Spatial-RAG: Spatial Retrieval Augmented Generation for Real-World Geospatial Reasoning Questions

Dazhou Yu[*] Emory University
Riyang Bao[*] Emory University
Ruiyu Ning Emory University
Jinghong Peng
Gengchen Mai University of Texas at Austin
Liang Zhao[†] Emory University

Abstract

Answering real-world geospatial questions—such as finding restaurants along a travel route or amenities near a landmark—requires reasoning over both geographic relationships and semantic user intent. However, existing large language models (LLMs) lack spatial computing capabilities and access to up-to-date, ubiquitous real-world geospatial data, while traditional geospatial systems fall short in interpreting natural language. To bridge this gap, we introduce Spatial-RAG, a Retrieval-Augmented Generation (RAG) framework designed for geospatial question answering. Spatial-RAG integrates structured spatial databases with LLMs via a hybrid spatial retriever that combines sparse spatial filtering and dense semantic matching. It formulates the answering process as a multi-objective optimization over spatial and semantic relevance, identifying Pareto-optimal candidates and dynamically selecting the best response based on user intent. Experiments across multiple tourism and map-based QA datasets show that Spatial-RAG significantly improves performance over strong baselines.

1 Introduction

Spatial reasoning questions are those that require spatial computing to resolve relationships between objects, positions, or movements in space. Extensive research has been conducted on abstract spatial reasoning tasks such as mental rotation, block manipulation, and robot navigation, which rely on simplified, small-scale, and often purely geometric representations, typically addressed using techniques from computer vision and robotics. In contrast, geospatial reasoning involves interpreting large-scale, real-world geographic data where spatial information is deeply entangled with rich semantics (Chen, 2014; Mai et al., 2021; Kefalidis et al., 2024). For example, urban routing decisions depend not only on road geometries but also on attributes such as traffic regulations, land use, and temporal constraints. Travel plan recommendations should not only consider minimizing the travel distance, but also maximize the quality of the attractions according to their descriptions and reviews (Xie et al., 2024).

Figure 1: A spatial reasoning question with nearby spatial objects. Areas satisfying the spatial constraint are highlighted in purple.

Geospatial reasoning has a longstanding role in AI research, yet classical methods—such as spatial databases and GIS query systems—lack the ability to effectively interpret users' natural language questions (Mai et al., 2021). On the other hand, large language models (LLMs) exhibit strong linguistic competence but struggle with spatial computing and geospatial grounding (Mai et al., 2024). Recent efforts to bridge this gap have focused on prompt engineering (Manvi et al., 2024b; Gurnee and Tegmark, 2024), but these approaches heavily rely on LLMs' internal knowledge, which remains limited in generalization and spatial reasoning capabilities, significantly suffering from geographic bias (Faisal and Anastasopoulos, 2023;

[*]Equal contribution.
[†]Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 11094–11112

Manvi et al., 2024a; Wu et al., 2024), and being susceptible to obsolescence as knowledge evolves (Chen et al., 2025). Some work has explored fine-tuning LLMs on spatial tasks (Ji and Gao, 2023; Manvi et al., 2024b; Zhang et al., 2024), but the resulting models are often tailored to narrow applications, constrained datasets, or specific geographic domains. Therefore, there remains a critical need for a general-purpose geospatial reasoning framework that **synergizes semantic understanding and spatial computation** while ensuring access to **real-world, vast, fast-changing, and complex geospatial data**.

To fill this gap, this paper aims to augment LLMs with capabilities of spatial reasoning and accessibility to real-world geospatial data. For example, as illustrated in Figure 1, answering the question requires LLMs to elicit and formulate the user's textual request into the problem of "finding points near the polyline" and solve it based on a geospatial map (database) with semantic information (e.g., customer reviews and location profiles). Then, it also requires inferring user intent to select the spatially and semantically preferred candidates. Thus, the system must seamlessly integrate structured spatial retrieval with unstructured text-based reasoning, ensuring both spatial accuracy and contextual understanding. Specifically, we extend Retrieval-Augmented Generation (RAG) into geospatial information retrieval and reasoning, bridging the gap between structured spatial databases and unstructured textual reasoning. RAG has demonstrated its effectiveness in knowledge-intensive tasks, such as question answering (QA) (Siriwardhana et al., 2023), by retrieving domain-specific documents to enhance LLM responses. However, existing RAG systems primarily focus on retrieving and generating textual content and lack the spatial intelligence required for spatial reasoning tasks, especially tasks that involve understanding and computing complex spatial relationships among geometries, including points, polylines, and polygons.

In this paper, we introduce *Spatial Retrieval-Augmented Generation (Spatial-RAG)*, a new framework that unifies text-guided spatial retrieval with spatially aware text generation in a multi-objective optimization scenario. Specifically, to identify spatially relevant candidate answers, we propose a novel spatial hybrid retrieval module synergizing spatial sparse and dense retrievers. To rank the candidates and generate the final answers, we propose to fuel the generator with retrieved results on the Pareto frontier based on a spatial and semantic joint ranking strategy. Our contributions are summarized as follows:

- **A generic spatial RAG framework**: We introduce spatial-RAG, the first framework that extends RAG to geospatial question answering, to tackle a broad spectrum of spatial reasoning tasks, such as geographic recommendation, spatially constrained search, and contextual route planning. Our approach seamlessly integrates spatial databases, LLMs, and retrieval-based augmentation, enabling effective handling of complex spatial reasoning questions directly within the familiar operational paradigm of LLMs.
- **Sparse-dense spatial hybrid retriever**: We propose a hybrid retrieval mechanism that combines spatial sparse retrieval (e.g., SQL-based structured queries) with spatial dense retrieval (e.g., LLM-powered semantic matching). This dual approach ensures that retrieved results align both spatially and semantically with the user's query, synergizing spatial computing and geographical text understanding.
- **Multi-objective guided spatial and semantic text generator:** To handle both spatial constraints and semantic intents in the spatial question-answering task, we introduce a multi-objective optimization framework that dynamically balances trade-offs between spatial and semantic relevancy to the user's query. This ensures that the generated responses are both geospatially accurate and linguistically coherent.
- **Real-world evaluation:** We evaluated our method on multiple real-world datasets consisting of user QA pairs about various spatial entities. The experiments demonstrate the model's ability to handle spatial reasoning questions grounded in real-world scenarios.

Through these innovations, Spatial-RAG significantly enhances the spatial reasoning capabilities of LLMs, bridging the gap between structured spatial databases and natural language QA.

2 Related Work

2.1 Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) enhances language models by dynamically retrieving external knowledge before generating a response, improving factual accuracy and contextual relevance (Fan et al., 2024; Lewis et al., 2020). This approach has proven effective for knowledge-intensive tasks

like open-domain question answering (QA) and has expanded from text documents to structured data types such as tables and graphs (He et al., 2024). However, while RAG has been explored for knowledge-grounded dialogues (Yu et al., 2024c), its application in spatial reasoning QA remains a key research gap, as existing systems often fail to integrate spatial computation effectively.

2.2 Geospatial Question Answering

Geospatial questions can be broadly categorized into two types. The first, textual knowledge-based questions (e.g., "What is the population of Los Angeles?"), can be answered using traditional QA and RAG methods because they do not require complex spatial computation (Liétard et al., 2021; Christmann and Weikum, 2024; Contractor et al., 2021b). The second and more challenging type involves spatial reasoning questions (e.g., "What is the position of object A relative to object B?"), which require a deep understanding of spatial data and relationships (Li et al., 2024). Beyond question answering, recent work has also explored specialized GeoAI models for heterogeneous multi-source spatial point prediction and for learning expressive representations of polygonal geometries (Yu et al., 2024a,b, 2025), underscoring the complexity of the spatial structures and data modalities that real-world geospatial reasoning systems must handle. Although studies have investigated the spatial capabilities of LLMs, they often struggle with accurate reasoning, even after fine-tuning or enriching semantic context by converting coordinates to addresses (Mai et al., 2024; Roberts et al., 2023; Li et al., 2023). Furthermore, many existing methods are limited by their reliance on predefined actions for specific tasks, highlighting the need for more robust and flexible reasoning frameworks.

3 Problem Formulation

In this study, our primary focus is on **Geospatial Reasoning Questions**. We formulate the problem as follows: Given a query q, the system aims to generate an answer y^*, which maximizes the joint spatial and semantic scores while satisfying the spatial constraints in the query q. For example, in Figure 1, the desired answer is a restaurant that satisfies a spatial constraint——it must be within walking distance of the route. Additionally, it should ideally be located along the street (spatial score) and preferably offer waffle fries (semantic score). This problem can be formulated as the following multi-objective optimization problem:

$$\begin{aligned} y^* &= \arg\max_y \ \lambda_s^T f_s(q,y) + \lambda_k^T f_k(q,y) \\ \text{s.t.} \quad & y \in C_s(q), \quad y \in C_k(q), \\ & \lambda_s \geq 0, \quad \lambda_k \geq 0, \\ & \mathbf{1}^T\lambda_s + \mathbf{1}^T\lambda_k = 1, \end{aligned} \tag{1}$$

where $f_s \in \mathbb{R}^{d_s}$ is the spatial relevance score vector, $f_k \in \mathbb{R}^{d_k}$ is the semantic relevance score vector, C_s is the spatial candidate set that satisfies the spatial constraints of the question, C_k is the semantic candidate set that satisfies the semantic constraints of the question, λ_s, λ_k are the spatial weights and semantic weights, respectively, y^* is the optimal answer, $\mathbf{1}^T\lambda_s + \mathbf{1}^T\lambda_k = 1$ ensures a normalized trade-off.

Note that this problem can have multiple valid solutions depending on the trade-off parameters λ_s and λ_k, forming a *Pareto frontier*. Existing approaches are unable to solve this problem effectively, as it demands a synergistic capability in both geospatial and semantic reasoning. Specifically: 1) it requires accurately determining whether a candidate y satisfies the spatial constraints expressed in the query q. 2) it involves ranking candidates based on both spatial and semantic relevance, which are interrelated yet provide complementary signals; and 3) it must ensure that no high-quality answers are overlooked under different trade-offs between spatial and semantic aspects. Current methods, which typically excel in either geospatial reasoning (e.g., spatial databases) or semantic reasoning (e.g., large language models and their variants), cannot address all of these requirements without substantial effort to integrate both capabilities seamlessly.

4 Spatial-RAG for Geospatial Reasoning Questions

4.1 Overview

Our Spatial-RAG framework, illustrated in Figure 2, operates in three key stages. First, to construct a spatial candidate set C_s, it must precisely define spatial constraints and then retrieve spatial objects that satisfy them. We achieve this by parsing natural language questions into spatial SQL queries to retrieve objects satisfying defined spatial constraints (Section 4.2). Second, to effectively compute spatial relevance $f_s(q,y)$ while integrating textual information, we propose a hybrid spatial

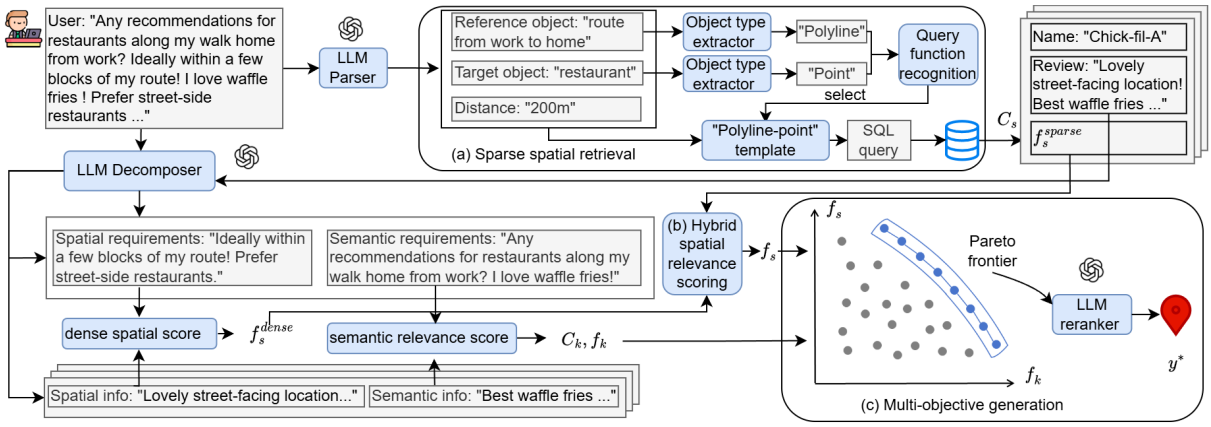

Figure 2: Illustration of the proposed Spatial-RAG framework.

retrieval scheme that combines sparse scores from the database with dense semantic similarity from text embeddings (Section 4.3). Third, it formulates a multi-objective optimization problem to balance spatial and semantic scores, computing a Pareto frontier of candidates from which the LLM generates an optimal response (Section 4.4).

4.2 Sparse Spatial Retrieval

The answer to a spatial reasoning question must meet specific spatial constraints $\mathcal{C}_s(q)$. The spatial candidate set $C_s(q)$ consists of all possible answers y that satisfy the spatial constraints:

$$C_s(q) = \{y \mid c_s(y, q) \leq 0, \forall c_s \in \mathcal{C}_s(q)\}, \quad (2)$$

where $c_s(y, q)$ is a constraint function that encodes a spatial condition (e.g., topological, directional, or distance-based) and $\mathcal{C}_s(q)$ is the set of all spatial constraints for question q. For example, if the spatial constraint requires y to be within a distance ϵ from a reference location l_q, then a possible constraint function is: $c_s(y, q) = \mathrm{d}(y, l_q) - \epsilon \leq 0$. This formulation ensures that only spatially valid answers are included in $C_s(q)$.

Addressing spatial constraints requires executing a spatial SQL query. This process involves identifying the appropriate query function, the reference spatial objects, the target spatial objects, and any necessary numerical parameters. Formally, a spatial SQL query can be expressed as:

$$Q_s = \mathcal{F}_s(G_r, G_t, \epsilon), \quad (3)$$

where $\mathcal{F}_s$ is the spatial function (e.g., proximity) that determines the relationship between objects, G_r and G_t are the reference and target objects, and ϵ is the set of numerical parameters (e.g., distance).

Given the diversity and complex nature of these constraints, we use a three-step incremental parsing process: 1) **Geometry recognition:** Identify reference G_r and target G_t objects and extract their geometric footprints. 2) **Query function selection:** Determine the spatial function $\mathcal{F}_s$ based on the intended relationship (e.g., containment). 3) **Parameter estimation:** Assign numerical constraints ϵ for precise filtering (e.g., buffer radius). This structured process enhances the LLM's ability to generate accurate and executable spatial SQL queries.

4.2.1 Geometry Recognition

Accurately identifying the spatial footprints of spatial objects is essential for parsing questions to spatial queries. We categorize spatial footprints $g \in \mathcal{G}$ into three types:

- **Point:** $\mathcal{G}_{\text{point}} = \{g \mid g \in \mathbb{R}^2, \dim(g) = 0\}$. Points, including multipoints, represents locations with negligible area. Examples include stop signs, points of interest, and a user's current location. In spatial databases, these entities are typically represented as the 'Point' geometry type.
- **Polyline:** $\mathcal{G}_{\text{line}} = \{g \mid g \subseteq \mathbb{R}^2, \dim(g) = 1\}$. Polylines, including multipolylines, represent one-dimensional objects with negligible width. Common examples include streets, streams, bus routes, and power lines. In spatial databases, these geometries are abstracted as the 'LineString' type.
- **Polygon:** $\mathcal{G}_{\text{polygon}} = \{g \mid g \subseteq \mathbb{R}^2, \dim(g) = 2\}$. Polygons, including multipolygons, represent two-dimensional objects that define enclosed areas. These geometries are essential for depicting regions such as census areas, parcels, counties, neighborhoods, and zoning areas.

The complexity of a query depends on the geome-

tries involved. A simple query might only involve points (e.g., "find nearest bus stop", and the spatial candidate set is $C_s = \{g | g \in \mathcal{G}_{\text{point}}, d(g, g_{\text{point}}) < \epsilon\}$ where $g_{\text{point}} \subseteq \mathcal{G}_{\text{point}}$ represents a point object (e.g., given location), ϵ is the distance threshold.), while a complex one could involve multiple types (e.g., "recommend a café along my route to the university campus", and the spatial candidate set is $C_s = \{g | g \in \mathcal{G}_{\text{point}}, g \in B(g_{\text{polyline}}, \epsilon) \cup g_{\text{polygon}}\}$, where $g_{\text{polyline}} \subseteq \mathcal{G}_{\text{polygon}}$ represents a polyline object (e.g., a route), $g_{\text{polygon}} \subseteq \mathcal{G}_{\text{polygon}}$ represents a polygonal region (e.g., a university campus), B is a buffer around the polyline, ϵ is the buffer size.).

By structuring spatial queries in this way, we ensure precise geometric representation, facilitating robust spatial reasoning and query execution.

4.2.2 Query Function Recognition and Parameter Estimation

After recognizing the geometries involved in a spatial query, the subsequent step is to determine the appropriate spatial query functions $\mathcal{F}_s$ required to handle various geometrical interactions. Despite the differing interactions among geometries, these can be uniformly addressed using distance functions $\mathrm{d}(g_r, g_t)$, which calculate the shortest distance between two geometrical entities $g_r, g_t \in \mathcal{G}$.

Formally, given sets of reference geometries $G_r \subseteq \mathcal{G}$ and target geometries $G_t \subseteq \mathcal{G}$, the spatial candidate set C_s can be defined as $\{g_t \in G_t \mid \exists g_r \in G_r, \mathrm{d}(g_r, g_t) \leq \epsilon\}$, if $\mathrm{d}(g_r, g_t) > 0$. Otherwise, it can be defined as $\{g_t \in G_t \mid \exists g_r \in G_r, g_r \cap g_t \neq \emptyset\}$. Parameters such as search radius or buffer distance ϵ are autonomously determined by the LLM, typically grounded in contextual understanding (e.g., estimated walking distance or area of interest). The parameter ϵ can be represented as: $\epsilon = \phi(q)$, where ϕ is a function that maps the context of the query q to an appropriate numerical value.

Once the geometries G_r, G_t, functions $\mathcal{F}_s$, and parameters ϵ are delineated, the system constructs the precise spatial query Q_s. This query can be formally expressed by Equation 3, which ensures exact retrievals from the spatial database, maintaining both accuracy and relevance in the results. By leveraging these mathematical formulations, the system effectively translates spatial reasoning tasks into executable queries, facilitating robust spatial intelligence within the LLM framework.

4.3 Hybrid Spatial Relevance Scoring

The spatial relevance score f_s consists of two components: a score derived from sparse spatial retrieval from the spatial database and a score from dense spatial retrieval based on text similarity between the question and the spatial descriptions of candidate objects. Formally, we define:

$$f_s = \lambda_p f_s^{\text{sparse}} + \lambda_d f_s^{\text{dense}}, \tag{4}$$

where λ_p and λ_d are weighting coefficients controlling the contribution of each score.

4.3.1 Sparse spatial relevance scoring

Sparse spatial relevance is computed directly from the spatial database by a spatial query function $\mathcal{F}_s$:

$$f_s^{\text{sparse}} = \begin{cases} \dfrac{1}{1 + d(g_r, g_t)}, & \text{if } g_r \cap g_t = \emptyset \\ 1, & \text{if } g_r \cap g_t \neq \emptyset \end{cases} \tag{5}$$

where g_r and g_t are reference and target spatial objects, respectively. $d(g_r, g_t)$ is a distance function measuring proximity in the spatial database. If g_t overlaps with g_r, we assign a perfect relevance score of 1. This ensures that objects within a region are maximally relevant, while those outside the region receive scores that decay with increasing distance.

4.3.2 Dense spatial relevance scoring

Dense spatial relevance is inferred from textual descriptions associated with spatial objects. We leverage an LLM to extract key spatial attributes from user queries and descriptions of candidates.

Given a user query q and a set of text descriptions d_t for spatial objects G_t, we extract the spatial requirements via an attention-based masking function: $v_{q,s} = \mathcal{E}(\mathcal{M}_s(q)), \quad v_{t,s} = \mathcal{E}(\mathcal{M}_s(d_t))$, where $v_{q,s}$ and $v_{t,s}$ are dense vector representations of spatial features from query q and text descriptions d_t. $\mathcal{M}_s$ is a function mapping input text to a spatial related text. $\mathcal{E}$ is the text encoder. The dense spatial relevance score f_s^{dense} is computed via cosine similarity between two vectors. (See details in Appendix B.2)

4.3.3 Hybrid spatial scoring as a generalized model

This hybrid approach generalizes both methods. If $\lambda_d = 0$, it becomes a purely sparse, distance-based ranking. If $\lambda_p = 0$, it is a purely dense, semantic-based ranking. When both are non-zero, it benefits from both explicit spatial constraints and implicit textual relevance.

4.4 Multi-objective Generation and Re-ranking

After computing both spatial (f_s) and semantic (f_k) relevance scores (details in Appendix A), the final stage is to select and generate the optimal answer. This is framed as a multi-objective optimization problem to resolve the inherent trade-off between the two dimensions (spatial and semantic).

4.4.1 Spatial-Semantic Pareto-Optimal Candidate Selection

To effectively manage the trade-off between spatial and semantic relevance, we first identify the set of Pareto-optimal candidates, $P(q)$. A candidate is considered Pareto-optimal if no other candidate is superior in both spatial and semantic scores. Formally, the Pareto frontier is defined as: $\{y \in C_s \cap C_k \mid \nexists\ y' \in C_s \cap C_k, f_s(q,y') \geq f_s(q,y)$ and $f_k(q,y') \geq f_k(q,y)$, with at least one strict inequality$\}$. This step efficiently filters the candidate pool to only the most viable options, ensuring no potentially optimal answer is discarded prematurely.

4.4.2 LLM-based Re-ranking and Generation

With the Pareto-optimal set $P(q)$ established, we leverage an LLM to make the final, context-aware trade-off decision. The LLM is prompted with the user's query, the candidates on the frontier, and their associated descriptions and sparse spatial scores. It then dynamically determines the context-specific weights (λ_s, λ_k) to select the final answer:

$$y^* = \arg\max_{y \in P(q)} \lambda_s^T f_s(q,y) + \lambda_k^T f_k(q,y), \quad (6)$$

Following this selection, the LLM generates a coherent natural language response.

5 Experiment

5.1 Experiment Setting

Datasets Our experiments are conducted on four real-world datasets. The **TourismQA-NYC** and **TourismQA-Miami** datasets (Contractor et al., 2019) consist of user questions about points of interest (POIs), with data sourced from TripAdvisor and other travel websites. We preprocessed this data by removing incomplete POIs and duplicate question-answer pairs. The other two datasets, **MapQA-Adjacent (MapQA-ADJ)** and **MapQA-Amenities_Around_Specific (MapQA-AME)** (Li et al., 2025), are designed to test spatial relationships. MapQA-ADJ focuses on topological queries (e.g., "What is adjacent to [location]?"), while MapQA-AME centers on proximity-based questions (e.g., "What amenities are within 50m of [location]?"). Detailed statistics for all datasets are provided in Table 1.

Table 1: Overview of Datasets

Statistic	TourismQA-Miami	TourismQA-NYC	MapQA-ADJ	MapQA-AME
#POIs	2,640	9,470	92,415	92,415
#QA Pairs	133	17,448	50	231

Evaluation metrics To evaluate the performance of methods, we employ four commonly used metrics: Precision@k, Recall@k, F1@k, and NDCG@k (Normalized Discounted Cumulative Gain up to rank k), $k \in 1, 3, 5, 10$. Precision@k measures the proportion of the top-k retrieved items that are relevant to the query, Recall@k represents the proportion of all relevant items that appear within the top-k results. F1@k is the harmonic mean of Precision@k and Recall@k. NDCG@k evaluates the quality of the ranking by considering the position of the relevant items. Higher values for all metrics indicate better performance. In all tables, we bold and underline the best score and bold the second-best score.

Models for comparison We use GPT4-Turbo as the LLM for all applicable methods. **GeoLLM** (Manvi et al., 2024b) enrich spatial context by adding spatial information of nearby spatial objects. **Naive RAG** (Lewis et al., 2020) saves all spatial objects' descriptions in a vector database and retrieves the most relevant objects based on vector similarity. **Text embedding** (TE) (Cakaloglu et al., 2020) is a greedy method minimizing the distance between the vector embeddings of the text description of the reference object and the target object. **Sort-by-distance** (SD) (Contractor et al., 2021a) ranks the candidate spatial objects based on their distance to the reference objects in the spatial question. **Spatial-text** (ST) computes the embeddings of the user's question and compares the similarity between the question embedding and the text description embedding of the target object. Additionally, the object's location is encoded as a distance score. The answer is determined based on the average of these scores. **O3-web** uses OpenAI o3 model configured with the web-search tool enabled via the user-location parameter. Due to the

limited access to this model, we only evaluated it on the TourismQA-Miami dataset.

5.2 Spatial-RAG vs. Baselines

The results presented in Tables 2 and 3 demonstrate the superior and consistent performance of our Spatial-RAG framework across all datasets. Specifically, Spatial-RAG achieves a 19.9% improvement in Precision@1 over the best baseline (ST) on TourismQA-NYC and improves NDCG by up to 32.0% on MapQA-ADJ and 52.6% on MapQA-AME. This significant lead is best understood by examining the inherent limitations of the baseline methods. Models like Naive RAG and TE, which rely solely on semantic similarity, often fail by retrieving spatially irrelevant candidates. Conversely, the Sort-by-Distance (SD) method is spatially precise but semantically naive, unable to capture user intent beyond simple proximity. While methods like ST and GeoLLM attempt to combine these signals, they lack a robust mechanism to enforce hard spatial constraints, leading to suboptimal candidate sets. Reasoning-oriented LLMs such as o3 are stronger at open-ended problem solving, but they still lack the specialized spatial indexing and database retrieval capabilities required for reliable geospatial grounding. In contrast, Spatial-RAG's multi-stage architecture directly addresses these flaws. Its initial sparse retrieval stage is crucial, acting as a powerful filter that uses the spatial database to ensure only geographically plausible candidates are considered. This significantly boosts recall—achieving a 95.4% higher Recall@10 than GeoLLM on TourismQA-Miami—by not prematurely dismissing valid options. Subsequently, our hybrid scoring and multi-objective re-ranking intelligently balance spatial and semantic relevance, ensuring the most holistically suitable items are ranked highest. This architectural advantage is why Spatial-RAG consistently excels in ranking-focused metrics like NDCG and Precision, confirming its status as a new state-of-the-art solution for geospatial reasoning. Given the multi-stage nature of the system, we also conducted a comprehensive component-level error analysis on the delivery failures to identify the primary sources of error in Appendix C.

5.3 Ablation Studies

To validate each component's contribution, we conducted ablation studies, with results summarized in Table 4. We assessed the impact of removing following modules: 1) the sparse spatial retriever (w/o sparse spatial), 2) the dense spatial scorer (w/o dense spatial), 3) the dense semantic scorer (w/o dense semantic), 4) the reranker (w/o reranking), 5) the Pareto selection (w/o Pareto). We also evaluated two system-level variations: 6) generating SQL from scratch without templates (Scratch) and 7) removing the entire RAG pipeline (w/o RAG). A study on the influence of different LLMs is detailed in Appendix E. The results affirm our design. The most severe performance drop occurred when the entire RAG pipeline was removed (w/o RAG), causing Precision@1 to plummet by 36.7%. This confirms that a retrieval-augmented strategy is fundamental to solving these complex queries. The second most critical component is the sparse spatial retriever; removing it resulted in a 32.8% decrease in Precision@1, underscoring the necessity of grounding the LLM in a structured spatial database. While the dense components have a smaller impact, their removal still degrades performance, demonstrating the value of capturing nuanced textual semantics. Interestingly, the Scratch variant saw only a minor 4.8% drop in Precision@1, highlighting the LLM's strong intrinsic capabilities, yet our template-guided approach provides an additional layer of reliability and consistency.

5.4 Case Studies

To provide a qualitative validation of our quantitative results, Figure 3 visualizes how Spatial-RAG successfully handles three distinct types of geospatial queries. As illustrated, our model correctly interprets the user's spatial intent in each case: a) a point-based query requiring a search within a radius, b) a polyline-based query for locations along a route, and c) a polygon-based query for places within a named region. This nuanced understanding is a significant advantage over simpler baselines. For instance, faced with the route-based query in b), a traditional method like Sort-by-Distance would likely generate an incorrect buffer zone around a single point, failing to capture the user's actual path. In contrast, Spatial-RAG correctly models the search along a polyline, ensuring a far more relevant initial candidate set. This ability to accurately parse the geometry of the query and apply the correct spatial constraints is precisely why our framework achieves superior recall and precision, as it begins the ranking process with candidates that are not just semantically similar but also geographically plausible.

Table 2: Performance comparison of models on TourismQA-Miami and TourismQA-NYC

Dataset	Method	Precision				Recall				F1				NDCG			
		@1	@3	@5	@10	@1	@3	@5	@10	@1	@3	@5	@10	@1	@3	@5	@10
TourismQA-Miami	GeoLLM	0.3158	0.2719	**0.2368**	**0.2053**	0.0365	0.0945	0.1246	0.1824	0.0591	0.1094	0.1311	0.1623	0.3158	0.3200	0.3268	0.3600
	Naive RAG	0.2973	0.2252	0.2054	0.1973	0.0814	0.1124	0.1491	0.2301	0.1005	0.1099	0.1310	0.1642	0.2973	0.2753	0.2849	0.3403
	TE	0.3421	**0.3070**	0.2316	0.1842	0.0567	**0.1544**	**0.1821**	0.2231	0.0781	**0.1558**	**0.1515**	0.1543	0.3421	0.3766	0.3890	**0.4748**
	SD	0.1053	0.0351	0.0211	0.0105	0.0075	0.0075	0.0075	0.0075	0.0138	0.0119	0.0105	0.0082	0.1053	0.1053	0.1053	0.1053
	ST	**0.3947**	0.1316	0.0789	0.0395	**0.0873**	0.0873	0.0873	0.0873	**0.1115**	0.0730	0.0569	0.0386	**0.3947**	**0.3947**	**0.3947**	0.3947
	O3-web	0.2895	0.2281	0.2000	0.1895	0.0309	0.0863	0.1261	**0.2401**	0.0540	0.1001	0.1246	**0.1791**	0.2895	0.2888	0.3182	0.4595
	Spatial-RAG	**0.5152**	**0.4242**	**0.3758**	**0.2939**	**0.1026**	**0.2037**	**0.2841**	**0.3565**	**0.1454**	**0.2246**	**0.2619**	**0.2656**	**0.5152**	**0.4852**	**0.4983**	**0.5079**
TourismQA-NYC	GeoLLM	0.3650	0.3292	0.3020	0.2725	0.0168	0.0433	0.0614	0.1033	0.0311	0.0688	0.0907	0.1328	0.3650	0.3626	0.3708	0.4248
	Naive RAG	**0.4923**	0.4447	0.4245	0.4181	0.0214	0.0556	0.0889	0.1678	0.0399	0.0932	0.1345	**0.2167**	**0.4923**	0.4551	0.4528	0.5058
	TE	0.2250	0.2433	0.2395	0.2325	0.0101	0.0330	0.0506	0.0934	0.0189	0.0541	0.0767	0.1199	0.2250	0.2873	0.3533	0.5142
	SD	0.2425	0.2492	0.2455	0.2402	0.0103	0.0314	0.0520	0.0945	0.0193	0.0520	0.0758	0.1194	0.2425	0.2694	0.2926	0.3624
	ST	0.4725	**0.4460**	**0.4608**	**0.4323**	**0.0238**	**0.0644**	**0.1026**	**0.1846**	**0.0433**	**0.1026**	**0.1475**	**0.2281**	0.4725	**0.4686**	**0.4745**	**0.5296**
	Spatial-RAG	**0.5665**	**0.4875**	0.4555	0.4251	**0.0274**	0.0611	0.0908	0.1691	**0.0483**	0.0996	0.1384	0.2153	**0.5665**	**0.5174**	**0.5065**	**0.5574**

(a) A "point within 1km radius" query: "... and staying in the waldorf astoria, we are looking for suggestions for new year? we were thinking of maybe booking a meal somewhere near central park, then seeing the new year in @ the park. . . . . .

(b) A "point within polyline" query: ... we move to Hilton New York at 1335 Ave Of Americas between 53 and 54. We have tickets for a Yankee game at 1:05...so we can get subway to Yankee Stadium? ... for dinner that evening - something simple.

(c) A "point within polygon" query: ...I will be staying in SOHO but will be doing things all over downtown so open to going anywhere the subway will take me. Good Italian, Mexican, Tapas etc are always safe options. But like to experience some fun trendy hip places...

Figure 3: Three POI queries with different reference objects.

Table 3: Performance comparison of models on MapQA-ADJ and MapQA-AME.

Dataset	Metric	Methods					
		GeoLLM	Naive RAG	TE	SD	ST	Spatial-RAG
MapQA-ADJ	Precision	0.4558	0.3455	0.4200	0.4600	**0.5600**	**0.5057**
	Recall	**0.6685**	0.6170	0.3750	0.4200	0.5150	**0.7819**
	F1	0.5073	0.4067	0.3880	0.4333	**0.5280**	**0.5625**
	NDCG	**0.6132**	0.5866	0.4200	0.4600	0.5600	**0.7391**
MapQA-AME	Precision	0.3729	0.2452	0.2751	0.4323	**0.5714**	**0.6645**
	Recall	**0.8333**	0.6075	0.2351	0.3923	0.5173	**0.9072**
	F1	0.4744	0.3263	0.2482	0.4054	**0.5346**	**0.7298**
	NDCG	**0.8199**	0.6018	0.2751	0.4323	0.5714	**0.8719**

6 Conclusion

We introduced Spatial-RAG, a framework that bridges the gap between the natural language understanding of LLMs and the computational power of structured spatial databases. By decomposing complex geospatial queries into a multi-stage process of sparse spatial retrieval, hybrid relevance scoring, and multi-objective generation, our approach grounds LLM reasoning in geographic reality. Extensive experiments demonstrate that Spatial-RAG establishes a new state-of-the-art in geospatial question answering, significantly outperforming existing methods.

Table 4: Ablation results on the TourismQA-NYC dataset.

Method	Precision				Recall				F1				NDCG			
	@1	@3	@5	@10	@1	@3	@5	@10	@1	@3	@5	@10	@1	@3	@5	@10
Spatial-RAG	0.5665	0.4875	0.4555	0.4251	0.0274	0.0611	0.0908	0.1691	0.0483	0.0996	0.1384	0.2153	0.5665	0.5174	0.5065	0.5574
w/o sparse spatial	0.3807	0.3545	0.3503	0.3307	0.0147	0.0431	0.0701	0.1305	0.0276	0.0701	0.1058	0.1667	0.3807	0.3608	***0.3570***	***0.3455***
w/o dense spatial	0.5569	0.4954	0.4628	0.4295	0.0263	0.0606	0.0919	0.1679	0.0466	0.0986	0.1392	0.2132	0.5569	0.5240	0.5132	0.5587
w/o dense semantic	0.4986	0.4311	0.4165	0.3806	0.0240	0.0553	0.0844	0.1499	0.0421	0.0883	0.1262	0.1896	0.4986	0.4679	0.4739	0.5219
w/o reranking	0.3932	0.3478	0.3461	0.3350	0.0140	0.0389	0.0628	0.1215	0.0263	0.0642	0.0966	0.1596	0.3932	0.3758	0.3897	0.4550
w/o Pareto	0.3874	0.3797	0.3609	0.3434	0.0164	0.0426	0.0660	0.1231	0.0306	0.0726	0.1042	0.1643	0.3874	0.4056	0.4123	0.4723
Scratch	0.5392	0.4935	0.4706	0.4343	0.0236	0.0535	0.0829	0.1574	0.0435	0.0916	0.1323	0.2105	0.5392	0.5105	0.5042	0.5374
w/o RAG	***0.3584***	***0.3333***	***0.3028***	***0.2644***	***0.0121***	***0.0323***	***0.0484***	***0.0846***	***0.0231***	***0.0568***	***0.0796***	***0.1191***	***0.3584***	***0.3570***	0.3653	0.4186

Acknowledgments

This work was partially supported by the NSF Grant No. 2403312, No. 2414115, No. 2007716,No. 2007976, No. 1942594, No. 1907805, Cisco Faculty Research Award, Sony Research Award, and NIH Grant No. R01AG089806. Gengchen Mai is supported by the NSF under Grant No. 2521631.

7 Limitations

Our method depends on existing spatial databases that provide accurate object footprints. In regions where such data are unavailable, an essential preliminary step would involve collecting spatial information and constructing the corresponding database. The evaluation in this paper also does not systematically cover the rapidly growing set of recent agentic systems, even though preliminary checks suggest that explicit spatial retrieval remains useful in those settings.

References

Tolgahan Cakaloglu, Christian Szegedy, and Xiaowei Xu. 2020. Text embeddings for retrieval from a large knowledge base. In *International Conference on Research Challenges in Information Science*, pages 338–351. Springer.

Wei Chen. 2014. Parameterized spatial sql translation for geographic question answering. In *2014 IEEE international conference on semantic computing*, pages 23–27. IEEE.

Xu Chen, Fangning Ren, Dazhou Yu, Lechen Dong, and Fang Liu. 2025. Autosolvate-agent: An autonomous agent for gpu-accelerated solution-phase quantum chemistry. In *Large Language Models for Scientific and Societal Advances*.

Philipp Christmann and Gerhard Weikum. 2024. Rag-based question answering over heterogeneous data and text. *arXiv preprint arXiv:2412.07420*.

Danish Contractor, Shashank Goel, Mausam, and Parag Singla. 2021a. Joint spatio-textual reasoning for answering tourism questions. In *Proceedings of the Web Conference 2021*, pages 1978–1989.

Danish Contractor, Barun Patra, Parag Singla, and 1 others. 2021b. Constrained bert bilstm crf for understanding multi-sentence entity-seeking questions. *Natural Language Engineering*, 27(1):65–87.

Danish Contractor, Krunal Shah, Aditi Partap, Parag Singla, and 1 others. 2019. Large scale question answering using tourism data. *arXiv preprint arXiv:1909.03527*.

Fahim Faisal and Antonios Anastasopoulos. 2023. Geographic and geopolitical biases of language models. In *Proceedings of the 3rd Workshop on Multi-lingual Representation Learning (MRL)*, pages 139–163.

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6491–6501.

Wes Gurnee and Max Tegmark. 2024. Language models represent space and time. In *The Twelfth International Conference on Learning Representations*.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *arXiv preprint arXiv:2402.07630*.

Yuhan Ji and Song Gao. 2023. Evaluating the effectiveness of large language models in representing textual descriptions of geometry and spatial relations (short paper). In *12th International Conference on Geographic Information Science (GIScience 2023)*, pages 43–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

Sergios-Anestis Kefalidis, Dharmen Punjani, Eleni Tsalapati, Konstantinos Plas, Maria-Aggeliki Pollali, Pierre Maret, and Manolis Koubarakis. 2024.

The question answering system geoqa2 and a new benchmark for its evaluation. *International Journal of Applied Earth Observation and Geoinformation*, 134:104203.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Fangjun Li, David C Hogg, and Anthony G Cohn. 2024. Advancing spatial reasoning in large language models: An in-depth evaluation and enhancement using the stepgame benchmark. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18500–18507.

Zekun Li, Malcolm Grossman, Mihir Kulkarni, Muhao Chen, Yao-Yi Chiang, and 1 others. 2025. Mapqa: Open-domain geospatial question answering on map data. *arXiv preprint arXiv:2503.07871*.

Zekun Li, Wenxuan Zhou, Yao-Yi Chiang, and Muhao Chen. 2023. Geolm: Empowering language models for geospatially grounded language understanding. *arXiv preprint arXiv:2310.14478*.

Bastien Liétard, Mostafa Abdou, and Anders Søgaard. 2021. Do language models know the way to rome? *arXiv preprint arXiv:2109.07971*.

Gengchen Mai, Weiming Huang, Jin Sun, Suhang Song, Deepak Mishra, Ninghao Liu, Song Gao, Tianming Liu, Gao Cong, Yingjie Hu, and 1 others. 2024. On the opportunities and challenges of foundation models for geoai (vision paper). *ACM Transactions on Spatial Algorithms and Systems*.

Gengchen Mai, Krzysztof Janowicz, Rui Zhu, Ling Cai, and Ni Lao. 2021. Geographic question answering: challenges, uniqueness, classification, and future directions. *AGILE: GIScience series*, 2:8.

Rohin Manvi, Samar Khanna, Marshall Burke, David B Lobell, and Stefano Ermon. 2024a. Large language models are geographically biased. In *International Conference on Machine Learning*, pages 34654–34669. PMLR.

Rohin Manvi, Samar Khanna, Gengchen Mai, Marshall Burke, David B Lobell, and Stefano Ermon. 2024b. Geollm: Extracting geospatial knowledge from large language models. In *The Twelfth International Conference on Learning Representations*.

Jonathan Roberts, Timo Lüddecke, Sowmen Das, Kai Han, and Samuel Albanie. 2023. Gpt4geo: How a language model sees the world's geography. *arXiv preprint arXiv:2306.00020*.

Shamane Siriwardhana, Rivindu Weerasekera, Elliott Wen, Tharindu Kaluarachchi, Rajib Rana, and Suranga Nanayakkara. 2023. Improving the domain adaptation of retrieval augmented generation (rag) models for open domain question answering. *Transactions of the Association for Computational Linguistics*, 11:1–17.

Nemin Wu, Qian Cao, Zhangyu Wang, Zeping Liu, Yanlin Qi, Jielu Zhang, Joshua Ni, Xiaobai Yao, Hongxu Ma, Lan Mu, and 1 others. 2024. Torchspatial: A location encoding framework and benchmark for spatial representation learning. *arXiv preprint arXiv:2406.15658*.

Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and Yu Su. 2024. Travelplanner: A benchmark for real-world planning with language agents. In *Forty-first International Conference on Machine Learning*.

Dazhou Yu, Xiaoyun Gong, Yun Li, Meikang Qiu, and Liang Zhao. 2024a. Self-consistent deep geometric learning for heterogeneous multi-source spatial point data prediction. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4001–4011.

Dazhou Yu, Yuntong Hu, Yun Li, and Liang Zhao. 2024b. Polygongnn: Representation learning for polygonal geometries with heterogeneous visibility graph. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4012–4022.

Dazhou Yu, Genpei Zhang, and Liang Zhao. 2025. Polyhedronnet: Representation learning for polyhedra with surface-attributed graph. In *The Thirteenth International Conference on Learning Representations*.

Jiong Yu, Sixing Wu, Jiahao Chen, and Wei Zhou. 2024c. Llms as collaborator: Demands-guided collaborative retrieval-augmented generation for commonsense knowledge-grounded open-domain dialogue systems. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13586–13612.

Yifan Zhang, Zhiyun Wang, Zhengting He, Jingxuan Li, Gengchen Mai, Jianfeng Lin, Cheng Wei, and Wenhao Yu. 2024. Bb-geogpt: A framework for learning a large language model for geographic information science. *Information Processing & Management*, 61(5):103808.

A Dense Semantic Retrieval and Ranking

In the previous section, we derived the spatial candidate set C_s and the spatial relevance score f_s. Now, we focus on obtaining the semantic candidate set C_k and the semantic relevance score f_k .

Given a query q , we define the semantic candidate set $C_k(q)$ as:

$$C_k(q) = \{y \mid c_k(y, q) \leq 0, \forall c_k \in \mathcal{C}_k(q)\}, \quad (7)$$

where:

- $c_k(y, q)$ is a constraint function that filters out spatial objects not satisfying the semantic intent of the query.
- $\mathcal{C}_k(q)$ is the set of all semantic constraints (e.g., topic matching, category relevance).

Each spatial object is associated with textual descriptions, including names, reviews, and additional metadata. However, these descriptions often contain irrelevant or verbose details that may obscure meaningful information. To address this, we use an LLM-based masking function $\mathcal{M}_k$ to remove spatially redundant information and retain only semantically relevant content. The resulting texts are then encoded into a dense embedding space by a text encoder $\mathcal{E}$. Specifically, given a spatial object text description d_t,user query q, the filtered text representation is:

$$v_{t,k} = \mathcal{E}(\mathcal{M}_k(d_t)) \quad v_{q,k} = \mathcal{E}(\mathcal{M}_k(q)). \tag{8}$$

The semantic relevance score is then computed using cosine similarity:

$$f_k = \frac{v_{q,k} \cdot v_{t,k}}{\| v_{q,k} \| \| v_{t,k} \|}. \tag{9}$$

This score quantifies how well the spatial object aligns with the query's semantic intent, irrespective of spatial factors.

B Implementation Details

B.1 Semantic Parsing for Spatial Database Query

For the geometry objects referenced in user queries, Spatial-RAG initially interacts with the spatial database to locate and match the described objects, such as specific points (e.g., a restaurant), roads, or defined areas and subsequently retrieves the pertinent geometrical data. In scenarios where the specified geometrical object does not exist premapped in the database, Spatial-RAG is designed to construct a temporary geometric object. This temporary object serves as a stand-in to facilitate spatial queries based on the user's descriptive input. This approach allows Spatial-RAG to handle dynamic spatial inquiries efficiently, even when direct matches are not immediately found within the existing database entries. By creating temporary geometrical representations, Spatial-RAG ensures that all spatial queries are processed accurately, maintaining the integrity and effectiveness of the system in delivering precise spatial information and responses.

Functionally, the same outcome might be achieved through different means, for example, searching for a restaurant near a street could involve searching within a buffered polyline or creating a polygon enclosing the polyline and searching within it. Such flexibility in the system implies various methods to achieve the same goal. This flexibility, however, poses a challenge if the LLM is tasked with generating a complete query directly, as it might lead to the production of hallucinatory, incorrect, or inexecutable code due to confusion or excessive complexity in interpreting spatial data. By structuring the process such that the LLM first identifies the geometry, then determines the function in a step-by-step manner, we mitigate the risks associated with generating errant queries.

B.2 Dense Retrieval

B.2.1 Text Encoder

We utilize OpenAI's text-embedding-3-small as our text encoder for both spatial and semantic dense retrieval.

B.2.2 Index Construction

We perform an offline preprocessing step where an LLM (GPT-4) summarizes the raw reviews and descriptions of each POI into two distinct descriptions: **Spatial Description**: Extracts spatial features (e.g., "view of the park," "outdoor seating,"). **Semantic Description**: Extracts non-spatial attributes (e.g., "Italian cuisine," "romantic atmosphere," "price range"). These descriptions are independently encoded into vectors and stored in PostgreSQL database to support the hybrid scoring mechanism.

B.2.3 Difference between f_s^{dense} and f_k

While both scores use cosine similarity, they operate on disjoint vector spaces derived from different underlying data summaries. In practice, f_s^{dense} (Dense Spatial Score) captures implicit geographic characteristics that sparse coordinates miss (e.g., "street view"), whereas f_k (Semantic Score) captures user preferences unrelated to location (e.g., "Italian food").

C Error Analysis

To better understand the sources of error in our Spatial-RAG framework, we conducted a detailed component-level error analysis on failed queries

Table 5: Breakdown of failures on the TourismQA-NYC dataset.

Issue Category	Percentage	Description
No relevant POIs	27.12%	The requested POI category does not exist in the area.
Dataset Errors	23.73%	Issues like coordinates falling outside valid city limits.
LLM Polygon Error	33.89%	The LLM misinterprets the boundary of a named region.
User Intent Error	8.47%	The user query contains conflicting requirements.
Ranking Format	6.78%	The LLM fails to generate correct JSON/List format.

Table 6: Token consumption breakdown (Average per query).

Stage	Input tokens	Output tokens
Spatial Extraction	1630.6	197.3
Semantic Extraction	673.6	99.0
Reranking	8334.7	65.6

from the TourismQA-NYC dataset in Table 5. Notably, 50.85% of "failures" (No relevant POIs + Dataset Errors) are actually legitimate "no solution" scenarios or data limitations, meaning the system correctly returned nothing rather than hallucinating a false answer. The largest source of algorithmic error is LLM Polygon Misinterpretation (33.89%). This indicates that the Re-ranker and Semantic Retriever are relatively robust, while the initial Spatial Extraction (specifically region-to-polygon mapping) is the bottleneck. This finding validates our focus on hybrid retrieval, as relying solely on LLM spatial knowledge (like GeoLLM) leads to higher errors in this specific category.

D Computational Resources Details and Cost Analysis

Spatial-RAG is implemented with several large language models (LLMs), including GPT-3.5-Turbo, GPT-4-Turbo, and Qwen2-7B (7B parameters), all used in a training-free setting. GPT-3.5-Turbo and GPT-4-Turbo are accessed via the OpenAI API, with pricing available on the official OpenAI website. The Qwen2-7B model is hosted on our own computing server equipped with four NVIDIA A6000 GPUs.

Each query in the Spatial-RAG pipeline involves three LLM API calls and one spatial database query. We report the token consumption for each stage in Table 6. On average, each LLM call consumes 10638.9/758.2 tokens (input/output). Spatial queries are executed on a PostgreSQL database with the PostGIS extension, with an average latency of 1.2 seconds per query. In contrast, sota LLM methods like OpenAI o3 consumes 70000-230000 tokens for one query, leading to significantly higher costs and latency.

E Influence of LLMs

To evaluate the robustness of our Spatial-RAG framework, we tested its performance when integrated with three different Large Language Models: GPT-4 Turbo, GPT-3.5 Turbo, and Qwen2-7B. The results are presented in Table 7.

The analysis shows that while different LLMs exhibit varied performance characteristics, the Spatial-RAG framework consistently delivers strong results across the board. For instance, on the complex **TourismQA-NYC** dataset, **GPT-4 Turbo** excels in precision (0.5665 Precision@1). In contrast, on the **TourismQA-Miami** dataset, the lighter **GPT-3.5 Turbo** achieves the best overall NDCG scores (0.5440 NDCG@10). Meanwhile, **Qwen** demonstrates a particular strength in recall and ranking diversity, achieving top scores in Recall on the Miami dataset and in higher-k NDCG on the NYC dataset.

This consistent high performance, despite the different strengths of the underlying LLMs, underscores the robustness of our framework. It demonstrates that Spatial-RAG is not overly sensitive to the choice of a specific model and can effectively leverage various LLMs to achieve state-of-the-art results in geospatial question answering.

F Prompt List

F.1 Prompt: Spatial Information Extraction

Extract spatial information from user queries, including object geometry type (Point, Polyline (Route), or Polygon (Region)), region name, distance, and buffer distance.

```
Analyze the following user query and extract spatial information: "{user_query}"

Current location context:
- Number of location points: {location_count}
- Multiple points detected: {is_multi_point}

First, determine the spatial query type based on these rules:
1. For single location point ({location_count == 1}):
   - Use Region-based if query explicitly mentions a region
   - Otherwise, use Point-based

```

Table 7: Ablation study for different LLMs.

Dataset	Method	Precision				Recall				F1				NDCG			
		@1	@3	@5	@10	@1	@3	@5	@10	@1	@3	@5	@10	@1	@3	@5	@10
TourismQA-Miami	**GPT4-Turbo**	**0.5152**	**0.4242**	0.3758	0.2939	0.1026	0.2037	0.2841	0.3565	0.1454	0.2246	0.2619	**0.2656**	**0.5152**	**0.4852**	**0.4983**	**0.5079**
	GPT3.5-Turbo	**0.5455**	**0.4141**	**0.4000**	**0.3152**	**0.1081**	**0.2072**	**0.2972**	**0.3732**	**0.1540**	**0.2262**	**0.2765**	**0.2817**	**0.5455**	**0.4920**	**0.5276**	**0.5440**
	Qwen	0.4091	0.3485	0.3091	0.2364	**0.1114**	**0.2402**	**0.3236**	**0.4167**	**0.1657**	**0.2586**	**0.2815**	0.2508	0.4091	0.3744	0.3960	0.4199
TourismQA-NYC	**GPT4-Turbo**	**0.5665**	**0.4875**	**0.4555**	**0.4251**	**0.0274**	**0.0611**	**0.0908**	**0.1691**	**0.0483**	**0.0996**	**0.1384**	**0.2153**	**0.5665**	**0.5174**	**0.5065**	**0.5574**
	GPT3.5-Turbo	**0.4611**	**0.4352**	0.4144	**0.4098**	**0.0188**	0.0485	0.0751	0.1493	**0.0350**	0.0828	0.1188	**0.2000**	**0.4611**	0.4545	0.4534	0.5097
	Qwen	0.4249	0.4169	**0.4227**	0.4070	0.0185	**0.0507**	**0.0837**	**0.1498**	0.0341	**0.0836**	**0.1267**	0.1967	0.4249	**0.5949**	**0.6334**	**0.6653**

```
2. For exactly two points ({location_count == 2}):
   - Use Route-based if query suggests path/route between points
   - Otherwise, fall back to Point/Region based rules

3. For multiple points ({location_count > 2}):
   - Only use Point-based or Region-based

Query types:
1. Point-based:
   - For "nearby" or "close": 1km in dense areas
   - For "walking distance": 2km
   - For "not too far": 3km

2. Route-based:
   - ONLY available with exactly 2 points
   - For walking routes: 1000m buffer
   - For general routes: 2000m buffer
   - For scenic/exploration: 3000m buffer
   - Consider terms: "route", "path", "between", "from...to", "along"

3. Region-based:
   - ONLY if query explicitly mentions these regions:
   Community/Sub-region names: {', '.join(region_names['nta_names'])}
   Borough names: {', '.join(region_names['boro_names'])}
   - Do NOT infer regions from landmarks

Return in strict JSON format:
{
    "query_type": "point" | "route" | "region",
    "region": "matched region name or null",
    "distance_km": number or null,
    "buffer_distance": number or null,
}
```

F.2 Prompt: Semantic Intent Extraction

Extract semantic intent from user queries, including spatial and nonspatial preference.

```
Analyze the following user query and extract constraints: "{user_query}"

First, determine the main purpose of the query by identifying key terms and context:

Restaurant (R) keywords and contexts:
- Direct terms: "restaurant", "food", "eat", "dining", "meal", "cuisine"
- Food types: "Chinese", "Thai", "Mexican", "Italian", "sushi", etc.
- Meal times: "breakfast", "lunch", "dinner", "brunch"
- Dining related: "menu", "dishes", "chef", "reservation"
- Even if staying at a hotel, if asking about food/dining, it's Restaurant (R)

Hotel (H) keywords and contexts:
- Must be explicitly looking for accommodation
- Direct terms: "hotel", "stay", "accommodation", "room", "book"
- Price per night (e.g., "$200/night")
- Hotel names (e.g., "Hyatt", "Marriott")
- Mentioning a hotel as location reference is NOT H type

Attraction (A) keywords and contexts:
- Direct terms: "visit", "see", "tour", "explore"
- Places: "museum", "park", "gallery", "theater"
- Activities: "sightseeing", "show", "performance"

Important rules:
1. Focus on what the user is ASKING FOR, not what they mention
2. If user mentions staying at a hotel but asks about restaurants, type is R
3. If query is about food/dining/restaurants, type must be R
4. Location references (e.g., "near Hotel X") don't determine type

For each constraint type, extract complete sentences that describe the requirements:


1. Spatial constraints: Where they want to go
   Example: "near Times Square" or "in the Upper West Side area"

2. User constraints: What specific requirements or preferences they have
   Example: "family-friendly restaurant with reasonable prices around $30 per person"
```

```
38
39 Please return strict JSON format without any
       comments:
40 {
41     "type": "R/H/A",
42     "spatial_constraints": "complete
       sentence describing location
       requirements or null",
43     "user_constraints": "complete sentence
       describing user preferences and
       requirements or null"
44 }
```

F.3 Prompt: Result Reranking

Rerank retrieved location results based on user query constraints.

```
1  As a local recommendation expert, please
       rank the following places based on user
       query constraints.
2
3  User Query Constraints:
4  - Spatial Constraints: {query_constraints['
       spatial_constraints']}
5  - User Preferences: {query_constraints['
       user_constraints']}
6
7  Candidate Places:
8  {json.dumps(places, ensure_ascii=False,
       indent=2)}
9
10 Please analyze how well each place matches
       the user constraints and return a sorted
        list of places.
11 Return format should be a JSON array
       containing sorted indices.
12 Only return the index array, e.g., [2,0,1,3]
        means the 3rd place is the best match,
       followed by 1st, 2nd, and 4th places.
13 Note: Must return indices for all places,
       array length should equal input place
       count ({len(places)}).
```

G Additional Case Studies

G.1 Case 1

User Query:

Going to be in Manhattan for a Broadway show this weekend and am looking for a recommendation for a good restaurant in the theatre district that is reasonably priced ($30 to $50 per person), good food and not a tourist trap, would love to avoid the hotel restaurants and experience a real NY experience. Does such a place exist? Someone suggested Carmine's but they are totally booked. (like all kinds of food) Any suggestions would be greatly appreciated - picking one out of the 5000+ listed is a daunting task.

Model Processing Summary:

The system first determines that the **reference object type** is **polygon**, and correctly identifies the **region as Manhattan**. It detects that the **target object type** is a restaurant **point**. Based on these constraints, a SQL query is constructed to retrieve **532 candidate locations** from the database.

The **spatial requirement** is:
"am looking for a recommendation for a good restaurant in the theatre district"

The **semantic requirements** are:
"reasonably priced ($30 to $50 per person), good food and not a tourist trap, avoid hotel restaurants, real NY experience"

The system computes vector similarities between the query and the reviews of all 532 candidates, considering both spatial and semantic relevance. The top 20 results based on combined score are passed to an LLM reranker, and the final **top-1 recommendation** is:

West Bank Cafe, 407 W 42nd St, New York City, NY 10036-6808

Reasoning:

- **Spatial match:** West Bank Cafe is located in the Theatre District of Manhattan, satisfying the user's regional constraint. Its location aligns well with proximity to Broadway shows.
- **Semantic alignment:** The restaurant is described as having reasonable pricing and good food, with a local, non-touristy atmosphere. These attributes align with the user's preferences for affordability and an authentic New York experience.

G.2 Case 2

User Query:

Going to NY in May, it is my gf's birthday on our last night – and I would be looking for a nice restaurant to spoil her :)! We are staying in Times Square – so would like somewhere close by! I would like either an American or Italian cuisine! Any advice would be much appreciated.

Model Processing Summary:

The model identifies the query as a **point** spatial search with a radius of **1.0 km** around **Times Square**. The **target object type** is recognized as a restaurant **point**. Based on these criteria, a SQL query returns **119 candidate locations** from the database.

The **spatial requirement** is:
"We are staying in Times Square – so would like somewhere close by!"

The **semantic requirements** are:
"it is my gf's birthday on our last night",
"I would like either an American or Italian cuisine!"

The system computes vector similarities based on spatial and semantic constraints. The top 20 candidates are re-ranked by an LLM reranker, and the final **top-1 recommendation** is:

Pasta Lovers, 142 W 49th St, New York City, NY 10019-6802

Reasoning:

- **Spatial match:** Pasta Lovers is located within 1 km of Times Square, satisfying the proximity requirement effectively.
- **Semantic alignment:** It offers Italian cuisine, aligning with the user's preference for either American or Italian food. The atmosphere and reviews suggest a cozy dining experience appropriate for a date, though it is not explicitly described as suitable for a birthday celebration.

G.3 Case 3

User Query:
Visiting NYC for the Marathon this weekend. Staying in apartment in Hell's Kitchen area (around 51st and 12th). Anyone have good recommendations for a great breakfast restaurant within an 8–10 block radius? Not worried about eating healthy the day before—looking for filling, traditional American-style breakfast! Also looking for ideas for a good filling dinner for Sunday evening after the race. We're not looking for fine dining, just something good in the area for myself, wife and 16 yo daughter. Thanks in advance for any good recommendations.

Model Processing Summary:
The model identifies the query as a **point** spatial query with a radius of **1.5 km**, derived from the "8–10 block" constraint. The **target object type** is restaurant **point**, and the approximate location is Hell's Kitchen near 51st St and 12th Ave. A SQL query retrieves **140 candidates** from the database based on this spatial condition.

The **spatial requirement** is:
"Staying in apartment in Hell's Kitchen area (around 51st and 12th), breakfast restaurant within an 8–10 block radius"

The **semantic requirements** include:
"traditional American-style breakfast", *"filling dinner after the race (Sunday evening), just something good in the area for myself, wife and 16 yo daughter."*

The system calculates vector similarity scores based on spatial and semantic requirements, and selects the top 20 results. These are re-ranked by an LLM reranker, and the final **top-1 recommendation** is:

Galaxy Diner, 665 9th Ave, New York City, NY 10036-3623

Reasoning:

- **Spatial match:** Galaxy Diner is located within a reasonable walking distance of the user's location, satisfying the "8–10 block radius" constraint in Hell's Kitchen.
- **Semantic alignment:** The diner offers traditional American breakfasts like pancakes, eggs, and bacon, aligning with the user's request for filling, non-healthy food before the race. It also accommodates a casual family atmosphere.

G.4 Case 4

User Query:
We are staying in Midtown, so figured it would be easier to go somewhere close by, but I'm not opposed to somewhere a little further away. Looking for a restaurant where we can go to dinner and dress up but not have it be a crazy price. I understand that a "dress up" place isn't usually cheap, but something on the lower end of the "dressy" price scale would be great :) Thanks!

Model Processing Summary:
The model correctly identifies the **reference object type** as **polygon** (**Midtown**). A SQL query retrieves **97 candidates** from the database based on this spatial condition.

The **spatial requirement** is:
"Staying in Midtown, somewhere close by, but not opposed to somewhere a little further away."

The **semantic requirement** is:
"restaurant where we can go to dinner and dress up but not have it be a crazy price, lower end of the 'dressy' price scale."

The model calculates spatial and user constraint similarities, ranks the top 20 results, and applies an LLM reranker. The final **top-1 recommendation** is:

Nocello, 257 W 55th St, New York City, NY 10019-5232

Reasoning:

(a) (b)

Figure 4: (a) Case 1; (b) Case 2

- **Spatial match:** Nocello is within a reasonable walking distance of Midtown, fulfilling the requirement of being nearby but not necessarily adjacent. It's described as a short walk from Restaurant Row and near Broadway, aligning well with the user's open spatial boundary.

- **Semantic alignment:** The restaurant is described as offering excellent food at a reasonable price, making it suitable for a "dress-up" dinner without the high-end cost. This aligns with the user's goal of finding something elegant but affordable.

G.5 Case 5

User Query:
Where are some good places to eat breakfast? We are staying in the southern tip of Manhattan near Battery Park, but it doesn't have to be contained to just that area. We are very open to ideas, but are going to be avoiding McDonalds, BK, etc. Could be larger restaurants but also would like to visit a few small places and would like to be able to sit and eat outside.

Model Processing Summary:
The model correctly identifies the **reference object type** as **polygon** (**battery park city-lower manhattan**). A SQL query retrieves **14 candidates** from the database based on this spatial condition.

The **spatial requirement** is:
"staying in the southern tip of Manhattan near Battery Park, but it doesn't have to be contained to just that area."

The **semantic requirements** are:
"avoiding McDonalds, BK, etc. Could be larger restaurants but also would like to visit a few small places and would like to be able to sit and eat outside."

The model ranks the top 10 by spatial and user relevance, and performs LLM reranking. The final **top-1 recommendation** is:

Stone Street Tavern, 52 Stone St, New York City, NY 10004-2604

Reasoning:

- **Spatial match:** The restaurant is located a short walk from Battery Park in the financial district, consistent with the user's desire to explore areas nearby but not strictly limited to Battery Park. The cobblestone street setting and access to outdoor space align well with the user's spatial intent.

- **Semantic alignment:** Stone Street Tavern offers outdoor seating and avoids fast-food

(a) (b)

Figure 5: (a) Case 3; (b) Case 4

chains. It provides a casual and local dining experience with bench-style outdoor tables, fitting the user's interest in both large and small sit-down places for breakfast.

G.6 Case 6

User Query:

We have tickets for the Saturday performance of War Horse at the Vivian Beaumont Theater at Lincoln Center. I would appreciate a recommendation for a reasonable pre-theater dinner.

Model Processing Summary:

The model correctly identifies the **reference object type** as **point** query, as the user provides a specific point of interest (**Vivian Beaumont Theater**) without mentioning a formal region. A walking distance of 2.0 km is assumed for pre-theater dining. A SQL query retrieves **255 candidates** from the database based on this spatial condition.

The **spatial requirement** is:

"Vivian Beaumont Theater at Lincoln Center" – implying a location within walking distance

The **semantic requirements** are:

"reasonable pre-theater dinner" — suggesting affordability and suitable timing for a theater schedule.

The model reranks the top 20 via LLM reranking. The final **top-1 recommendation** is:

Bar Boulud, 1900 Broadway, New York City, NY 10023-7004

Reasoning:

- **Spatial match:** Bar Boulud is located directly across from Lincoln Center, fulfilling the proximity constraint perfectly for a pre-theater dinner.
- **Semantic alignment:** The restaurant is known for accommodating theatergoers and offers timing suitable for pre-show dining. However, user reviews mention that while the food and service are excellent, prices may be higher than what the user considers "reasonable".

G.7 Case 7

User Query:

I want to walk the Highline on my forthcoming trip to NY, and wanted some recommendations for a good spot for a sit down lunch somewhere close to one of the exits off the Highline. Will be starting north to south. Don't mind what type of food, just nice atmosphere required. Have looked at MenuPages but so many restaurants not sure if anyone has any particular favourites?

(a) (b)

Figure 6: (a) Case 5; (b) Case 6

Model Processing Summary:
The model correctly identifies the **reference object type** as **polyline**. This is inferred from the user's mention of walking from the north to south along the Highline, with interest in locations near the path (Highline exits). Two coordinates representing the start and end of the route are provided. The buffer distance is set to 500 meters.

The **spatial requirement** is:
"somewhere close to one of the exits off the Highline."

The **semantic requirement** is:
"sit down lunch, nice atmosphere required."

The system extracts 52 candidate restaurants along the route, ranks them by spatial and user constraints, and performs LLM reranking on the top 20. The final **top-1 recommendation** is:

Barbuto, 775 Washington St, New York City, NY 10014-1748

Reasoning:

- **Spatial match:** Barbuto is located directly off the Highline and near one of its southern exits, which aligns precisely with the user's request for proximity to the walking path.
- **Semantic alignment:** The restaurant offers a sit-down experience with a "fancy but cozy" atmosphere, partially fulfilling the user's request for a "nice atmosphere." However, some reviews describe the food as average, which may not fully satisfy quality expectations.

G.8 Case 8

User Query:
We arrive Sat night at 10:30 (unfortunately AA changed our flight from an earlier one). We are spending 1 night at RCA at 142 W 49th between 6th and 7th. Sunday we move to Hilton New York at 1335 Ave Of Americas between 53 and 54. We have tickets for a Yankee game at 1:05. (We will also be going to Dizzy's Coca Cola at 9PM.) Could you guys give me some tips on the best way to handle the hectic morning? Leave our bags at RCA? Move them to Hilton for storage so we can get subway to Yankee Stadium? Can you give me some recommendations for dinner that evening - something simple. Thanks!

Model Processing Summary:
The model correctly identifies the **reference object type** as **polyline**, given that the user describes movement between two locations: Hilton New York and Yankee Stadium. The user's intention to find a dinner spot along that route aligns with this classification. The buffer distance is set to **500**

(a) (b)

Figure 7: (a) Case 7; (b) Case 8

meters.

"The **spatial requirement** is:
Implied need for proximity to Hilton New York or Yankee Stadium for convenience."

The **semantic requirements** are:
"We will also be going to Dizzy's Coca Cola at 9PM." and "recommendations for dinner that evening - something simple."

The system identifies 50 restaurant candidates within the route buffer, computes user and temporal similarity, and applies LLM reranking on the top 20. The final **top-1 recommendation** is:

Good Enough to Eat, 520 Columbus Ave, Frnt A, New York City, NY 10024-3404

Reasoning:

- **Spatial match:** This place is within the buffer range, and the restaurant is reasonably accessible and could fit into the evening schedule before Dizzy's Coca Cola.
- **Semantic alignment:** The user requested a simple dinner. The reviews highlight casual meals like pancakes, BLT omelets, and quick service, matching the preference for something simple—though most mentions are for breakfast/brunch.