Work Plan

Project Title: Develop a predictive model to accurately forecast hourly traffic volumes at different road junctions based on historical traffic data
Prepared by: Virendra
Date: 08.12.2025

1. Project Overview

The goal of this Mentorship project is to build a robust, production-ready predictive model to forecast hourly traffic volumes for multiple road junctions using historical traffic data. These forecasts will help in traffic management, planning, and decision-making by providing short-term (next 24–168 hours) predictions for each junction.

2. Objectives

- Assess dataset quality, coverage, missingness, anomalies
- Perform EDA to understand temporal patterns, seasonality, junction behavior
- Engineer time-series features (lags, rolling stats, calendar features)
- Train and compare ML models (LightGBM/XGBoost, plus optional sequence models)
- Evaluate using time-series cross-validation (MAE, RMSE, MAPE)
- Produce final hourly forecasts in required CSV format
- Prepare documentation and final report

3. Scope & Deliverables

Deliverables include:

- **Work_Plan_Virendra.pdf** (this document)
- **eda_notebook.ipynb**
- **feature_engineering.py**
- **train_model.py**
- **model_lgbm.pkl** (best model)
- **forecast_next24.csv**
- **evaluation_report.pdf**
- **README.md**

Upload Requirements:

- Document must be **<10 MB**
- Use filename format: **title_name**
- If PDF is mentioned, upload **only PDF**
- You may upload any additional research material

4. Methodology & Steps

Phase A – Data Understanding & Cleaning

- Inspect columns, parse timestamps
- Verify hourly frequency, resample if needed
- Handle missing values and outliers

Phase B – EDA

- Visualize hourly, daily, weekly patterns
- Create heatmaps, seasonal plots
- Compare junction-level behavior
- (Optional) Analyze external data like weather/holidays

Phase C – Feature Engineering

- Calendar features: hour, day_of_week, month, is_weekend
- Lag features: lag_1, lag_24, lag_168
- Rolling windows (3h, 6h, 24h)
- Junction encoding (label/target)
- Fourier terms for seasonality (optional)

Phase D – Modeling & Validation

- Baselines: naïve, last-week
- Main model: **LightGBM**
- Alternatives: XGBoost, RandomForest, LSTM/Transformer
- Time-series cross-validation
- Evaluate with MAE, RMSE, MAPE

Phase E – Forecasting & Packaging

- Implement multi-step forecasting
- Generate final CSV with
  timestamp, junction_id, forecast_count
- Create validation graphs

5. Timeline (7-day plan)

Date	Tasks
08.12.2025	Finalize Work Plan; inspect dataset
09.12.2025	Data cleaning + EDA; create eda_notebook. ipynb
10.12.2025	Feature engineering + baseline models
11.12.2025	Train ML models; hyperparameter tuning
12.12.2025	Generate final forecasts; evaluate holdout window
13.12.2025	Prepare evaluation_report.pdf, README, packaging
14.12.2025	Final review and submission

6. Tools & Environment

- Python 3.8+
- Libraries: pandas, numpy, scikit-learn, lightgbm, xgboost, matplotlib, seaborn, joblib
- Jupyter Notebook / Google Colab
- GitHub or Drive for version control and sharing

7. Risks & Mitigation

- **Sparse data:** Use global models + imputation
- **Overfitting:** Track per-junction errors, try alternative models
- **Large file size:** Compress PDFs, include only required outputs

8. Submission Checklist

- File name: **Work_Plan_Virendra.pdf**
- Size <10 MB
- If sharing via Drive → Access: **Anyone with link → Viewer**

9. Next Steps

- Begin EDA and feature engineering
- Set up modeling pipeline
- Prepare notebooks and scripts in a reproducible structure

Leveraging RAG-LLMs for Urban Mobility Simulation and Analysis

Yue Ding, Conor McCarthy, Kevin O'Shea and Mingming Liu

***Abstract*— With the rise of smart mobility and shared e-mobility services, numerous advanced technologies have been applied to this field. Cloud-based traffic simulation solutions have flourished, offering increasingly realistic representations of the evolving mobility landscape. LLMs have emerged as pioneering tools, providing robust support for various applications, including intelligent decision-making, user interaction, and real-time traffic analysis. As user demand for e-mobility continues to grow, delivering comprehensive end-to-end solutions has become crucial. In this paper, we present a cloud-based, LLM-powered shared e-mobility platform, integrated with a mobile application for personalized route recommendations. The optimization module is evaluated based on travel time and cost across different traffic scenarios. Additionally, the LLM-powered RAG framework is evaluated at the schema level for different users, using various evaluation methods. Schema-level RAG with XiYanSQL achieves an average execution accuracy of 0.81 on system operator queries and 0.98 on user queries.**

***Index Terms*— Smart Mobility, Shared E-Mobility, Traffic Simulation, Route Optimization, RAG, Cloud-Based Platform.**

I. Introduction

Recent advances in smart mobility, information technologies, and data analytics have enabled the development of sustainable transportation solutions [1]. Building on these, the Mobility as a Service (MaaS) paradigm integrates public transit, shared mobility, and private vehicles into a unified, user-centric platform. Within MaaS, shared e-mobility services like e-cars, e-bikes, and e-scooters provide flexible, on-demand alternatives to private vehicles, promoting sustainable urban travel [2]. With the growing deployment of eHubs, which are clusters of shared mobility services such as e-cars and e-bikes, there is an increasing need for simulation-based toolsets to evaluate their impact on urban mobility.

Existing commercial e-mobility platforms [3], [4] are already operational across the globe. While these tools are widely used, their proprietary nature imposes significant limitations for researchers and practitioners aiming to explore diverse scenarios and customize solutions. Simulation-based solutions have emerged as a preferable alternative, offering flexibility and adaptability for studying various e-mobility scenarios. In this context, traditional traffic simulation platforms such as AIMSUN [5] and VISSIM [6] primarily address traffic congestion and scheduling, whereas emerging e-mobility solutions focus on energy management and sustainability. These platforms offer some space for customization but are still not open-sourced. Among these, the open-source tool Simulation of Urban MObility (SUMO) has gained widespread adoption in the research community due to its versatility and extensibility [7]. However, despite its strengths, SUMO has certain limitations, particularly in handling large-scale scenarios and integrating with modern cloud-based infrastructures. These limitations become critical in shared e-mobility, where factors such as poor service integration, limited scalability, lack of user-centric solutions, and inadequate energy consumption forecasting remain insufficiently addressed [8].

To address these challenges, there is a growing need for cloud-based SUMO solutions that leverage the scalability of cloud platforms. Recent advancements in cloud-based SUMO focus on enhancing large-scale simulations and real-time data processing, but most studies concentrate on vehicular network optimization rather than shared e-mobility. [9] offers a high-level cloud-based SUMO framework may be limited to simple queries or pure dashboard-based designs which is non-user-friendly. Meanwhile, the emergence of Large Language Model (LLM)s has opened new opportunities for enhancing intelligent decision-making in traffic-related domains, including delivery scheduling, location prediction, and secure communications. Building on this potential, we propose an LLM-powered, cloud-based platform that integrates dynamic traffic simulation with natural language interaction, aiming to bridge the gap between traditional simulation tools and user-friendly, adaptive shared mobility solutions. In particular, our platform supports scenario-based deployment and evaluation of eHub distributions, addressing the lack of scalable simulation toolsets for shared mobility infrastructures. This work builds upon our previous research on energy modeling [10], optimization algorithms [11], and the multi-modal optimization framework for shared e-mobility [12]. Thanks to the modular and data-centric architecture, our platform can be easily adapted for digital twin modeling and deployment in different regions. It separates the system logic from specific datasets, meaning that with appropriate traffic and mapping information, the platform can recreate realistic local mobility patterns without extensive reengineering. The platform's key contributions are outlined next:

- **Comprehensive Cloud-based Open-Source Shared e-mobility Platform:** We present an open-source framework for shared e-mobility with modular components for traffic simulation, docking station deployment, multi-modal transport, energy-aware route planning, and user interaction via an Android app using an agent-in-

Y. Ding is with the Centre for Research Training in Machine Learning (ML-Labs) at Dublin City University. C. McCarthy, K. O'Shea and M. Liu are with the Insight Centre for Data Analytics and the School of Electronic Engineering, Dublin City University, Dublin, Ireland. This work has emanated from research supported in part by Insight Centre for Data Analytics at Dublin City University under Grant Number *SFI/12/RC/2289_P2* and by Éireann – Research Ireland through the Research Ireland Centre for Research Training in Machine Learning (18/CRT/6183). *Corresponding author: Mingming Liu. Email:* mingming.liu@dcu.ie.

the-loop approach. A Dockerized architecture ensures cloud scalability. Source code will be released upon paper acceptance.

- **LLM-powered Traffic Simulation Data and Travel History Data Query Enhancement:** We present the first Retrieval-Augmented Generation (RAG)-based approach tailored for post-simulation querying over structured traffic simulation data and user travel data. By integrating LLMs with schema-level RAG, our platform enables natural language interaction with simulation outputs for different types of users, including end users and system operators. The schema-level RAG framework supports context-aware Text-to-SQL generation, achieving an average execution accuracy of 0.98 on user queries and 0.81 on system operator queries. This empowers both users and system operators to explore personalized, fine-grained insights going beyond predefined charts or views, and makes simulation data more understandable and accessible in real-world scenarios.

The remainder of the paper is organized as follows: Section II presents a literature review of relevant platforms and provides a comparison with ours. Section III introduces the architecture and details of the proposed platform. introduces the technical details for LLM layer. Section V outlines the experimental setup and analyzes the results. Finally, Section VI discusses the conclusions and future work.

II. Literature Review

Recent developments in e-mobility platforms aim to tackle urban congestion, environmental concerns, and transport optimization. However, current solutions often lack integration of energy management, real-time cloud simulation, and user-centric features. This section reviews classical traffic simulation platforms, e-mobility platforms, RAG-powered transportation systems, and commercial and public shared e-mobility platforms, highlighting their strengths, limitations, and gaps. It also emphasizes our platform's unique contributions in providing a scalable, multi-modal, and user-focused approach to shared e-mobility simulation and optimization.

A. Transportation Simulation Platform

Classical traffic simulation platforms, such as AIMSUN [5] and VISSIM [6], primarily address issues like congestion and scheduling through microscopic modeling [13]. SUMO [7] further introduced a hybrid approach combining different mathematical models for algorithm evaluation. While traditional platforms focus on congestion and accident analysis, recent efforts have begun integrating e-mobility to assess impacts on energy, urban economies, and communities. Examples include digital twin-based platforms linking e-mobility and smart grids [14], sustainable EV adoption tools [15], and data-driven EV impact simulations [16]. Although studies have explored shared e-mobility impacts [17], [18], few have developed scalable simulation platforms with native support for shared e-mobility and eHub distribution, which is often oversimplified in smart city contexts [19]. [8] proposed a multi-agent platform using deep reinforcement learning for EV fleet management, but research on scalable, multi-modal shared e-mobility simulation remains limited.

B. LLM-powered Transportation System

A notable starting point of LLM application in transportation is the work presented in [20], which discusses the utilization of LLMs and RAG in Smart Urban Mobility. However, the study primarily focuses on system design without delving into the implementation details. For traffic scenario generation, the RealGen framework introduced in [21] offers a promising approach. RealGen synthesizes new scenarios by combining behaviors from multiple retrieved examples in a gradient-free manner. These behaviors can originate from predefined templates or tagged scenarios, providing flexibility and adaptability in scenario creation. In the domain of intelligent driving assistance, the IDAS system presented in [22] demonstrates an efficient approach to reading and comprehending car manuals for immediate, context-based aid. This system provides new insights into enhancing driving assistance by leveraging the capabilities of RAG models. Compared to these works, ChatSUMO [23] provides a more comprehensive and operational framework by integrating LLMs into the full pipeline of SUMO traffic scenario generation, and customization, significantly reducing the technical barriers for users without traffic simulation expertise. In contrast, our approach focuses specifically on the post-simulation stage, extending LLM integration on the user side to enable schema-level interaction with structured simulation outputs. While prior work primarily emphasizes data generation or simulation modeling, we target the downstream task of querying and interpreting simulation results. Unlike conventional systems that only support fixed dashboard metrics or static visualizations, our platform uniquely integrates RAG-LLM interfaces to allow flexible, natural language querying over arbitrary simulation results.

III. Cloud-Based Platform Architecture

In this section, we present a detailed of the system's architecture and implementation, focusing on its core components and their interconnections.

A. System Overview

The platform's architecture, shown in Fig. 1, integrates key components for real-time traffic simulation, data processing, and user interaction. SUMO generates traffic data, which flows through Google Pub/Sub for efficient streaming and into Google Cloud Bigtable (GCB) for storage and analysis. GCB supports real-time visualization via the dashboard and optimal route calculations in the optimization module. Users interact through the dashboard to monitor traffic and manage e-mobility options, while the mobile app provides route recommendations. The schema-level RAG module uses GCB simulation data to generate context-aware responses.

B. Cloud Layer

*1) **Traffic Simulator Module:*** The simulation module is a critical component designed to emulate real-world traffic

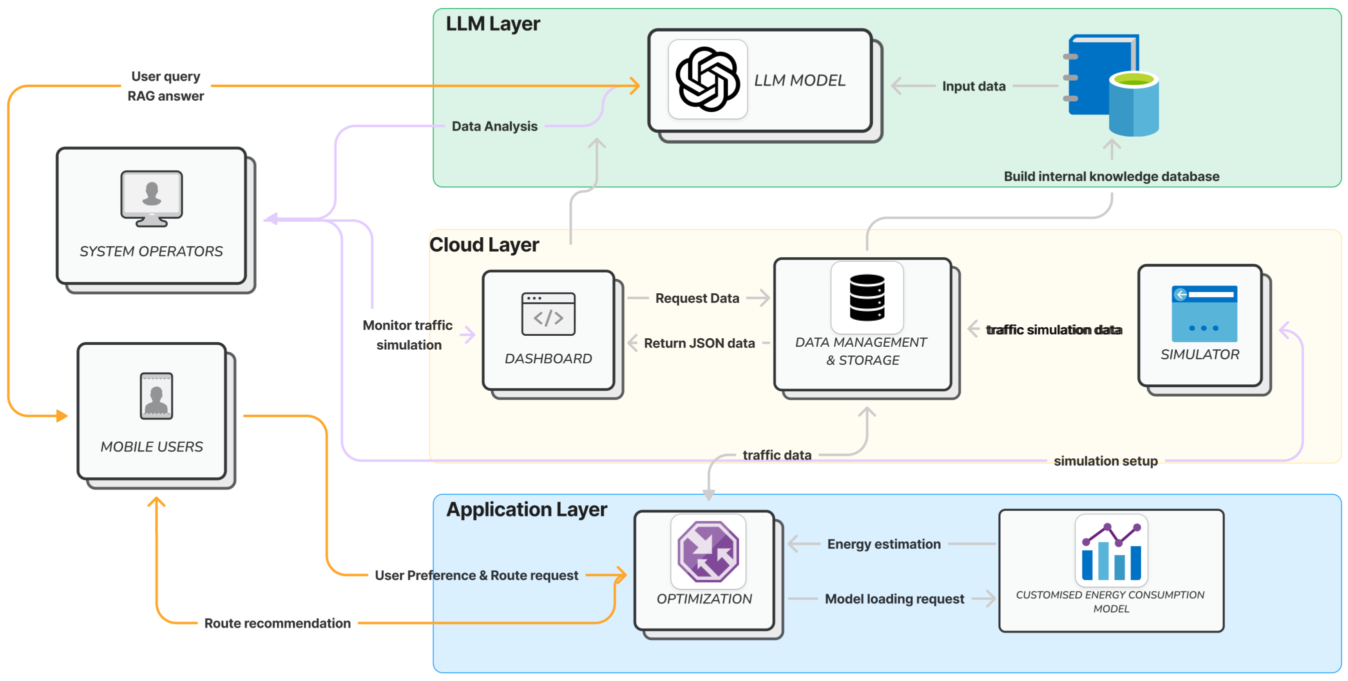


Fig. 1: System flow diagram showing the interactions between components.

conditions for evaluating the shared e-mobility framework. It provides a realistic environment to test and validate various aspects, such as traffic patterns, vehicle availability, and user interactions, ensuring that the platform's recommendations are reliable and effective. The simulation process begins with a scenario setup initiated by the system operator, who configures the simulation environment using a JSON file to define key parameters and load configuration data. A central server manages user profiles and processes route requests, forwarding them to a multi-modal optimization module. By leveraging a cloud-based implementation of SUMO, the module ensures scalability and accessibility, making it capable of handling complex, large-scale traffic simulations with efficiency. Pedestrian and bicycle trips in SUMO were generated using `randomTrips`, with spawn rates varying by period as shown in Table I.

TABLE I: Bicycle and Pedestrian Route File Configuration

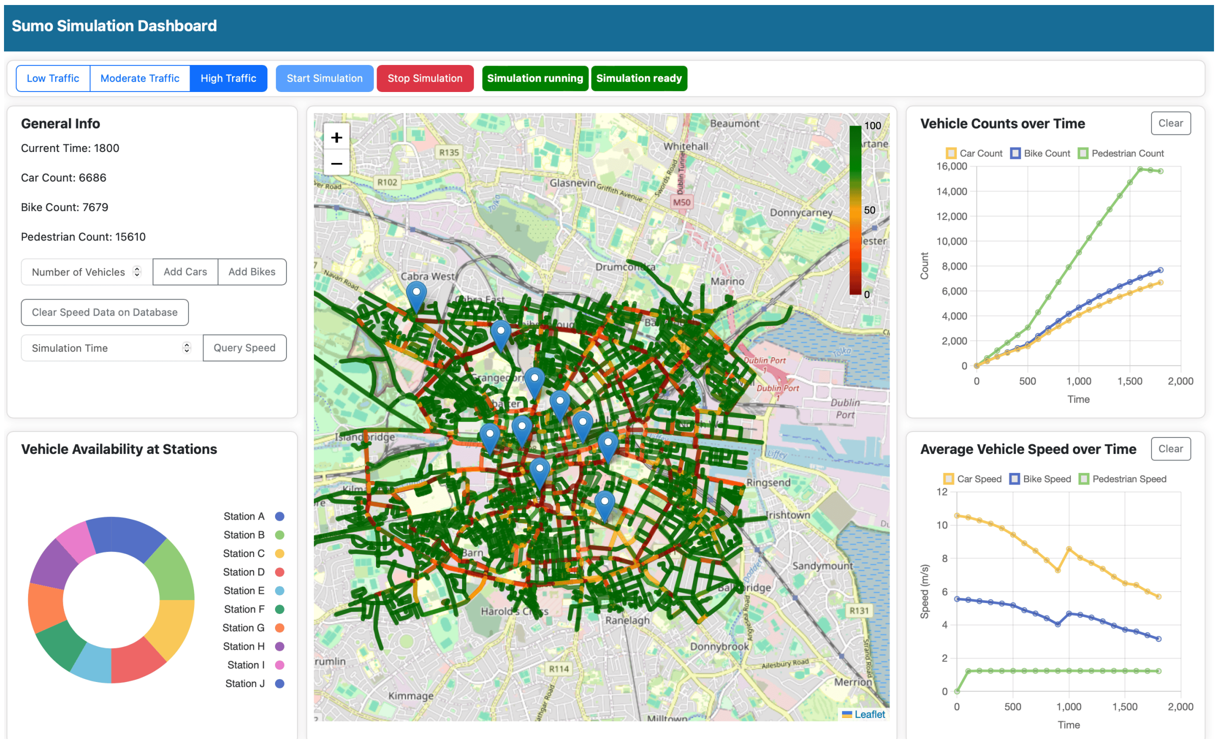
Route Type	Simulation Time (seconds)	Spawn Rate (seconds)
Bicycles	0 - 20,000	7
	20,000 - 62,000	1.5
	62,000 - 70,000	2.5
	70,000 - 86,400	5
Pedestrians	0 - 20,000	2
	20,000 - 62,000	0.75
	62,000 - 70,000	0.8
	70,000 - 86,400	1.5

*2) **Data Management and Storage Module:*** The data storage module, built on Google Cloud Bigtable (GCB), ensures high-throughput, low-latency storage of time-series traffic data from SUMO simulations and supports the RAG module by providing schema-aware data access. Using SSD-backed storage and Google Cloud Pub/Sub for ingestion, GCB organizes traffic and station data by timestamp and edge ID, while vehicle availability is stored separately. A subscriber script ingests and batches SUMO data efficiently to maintain real-time performance.

*3) **Dashboard:*** The interactive dashboard is served as the central interface for controlling and visualizing various aspects of the shared e-mobility emulation platform. It displays real-time traffic simulations for the Dublin City Centre (DCC) map, utilizing GCB as the backend for handling large-scale, time-series traffic data generated by the SUMO system. It ensures high performance and scalability, offering users a seamless way to set up scenarios, control simulations, and analyze traffic patterns in real-time, as illustrated in Fig. 2.

C. Application Layer

*1) **Optimization Module:*** The multimodal transportation optimization module is designed to provide user-centric solutions while leveraging a strategically pre-designed distribution of transportation stations. This framework integrates real-time traffic simulation data from SUMO with user-specific inputs such as origins, destinations, and transportation preferences. At the core of the framework is a Mixed-Integer Linear Programming (MILP) model, which optimizes travel time while adhering to constraints such as energy consumption and maximum transfer times. and support multi-modal e-mobility transportation. The detailed solution has been given in previous work [11]. The energy consumption is predicted by energy model in [10]. We employ the user-in-the-loop design principle to allow users to visualize and interact with optimal multi-modal routes under different scenario configurations, such as varying eHub distributions or traffic conditions. Unlike traditional static routing systems, our approach enables users to explore dynamic trade-offs between travel time, energy consumption, and transfer


Sumo Simulation Dashboard

General Info

Current Time: 1800

Car Count: 6686

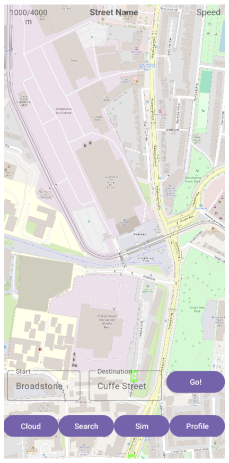
Bike Count: 7679

Pedestrian Count: 15610

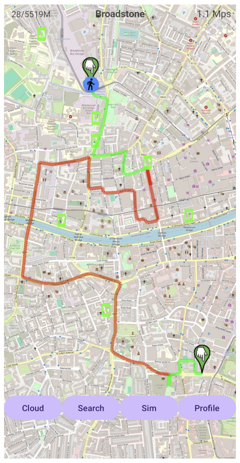
Fig. 2: Outlook of the Interactive Dashboard.

flexibility in real-time, fostering a deeper understanding of sustainable mobility choices.

*2) **Android Mobile Application**:* An Android app has been developed to improve e-mobility route planning. Users input their origin, destination, and trip preferences. Leveraging real-time traffic simulation data, the app generates optimized routes based on time and energy constraints. Routes are visualized dynamically with color-coded transport modes and estimated speed. Fig. 3a and Fig. 3b illustrate the input and route display features. The app offers an efficient travel solution with real-time data integration.



(a) User input of start and destination locations.



(b) Route with different transportation modes.

Fig. 3: Screenshots of the Android app: users select start and destination points (a), and the app recommends a route (b), with green for walking and red for e-scooter paths.

IV. LLM Layer: RAG Module

We introduce a two-stage Text-to-SQL generation framework that integrates schema-aware retrieval with LLMs. Our system supports two types of structured databases: (i) system-level traffic simulation data and (ii) user-level historical travel records. We assume two types of users in the system: system operators, who are responsible for managing and monitoring simulation data; and mobile users, who interact with the system based on their travel history. The implementations steps are as follows:

*1) **M-Schema Generation**:* M-Schema is proposed by the XiYanSQL project,[1] is a structured format that encodes relational database schemas into a model-friendly JSON, preserving both table- and column-level semantics. Compared to unstructured table documentation, it better supports LLMs in aligning natural language queries with database structure. We adopt the official open-source M-Schema framework, which enables automatic conversion from relational databases. To generate M-Schema from our PostgreSQL database which is linked with GCB, we follow these steps:

1) Connect to the database: We use SQLAlchemy to connect to the PostgreSQL instance.
2) Parse schema: The `schema_engine.py` script introspects the database schema and extracts metadata including table names, primary/foreign key relationships, and column types.
3) Generate M-Schema JSON: The extracted schema is transformed into a JSON file containing structured representations for each table and its columns.

[1]https://github.com/XGenerationLab/M-Schema

*2) **Schema Embedding and Storage**:* To support schema-aware generation, we first convert all database M-Schema descriptions into vector representations. Each schema is written as a separate natural language text file and loaded using a preprocessing script. We use the `all-MiniLM-L6-v2` model from SentenceTransformers to encode these schema texts into dense embeddings. The resulting vectors are stored in a local Chroma vector database, where each schema document is indexed with a unique ID and associated metadata.

*3) **Schema Indexing**:* At query time, a user-provided natural language question is encoded into the same embedding space using the same transformer model. A vector similarity search is then performed against the stored schema embeddings using cosine similarity, defined as:

$$sim(\boldsymbol{q}, \boldsymbol{d}) = \frac{\boldsymbol{q} \cdot \boldsymbol{d}}{\|\boldsymbol{q}\| \, \|\boldsymbol{d}\|} \tag{1}$$

where $\boldsymbol{q}$ and $\boldsymbol{d}$ represent the embedding vectors of the query and a schema document, respectively.

We retrieve the top-3 most semantically relevant schema documents based on their similarity scores. These documents are then concatenated to form the context input for LLMs. This retrieval mechanism supplies schema-specific context prior to SQL generation.

*4) **SQL Generation**:* We adopt five different LLMs to generate SQL queries given a natural language question and the retrieved schema context:

- **XiYanSQL:** A schema-aware Text-to-SQL model using M-Schema as input, selected for its top-three ranking on BirdSQL[2] and its publicly available API.
- **GPT-4:** OpenAI's general-purpose LLM with strong Text-to-SQL performance, including on BirdSQL.
- **GPT-4o:** A low-latency, cost-efficient variant of GPT-4 that maintains competitive Text-to-SQL performance and ranks highly on BirdSQL.
- **Gemini 1.5 Pro:** Google's multimodal model excelling in long-context and Text-to-SQL tasks.
- **Gemini 2.0 Flash:** A faster, enhanced Gemini variant optimized for everyday tasks.

V. Experiments and Evaluation

In this section, we introduce the experiments which are conducted to evaluate the platform and analyze the results.

A. Experiment Setup

1) Simulation Setup: The DCC map was selected for funding requirements and its prior use in a study on autonomous vehicles. Covering a 5 km x 3.5 km area, the existing SUMO map provided a solid foundation for simulating a typical 24-hour workday using traffic data from Dublin's SCATS system, which collects vehicle counts every 6 minutes from 480 locations. Data from Transport Infrastructure Ireland, averaged over several months, modeled realistic traffic behavior [24]. To support multi-modal simulations, modifications were made to include pedestrian and bicycle lanes using SUMO's `Netconvert` tool. Traffic data from the 2022 Irish census [25], showing 87,000 walkers and 37,000 cyclists, were incorporated, with peak times between 5:30 am and 7 pm.

[2]`https://bird-bench.github.io/`

2) Optimization Module Evaluation Setup: The evaluation of our optimization model was conducted using simulated data generated from a traffic simulator. The traffic scenarios considered include *Low Traffic*, *Medium Traffic*, and *High Traffic*, reflecting varying congestion levels within an urban environment. The transportation network was modeled using the road network provided in the DCC.net.xml file, which was parsed and transformed into a directed graph using the NetworkX library. The optimization problem was formulated and solved using the PuLP library with the CBC solver, ensuring efficient computation of the optimal routes for different traffic conditions. The setup involved generating paths for given 400 Origin-Destination (OD) pairs. Vehicle types considered include e-cars, e-bikes, and e-scooters, along with walking as an option. The performance of the module was assessed by analyzing the extra time costs under different traffic conditions.

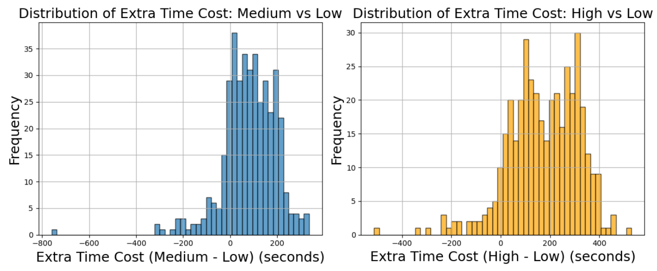


Fig. 4: Distribution of extra travel time cost under different traffic scenarios.

3) RAG Module Evaluation Setup: The Schema-Level RAG module was evaluated from two perspectives: system operators and mobile users. For system operators, the retrieved data consisted of traffic simulation outputs. For mobile users, the queried data included each user's trip history stored on cloud. The evaluation for the Schema-Level RAG module was implemented in Python with the following key components:

Traffic Data: A database containing edge-level traffic information such as edge_id, simulation_time, pedestrian_speed, bike_speed, and car_speed, as well as station-level data including vehicle types and battery levels.

User Travel History: Simulated user-level trajectory data, including start_edge, end_edge, travel time cost, execution time, and optimal path sequence.

Evaluation Queries: A set of 100 natural language questions and corresponding ground-truth SQL queries were manually created for each type of user, resulting in 200 annotated QA pairs. These ensure high-quality evaluation and reliable SQL correctness checking.

Large Language Models: XiYanSQL, GPT-4, GPT-4o, Gemini 1.5 Pro, Gemini 2.0 Flash.

Embedding Model: `all-MiniLM-L6-v2` for similarity-based retrieval and question similarity scoring.

Model	Domain	Execution Accuracy	F1	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-1	ROUGE-2	ROUGE-L
XiYanSQL	System	**0.81**	**0.9183**	**0.8360**	**0.7825**	**0.7212**	0.6822	**0.8916**	**0.7802**	**0.8867**
GPT-4	System	0.74	0.8870	0.8101	0.7647	0.7238	**0.6959**	0.8753	0.7594	0.8710
GPT-4o	System	0.70	0.8814	0.7880	0.7291	0.6664	0.6272	0.8664	0.7297	0.8637
Gemini 2.0 Flash	System	0.66	0.8746	0.7026	0.6010	0.5224	0.4718	0.7792	0.5633	0.7703
Gemini 1.5 Pro	System	0.65	0.8613	0.7594	0.7083	0.6705	0.6433	0.8375	0.7052	0.8319
XiYanSQL	User	**0.98**	**0.9755**	**0.9526**	**0.9371**	**0.9125**	**0.8999**	**0.9698**	**0.9358**	**0.9698**
GPT-4	User	0.89	0.9119	0.9330	0.9096	0.8889	0.8765	0.9483	0.9014	0.9467
Gemini 2.0 Flash	User	0.88	0.9061	0.9213	0.8902	0.8694	0.8512	0.9390	0.9126	0.9529
GPT-4o	User	0.84	0.8981	0.9052	0.8721	0.8380	0.8185	0.9329	0.8671	0.9321
Gemini 1.5 Pro	User	0.83	0.8526	0.8439	0.8172	0.7275	0.7192	0.8590	0.8267	0.8973

TABLE II: Evaluation of SQL generation on system-level and user-level queries. All scores are reported as decimals.

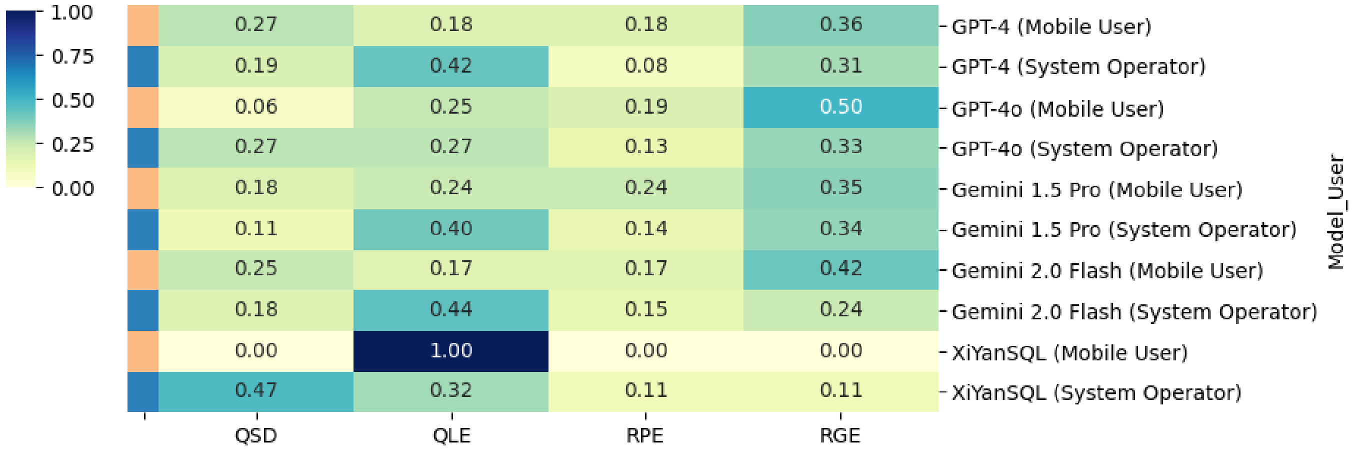


Fig. 5: Error Type Proportions Across Different Models and Users

System Operator Question:	Which edge_id has the most stations?
Ground Truth SQL:	SELECT edge_id, COUNT(*) AS station_count FROM stations GROUP BY edge_id ORDER BY station_count DESC LIMIT 1;
Generated SQL:	SELECT edge_id, COUNT(station_id) AS station_count FROM stations GROUP BY edge_id ORDER BY station_count DESC LIMIT 1;
User Question:	Find the top 3 most frequent destinations.
Ground Truth SQL:	SELECT end_edge, COUNT(*) AS freq FROM user_paths GROUP BY end_edge ORDER BY freq DESC LIMIT 3;
Generated SQL:	SELECT end_edge, COUNT(*) AS end_edge_count FROM user_paths GROUP BY end_edge ORDER BY end_edge_count DESC LIMIT 3;

TABLE III: Correct SQL Predictions from XiYanSQL

System Operator Question:	Get the average bike speed for the road segments that have at least one station.
Ground Truth SQL:	SELECT o.edge_id, AVG(o.bike_speed) AS avg_bike_speed FROM online_demo o JOIN stations s ON o.edge_id = s.edge_id GROUP BY o.edge_id;
Generated SQL:	SELECT AVG(bike_speed) AS average_bike_speed FROM online_demo WHERE edge_id IN (SELECT edge_id FROM stations);
User Question:	Find paths where the optimal route contains more than 4 road segments.
Ground Truth SQL:	SELECT * FROM user_paths WHERE LENGTH(optimal_path_sequence) - LENGTH(REPLACE(optimal_path_sequence, ’(’, ”)) > 4;
Generated SQL:	SELECT * FROM user_paths WHERE LENGTH(optimal_path_sequence) - LENGTH(REPLACE(optimal_path_sequence, ’,’, ”)) > 4;

TABLE IV: Incorrect SQL Predictions from XiYanSQL

Evaluation Metrics: For structured SQL generation evaluation, we follow the Spider benchmark [26]:

- **Execution Accuracy:** Whether executing the predicted SQL matches the ground-truth result.
- **Component Match (F1):** Token overlap across SELECT, WHERE, GROUP BY, and ORDER BY clauses.
- **BLEU Scores (1–4):** N-gram overlaps with smoothing to measure surface similarity.
- **ROUGE Scores:** ROUGE-1, ROUGE-2, and ROUGE-L for assessing overlap and fluency.

B. Evaluation and Analysis

1) Optimization Module: The results of the optimization module evaluation are presented in Fig. 4, which compares the extra travel time cost required for all OD pairs across different traffic scenarios. The figure displays two histograms: one comparing medium vs. low traffic scenarios and another comparing high vs. low traffic scenarios. These distributions provide insights into the additional delays introduced by increasing congestion levels.

The key observations from the results are as follows:

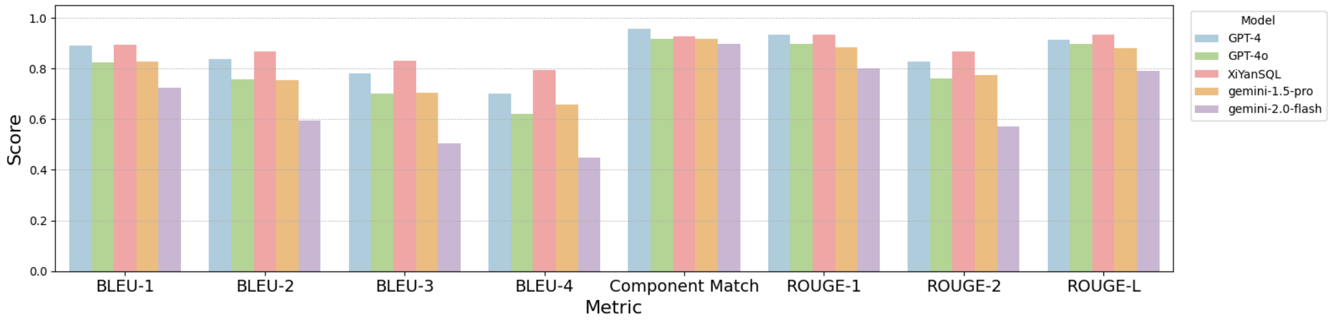


Fig. 6: Comparison of Median Value of Evaluation Metrics for Models on System Operator Queries

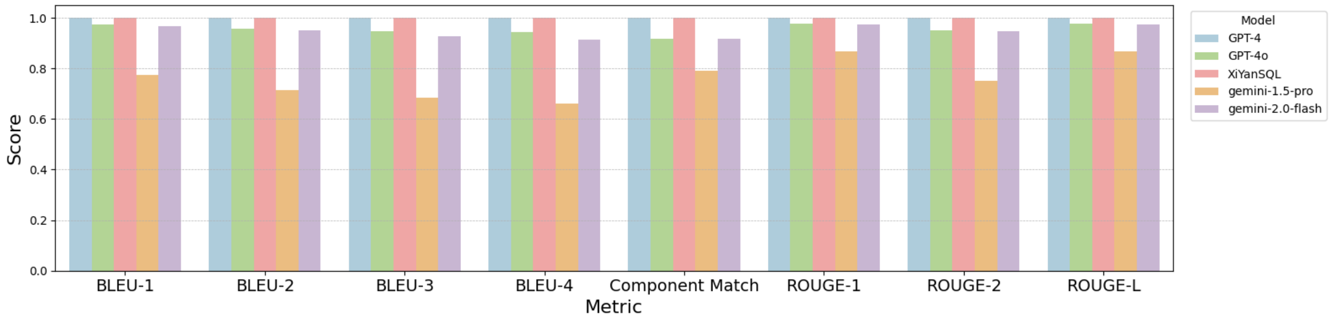


Fig. 7: Comparison of Median Value of Evaluation Metrics for Models on User Queries

- **Medium vs. Low Traffic:** The histogram shows a moderate increase in extra travel time with a wider spread compared to the low traffic scenario, and the Extra Time Cost (Medium - Low) expectation is 81.01 seconds. This suggests that congestion begins to impact route efficiency, but remains manageable.
- **High vs. Low Traffic:** A significant increase in travel time cost is observed, with a much broader distribution, and the Extra Time Cost (High - Low) expectation is 172.42 seconds. This indicates higher variability and suggests that congestion has a substantial effect on route optimization, leading to more frequent delays.

2) RAG module: For schema-level RAG, as shown in Table II, XiYanSQL achieves the highest execution accuracy with 0.81 on system operator queries and 0.98 on user-level queries. GPT-4 obtains the highest BLEU-4 score of 0.6959 on system operator queries. Overall, models perform better on user queries due to simpler schema structures. Fig. 6 illustrates the median scores across multiple evaluation metrics for each model on system operator queries. GPT-4 and XiYanSQL consistently perform well across BLEU and ROUGE metrics, with GPT-4 showing the highest BLEU-1 to BLEU-4 scores and strong component match accuracy. Fig. 7 presents the median scores across different evaluation metrics on user queries. XiYanSQL maintains consistently high median values in all metrics including BLEU, Component Match and ROUGE, indicating both accuracy and stability in its outputs. GPT-4 and GPT-4o also perform well, though slight drops appear in BLEU-3 and BLEU-4. In contrast, the Gemini models, especially gemini-1.5-pro, have lower median scores in BLEU-3, BLEU-4 and ROUGE-2, reflecting difficulties in generating coherent and complete responses. These trends support that simpler schemas in user queries yield better and more consistent model performance.

Furthermore, we define four error types to analyze model performance: Query Structure Differences (QSD), including mismatched fields, incorrect use of LIMIT, wrong table choices, and aggregation mistakes; Query Logic Errors (QLE), such as incomplete or incorrect JOIN and filter conditions; Result Precision Errors (RPE), caused by missing operators like DISTINCT or ABS; and Result Granularity Errors (RGE), related to selecting too many or too few fields. This taxonomy extends [27] by distinguishing precision from granularity issues. As visualized in Fig. 5, each cell represents the proportion of a specific error type normalized by the total number of errors for each model-user pair. XiYanSQL shows a high rate of structure-related errors on system operator queries, with QSD accounting for 47% of its total. GPT-4 and GPT-4o exhibit the highest share of logic errors, at 42% and 44% respectively, especially under complex conditions. Gemini models show more than 30% granularity errors on user queries, indicating difficulty in selecting the appropriate number of fields. Precision errors remain consistently low across all models, generally below 20%. These results suggest that schema alignment is model-specific, while logic and granularity issues are common across schema-level RAG.

Table III and Table IV present representative examples of correct and incorrect SQL generated by XiYanSQL. The correct cases demonstrate the model's ability to capture the core intent and structure of the query, even when using alternative but semantically equivalent SQL formulations. In contrast, the incorrect cases reveal common failure patterns, such as mismatched aggregation conditions, which often lead to semantically inaccurate results despite syntactic similarity.

VI. Conclusion and Future Works

This work presents a comprehensive shared e-mobility platform that integrates cloud-based simulation, real-time data processing, and RAG-enhanced decision-making to address challenges in sustainable urban transportation. The platform supports dynamic multi-modal routing and flexible docking-based systems through a scalable and user-centric design. Schema-level RAG evaluation highlights XiYanSQL's high execution accuracy of 0.81 on system operator queries and 0.98 on user queries. Error analysis reveals that logic errors are the predominant failure type for most models, while granularity issues are more common in user-level queries. It is important to note that, although the LLM-based RAG framework perform robustly overall, generated responses are not always perfect. Users and system operators should treat LLM outputs as supportive decision-making aids rather than definitive conclusions. Based on our evaluation, we recommend cross-checking critical decisions against simulation data or dashboard outputs when high reliability is required. Future work will focus on extending the system to support multi-objective optimization and multi-user scenarios, enabling greater adaptability to diverse real-world demands. As urban mobility becomes increasingly complex and interconnected, integrating public transportation networks into the platform is a natural next step. Additionally, the RAG framework will be enhanced by incorporating external multi-modal knowledge sources to further improve the faithfulness and contextual grounding of generated responses. Beyond these directions, we also plan to extend the platform toward digital twin modeling by linking to real-world transportation data and live traffic conditions. While the current platform focuses on arbitrary scenario generation with predefined traffic flow configurations using historical traffic data, future enhancements will aim to support more dynamic and real-world simulations and multi-modal data inputs, including text and images, to further improve realism and decision-making support.

References

[1] X. Zhang, R. Zheng, J. Huo, H. Yang, and Y. Jiang, "Factors influencing the market share of e-bike sharing: evidence from new york city," *Transportation*, vol. 50, no. 6, pp. 2045–2067, dec 2023.

[2] M. Hasselwander, S. Nieland, K. Dematera-Contreras, and M. Goletz, "Maas for the masses: Potential transit accessibility gains and required policies under mobility-as-a-service," *Multimodal Transportation*, vol. 2, no. 3, p. 100086, Sep. 2023.

[3] Bird, "Bird official website," 2023, available at: https://www.bird.co/ (accessed: 23.04.2023).

[4] Moovit, "Moovit san francisco bay area points of interest," 2023, available at: https://moovitapp.com/sf_bay_area_ca-22/poi/en-gb (accessed: 08.05.2023).

[5] J. Barceló and J. Casas, "Dynamic network simulation with aimsun," *Simulation approaches in transportation analysis: Recent advances and challenges*, pp. 57–98, 2005.

[6] M. Fellendorf and P. Vortisch, "Microscopic traffic flow simulator vissim," *Fundamentals of traffic simulation*, pp. 63–93, 2010.

[7] D. Krajzewicz, G. Hertkorn, C. Rössel, and P. Wagner, "Sumo (simulation of urban mobility)-an open-source traffic simulation," in *Proceedings of the 4th middle East Symposium on Simulation and Modelling (MESM20002)*, 2002, pp. 183–187.

[8] M. Luo, B. Du, W. Zhang, T. Song, K. Li, H. Zhu, M. Birkin, and H. Wen, "Fleet rebalancing for expanding shared e-mobility systems: A multi-agent deep reinforcement learning approach," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[9] P. H. Nguyen, A. Hugo, K. Svantorp, and B. M. Elnes, "Towards a simulation framework for edge-to-cloud orchestration in c-its," in *2020 21st IEEE International Conference on Mobile Data Management (MDM)*. IEEE, Jun. 2020, p. 354–358.

[10] Y. Ding, S. Yan, M. H. Shah, H. Fang, J. Li, and M. Liu, "Data-driven energy consumption modelling for electric micromobility using an open dataset," in *2024 IEEE Transportation Electrification Conference and Expo (ITEC)*. IEEE, Jun. 2024, p. 1–7.

[11] M. H. Shah, Y. Ding, S. Zhu, Y. Gu, and M. Liu, "Optimal design and implementation of an open-source emulation platform for user-centric shared e-mobility services," 2024.

[12] M. H. Shah, J. Li, and M. Liu, "On scalable design for user-centric multi-modal shared e-mobility systems using milp and modified dijkstra's algorithm," 2024.

[13] H. Xiao, R. Ambadipudi, J. Hourdakis, and P. Michalopoulos, "Methodology for selecting microscopic simulators: Comparative evaluation of aimsun and vissim," 2005.

[14] K.-S. Cho, S.-W. Park, and S.-Y. Son, "Digital twin-based simulation platform with integrated e-mobility and distribution system," in *CIRED Porto Workshop 2022: E-mobility and power distribution systems*, vol. 2022. IET, 2022, pp. 1158–1162.

[15] M. Ferrara, B. Monechi, G. Valenti, C. Liberto, M. Nigro, and I. Biazzo, "A simulation tool for energy management of e-mobility in urban areas," in *2019 6th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, 2019, pp. 1–7.

[16] D. Echternacht, I. El Haouati, R. Schermuly, and F. Meyer, "Simulating the impact of e-mobility charging infrastructure on urban low-voltage networks," in *NEIS 2018; Conference on Sustainable Energy Supply and Energy Storage Systems*. VDE, 2018, pp. 1–6.

[17] T. Campisi, N. Ali, K. D. Alemdar, Ö. Kaya, M. Y. Çodur, and G. Tesoriere, "Promotion of e-mobility and its main share market: Some considerations about e-shared mobility," in *AIP Conference Proceedings*, vol. 2611, no. 1. AIP Publishing LLC, 2022, p. 060002.

[18] F. Liao and G. Correia, "Electric carsharing and micromobility: A literature review on their usage pattern, demand, and potential impacts," *International Journal of Sustainable Transportation*, vol. 16, no. 3, pp. 269–286, 2022.

[19] S.-Y. Son and S.-W. Park, "Future of shared e-mobility operation service platforms considering distribution systems in smart cities," in *CIRED Porto Workshop 2022: E-mobility and power distribution systems*, vol. 2022. IET, 2022, pp. 1029–1033.

[20] H. Xu, J. Yuan, A. Zhou, G. Xu, W. Li, X. Ban, and X. Ye, "Genai-powered multi-agent paradigm for smart urban mobility: Opportunities and challenges for integrating large language models (llms) and retrieval-augmented generation (rag) with intelligent transportation systems," 2024.

[21] W. Ding, Y. Cao, D. Zhao, C. Xiao, and M. Pavone, *RealGen: Retrieval Augmented Generation for Controllable Traffic Scenarios*. Springer Nature Switzerland, Oct. 2024, p. 93–110.

[22] L.-B. Hernandez-Salinas, J. Terven, E. A. Chavez-Urbiola, D.-M. Córdova-Esparza, J.-A. Romero-González, A. Arguelles, and I. Cervantes, "Idas: Intelligent driving assistance system using rag," *IEEE Open Journal of Vehicular Technology*, vol. 5, p. 1139–1165, 2024.

[23] S. Li, T. Azfar, and R. Ke, "Chatsumo: Large language model for automating traffic scenario generation in simulation of urban mobility," *IEEE Transactions on Intelligent Vehicles*, pp. 1–12, 2024.

[24] M. Guériau and I. Dusparic, "Quantifying the impact of connected and autonomous vehicles on traffic efficiency and safety in mixed traffic," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–8.

[25] Central Statistics Office, "Census of population 2022 - summary results," https://data.cso.ie, 2023, [Accessed: Oct. 17, 2024].

[26] T. Yu, R. Zhang, K. Yang, M. Yasunaga, D. Wang, Z. Li, J. Er, and D. Radev, "Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018, pp. 3911–3921.

[27] J. Shen, C. Wan, R. Qiao, J. Zou, H. Xu, Y. Shao, Y. Zhang, W. Miao, and G. Pu, "A study of in-context-learning-based text-to-sql errors," 2025.