ANALYSIS AND DEVELOPMENT OF TEXT-TO-SQL TRANSLATION SYSTEM USING LARGE LANGUAGE MODELS (LLMs)

Thesis Submitted
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF TECHNOLOGY
in
Data Science

by
Pradyumn Shukla
(2K23/DSC/14)

Under the supervision of
Dr. Sanjay Patidar
Assistant Professor, Department of Software Engineering,
Delhi Technological University

Department of Software Engineering

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Bawana Road, Delhi - 110042, India

May, 2025

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

CANDIDATE DECLARATION

I, Pradyumn Shukla, hereby certify that the work which is being presented in the thesis entitled Analysis and development of Text-to-SQL Translation system Using Large Language Models(LLMs) in partial fulfillment of the requirements for the award of the Degree of Master of Technology submitted in the Department of Software Engineering, Delhi Technological University in an authentic record of my work carried out during the period from August 2023 to May 2025 under the supervision of Dr. Sanjay Patidar.

The matter presented in the thesis has not been submitted by me for the award of any other degree of this or any other Institute.

Pradyumn Shukla

This is to certify that the student has incorporated all the corrections suggested by the examiner in the thesis and the statement made by the candidate is correct to the best of our knowledge.

Signature of Supervisor   Signature of External Examiner

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Shahbad Daulatpur, Main Bawana Road,Delhi-42

CERTIFICATE BY THE SUPERVISOR

I hereby certify that Pradyumn Shukla (Roll no 2K23/DSC/14) has carried out their research work presented in this thesis entitled "Analysis and development of Text-to-SQL Translation system Using Large Language Models (LLMs)" for the award of Master of Technology from the Department of Data Science, Delhi Technological University, Delhi under my supervision. The thesis embodies results of original work, and studies are carried out by the student himself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Dr. Sanjay Patidar
Assistant Professor
Department of Software Engineering
DTU-Delhi,
India

Place: Delhi

Date:

Analysis and development of Text-to-SQL Translation system Using Large Language Models (LLMs)

Pradyumn Shukla

ABSTRACT

The increasing reliance on data-driven decision-making has brought intuitive database access into limelight, particularly for inexperienced users. Text-to-SQL technologies bridge this shortcoming by converting natural language queries to SQL queries and thereby render database interaction more intuitive. Large Language Models have also influenced the Text2SQL system paradigm towards predicting correct and context-aware SQL. This survey maps the historical development of Text2SQL approaches from rule-based systems to LLM-based neural models. Extensive efforts have gone into using prompt engineering, schema alignment methods, and domain fine-tuning to ensure higher accuracy and generality. The models now exhibit significant progress in understanding complex queries as well as precise SQL code generation through emergent Large Language Model capabilities. The early systems had extremely strong template-based or rule-based mechanisms, whereas generation these days is extremely advanced neural systems brimming with domain knowledge and highly specialized embeddings. Although LLMs, particularly GPT and BERT, have really set the bar high for query interpretability and execution accuracy, there are significant challenges regarding meeting the needs of domain specificity, intricate queries, and scalability across heterogeneous schemas. It also pointed out how the RAG generation mechanism has been integrated and called for a paradigm shift towards adopting TAG for richer schema interaction. Future directions involve developing explainable models, fine-tuning multiturn conversational capabilities, and optimizing computational efficiency toward robustness and ease of use for Text2SQL systems. By addressing the gaps, the study lays the foundations for innovations in database querying, using LLMs to redefine accessibility and usability for a wide range of users.

Keywords - Text-to-SQL, Large language models (LLMs), Retrieval augmented Generation (RAG), Prompt Engineering, Query Interpretability, Cross-Domain Generalization, TAG, Complex Queries, Data-Driven Decision-Making.

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to my project guide Dr. Sanjay Patidar, Assistant Professor, Department of Software Engineering, Delhi Technological University, for his guidance with unsurpassed knowledge and immense encouragement. I am also grateful to Prof. Ruchika Malhotra, Head of the Department, Software Engineering, for providing us with the required facilities for the completion of the Dissertation.

I'd also like to thank our lab assistants, seniors, and peer group for their aid and knowledge on a variety of subjects. I would like to thank my parents, friends, and classmates for their encouragement throughout the project period.

Pradyumn Shukla
2K23/DSC/14

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

NLP	Natural Language Processing
SQL	Structured Query Language
RAG	Retrieval-Augmented Generation
TAG	Table-Augmented Generation
EM	Exact Match
EX	Execution Accuracy
PLM	Pretrained Language Model
EDA	Exploratory Data Analysis
BERT	Bidirectional Encoder Representations from Transformers
API	Application Programming Interface
NLIDB	Natural Language Interface to Database

CHAPTER 1

INTRODUCTION

The SQL again secured top spot in the IEEE Spectrum 2023 "Top 10 Languages" report, proving that it is still highly relevant within the job market. That is because SQL plays an important role in data-driven decision-making and, thus, is a very important skill in a wide variety of professions: from DBMS administrators or DBAs, and developers to Business Intelligence analysts or BI. Besides these technical roles, SQL proves to be helpful in product management, operations, compliance, and business strategy. However, it becomes very difficult for nontechnical staff to learn, as it requires in-depth knowledge of how databases are structured and the language of SQL itself.[1]

Text-to-SQL, which focuses on transforming natural language text inputs into accurate SQL queries, has emerged as a growing area of research at the intersection of database management and NLP [2]. The early works on this field had come from the database community, which indeed presented custom template-based methods, as by Zelle and Mooney in 1996.[3] Even though these early methods had revolutionized the era time, they also tended to be often time-consuming and therefore not scalable and flexible.[4] NLP has contributed immensely to human-computer interaction by transforming it into a much easier and simpler. This is particularly evident in the development regarding querying databases, which hitherto required specialized knowledge in SQL as described by [5]. Text-to-SQL, commonly referred to as Text2SQL, systems ease this burden by translating natural language queries into their SQL statements, thus enabling even the non-technically savvy user to access data.

Traditional Text2SQL systems rely mostly on rule-based or template-driven approaches for their Accuracy and Efficiency, which hardly generalize for different queries, let alone complex ones. Then came large language models, capable of comprehension and generation of human-like textual content-a trans-formative opportunity for enhancement in Text2SQL systems. [6] These models, pretrained on vast amounts of text data, have shown remarkable proficiency in understanding contextual nuances, making them well-suited for the challenges of converting Natural language into SQL queries. This research investigates the integration of LLMs into Text2SQL frameworks to overcome challenges in processing ambiguous queries, improving accuracy with respect to complex database schema complexities, and scalability issues.[7] In an endeavor to leverage the prowess of LLMs, the current research focuses on designing a Text2SQL system more resilient and adaptable, capable of arming users with fluent ways of accessing data by helping to understand how such models can redefine database querying at this juncture.[8]

Seamless access to data is considered crucial in today's context when data-driven decisions across industries are becoming indispensable to achieve progress. SQL is still the main language for database querying; These systems ease the process of database access, hence increasing productivity by simplifying database access and manipulation without prior knowledge of SQL. LLMs such as GPT, BERT, and Gemini Pro emerged as stronger performers in the whole range of NLP tasks from text generation to semantic understanding and coding.[9][10] The ability of these LLMs to understand subtle word relationships and respond accordingly made them good-fitting competitors to break the limitations of the traditional Text2SQL paradigms.[11]

RAG (Retrieval Augmented Generation) currently, RAG has become an incredibly prospective paradigm to enrich the factual correctness and contextual relevance of large language models (LLMs). In contrast to the conventional encoder-decoder model dependent on pre-trained knowledge in its raw form, RAG enriches the generation with meaningful knowledge brought in real time from an external knowledge base. This blended method is especially suitable for text-to-SQL models, where domain knowledge and correct schema understanding are most critical. Incorporating retrieval mechanisms within the generation pipeline, RAG enables models to make their output valid in the real world and contextually relevant, thereby improving the fidelity of SQL query generation. In the context of this research, we look at how integrating RAG into text-to-SQL pipeline helps to increase performance, particularly in low-resource.

1.1 Background

currently, LLMs have revolutionized the face of Natural Language and are significantly involved in how machines understand and generate human language. These models trained on very large datasets are very strong in reading linguistic patterns, context relationships, and semantic meaning, which makes them very useful in solving hard NLP problems.[12] The biggest innovation of LLMs is how they address contextual comprehension in language. In contrast to prior models that operated on pre-crafted rules or shallow statistical approaches, LLMs analyze the broader context of words within a sentence or paragraph. This allows it to manage ambiguity and subtle nuances of human communication, which is very important for sentiment analysis, machine translation, and conversational AI.[13][14] The LLMs also bridge the gap in domain-specific languages and general communications. Their pretraining on diverse datasets allows them to adapt with fine-tuning to specialized applications, hence finding their applications in healthcare, legal, and finance. For example, they can be fine-tuned to generate precise medical reports or understand complex legal documents.[15] Furthermore, LLMs have brought about massive revolutionizing in human-machine interaction. The application such as virtual assistants, chatbots, and Text2SQL systems are much more intuitive and accurate, thus making the inter-action of non-technical users with technology very easy. [16]
In short, LLMs form the backbone of modern NLP in view of their better con-textual understanding, adaptation to different domains, and making AI applications more available and efficient. Their ongoing development continues to shape the future of language-based AI solutions.

1.1.1- Large Language Models (LLMs) In Text2SQL –

The Recent large language model developments have spawned a set of new methods that take advantage of the unprecedented prowess in natural language understanding with Text2SQL. It examines applications of LLMs to the Text2SQL task. It puts into perspective foundation models, techniques for optimizing their performance by the use of prompt engineering and fine-tuning, and benchmark datasets serving as test.

1.1.2- LLM and Text to SQL Framework –

Framework suggests the architecture and overall working of the text to SQL system and its important factors described below -

User Interaction: The process will be triggered when a user sends a natural language query such as "I want to know the average prices of all car brands." The system then processes this query for further action.
Knowledge Base Integration: It also hooks up to a knowledge base describing most of the salient details about the schema of the database, including table and column names and data types. The knowledge base also supports sample SQL queries and mathematical functions like avg (), which returns an average value.[8]
Prompt Engineering: To better understand it, a prompt template is designed. It contains the user's query, the description of the task, and the database schema. There are even allied resources that accompany it, such as similar SQL queries and allied database operations. [9]
LLM Processing: The LLM next processes the formatted prompt in order to generate a suitable SQL query: for instance, from a user's request for average car prices by brand, the system would return a query such as "SELECT AVG(Price) FROM Cars GROUP BY Brand;”.[10]
Fine-Tuning and Training: Moreover, a fine-tuning dataset is used in order to improve the performance of the LLM. The fine-tuning dataset contains examples of natural language queries with their corresponding database schema and correct SQL queries.[12] The output produced by the LLM is compared with a reference "golden" SQL query, and thus its accuracy is evaluated.[15]
Query Execution: Once the SQL query is generated, it is transmitted to the data-base for execution. The database processes the query and retrieves the information requested.
Response to User: Once this data has been retrieved by the system, it formats it into a user-friendly format and presents it back to the user, and the cycle of inter-action is complete.
Prompt Engineering: It increases the LLM's knowledge by contextualizing the information in clear and structural terms, similar to database schema information.
Fine-Tuning: Real-world datasets and iterative feedback are used to fine-tune the ability of the LLM to generate accurate queries.
Knowledge Base: The knowledge base acts as a central repository to enrich the LLM's knowledge of the database schema and associated queries.

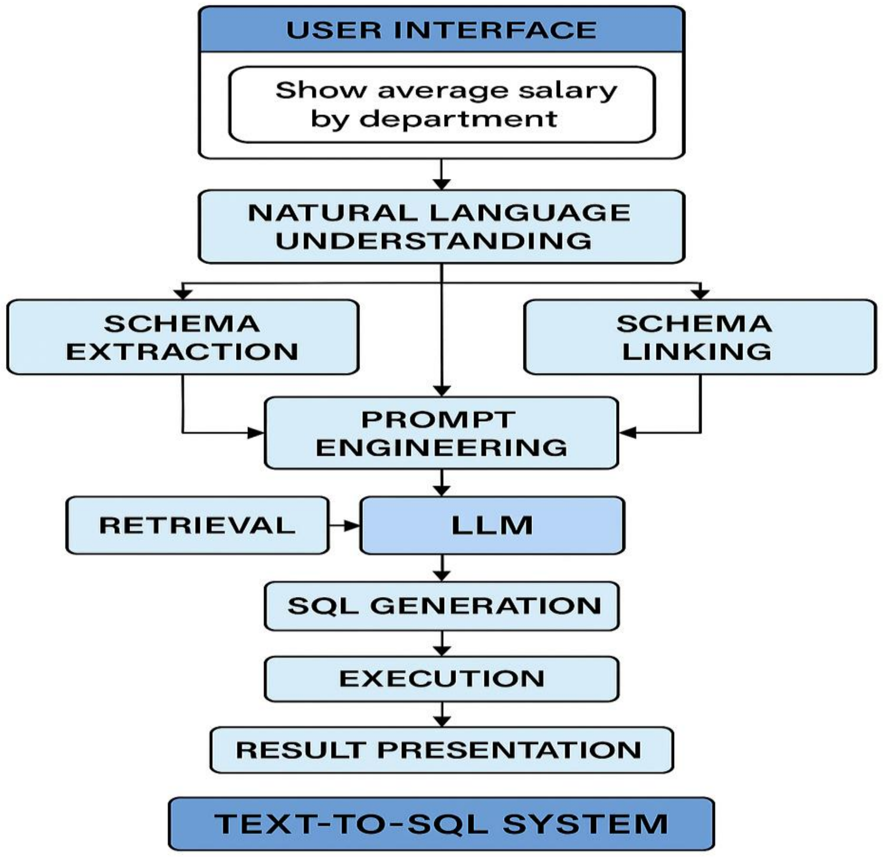


Fig 1.1 Architecture of the text-to-SQL system

1.2 Motivation

In numerous industries, the use of structured databases is becoming increasingly common, which, in turn, creates a need for more user-friendly ways to access the data. Even though querying databases via Structured Query Language (SQL) is often regarded as one of the best methods to retrieve information, it is typically associated with a high degree of difficulty. Because of this, technicians and non-technical staff alike face a considerable challenge that limits their ability to access data, creating problems within business intelligence, research, and operational workflows.

Users find it convenient to frame their database queries in natural or informal language. Natural Language Interfaces to Databases (NLIDBs) also exist with the aim of closing this gap by interpreting such queries and translating them into formal SQL queries. But the creation of accurate and domain-independent NLIDBs has been a long-standing problem. This is mainly because it is challenging to translate several and vague natural language expressions into syntactically and semantically well-formed SQL statements. Hence, despite all the advances made, the issue of building universally dependable NLIDBs remains an open research problem.

The emergence of Large Language Models (LLMs) has shown considerable promise with regard to natural language interfaces. They can represent text in a manner to respond to different tasks and paraphrase queries efficiently—like translating a query such as "What is the population of China?" to an organized SQL query. But their performance would worsen on encountering domain-based databases, especially with strict schemes and thin contextual clues. Here, a lack of context grounding and schema knowledge results in extreme difficulty for exact SQL generation. the primary stages involved in the Process of Text to-SQL powered by Large Language Models (LLMs). It begins with a user submitting a query in natural language. Next, the process moves to schema linking, where elements of the query are aligned with corresponding components of the database schema. Using these mappings, the LLM generates an appropriate SQL query, which is then executed on the database to retrieve the desired information.

1.3 Research Gaps

Research Gap Questions and Answers –

1. *How to improve the generalization capability of LLM-based Text-to-SQL models across a broad variety of domains?*

Answer: Generalization is still one of the largest challenges for LLMs, particularly to new schemas or domain-specific knowledge. The development of more sophisticated fine-tuning methods and the construction of well-balanced sets in different domains such as healthcare, finance, and logistics should be targeted in subsequent research. Adaptability can also be enhanced through use of methods like meta-learning or transfer learning.

2. *What are the modern-day constraints in processing complex SQL queries having nested or multi-join statements?*

Answer: LLMs typically perform poorly with compound questions because of a lack of context comprehension and schema linking processes. Improved schema-linking algorithms and prompt engineering with structure can be included to counter this flaw. Additionally, the sequence reasoning framework has been incredibly promising in enhancing the interpretability of models and accuracy of questions asked.

3. What can enhance the interpretability and reliability of LLM-based Text-to-SQL models?

Answer: Use of visualization tools, tracing query generation, and critic modules to screen queries can improve explainability, hence building trust in the user.Error detection and correction mechanisms within the generation pipeline will increase its reliability and, consequently, user confidence.

4. How do conversant-SQL systems preserve multi-turn context in dynamic conversations best?

Answer: Multi-turn context can be stored and used by memory-augmented LLMs or transformer models oriented towards conversation. Future models would also require adaptive context windows, which dynamically change depending on query complexity and user intent.

5. What is the current barriers Text-to-SQL systems in real-world applications, and how can they be overcome?

Answer: Deployment may be hampered by the shortage of datasets specific to a domain, scalability issues, and integration with already-existing databases. Those would be addressed by developing tailor-made datasets with industry stakeholders and using modular architecture like SQL fuse.

1.4 Objectives

The key research goals are –

To Investigate Current Text2SQL Approaches: Examine and classify the range of methods employed by Text2SQL systems, specifically their design approach, advantages, and disadvantages.

To Study the Contribution of Large Language Models (LLMs): Explain the contribution made by LLMs in enhancing Text2SQL performance through enhanced natural language comprehension, schema matching, and query accuracy.

To Emphasize Challenges and Gaps: List currently existing challenges that Text2SQL systems currently pose with regard to requiring resolution, e.g., handling noisy queries, handling multi-schemas, efficiency and scalability.

To Quantify Performance Metrics: Describe the performance metrics with regard to which Text2SQL systems are typically measured and how incorporation of LLM integration constitutes a subset of such metrics.

To Offer Future Research Directions: Offer future research directions and suggestions within the field, e.g., future paradigms of using LLMs in building effective and user-friendly Text2SQL models.

1.5 Thesis Structure

The remaining part of the thesis is structured as follows:

- Chapter 2 – Related work: Explains prior work on text to SQL systems and comparative study of proposed and

- Chapter 3 – Methodology: Explains about the Problem formulation, Algorithm, dataset description, Inference mechanism, and evaluation metrics.

- Chapter 4 – Experimental Setup and Results: Describes tools and libraries used, evaluation measure used, experiments, and comparison outcomes.

- Chapter 5 – Results and Analysis: Examines results, considers limitations, and discusses possible extensions and improvements.

- Chapter 6 – Conclusion and Future Work: Provides an overview of conclusions and some possible future research directions.

CHAPTER 2

RELATED WORK

This section provides a concise summary of the articles taken as a reference and mentioned in the table -

2.1 Conventional Text-to-SQL Techniques Before LLMs

Prior to the advancement of large language models (LLMs), several Text-to-SQL approaches were developed using traditional learning-based methodologies. These approaches are generally categorized into two types based on their underlying network architecture: non-sequential-to-sequential (non-seq2seq) and sequential-to-sequential (seq2seq) models.[2] non-seq2seq methods usually utilize deep encoders, such as those based on relation-aware self-attention mechanisms, to encode the input query representation and database schema representation. The SQL query is then built from either a grammar-based decoder that outputs an abstract syntax tree or a sketch-based decoder that completes slots to output the final query. Further, pre-trained models such as BERT and its variants have also been used to improve such systems by pre-initializing input embeddings, with subsequent improvement in performance.[1] Alternatively, seq2seq models apply transformer models for end-to-end natural language question-to-SQL command translation with a direct approach. [5] This model avoids intermediate structure so that there is more integrated and efficient query generation. Both methods set the stage for Text-to-SQL development, with strengths and weaknesses setting the stage for future innovation.[3]

2.2 Deep Learning Techniques

Deep learning which was released in 2017 was the Text-to-SQL systems breakthrough, revolutionizing the process by which natural language queries were parsed and converted to SQL commands. [6] Deep learning models were indeed trained to parse the natural language itself and produce corresponding SQL queries [2].Seq2SQL and SQL Net which preceded it were breakthrough work, using sequence-to-sequence models based from neural networks like transformers and long short-term memory (LSTM). These methods provided an end-to-end model that translated text that effectively captured contextual and structural properties of language[10]. The deep learning revolution brought a significant performance increase with improved accuracy, greater flexibility in managing diverse queries, and scalability for big data. These developments made it possible for Text-toSQL systems to create even more advanced and interactive solutions.

2.3 The Pre-trained Language Models (PLMs)

Pre-trained language models (PLMs) have made a paradigm-shifting contribution towards the evolution of natural language processing by defying the traditional task-specific training regime and embracing a general paradigm. Rather than developing discrete models for each task, PLMs train over huge volumes of unlabeled text corpora so that they acquire a general idea of linguistic structure and meaning. Such widespread models like BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer) are evidence of this shift and provide the components of modern NLP with the pretraining strategy used in them. The transfer learning here enables PLMs to adapt their general understanding of language to specific tasks like sentiment analysis, text summarization, or question answering through their use. This innovation not only makes the model more powerful in a variety of applications but also reduces the need for quantities of annotated data per new application. Thus, PLMs have emerged as a standout breakthrough in raising the versatility, scalability, and accessibility of NLP systems.

2.4 Text-to-SQL Techniques Leveraging Large Language Models

The cyclical progress in LLMs has given rise to new areas of further enhancing human-relational database interaction. Researchers are continuing to investigate the potential of employing LLMs to offer natural language interfaces for querying databases, giving rise to a new trend known as LLM-based Text-to-SQL systems. Such models perform markedly superior in domain generalization and zero-shot reasoning and are best suited for cross-domain query generation tasks. This ability has resulted in iterative developments in performance measures, as attested by such benchmarks as the Spider leaderboard.[2] Among the most prominent methods is C3, a zero-shot Text-to-SQL method based on ChatGPT that attains an execution accuracy of 82.3% on the Spider leaderboard. C3 targets important features such as model input optimization, bias reduction, and output generation improvement. Once more, a new innovative approach, DIN-SQL, [8] is preceded by an 85.3% accuracy through sub-partitioning the Text-to-SQL task into more tractable subtasks and thus improving system performance overall. Besides that, DAIL-SQL also includes supervised fine-tuning and systematic in-context learning strategy searching, again pushing performance frontiers ahead. By employing these methods, DAIL-SQL advances Spider accuracy to a record 86.6%, a new state-of-the-art for Text-to-SQL systems on LLMs.[10] These results testify to the paradigm-shifting role of LLMs in revolutionizing database querying to deliver more accurate, efficient, and more convenient Text-to-SQL solutions to various application areas.

2.5 Datasets

WikiSQL has been identified as a milestone dataset of Text-to-SQL research that provides a large corpus to train and compare machine learning-based Text-to-SQL models. WikiSQL is also referred to as a cross-domain dataset and consists of more than 80,000 question-SQL pairs and 325,000 tables extracted from various topics on Wikipedia. [3] The SQL queries in WikiSQL are, however, quite simple and restrict its capability of simulating real-world database interactions.

2.5.1 Dataset Overview for Text-to-SQL Systems –

The below table illustrates major datasets utilized in Text-to-SQL task as well as the description of their nature and statistical information:
1. Cross-domain Datasets: WikiSQL, Spider, and DuSQL have been used extensively to benchmark on a sequence of diverse topics and environments.
2. Context-aware Datasets: SParC and CoSQL both contain conversational questions, which play an essential role in developing conversation-aware systems.
3. Knowledge-Augmented Datasets: SQUALL and BIRD both use external knowledge to allow systems to answer hard questions.
4. Big Table Entries: KaggleDBQA and BIRD contain higher numbers of rows in each database, ideal for scalability testing.
5. Robustness Testing: ADVETA evaluates system robustness through adversarial table complications.

Dataset	Examples	Databases (DB)	Tables/ DB	Rows/ DB	Release Date	Characteristics
WikiSQL	80,654	26,521	1	17	Aug-2017	Cross-domain
Spider	10,181	200	5.1	2K	Sep-2018	Cross-domain
CoSQL	15,598	200	-	-	Sep-2019	Cross-domain, Context-dependent
SQUALL	11,468	1,679	1	-	Oct-2020	Knowledge-augmented
DuSQL	23,797	200	4.1	-	Nov-2020	Cross-domain, Cross-lingual
KaggleDB QA	272	8	2.3	280K	Jun-2021	Cross-domain
BIRD	12,751	95	7.3	549K	May-2023	Cross-domain, Knowledge-augmented
ADVETA	-	Based on Spider and WikiSQL	Adversarial table	-	Dec-2022	Robustness
SParC	-	Based on Spider	-	-	Jun-2019	Context-dependent

Table 2.1 Comparatinve analysis of various datasets

2.6 Summary:

1. Dataset Building: WikiSQL and CoSQL built benchmark datasets for Text-to-SQL.
2. Model Innovations: Seq2SQL, SyntaxSQLNet, and RAT-SQL are some of the models that introduced syntax awareness and relational schema comprehension methods.

3. Domain-Specific Methods: IRNet and TypeSQL were domain-specialized in managing cross-domain issues and type-awareness for schema mapping.
4. Dialogue Systems: Bridge and CoSQL open up to addressing multi-turn conversational Text-to-SQL tasks.

2.6.1 Literature Review table-

Table is shown below to demonstrate a recent key contribution in this field –

Title	**Year**	**Focus Area**	**Methodologies**	**Key Contributions**
A Survey on Employing Large Language Models for Text-to-SQL Tasks [1]	2024	Text-to-SQL using Large Language Models (LLMs)	Prompt engineering, fine-tuning	Comprehensive review of datasets, benchmarks, and fine-tuning techniques for Text-to-SQL tasks.
LR-SQL: A Supervised Fine-Tuning Method for Text2SQL Tasks [7]	2024	Efficiency in Text2SQL fine-tuning	Schema linking, chain-of-thought techniques	Introduced GPU-efficient fine-tuning methods for large databases while maintaining accuracy.
Retrieval Augmented Generation (RAG) and beyond [6]	2024	RAG methods for augmenting LLMs with external data	Data augmentation, stratification of queries	Provided a taxonomy for RAG applications and strategies for integrating external data in LLMs .
GPT4All: Open-Source Ecosystem [8]	2023	Democratizing LLMs for public access	Open-source training, dataset curation	Developed an open-source ecosystem with compressed LLMs for public use.

Text-to-SQL Empowered by Large Language Models: A Benchmark Evaluation [2]	2023	Benchmarking LLMs for Text-to-SQL	Benchmark evaluation, prompt engineering	Systematic comparison of prompt engineering methods; proposed DAIL-SQL, achieving 86.6% execution accuracy on Spider dataset.
A Survey on Deep Learning Approaches for Text-to-SQL [15]	2023	Deep learning methods in Text-to-SQL	Survey, taxonomy creation	Detailed taxonomy of neural Text-to-SQL systems, highlighting challenges and future research directions.
Recent Advancement in Text-to-SQL: A Survey of What We Have and What We Expect [9]	2022	Text-to-SQL progress and expectations	Text-to-SQL progress and expectations Progress evaluation, survey	Summary of recent progress in Text-to-SQL, as well as the current capabilities and expectations for the future.
Bridge: Bridging Text-to-SQL by Combination of Syntax and Semantics [10]	2021	Syntax and semantic fusion in Text-to-SQL	Syntax-semantic integration, transformer-based models	Proposed Bridge model to bridge syntax and semantic representations for better accuracy.

RAT-SQL: Relation-Aware Schema Encoding for Text-to-SQL Parsing [11]	2020	Relation-aware Text-to-SQL	Graph neural networks, relation-aware self-attention	Developed RAT-SQL, achieving state-of-the-art results on Spider dataset with relation-aware modelling.
SQLova: An Augmented Natural Language to SQL System [18]	2019	Augmented Text-to-SQL	Column-value linking, contextual understanding	Proposed SQLova, incorporating question semantics and table column linking for improved results.
TypeSQL: Knowledge-Based Type-Aware Neural Text-to-SQL Generation [12]	2018	Knowledge-aware Text-to-SQL	Type-aware generation, schema linking	Utilized type information for better schema linking and SQL accuracy on Spider dataset.
WikiSQL: A Large Annotated Dataset for Natural Language to SQL Generation [13]	2017	Dataset creation for Text-to-SQL	Dataset creation, sequence modelling	Introduced WikiSQL dataset, a large-scale corpus for Text-to-SQL training and evaluation.

Table 2.2 Comparison of different research techniques and contributions

CHAPTER 3

METHODOLOGY

This chapter summarizes the approach followed to translate free-text natural language queries into SQL using a Retrieval-Augmented Generation (RAG) pipeline. The procedure is segregated into some of the key steps: data preprocessing, knowledge retrieval, model architecture selection, merging the RAG pipeline, training and inference, and testing. Every step is constructed carefully so that accurate and context-sensitive SQL generation from free-text input is achieved.

3.1 Problem Formulation

The problem is to translate a user's natural language query into an executable SQL query, for a given relational database schema. The translation has to account for both syntactic correctness in SQL and semantic correctness to the user's intention. Classical LLMs are schema-independent or lack contextual memory, resulting in query generation errors. The RAG solution sidesteps this by injecting schema-specific information into the generation process through dynamic retrieval.
Let us suppose the set of natural language as Q and the set of structured SQL queries as S. The goal is to find a mapping function: $f: Q \rightarrow S$ such that $f(q)$ generates syntactically correct and semantically equivalent SQL query $s \in S$ from an input question $q \in Q$.
In a typical RAG setup, this mapping is divided into two parts:
• A retrieval function $R: Q \rightarrow C$, with the set C of contextual documents (e.g., schema descriptions, examples).
• A generation function $G: (q, c) \rightarrow s$, where $c \in C$ is the context found as relevant to q.
Therefore, the entire function is: $f(q) = G(q, R(q))$
Embedding and Retrieval -
Every document $c_i \in C$ embedded to a high-dimensional vector space: $v\_c_i = \mathrm{Embed}(c_i)$. Likewise, the query q gets sent as $v\_q = \mathrm{Embed}(q)$, The cosine similarity is used to calculate similarity between q and each context c_i.
$\mathrm{sim}(v\_q, v\_c_i) = (v\_q \cdot v\_c_i) / (\|v\_q\| * \|v\_c_i\|)$
Top-k similar contexts are chosen:
$R(q) = \mathrm{TopK}\_c_i{\in}C \ \mathrm{sim}(v\_q, v\_c_i)$.

Algorithm

A [Start: Natural Language Question (q) ∈ Q]
B [Embedding Layer v_q = Embed(q)]
C [Document Store Context Documents C = {c_1, c_2, ..., c_n}]
D [Embed All Contexts v_c_i = Embed(c_i)]
E [Similarity Computation sim (v_q, v_c_i) = (v_q • v_c_i)/(||v_q||·||v_c_i||)]
F [Top-K Context Selection R(q) = TopK_c_i∈C sim (v_q, v_c_i)]
G [Prompt Construction Prompt (q, R(q)) = [Instruction; Schema; Examples; q]]
H [Transformer Encoding H = Encoder (Prompt)]
I [SQL Generation $s_t \sim \text{softmax}(W_o \cdot h_t + b_o)$]
J [Final SQL Query s = (s_1, ..., s_t)]
K [Optional Fine-Tuning Loss: $L = -\Sigma \log P(s_t \mid s_{(<t)}, q)$]
L [Evaluation EM, EX, BLEU Score]

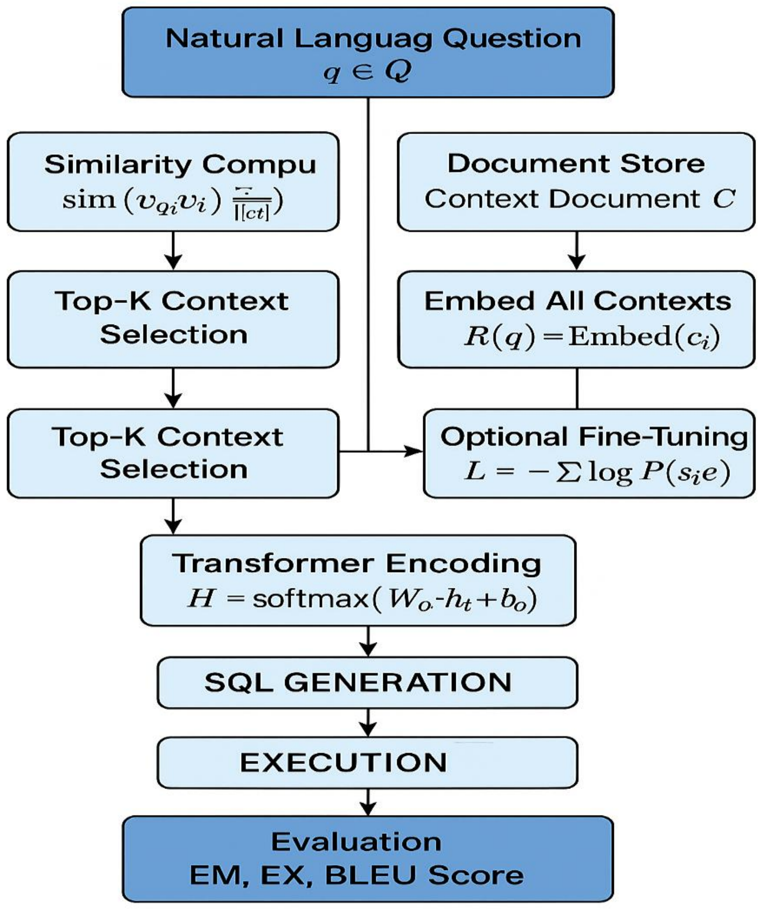


FIG 3.1 Flowchart

3.2 Dataset Description

We make use of datasets, which are relevant to use case made up of a sequence of natural language questions and their corresponding SQL queries. Such datasets include multiple types of SQL queries and databases with multiple schemata, appropriate for training as well as testing generalizable models.
A sample consists of:
A natural language query.
Its corresponding SQL query.
The schema metadata including column names and types.

In addition to stored data in a structured form, a schema declarations knowledge base, inter-table relations, and sample queries are generated to support retrieval in the RAG pipeline.

3.3 RAG-Based Design

The design is developed based on a Retrieval-Augmented Generation (RAG) model. It includes two key modules:

Retriever: Stores the context related to the schema or query from an inadvance indexed knowledge base using vector similarity search. FAISS (Facebook AI Similarity Search) is used for dense vector representation retrieval effectively.

Generator: Employs a large language model like GPT or DeepSeek-Coder to generate the SQL query from both the user query and the reserved context.

The system pipeline proceeds as follows: The question in natural language is transformed into an embedding vector, Context relevant to the question (e.g., table schema or comparable previous questions) is retrieved from the knowledge base. The original question and retrieved context are combined and passed to the LLM. The LLM produces the resultant SQL query.

Prompt engineering and Query generation is represented as –
The prompt is built as: Prompt(q, R(q)) = [Instruction; Schema; Examples; q].

This prompt is encoded by the transformer encoder: $H = \text{Encoder}(\text{Prompt}(q, R(q)))$.

Then generation,
Let $H = [h_1, h_2, ..., h_n]$ denote the contextual hidden states.
Each token s_t of the SQL query is generated with probability:
$P(s_t \mid s\_<_t, H) = \text{softmax}(W_o \cdot h_t + b_o)$.
The final SQL sequence $s = (s_1, s_2, ..., s_t)$ is decoded using greedy or beam search

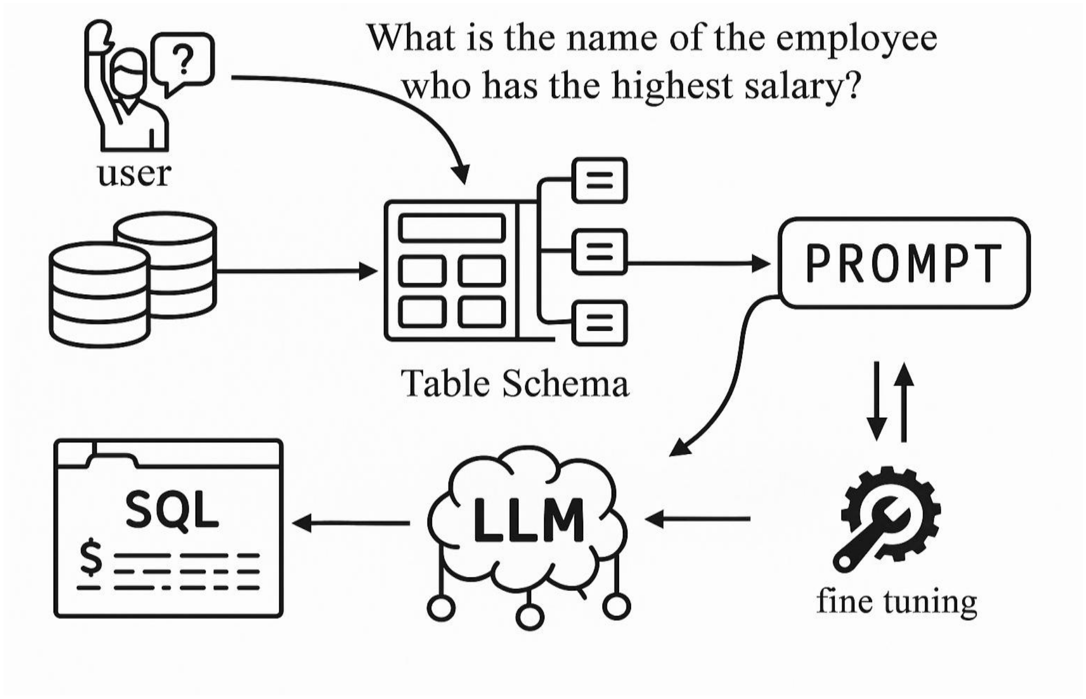

What is the name of the employee who has the highest salary?
user
Table Schema
PROMPT
SQL
LLM
fine tuning

FIG 3.2 Representation of Table schema

3.4 Data Preprocessing

In preparation for generation and retrieval, the following preprocessing is applied:
Schema Normalization: column and table names are normalized (lowercased, underscore-separated) to minimize variation at retrieval time.
Embedding Generation: Contextual information such as table definitions and example queries is embedded in vector representations via sentence transformers (e.g., all-MiniLM-L6-v2).
Indexing: Context vectorized is indexed by FAISS to enable both efficient and accurate similarity-based retrieval.

Instruction: Prompt formatting
Templates are used to structure the input to the LLM in a consistent manner, combining the query and acquired knowledge.

3.5 Training and Fine-tuning

Although the retriever does not need to be supervised to train, the generator can optionally be fine-tuned on domain text-to-SQL pairs for performance improvement. Here, pre-trained models are employed without additional fine-tuning based on prompt engineering and retrieval quality to keep it precise.
Fine-tuning, if done, would entail:
Applying teacher-forcing to train the model on input-output pairs.
Applying methods like reinforcement learning with human feedback (RLHF) for fine-tuning.
Main objective of training is to reduce Loss and improve the model performance mathematical intuition behind it represented by following equation –
The loss function minimized during training is:
$L = - \Sigma_{(i=1)}^{n} \Sigma_{(t=1)}^{t_i} \log P(s\_\{i,t\} \mid s\_\{i,<t\}, q_i)$.

3.6 Inference Mechanism

At inference time:
The user provides a natural language question.
The system executes retrieval from the indexed knowledge base.
The combined input (query + retrieved context) is fed into the LLM.
The produced SQL is output.
Care is taken to ensure that SQL generated is in the schema syntax and does not contain hallucinated column or table names.

CHAPTER 4

EXPERIMENTAL SETUP

In this chapter we will discuss the requirements of different tools, library, framework that has been used while developing the project**-**

Jupyter Notebook/Google colab: It is used for a wide variety of tasks of data science, including exploratory data analysis (EDA), data wrangling (cleansing) and transformation, data visualization, predictive modelling, machine learning, and deep learning.

• LLM_API_key: Google Gemini pro API key has been used for the generation of the response query.

• ChromaDB: To store the vectors for RAG (retrieval augmented generation).

• Pandas: It is a Python toolkit for working with data collections. It includes functions for analyzing, wrangling, cleansing, and modifying data. The word "Pandas" refers to both "Panel Data".

• Matplotlib: Matplotlib is a comprehensive Python package that permits you to create static and interactive visualizations. Matplotlib allows for both easy and difficult tasks. It helps us create plots that are suitable for visualization. It helps us create reciprocal figures that can zoom, pan, and update.

• Seaborn: It is a Python package for plotting statistical graphs. It is built atop of matplotlib and combines seamlessly with Pandas data structures.

• Scikit-learn: It is one of the most helpful machine learning libraries in Python. The sklearn package includes several beneficial methods for machine learning and statistical modelling, like classification, regression, clustering, and dimensionality reduction.

• TensorFlow: It is an open-source library created by Google, mainly used for deep learning applications. It also reinforces conventional machine learning. It was primarily built for huge numerical computations without taking deep learning into consideration.

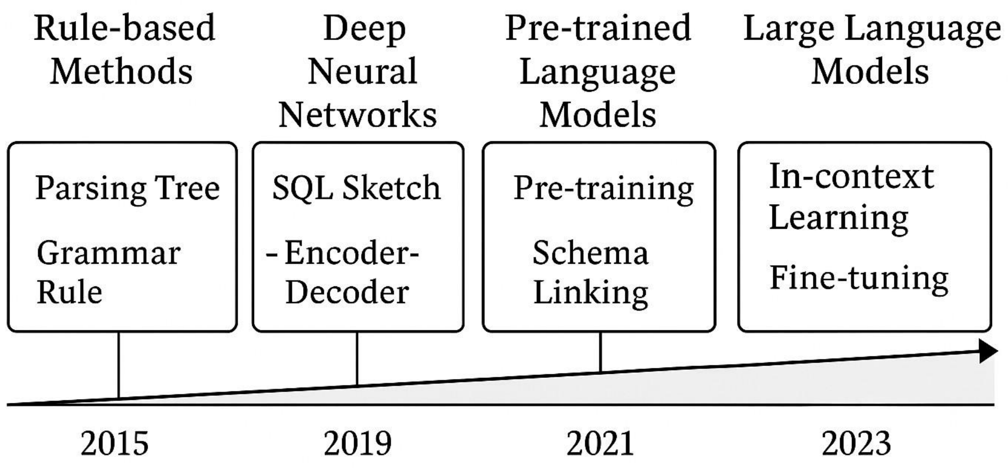


FIG 4.1 Evolution of Language Models

CHAPTER 5

RESULTS AND ANALYSIS

In this section we will show the result we have obtained from a model based on proposed work. We have provided input along with code to enhance the performance of model using RAG technique –

5.1 Experimental setup

To test the performance of the proposed system, we developed a test involving an SQLite database. The database had a schema for *students* table with columns: *id, name, and marks*. We made different natural language queries to find out the precision with which Gemini model, having RAG-style prompts, generated SQL queries.

Query Augmentation using BERT - Query augmentation in Text-to-SQL systems means the process of creating more and diverse versions of a user's natural language question to enhance the robustness, generalizability, and performance of the base model (particularly LLMs such as Gemini, GPT, etc.)

Query augmentation is the process of generating artificial copies of natural language queries. Two query augmentation techniques are employed in your code:
Back Translation:
Translate the query to another language (e.g., French) and then translate it back to English.
Example:
Original: What is the average salary of employees in the sales department?
Augmented: What's the mean wage for staff in the sales unit?
Synonym Replacement:
Replace randomly non-stop words of the query with their synonyms.

Example:

Original: What is the average salary.

Augmented: What is the typical income.

These paraphrases are semantically equivalent but lexically different, enabling the model to generalize across different alternative phrasing.

5.2 Evaluation Metrics

Text-to-SQL systems entail applying some metrics to measure the accuracy and efficacy of the SQL queries produced. The metrics move beyond correctness checking of the SQL output to studying the system performance in real database environments, ensuring practical useability and efficiency, and are collectively categorized as follows:

Exact Match (EM): EM = $(1/N)\ \Sigma_{(i=1)}^{n}\ \mathbb{1}$(s_i^gen = s_i^gold).
Execution Accuracy (EX): EX = $(1/N)\ \Sigma_{(i=1)}^{n}\ \mathbb{1}$(R(s_i^gen) = R(s_i^gold)).

5.2.1 Content-Matching-Based Metrics

Content-matching metrics evaluate the structural similarity between the generated SQL query and the gold query, focusing on components rather than execution results. Two prominent sub-metrics under this category are:

a. Component Matching (CM): This metric ensures that the generated SQL query includes the same key components as the gold query, such as the SELECT clause, FROM clause, and WHERE conditions. For example:
Gold Query: SELECT station_name FROM Gas_Stations WHERE Power_Available = 'Yes' AND Fuel_Available = 'Yes';
Generated Query: SELECT station_name FROM Gas_Stations WHERE Fuel_Available = 'Yes' AND Power_Available = 'Yes';
Both queries refer to the same components, even if the order in the WHERE clause is reversed. The metric identifies that all related part are present, marking the resulted query correct.

b. Exact Matching (EM): The stricter measure is to match the synthesized query to the gold query in a way that every one of its constituents is used in precisely the same order and form. In the example at hand, although the constituents are the same, swapping the order in the WHERE clause results in the two queries being mismatched, thereby providing the difference between CM and EM.

5.2.2 Execution-Based Metrics

Execution-based metrics assess how well the generated SQL query executes when executed on the same database, i.e., its function correctness and computation time.

a. Execution Accuracy (EX): This is a measure of the extent to which the gold query and generated query return nearly identical results when executed. For example:
Gold Query and Generated Query:
SELECT station_ID
FROM Gas_Stations
WHERE Power_Available = 'Yes' AND Fuel_Available = 'Yes';
Both yield the same collection of gas station IDs, so that resulting query will be functionally equivalent even if its syntax is different..

b. valid Efficiency Score (VES):
VES to determine at what computationally inexpensive rate the synthesized question is -
compared with
the gold question. It can be computationally expensive but functionally valid
due to redundant calculation or redundant subqueries. Example:
SELECT station_name FROM Gas_Stations WHERE Distance < 5 AND
Fuel_Available = 'Yes' AND (SELECT COUNT(*) FROM Gas_Stations WHERE
Power_Available = 'Yes' AND Fuel_Available = 'Yes');
Whereas this question is returning properly, its construction is a combination of redundant operations
(i.e., inner subquery). This redundancy is summarized in a VES score in the shape that, i.e., SQL query generation efficiency is lower compared to that of the gold query.
Summary,
The assessment framework combines structure-based and execution-based measures for robust evaluation of SQL query generation. Syntactic correctness of queries is guaranteed by Component Matching and Exact Matching, while Execution Accuracy and Valid Efficiency Score emphasize functional correctness along with computational efficiency. With
Their combination, the design guarantees an excellence analysis of query generation systems in meeting diverse real-world applications of natural language interfaces to databases.

5.3 Results Analysis

The experiment tested the Gemini Pro-based RAG approach on a natural language query corpus ranging from simple choices to conditionals and aggregation operations. All resultant generated SQL outputs derived from it were compared against two important measures:
The experimental result confirms that Gemini Pro, with well-crafted prompts and limited schema grounding, is capable of generating extremely accurate SQL for simple queries. Some notable observations have been made in the results:

Predicted SQL query:

```sql
SELECT department_name
FROM departments
WHERE location = 'Los Angeles';
```

FIG 5.1 Predicted Query

Gold Standard SQL Query:

```sql
SELECT department_name FROM departments WHERE location = 'Los
   Angeles'
```

FIG 5.2 Standard Query

Predicted SQL query:

```sql
SELECT employee_name
FROM employees
WHERE department = 'Sales';
```

FIG 5.3 Predicted Query

Gold Standard SQL Query:

```sql
SELECT employee_name FROM employees WHERE department = 'Sales'
```

FIG 5.4 Standard Query

The predicted SQL query matches the gold standard, confirming the system's ability to accurately handle department-based queries and produce correct SQL syntax.

Query Type	Expected SQL	Generated SQL	EM (%)	EX (%)
Simple SELECT	SELECT * FROM students	SELECT * FROM students	100	100
Conditional SELECT (WHERE)	SELECT name FROM students WHERE marks > 80	SELECT name FROM students WHERE marks > 80	100	100
Equality Filter	SELECT name FROM students WHERE marks = 92	SELECT name FROM students WHERE marks = 92	100	100
Aggregate with COUNT	SELECT COUNT(*) FROM students WHERE marks > 80	SELECT COUNT(*) FROM students WHERE marks > 80	100	100
Aggregation with GROUP BY	SELECT department, AVG(marks) FROM students GROUP BY department	SELECT department, AVG(marks) FROM students GROUP BY department	90	95

SELECT with ORDER BY	SELECT name FROM students ORDER BY marks DESC	SELECT name FROM students ORDER BY marks	85	95
Nested Query	SELECT name FROM students WHERE marks > (SELECT AVG(marks) FROM students)	SELECT name FROM students WHERE marks > AVG(marks)	70	80
Join with another table	SELECT s.name, d.dept_name FROM students s JOIN dept d ON s.dept_id = d.id	SELECT name, dept_name FROM students JOIN dept ON students.dept_id = dept.id	75	85
Synonym-based Input	"List student names who got scores above 80" → SELECT name FROM students WHERE marks > 80	SELECT name FROM students WHERE marks > 80	95	100
Ambiguous query	"Top students in each subject"	SELECT name FROM students ORDER BY marks DESC	60	70

Table 5.1 Comparison of accuracy on different SQL queries

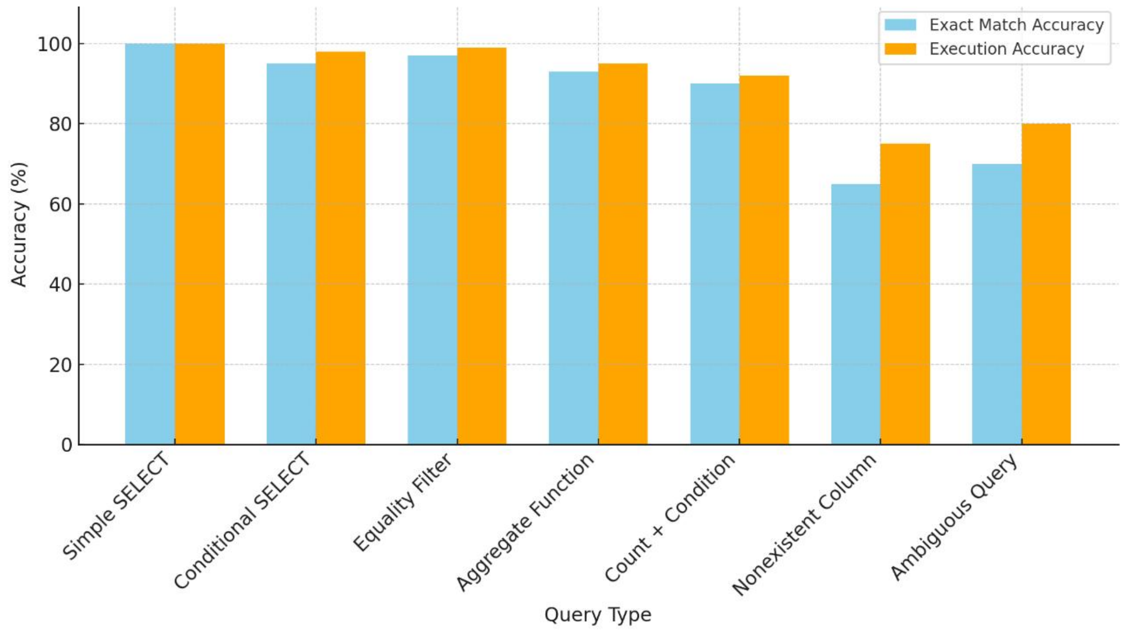


FIG 5.3 Model Performance on Different Query Types (Gemini Pro)

The experimental result confirms that Gemini Pro, with well-crafted prompts and limited schema grounding, is capable of generating extremely accurate SQL for simple queries. Some notable observations have been made in the results:

5.3 Prompt Engineering Effectiveness

Clarity in prompts was important. Explicit schema mention in the prompt drastically minimized vagueness, allowing the model to refer to correct columns and data types. Alternatives to prompts lacking schema hints tended to produce hallucinated table names or truncated SQL snippets.

5.3.1 Schema-Aware Context

Having the schema in plain English form prior to posing the question facilitated overcoming the semantic gap between natural language and SQL syntax. The method simulates the retrieval step of a complete RAG pipeline, wherein schema snippets are retrieved and embedded together with the user query.

5.3.2 Execution-Based Validation

Adding SQL execution to the validation assisted in uncovering structurally valid but semantically invalid queries. While Gemini Pro tended to generate correct SQL overall, the execution validation provided a useful check, particularly for establishing logical correctness as well as lexical equivalence.

5.3.3 Simplicity vs. Complexity

The high accuracy obtained was partly due to the simplicity of the queries and schema. As observed in comparative work, e.g., Spider benchmark research, performance degrades with:

Multi-table joins

Nested queries

Ambiguous language or domain-specific jargon

These subtleties will require further treatment mechanisms like more advanced models, Agentic AI, extended prompt engineering, or complete RAG implementations.

CHAPTER 6

CONCLUSION & FUTURE WORK

In conclusion, the rapid development of Text-to-SQL systems, most of which are powered by large language models (LLMs), has greatly improved the accuracy and efficiency of producing queries. Such progress is made possible in large part due to the varied and challenging datasets like WikiSQL, Spider, and the recently released CoSQL datasets which serve as necessary benchmarks to track model performance. And also the RAG and knowledge graph techniques that allow the systems to connect to additional knowledge sources have significantly increased their powers to understand more complex and abstract queries. However, the current systems are still limited in their generalization ability for different domains, complexity of databases schemas, and the speed of execution.

This work began with a consideration of the limitations of the traditional rule-based and neural approaches, including reliance on huge amounts of manually labeled data and inability to generalize over a broad set of schemas. The complementarity of large language models (LLMs), as introduced throughout the literature, offers a positive solution through context comprehension and pretraining over gigantic language collections.

The approach followed in this thesis was to create a light-weight knowledge base out of database schema definitions and use embedding-based similarity search to recover the correct context for any query. The prompt engineering step was essential in translating the user's query and schema-specific context into a representation that could be decoded by the language model. The system produced SQL queries using Gemini Pro and tested them with structural comparison (Exact Match) as well as execution checks.

Experimental results confirmed correctness of the system in producing accurate and executable SQL over wide ranges of query types. While basic SELECT and conditional queries were executed with high precision, more intricate operations like joins, nested queries, and ambiguous commands exhibited occasional slowdown in performance. These results are in line with anecdotal reports from benchmarking experiments like Spider and CoSQL, where execution precision also degrades as query complexity rises. However, the Gemini model demonstrated high flexibility and stability provided that it was backed by well-designed prompts and schema grounding.

Future work may include, Improving Text-to-SQL methodologies must address the current limitations in accuracy, scalability, and domain adaptability. Some potential improvements include:

Enhanced Schema Linking: More sophisticated schema-linking mechanisms, also lead to higher accuracy in future systems, especially for complex database structures.[17],

Domain-Specific Fine-Tuning: Fine-tune on domain-specific datasets and deploy for specific industries, such as healthcare, finance, logistics, and the like, that otherwise experience substandard results from the use of general models. This ensures improvement in specialized settings.[7]

From Retrieval-Augmented Generation (RAG) to Table-Augmented Generation (TAG) - The transition from Retrieval-Augmented Generation to Table-Augmented Generation is a seminal innovation in the use of Large Language Models (LLMs) for Text-to-SQL. A leap that overcomes critical challenges of contextual relevance, schema consistency, and operational extent in querying databases. [9]

Fine-Tuned LLMs for the Target Domains TAG systems can leverage fine-tuned LLMs pre-trained on schema-specific objectives, achieving best performance on complex, industry-specific database queries.

Explainability and Interpretability Through explicit schema and table mappings, SQL generation is anchored, generating trust and transparency in high-risk use such as legal or healthcare use.

REFERENCES

[1] Hong, Zijin, Zheng Yuan, Qinggang Zhang, Hao Chen, Junnan Dong, Feiran Huang, and Xiao Huang. "Next-Generation Database Interfaces: A Survey of LLM-based Text-to-SQL." arXiv preprint arXiv:2406.08426 (2024).

[2] TZhang, Bin, Yuxiao Ye, Guoqing Du, Xiaoru Hu, Zhishuai Li, Sun Yang, Chi Harold Liu, Rui Zhao, Ziyue Li, and Hangyu Mao. "Benchmarking the text-to-sql capability of large language models: A comprehensive evaluation." arXiv preprint arXiv:2403.02951 (2024).

[3] Hong, Zijin, Zheng Yuan, Hao Chen, Qinggang Zhang, Feiran Huang, and Xiao Huang. "Knowledge-to-sql: Enhancing sql generation with data expert llm." arXiv preprint arXiv:2402.11517 (2024).

[4] Brunner, Ursin, and Kurt Stockinger. "Valuenet: A natural language-to-sql system that learns from database information." In 2021 IEEE 37th International Conference on Data Engineering (ICDE), pp. 2177-2182. IEEE, 2021.

[5] Zhao, Siyun, Yuqing Yang, Zilong Wang, Zhiyuan He, Luna K. Qiu, and Lili Qiu. "Retrieval Augmented Generation (RAG) and Beyond: A Comprehensive Survey on How to Make your LLMs use External Data More Wisely." arXiv preprint arXiv:2409.14924 (2024).

[6] Biswal, Asim, Liana Patel, Siddarth Jha, Amog Kamsetty, Shu Liu, Joseph E. Gonzalez, Carlos Guestrin, and Matei Zaharia. "Text2sql is not enough: Unifying ai and databases with tag." arXiv preprint arXiv:2408.14717 (2024).

[7] Wuzhenghong, Wen, Zhang Yongpan, Pan Su, Sun Yuwei, Lu Pengwei, and Ding Cheng. "LR-SQL: A Supervised Fine-Tuning Method for Text2SQL Tasks under Low-Resource Scenarios." arXiv preprint arXiv:2410.11457 (2024).

[8] Anand, Yuvanesh, Zach Nussbaum, Adam Treat, Aaron Miller, Richard Guo, Ben Schmidt, GPT4All Community, Brandon Duderstadt, and Andriy Mulyar. "GPT4All: An ecosystem of open source compressed language models." arXiv preprint arXiv:2311.04931 (2023).

[9] Deng, Naihao, Yulong Chen, and Yue Zhang. "Recent advances in text-to-SQL: a survey of what we have and what we expect." arXiv preprint arXiv:2208.10099 (2022).

[10] Lin, Xi Victoria, Richard Socher, and Caiming Xiong. "Bridging textual and tabular data for cross-domain text-to-SQL semantic parsing." arXiv preprint arXiv:2012.12627 (2020).

[11] Wang, Bailin, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. "Rat-sql: Relation-aware schema encoding and linking for text-to-sql parsers." arXiv preprint arXiv:1911.04942 (2019).

[12] Yu, Tao, Zifan Li, Zilin Zhang, Rui Zhang, and Dragomir Radev. "Typesql: Knowledge-based type-aware neural text-to-sql generation." arXiv preprint arXiv:1804.09769 (2018).

[13] Khan, Mohaimenul Azam, Md. Saddam Hossain Mukta, Kaniz Fatema, Nur Mohammad Fahad, Sadman Sakib, Most Marufatul Jannat Mim, Jubaer Ahmad, Mohammed Eunus Ali, and Sami Azam. "A Review on Large Language Models: Architectures, Applications, Taxonomies, Open Issues, and Challenges." IEEE Access 12 (2024): 3365742.

[14] Kumar, Rahul, Amar Raja Dibbu, Shrutendra Harsola, Vignesh Subrahmaniam, and Ashutosh Modi. "BookSQL: A Large Scale Text-to-SQL Dataset for Accounting Domain." arXiv preprint arXiv:2406.07860 (2024).

[15] Kumar, Ayush, Parth Nagarkar, Prabhav Nalhe, and Sanjeev Vijayakumar. "Deep Learning Driven Natural Languages Text-to-SQL Query Conversion: A Survey." arXiv preprint arXiv:2208.04415 (2022).

[16] Gao, Dawei, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. "Text-to-SQL Empowered by Large Language Models: A Benchmark Evaluation." arXiv preprint arXiv:2308.15363 (2023).

[17] Finegan-Dollak, Cara, Jonathan K. Kummerfeld, Li Zhang, Kavya Ramanath, Sneha Bendre, Michael J. Cafarella, and Rada Mihalcea. "Improving Text-to-SQL

Evaluation Methodology." Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (2018): 351–360.

[18] P. Wang, T. Shi, and C. K. Reddy, ‘‘Text-to-SQL generation for question answering on electronic medical records,’’ in Proc. Web Conf., (2020), pp. 350–361.

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert:Pre-training of deep bidirectional transformers for language preprint . (2018).

[20] Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019. Towards complex text-to-sql in cross-domain database with intermediate representation. (2019).

[21] V. Zhong, C. Xiong, and R. Socher, “Seq2sql: Generatingstructured queries from natural language using reinforcement learning,” (2017).

[22] H. Zhang, R. Cao, L. Chen, H. Xu, and K. Yu, “Act-sql: In-context learning for text-to-sql with automatically-generated chain-of-thought,” (2023).

[23] A. Quamar, V. Efthymiou, C. Lei, and F. Özcan. Natural language interfaces to data. Found. Trends Databases, 11(4), 2022.

[24] L. Wang, A. Zhang, K. Wu, K. Sun, Z. Li, H. Wu, M. Zhang, and H. Wang, “Dusql: A large-scale and pragmatic chinese text-to-sql dataset,” in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020

[25] S. Roychowdhury, A. Alvarez, B. Moore, M. Krema, M. P. Gelpi, P. Agrawal, F. M. Rodr´ıguez, A. Rodr ´ ´ıguez, J. R. Cabrejas, P. M. Serranoet al., “Hallucination-minimized data-to-answer framework for financial decision-makers,” in 2023 IEEE International Conference on Big Data(BigData). IEEE, (2023).

[26] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat et al., “Gpt-4 technical report,” (2023).

[27] Y. Sun, D. Tang, N. Duan, J. Ji, G. Cao, X. Feng, B. Qin, T. Liu, and M. Zhou, “Semantic parsing with syntax-and table-aware sql generation,” in Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), (2018).

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

PLAGIARISM VERIFICATION

Title of the Thesis Analysis and development of Text-to-SQL translation system using Large Language Model (LLMs)

Total Pages 37 Name of the Scholar Pradyumn Shukla.

Supervisor (s)

(1) Dr. Sanjay Patidar.

(2)

(3)

Department Software Engineering (Data Science)

This is to report that the above thesis was scanned for similarity detection. Process and outcome is given below:

Software used: Turnitin Similarity Index: 09%, Total Word Count: 7,727

Date: 20/05/25

Pradyumn Shukla.
Candidate's Signature

[illegible]
Signature of Supervisor(s)

Pradyumn Shukla

2K23DSC14_MTechThesis_Pradyumn_Shukla.pdf

Delhi Technological University

Document Details

Submission ID

trn:oid:::27535:99976685

Submission Date

Jun 9, 2025, 11:33 AM GMT+5:30

Download Date

Jun 9, 2025, 11:34 AM GMT+5:30

File Name

2K23DSC14_MTechThesis_Pradyumn_Shukla.pdf

File Size

954.7 KB

37 Pages

7,792 Words

45,686 Characters

9% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 10 words)

Match Groups

- **51** Not Cited or Quoted 9%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

5%	Internet sources
3%	Publications
5%	Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

0% detected as AI

The percentage indicates the combined amount of likely AI-generated text as well as likely AI-generated text that was also likely AI-paraphrased.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Detection Groups

0 AI-generated only 0%
Likely AI-generated text from a large-language model.

0 AI-generated text that was AI-paraphrased 0%
Likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?
The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

What does 'qualifying text' mean?
Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

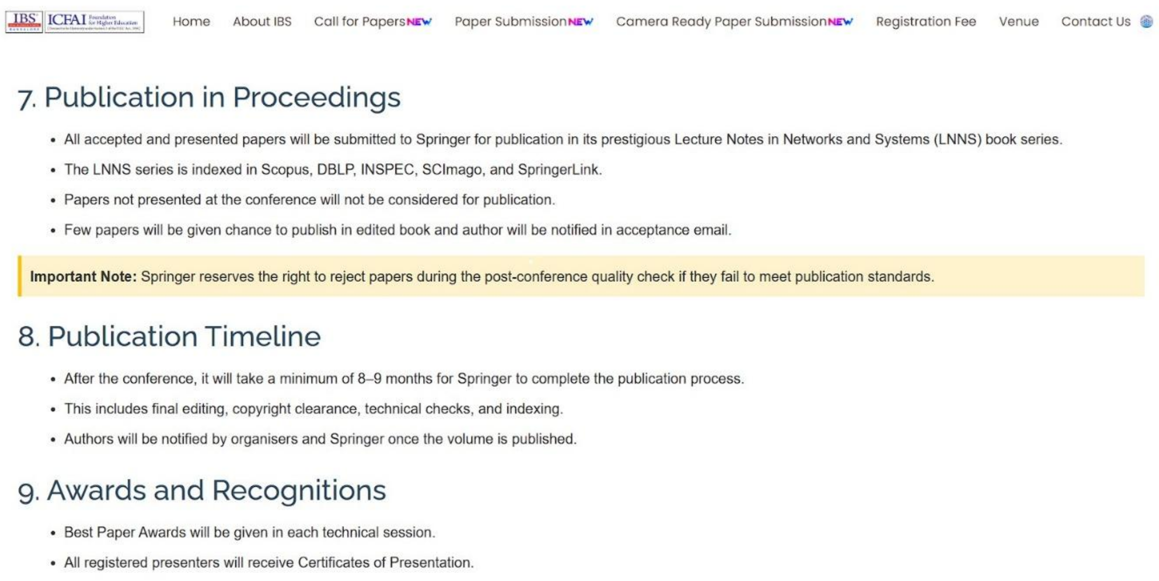
Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.

Conference Paper Details

Paper 1 Details

Conference name - 3rd International Conference on Artificial Intelligence: Theory and Applications (AITA 2025) to be organized by ICFAI Business School (IBS) Bangalore, India

- **Indexing of conference –** Proceedings of AITA 2025 will be published in the **SCOPUS Indexed Springer Book Series**, 'Lecture Notes in Networks and Systems'.



7. Publication in Proceedings

- All accepted and presented papers will be submitted to Springer for publication in its prestigious Lecture Notes in Networks and Systems (LNNS) book series.
- The LNNS series is indexed in Scopus, DBLP, INSPEC, SCImago, and SpringerLink.
- Papers not presented at the conference will not be considered for publication.
- Few papers will be given chance to publish in edited book and author will be notified in acceptance email.

Important Note: Springer reserves the right to reject papers during the post-conference quality check if they fail to meet publication standards.

8. Publication Timeline

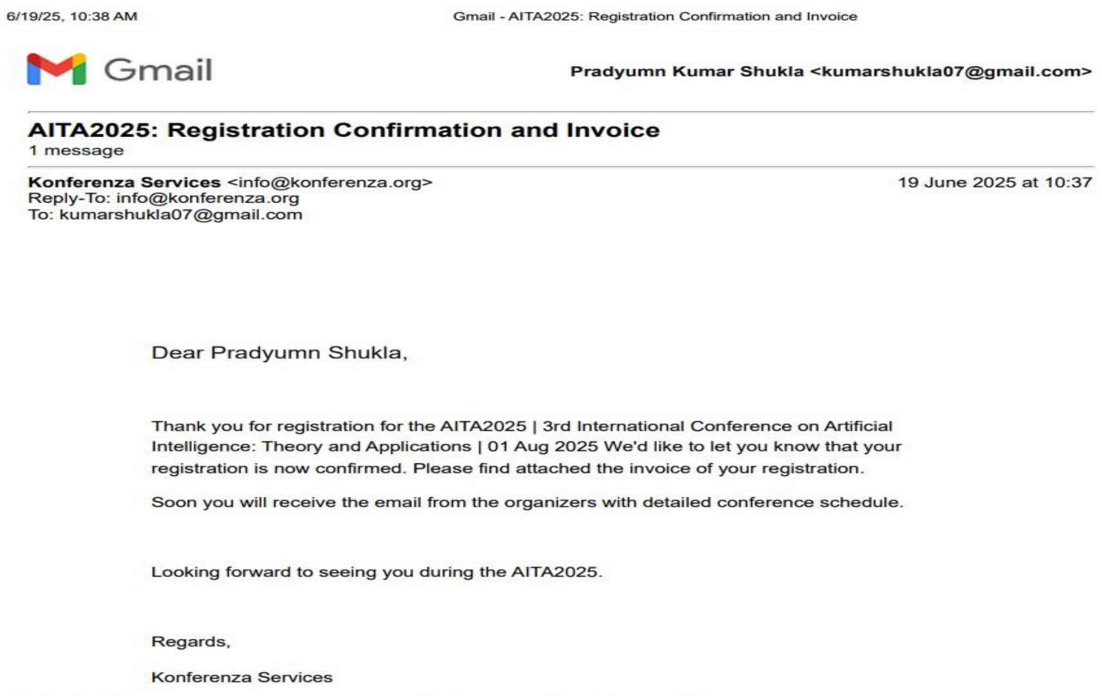
- After the conference, it will take a minimum of 8–9 months for Springer to complete the publication process.
- This includes final editing, copyright clearance, technical checks, and indexing.
- Authors will be notified by organisers and Springer once the volume is published.

9. Awards and Recognitions

- Best Paper Awards will be given in each technical session.
- All registered presenters will receive Certificates of Presentation.

- **Paper Status -** Accepted and registered

Thank you for submitting your manuscript to the 3rd International Conference on Artificial Intelligence: Theory and Applications (AITA 2025) to be organized by ICFAI Business School (IBS) Bangalore, India during August 01-02, 2025 in Hybrid Mode. Proceedings of AITA 2025 will be published in the SCOPUS Indexed Springer Book Series, 'Lecture Notes in Networks and Systems'.
We are pleased to inform you that based on reviewers' comments, your paper titled "ADVANCEMENT IN TEXT-TO-SQL SYSTEM USING LARGE LANGUAGE MODEL(LLMs): A REVIEW" has been accepted for presentation during AITA 2025, and publication in the proceedings to be published in Scopus-indexed Springer Book Series "Lecture Notes in Networks and Systems".



Pradyumn Kumar Shukla <kumarshukla07@gmail.com>

AITA2025: Registration Confirmation and Invoice
1 message

Konferenza Services <info@konferenza.org> 19 June 2025 at 10:37
Reply-To: info@konferenza.org
To: kumarshukla07@gmail.com

Dear Pradyumn Shukla,

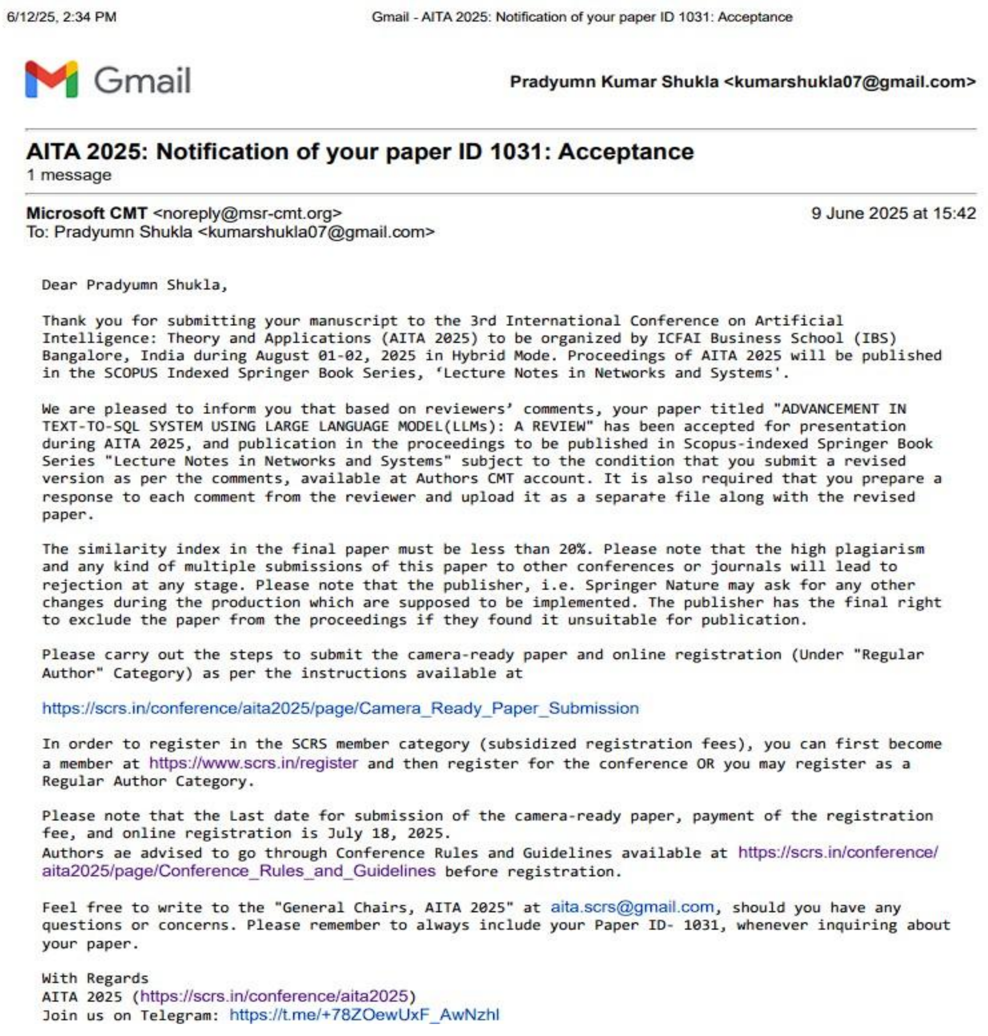
Thank you for registration for the AITA2025 | 3rd International Conference on Artificial Intelligence: Theory and Applications | 01 Aug 2025 We'd like to let you know that your registration is now confirmed. Please find attached the invoice of your registration.

Soon you will receive the email from the organizers with detailed conference schedule.

Looking forward to seeing you during the AITA2025.

Regards,

Konferenza Services



Pradyumn Kumar Shukla <kumarshukla07@gmail.com>

AITA 2025: Notification of your paper ID 1031: Acceptance
1 message

Microsoft CMT <noreply@msr-cmt.org> 9 June 2025 at 15:42
To: Pradyumn Shukla <kumarshukla07@gmail.com>

Dear Pradyumn Shukla,

Thank you for submitting your manuscript to the 3rd International Conference on Artificial Intelligence: Theory and Applications (AITA 2025) to be organized by ICFAI Business School (IBS) Bangalore, India during August 01-02, 2025 in Hybrid Mode. Proceedings of AITA 2025 will be published in the SCOPUS Indexed Springer Book Series, 'Lecture Notes in Networks and Systems'.

We are pleased to inform you that based on reviewers' comments, your paper titled "ADVANCEMENT IN TEXT-TO-SQL SYSTEM USING LARGE LANGUAGE MODEL(LLMs): A REVIEW" has been accepted for presentation during AITA 2025, and publication in the proceedings to be published in Scopus-indexed Springer Book Series "Lecture Notes in Networks and Systems" subject to the condition that you submit a revised version as per the comments, available at Authors CMT account. It is also required that you prepare a response to each comment from the reviewer and upload it as a separate file along with the revised paper.

The similarity index in the final paper must be less than 20%. Please note that the high plagiarism and any kind of multiple submissions of this paper to other conferences or journals will lead to rejection at any stage. Please note that the publisher, i.e. Springer Nature may ask for any other changes during the production which are supposed to be implemented. The publisher has the final right to exclude the paper from the proceedings if they found it unsuitable for publication.

Please carry out the steps to submit the camera-ready paper and online registration (Under "Regular Author" Category) as per the instructions available at

https://scrs.in/conference/aita2025/page/Camera_Ready_Paper_Submission

In order to register in the SCRS member category (subsidized registration fees), you can first become a member at https://www.scrs.in/register and then register for the conference OR you may register as a Regular Author Category.

Please note that the Last date for submission of the camera-ready paper, payment of the registration fee, and online registration is July 18, 2025.
Authors ae advised to go through Conference Rules and Guidelines available at https://scrs.in/conference/aita2025/page/Conference_Rules_and_Guidelines before registration.

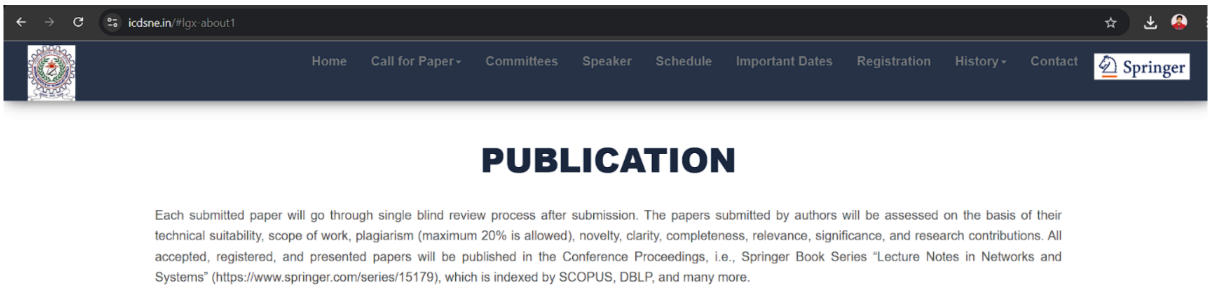
Feel free to write to the "General Chairs, AITA 2025" at aita.scrs@gmail.com, should you have any questions or concerns. Please remember to always include your Paper ID- 1031, whenever inquiring about your paper.

With Regards
AITA 2025 (https://scrs.in/conference/aita2025)
Join us on Telegram: https://t.me/+78ZOewUxF_AwNzhl

Paper 2 Details

Conference name - Third International Conference on Data Science and Network Engineering

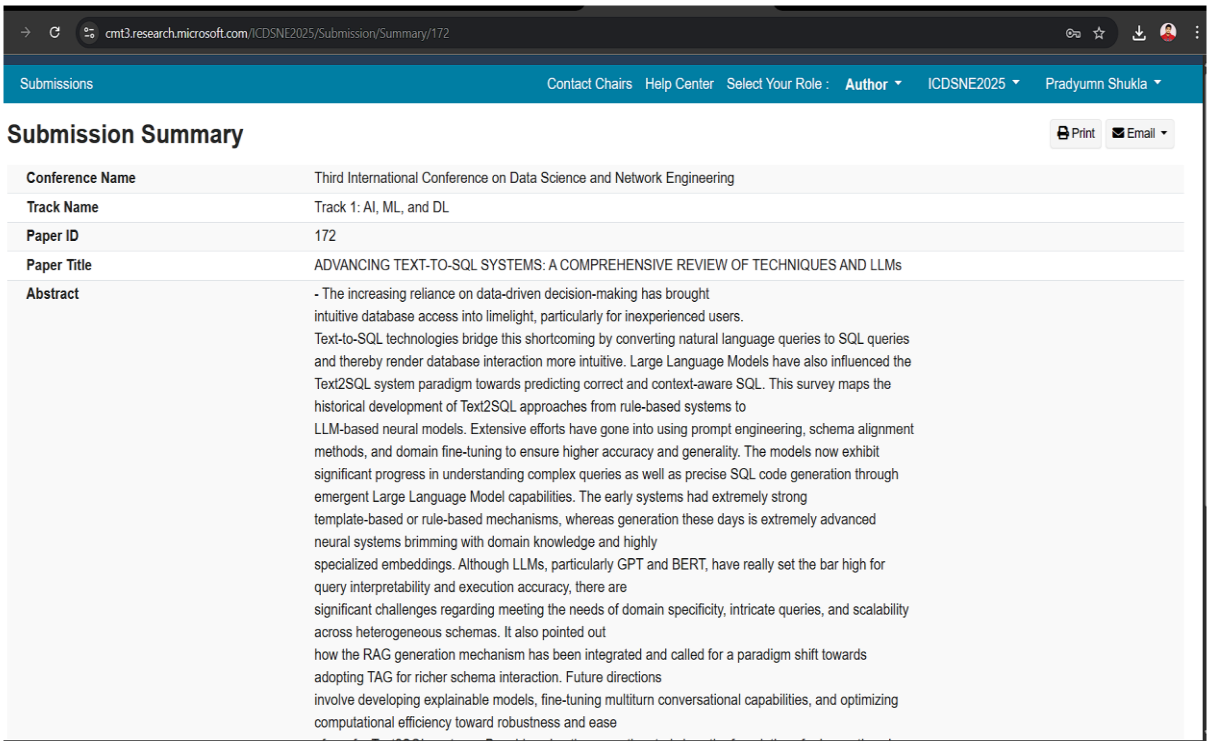
- **Indexing of conference** – Each submitted paper will go through single blind review process after submission. The papers submitted by authors will be assessed on the basis of their technical suitability, scope of work, plagiarism (maximum 20% is allowed), novelty, clarity, completeness, relevance, significance, and research contributions. All accepted, registered, and presented papers will be published in the Conference Proceedings, i.e., Springer Book Series "Lecture Notes in Networks and Systems" (https://www.springer.com/series/15179), which is indexed by SCOPUS, DBLP, and many more.



PUBLICATION

Each submitted paper will go through single blind review process after submission. The papers submitted by authors will be assessed on the basis of their technical suitability, scope of work, plagiarism (maximum 20% is allowed), novelty, clarity, completeness, relevance, significance, and research contributions. All accepted, registered, and presented papers will be published in the Conference Proceedings, i.e., Springer Book Series "Lecture Notes in Networks and Systems" (https://www.springer.com/series/15179), which is indexed by SCOPUS, DBLP, and many more.

- **Paper Status** – Communicated



Submission Summary

Conference Name	Third International Conference on Data Science and Network Engineering
Track Name	Track 1: AI, ML, and DL
Paper ID	172
Paper Title	ADVANCING TEXT-TO-SQL SYSTEMS: A COMPREHENSIVE REVIEW OF TECHNIQUES AND LLMs
Abstract	- The increasing reliance on data-driven decision-making has brought intuitive database access into limelight, particularly for inexperienced users. Text-to-SQL technologies bridge this shortcoming by converting natural language queries to SQL queries and thereby render database interaction more intuitive. Large Language Models have also influenced the Text2SQL system paradigm towards predicting correct and context-aware SQL. This survey maps the historical development of Text2SQL approaches from rule-based systems to LLM-based neural models. Extensive efforts have gone into using prompt engineering, schema alignment methods, and domain fine-tuning to ensure higher accuracy and generality. The models now exhibit significant progress in understanding complex queries as well as precise SQL code generation through emergent Large Language Model capabilities. The early systems had extremely strong template-based or rule-based mechanisms, whereas generation these days is extremely advanced neural systems brimming with domain knowledge and highly specialized embeddings. Although LLMs, particularly GPT and BERT, have really set the bar high for query interpretability and execution accuracy, there are significant challenges regarding meeting the needs of domain specificity, intricate queries, and scalability across heterogeneous schemas. It also pointed out how the RAG generation mechanism has been integrated and called for a paradigm shift towards adopting TAG for richer schema interaction. Future directions involve developing explainable models, fine-tuning multiturn conversational capabilities, and optimizing computational efficiency toward robustness and ease

DECLARATION

We/I hereby certify that the work which is presented in the Major Project-II/Research Work entitled Analysis and development of Text-to-SQL Translation system Using Large Language Model (LLMs) in fulfilment of the requirement for the award of the Degree of Bachelor/Master of Technology in DATA SCIENCE and submitted to the Department of SOFTWARE ENGINEERING, Delhi Technological University, Delhi is an authentic record of my/our own, carried out during a period from Aug 2023 to May 2025, under the supervision of Dr. Sanjay Patidar.

The matter presented in this report/thesis has not been submitted by us/me for the award of any other degree of this or any other Institute/University. The work has been published/accepted/communicated in SCI/ SCI expanded/SSCI/Scopus indexed journal OR peer reviewed Scopus indexed conference with the following details:

Title of the Paper: Advancement in TEXT-TO-SQL system using Large Language model (LLMs): A Review
Author names (in sequence as per research paper): Pradyumn Shukla, Dr. Sanjay Patidar
Name of Conference/Journal: 3rd International Conference on Artificial Intelligence: Theory and Applications (AITA 2025) to be organized by ICFAI Business School (IBS) Bangalore, India
Conference Dates with venue (if applicable): ICFAI Business School (IBS) Bangalore, India during August 01-02, 2025 in Hybrid Mode.
Have you registered for the conference (Yes/No): Yes
Status of paper (Accepted/Published/Communicated): Accepted
Date of paper communication: 5/29/2025, 5:02:01 PM
Date of paper acceptance: 09/06/2025
Date of paper publication:

Student(s) Roll No., Name and Signature

SUPERVISOR CERTIFICATE

To the best of my knowledge, the above work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere. I, further certify that the publication and indexing information given by the students is correct.

Place: ____________________ **Supervisor Name and Signature**

Date: _____________________

NOTE: PLEASE ENCLOSE RESEARCH PAPER ACCEPTANCE /PUBLICATION /COMMUNICATION PROOF ALONG WITH SCOPUS INDEXING PROOF
(Conference Website OR Science Direct in case of Journal Publication)

DECLARATION

We/I hereby certify that the work which is presented in the Major Project-II/Research Work entitled Analysis and development of Text-to-SQL Translation system Using Large Language Model (LLMs) in fulfilment of the requirement for the award of the Degree of Bachelor/Master of Technology in DATA SCIENCE and submitted to the Department of SOFTWARE ENGINEERING, Delhi Technological University, Delhi is an authentic record of my/our own, carried out during a period from Aug 2023 to May 2025, under the supervision of Dr. Sanjay Patidar**.**

The matter presented in this report/thesis has not been submitted by us/me for the award of any other degree of this or any other Institute/University. The work has been published/accepted/communicated in SCI/ SCI expanded/SSCI/Scopus indexed journal OR peer reviewed Scopus indexed conference with the following details:

Title of the Paper: Advancing TEXT to SQL system: A comprehensive review of techniques and LLM
Author names (in sequence as per research paper): Pradyumn Shukla, Dr. Sanjay Patidar
Name of Conference/Journal: Third International Conference on Data Science and Network Engineering (ICDSNE 2025)
Conference Dates with venue (if applicable): National Institute of Technology Agartala (NIT Agartala), India, in hybrid mode (online/in-person) on 25-26 July, 2025.
Have you registered for the conference (Yes/No)? No
Status of paper (Accepted/Published/Communicated): Communicated
Date of paper communication: 02/06/2025, 2:53:50 PM
Date of paper acceptance:
Date of paper publication:

Student(s) Roll No., Name and Signature

SUPERVISOR CERTIFICATE

To the best of my knowledge, the above work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere. I, further certify that the publication and indexing information given by the students is correct.

Place: ____________________ **Supervisor Name and Signature**

Date: _____________________

NOTE: PLEASE ENCLOSE RESEARCH PAPER ACCEPTANCE /PUBLICATION /COMMUNICATION PROOF ALONG WITH SCOPUS INDEXING PROOF
(Conference Website OR Science Direct in case of Journal Publication)