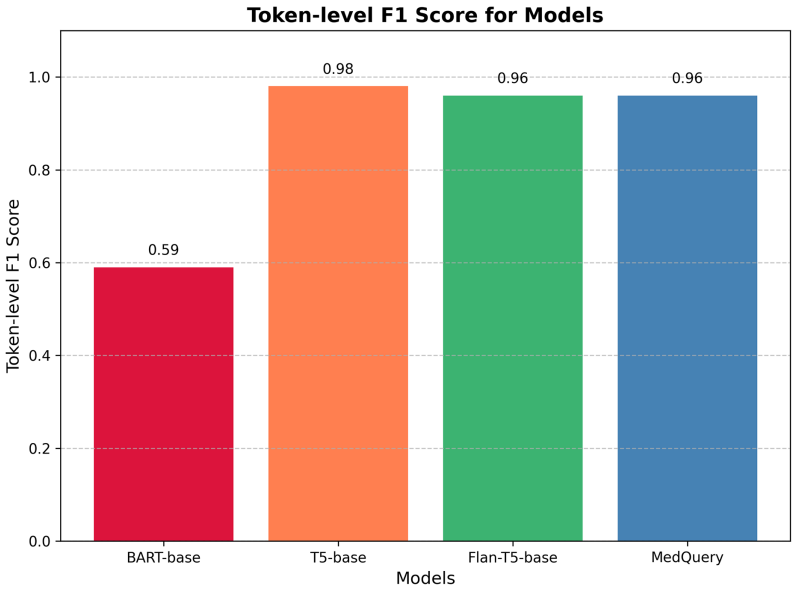


Figure 9 **Comparison of token-level F1-score across different models.**
Full-size DOI: 10.7717/peerj-cs.3467/fig-9

Figure 10 **Convergence analysis of training loss across different models.**
Full-size DOI: 10.7717/peerj-cs.3467/fig-10

0.0231 at 500 steps to 0.0002 at 6,000 steps. It demonstrates its ability to quickly learn and optimize, likely due to its domain-specific fine-tuning for database query generation tasks. In contrast, T5 and FLAN T5, despite their strong general-purpose capabilities, exhibit slower and less stable convergence, with T5's loss reducing from 0.0577 to 0.0038, and FLAN T5 from 0.0530 to 0.0049.

Table 2 Ablation study comparing full fine-tuning, LoRA, QLoRA and LoRA + QLoRA configurations of the CodeT5 model.

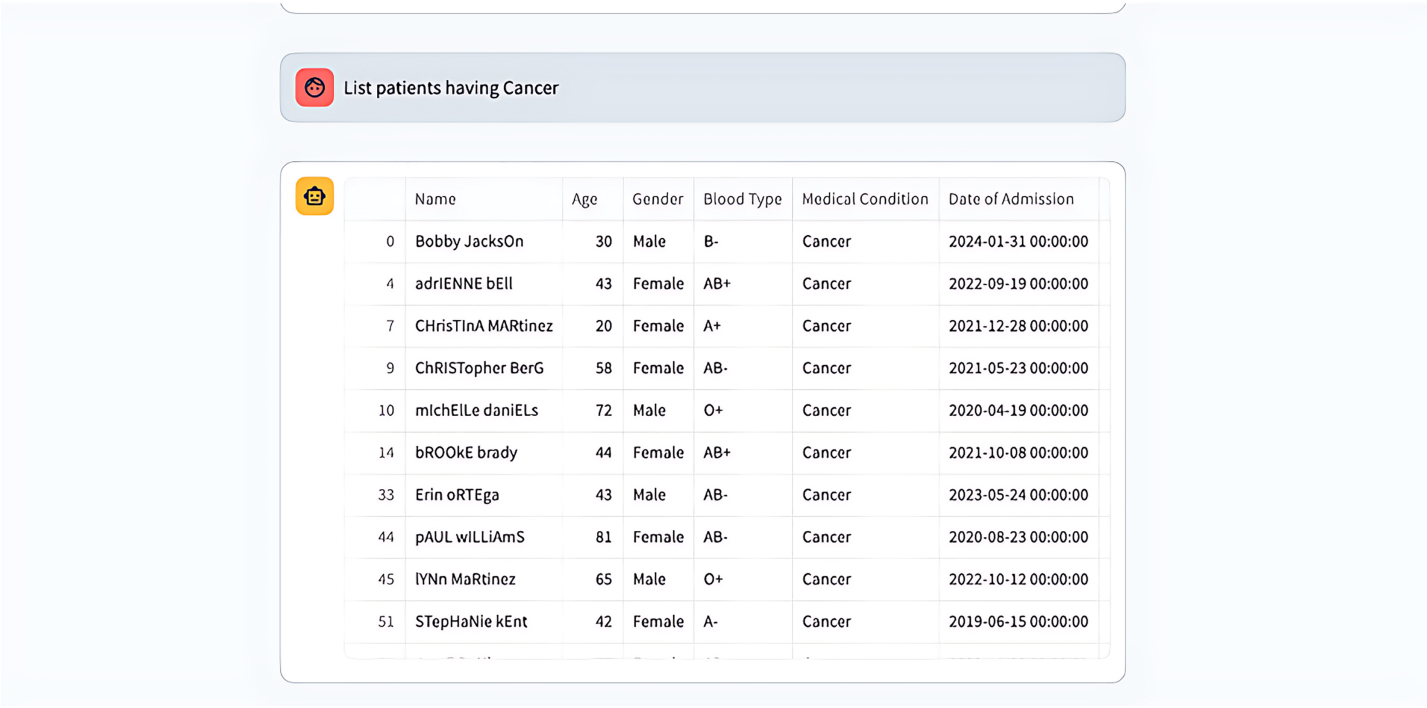
Configuration	Trainable parameters	GPU memory (GB)	Training time (h)	BLEU	Exact match (%)	ROUGE-L	Levenshtein ↓	Jaccard ↑	Token-F1
Full fine-tuning (CodeT5)	223 M (100%)	10.8	4.2	79.50	33.8	0.890	29.4	0.82	0.95
LoRA only	1.77 M (1.15%)	5.1	3.2	80.20	34.5	0.893	28.7	0.82	0.96
QLoRA only	223 M (quantized 4-bit)	4.2	3.0	79.85	34.1	0.891	28.2	0.81	0.95
LoRA + QLoRA (Proposed)	1.77 M (1.15%)	3.8	2.8	81.68	35.20	0.897	26.99	0.83	0.96

To quantify the contributions of the LoRA and QLoRA components, an ablation study was conducted using four configurations of the CodeT5 backbone: (i) full fine-tuning, (ii) LoRA-only fine-tuning, (iii) QLoRA-only fine-tuning, and (iv) combined LoRA + QLoRA fine-tuning (Proposed Model). The LoRA configuration employed a rank (r) of 8, scaling factor (α) of 16, and dropout rate of 0.05, applied to the attention and feed-forward projection layers. QLoRA utilized NormalFloat4 (NF4) quantization with double quantization enabled and was optimized using the PagedAdamW optimizer. All experiments were trained for 30 epochs with a learning rate of 2e−4 and a fixed random seed of 42 for reproducibility.

As shown in Table 2, the ablation results demonstrate that LoRA alone maintains comparable accuracy to full fine-tuning while reducing GPU memory usage by nearly 50% and training time by ~24%. QLoRA-only fine-tuning yields further memory reduction due to quantization, lowering VRAM usage to 4.2 GB, though with a minor drop in BLEU and ROUGE scores compared to LoRA-only. When both techniques are combined, the LoRA + QLoRA configuration achieves the best overall balance—offering the lowest memory footprint (3.8 GB), fastest training (2.8 h), and highest accuracy (BLEU = 81.68, ROUGE-L = 0.897, Token-F1 = 0.96). These findings confirm that the integration of LoRA and QLoRA delivers the most computationally efficient yet accurate solution for EMR text-to-SQL generation. The table summarizes trainable parameters, GPU memory usage, training time, and key evaluation metrics across four configurations of the CodeT5 model.

The superior performance of the proposed model results from its optimised architecture and targeted training for specialised database queries. A key innovation is the integration of text-code contrastive learning, which improves semantic alignment between natural language queries and their corresponding SQL representations. This mechanism, absent in baseline models such as BART, T5, and Flan-T5, enables the model to better capture both intent and structure. Consequently, the model achieves faster convergence, more accurate learning, minimal fluctuations in training loss, and significantly higher BLEU, ROUGE, and Exact Match scores.

To demonstrate the practical utility and real-world integration of the proposed system in Electronic Medical Record (EMR) environments, two representative use cases are presented. These examples illustrate the complete pipeline—from natural language input to SQL query generation and final output presentation—in a manner that is intuitive for



	Name	Age	Gender	Blood Type	Medical Condition	Date of Admission
0	Bobby JacksOn	30	Male	B-	Cancer	2024-01-31 00:00:00
4	adrIENNE bEll	43	Female	AB+	Cancer	2022-09-19 00:00:00
7	CHrisTInA MARtinez	20	Female	A+	Cancer	2021-12-28 00:00:00
9	ChRISTopher BerG	58	Female	AB-	Cancer	2021-05-23 00:00:00
10	mIchElLe daniELs	72	Male	O+	Cancer	2020-04-19 00:00:00
14	bROOkE brady	44	Female	AB+	Cancer	2021-10-08 00:00:00
33	Erin oRTEga	43	Male	AB-	Cancer	2023-05-24 00:00:00
44	pAUL wILLiAmS	81	Female	AB-	Cancer	2020-08-23 00:00:00
45	lYNn MaRtinez	65	Male	O+	Cancer	2022-10-12 00:00:00
51	STepHaNie kEnt	42	Female	A-	Cancer	2019-06-15 00:00:00

Figure 11 **Tabular output for the query "List patients having Cancer".** Full-size DOI: 10.7717/peerj-cs.3467/fig-11

clinicians and administrators. In the first use case, a clinician inputs the query, "List patients having Cancer." The system generates the SQL query:

```
SELECT Name, Age, Gender, Blood_Type, Medical_Condition, Date_of_Admission
FROM Patients
WHERE Medical_Condition = 'Cancer';
```

This retrieves relevant patient records and presents them in a structured tabular format as shown in Fig. 11, allowing clinicians to efficiently review individual patient details such as age, gender, blood type, and date of admission. This functionality facilitates the identification of patient cohorts for follow-up or clinical assessment, eliminating the need for manual database operations. In the second use case, an administrator enters the query, "Pie chart for Blood group," prompting the system to generate the following SQL query:

```
SELECT Blood_Type, COUNT(*)
FROM Patients
GROUP BY Blood_Type;
```

Here, the system recognises the intention for a statistical summary and displays a pie chart illustrating the percentage distribution of blood types among patients as shown in Fig. 12. This visual output supports administrative decisions, such as managing blood bank inventories or analysing patient demographics. These use cases showcase the system's

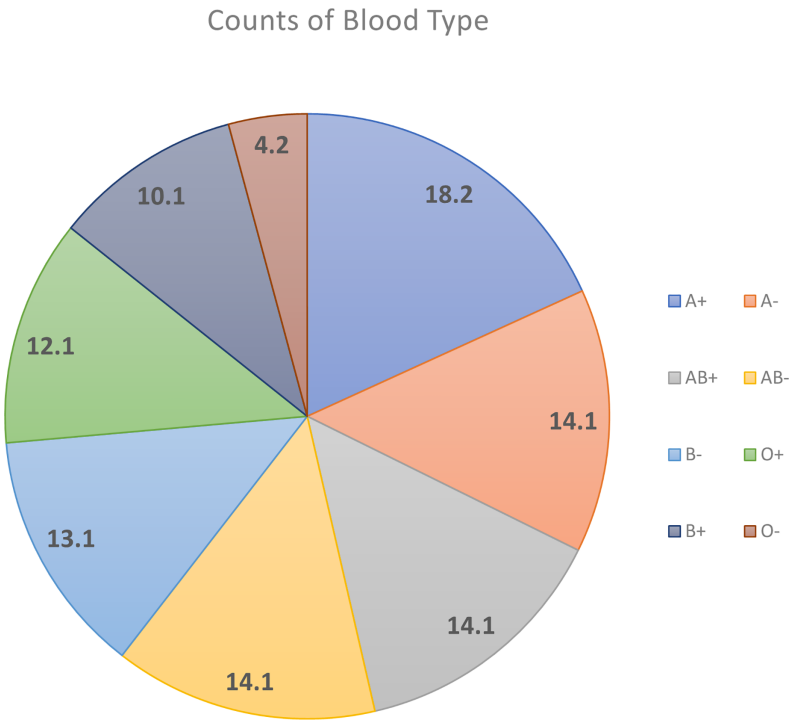


Figure 12 **Pie chart showing blood type distribution among patients.**
Full-size DOI: 10.7717/peerj-cs.3467/fig-12

ability to accurately interpret domain-specific natural language queries, generate executable SQL commands, and deliver user-friendly outputs in both textual and visual formats. This integrated workflow enhances accessibility to EMR data, enabling better-informed decision-making in both clinical and operational contexts.

Although MedQuery demonstrates strong performance across most natural language inputs, an error analysis was conducted to identify common failure patterns. Ten representative mispredicted or partially correct SQL generations were manually examined. For example, the model occasionally produced SUM() instead of AVG() in aggregation tasks or omitted the GROUP BY clause for grouped computations. Case sensitivity issues like "Asthma" *vs* "asthma" and inconsistent data representations (*e.g.*, 1 *vs* "1") also caused mismatches during query execution. Logical errors were another category, including incorrect use of AND instead of OR in filter conditions or incomplete constraints such as retrieving all asthma patients instead of only female asthma patients. In a few instances, operator or format inconsistencies, such as using "!=" instead of "<>" or dates in "2024/05/10" instead of "2024-05-10," were observed. Despite these issues, the model generally preserved the semantic intent of the user query, with most failures localized to specific clauses rather than complete misinterpretations.

The key contribution of this work is the development of MedQuery, a parameter-efficient, domain-tuned text-to-SQL model for the EMR domain. Leveraging LoRA and QLoRA for lightweight fine-tuning of the CodeT5 backbone, the system minimizes trainable parameters and GPU usage while maintaining high accuracy. It implements an

end-to-end NLP-to-SQL pipeline that translates natural language queries into executable SQL and visualizes results through charts and tables. Experimental results show that MedQuery outperforms baselines such as BART, T5, and Flan-T5 across BLEU, ROUGE, and Exact Match metrics, with further validation from Levenshtein, Jaccard, and Token-level F1-scores, demonstrating both efficiency and robustness.

CONCLUSIONS

In this work, we proposed a natural language interface powered by CodeT5 for querying medical databases. Our approach addresses the limitations of conventional query systems by enabling users to interact with databases using natural language. We evaluated the performance of the proposed model, MedQuery, in comparison to T5 and BART across various evaluation metrics, including BLEU score, ROUGE metrics, exact match accuracy, Jaccard similarity, Levenshtein distance, and token-level F1-score. The results demonstrate that MedQuery outperforms T5 and BART across BLEU, ROUGE, exact match, Jaccard similarity, and Levenshtein distance, while achieving competitive performance with a slightly lower token-level F1-score (0.96 *vs.* 0.98 for T5). These findings highlight MedQuery's ability to generate high-quality queries and ensure accurate similarity with reference outputs. Overall, the proposed model shows strong potential to streamline database interactions, particularly in healthcare, by providing accurate and efficient natural language querying.

Limitations: Despite these benefits, the proposed work has some limitations. Because it is optimized for medical datasets, its performance may be hindered when creating SQL queries for other domains. Furthermore, while it improves accuracy, it may still struggle with highly complex multi-condition queries or nested SQL structures. Another significant drawback is the absence of effective error-handling capabilities; if a query fails due to schema mismatches or poor SQL syntax, the model does not self-correct or offer alternatives.

Future work: To address these challenges, future directions include extending MedQuery to support multi-turn dialogue systems, enabling the interactive refinement of queries in real-time. Enhancements will also focus on improving the ability to handle nested and complex logical SQL queries, as well as broadening the system's applicability to diverse EMR datasets and cross-domain scenarios. Additionally, integrating robust error-handling and self-correction mechanisms will strengthen system reliability. Finally, efforts will be made to improve scalability and interpretability to support deployment in real-world healthcare environments.

ADDITIONAL INFORMATION AND DECLARATIONS

Funding

The authors received no funding for this work.

Competing Interests

The authors declare that they have no competing interests.

Author Contributions

- Gomathi Bakthavatchalu conceived and designed the experiments, performed the experiments, analyzed the data, performed the computation work, prepared figures and/or tables, authored or reviewed drafts of the article, and approved the final draft.
- Saravana Balaji Balasubramanian conceived and designed the experiments, performed the experiments, analyzed the data, performed the computation work, authored or reviewed drafts of the article, and approved the final draft.
- Omar Khattab performed the experiments, analyzed the data, authored or reviewed drafts of the article, and approved the final draft.
- Mhd Omar Al-Kadri performed the experiments, prepared figures and/or tables, and approved the final draft.
- Dhanushkumar Vadivel conceived and designed the experiments, performed the computation work, prepared figures and/or tables, and approved the final draft.
- Harish Prasath V. g. conceived and designed the experiments, analyzed the data, performed the computation work, prepared figures and/or tables, and approved the final draft.
- Srinivasa Pradeep S. analyzed the data, performed the computation work, prepared figures and/or tables, and approved the final draft.
- Somaiyeh MahmoudZadeh analyzed the data, performed the computation work, authored or reviewed drafts of the article, and approved the final draft.

Data Availability

The following information was supplied regarding data availability:

The code is available in the Supplemental Files.

The data is available at GitHub:
https://github.com/wangpinggl/TREQS.

Supplemental Information

Supplemental information for this article can be found online at http://dx.doi.org/10.7717/peerj-cs.3467#supplemental-information.

REFERENCES

Badawy M, Ramadan N, Hefny HA. 2024. Big data analytics in healthcare: Data sources, tools, challenges, and opportunities. *Journal of Electrical Systems and Information Technology* **11(63)**:1–20 DOI 10.1186/s43067-024-00190-w.

Brown T, Mann B, Ryder N, Subbiah M, Kaplan JD, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, Agarwal S. 2020. Language models are few-shot learners. In: *Advances in Neural Information Processing Systems*, Vol. 33, 1877–1901.

Chen X, Zhang C, Zhang L. 2021. A comprehensive evaluation of text-to-SQL models. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. DOI 10.48550/arXiv.2108.01074.

Chung HW, Hou L, Longpre S, Zoph B, Tay Y, Fedus W, Li Y, Wang X, Dehghani M, Brahma S, Webson A. 2022. Scaling instruction-finetuned language models. *Journal of Machine Learning Research* **25(70)**:1–53.

Devlin J, Chang MW, Lee K, Toutanova K. 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)*. Minneapolis, MN: Association for Computational Linguistics, 4171–4186.

Fan J, Gu Z, Zhang S, Chen Z, Cao L, Li G, Madden S, Du X, Tang N. 2024. Combining small language models and large language models for zero-shot NL2SQL. *Proceedings of the VLDB Endowment* **17(11)**:2750–2763 DOI 10.14778/3681954.3681960.

Houlsby N, Giurgiu A, Jastrzebski S, Morrone B, Laroussilhe QD, Gesmundo A, Attariyan M, Gelly S. 2019. Parameter-efficient transfer learning for NLP. In: *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2790–2799.

Jaccard P. 1912. The Distribution of the Flora in the Alpine Zone. *New Phytologist* **11(2)**:37–50 DOI 10.1111/j.1469-8137.1912.tb05611.x.

Johnson AE, Pollard TJ, Shen L, Lehman LH, Feng M, Ghassemi M, Moody B, Szolovits P, Celi LA, Mark RG. 2016. MIMIC-III, a freely accessible critical care database. *Scientific Data* **3(1)**:160035 DOI 10.1038/sdata.2016.35.

Lalitha S, Prashanthi G, Puranam S, Vemula SR, Doulathbaji P. 2023. Natural language to SQL: automated query formation using NLP techniques. *E3S Web of Conferences* **391**:1115 DOI 10.1051/e3sconf/202339101115.

Levenshtein VI. 1966. Binary codes: capable of correcting deletions. *Insertions, and Reversals. Soviet Physics Doklady* **10(8)**:707–710 DOI 10.1109/isit.1994.394907.

Lewis M, Liu Y, Goyal N, Ghazvininejad M, Mohamed A, Levy O, Stoyanov V, Zettlemoyer L. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In: *Proceedings of the 58th annual meeting of the Association for Computational Linguistics*, 7871–7880.

Lin CY. 2004. ROUGE: a package for automatic evaluation of summaries. In: *Proceedings of the Workshop on Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, 74–81.

Loshchilov I, Hutter F. 2019. Decoupled weight decay regularization. In: *International Conference on Learning Representations (ICLR)*. DOI 10.48550/arXiv.1711.05101.

Lu Q, Dou D, Nguyen T. 2022. ClinicalT5: a generative language model for clinical text. In: *Findings of the Association for Computational Linguistics: EMNLP 2022*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, 5436–5443 DOI 10.18653/v1/2022.findings-emnlp.398.

Luo R, Sun L, Xia Y, Qin T, Zhang S, Poon H, Liu TY. 2022. BioGPT: generative pre-trained transformer for biomedical text generation and mining. *Briefings in Bioinformatics* **23(6)**:bbac409 DOI 10.1093/bib/bbac409.

NVIDIA. 2018. *Tesla T4: high-performance deep learning acceleration*. Santa Clara, CA: NVIDIA Corporation. *Available at https://www.nvidia.com/en-us/data-center/tesla-t4/*.

Papineni K, Roukos S, Ward T, Zhu W. 2002. BLEU: a method for automatic evaluation of machine translation. In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, 311–318.

Radford A, Narasimhan K, Salimans T, Sutskever I. 2018. Improving language understanding by generative pre-training. *OpenAI* **3**:1–12.

Raffel C, Shazeer N, Roberts A, Lee K, Narang S, Matena M, Zhou Y, Li W, Liu PJ. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research* **21(140)**:1–67.

Ren T, Fan Y, He Z, Huang R, Dai J, Huang C, Jing Y, Zhang K, Yang Y, Wang XS. 2024. PURPLE: Making a Large Language Model a Better SQL Writer. ArXiv DOI 10.48550/arXiv.2403.20014.

Sai AB, Mohankumar AK, Khapra MM. 2022. A survey of evaluation metrics used for natural language generation systems. *ACM Computing Surveys* **55(2)**:1–39 DOI 10.1145/3485766.

Sanh V, Debut L, Chaumond J, Wolf T. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper, and lighter. ArXiv DOI 10.48550/arxiv.1910.01108.

Singhal K, Azizi S, Tu T, Mahdavi SS, Wei J, Chung HW, Scales N, Tanwani A, Cole-Lewis H, Pfohl S, Payne P, Seneviratne M, Gamble P, Kelly C, Babiker A, Schärli N, Chowdhery A, Mansfield P, Demner-Fushman D, Agüera y Arcas B, Webster D, Corrado GS, Matias Y, Chou K, Gottweis J, Tomasev N, Liu Y, Rajkomar A, Barral J, Semturs C, Karthikesalingam A, Natarajan V. 2023. Large language models encode clinical knowledge. *Nature* **620(7972)**:172–180 DOI 10.1038/s41586-023-06291-2.

Sivasubramaniam S, Osei-Akoto C, Zhang Y, Stockinger K, Fürst J. 2024. SM3-text-to-query: synthetic multi-model medical text-to-query benchmark. In: *38th Conference on Neural Information Processing Systems (NeurIPS 2024)*. Curran Associates, Inc., 88627–88663.

Sivasubramaniam S, Osei-Akoto CE, Zhang Yi, Stockinger K, Fürst J. 2024. SM3-text-to-query: synthetic multi-model medical text-to-query benchmark. *Advances in Neural Information Processing Systems* **37**:88627–88663 DOI 10.52202/079017-2812.

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. 2017. Attention is all you need. In: *31st Conference on Neural Information Processing Systems (NIPS 2017)*. Long Beach, CA, USA, Vol. 30, 5998–6008.

Wei J, Wang X, Schuurmans D, Bosma M, Xia F, Chi E, Le QV, Zhou D. 2022. Chain of thought prompting elicits reasoning in large language models. In: *Proceedings of the 36th International Conference on Neural Information Processing Systems*, Vol. 35, 24824–24837.

Xie S, Zhang Y, Li Y, Zhang R. 2021. A data-driven approach to enhance text-to-SQL generation for healthcare. *Journal of Biomedical Informatics* **119(14)**:103826 DOI 10.1016/j.jbi.2021.103826.

Yoon C, Rumshisky A, Szolovits P. 2020. MIMICSQL: a large-scale text-to-SQL dataset for healthcare queries. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. DOI 10.48550/arXiv.2010.12773.

Zhang Y, Liu L. 2019. Natural language processing for SQL query generation in healthcare applications. In: *Proceedings of the 5th International Conference on Artificial Intelligence in Medicine*, 312–318.