SpaRAGraph: Spatial Reasoning using Retrieval-Augmented Generation

THANASIS GEORGIADIS, Department of Computer Science and Engineering, University of Ioannina & Archimedes, Athena Research Center, Greece
JOHN PAVLOPOULOS, Department of Informatics, Athens University of Economics and Business & Archimedes, Athena Research Center, Greece
NIKOS MAMOULIS, Department of Computer Science and Engineering, University of Ioannina & Archimedes, Athena Research Center, Greece

The advent of large language models (LLMs) has enabled powerful applications across several domains such as science, healthcare, finance, and law. However, the spatial inference capabilities of LLMs are limited. Our goal is to facilitate more accurate LLM responses to spatial queries. To this end, we leverage inference-time retrieval augmented generation (RAG) to enrich LLM context using external data. We present SpaRAGraph, a framework that i) performs spatial-to-RDF data processing to capture spatial relations between nearby entities, ii) indexes relation-RDFs using a graph to facilitate semantic traversal, and iii) retrieves the relevant context to a question at inference time, contextualizing it with factual, spatial information enhancing the LLM's accuracy. Additionally, we present a spatial reasoning benchmark that challenges LLMs on binary, multiclass and multilabel classification tasks on real-world, spatial entities. Overall, SpaRAGraph sets the ground for using spatial knowledge retrieval techniques to improve LLM effectiveness in spatial reasoning tasks.

CCS Concepts: • **Information systems → Spatial-temporal systems**; **Retrieval models and ranking**; *Retrieval tasks and goals*.

Additional Key Words and Phrases: Spatial Data, RAG, language models, graphs

ACM Reference Format:
Thanasis Georgiadis, John Pavlopoulos, and Nikos Mamoulis. 2026. SpaRAGraph: Spatial Reasoning using Retrieval-Augmented Generation. 1, 1 (February 2026), 31 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Retrieval augmented generation (RAG) [20] improves the performance of generative models, such as large language models (LLMs) by retrieving relevant information from external sources and providing it as context to improve the response quality of the models. RAG has been especially useful when we need to generate responses based on large and complex sources of knowledge that have not been used in the model training process. The success of RAG has brought opportunities for new research in data management and information retrieval toward improving LLM effectiveness [9].

Authors' Contact Information: Thanasis Georgiadis, ageorgiadis@cs.uoi.com, Department of Computer Science and Engineering, University of Ioannina & Archimedes, Athena Research Center, Ioannina, Greece; John Pavlopoulos, Department of Informatics, Athens University of Economics and Business & Archimedes, Athena Research Center, Athens, Greece, ipavlopoulos@aueb.gr; Nikos Mamoulis, Department of Computer Science and Engineering, University of Ioannina & Archimedes, Athena Research Center, Ioannina, Greece.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2026 Copyright held by the owner/author(s). Publication rights licensed to ACM.
Manuscript submitted to ACM

Spatial data collections are typically in structured format and stored in database systems such as PostgreSQL[1] and Oracle Spatial[2], or GIS like QGIS[3]. The relations between all pairs of spatial data entities on a map are not explicitly stored or used in the training process of a foundation model, so existing models are not trained with such knowledge. Only a limited number of spatial relations are typically found in training sources of LLMs such as Wikipedia articles or public-domain books. For instance, the introductory paragraph of the Wikipedia entry on Greece states: *"Greece, officially the Hellenic Republic, is a country in Southeast Europe. Located on the southern tip of the Balkan Peninsula, Greece shares land borders with Albania to the northwest, North Macedonia and Bulgaria to the north, and Turkey to the east. The Aegean Sea lies to the east of the mainland, the Ionian Sea to the west, and the Sea of Crete and the Mediterranean Sea to the south."* This passage conveys a significant amount of spatial information in natural language about Greece's relative topology and location to its neighboring geographic entities (e.g., countries and seas). In contrast, spatial databases may include precise coordinate data and geometrical representations spatial entities, but often lack such relational descriptions, especially in terms of natural, human-interpretable spatial context. However, when a spatial relation involves two entities that are not direct neighbors, such information is more than likely to not be explicitly stated in any pre-training data. In such cases, the relative spatial relation must be computed at inference time. This introduces additional complexity and latency, as it requires a dedicated spatial database system to be integrated with the LLM with a supporting toolchain, capable of categorizing the user's question, translating it into a database query, executing the query on the spatial dataset, and returning the result. Only then can the LLM compose a meaningful and accurate natural language response grounded in factual spatial data. Moreover, when the spatial entities involved are less well-known, such as ZIP codes or parks, or belong to more specialized domains, like soil quality maps or mineral deposit distributions, it is far less likely that their spatial relations are explicitly described in any text used during pre-training, posing a significant challenge for LLMs to infer accurate spatial knowledge solely from prior textual data.

These observations motivate the central research question explored in this paper: *To what extent can LLMs infer spatial knowledge over text, and can such inference be outsourced so that LLMs focus on tasks they are powerful in?* To address this question, we introduce SpaRAGraph, a framework that enhances language model generation through RAG by supplying spatially enriched context. This enables the inference of spatial knowledge from text without requiring model re-training or fine-tuning. The first major challenge lies in the nature of spatial data: it is typically stored in structured, non-linguistic formats (e.g., records or geometries), making it inherently incomprehensible to LLMs in its raw form. Therefore, a spatial-to-text pre-processing step is required to translate this data into natural language descriptions that can both test and guide LLMs' spatial reasoning abilities. The second challenge is ensuring that the system remains efficient and lightweight at inference time, avoiding computationally intensive operations that could slow down response generation. Additionally, we propose the Spatial Reasoning Benchmark (SRB), a novel benchmark designed to evaluate models on binary, multiclass, and multilabel classification tasks involving real-world geographic divisions within the United States and their actual topological relationships. We leverage SRB to evaluate SpaRAGraph, highlighting its effectiveness in facilitating accurate spatial reasoning of small, open models across these tasks.

Compared to other Q&A frameworks employed on spatial data [24, 39, 52], SpaRAGraph differentiates in three dimensions; the first is that it generates per-question context at inference time for guiding the model's response using factual information. Second, it leverages a relational RDF graph to efficiently retrieve and combine spatial relations, minimizing additional latency in the overall process. Third, the graph-based spatial context generation is non-stochastic

---

[1] https://www.postgresql.org/

[2] https://www.oracle.com/database/spatial/

[3] https://www.qgis.org/

and follows a pre-defined spatial relation composition matrix, enabling both faster and more accurate spatial reasoning than pure, LLM-based approaches. Overall, SpaRAGraph performs deterministic, rule-based context generation and relation composition, with the LLM serving as a medium for expressing spatial queries and their results in natural language. The LLM also plays a vital role to providing informed answers when the generated context is not fully explicit and when the inference of other types of information is needed.

The contributions of this paper can be summarized as follows:

- We introduce SpaRAGraph, a novel end-to-end framework for spatial reasoning through inference over spatial text, which leverages graph indexing on spatial RDF data.
- SpaRAGraph comes with the $SpaTex$ module, a scalable, specialized tool for spatial-to-RDF data generation that captures spatial relations between neighboring entities in a dataset.
- SpaRAGraph applies a graph traversal approach assisted by a spatial relation composition matrix on the RDF graph generated by the $SpaTex$ module to conduct spatial reasoning and deterministically generate the appropriate context for the LLM.
- For the evaluation of SpaRAGraph, we introduce SRB, a benchmark that challenges LLMs on binary, multiclass and multilabel classification tasks on real-world, spatial entities.
- Our evaluation demonstrates that SpaRAGraph significantly boosts model performance across all tasks in the SRB datasets, with impressive improvements on the F1 score in different LLMs (an average improvement of 36 percentage points). All tested models benefit from SpaRAGraph to varying degrees, with minimal impact on response time.

The rest of the paper is structured as follows: Section 2 reviews the necessary background and related work. In Section 3 we introduce SpaRAGraph, providing an overview of its architecture and usage examples. In Section 4, we describe the pre-processing stage of SpaRAGraph, during which spatial data is transformed into RDF using $SpaTex$ and indexed using a graph index to enable fast semantic search. In Section 5, we dive deep into how SpaRAGraph uses the graph to generate per-question context that helps LLMs respond more accurately on spatial questions. In Section 6, we introduce SRB, comprised of three synthetic datasets on real-world data that test a model's spatial reasoning in classification tasks. Section 7 presents a comprehensive evaluation of our approach, analyzing its various aspects and testing its performance across a range of open-source models. The paper concludes with Section 8, where we present our conclusions and discuss future work.

2 Background & Related Work

2.1 Retrieval Augmented Generation

Concerning how well an LLM exploits information beyond its pre-trained knowledge base, several RAG benchmarks exist to serve for evaluation. Most of them study the efficiency of the retrieval and the response generation by means of question-answering instances. Specifically, the main aspects studied are: context relevance, i.e., how pertinent the retrieved context to the query is; context utilization, i.e., the extent of the context that is used by the generator to produce the response; error handling, i.e., the ability to handle errors that exist in documents; and completeness, i.e., how well the response incorporates all the relevant information in the context. RGB [2] focuses on data that pertain to news while RAGBench [11] covers different domains. CRAG [48] is a comprehensive factual question-answering benchmark that aims to define types of questions from different domains given their diverse and dynamic nature. BERGEN [33] emphasizes on the LLM-based semantic evaluation of answers, highlighting the importance of using efficient retrievers as they can affect the RAG response generation. MIRAGE [47] measures the accuracy of the predicted correct answer choices on multi-choice questions for the medical domain. Similarly, LegalBench-RAG [29] emphasizes in the legal domain measuring the effectiveness of the retrieval phase and the legal reasoning capacity of LLMs. UDA [16] focuses on the RAG assessment on lengthy and highly unstructured external data such as those found in PDFs and HTML tables. MultiHop-RAG [42] assesses multi-hop queries, i.e. queries that require retrieving information from multiple documents to reason and arrive at an answer. It evaluates the quality of the retrieved set for the query and the reasoning capability of the LLM.

Graphs have gained significant attention in RAG research [13, 25, 28], being applied to a variety of tasks such as summarization, question answering, and knowledge graph reasoning. However, to the best of our knowledge, no prior work has leveraged graph-based indexing and retrieval for spatial data, nor has it systematically evaluated graph-enhanced (or just plain) RAG in the context of spatial reasoning, an emerging and distinct research challenge for LLMs.

2.2 Spatial Reasoning and GeoAI

LLMs have demonstrated strong reasoning capabilities through chain-of-thought (CoT) prompting [45, 50]. This raises the possibility that LLMs could infer the spatial relation between two entities (even when it is not explicitly stored) by reasoning over known spatial relations involving intermediate entities. Li et al. [21] follow this approach, introducing the "Advancing Spatial Reasoning" (ASR) method, which enhances spatial reasoning on large GPT models using CoT and Tree-of-Thoughts (ToT) prompting on the StepGame benchmark [38]. Both ASR and the StepGame benchmark investigate and test directional relations (similar to cardinal directions but not for geographic entities) in a multi-hop reasoning setting that challenges the model's capacity for spatial inference. In contrast to our non-stochastic, rule-based approach, techniques such as CoT and ToT often introduce errors and are significantly slower at inference time, whilst SpaRAGraph is built upon the RAG paradigm, automatically retrieving and generating relevant context to support and enhance spatial reasoning through factual data. Additionally, the StepGame benchmark emphasizes understanding various phrasings of positional directions (such as "6 o'clock" or "to the left") which are inherently egocentric and context-dependent. While effective for evaluating spatial reasoning in local or embodied environments, these expressions are not compatible with geospatial data, which relies on an absolute, coordinate-based reference system (e.g., latitude and longitude) tied to the Earth's surface. As such, StepGame is ill-suited for assessing models' understanding of global spatial relations like cardinal directions and topological relationships.

A more theoritical but preliminary analysis of various spatial relations between entities and the limitations of LLMs in comprehending them, is presented in [5]. However, this work does not examine geographic spatial relations, such as cardinal directions. Related work that leverages external spatial information to assist LLMs includes GeoLLM [24], GeoLLM-Engine [39] and GeoGPT [52]. GeoLLM focuses on regression tasks such as the prediction of population density; it uses auxiliary map data from OpenStreetMap from which the nearby locations of the given (query) location are fetched and passed to the LLM as a fine-tuned prompt. GeoLLM-Engine is an environment of tool agents for earth observation applications. It capitalizes a LLM in order to convert natural language instructions into a set of tasks over satellite images. For this, it performs function calls to geospatial APIs, dynamic maps/UIs and external multimodal knowledge bases. GeoGPT employs an LLM for interpreting the users' demands from the input and calls an external GIS tool from a pool of available ones to solve the task. Some of these tools serve processes that pertain to data collection, data loading and data analysis. GeoLLM, GeoLLM-Engine, and GeoGPT employ a distinct methodology from SpaRAGraph, focusing on tasks unrelated to spatial reasoning.

Another line of research fine-tunes an LLM to enhance its understanding of spatial context. MaaSDB [30] is a vision paper that envisions a spatial database system for enhanced user accessibility by training LLMs on data retained in a spatial database. In this way, the machine learning models can be utilized as a spatial database, enabling a new generation-based query paradigm that replaces the traditional retrieval-based one. LLM-Geo [23] is a prototype that operates as an autonomous GIS that can produce and execute Python code for spatial data loading and visualization. By exploiting the capabilities of the LLM natural language understanding, reasoning and code generation, it manages to generate at first a step-by-step workflow that is formed as a directed acyclic graph given users' data and spatial question. The graph consists of a series of connected operations and nodes. The LLM is reused, as the graph is passed to it in order to generate code in each operation node. Then, the generated code is collected and submitted to the LLM along with the graph and the users' input to create the final program. The program is executed producing the results that can be static maps, charts, new datasets, etc.

Similarly, benchmarking LLMs for their temporal [44] and spatiotemporal reasoning capabilities [19, 22, 31] has gained growing attention. Research in this area is still relatively new, with recent efforts primarily focused on developing benchmarks to establish foundations for future studies and model evaluation. These studies' main area of focus is the future location prediction task, rather than the spatial reasoning over static entities.

Graphs have recently received considerable attention in RAG-related research [28], with a particular emphasis on knowledge graphs [36, 46]. DistRAG [35] is one such approach that focuses on distance-related queries by representing entities as nodes within a graph and encoding their relative distances as edges. It then formulates SPARQL queries that capture the semantics of the input question, enabling the retrieval of the correct answer regarding the distance property. GS-SQL [51] is a graph-based text-to-SQL model that defines an abstract syntax for text-to-query translation, focusing on accurately transforming spatial queries from natural language into SQL. This approach relies on an existing external database with a known schema to generate precise SQL queries, which are then used to query the database either during inference or when required by the user.

Table 1 provides an overview of related work and their respective features when compared with our proposed framework, SpaRAGraph. Notably, none of the existing approaches employ RAG to dynamically generate context during response generation besides DistRAG and SpaRAGraph, with DistRAG focusing exclusively on distance-related queries. Moreover, nearly all related work is evaluated on large, closed-source models (e.g., GPT models), overlooking smaller, open-source models that are more accessible to typical users due to their lower cost and ability to run on commodity GPUs. Furthermore, with the exception of ASR, all methods rely on post-processing spatial data after inference. This necessitates the use of external toolchains that must be both highly accurate and capable of fast spatial computation. ASR differs in that it utilizes CoT reasoning to infer spatial relations between entities at inference time, based on the StepGame benchmark. We elaborate on the specific distinctions between these approaches and SpaRAGraph below.

GeoLLM is a fine-tuning and prompt engineering approach tailored for geospatial tasks, leveraging contextual information extracted from OpenStreetMap [26] along with external toolchains. In contrast to SpaRAGraph, GeoLLM relies on including spatial data in the prompt, assuming that the model can interpret spatial formats, although such representations are not native to language models. Additionally, GeoLLM is currently limited to value-prediction tasks rather than topological or spatial reasoning and requires all spatial entities to be represented as single points rather than complex polygonal geometries. Consequently, GeoLLM is designed for fundamentally different objectives and is not directly comparable to SpaRAGraph.

GeoLLM-Engine is a geospatial copilot application, allowing live interaction with the user through natural language. Its end-to-end pipeline translates and executes a user's queries using geospatial API tools, maps and multimodal databases.

Table 1. An overview of related work on Spatial Reasoning and Geospatial LLM approaches and their features.

	Text-to-Query	CoT	Prompt Engineering	Data Post-Processing	RAG
GeoLLM			✓	✓	
GeoLLM-Engine	✓			✓	
GeoGPT	✓	✓		✓	
LLM-Geo	✓			✓	
ASR		✓			
GS-SQL	✓			✓	
DistRAG	✓				✓
SpaRAGraph					✓

Unlike SpaRAGraph, the generation/LLM component of GeoLLM-Engine is used mostly to translate and breakdown the user's query into a series of operations, rather than use it directly to generate a response.

GeoGPT employs CoT reasoning to decompose user queries, subsequently executing them by selecting appropriate tools from a pool of external GIS services that operate on spatial data stored in a dedicated spatial database. In contrast to SpaRAGraph, the language model in GeoGPT functions primarily as a translator between the user and the database, rather than playing an active role in generating the final response.

LLM-Geo generates and executes geospatial analysis workflows by translating user queries into a sequence of operations and generating Python code to perform these tasks after inference. Unlike SpaRAGraph, LLM-Geo uses the LLM primarily as a scheduler and planner to identify and organize the necessary operations, rather than relying on it as the main component to directly generate the final response.

ASR is similar to SpaRAGraph in that both rely on the model as a central component for spatial reasoning and response inference. However, ASR performs each reasoning step internally within the LLM, whereas SpaRAGraph employs non-stochastic transitional algebra to accurately combine reasoning steps into a coherent overall summary. Additionally, ASR does not utilize dynamic retrieval or augmentation, focusing exclusively on the StepGame benchmark, which provides explicit context per question using synthetic data.

GS-SQL focuses exclusively on improving text-to-SQL translation accuracy for geospatial queries, a task that is fundamentally different from what SpaRAGraph accomplishes.

StepGame operates with per-story, randomly generated entities that exist solely within a localized, relative scope (without persistence across any broader topology). Our proposed Spatial Reasoning Benchmark (SRB) fundamentally differs from StepGame, as it deals with real-world entities embedded in a shared global data space. Furthermore, StepGame evaluates model responses using multilabel classification, whereas SRB assesses models through binary, multiclass, and multilabel classification tasks. Additionally, StepGame relies on a variety of relative directional relations that are not applicable in geospatial contexts, whereas SRB employs cardinal directions and topological relations, which are standard in geospatial reasoning. Inspired by StepGame, we propose and utilize SRB to evaluate SpaRAGraph, as it better aligns with approaches aiming to enhance geospatial reasoning capabilities.

3 SpaRAGraph Overview

This section presents an overview of SpaRAGraph, describing its end-to-end pipeline and showcasing its usage and benefits through example scenarios.

Spatial Textual Context. The geometry of a spatial entity is represented by a sequence of geographic coordinates (longitude, latitude). To compute spatial relations between entities from their raw representations, costly operations, such as line intersection detection, point-in-polygon tests (to detect containment of an object into another), and distance calculations (for proximity detection) must be applied [6]. Additionally, spatial, domain-specific knowledge is missing from foundation models, giving room for improvement via RAG. We hypothesize that if spatial knowledge is expressed comprehensively (via natural language) and concisely (lack of noise, redundancy) in textual form, then the LLM may be able to infer spatial relationships between objects. For example, consider the prompt: *"What is the relative location of A to C, if A is west of B and B is west of C?"* This question asks for the spatial relationship between entities A and C while providing intermediate context from which the relationship can be inferred. When we presented this prompt to Llama-3.1-8B-Instruct, it responded as follows: *"To find the relative location of A to C, let's break it down: A is west of B, B is west of C. This means A is to the west of B, and B is to the west of C. So, A is to the west of C."* The LLM successfully inferred the correct spatial relation between A and C using the context provided with the question, purely over text. Without such context, however, the same model fails to infer the correct relationship between entities (Figure 4).

3.1 Pipeline

Figure 1 illustrates the full SpaRAGraph framework, which is divided into two main stages: pre-processing and inference. **During pre-processing** (performed once unless the underlying datasets change), all spatial datasets are processed through $SpaTex$. This module computes precise spatial relationships (both topological and directional) between entities by analyzing their geometries. The resulting relationships are encoded as RDF triplets and indexed into a global spatial graph. A relation composition matrix is also defined, capturing all possible transitions between spatial relations to support multi-hop reasoning across the graph.

At inference time, when a user poses a spatial question, SpaRAGraph uses named entity recognition (NER) via spaCy [14] to extract mentioned spatial entities. These are matched to graph nodes using FAISS-based similarity search. The shortest paths connecting matched entities are then computed, and the corresponding RDF triplets are retrieved. Using the relation composition matrix, these triplets are summarized into a coherent spatial context, which is appended to the user's original query. This enriched prompt is finally passed to the LLM for response generation.

3.2 Usage

An example interaction with our framework is shown in Figure 2, demonstrating how it facilitates spatial reasoning and natural language dialogue by automatically generating the necessary contextual information, relieving the user of the burden to supply it manually. We focus on generation tasks involving spatial relations (e.g., addressing the following question: *"Is Dickinson County, Kansas east of Douglas County, Kansas?"*), implementing and testing our framework using spatial data. Nonetheless, our approach can be generalized to assist any RAG approach that involves complex relations between objects that can be supported by inference rules. Additionally, SpaRAGraph is compatible with existing RDF datasets, bypassing the need for spatial-to-RDF pre-processing, provided that the RDF data encodes semantically meaningful relations between entities. For effective traversal and inference, a composition matrix or algebra must be defined over the relation set, enabling the combination of any pair of relations to compute a composed relation between any two connected entities in the graph.

Figure 3 presents the complete map of Kansas counties, with the possible shortest paths highlighted through which the spatial relation between Dickinson County and Douglas County was inferred using the RDF graph. Assuming that each pair of neighboring counties has their spatial relation explicitly encoded in RDF, the relation between the two counties can

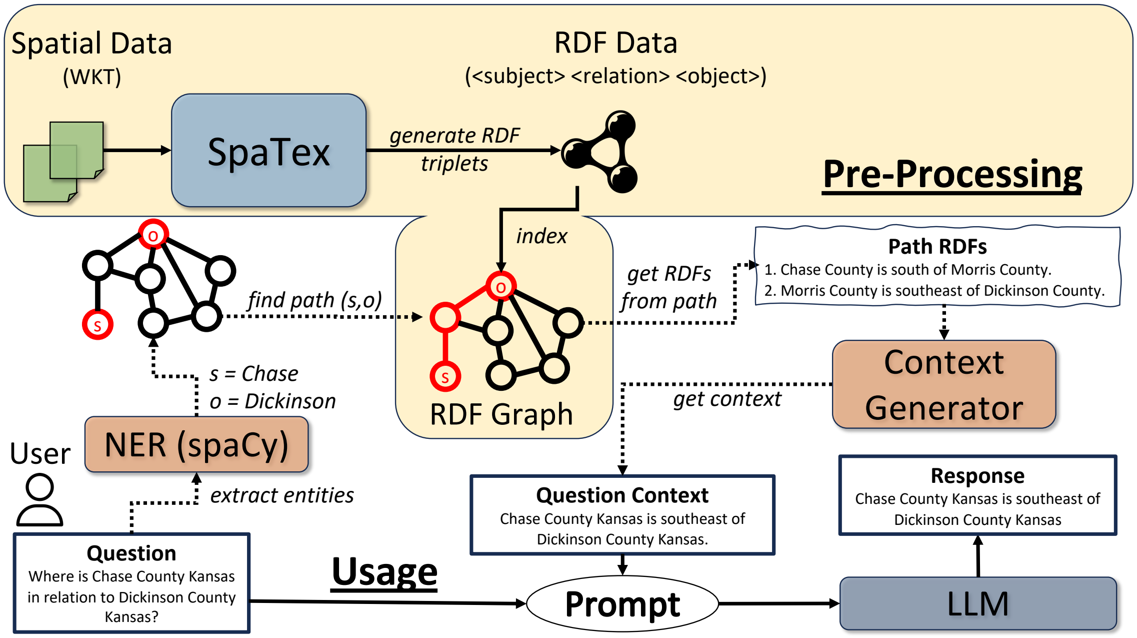
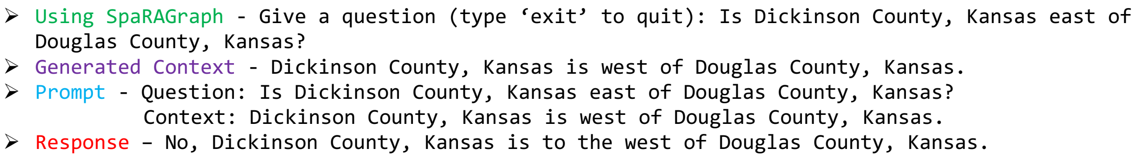

Spatial Data
(WKT)
SpaTex
generate RDF triplets
RDF Data
(<subject> <relation> <object>)
Pre-Processing
index
o
s
find path (s,o)
RDF Graph
get RDFs from path
Path RDFs
1. Chase County is south of Morris County.
2. Morris County is southeast of Dickinson County.
Context Generator
get context
s = Chase
o = Dickinson
NER (spaCy)
User
extract entities
Question
Where is Chase County Kansas in relation to Dickinson County Kansas?
Usage
Question Context
Chase County Kansas is southeast of Dickinson County Kansas.
Prompt
Response
Chase County Kansas is southeast of Dickinson County Kansas
LLM

Fig. 1. SpaRAGraph's overview divided into two stages: pre-processing, performed once and consisting of *SpaTex*'s spatial RDF generation and their indexing in a graph; and usage, which illustrates user interaction with SpaRAGraph, showing how context is generated in the back-end through named entity recognition and path search on the RDF graph, before being appended to the prompt and sent to the LLM.



```
➢ Using SpaRAGraph - Give a question (type ‘exit’ to quit): Is Dickinson County, Kansas east of
  Douglas County, Kansas?
➢ Generated Context - Dickinson County, Kansas is west of Douglas County, Kansas.
➢ Prompt - Question: Is Dickinson County, Kansas east of Douglas County, Kansas?
           Context: Dickinson County, Kansas is west of Douglas County, Kansas.
➢ Response – No, Dickinson County, Kansas is to the west of Douglas County, Kansas.
```

Fig. 2. Example interaction with SpaRAGraph when given a spatial question. The context is automatically generated and appended to the prompt, to guide the model's response.

be derived compositionally. For instance, Dickinson County is west of Geary County, Geary County is west of Wabaunsee County, and so on, allowing us to infer that Dickinson County is west of Douglas County by chaining these intermediate spatial relations step by step. In the example of Figure 2, SpaRAGraph helps Llama-3.1-8B-Instruct respond correctly to a question regarding the relative direction of two counties in Kansas, by enriching the original prompt with the necessary context for an accurate response. With the context-enriched prompt, the model is able to answer the question accurately.

Without the generated context, the model may either be unable to respond or hallucinate on the answer based on the general knowledge it might possess. In Figure 4, for example, we prompted Meta's Llama-3.1-8B-Instruct model with the question: *"What is the relative location of Dickinson County to Douglas County in Kansas?"* Based on its response, it becomes evident that the model possesses a general understanding of the locations of the two Counties within the State of Kansas, but it is ultimately unable to infer their relative spatial positioning and hallucinates, responding incorrectly. This

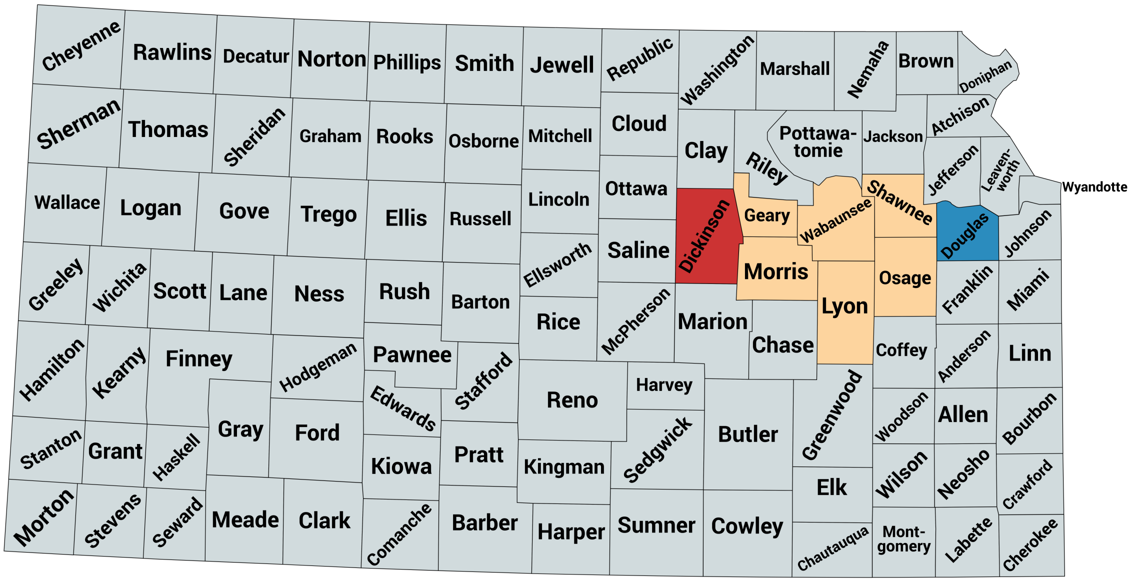
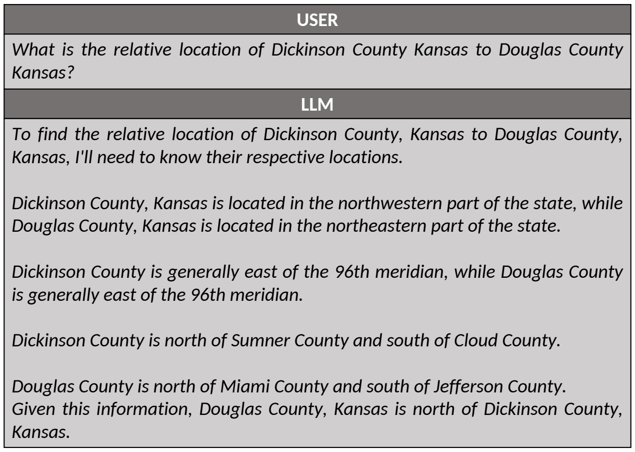


Fig. 3. Map of all counties in Kansas, illustrating possible shortest paths (yellow) between Dickinson County (red) and Douglas County (blue). These paths enable inference of the relative spatial relation between the two entities through intermediate connections. The map was created on https://www.mapchart.net



USER

What is the relative location of Dickinson County Kansas to Douglas County Kansas?

LLM

To find the relative location of Dickinson County, Kansas to Douglas County, Kansas, I'll need to know their respective locations.

Dickinson County, Kansas is located in the northwestern part of the state, while Douglas County, Kansas is located in the northeastern part of the state.

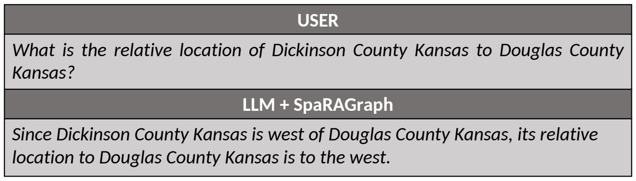
Dickinson County is generally east of the 96th meridian, while Douglas County is generally east of the 96th meridian.

Dickinson County is north of Sumner County and south of Cloud County.

Douglas County is north of Miami County and south of Jefferson County.
Given this information, Douglas County, Kansas is north of Dickinson County, Kansas.

Fig. 4. Llama-3.1-8B-Instruct response to an example spatial question.

limitation highlights the model's deficiency in spatial reasoning based solely on textual information. On the other hand, Figure 5 shows the model's response to the same question **when employed with SpaRAGraph**. The context generated by SpaRAGraph was deterministically constructed and explicitly conveyed the relative locations of the two counties to the model, thereby guiding and informing its response internally.



USER

What is the relative location of Dickinson County Kansas to Douglas County Kansas?

LLM + SpaRAGraph

Since Dickinson County Kansas is west of Douglas County Kansas, its relative location to Douglas County Kansas is to the west.

Fig. 5. Llama-3.1-8B-Instruct response to an example spatial question when employed with SpaRAGraph.

4 Pre-processing

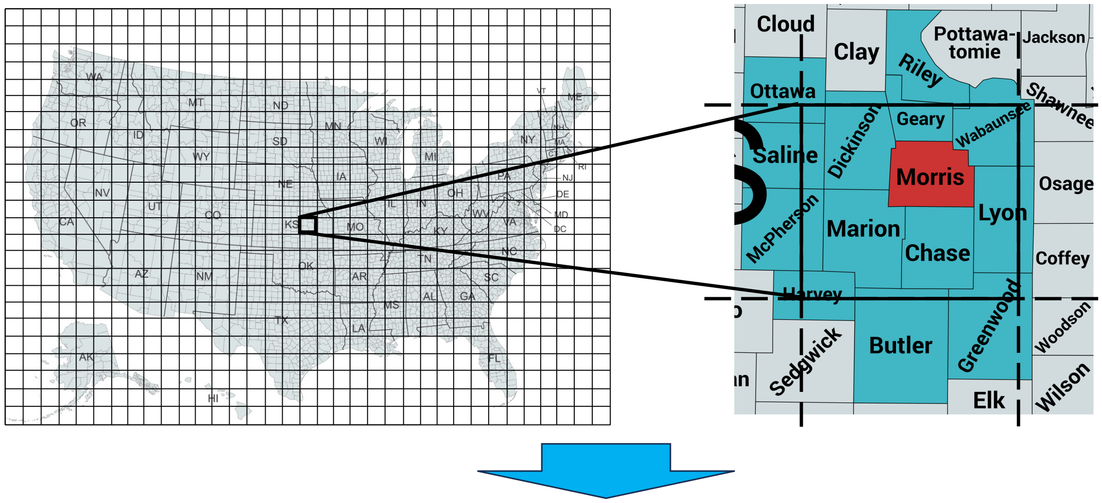
SpaRAGraph relies on the pre-processing of raw spatial data to generate synthetic RDF data that captures key relationships between entities. By explicitly storing a subset of spatial relations from the dataset, SpaRAGraph can infer relations that are not explicitly stored and enriches model responses with related context. The pre-processing stage consists of two components. First, the $SpaTex$ component processes all input spatial data and generates RDF triplets that capture meaningful spatial relations among entities. Second, these RDF triplets are organized into a bidirectional graph index, forming a per-relation topological structure over all unique entities. This enables efficient computation of relationships between entities by traversing the graph through intermediate nodes.

4.1 $SpaTex$: Spatial Relation RDF Generator

Spatial knowledge may contain various different aspects and metrics, such as the distance between entities, their topological relationships (e.g. adjacent, intersect) and the cardinal direction of an entity in relation to another one (e.g. north, southwest). We refer to any type of relation between two geographical entities as a *spatial relation*. Note that determining the cardinal direction between two non-point, arbitrary polygons (such as countries or lakes) can be inherently subjective. Different parts of one object may lie north, east, or northeast of parts of the other. To produce a definitive spatial relation, we compute the centroids of each polygon and determine the cardinal direction based on the relative position of these centroids. While this approach simplifies the spatial representation, it introduces some distortion in the resulting RDF triplets, rendering them approximate. Consequently, any inference derived from these relations should also be considered approximate.

To extract these spatial relations and generate RDF triplets that describe them comprehensively and concisely, we introduce $SpaTex$, a rule-based spatial-to-RDF data generator that takes as input spatial data collections in raw format (WKT, CSV, etc.). An overview of $SpaTex$ is shown in Figure 6. The output is a collection of RDF triplets that encapsulate the spatial relations between (nearby) pairs of objects. We aim for the RDF triplets to be as concise and simple as possible, minimizing redundancy by ensuring that each subject and object consists of a single entity name, and that each predicate captures only the spatial relation, free of any descriptive or “flavor” text. However, predicates must still be concise and semantically informative enough for an LLM to effectively infer implicit spatial relationships not explicitly encoded in the data.

Generating spatial RDF triplets that capture nearby entities’ spatial relations is computationally challenging. The total number of pairwise spatial relations between entities on a map is quadratic, making their generation and encoding a challenging task. In SpaRAGraph, we address this scalability challenge by i) dividing the map into numerous local partitions, ii) computing non-trivial spatial relations between all pairs of entities within each local region, and iii) structuring the computed relations in a clear and comprehensive RDF format to enhance the model’s ability to infer non-local relations.



<subject> <predicate> <object>
<Morris County Kansas> **<north>** <Chase County Kansas>
<Chase County Kansas> **<south>** Morris County Kansas>
<Morris County Kansas> **<northeast>** Marion County Kansas>
<Marion County Kansas> **<southwest>** Morris County Kansas>
<Morris County Kansas> **<east>** Dickinson County Kansas>
<Dickinson County Kansas> **<west>** Morris County Kansas>
...

Fig. 6. The Spatial relation identification process by *SpaTex* that uses a global grid to group nearby entities and compute their spatial relations, outputting them in RDF format.

The partitioning approach employed by *SpaTex* has two advantages. First, we avoid computing an excessive (and redundant) number of spatial relations, which can be inferred; for two entities (e.g., Counties) in different partitions, their relation should be disjoint and their relative cardinal direction can be inferred by the cardinal directions of entities that enclose them (e.g., States). Second, each partition is processed independently and in parallel, scaling up the relation generation process. The input spatial data must be accompanied by metadata that provides a human-readable name for each entity, enabling reference in natural language. RDF data often uses URIs, which may or may not include meaningful names. When the URIs already contain readable names, *SpaTex* retains them as node labels in the graph. However, if the URIs consist of non-descriptive identifiers or links, *SpaTex* replaces them with custom URIs derived from the provided metadata (names) to ensure compatibility with natural language processing.

For the detection of topological relations, we use the standard Dimensionally Extended 9-Intersection model (DE-9IM) [3]. DE-9IM defines a 3×3 matrix where the rows and columns represent two objects' interior, boundary and exterior areas. Note that, by the definition of DE-9IM, spatial relations are asymmetric. This means that any pair of entities can have at most one spatial relation in a given direction, and at most its inverse in the opposite direction. In other words, a single pair of entities cannot simultaneously be related by multiple distinct spatial predicates. The combination of values in the matrix defines the exact topological relationship for two objects. Moreover, *SpaTex* calculates the cardinal direction between nearby objects in relation to one another, as well as their in-between distance and their common area (if any) in square kilometers. For two input spatial datasets R and S, $SpaTex$ performs a *spatial join* $R \bowtie S$ between them, an operation that identifies all pairs of objects $\langle (r, s) | r \in R, s \in S \rangle$ that intersect with each other. For each dataset, a *self-join* is performed ($R \bowtie R$ and $S \bowtie S$), to identify relations between objects in the same dataset as well.

The vast majority of object pairs in real-world spatial datasets are *disjoint* [12], so we only detect and generate non-disjoint topological relations, as disjointness can be implied. This saves us both the effort and the overhead of encoding and retrieving disjoint relations. In general, spatial relations between objects that are disjoint and far from each other can be inferred by LLMs and do not need to be explicitly defined in the context. For example, describing two entities as adjacent implies that their borders touch and thus, LLMs can infer that since they touch, they are not disjoint with each other.

SpaRAGraph takes advantage of spatial reasoning as much as possible to reduce the volume of the generated relations by $SpaTex$. To this end, we partition the data space using a uniform grid and assign each spatial entity to the partitions (i.e., tiles) that it spatially overlaps. $SpaTex$ then performs a partition-to-partition spatial join [27] for each cell; hence, we only compute and generate the spatial relations between objects of the same tile. For any pair of objects in a partition, we first compare their Minimum Bounding Rectangle (MBR(r)). If the MBRs do not intersect, then we only compute the relative cardinal direction between them (e.g., north of); otherwise, we compute the DE-9IM matrix. For overlapping objects, we only generate the topological relation (e.g., overlaps, inside, covers); if the objects are adjacent, we also compute their cardinal direction relation.

4.2 Graph-based Topology Index

We index the spatial RDF triplets generated by $SpaTex$ using a directed graph $G = (V, E)$, where each node $v \in V$ represents an entity from the original data and each edge $e = (v_1, v_2) \in E$ corresponds to a spatial relation (predicate) from $v1$ (subject) to $v2$ (object).

For each pair of entities, $SpaTex$ generates bi-directional RDF triplets. For example, both <The State of Kansas> <contains> <Morris County Kansas> and <Morris County Kansas> <inside> <The State of Kansas> are created. This results in two edges between the nodes <Morris County Kansas> and <The State of Kansas>, one labeled <contains>, the other <inside>. This bi-directionality supports a greater number of possible paths during graph indexing, resulting in more flexible graph traversal. An illustration of an example graph index for a set of RDF triplets is shown in Figure 7. Recall that $SpaTex$ computes spatial relations only between entities that fall within the same grid cell.

The total number of nodes $|V|$ in the graph corresponds to the number of unique entities in the original input data. The number of edges $|E|$, however, depends on the granularity of the grid used by $SpaTex$ when computing spatial relations. A fine-grained grid results in fewer RDF triplets, as each object is compared against a smaller set of nearby neighbors. In contrast, a coarse-grained grid leads to a significantly higher number of relations, since each entity is compared with more entities within the same larger cell. Grid granularity is directly correlated with the size of the objects (in terms of area covered). We found that relatively coarse-grained grids (e.g., 1,000 cells per dimension) are sufficient for larger entities, such as U.S. Counties and States. In contrast, smaller entities (such as Zipcodes) benefit from finer grids (e.g., 10,000 cells per dimension), which produce a meaningful number of spatial relations without generating an excessive number of RDF triplets.

We hypothesize that a small number of spatial RDF triples is sufficient to help models infer missing relations, allowing them to answer questions about entities whose relationships are not explicitly stored by reasoning over the available information. The graph index represents the final step of the pre-processing stage and must be loaded and ready at

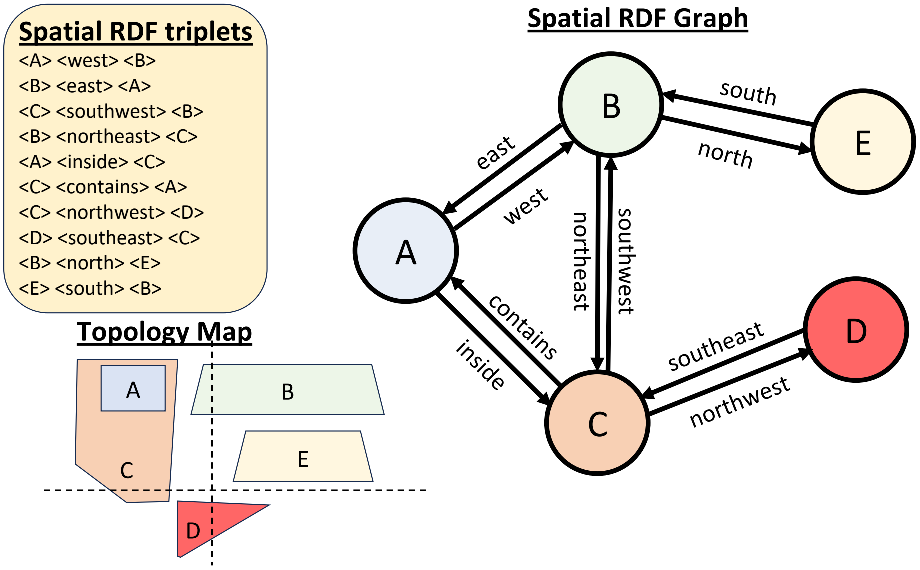


Fig. 7. An example graph constructed from a set of spatial RDF triplets. The map illustrates the spatial arrangement of objects in the data space, with dotted lines indicating the grid cells defined by $SpaTex$ and the resulting bi-directional index built over them.

inference time. Since the spatial relationships between entities typically remain static, there is usually no need to repeat the pre-processing pipeline (from $SpaTex$'s RDF generation to graph indexing) more than once.

4.3 Spatial Relation Composition matrix

Building on the foundations established by DE-9IM [3, 4], RCC8 [32], and Cardinal Direction Composition [40], we define a simplified composition matrix for pairs of spatial relations (Table 2). The matrix sets the rules for composing an overall spatial relation between two entities A and B, based on their relation with an intermediate entity I. Specifically, for any two RDF triplets <A> <p_1> <I> and <I> <p_2> <B>, the matrix defines the composition of the two predicates p_1, p_2 as $p_1 \rightarrow p_2 = p_3$, forming a new relation p_3 such that the RDF triplet <A> <p_3> <B> describes accurately the spatial relation between entities A and B.

Topological relations such as *inside*, *contains*, *intersects*, and *meets* (adjacency) often become redundant once combined with cardinal directions. For example, for two example RDF triplets <o_1> <$inside$> <o_2> and <o_2> <$south$> <o_3>, then o_1 inherits that directional relation from o_2 to o_3, generating <o_1> <$south$> <o_3>. The same goes for the *contains* relation. When not combined with a cardinal direction, the intersection/containment relations usually carry on, as long as they remain the same. For example, if o_1 is inside of o_2 and o_2 is inside of o_3, then o_1 is inside of o_3 as well. In our case, through $SpaTex$'s global grid and per-cell processing of neighboring entities, all non-disjoint topological relations will be generated and explicitly stored, meaning that they won't have to be inferred. Hence, SpaRAGraph simplifies paths by discarding topological relations when they co-occur with directional ones, preserving only the latter. Regarding cardinal directions, Table 2 presents a simplified form of the transition rules from [40] by selecting a single representative transition where multiple transitions are possible. This reduction is deliberate: it reduces ambiguity presented to the model and enforces a consistent, conventional mapping for composed directions. The simplification trades some formal completeness

Table 2. Relation composition matrix with a right arrow (→) denoting left-to-right composition. The relations *inside* and *contains* indicate that one entity lies entirely within another, and vice versa. The relation intersects refers to a general overlap where two entities share area and their boundaries cross, but no more specific topological relation or cardinal direction applies. The "???" transitions denote uncertainty and require special handling.

→	**N**	**NE**	**E**	**SE**	**S**	**SW**	**W**	**NW**	**inside**	**contains**	**intersects**
N	N	NE	NE	E	???	W	NW	NW	N	N	N
NE	NE	NE	E	E	E	???	N	N	NE	NE	NE
E	NE	NE	E	SE	SE	SE	???	N	E	E	E
SE	E	E	SE	SE	SE	S	E	???	SE	SE	SE
S	???	E	SE	S	S	S	SW	W	S	S	S
SW	W	???	S	S	SW	SW	W	W	SW	SW	SW
W	NW	N	???	E	SW	W	W	NW	W	W	W
NW	NW	N	N	???	W	W	NW	NW	NW	NW	NW
inside	N	NE	E	SE	S	SW	W	NW	inside	???	???
contains	N	NE	E	SE	S	SW	W	NW	???	contains	???
intersects	N	NE	E	SE	S	SW	W	NW	???	???	???

for clarity and predictability. As a result, certain corner cases such as relations between very distant or irregularly shaped entities, may be prone to loss of directional precision. An alternative approach would be for SpaRAGraph to compose all possible cardinal directions in such cases, allowing the model to reason about potential approximations and error margins. In this work, however, we adopt the simplified conventions shown in Table 2 to enable automated evaluation and the design of a deterministically testable benchmark.

On the other hand, cross-topological combinations are much harder to deduce accurately. For example, for two RDF triplets <o_1> <$inside$> <o_2> and <o_2> <$contains$> <o_3>, the combined relation $inside \rightarrow contains = p_c$ does not, by itself, convey meaningful information about the relation between o_1 and o_3. We can disregard cases where $p_c = inside$, as $SpaTex$ would have already computed and explicitly stored the RDF triplet <o_1> <$inside$> <o_3>. The same applies to the $contains$ relation. Additionally, this composition indicates that both o_1 and o_3 are inside the same, larger entity o_2. For their explicit relation not to exist in the graph, they have to be far away enough so that they do not fall inside the same cell of $SpaTex$'s grid as well. Aside from this, it remains unclear what the correct relation p_c is, based solely on the given pair of RDF triples.

Another special case is the $intersects$ relation, which does not convey much useful information in combination with other topological relations, as it represents the broadest superset of all non-disjoint relations, expressing intersection in its most general form. For example, the composed relation between two RDF triplets <o_1> <$intersects$> <o_2> and <o_2> <$intersects$> <o_3> can not be $intersects$, $contains$ or $inside$, otherwise it would have been explicitly generated by $SpaTex$. On the other hand, for such case to appear, o_2 must have large extent, so that o_1 and o_3 intersect with it inside different cells in $SpaTex$'s grid, otherwise their relation would have been stored explicitly. In such cases, entities o_1 and o_3 may only be related by a cardinal direction, which cannot be deduced based solely on this pair of RDF triplets.

Similarly, cardinal direction composition may also lead to uncertain cases. For example, if entity <o_1> <$north$> <o_2> and <o_2> <$south$> <o_3>, then <o_1> could be either <$north$>, <$south$>, or even <$inside$>, <$contain$>, <$intersects$> or <$meets$> with <o_3>. However, given the way $SpaTex$ generates the RDF triplets, the latter cases would have been explicitly represented if they were true. Therefore, in such scenarios, <o_1> could only be either <$north$> or <$south$>

of <o_3>. In practice, these 180-degree reversals in directionality are rare. This is because the spatial partitioning and relation-generation strategy used by *SpaTex* enables shortest-path traversal between distant entities to remain generally single-directional. As a result, when composing relations from one entity to another, the resulting paths rarely contain all possible cardinal directions within a single traversal.

These rare-but-possible edge cases are labeled with three question marks ("???") in the composition matrix, as they require specialized handling by SpaRAGraph. We elaborate on how such cases are managed in Section 5.2. Moreover, as discussed in Section 4.1, defining the relative direction between two entities using a single cardinal direction term is often only partly accurate and inherently approximate. This approximation carries over into the estimations performed during the compositions shown in Table 2.

5 Spatial Reasoning with SpaRAGraph

In this section, we detail how SpaRAGraph utilizes the graph index constructed over the spatial RDF data to generate relevant context for a given spatial question during inference. The process consists of two main stages: First, SpaRAGraph identifies the spatial entities referenced in the question and traverses the RDF graph to determine their relative spatial relation. Then, it synthesizes this information into a coherent context, which is appended to the original question and provided to the language model to support accurate and informed reasoning.

5.1 Graph Traversal

Named Entity Recognition When a spatial question is posed, the first step is to identify the entities it references. We use spaCy [14] for this NER task. spaCy must be configured to recognize the types of entities present in the RDF data. For example, in our experimental evaluation, we use data from States, Counties, and Zipcodes in the U.S., so we customized spaCy's rules to distinguish between these entity types and accurately identify them. Note that the extracted entities are returned in the order they appear in the user's prompt. More advanced recognition rules can be defined to further guide SpaRAGraph's graph traversal and context generation, such as identifying specific relations to target in the graph or retrieving multiple paths between two entities. However, in this initial version of SpaRAGraph, we simply use spaCy to extract all referenced entities.

Entity-Node Matching Once the referenced entities are extracted, they must be matched to the corresponding nodes in the RDF graph. However, node names are stored as URIs, which may not match the way users refer to the same entities. For instance, a user might refer to "The State of Kansas" as "Kansas State", simply "Kansas", or by its abbreviation, "KS." To bridge this gap, we perform entity-to-node matching via similarity search. All node labels are indexed using FAISS [18] to enable fast approximate similarity search. Each extracted entity is then queried against this index, and the most similar node label is returned as its match.

Path Search The identified entities are then paired into $(start, end)$ tuples, defining the list of paths to be searched in the RDF graph. A path p in a graph $G = (V, E)$ is defined as $p = (v_0, v_1, ..., v_k)$ such that all edges (i.e. pairs of vertices) $v_i, v_{i+1} \in E$ for all $i = 0, 1, ..., k - 1$. Note that, due to the way *SpaTex* computes spatial relations, a given pair of nodes can have at most one edge in each direction between them. Depending on the task, multiple paths —possibly in both directions— may need to be considered. To determine the spatial relation between two entities, a single path may suffice. However, when querying the relation of one entity with multiple others, it is necessary to traverse additional paths to comprehensively capture all relevant relations. For a given pair $(subject, object)$, SpaRAGraph first attempts to find the shortest path (the one with the fewest edges) from $subject$ to $object$. We use Breadth-First Search (BFS) because all edges in the graph are considered to have equal weight, making it effectively an unweighted graph. BFS is optimal for such cases, as it guarantees the discovery of the shortest path in terms of the number of edges (i.e., the minimal number of intermediate RDF triples) from the $subject$ to the $object$. If no path is found, it then searches in the reverse direction, from $object$ to $subject$. If no path exists in either direction, no contextual information is generated for that entity pair. For questions involving multiple entities, paths for all possible $(start, end)$ combinations may need to be computed, depending on the task. In this version of SpaRAGraph, the path search strategy is defined by the task type. In our experimental evaluation, we test SpaRAGraph on binary, multiclass, and multilabel classification tasks. For binary classification, we extract a single $(subject, object)$ pair based on the order of appearance in the question. If no path is found from subject to object, we attempt the reverse direction, from object to subject. We follow the same strategy for multiclass classification, where each question involves two entities and a set of relation options, only one of which is correct. Again, we search for a path from subject to object and fall back to the reverse if necessary. In multilabel classification, each question involves a single subject entity and multiple candidate object entities. The task is to identify all (if any) object entities that satisfy the specified relation in the question. In this case, we compute paths from the subject to each candidate object. If no path is found in that direction, we then check the reverse direction on a per-pair basis.

5.2 Context Generation

At this stage, SpaRAGraph has identified a list of paths between the entities mentioned in the question. For a path p from the subject to the object, each hop in the path corresponds to a spatial RDF triple. This creates a natural, sequential order in which to process the triples (from subject to object) in order to determine their overall spatial relationship. The next step is to generate context based on the identified path(s) and use it to augment the user's question, providing the LLM with relevant spatial information to guide its response so that it answers correctly.

We semantically evaluate a path through deterministic, rule-based means, following the relation composition matrix (Table 2). Figures 8 and 9 show two end-to-end examples of how graph traversal and context generation is done for two spatial questions on the example data and graph of Figure 7. In Figure 8, the spatial relation between entities A and D is not explicitly stored as an RDF, so the path from A to D is longer than one hop. Combining the *inside* and *northwest* relations that A and D have with intermediate entity C respectively, leads to the conclusion that A is northwest of D (as the 'inside' topological relation is discarded when combined with a directional one). We can verify from the map that entity A is indeed northwest of entity D, confirming that the generated context is accurate. This context is then provided alongside the original spatial question to the LLM, guiding its response with factually grounded information about the referenced entities. In the example of Figure 9, the path from entity D to entity E traverses through entities C and B, composing each spatial relation step-by-step and ultimately concluding that entity D is south of entity E. This result is approximately correct, offering the LLM a useful hint about D's location relative to E. However, the more precise relation would be that D is southwest of E , with this loss of precision being due to the undetermined composition absorption approach. While still generally correct, this detail cannot be inferred from the available data without directly accessing the entities' geometries and computing their exact relative cardinal direction.

Furthermore, due to the structure of our spatial RDFs and their indexing in the graph, adjacency relations (*meets*) typically become obsolete beyond the first hop in the path. This is because all adjacency relations are explicitly captured in the graph, as $SpaTex$ groups nearby entities into grid cells and computes relations among neighboring entities. Hence, the shortest path in the graph between two adjacent entities will be a single hop, namely, the RDF that describes their adjacency and relative direction to one another.

We handle composition edge cases (marked as "???" in Table 2) as follows: if the shortest path between two nodes contains any of these undetermined compositions, SpaRAGraph iteratively searches for the next shortest path, continuing

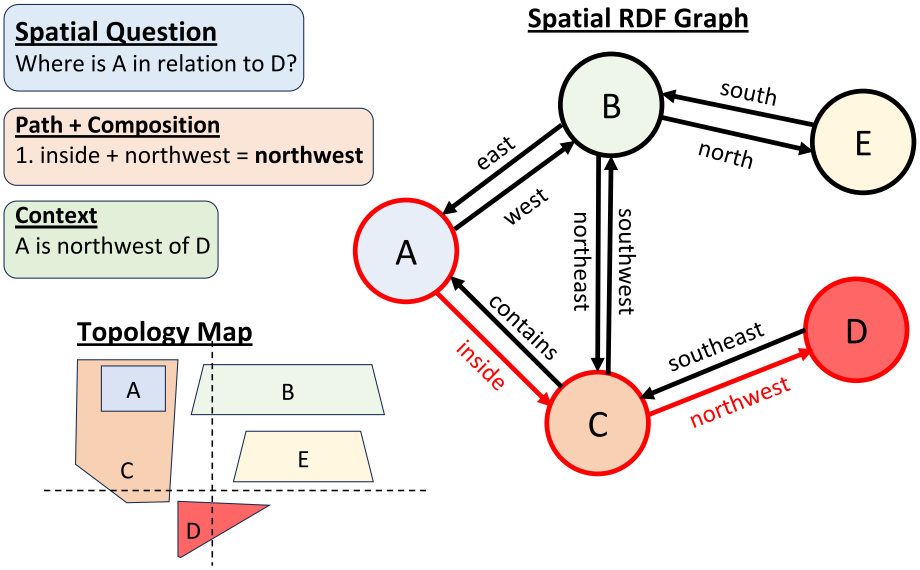


Fig. 8. Path traversal and context generation for an example spatial question based on the data in Figure 7. SpaRAGraph identifies a 2-hop path, performs one directional composition, and generates context that accurately captures the spatial relation between the entities based on the map.

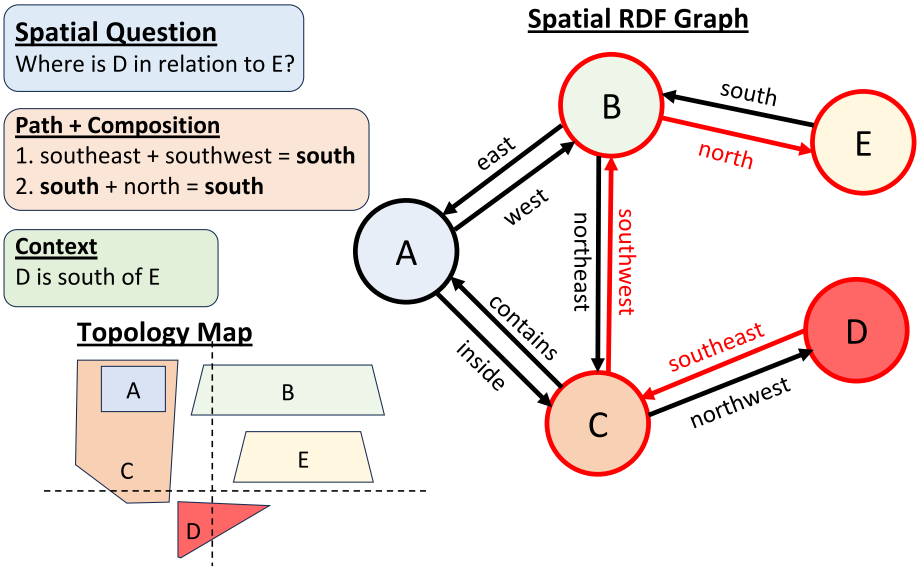


Fig. 9. Path traversal and context generation for an example spatial question based on the data in Figure 7. SpaRAGraph identifies a 3-hop path, performs two directional compositions, and generates context that is approximately correct for entities D and E. in reality, D lies southwest of E based on the map.

this process until it finds a path that includes at least one valid composition (up to a threshold). If no such paths exist, SpaRAGraph can either handle uncertain rules by retaining the incoming one (e.g., south + north = south), or generate contextual input that aggregates all RDF triplets from the top k candidate paths. The latter strategy provides the LLM with as much relevant information as possible, even when composition data is incomplete or ambiguous. However, reasoning over large sets of spatial RDF triplets is more challenging for the model; therefore, we adopt the first approach as the default. In our Spatial Reasoning Benchmark described in Section 6, there are no uncertain cases in the Yes/No and Radio question types. Only two questions in the Checkbox set involve uncertain transitions along their shortest paths, both of which are successfully resolved using the second-shortest path.

Based on how cardinal direction relations are computed by $SpaTex$ and the inherently approximate composition matrix, our External Reasoning approach is by extension approximate as well; however, it provides a fast and efficient way to semantically compose spatial relations over text, enabling the generation of contextual information that can assist an LLM in answering spatial questions more accurately.

6 SRB - The Spatial Reasoning Benchmark

Evaluating the generation performance of a retrieval-augmented method such as SpaRAGraph on real-world spatial data requires two components: (i) datasets that are loaded and query-ready in an external database (in our case, RDF triplets that have been generated and indexed in the graph), and (ii) a set of question–answer pairs that either directly relate to the data in the database or are at least relevant to it. Existing spatial reasoning benchmarks, such as StepGame [38], do not involve real-world, globally positioned spatial entities. As a result, they are not well-suited for evaluating RAG approaches, unless a small, per-question index is artificially constructed. However, such setups fall short of meaningfully testing a method's scalability and retrieval capabilities.

To thoroughly evaluate SpaRAGraph, we introduce three datasets comprising natural language spatial questions based on real-world geographic divisions in the United States, specifically States, Counties, and ZIP Codes (hereafter referred to as Zipcodes). These questions evaluate topological relationships to benchmark methods that enhance spatial inference in large language models. The three datasets test binary, multiclass, and multilabel classification tasks, respectively, using natural language texts containing spatial knowledge.

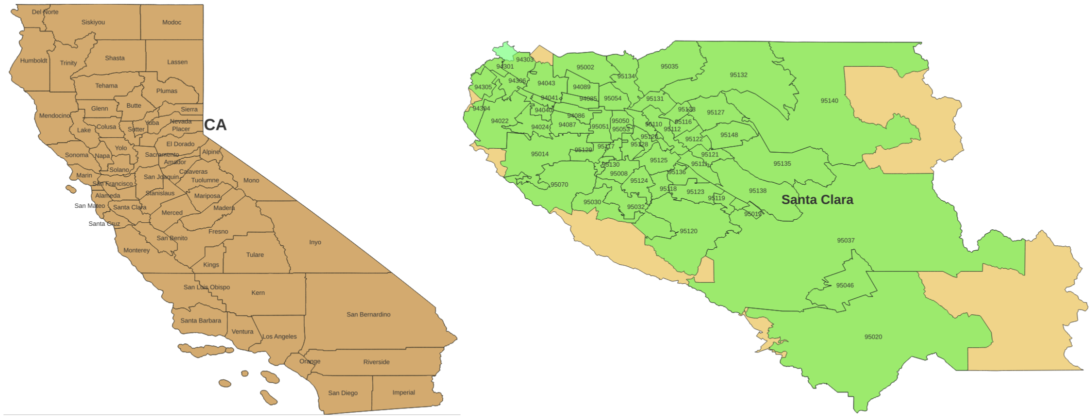
Our generated benchmarking datasets comprise semantically meaningful questions (and answers), which are produced from raw spatial data. Specifically, the questions focus on the relative locations of geographic divisions within the United States, such as states and counties. Each dataset contains 1,000 questions of the same type (Yes/No, Single-Choice, or Multi-Choice) and includes the correct answers for training and evaluation purposes. Table 3 summarizes these question types along with an example for each case.

6.1 Source Data

We used the TIGER 2015 real-world datasets [8, 41] for U.S. States (50 entities), Counties (3,225 entities), and Zipcodes (33,144 entities) as the foundation for generating our spatial questions. Figure 10 presents example entities from these datasets to illustrate the relative scale of the different geographic divisions. All entities have non-overlapping areas within the same type (e.g., State borders do not intersect with other States), as these geographic divisions are defined by official geopolitical boundaries established by governmental authorities. In the United States, such boundaries are determined and maintained by entities like the U.S. Census Bureau and state legislatures, ensuring that units like Zipcodes, Counties, and States are usually hierarchically ordered and divisions of the same type are mutually exclusive in area.

Table 3. Overview of the different question types in the dataset, each illustrated with an example.

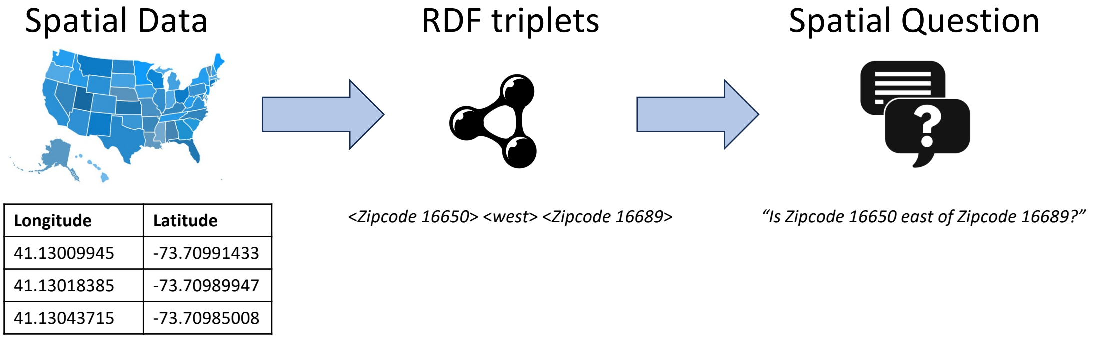
Type	**Example Question**	**Example Answer(s)**
Yes/No	Is Zipcode 62546 northeast of Zipcode 62560?	yes
Radio	Select exactly one option (a-e) that best describes the relationship of Douglas County Washington in relation to Zipcode 98848 in terms of geography. Options: a. contains b. adjacent to and northwest of c. adjacent to and north of d. southeast of e. none of the above	e
Checkbox	Select all options that are adjacent to and west of The State of Illinois? You may choose one or more options. Options: a. Zipcode 47993 b. Zipcode 47991 c. Zipcode 63782 d. Zipcode 62324 e. None of the above	a,b



(a) The State of California and its Counties

(b) Santa Clara County, CA and its Zipcodes

Fig. 10. Illustrative example showing the State of California and its Counties (a), with a focus on the Zipcodes within Santa Clara County, CA (b), to highlight the relative scale of the geographic divisions in our source data. (Visualized on QGIS [1].)



Longitude	**Latitude**
41.13009945	-73.70991433
41.13018385	-73.70989947
41.13043715	-73.70985008

<Zipcode 16650> <west> <Zipcode 16689>

"Is Zipcode 16650 east of Zipcode 16689?"

Fig. 11. Overview of our pipeline for generating spatial questions from raw geographic data. The process involves calculating spatial relationships and transforming them into RDF triplets using $SpaTex$. Then, structured yes/no, single-choice, and multiple-choice questions are randomly generated from the set of RDF triplets.

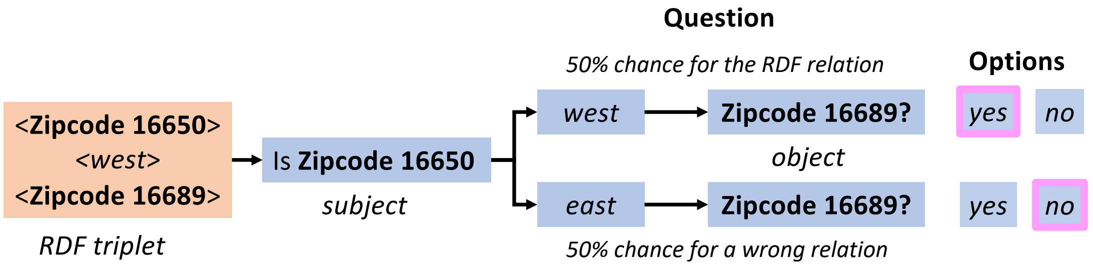


Fig. 12. Example of Yes/No random question generation from a randomly sampled RDF triplet. The correct answer is indicated with a bolded boundary.

6.2 Construction

To generate semantically meaningful questions grounded in the data stored in our database, we leverage the same set of RDF triples produced by $SpaTex$ during SpaRAGraph's pre-processing phase. From these triplets, we automatically generate randomized natural language questions tailored to binary, multiclass, and multilabel classification tasks, all framed as question–answer (QA) pairs. An overview of this process is shown in Figure 11. The first step in the pipeline (transforming spatial data into RDF triplets) is performed by $SpaTex$.

6.2.1 Yes/No - Binary Classification. Figure 12 illustrates the generation process of questions with Yes/No answers. For a randomly selected triplet out of the available ones (e.g., the leftmost box in Figure 12), we assign a Yes/No answer with 50-50 chance. For questions with a positive answer, we simply transform the RDF triplet in the form of question. For example, the triplet <Santa Clara County California> <south> <Alameda County, California> becomes the question: *Is Santa Clara County, California south of Alameda County, California?* For generating questions with a negative answer, we retain the same entities but alter the relation to its semantic opposite. In the case of cardinal directions, we use the inverse (e.g., 'north' becomes 'south', 'southwest' becomes 'northeast'). For topological relations, we replace each with its logical counterpart. For example, 'contains' becomes 'is inside of' and vice versa. Regarding adjacency, since it always appears with a cardinal direction, we keep the adjacency term unchanged and invert only the cardinal direction for simplicity. This approach ensures that the resulting question is definitely false while preserving grammatical and semantic coherence. The phrasing of the questions is rule-based, ensuring that all questions follow a consistent syntactic structure.

6.2.2 Radio - Multiclass Classification. To generate multiple-choice questions, we begin by randomly sampling a triplet from the available dataset. Each question includes five answer options, labeled (a) through (e), with only one being correct based on the sampled triplet. Option (e) is always reserved for 'none of the above', allowing for cases where the correct answer is intentionally excluded from the listed options.

We introduce such cases with a 20% probability. In these instances, four incorrect relations are randomly selected with equal probability from the available pool (excluding the correct one), shuffled, and assigned to options (a) through (d), making (e) the correct answer. In the remaining 80% of the cases, where the correct relation is included, we randomly select three incorrect relations (excluding the correct one), shuffle them along with the correct relation, and assign them to options (a) through (d), with (e) remaining as 'none of the above'. This approach ensures that the correct answer may appear in any of the five options.

The phrasing of Radio questions is always in the following format: *Select exactly one option (a-e) that best describes the relationship of <subject> in relation to <object> in terms of geography. Options: a.* $<relation_1>$ *b.* $<relation_2>$ *c.*

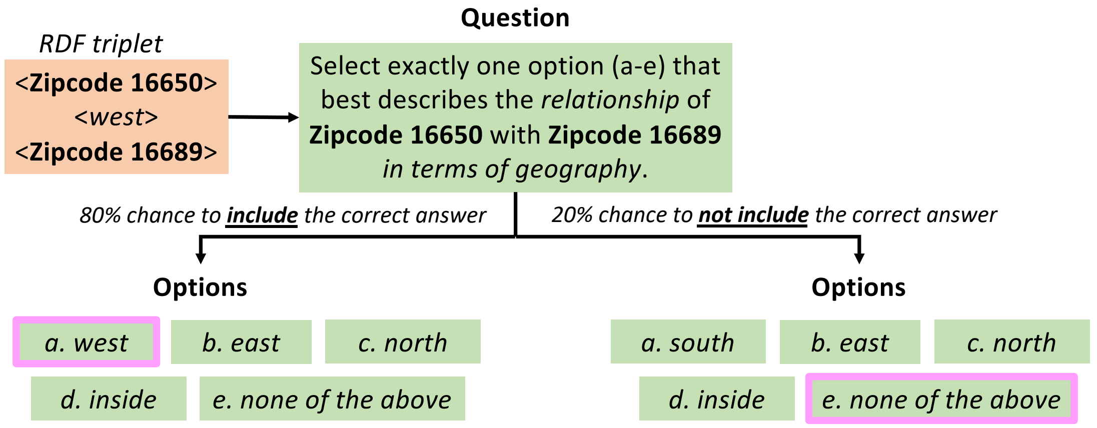

Question
RDF triplet
<Zipcode 16650>
<west>
<Zipcode 16689>
Select exactly one option (a-e) that best describes the relationship of Zipcode 16650 with Zipcode 16689 in terms of geography.
80% chance to include the correct answer
20% chance to not include the correct answer
Options
a. west
b. east
c. north
d. inside
e. none of the above
Options
a. south
b. east
c. north
d. inside
e. none of the above

Fig. 13. Radio (single-choice) random question generation from a randomly selected RDF triplet. The correct answer is indicated with a bolded boundary.

<relation$_3$> d. <relation$_4$> e. none of the above. In this format, the instruction is embedded within the question itself, while the answer choices are presented separately, with exactly one correct option among (a) through (e).

6.2.3 Checkbox - Multilabel Classification. For checkbox queries, we first construct a relation map for each subject entity before sampling a random triplet. Specifically, given an RDF triplet <subject> <predicate> <object>, we identify all objects in the dataset that share the same relation *predicate* with the *subject*. This results in a grouped set of related entities per relation, making it easier to identify and use them as multi-label answer options.

Next, we sample a random triplet and use the relation map to select a random relation for the triplet's subject, along with all objects that share that relation with the subject. This set forms the pool of correct answer options for the query. To construct the incorrect options, we gather all other objects that are related to the same subject via a different relation. These objects serve as distractors, as they do not satisfy the target relation specified in the sampled triplet.

We then randomly determine the number of correct answers c to include in the question, selecting between 0 and 4 options from the pool of correct entities. If the pool contains fewer than c entities, c is reduced incrementally until sampling is possible. The pool is guaranteed to contain at least one entity; the object of the RDF triple that initially introduced the target relation into the pool.

The remaining options are filled with randomly chosen distractors from the incorrect set, and all answer choices are shuffled. Option (e) is always 'none of the above', enabling the inclusion of questions with no correct answer. For example, Figure 14 illustrates a case in which an entity (Zipcode 16650) appears as the subject in two triplets involving the *west* relation. Based on these triplets, a checkbox query is randomly generated to include zero, one, or both of the correct objects among the answer options.

7 Experimental Evaluation

Question Answering (QA) To assess the performance of SpaRAGraph, we evaluate it on binary, multiclass, and multilabel classification tasks using SRB, introduced in Section 6. For each model, we run every question three times and retain the most representative response, as smaller models often produce varied outputs across runs. Evaluating the spatial inference capabilities of SpaRAGraph on open-ended, complex spatial questions is inherently challenging, as it would

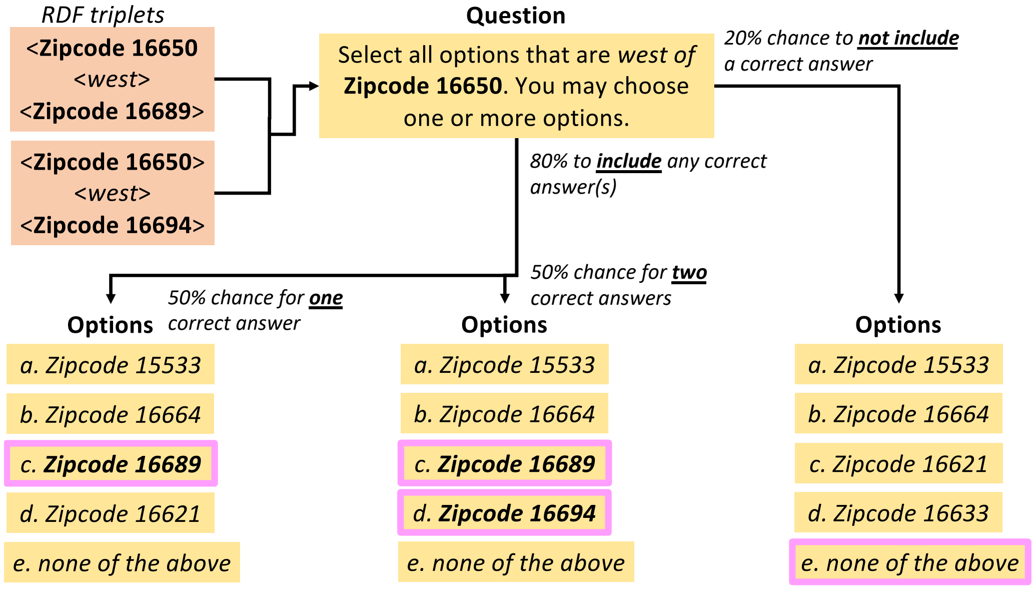

RDF triplets
<Zipcode 16650 <west> <Zipcode 16689>
<Zipcode 16650> <west> <Zipcode 16694>
Question
Select all options that are west of Zipcode 16650. You may choose one or more options.
20% chance to not include a correct answer
80% to include any correct answer(s)
50% chance for one correct answer
50% chance for two correct answers
Options
a. Zipcode 15533
b. Zipcode 16664
c. Zipcode 16689
d. Zipcode 16621
e. none of the above
Options
a. Zipcode 15533
b. Zipcode 16664
c. Zipcode 16689
d. Zipcode 16694
e. none of the above
Options
a. Zipcode 15533
b. Zipcode 16664
c. Zipcode 16621
d. Zipcode 16633
e. none of the above

Fig. 14. Checkbox (multiple-choice) random question generation from all triplets regarding a randomly selected entity (Zipcode 16650). The correct answers are shown with a boundary.

require textual ground truth annotations and expert evaluators to assess the correctness of generated responses. To date, no evaluation framework has been established with proven effectiveness for open-ended spatial reasoning tasks. Additionally, the StepGame benchmark [21] is comprised of randomly generated, locally-scoped synthetic entities, which prevents the use of a global index (such as our RDF graph) for capturing spatial relations at a broader, global level. Moreover, StepGame focuses solely on multilabel classification. Hence, SRB provides a practical foundation for preliminary yet detailed evaluation, focusing on structured but non-trivial tasks such as binary, multiclass, and multilabel classification over real-world ontologies.

Subject Entities States in the U.S. cover much larger areas than Counties or Zipcodes, and their names are more widely recognized beyond local contexts. Consequently, they are more likely to be referenced in bibliographic sources, online content, or other types of text. Such widely available geographic information increases the likelihood that topological relationships between states appeared in the documents used to train certain foundation models. Table 4 breaks down the datasets by the types of entities referenced in the questions. Yes/No and Radio questions always involve a relation between two entities (which may be of the same or different types), whereas Checkbox questions refer to a single entity (see Figures 12-14). Additionally, approximately half of the questions in each dataset pertain to Zipcodes, which is due to the uniform sampling from the source data to generate the questions, which comprise a significantly higher number of Zipcodes (33,144) compared to Counties (3,225) or States (50).

Embeddings & Indexing We implement SpaRAGraph in Python 3.9. All RDF-related operations such as reading, writing, indexing the graph and path finding, are handled using the RDFLib library. To embed both the RDF graph's node labels and the extracted entities from user questions, we employ Sentence Transformers [34], specifically the all-MiniLM-L6-v2 model. For approximate k-nearest neighbors (akNN) search over these embeddings, we use FAISS [18] for efficient indexing and similarity-based querying.

Table 4. Breakdown of question datasets by type of entities referenced in questions.

	States only	Counties only	Zipcodes only	State-County	State-Zipcode	Zipcode-County
Yes/No	N/A	3.5%	56.2%	0.8%	9.6%	28.0%
Radio	0.1%	5.7%	55.0%	1.3%	9.8%	27.1%
Checkbox	12.4%	35.6%	50.5%	N/A	N/A	N/A

Table 5. The models that we tested on SpaRAGraph.

Model	# of Parameters	Quantization
meta-llama/Llama3.1-8B-Instruct [7]	8B	4 bits
mistralai/Mistral-7B-Instruct [17]	7B	4 bits
Qwen/Qwen2.5-7B-Instruct [43]	7B	8 bits
lzq677/GeoCode-GPT-7B [15]	7B	4 bits

Models & Setup In our experiments, we test the models shown in Table 5 employed with and without SpaRAGraph. Specifically, GeoCode-GPT-7B is a Llama-7B variant fine-tuned for geospatial code generation, a task not perfectly aligned with our evaluation but still broadly relevant. We focused on small, open-source models because they provide a cost-effective and publicly accessible option for using SpaRAGraph, even on commodity GPUs. Our argument is that if SpaRAGraph can significantly enhance the performance of these smaller models on our RAG benchmark tasks, then larger or closed models may not be necessary for many practical applications. This approach prioritizes accessibility and reproducibility, demonstrating that spatially informed reasoning can be effectively achieved without relying on resource-intensive models. All our experiments were conducted single-threaded on a machine with Ubuntu 20.04 OS, 64GB of memory and an Intel Core i9 3.60GHz processor. The models were loaded in an NVIDIA GeForce RTX 3060 with 12GB of memory.

To ensure statistical reliability, all generation experiments were conducted three times, with the reported scores representing the average across these runs. The total execution times for running each dataset once with and without SpaRAGraph is shown in Table 6. Notably, the task complexity affects the total execution time, as simpler tasks such as Yes/No questions are processed more quickly, while more complex formats like Radio and Checkbox questions require longer inference times. Using SpaRAGraph introduces a time overhead of approximately 6% to 29%. This increase is partly due to the RAG process (which includes the NER, node matching, graph traversal, and context generation) and partly due to longer generation times caused by the expanded prompt size.

Most individual components in SpaRAGraph can be independently optimized, further enhancing overall performance thanks to its highly modular architecture. For instance, incorporating state-of-the-art graph indexes and path traversal algorithms can significantly reduce the time required for context generation. Moreover, both accuracy and efficiency can be improved by integrating more advanced NER or similarity search techniques. Ultimately, the primary bottleneck in the pipeline tends to be the model's response generation, rather than any of SpaRAGraph's internal modules.

7.1 SpaRAGraph Generation Evaluation

To evaluate model performance across the three classification tasks, we apply task-appropriate metrics. For binary (Yes/No) questions, we report Precision (P), Recall (R), and F1 score. For multiclass (Radio) questions, where only one option is correct, we use macro-averaged Precision, Recall, and F1 score to account for class imbalance. For multilabel (Checkbox) questions, we adopt a sample-based evaluation: each question is treated as a separate sample, and Precision,

Table 6. Comparison of models' average execution time after three runs, in seconds ± the standard error of the mean rounded up, on the Spatial Reasoning Benchmark, with and without SpaRAGraph in zero-shot setting.

Model	**Yes/No (✓/✗)**	**Radio (●)**	**Checkbox [x]**
Llama-3.1-8B-Instruct	248±2	286±4	434±4
+ SpaRAGraph	**278±2**	**328±2**	**473±1**
Mistral-7B-Instruct	315±1	360±1	509±1
+ SpaRAGraph	**334±2**	**429±1**	**655±1**
Qwen2.5-7B-Instruct	316±1	328±4	461±2
+ SpaRAGraph	**351±1**	**381±3**	**544±5**
GeoCode-GPT-7B	224±1	229±5	229±1
+ SpaRAGraph	**242±4**	**355±3**	**441±4**

Table 7. Performance comparison of models on the Spatial Reasoning Benchmark, with and without SpaRAGraph in zero-shot setting. The scores (%) are the averages across three different response generation cycles and the standard error of the mean across all scores does not exceed 1%.

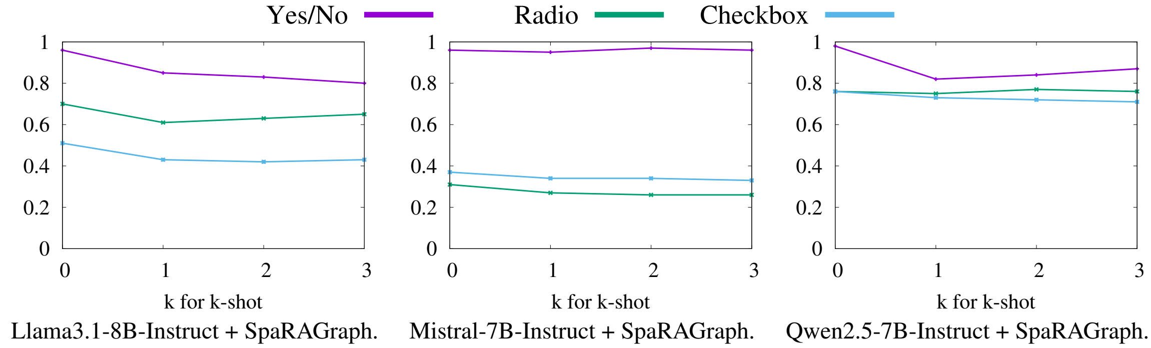
	Yes/No (✓/✗)			**Radio (●)**			**Checkbox [x]**		
Model	P	R	F1	macro-P	macro-R	macro-F1	sample-P	sample-R	sample-F1
Llama-3.1-8B-Instruct	50	80	61	23	21	17	24	50	30
+ SpaRAGraph	**94**	**98**	**96**	**73**	**70**	**70**	**44**	**68**	**51**
Mistral-7B-Instruct	52	50	51	23	21	14	24	45	29
+ SpaRAGraph	**93**	**98**	**96**	**57**	**34**	**31**	**30**	**58**	**37**
Qwen2.5-7B-Instruct	83	22	35	27	27	26	34	50	39
+ SpaRAGraph	**100**	**97**	**98**	**83**	**82**	**76**	**72**	**85**	**76**
GeoCode-GPT-7B	55	29	38	18	20	16	31	42	33
+ SpaRAGraph	**94**	**98**	**96**	**37**	**33**	**33**	**29**	**43**	**32**

Recall, and F1 score are computed by comparing the predicted set of labels to the ground truth for that sample. These metrics are then averaged across all samples, allowing us to assess how accurately the model predicts multiple relevant labels per question, while also capturing partial correctness. Model generation temperature was set to 0.7 across all runs.

7.1.1 Effectiveness.

We experimentally demonstrate the effectiveness of SpaRAGraph on all models by comparing its response accuracy across all three tasks to the accuracy of the same models when used out-of-the-box, in a zero-shot setting. Table 7 summarizes the performance of all evaluated models across the three tasks, highlighting the significant improvements achieved when augmented with SpaRAGraph.

In particular, all models exhibit the most substantial gains on the Yes/No (binary) classification task achieving near-perfect F1 scores compared to their standalone performance, which ranges from 35% to 61%. In Radio questions, which pose a greater challenge for LLMs than binary (Yes/No) queries due to their multi-class, single-answer nature, the improvements introduced by SpaRAGraph are great for both Llama-3.1-8B and Qwen2.5-7B. However, Mistral-7B and GeoCode-GPT-7B struggle even when assisted by SpaRAGraph, suggesting that the increased complexity of selecting a single correct option among several may exceed its reasoning capabilities, despite the additional context. A similar trend is observed in Checkbox questions, where the improvements from SpaRAGraph are somewhat smaller but present. Overall, Qwen2.5-7B consistently outperforms the other models and has the highest performance gains when employed



Llama3.1-8B-Instruct + SpaRAGraph. Mistral-7B-Instruct + SpaRAGraph. Qwen2.5-7B-Instruct + SpaRAGraph.

Fig. 15. F1, macro-F1 and sample-F1 scores (Y axis) of each model for Yes/No, Radio and Checkbox questions, respectively, in k-shot settings ($k \in [0, 3]$).

with SpaRAGraph, highlighting the effectiveness of the generated context in guiding its responses more accurately. Additionally, while GeoCode-GPT-7B is fine-tuned for geospatial code generation, it underperforms general-purpose models on QA and classification tasks involving spatial reasoning. We therefore exclude GeoCode-GPT-7B from the remainder of our experiments, as it is evident that it offers no advantage over the base models.

7.1.2 Zero- vs Few-shot. We evaluated SpaRAGraph under zero-shot, one-shot, two-shot, and three-shot settings using prompts with static examples (shots). Due to the nature of our tasks and the inherent structure of spatial relations, these examples do not function as context in the conventional sense, meaning that they lack factual information about the specific entities involved in the test queries. Instead, the examples are intended to help models infer relational patterns, such as recognizing inverse spatial relationships. For instance, a static example might be: *"Entity A is south of Entity B. Where is B relative to A? // B is north of A."* Figure 15 illustrates the limited benefits (that worsen as the number of examples increases) which few-shot learning brings when combined with SpaRAGraph. While static examples can occasionally guide the models toward generating correct responses, they generally lack the contextual grounding that SpaRAGraph provides. In many cases, these examples confuse the model more than they help. We omit results from few-shot learning with randomly selected examples, as these are almost sure to introduce irrelevant information to each question, further impairing model performance.

In general, SpaRAGraph is specifically designed to be effective with out-of-the-box, untuned models operating in a zero-shot setting. Its structured guidance and contextual grounding are sufficient to elicit accurate and relevant responses from the model without the need for additional examples. As such, few-shot learning becomes unnecessary, as SpaRAGraph already provides the essential task understanding and relational context that examples would otherwise aim to convey. However, as spatial questions grow more complex, few-shot learning may offer benefits by guiding the model on how to reason with the generated context.

7.1.3 RAG Evaluation. We compare SpaRAGraph against a custom baseline RAG approach. In this baseline, RDF triplets are vectorized, and similarity search is performed between the embedding of a user query and the embeddings of the RDF triplets. The top-k most relevant triplets are then retrieved and provided to the model as contextual information alongside the original query for response generation.

This baseline is designed in line with prior work on natural language querying of graphs [10, 49]. Although much of this literature focuses on tasks such as link prediction, edge existence or cycle check, it shares a key characteristic with

Table 8. Performance comparison of the Llama-3.1-8B-Instruct model on the Spatial Reasoning Benchmark, between SpaRAGraph and the baseline RAG method in zero-shot setting. The scores (%) are the averages across three different response generation cycles and the standard error of the mean across all scores does not exceed 1%.

	Yes/No (✓/✗)			**Radio (●)**			**Checkbox [x]**		
Model	P	R	F1	macro-P	macro-R	macro-F1	sample-P	sample-R	sample-F1
Llama-3.1-8B-Instruct	50	80	61	23	21	17	24	50	30
+ Baseline-1	73	62	67	40	23	17	33	50	37
+ Baseline-5	63	80	71	47	25	19	37	61	42
+ Baseline-10	62	85	72	45	25	19	36	63	43
+ Baseline-20	67	88	76	42	29	28	35	67	43
+ Baseline-50	61	87	72	40	27	26	33	**70**	42
+SpaRAGraph	**94**	**98**	**96**	**73**	**70**	**70**	**44**	68	**51**

our task: effective prompt engineering is required to formulate queries that are meaningful with respect to the graph's nodes, edges, and underlying semantics (i.e. the spatial topology of geo-entities). In our setting, the SRB datasets already contain questions derived directly from the RDF triplets and therefore adhere to the insights established in prior work.

Before inference time, each RDF triplet is first converted into a concise natural language sentence by removing URI-specific formatting and concatenating the subject, predicate, and object into a textual expression of their relation. These sentences (we hereby refer to as *relation texts*, as they refer to spatial relations in natural language), are embedded using the sentence-transformers/all-MiniLM-L6-v2 model [37] and indexed with FAISS [18] to enable efficient similarity-based retrieval.

In Table 8, we compare the effectiveness of SpaRAGraph on the SRB datasets against this baseline by retrieving the top-k most relevant relations per query, where $k \in 1, 5, 10, 20, 50$. All approaches were evaluated with the Llama-3.1-8B-Instruct model in a zero-shot setting. Note that both SpaRAGraph and the baselines index and query the same RDF triplets. The baseline RAG method for the different values of k improves the model on the benchmark, but SpaRAGraph achieves a significantly higher Precision, Recall and F1 across all datasets.

A key observation is that as the number of retrieved relation texts k increases, the baseline method's accuracy (measured by F1 score) initially improves but eventually degrades. This suggests that while adding relational context is beneficial, in principle, a large volume of retrieved relations introduces also unnecessary information that can confuse the model rather than assist it.

This baseline retrieval strategy is effective when an explicit relation directly connects the queried entities already in the dataset (which is largely the case in the SRB datasets), but it becomes highly prone to inaccuracy on more complex queries that require multi-hop reasoning across multiple relational paths.

Thus, a major limitation of the baseline method compared to SpaRAGraph arises in cases where answering a query requires combining information from multiple relation texts. Retrieving only the top-k relation texts most similar to the query does not guarantee that sufficient information will be available for generating precise context. Consider the example query: "What is the relative location of A to C?", which refers to two spatial entities, A and C. Suppose the available relation texts include (among others) the following subset: "A is north of B", "B is north of C", "A is south of D", "C is west of E", "A is south of F", "C is east of G". The correct answer can be inferred by jointly considering the first two relations. However, when $k = 2$, there is no guarantee that these two relation texts will be retrieved, since most relations share similar semantic similarity with the query by mentioning only one of the two entities.

Dataset	MRR	FMR	Precision
Yes/No	0.99	0.97	0.99
Radio	1.00	0.98	0.99
Checkbox	1.00	0.97	0.99

Table 9. SpaRAGraph's retrieval accuracy expressed through its MRR, FMR (percentage of prompts that had all their entities retrieved correctly) and Precision (percentage of all retrieved entities that are correct for their respective prompts). Metrics are shown per individual dataset, as Yes/No and Radio questions always regard two entities each, whilst SRB's Checkbox questions always pertain five entities.

Furthermore, a consistent pattern across datasets is that as the value of k increases, Precision decreases while Recall increases. The drop in Precision is already reflected in the corresponding decline in overall accuracy, as the larger volume of contextual relations introduces noise that confuses the model and leads to a higher number of false positives. At the same time, the expanded context enables the model to capture more relevant information resulting in higher Recall, with the baseline even surpassing SpaRAGraph in that metric on the Checkbox dataset for $k = 50$.

However, this trade-off between Precision and Recall ultimately degrades overall performance. Although adding contextual triplets yields some initial improvement, performance remains largely insensitive to further increases in k. In other words, simply increasing the number of relation texts included in the context does not meaningfully improve the model's ability to generate correct responses.

Addressing these limitations is precisely the motivation behind SpaRAGraph, which deterministically composes RDF triplets to extract factually correct information and generate a compact and concise spatial context that substantially benefits the model.

7.2 Retrieval Accuracy

SpaRAGraph's retrieval accuracy primarily depends on two factors: a) the effectiveness of NER, i.e., extracting all relevant entities from a prompt, and b) the accuracy of matching these entities to their corresponding nodes in the graph via similarity search. For NER, we use spaCy, which supports custom rule-based entity extraction tailored to the specific characteristics of the data. For similarity search, we leverage FAISS, a library known for its high-performance and accurate vector similarity search, enabling fast and precise entity-node matching in the graph. The tasks of accurately extracting entities from text and identifying the most similar nodes in the graph are orthogonal to SpaRAGraph's core functions of path traversal and context generation. Therefore, state-of-the-art solutions (such as spaCy for NER or FAISS for similarity search) can be integrated to minimize retrieval inaccuracies without interfering with the framework's primary operations. This compartmentalization allows for easier tuning of the various components in SpaRAGraph.

To thoroughly evaluate SpaRAGraph on retrieval accuracy, we employ three key metrics:

- Mean Reciprocal Rank (MRR) – This metric assesses the rank of the ground truth (i.e., the correct entities) within the list of retrieved entities. It is calculated as the reciprocal of the rank of the first correct entity, averaged across all questions.
- Full Match Rate (FMR) – This measures the percentage of prompts for which all related entities were correctly retrieved, with no omissions.
- Precision – This reflects the proportion of retrieved entities that are correct, indicating the accuracy of the results with respect to the entities relevant to each prompt.

Table 9 presents the performance of SpaRAGraph in accurately retrieving the target entities used for graph traversal across each dataset in the SRB. SpaRAGraph achieves near-perfect scores in MRR, FMR, and Precision across all datasets, demonstrating robust entity matching performance, even in scenarios where entities have highly similar names (e.g., Zipcode 00951 vs. Zipcode 00952). The very few retrieval errors observed typically involve such edge cases. For instance, in questions involving counties in West Virginia, the entity recognizer (spaCy) may include the state name alongside county names, resulting in additional graph paths being explored and introducing noise into the generated context. Even these few issues, however, could be easily mitigated through refinements, e.g., calibrating spaCy or adopting more standardized naming conventions for RDF subjects and graph nodes.

Optimizing NER and entity-to-node similarity search is beyond the scope of this work. Nonetheless, existing state-of-the-art solutions for both tasks are readily compatible with SpaRAGraph and can be integrated to further enhance its performance.

8 Conclusions

This study introduces SpaRAGraph, a novel approach for enhancing the ability of LLMs to answer spatial queries using RAG. Our method is applicable to a wide range of spatial data, including existing RDF datasets. However, several dataset-specific pre-processing steps are required before loading any spatial RDF dataset into SpaRAGraph. First, the NER module must be configured to recognize all types of spatial entities present in the dataset (e.g., cities, countries, or other domain-specific naming conventions). In addition, a relation composition matrix must be defined (similarly to Table 2) to specify semantically meaningful transitions between all relations and predicates, thereby enabling effective graph traversal and rule-based relation composition. The $SpaTex$ module is dataset-agnostic: it operates on any spatial data in WKT format and automatically generates the spatial RDF enriched with the cardinal direction and topological relations used in this paper. As a result, any publicly available WKT-formatted spatial dataset can be used directly in SpaRAGraph without requiring additional pre-processing Our experimental analysis shows that SpaRAGraph improves the response generation of three LLMs for spatial questions by 35-63 points (47 average) in binary, 17-53 points (40 average) in multiclass and 8-37 points (23 average) in multilabel classification tasks.

We also introduce the first spatial reasoning datasets, along with a benchmark designed to evaluate model performance on binary, multiclass, and multilabel classification tasks involving spatial questions. Our datasets are grounded in accurate, real-world topological relationships among States, Counties, and Zipcodes in the U.S., enabling robust evaluation of spatial reasoning capabilities in large language models in both RAG and non-RAG scenarios.

The ultimate goal of this work is to evaluate models on open-ended spatial reasoning tasks, allowing evaluators to assess model predictions against textual ground truths while examining the evaluators' own effectiveness and accuracy. In theory, a future version of SpaRAGraph could employ dedicated classifiers to categorize questions by task type (e.g., binary, open-ended, etc.) and apply specialized methods accordingly. For instance, open-ended questions involving a single entity and a topological relation or direction could be answered by traversing the RDF graph starting from the specified entity, following only those edges that satisfy the relational constraints, up to a predefined traversal threshold.

We also plan to expand our datasets to include global coverage and incorporate additional entity types (such as lakes, parks and rivers) and more spatial relation types (such as distance and area metrics). This will increase both the diversity and complexity of spatial relationships, enabling more rigorous evaluation of spatial inference across varied data types. If the resulting RDF graphs are too large to fit in memory, standard hybrid disk–memory graph-database and indexing techniques can be employed as a performance trade-off. As with most geospatial datasets, the input data is typically static and not subject to frequent updates. Nevertheless, changes to the original data can be propagated to regenerate the corresponding RDF triples and selectively update SpaRAGraph's graph index, without requiring full reprocessing of the entire dataset. Additionally, we plan to enhance the robustness and accuracy of our composition matrix (Table 2) and grid traversal strategy by incorporating probabilistic methods for aggregating relations across longer paths.

Supplementary Material. We share our code in GitHub in the following repository link: https://github.com/ThanGeo/SpaRAGraph. Additionally, we make our complete spatial reasoning benchmark publicly available on HuggingFace: https://huggingface.co/datasets/Rammen/SpatialReasoning and the individual datasets grouped per question type in the following repository: https://github.com/ThanGeo/spatial-inference-benchmark.

Acknowledgments. This work has been supported by project MIS 5154714 of the National Recovery and Resilience Plan Greece 2.0 funded by the European Union under the NextGenerationEU Program.

References

[1] 2024. QGIS. www.qgis.org.

[2] Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. 2024. Benchmarking Large Language Models in Retrieval-Augmented Generation. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (Eds.). AAAI Press, 17754–17762.

[3] Eliseo Clementini, Paolino Di Felice, and Peter van Oosterom. 1993. A Small Set of Formal Topological Relationships Suitable for End-User Interaction. In *SSD (Lecture Notes in Computer Science)*. Springer.

[4] Eliseo Clementini, Jayant Sharma, and Max J. Egenhofer. 1994. Modelling topological spatial relations: Strategies for query processing. *Comput. Graph.* 18, 6 (1994), 815–822.

[5] Anthony G. Cohn and José Hernández-Orallo. 2023. Dialectical language model evaluation: An initial appraisal of the commonsense spatial reasoning abilities of LLMs. *CoRR* abs/2304.11164 (2023).

[6] Mark de Berg. 1997. *Computational geometry: algorithms and applications*. Springer.

[7] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. 2024. The Llama 3 Herd of Models. *CoRR* abs/2407.21783 (2024).

[8] Ahmed Eldawy and Mohamed F. Mokbel. 2015. SpatialHadoop: A MapReduce Framework for Spatial Data. In *31st IEEE International Conference on Data Engineering, ICDE 2015, Seoul, South Korea, April 13-17, 2015*. 1352–1363.

[9] 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 *ACM SIGKDD*.

[10] Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. 2023. Talk like a graph: Encoding graphs for large language models. *arXiv preprint arXiv:2310.04560* (2023).

[11] Robert Friel, Masha Belyi, and Atindriyo Sanyal. 2024. RAGBench: Explainable Benchmark for Retrieval-Augmented Generation Systems. *CoRR* abs/2407.11005 (2024).

[12] Thanasis Georgiadis and Nikos Mamoulis. 2023. Raster Intervals: An Approximation Technique for Polygon Intersection Joins. In *ACM SIGMOD*.

[13] 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. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

[14] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python. doi:10.5281/zenodo.1212303

[15] Shuyang Hou, Zhangxiao Shen, Anqi Zhao, Jianyuan Liang, Zhipeng Gui, Xuefeng Guan, Rui Li, and Huayi Wu. 2025. GeoCode-GPT: A large language model for geospatial code generation. *Int. J. Appl. Earth Obs. Geoinformation* 138 (2025), 104456.

[16] Yulong Hui, Yao Lu, and Huanchen Zhang. 2024. UDA: A Benchmark Suite for Retrieval Augmented Generation in Real-world Document Analysis. *CoRR* abs/2406.15187 (2024).

[17] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. *CoRR* abs/2310.06825 (2023).

[18] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data* (2019).

[19] Siqi Lai, Yansong Ning, Zirui Yuan, Zhixi Chen, and Hao Liu. 2025. USTBench: Benchmarking and Dissecting Spatiotemporal Reasoning of LLMs as Urban Agents. *CoRR* abs/2505.17572 (2025).

[20] Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *NeurIPS*.

[21] 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 *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*. AAAI Press, 18500–18507.

[22] Wenbin Li, Di Yao, Ruibo Zhao, Wenjie Chen, Zijie Xu, Chengxue Luo, Chang Gong, Quanliang Jing, Haining Tan, and Jingping Bi. 2025. STBench: Assessing the Ability of Large Language Models in Spatio-Temporal Analysis. In *Companion Proceedings of the ACM on Web Conference 2025, WWW 2025, Sydney, NSW, Australia, 28 April 2025 - 2 May 2025*. ACM, 749–752.

[23] Zhenlong Li and Huan Ning. 2023. Autonomous GIS: the next-generation AI-powered GIS. *Int. J. Digit. Earth* (2023).

[24] Rohin Manvi, Samar Khanna, Gengchen Mai, Marshall Burke, David B. Lobell, and Stefano Ermon. 2024. GeoLLM: Extracting Geospatial Knowledge from Large Language Models. In *ICLR*. OpenReview.net.

[25] Costas Mavromatis and George Karypis. 2024. GNN-RAG: Graph Neural Retrieval for Large Language Model Reasoning. *CoRR* abs/2405.20139 (2024).

[26] OpenStreetMap contributors. 2017. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org.

[27] Jignesh M. Patel and David J. DeWitt. 1996. Partition Based Spatial-Merge Join. In *ACM SIGMOD*.

[28] Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, and Siliang Tang. 2024. Graph Retrieval-Augmented Generation: A Survey. *CoRR* abs/2408.08921 (2024).

[29] Nicholas Pipitone and Ghita Houir Alami. 2024. LegalBench-RAG: A Benchmark for Retrieval-Augmented Generation in the Legal Domain. *CoRR* abs/2408.10343 (2024).

[30] Jianzhong Qi, Zuqing Li, and Egemen Tanin. 2023. MaaSDB: Spatial Databases in the Era of Large Language Models (Vision Paper). In *ACM SIGSPATIAL*.

[31] Pengrui Quan, Brian Wang, Kang Yang, Liying Han, and Mani Srivastava. 2025. Benchmarking Spatiotemporal Reasoning in LLMs and Reasoning Models: Capabilities and Challenges. *CoRR* abs/2505.11618 (2025).

[32] David A. Randell, Zhan Cui, and Anthony G. Cohn. 1992. A Spatial Logic based on Regions and Connection. In *Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR'92). Cambridge, MA, USA, October 25-29, 1992*. Morgan Kaufmann, 165–176.

[33] David Rau, Hervé Déjean, Nadezhda Chirkova, Thibault Formal, Shuai Wang, Vassilina Nikoulina, and Stéphane Clinchant. 2024. BERGEN: A Benchmarking Library for Retrieval-Augmented Generation. *CoRR* abs/2407.01102 (2024).

[34] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. https://arxiv.org/abs/1908.10084

[35] Nicole R. Schneider, Nandini Ramachandran, Kent O'Sullivan, and Hanan Samet. 2025. DistRAG: Towards Distance-Based Spatial Reasoning in LLMs. *CoRR* abs/2506.03424 (2025).

[36] Priyanka Sen, Sandeep Mavadia, and Amir Saffari. 2023. Knowledge Graph-augmented Language Models for Complex Question Answering. *Proceedings of the 1st Workshop on Natural Language Reasoning and Structured Explanations (NLRSE)* (2023).

[37] SentenceTransformers. 2024. *miniLM*. https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

[38] Zhengxiang Shi, Qiang Zhang, and Aldo Lipani. 2022. StepGame: A New Benchmark for Robust Multi-Hop Spatial Reasoning in Texts. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. AAAI Press, 11321–11329.

[39] Simranjit Singh, Michael Fore, and Dimitrios Stamoulis. 2024. GeoLLM-Engine: A Realistic Environment for Building Geospatial Copilots. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024 - Workshops, Seattle, WA, USA, June 17-18, 2024*. IEEE, 585–594.

[40] Spiros Skiadopoulos and Manolis Koubarakis. 2004. Composing cardinal direction relations. *Artif. Intell.* 152, 2 (2004), 143–171.

[41] SpatialHadoop. 2015. *TIGER datasets*. http://spatialhadoop.cs.umn.edu/datasets.html

[42] Yixuan Tang and Yi Yang. 2024. MultiHop-RAG: Benchmarking Retrieval-Augmented Generation for Multi-Hop Queries. *CoRR* abs/2401.15391 (2024).

[43] Qwen Team. 2024. Qwen2.5: A Party of Foundation Models. https://qwenlm.github.io/blog/qwen2.5/

[44] Yuqing Wang and Yun Zhao. 2024. TRAM: Benchmarking Temporal Reasoning for Large Language Models. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*. Association for Computational Linguistics, 6389–6415.

[45] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

[46] Yike Wu, Nan Hu, Sheng Bi, Guilin Qi, Jie Ren, Anhuan Xie, and Wei Song. 2023. Retrieve-Rewrite-Answer: A KG-to-Text Enhanced LLMs Framework for Knowledge Graph Question Answering. *CoRR* abs/2309.11206 (2023).

[47] Guangzhi Xiong, Qiao Jin, Zhiyong Lu, and Aidong Zhang. 2024. Benchmarking Retrieval-Augmented Generation for Medicine. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, 6233–6251.

[48] Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze Daniel Gui, Ziran Will Jiang, Ziyu Jiang, Lingkun Kong, Brian Moran, Jiaqi Wang, Yifan Ethan Xu, An Yan, Chenyu Yang, Eting Yuan, Hanwen Zha, Nan Tang, Lei Chen, Nicolas Scheffer, Yue Liu, Nirav Shah, Rakesh Wanga, Anuj Kumar, Wen-tau Yih, and Xin Luna Dong. 2024. CRAG - Comprehensive RAG Benchmark. *CoRR* abs/2406.04744 (2024).

[49] Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. 2024. Language is all a graph needs. In *Findings of the association for computational linguistics: EACL 2024*. 1955–1973.

[50] Dazhou Yu, Riyang Bao, Gengchen Mai, and Liang Zhao. 2025. Spatial-RAG: Spatial Retrieval Augmented Generation for Real-World Spatial Reasoning Questions. *CoRR* abs/2502.18470 (2025).

[51] Xu Zhang, Feiyang Xiao, Liang Yan, and Zhiqing Zhang. 2024. GS-SQL: Modeling Spatial Semantics in Spatial Text-to-SQL. In *International Joint Conference on Neural Networks, IJCNN 2024, Yokohama, Japan, June 30 - July 5, 2024*. IEEE, 1–7.

[52] Yifan Zhang, Cheng Wei, Shangyou Wu, Zhengting He, and Wenhao Yu. 2023. GeoGPT: Understanding and Processing Geospatial Tasks through An Autonomous GPT. *CoRR* abs/2307.07930 (2023).