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

Dazhou Yu [*1] **Riyang Bao** [*1] **Gengchen Mai** [2] **Liang Zhao** [1]

Abstract

Spatial reasoning remains challenging for Large Language Models (LLMs), which struggle with spatial data retrieval and reasoning. We propose Spatial Retrieval-Augmented Generation (Spatial-RAG), a framework that extends RAG to spatial tasks by integrating sparse spatial retrieval (spatial databases) and dense semantic retrieval (LLM-based similarity). A multi-objective ranking strategy balances spatial constraints and semantic relevance, while an LLM-guided generator ensures coherent responses. Experiments on a real-world tourism dataset show that Spatial-RAG significantly improves spatial question answering, bridging the gap between LLMs and spatial intelligence.

Figure 1. An example of a real-world spatial reasoning question with nearby spatial objects. Areas that satisfy the spatial constraint are highlighted in blue.

1. Introduction

Spatial questioning has long been a fundamental domain (Obe & Hsu, 2021), which adeptly handles a variety of spatial questions, from identifying closest neighbors to detecting line-polygon intersections. However, traditional spatial questioning systems rely on specialized spatial query languages, which are vastly different from human language, making them inaccessible to layman users. More critically, these systems lack the ability to infer complex spatial and semantic relationships from the nuanced, context-rich nature of human text, limiting their applicability to real-world question-answering scenarios. Recent advancements in Large Language Models (LLMs) have transformed many fields within machine learning (ML), particularly in understanding and generating human-like text. This progress has inspired early efforts to bridge the gap between spatial questioning and natural language by directly eliciting spatial knowledge from LLMs. These efforts span a range of applications, including geographic encyclopedic question answering (Mai et al., 2021; 2020; Chen, 2014; Scheider et al., 2021; Contractor et al., 2021a), geolocalization (Zhou et al., 2024; Haas et al., 2024), and automated High-definition map generation (Li et al., 2022; Liu et al., 2023). Despite these advancements, recent studies indicate that LLMs exhibit significant limitations in spatial reasoning (Mai et al., 2024; Roberts et al., 2023), and struggle with even basic spatial tasks, such as geoparsing (Mai et al., 2024) and understanding relative spatial relationships (Majic et al., 2024). This gap becomes particularly evident when handling real-world spatial reasoning tasks, such as the one illustrated in Figure 1, where the system must recommend a restaurant along a specific route within a predefined area.

This paper tackles a novel challenge: augmenting LLMs with spatial reasoning capabilities. Specifically, we extend Retrieval-Augmented Generation (RAG) into spatial 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

[*]Equal contribution [1]Department of Computer Science, Emory University, Atlanta, United States [2]Department of Geography and the Environment, University of Texas at Austin, Austin, United States. Correspondence to: Gengchen Mai <gengchen.mai@austin.utexas.edu>, Liang Zhao <liang.zhao@emory.edu>.

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.

Spatial questioning is intrinsically challenging, requiring a synergistic combination of text-based inference and spatial computations. For example, as illustrated in Figure 1, answering the question requires LLM to elicit and formulate the user's textual request into the problem of "finding points near the polyline" and solve it based on a spatial map (database). Then, it also requires inferring user preference 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. To achieve this, we introduce *Spatial Retrieval-Augmented Generation (Spatial-RAG)*, a novel framework that unifies text-guided spatial retrieval with spatially aware text generation. Specifically, to identify spatially relevant candidate answers, we propose a novel spatial hybrid retrieval module combining 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 spatial 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 sparse retrieval (SQL-based structured queries) with dense retrieval (LLM-powered semantic matching). This dual approach ensures that retrieved results align both spatially and semantically with the user's query, significantly improving retrieval accuracy in spatial contexts.

• **Multi-objective guided spatial text generator:** To handle both spatial constraints and textual reasoning in the spatial question-answering task, we introduce a multi-objective optimization framework that dynamically balances trade-offs between spatial and semantic relevance. This ensures that generated responses are both geometrically accurate and linguistically coherent.

• **Real-World Evaluation:** We evaluated our method on a real-world dataset collected from a Tourism website that has user questions and reviews about different spatial entities. Experiments on this dataset reveal the ability to handle real-world spatial reasoning questions.

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

2. Related Work

2.1. Retrieval Augmented Generation

Retrieval-Augmented Generation (RAG) is a hybrid approach that integrates retrieval systems and generative models to enhance factual accuracy and contextual relevance in natural language generation (Fan et al., 2024). Unlike conventional language models that rely solely on parametric memory, RAG dynamically retrieves relevant external knowledge before generating a response. One of the foundational works in RAG is Lewis et al. (2020), where a retrieval module fetches relevant passages from a large-scale knowledge corpus (e.g., Wikipedia), which are then fused with the question context to generate a more informed response. This technique has proven particularly effective in open-domain question answering (QA), fact verification, and context-aware text generation. RAG systems have expanded beyond text and document retrieval to incorporate a wide variety of data types (He et al., 2024) — tables, graphs, charts, and diagrams. While RAG has been widely explored, its application in spatial reasoning question answering remains an unexplored research area. Existing studies have primarily focused on knowledge-grounded dialogues (Yu et al., 2024) but often struggle with integrating spatial computation into the question-answering process effectively.

2.2. Spatial Questions

Spatial questions in domain-specific applications can generally be categorized into two distinct types: **1) Textual Knowledge-based Spatial Questions** These are spatial questions that can be answered by traditional QA methods without the need for spatial computation and reasoning (Liétard et al., 2021). For example, the question *"What is the population of Los Angeles city?"* falls under this category. Despite their spatial context, these questions are essentially text-based and, hence, can be effectively addressed using traditional Retrieval-Augmented Generation (RAG) methods (Christmann & Weikum, 2024). **2) Spatial Reasoning Questions** This category encapsulates spatial questions that demand a model's capability to comprehend and reason with spatial data and spatial relationships. A common example is a model being presented with textual information describing the spatial relationships among multiple objects (Li et al., 2024). An example question could be, *"What is the position of object A relative to object B?"*, where objects *A* and *B* are locations or entities specified on the map. Resolving such queries requires a profound understanding of spatial concepts and robust reasoning skills, which largely depend on the model's training to handle spatial data. Several studies (Mai et al., 2024; Roberts et al., 2023) have investigated the capacity of LLMs to understand spatial concepts, yet these models often struggle with accurate reasoning even after fine-tuning. Other research (Li et al., 2023) has attempted to enhance this ability by converting geolocation coordinates into addresses to enrich the semantic context. However, these improvements tend to be marginal and are mostly limited to straightforward reasoning tasks like describing positions. Moreover, many existing methods rely on predefined sets of actions tailored to specific tasks.

3. Problem Formulation

In this study, our primary focus is **Spatial Reasoning Questions**. We formulate the problem as follows: Given a question q, the system aims to generate an answer y,

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

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.

To solve this problem, there are three questions to answer: 1) How to acquire the spatial candidate set C_s? 2) How to synergize the spatial and texts by evaluating $f_s(q,y)$? 3) How to trade off the spatial and semantic aspects?

4. Methodology

4.1. Overview

Our proposed framework, Spatial Retrieval-Augmented Generation (Spatial-RAG), is illustrated in Figure 2. Spatial-RAG consists of three key stages: First, to construct the spatial candidate set C_s, the system must precisely define spatial constraints and then retrieve spatial objects that satisfy them. As depicted in Figure 2 (Sparse Spatial Retrieval), we achieve this by parsing the input natural language questions into a spatial SQL query, which will be executed on the spatial database to efficiently retrieve relevant spatial objects from the database. This process is detailed in Section 4.2. Second, to effectively compute spatial relevance $f_s(q,y)$ while integrating textual information, we propose a hybrid spatial retrieval scheme. As shown in Figure 2 (Dense Retrieval Module), this method combines sparse spatial relevance scores from the database with dense semantic similarity scores from text embeddings. This enables the system to rank retrieved spatial objects based on their spatial relevance to the input question, as detailed in Section 4.3. Third, given both spatial and semantic constraints, we formulate a multi-objective optimization problem to balance these factors. The system computes the Pareto front of candidate answers, and the LLM dynamically trades off among these solutions to generate an optimal response. This step is covered in Section 4.4.

4.2. Sparse Spatial Retrieval

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

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

where $c_s(y,q)$ represents a constraint function that encodes a spatial condition (e.g., topological, directional, or distance-based constraints), $\mathcal{C}_s(q)$ is the set of all spatial constraints associated with the 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:

$$c_s(y,q) = \mathrm{d}(y,l_q) - \epsilon \leq 0. \tag{3}$$

This formulation ensures that only spatially valid answers are included in $C_s(q)$.

Addressing spatial constraints requires executing a well-defined spatial SQL query within a spatial database. 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:

$$Q_s = \mathcal{F}_s(G_r, G_t, \epsilon) \tag{4}$$

where $\mathcal{F}_s$ is the spatial query function that determines the relationship between objects. G_r represents the set of reference objects extracted from the question. G_t represents the set of target objects that are potential answers. ϵ is the set of numerical parameters and spatial relationships governing the spatial constraint (e.g., distance threshold, topological relations).

Given the diversity and potentially complex nature of these constraints, Large Language Models (LLMs) often struggle to directly construct a complete and executable spatial query from user input. To bridge this gap, we structure the spatial query incrementally, allowing the LLM to systematically populate the required components.

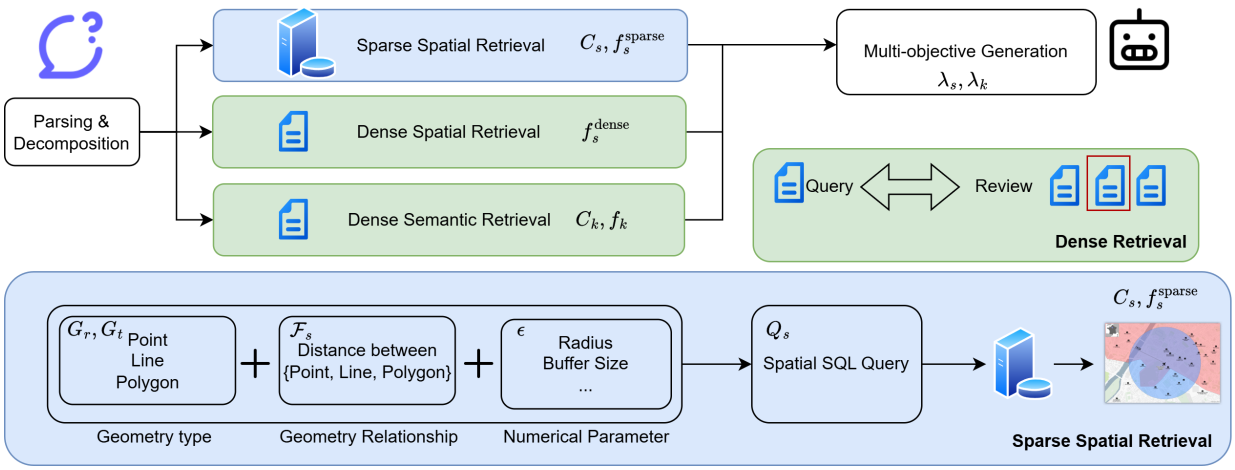


Figure 2. Illustration of the proposed Spatial-RAG framework.

Our approach follows three key steps: 1) **Geometry Recognition:** Identify and extract the reference spatial objects G_r and candidate target spatial objects G_t from the user's input and extract their spatial footprints – geometries. 2) **Query Function Selection:** Determine the appropriate spatial function $\mathcal{F}_s$ based on the intended spatial relationship (e.g., containment, proximity). 3) **Parameter Estimation:** Assign numerical constraints ϵ to ensure precise spatial filtering (e.g., buffer radius).

By formalizing this structured process, we enhance the LLM's ability to generate accurate and executable spatial SQL queries. This, in turn, improves the system's capability to handle complex spatial reasoning questions effectively.

4.2.1. Geometry Recognition

In spatial reasoning tasks, accurately identifying spatial objects and extracting their spatial footprints (i.e., geometries) are essential for parsing questions to spatial queries. Spatial footprints of spatial objects, denoted as $g \in \mathcal{G}$, can generally be categorized into three fundamental types: points, polylines, and polygons. Formally, we define these categories as follows:

- **Point:** $\mathcal{G}_{\text{point}} = \{g \mid g \in \mathbb{R}^2, \dim(g) = 0\}$ This category includes single points and multipoints, representing locations with negligible area. Examples include stop signs, address points, 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 linear 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 spatial query depends on the types of spatial footprints of objects involved. For simpler queries, such as "finding the nearest bus stop from a given location", only point geometries are required, and the spatial candidate set is

$$C_s = \{g | g \in \mathcal{G}_{\text{point}}, d(g, g_{\text{point}}) < \epsilon\} \tag{5}$$

where $g_{\text{point}} \subseteq \mathcal{G}_{\text{point}}$ represents a point object (e.g., given location), ϵ is the distance threshold. For more complex queries, such as "I will walk from home to the university campus along 7th Street and Jones Street; please recommend a café where I can buy breakfast on my walk.", multiple geometry types must be considered, and the spatial candidate set is

$$C_s = \{g | g \in \mathcal{G}_{\text{point}}, g \in B(g_{\text{polyline}}, \epsilon) \cup g_{\text{polygon}}\} \tag{6}$$

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:

$$\begin{cases} \{g_t \in G_t \mid \exists g_r \in G_r, \mathrm{d}(g_r, g_t) \leq \epsilon\}, & \text{if } \mathrm{d}(g_r, g_t) > 0, \\ \{g_t \in G_t \mid \exists g_r \in G_r, g_r \cap g_t \neq \emptyset\}, & \text{if } \mathrm{d}(g_r, g_t) = 0. \end{cases} \tag{7}$$

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 4, 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 Objects Ranking

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 spatial dense retrieval based on text similarity between the question and the spatial descriptions of candidate objects. Formally, we define:

$$f_s = \lambda_s f_s^{\text{sparse}} + \lambda_d f_s^{\text{dense}}, \tag{8}$$

where λ_s 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 using explicit spatial relationships. The score is determined by the spatial query function $\mathcal{F}_s$, which computes the distance between reference and target objects. Formally, we define:

$$f_s^{\text{sparse}} = \begin{cases} \dfrac{1}{1 + d(g_r, g_t)}, & \text{if } g_r \cap g_t = \emptyset \\ 1, & \text{if } g_r \cap g_t \neq \emptyset \end{cases} \tag{9}$$

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

Unlike sparse 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 compare them with the descriptions of candidate objects.

Extracting Spatial Requirements Given a user query q and a set of text descriptions d_t for spatial objects G_t, we extract the relevant spatial content via an attention-based masking function:

$$v_{q,s} = \mathcal{E}(\mathcal{M}_s(q)), \quad v_{t,s} = \mathcal{E}(\mathcal{M}_s(d_t)), \tag{10}$$

where $v_{q,s}$ and $v_{t,s}$ are dense vector representations of spatial features, and $\mathcal{M}$ is the extraction function mapping input text to a spatial related text, $\mathcal{E}$ is the text encoder.

Ranking via Cosine Similarity The relevance score is computed via cosine similarity:

$$f_s^{\text{dense}} = \frac{v_{q,s} \cdot v_{t,s}}{\| v_{q,s} \| \| v_{t,s} \|}. \tag{11}$$

4.3.3. Hybrid Ranking as a Generalized Model

We can demonstrate that hybrid ranking generalizes both sparse and dense ranking approaches:

- **Sparse-Only Case:** If $\lambda_d = 0$, then $f_s = \lambda_s f_s^{\text{sparse}}$, reducing to a purely distance-based ranking.
- **Dense-Only Case:** If $\lambda_s = 0$, then $f_s = \lambda_d f_s^{\text{dense}}$, reducing to a purely semantic-based ranking.
- **Hybrid Case (General):** If both weights are nonzero, hybrid ranking benefits from both explicit spatial constraints and implicit semantic relevance, leading to a more comprehensive ranking mechanism.

This formulation ensures that hybrid ranking outperforms any single-ranking approach by capturing both spatial proximity and semantic alignment.

4.4. Multi-objective Generation

The semantic candidate set C_k and the semantic relevance score f_k are calculated based on dense vector similarity, we put the details in appendix A. After all the scores and candidate sets are acquired, the problem becomes a multi-objective optimization problem since each perspective (spatial and semantic) contributes independently.

4.4.1. PARETO FRONT COMPUTATION

Given the spatial and semantic relevance scores, our goal is to identify the Pareto-optimal candidates that achieve the best trade-off between these objectives. A candidate y is Pareto-optimal if no other candidate dominates it in both spatial and semantic relevance. Formally, the Pareto front $P(q)$ is defined as:

$$\begin{aligned} P(q) = \{ y \in C_s \cap C_k \mid \nexists\, y' \in C_s \cap C_k, \\ f_s(q, y') \geq f_s(q, y) \text{ and } f_k(q, y') \geq f_k(q, y), \\ \text{with at least one strict inequality} \}. \end{aligned} \tag{12}$$

This ensures that each candidate in $P(q)$ is non-dominated, meaning no other candidate is strictly better in both spatial and semantic relevance.

4.4.2. LLM-BASED TRADE-OFF DECISION

Once the Pareto front $P(q)$ is determined, we use an LLM to dynamically balance the trade-offs between spatial constraints and semantic preferences based on the context of the user query. Specifically, the LLM receives the user query, sparse spatial relevance scores, and spatial object descriptions as input:

$$I = \{ q, (f_s^{\text{sparse}}(q, y)\, , d_y)\ \ , \forall y \in P(q) \}\, . \tag{13}$$

A dynamic weighting function $\lambda_s, \lambda_k = h(I)$ based on contextual information is extracted from the input, adjusting the importance of spatial vs. semantic relevance, where h is a learned function capturing query-specific trade-offs.

The top-ranked candidate y^* is selected by LLM:

$$y^* = \arg \max_{y \in P(q)} \lambda_s^T f_s(q, y) + \lambda_k^T f_k(q, y), \tag{14}$$

and the LLM generates a natural language response.

The system adapts to different query contexts instead of using a fixed weighting scheme. By structuring decision-making into discrete steps (candidate filtering $\rightarrow$ Pareto selection $\rightarrow$ trade-off balancing $\rightarrow$ response generation), the LLM avoids generating infeasible or illogical results. This structured approach maximizes accuracy and usability, ensuring that the system's final response aligns closely with the original user intent.

5. Experiment

5.1. Experiment Setting

5.1.1. DATASETS

TourismQA (Contractor et al., 2019) dataset contains user questions crawled from TripAdvisor posts from 50 cities around the world. Reviews of restaurants, attractions, and hotels for each city are crawled from travel forums and hotel booking websites. We select two two popular tourist cities, New York City and Miami, to evaluate the performance of different methods. For New York City, the original dataset contains information on 9,470 Points of Interest (POIs), with a total of 17,448 QA pairs. For Miami, the dataset includes 2,640 POIs and 133 QA pairs. Our preprocessing steps on the dataset include removing POIs with empty review information and eliminating duplicate QA pairs.

5.1.2. EVALUATION METRICS

We assess the experimental results from multiple dimensions: 1) Delivery Rate: Following (Xie et al., 2024), this metric assesses whether the method can deliver a result successfully. 2) Spatial Sparse Pass Rate: This metric evaluates whether the parsed spatial query is correct. 3) Spatial Dense Pass Rate: This metric evaluates whether the answer satisfies the spatial-related semantic constraints in the question. 4) Semantic Pass Rate: This metric evaluates whether the answer meets the semantic constraints in the question. We use GPT-4o to evaluate the last three metrics.

5.1.3. MODELS FOR COMPARISON

We used GPT-3.5-Turbo and GPT-4-Turbo as the LLM in our framework to assess the impact of LLM capability on performance and compared it against the following four methods: **Sort-by-distance** (SD) (Contractor et al., 2021b): This method ranks the candidate spatial objects based on their distance to the reference objects in the spatial question. **Text embedding** (TE) (Cakaloglu et al., 2020): This greedy method minimizes the distance between the vector embeddings of the text description of the reference object and the target object. **Spatial-text** (ST): This approach 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 then determined based on the weighted average of these scores. **Naive RAG** (Lewis et al., 2020): This method saves all spatial objects' descriptions in a vector database and retrieves the most relevant objects based on vector similarity. **GeoLLM** (Manvi et al., 2024): This method encodes the spatial objects to address and enrich its context by adding spatial information of nearby spatial objects.

5.2. Main Results

The main results are summarized in 1 For the NYC dataset, Spatial-RAG demonstrates minimal differences in Delivery Rate when deployed with different LLMs. Approximately 86.1% of the questions are successfully processed. The failure cases where the framework is unable to generate valid recommendations can be classified into two main categories:

City	Model	Delivery Rate	Spatial Sparse Pass Rate	Spatial Dense Pass Rate	Semantic Pass Rate
NYC	SD	**100**	-	57.1	32.1
	TE	99.6	-	48.6	54.6
	ST	**100**	-	53.6	**55.9**
	Naive RAG	99.8	-	52.0	54.5
	GeoLLM	99.8	-	69.5	42.8
	Spatial-RAG (GPT-3.5-Turbo)	87.2	**67.0**	64.4	47.4
	Spatial-RAG (GPT-4-Turbo)	86.1	65.0	**71.6**	50.1
Miami	SD	**100**	-	36.8	22.4
	TE	**100**	-	28.9	39.5
	ST	**100**	-	26.3	42.1
	Naive RAG	**100**	-	31.6	40.8
	GeoLLM	**100**	-	52.6	36.8
	Spatial-RAG (GPT-3.5-Turbo)	86.8	75.8	**57.6**	43.9
	Spatial-RAG (GPT-4-Turbo)	86.8	**81.8**	51.5	**45.5**

Table 1. Performance comparison of models in New York City (NYC) and Miami. The framework is deployed on GPT-4-Turbo and GPT-3.5-Turbo and compared its performance with several baseline models, including SD, TE, ST, Naive RAG, and GeoLLM.

Model	Delivery Rate	Spatial Sparse Pass Rate	Spatial Dense Pass Rate	Semantic Pass Rate
Spatial-RAG (GPT-4-Turbo)	86.1	65.0	71.6	50.1
w/o sparse spatial	**98.9**	-	53.3	**78.4**
w/o dense spatial	89.3	61.5	68.4	49.6
w/o dense semantic	85.9	**72.8**	**75.9**	34.8

Table 2. Different modules of the proposed framework are removed for ablation study.

Around 12.9% of the questions fail to retrieve any spatial objects from the spatial database, either due to polygon recognition errors or because the specified region in the SQL query lacks relevant spatial objects. The remaining 0.9% of cases stem from the LLM's inability to correctly re-rank the retrieved results during the reranking process.

Additionally, the performance gap of our framework based on two LLMs in generating spatial SQL queries is insignificant. However, GPT-4-Turbo outperforms GPT-3.5-Turbo by approximately 7% in Spatial Dense Pass Rate and also demonstrates a slight advantage in Semantic Pass Rate, and both models perform slightly below the optimal values (by 5% and 8%, respectively). This is due to the trade-off made when jointly considering both spatial proximity and the user's personal preferences.

Since the SD method returns the nearest spatial object, it achieves a higher spatial dense retrieval score than the TE, ST, and Naive RAG methods. However, its exclusive reliance on distance results in inferior performance on other metrics. For GeoLLM, as it considers only the names of spatial objects and their distances, it also achieves competitive performance in Spatial Dense Pass Rate. In contrast, both the TE and ST methods consider semantic contexts, leading to superior performance in this aspect compared to other models. The Naive RAG and ST methods jointly optimize for both spatial dense retrieval and user semantic dense retrieval, resulting in comparable performance between the two models.

For Miami, which differs significantly from NYC in terms of area and geography, Spatial-RAG also demonstrates strong performance. The performance of the baseline models almost remains consistent with the patterns observed in the previous analysis.

5.3. Ablation Study

We conducted three ablation studies to assess the importance of different modules in our framework. We remove the sparse spatial module, dense spatial module, and dense semantic module, respectively, and denote the ablation versions as w/o sparse spatial, w/o dense spatial, and w/o dense semantic. The results are summarized in Table 2. It's found that there is a significant increase in Delivery Rate after removing the sparse spatial module, as this eliminates the need to determine the geometry type. In this case, the model relies solely on semantic information, resulting in a lower spatial score but an increased emphasis on semantic relevance. Similarly, when the dense semantic module is removed, the Spatial Dense Pass Rate reaches its highest value, as the model depends entirely on spatial modules for retrieval. However, this also leads to a lower Semantic Pass

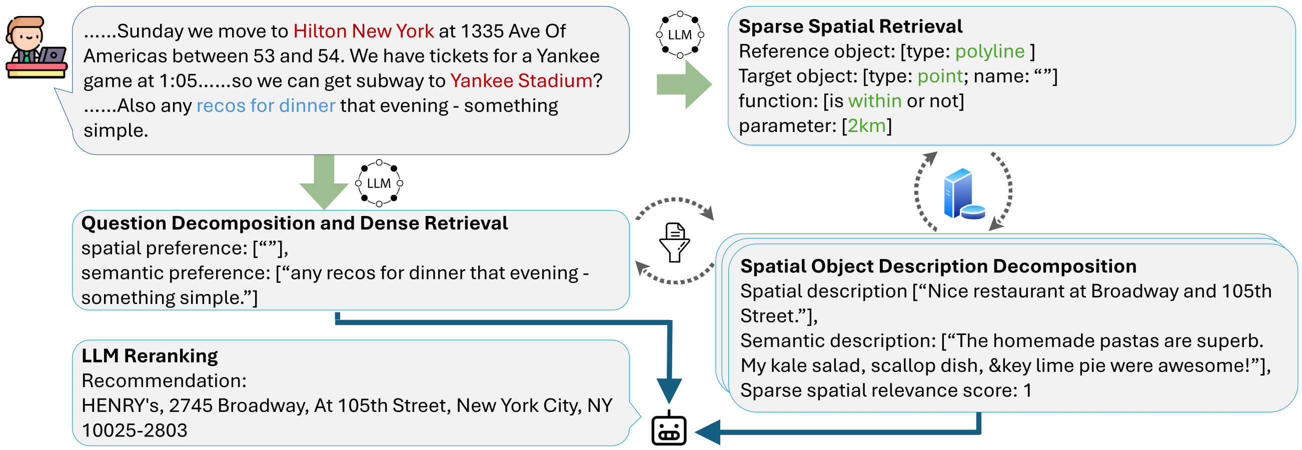


Figure 3. An example of how Spatial-RAG operates: Given a question, 1) **Sparse Spatial Retrieval:** an LLM parses the natural language question into a spatial SQL query for spatial database, retrieving spatial objects that meet the spatial constraints and the sparse spatial relevance scores. 2) **Question Decomposition and Dense Retrieval:** In parallel, Spatial-RAG decomposes the question into spatial and semantic components and compares them with the descriptions of spatial objects to perform dense retrieval, filtering out irrelevant ones. 3) **LLM Reranking:** A language agent balances the spatial and semantic aspects to rerank the candidate and generate the final answer.

Rate, as semantic information is no longer incorporated into the ranking process. Overall, our spatial-RAG framework integrates both spatial and semantic information, leveraging multi-objective optimization to provide more effective and well-balanced spatial object recommendations.

5.4. Case Study

An example of polyline search is given in Figure 3. Spatial-RAG effectively identifies the user's intent within a noisy query, detecting that the user needs to travel between two locations (**Yankee Stadium** and **Hilton New York**) and is specifically looking for dinner recommendations (“any recos for dinner that evening - something simple”). Based on this, Spatial-RAG classifies the spatial operation as a ”point within polyline” query and applies a 2km buffer distance. Within this region, it filters restaurant spatial objects (shown in Fig. 4). In contrast, traditional methods, such as distance-based approaches, typically generate a buffer zone around a single point, which may not necessarily include POIs along the user's travel route. Our framework provides a more context-aware understanding of the user's spatial intent, enabling more precise and relevant recommendations based on both location and user preferences.

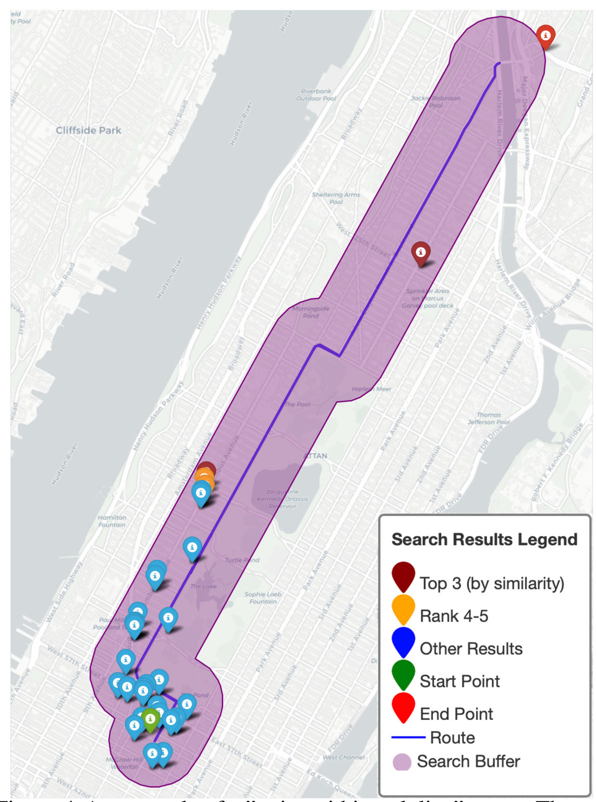


Figure 4. An example of a ”point within polyline” query. The user intends to travel between Yankee Stadium and Hilton New York, seeking dinner recommendations along the route. The buffered region (2 km) is highlighted, and restaurants are selected.

6. Conclusion

Spatial-RAG enhances LLMs' spatial reasoning by integrating structured spatial retrieval with natural language understanding, bridging the gap between spatial databases and AI-driven question answering. Our framework improves spatial intelligence, enabling applications in urban planning, tourism, navigation, and geographic QA. Extensive evaluations show that Spatial-RAG outperforms existing methods, highlighting its potential to advance spatial analysis, autonomous navigation, and augmented reality. Beyond improving spatial reasoning, Spatial-RAG lays the foundation for future multimodal models that

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Cakaloglu, T., Szegedy, C., and Xu, X. Text embeddings for retrieval from a large knowledge base. In *International Conference on Research Challenges in Information Science*, pp. 338–351. Springer, 2020.

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

Christmann, P. and Weikum, G. Rag-based question answering over heterogeneous data and text. *arXiv preprint arXiv:2412.07420*, 2024.

Contractor, D., Shah, K., Partap, A., Singla, P., et al. Large scale question answering using tourism data. *arXiv preprint arXiv:1909.03527*, 2019.

Contractor, D., Goel, S., Mausam, and Singla, P. Joint spatio-textual reasoning for answering tourism questions. In *Proceedings of the Web Conference 2021*, pp. 1978–1989, 2021a.

Contractor, D., Goel, S., Mausam, and Singla, P. Joint spatio-textual reasoning for answering tourism questions. In *Proceedings of the Web Conference 2021*, pp. 1978–1989, 2021b.

Fan, W., Ding, Y., Ning, L., Wang, S., Li, H., Yin, D., Chua, T.-S., and Li, Q. 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*, pp. 6491–6501, 2024.

Haas, L., Skreta, M., Alberti, S., and Finn, C. Pigeon: Predicting image geolocations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12893–12902, 2024.

He, X., Tian, Y., Sun, Y., Chawla, N. V., Laurent, T., LeCun, Y., Bresson, X., and Hooi, B. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *arXiv preprint arXiv:2402.07630*, 2024.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

Li, F., Hogg, D. C., and Cohn, A. G. 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, pp. 18500–18507, 2024.

Li, Q., Wang, Y., Wang, Y., and Zhao, H. Hdmapnet: An online hd map construction and evaluation framework. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 4628–4634. IEEE, 2022.

Li, Z., Zhou, W., Chiang, Y.-Y., and Chen, M. Geolm: Empowering language models for geospatially grounded language understanding. *arXiv preprint arXiv:2310.14478*, 2023.

Liétard, B., Abdou, M., and Søgaard, A. Do language models know the way to rome? *arXiv preprint arXiv:2109.07971*, 2021.

Liu, Y., Yuan, T., Wang, Y., Wang, Y., and Zhao, H. Vectormapnet: End-to-end vectorized hd map learning. In *International Conference on Machine Learning*, pp. 22352–22369. PMLR, 2023.

Mai, G., Janowicz, K., Cai, L., Zhu, R., Regalia, B., Yan, B., Shi, M., and Lao, N. Se-kge: A location-aware knowledge graph embedding model for geographic question answering and spatial semantic lifting. *Transactions in GIS*, 24(3):623–655, 2020.

Mai, G., Janowicz, K., Zhu, R., Cai, L., and Lao, N. Geographic question answering: challenges, uniqueness, classification, and future directions. *AGILE: GIScience series*, 2:8, 2021.

Mai, G., Huang, W., Sun, J., Song, S., Mishra, D., Liu, N., Gao, S., Liu, T., Cong, G., Hu, Y., et al. On the opportunities and challenges of foundation models for geoai (vision paper). *ACM Transactions on Spatial Algorithms and Systems*, 2024.

Majic, I., Wang, Z., Janowicz, K., and Karimi, M. Spatial task-explicity matters in prompting large multimodal models for spatial planning. In *Proceedings of the 7th ACM SIGSPATIAL International Workshop on AI for Geographic Knowledge Discovery*, pp. 99–105, 2024.

Manvi, R., Khanna, S., Mai, G., Burke, M., Lobell, D. B., and Ermon, S. Geollm: Extracting geospatial knowledge from large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

Obe, R. and Hsu, L. S. *PostGIS in action*. Simon and Schuster, 2021.

Roberts, J., Lüddecke, T., Das, S., Han, K., and Albanie, S. Gpt4geo: How a language model sees the world's geography. *arXiv preprint arXiv:2306.00020*, 2023.

Scheider, S., Nyamsuren, E., Kruiger, H., and Xu, H. Geo-analytical question-answering with gis. *International Journal of Digital Earth*, 14(1):1–14, 2021.

Siriwardhana, S., Weerasekera, R., Wen, E., Kaluarachchi, T., Rana, R., and Nanayakkara, S. 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, 2023.

Xie, J., Zhang, K., Chen, J., Zhu, T., Lou, R., Tian, Y., Xiao, Y., and Su, Y. Travelplanner: A benchmark for real-world planning with language agents. In *Forty-first International Conference on Machine Learning*, 2024.

Yu, J., Wu, S., Chen, J., and Zhou, W. 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*, pp. 13586–13612, 2024.

Zhou, Z., Zhang, J., Guan, Z., Hu, M., Lao, N., Mu, L., Li, S., and Mai, G. Img2loc: Revisiting image geolocalization using multi-modality foundation models and image-based retrieval-augmented generation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2749–2754, 2024.

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:

$$C_k(q) = \{y \mid c_k(y, q) \leq 0, \forall c_k \in \mathcal{C}_k(q)\}, \tag{15}$$

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:

$$v_{t,k} = \mathcal{E}(\mathcal{M}_k(d_t)) \quad v_{q,k} = \mathcal{E}(\mathcal{M}_k(q)). \tag{16}$$

The semantic relevance score is then computed using cosine similarity:

$$f_k = \frac{v_{q,k} \cdot v_{t,k}}{\parallel v_{q,k} \parallel \parallel v_{t,k} \parallel}. \tag{17}$$

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 pre-mapped 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.

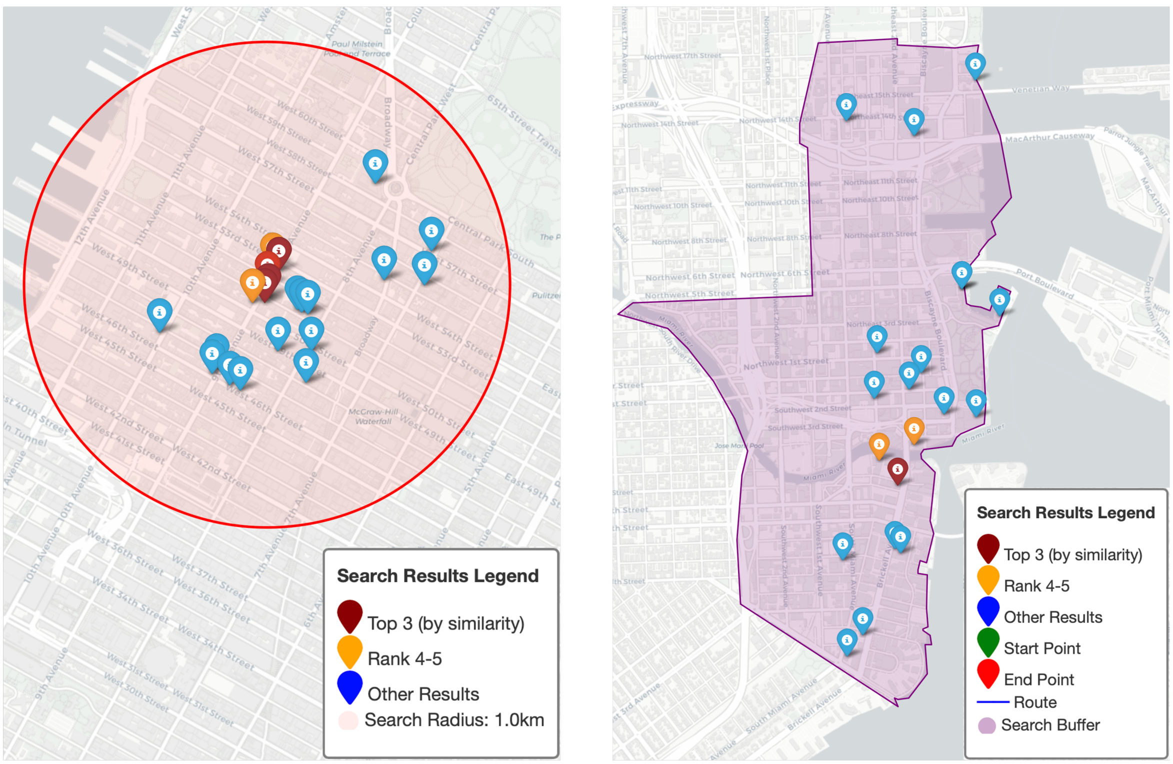
B.2. Semantic Retrieval

While spatial databases address spatial constraints based on the query and spatial database, the actual scenario may be complex, for instance, a hotel may be far from the airport on the map but provide a shuttle, which makes it spatially more convenient than a hotel closer but do not provide a shuttle.

Each spatial object is accompanied by textual descriptions, such as names and reviews. However, the text often contains verbose and irrelevant details that hinder effective decision-making. Moreover, for areas with a high density of POIs that meet spatial requirements, it becomes impractical to input all the text information into an LLM (Large Language Model). To manage the data volume and improve relevance to specific queries, these descriptions are summarized across two perspectives: spatial factors and user preferences. We utilize an LLM to preprocess and summarize spatial objects' reviews offline, storing the results in the database for future comparison. Similarly, the user preferences in the query are dynamically extracted during the online processing stage. The textual aspects of the user preferences and reviews are compared using cosine similarity scores.

C. Other Case Studies

Fig. 5 presents additional case studies, illustrating two spatial object selection strategies: filtering spatial objects within a buffered radius around the user and retrieving spatial objects contained within a specific polygon region.



(a) (b)

Figure 5. (a) An example of a "point within a ϵ-mile radius" query (b) An example of a "point within polygon" query