TAXONOMY-DRIVEN MAP QUERY AGENT DESIGN AND EVALUATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) equipped with tools can resolve map queries expressed in natural language. However, existing systems often assemble their tools and evaluation around target benchmarks, inheriting their capability gaps and lacking an independent reference for assessing coverage. We address this limitation with a faceted taxonomy of capabilities for map queries over 2D vector map data, validated through LLM and human-expert annotation on three datasets. From this taxonomy, we systematically construct a twelve-tool inventory with minimized functional overlap and synthesize evaluation data that covers every capability it defines. We build a simple agent to compose these tools into executable query plans. Experiments across three LLM families show that our agent improves on baseline results for two external benchmarks while maintaining relatively stable accuracy across planner scales. The ablation shows that overlapping tool interfaces reduce accuracy more for smaller planners, supporting the taxonomy's role in the systematic construction of tool inventories with minimal functional overlap.

1 INTRODUCTION

Geospatial data increasingly underpin location-based services, motivating research on geographic information systems (GIS) (Schiller & Voisard, 2004; Longley & Cheshire, 2017), in which spatial information retrieval is an essential building block for geographic queries that range from selecting map objects to deriving their aggregate spatial properties (Purves et al., 2018; Jones et al., 2008b; Mandl et al., 2008; Werner, 2025). Conventionally, such queries are issued through structured query languages over spatial databases and map services, e.g., PostGIS (PostGIS contributors, 2023) and OpenStreetMap (OSM) (OpenStreetMap contributors, 2017; Olbricht, 2015). In this context, large language models (LLMs) further enable users to express queries in natural language (NL), advancing geographic question answering (QA) (Mai et al., 2021) across a range of applications (Asai et al., 2023; Huang et al., 2024; Liang et al., 2025; Lee et al., 2025; Khellaf et al., 2025).

However, reliably resolving map queries in NL remains non-trivial. LLMs are prone to hallucinating geographic facts (Rawte et al., 2023; Manvi et al., 2024a; Wang et al., 2025b). Reliable resolution therefore requires obtaining these facts from authoritative geospatial data (Ning et al., 2025; Yu et al., 2026; Liang et al., 2026). To this end, LLMs are increasingly equipped with callable tools, and current practice typically wraps an entire spatial backend as a single tool within a broader toolbox (Qin et al., 2024; Lu et al., 2026). Nevertheless, such monolithic wrapping falls short on complex map queries spanning multiple capabilities, motivating agentic designs in which LLM agents select and compose fine-grained tools per query (Hasan et al., 2026; Bao et al., 2026).

Although these agentic designs outperform monolithic wrapping, their tool inventories generally inherit the scope and the blind spots of the benchmarks, leaving essential capabilities, such as context resolving (Levinson, 1983; Jones et al., 2008b), grounding (Hill, 2000; Gritta et al., 2018), geometry derivation (Shekhar et al., 1999), and network analysis (Klippel et al., 2005), inadequately addressed or entirely absent in both datasets and agent designs (Kefalidis et al., 2023; Liu et al., 2025; Dihan et al., 2025; Hasan et al., 2026; Bao et al., 2026), as detailed later in Table 1.

The root cause is the absence of an analytical model that systematically delineates the capability space a map query agent should support. To address this, this paper proposes a faceted taxonomy of capabilities for map queries over 2D vector map data, empirically validated through LLMs and a

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

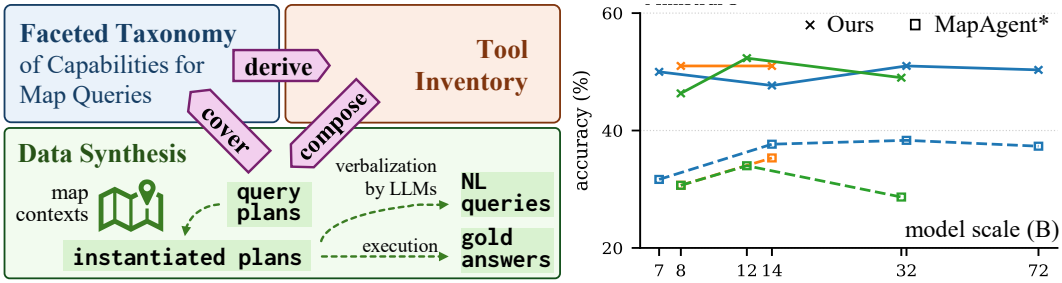

Figure 1: The faceted taxonomy defines map-query capabilities and guides tool derivation. These tools compose query plans, and the synthesized data thereby covers every taxonomy leaf, closing the loop from capabilities to execution. Our agent built from the derived tools consistently outperforms MapAgent (Hasan et al., 2026) on MapEval-API (Dihan et al., 2025) across Qwen 2.5, Ministral 3, and Gemma 4 at varying scales under exact-match scoring* (Appendix E.3).

human expert on three datasets. As illustrated in Figure 1, first, taxonomy leaves with identical input/output signatures are grouped into tools, yielding an inventory with traceable capability coverage and minimal functional overlap. Second, the taxonomy guides the construction of query plans that collectively cover all its leaves. These plans are instantiated against map data, verbalized into NL queries, and executed to obtain gold answers, forming a dataset with explicit capability coverage. The taxonomy is not intended to be definitive. Its contribution is a common analytical foundation for coverage analysis, tool derivation, and data synthesis. As requirements evolve, the taxonomy can serve as a shared reference for identifying corresponding changes to tools and evaluation data.

Our agent is built from the derived tools and evaluated on MapQA (Li et al., 2025), MapEval-API (Dihan et al., 2025), and our synthesized data across three LLM series. Figure 1 illustrates that even with the smallest planner, our agent outperforms every MapAgent configuration (Hasan et al., 2026). Accuracy remains relatively stable across planner scales on the external benchmarks. An ablation further suggests that smaller planners are particularly sensitive to overlapping tool interfaces, supporting the value of clear functional boundaries constructed from the taxonomy.

2 Related Work

LLMs are increasingly integrated into geospatial systems for NL interfaces over spatial data. Existing approaches generally fall into two families differing in how LLMs use the underlying data.

Representational transformation. This family converts geospatial data into a form that LLMs can consume. For instance, Manvi et al. (2024a;b) augment original user queries with textual OSM contexts, thereby improving geographic knowledge extraction from LLMs. While prompt augmentation supplies geo-context statically, retrieval-augmented generation (RAG) fetches it dynamically on demand (Lewis et al., 2020), inspiring a range of geospatial adaptations (Wetzel et al., 2025; Ghonim et al., 2025; Liang et al., 2025; Yu et al., 2026). Knowledge graphs (KGs) offer a structured alternative by linking geo-entities with their related metadata (Dsouza et al., 2021), motivating KG-RAG methods (Ma et al., 2026). However, this family of methods is unsuitable for NL map queries. Textual prompting flattens geometry into text, destroying spatial relations on which queries depend. RAG re-indexes structured spatial databases into vectors, degrading exact spatial predicates to approximate embedding similarity. While KGs address different concerns, i.e., using spatial data as geo-references for non-spatial assertions, and fall outside the scope of this paper.

Tool invocation. This family preserves spatial data in its native structured form and equips LLMs with tools that operate over it. Most work targets general GIS workflows, where agents orchestrate domain libraries and code execution (Singh et al., 2024; Ning et al., 2025; Akinboyewa et al., 2025; Lin et al., 2025; Lee et al., 2025). A parallel line of research translates NL spatial queries into executable languages, such as OverpassQL (Lawrence & Riezler, 2016; 2018; Staniek et al., 2024) and SQL (Liu et al., 2023; 2025). MapAgent (Hasan et al., 2026) and Spatial-Agent (Bao et al.,

C	**context binding**	**R**	**entity retrieval**	**D**	**derivation**	**N**	**network analysis**
C1	spatial context binding‡	R1	thematic retrieval	D1	reduction	N1	route computation
C2	temporal context binding‡	R2	attribute retrieval	D1.1	measurement	N2	stop sequencing
G	**grounding**	R3	spatial retrieval	D1.2	aggregation	N3	origin-destination (OD)/cost matrix
G1	geocoding	R3.1	topological retrieval	D2	geometry derivation	N4	reachability
G2	reverse geocoding	R3.2	metric retrieval	D2.1	overlay		
G3	entity/toponym resolution	R3.3	directional retrieval	D2.2	transformation		
G4	deictic resolution‡	R4	temporal retrieval	D3	ranking		
				D4	projection‡		

Figure 2: The proposed faceted taxonomy of capabilities for map queries. Table 7 in Appendix A.1 provides the definitions of all leaf capabilities with typical examples.

2026) are the most closely related systems. The former composes map tools for NL map queries, while the latter constructs geospatial workflows using core concepts in GIS and functional roles.

However, the scope and granularity of their tool inventories remain system-specific design choices because they never reference an explicit analytical model of map queries. Without such a reference, three issues arise. (i) *Coverage*: functions outside the patterns considered during system and benchmark design may be absent from both the tool inventory and its evaluation. For instance, MapAgent lacks tools for essential geometry derivation and isochrone calculation. Table 1 lists concrete gaps. (ii) *Functional overlap*: tools with shared functions are undetected, inflating the action space the LLM must navigate (Qin et al., 2024; Patil et al., 2024). (iii) *Independent auditability*: omissions, redundancies, and evaluation coverage cannot be systematically assessed against an external characterization of what map queries may require. Consequently, tools and benchmarks remain mutually dependent, with no external standard against which their adequacy can be evaluated.

This paper addresses this gap with a validated taxonomy of capabilities for map queries (Section 3) as the analytical model, from which a tool inventory with minimized functional overlap and data synthesis with comprehensive capability coverage (Section 4) are jointly derived by construction.

3 Faceted Taxonomy for Map Queries

Structured characterization of open-ended NL map queries provides a common reference for analyzing concrete systems and datasets. In this context, taxonomies have been developed for related topics, including the classification of topological spatial relations (Clementini et al., 1994), spatial queries (Carniel, 2024), and geographic QA (Mai et al., 2021). These works motivate the proposed faceted taxonomy, which delineates the capabilities a map-query agent must support. The taxonomy is grounded in established spatial-query conventions and validated on three NL map-query datasets, providing a principled foundation for downstream tool derivation and data synthesis.

3.1 Design and Structure

Design principles. Three principles govern the taxonomy design. First, the taxonomy is *implementation-independent*, as binding it to specific stacks would couple it to incidental engineering choices and obscure its portability. Second, the taxonomy operates at the *capability* level, where taxonomy leaves denote capabilities required by a map query system to interpret and execute user requests. A coarser user-intent level leaves intermediate capabilities implicit, e.g., user localization, while a finer operator-level decomposition inflates and destabilizes the label space. Between these extremes, the *capability* level is expressive enough to delineate tool boundaries, yet compact enough to remain stable. Finer decomposition is introduced only when a domain-established distinction or an operationally consequential signature difference motivates the split. Third, the taxonomy is *faceted*. A faceted organization allows a single query to engage multiple capabilities both intra- and inter-facet, whereas a strict hierarchy would force orthogonal types into arbitrary nesting.

Taxonomy. Figure 2 presents the proposed taxonomy, scoped to 2D vector map queries. The capability space is partitioned into five facets. Leaf definitions and examples are listed in Table 7.

```
{ "Query": "What is the size of Brandenburg?",
  "Output": {
    "solvable": true, "unsolvable_reason": "",
    "assignments":
      [{"label": "G3",    "evidence": "Brandenburg is a named region"},
       {"label": "D1.1", "evidence": "size means area of the region"}]}}
```

Listing 1: An example of an annotated query for empirical validation.

Table 1: Missing capabilities of existing map-query datasets and agents, relative to the taxonomy (Figure 2). A capability is missing when no dataset question type requires it or no tool provides it.

Datasets/Agents	Missing
GeoQuestions1089 (Kefalidis et al., 2023)	G2, R4, D2.1, D2.2, D4, all N leaves
MapEval (Dihan et al., 2025)	C2, G2, G4, D2.1, D2.2, D4, N4
MapQA (Li et al., 2025)	C1, C2, G2, R4, D1.2, D3, D4, all N leaves
MapAgent (Hasan et al., 2026)	G2, R3.1, R3.3, R4, D1.1, D2.1, D2.2, N3, N4
Spatial-Agent (Bao et al., 2026)	D2.1, N4

Context binding (C) captures implicit environmental inputs, such as user location and current time, that must be resolved before query execution (Abowd et al., 1999; Schiller & Voisard, 2004; Purves et al., 2018). Although seldom lexicalized, such inputs are essential to disambiguate common requests such as *open restaurants nearby* (Levinson, 1983; Schilit et al., 1994; Jones et al., 2008b). **Grounding** (G) resolves explicit referring expressions to concrete map referents by combining gazetteer evidence with geographic priors (Hill, 2000; Gritta et al., 2018; Zhang & Bethard, 2024), spanning geocoding and its inverse alongside toponym and deictic resolution (Jones et al., 2008a; Vasardani et al., 2013). **Entity retrieval** (R) returns map entities satisfying thematic, attribute, spatial, or temporal constraints, typically through unary selection, binary (semi-)join, and boolean composition, where R3 adopts the standard decomposition into topological (Egenhofer & Franzosa, 1991; Clementini et al., 1993; 1994), metric, and directional retrieval (Carniel, 2024). **Derivation** (D) returns outputs computed from existing entities, covering capabilities consistent with established spatial database conventions (Shekhar et al., 1999; Open Geospatial Consortium, 2011). **Network analysis** (N) produces results derived from map networks, including route computation, stop sequencing, cost matrices, and reachability (Klippel et al., 2005).

3.2 Empirical validation via annotation

Although grounded in established conventions, the taxonomy remains a structural assertion about the capability space, necessitating validation before downstream tool derivation. We therefore annotate NL map queries from three datasets with diverse query styles under the taxonomy, which exposes both capabilities the taxonomy omits and capabilities that no query exercises, thereby avoiding the circularity of testing it on queries written to conform to it, and dependence on any single benchmark.

Annotation. Datasets for annotation are GeoQuestions1089 (Kefalidis et al., 2023), MapEval (textual) (Dihan et al., 2025), and MapQA (Li et al., 2025), comprising 4,543 queries. Each query receives a leaf assignment with a solvability flag, as exemplified in Listing 1. A query is considered solvable when the taxonomy covers all its capability requirements. These datasets deliberately include out-of-scope queries, e.g., requiring population data beyond 2D maps. This negative class prevents annotators from trivially marking every query solvable. Two LLMs, Qwen2.5-72B-Instruct (Yang et al., 2024) and Llama-3.3-70B-Instruct (Grattafiori et al., 2024), annotate queries under a prompt presenting the taxonomy, labeling rules, and few-shot examples. A human expert independently annotates a subset under the same prompt, providing a reliable anchor. Datasets, prompts, and the LLM/human-annotation settings are detailed in Appendix A.2.

Analysis. Annotation reliability is measured by inter-annotator agreement (IAA) on solvability and leaf assignment among three annotators. Given the heavy imbalance in solvability ($\sim$97%),

Table 2: Tool inventory derived from the proposed taxonomy. Table 11 provides descriptions of **data_types**, e.g., **entity_set**, **geom**, which also lists implicit adaptations among them.

Tools	Leaves	Input	Output
geocode	G1, G3	**text** for query	relevance-ranked **entity_set**
rev_geocode	G2	**coord**	**text** (address record)
retrieve	all R leaves	**entity_set**, optional **geom**	**entity_set**
measure	D1.1	**geom**	**scalar**
aggregate	D1.2	**entity_set**, aggregation operators/grouping parameters	**scalar** or grouped **entity_set**
overlay	D2.1	[**geom**, **geom**], with $\cup$, $\cap$, or $\setminus$	**geom**
geo_transform	D2.2	**geom**, operators (e.g., centroid)	**geom**
rank	D3	**entity_set**, rank parameters	sorted **entity_set**
route	N1	**coords**, travel mode, constraints	**route**, distance, duration
optimize_stops	N2	**coords**, travel mode, objective	ordered **coords**
cost_matrix	N3	[**coords**, **coords**] (sources and targets), travel mode	cost **matrix** with distance and duration per source-target pair
isochrone	N4	**coord**, travel mode, budget	reachable **polygon**

AC1 is used to avoid the prevalence paradox (Gwet, 2008), reaching 0.994 with a 95% bootstrap confidence interval (CI) of $[0.981, 1.000]$ and pairwise AC1 of 0.990 between LLMs. Leaf assignment is multi-label, and IAA is therefore measured by Krippendorff's α (Krippendorff, 2019) under the MASI distance (Passonneau, 2006), reaching 0.764 with a 95% CI of $[0.706, 0.820]$, above the conventional 0.667 threshold. Such high consistency across multiple datasets and annotators justifies the taxonomy's use as a shared reference for map-query agent design and evaluation.

Beyond annotation reliability, Table 1 shows that five capabilities in the proposed taxonomy are absent from all three benchmarks when judged against their documented question types (G2, D2.1, D2.2, D4, N4). Since these capabilities are well established in the literature, their absence reflects dataset deficiencies, which are propagated to the agents calibrated against them: MapAgent misses nine capabilities, whereas Spatial-Agent ignores D2.1 and N4. This analysis demonstrates the necessity of the proposed taxonomy, which can render such omissions measurable and provide a principled reference against which inventories and benchmarks can be audited. More results including leaf utilization and pairwise agreement statistics are reported in Appendix A.2.5.

4 Tool Derivation and Data Synthesis

With the validated taxonomy in place, tools are subsequently inventoried by grouping leaves with shared input/output signatures, followed by data synthesis ensuring complete capability coverage.

4.1 Tool derivation

The taxonomy delineates capabilities for map queries rather than executable actions. Hence, mapping each leaf directly to a tool is inappropriate. Specifically, C1, C2, and G4 are contextual, resolved from session state prior to tool dispatch, whereas D4 is parametric and shapes answer presentation. These four leaves are thus marked ‡ as *non-executable leaves* and excluded from tool derivation. Among the remaining leaves, several share a single execution signature. Mapping them to separate tools would replicate identical actions and unnecessarily complicate agent planning. Therefore, taxonomy leaves are grouped by identical input/output signature into a single tool, yielding the twelve-tool inventory in Table 2, where the grouping procedure inherently reduces functional overlap among tools. Although the R leaves are grouped into one tool, Table 10 shows that their per-leaf utilization differs markedly across datasets, which justifies their retention as distinct entries in the taxonomy. Following text-to-SQL approaches that decompose requests into query fragments (Wang et al., 2020; Eyal et al., 2023; Wang et al., 2025a), the R and D tools are designed to return deferred handles as lazy Common Table Expressions (CTEs). A compiler then assembles these handles into a single WITH clause, consolidating multiple operations into a single query execution.

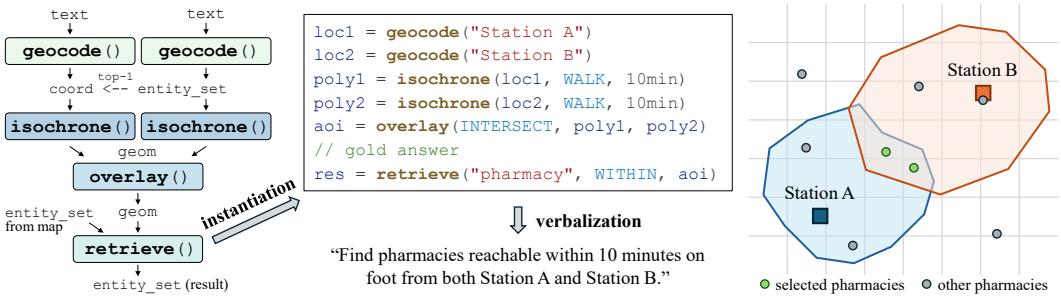

Figure 3: Plan instantiation and verbalization. The symbolic plan is instantiated against map data. The gold answer is obtained by executing the plan and verified by re-execution. The instantiated plan is then verbalized into an NL query. The right side depicts the synthesized query and answer.

4.2 Data Synthesis

We introduce a dataset synthesis pipeline that constructs aligned NL map queries, executable plans, and verified gold answers, all grounded in real map data. It consists of three stages: plan generation, instantiation, and verbalization. Figure 3 traces one generated symbolic plan through instantiation against map data and subsequent verbalization into a NL map query.

Plan generation. A map query typically requires several tool calls whose results feed later ones. We therefore represent it as a typed directed acyclic graph (DAG) over the tools in Table 2, where nodes are tool calls and edges carry results forward. Generation proceeds backward from one of seven answer goals: *entities*, *count*, *region*, *area*, *route*, *route-length*, *travel-costs*, with each expansion applying a production rule that fixes the operation, its input/output, and the tool call. Admissible plans satisfy the type and substitution relations in Table 11 on every edge, require every node to contribute to the single answer node, and exclude redundant or inherently empty operations. Non-executable leaves (‡) are selectively injected as context or output-control requirements. Plans follow one of four topologies, namely SINGLE, CHAIN (exactly one downstream node), BRANCH-JOIN (independently produced results merged downstream), and SHARED-RESULT (one result sent to multiple consumers). Sampling across these topologies governs structural diversity. Note that the answer goals restrict only the form of the final answer. Recursive rule composition and topology-controlled sampling can preserve diversity of the resulting DAG structures (see Appendix C.4).

Instantiation and Verbalization. Instantiation replaces the external inputs and symbolic parameters of a plan with entities and values sampled from a map backend. The binding process preserves dependencies among operands, such as a shared reference location or a region consumed by multiple nodes. The instantiated plan is then executed in topological order, applying the substitutions in Table 11 where required. Its execution result becomes the gold answer. Replaying the serialized plan and bindings verifies that the answer is reproducible and that the instance satisfies the tool preconditions. An LLM verbalizes the instantiated plan from its operations, bound operands, constraints, and requested output, without observing the gold answer. A second LLM call serves as a judge of semantic equivalence (Zheng et al., 2023). Concretely, it checks whether the NL query expresses the complete plan without adding unsupported requirements. Accepted instances form aligned $\{\text{query}, \text{plan}, \text{answer}\}$ triples. Further methodological details are provided in Appendix C.

5 Agent Design

The taxonomy-derived tool inventory listed in Table 2 completely covers the executable capability space for NL map queries. We compose these tools with a deliberately simple agent, in which a single off-the-shelf LLM proposes plan DAGs that the agent validates and executes to produce answer candidates. This design departs from the hierarchical module routing of MapAgent and the template-based workflow construction of Spatial-Agent. Our hypothesis is that a complete, minimally overlapping tool inventory simplifies tool choice, especially for smaller planners with weaker planning performance under the same agent design, which we examine in Section 6.

Planning. The planner receives the query and any supplied inputs, including context resolved from the *non-executable leaves*. The prompt specifies the tool catalog, the map-tag vocabulary, and few-shot query-plan examples. It constructs a DAG of tool calls whose arguments may include literals, supplied inputs, and outputs of preceding calls, and it designates the answer-producing call. Results may be shared across later calls (SHARED-RESULT), or combined in a single call (BRANCH-JOIN).

Validation and selection. The agent generates several candidate plans and checks that each uses available tools, admissible arguments, and valid references to supplied entities or earlier results. If none passes, the agent returns validation feedback and allows up to two revision attempts. The agent executes each valid candidate, excluding execution failures when successful candidates exist and preferring nonempty results and nonzero scalar values. It selects the most frequent tool-dependency structure, breaking ties by the most frequent combination of arguments and observed outputs within that structure. If the selected plan fails during execution, returns no rows, or yields a zero scalar, the agent asks the LLM to revise it. The revised plan replaces the original only if it removes the execution error, turns an empty result into a nonempty one, or turns a zero scalar into a nonzero one.

Execution and output. The executor follows the dataflow of the plan DAG, passing each tool's result to its downstream consumers. Spatial database operations remain deferred until needed, when dependent spatial calls are compiled into a CTE query. This makes a connected spatial subplan run as one database query without materializing each intermediate result separately. The result designated by the plan becomes the agent's answer and is scored against the gold answer during evaluation.

6 EXPERIMENTS

To test the hypothesis in Section 5, we evaluate our agent on three datasets, including one synthesized per Section 4.2, and compare it with two other baselines where applicable.

6.1 SETTINGS

Our implementation is primarily built within the openly licensed OSM ecosystem. OSM snapshots (`.pbf`) serve as the only source of truth for backend initialization, thereby avoiding the potential reproducibility issues of commercial map services (e.g., the Google Maps API), whose data are updated continuously and may yield different answers for the same query over time.

Dataset. We conduct data synthesis on two OSM snapshots dated `260101` for two regions: Bayern, Germany (`BY`) and Illinois, USA (`IL`). Following the procedure elaborated in Section 4.2 and the settings listed in Table 13, we obtain 600 data samples (300 per region), each comprising an NL query, an executable gold plan, and a verified gold answer. The set contains 93 distinct DAG structures and exercises every taxonomy leaf and tool at least three times per region. More details are given in Appendix C.4. Listing 2 presents one of the 600 synthesized samples.

The synthesized queries and the tool inventory both derive from the same taxonomy (Figure 1), and our agent is assembled from these tools, hence validating the agent on these queries alone would be circular. We therefore consider two external datasets constructed independently of our pipeline. The first is the full 948-question test split of MapQA (Li et al., 2025). The second is MapEval-API (Dihan et al., 2025), which contains 300 multiple-choice questions based on the Google Maps API.

Backend. Three backends jointly support the tools in Table 2, each covering a disjoint subset. Specifically, Nominatim (Nominatim contributors, 2025) handles `geocode` and `rev_geocode`. PostGIS serves as the sole backend for the `R` and `D` tools, although the Overpass API is the canonical OSM-native choice (Olbricht, 2015), following Appendix D. Valhalla (Valhalla contributors, 2026) serves as the network analysis backend for the `N` tools, natively supporting `isochrone` computation (Hart et al., 1968), a capability absent in the popular alternative OSRM (Luxen & Vetter, 2011). Backends are containerized via Docker (Merkel, 2014), and LLMs served by vLLM (Kwon et al., 2023). All services expose local HTTP endpoints, yielding a uniform tool invocation interface. For MapEval-API, essential adaptations are additionally developed for the Google Maps API.

Table 3: Results on the test split of MapQA across LLM backbones, reported over all nine question types and over the six types covered by Spatial-Agent (Appendix E.1).

Series	Scale	acc.	acc. (6 types)	#err.
Qwen 2.5	72B	42.93	64.06	0
Qwen 2.5	32B	41.46	62.95	0
Qwen 2.5	14B	40.30	62.95	2
Qwen 2.5	7B	43.78	64.29	0
Ministral 3	14B	38.50	63.39	0
Ministral 3	8B	38.40	62.72	0
Gemma 4	31B	43.04	64.06	1
Gemma 4	12B	40.40	62.95	0
Gemma 4	E4B	37.13	60.04	0

Table 4: Comparison on MapEval-API across LLM backbones. MapAgent* results are evaluated using exact-match scoring (Appendix E.3).

Model		Ours		MapAgent*	
Series	Scale	acc.	#err.	acc.	#err.
Qwen 2.5	72B	50.33	0	37.33	0
Qwen 2.5	32B	51.00	0	38.33	0
Qwen 2.5	14B	47.67	5	37.67	0
Qwen 2.5	7B	50.00	3	31.67	1
Ministral 3	14B	51.00	0	35.33	0
Ministral 3	8B	51.00	1	30.67	0
Gemma 4	31B	49.00	7	28.67	1
Gemma 4	12B	52.33	4	34.00	1
Gemma 4	E4B	46.33	3	30.67	0

Table 5: Results on the synthesized dataset across LLM backbones, reported per region (BY/IL).

Series	Scale	BY	$F1_{tool}$	acc.	#err.	IL	$F1_{tool}$	acc.	#err.
Qwen 2.5	72B		94.56	76.33	0		95.02	77.33	1
Qwen 2.5	32B		98.24	84.67	0		99.27	87.67	3
Qwen 2.5	14B		97.04	77.00	9		97.05	78.00	3
Qwen 2.5	7B		91.40	62.33	23		91.05	66.67	21
Ministral 3	14B		97.38	78.00	2		97.77	81.33	4
Ministral 3	8B		96.19	77.67	2		96.83	78.67	2
Gemma 4	31B		97.39	85.33	0		97.57	85.67	0
Gemma 4	12B		98.21	77.67	5		97.95	84.33	2
Gemma 4	E4B		95.94	74.33	18		95.71	79.67	14

Baselines and backbones. We compare our agent with MapAgent (Hasan et al., 2026) and Spatial-Agent (Bao et al., 2026), which are the most closely related systems, as discussed in Section 2. Regarding the LLMs, we select three families (instruction-tuned preferred), namely Qwen 2.5 (7B, 14B, 32B, and 72B), Ministral 3 (8B and 14B) (Liu et al., 2026), and Gemma 4 (E4B, 12B, and 31B) (Gemma Team et al., 2026), where gemma-4-E4B-it is an 8B model with 4.5B effective parameters. Deliberately spanning three families enables us to observe the sensitivity of agent performance to model scale and makes it possible to test whether the resulting trend holds across families.

Please note that although both baselines present OSM-based MapQA results in the papers, their public code repositories provide support only for the Google Maps API. Meanwhile, the Spatial-Agent release does not include the template retrieval path or the explicit GeoFlow constraint validation described in its paper. Therefore, neither system can be evaluated on all three datasets. We will use the reported results when applicable. Appendix E provides more details about baseline reproduction.

6.2 Results

We report answer accuracy (acc., %) and the number of queries with planning errors remaining after the allowed retries (#err.) on all three datasets. For the synthesized dataset, we additionally report $F1_{tool}$, the multiset F1 score between the selected tools and those of the gold plan.

Tables 3 and 4 summarize the results on MapQA and MapEval-API, respectively. Since the released baseline implementations do not fully support evaluation across both datasets, we compare with Spatial-Agent's reported MapQA results and re-evaluate MapAgent on MapEval-API (Appendix E). For MapQA, we additionally report accuracy on the six question types covered by Spatial-Agent (Appendix E.1). Our agent achieves up to 64.29%, exceeding Spatial-Agent's best reported result of 62.45%. With Qwen-2.5-72B-Instruct, our agent achieves 64.06%, outperforming Spatial-Agent's 61.45%. For MapEval-API, our agent consistently outperforms MapAgent across all backbones by

Table 6: Answer accuracy (%) on the synthesized dataset under the original tool inventory and a merged variant with functional overlap. Parenthesized differences compare each Merged row with its Original counterpart and are in percentage points. Red text marks differences above 5 points. Numbers in the headers are query counts. All is the 600 synthesized queries, Overlap the 154 queries whose gold plans compose retrieval with ranking or aggregation, where Rank and Agg. (aggregation) divide those 154 by which operation the composition uses.

Series	Scale	Tools	All (600)	Overlap (154)	Rank (82)	Agg. (72)
Qwen 2.5	7B	original	62.33	43.51	42.68	44.44
Qwen 2.5	7B	merged	59.67 (-2.67)	31.17 (-12.34)	40.24 (-2.44)	20.83 (-23.61)
Qwen 2.5	72B	original	77.00	79.22	79.27	79.17
Qwen 2.5	72B	merged	78.83 (+1.83)	81.17 (+1.95)	79.27 (+0.00)	83.33 (+4.17)
Ministral 3	8B	original	78.17	68.18	60.98	76.39
Ministral 3	8B	merged	75.67 (-2.50)	59.09 (-9.09)	59.75 (-1.23)	58.33 (-18.06)
Ministral 3	14B	original	79.67	80.52	81.70	79.17
Ministral 3	14B	merged	79.33 (-0.34)	79.22 (-1.30)	78.04 (-3.66)	80.56 (+1.39)
Gemma 4	E4B	original	77.00	78.57	84.14	72.22
Gemma 4	E4B	merged	73.50 (-3.50)	69.48 (-9.09)	81.70 (-2.44)	55.55 (-16.67)
Gemma 4	31B	original	85.50	88.31	89.02	87.50
Gemma 4	31B	merged	85.33 (-0.17)	87.66 (-0.65)	89.02 (+0.00)	86.11 (-1.39)

up to 20.33 points under exact-match scoring (Appendix E.3). These results demonstrate that the accuracy of our agent remains relatively stable across backbone scales on both benchmarks.

Table 5 lists results on the synthesized dataset. The differences across backbone scales are more pronounced. Qwen-2.5-7B-Instruct achieves 62.33% and 66.67% accuracy in `BY` and `IL`, respectively, compared with 84.67% and 87.67% for the 32B model. Gemma 4 exhibits a similar gap between its smallest and largest backbones, whereas the two Ministral 3 models perform comparably. $\texttt{F1}_{\texttt{tool}}$ remains above 91% across all configurations, although the smallest Qwen and Gemma models incur more planning errors. These observations motivate a closer examination of whether smaller and larger backbones are equally sensitive to tool design, which we investigate in the following ablation.

6.3 Ablation

We examine this sensitivity by comparing two tool inventories that differ in how much functional overlap they expose. The original tool inventory keeps retrieval, ranking, and aggregation as separate tools, whereas the merged variant additionally reaches ranking and aggregation through `retrieve` while retaining the separate tools, so that the same operation is available through two tools. Both inventories are evaluated on the same queries under identical settings. Table 6 reports accuracy (%) for both inventories (original/merged) across three LLM families. The 154 queries whose gold plans combine retrieval with ranking or aggregation show the merged inventory to be generally worse than the original, and much worse for smaller planners: Qwen-2.5-7B-Instruct loses 12.34 points while 72B gains 1.95, Ministral-3-8B loses 9.09 while 14B loses only 1.30, and gemma-4-E4B-it loses 9.09 while 31B loses only 0.65 (see Appendix F for an example). The loss comes from aggregation, where the merged interface leads the 7B planner to add an unnecessary grouping argument on 37 of the 72 count queries, while ranking is essentially unaffected. Minimal functional overlap is therefore most valuable for smaller planners, which is the design benefit our inventory provides.

7 Conclusion

We propose a faceted taxonomy of capabilities for NL map queries, validated through LLM and human-expert annotation on three datasets. This taxonomy provides a common foundation for deriving a tool inventory with minimal functional overlap and synthesizing evaluation data covering all taxonomy leaves. The resulting agent maintains relatively stable accuracy across planner scales on external benchmarks, although smaller planners remain less accurate on the synthesized dataset. The ablation study suggests that smaller planners are particularly sensitive to overlapping tool interfaces, demonstrating the value of clear functional boundaries constructed from the proposed taxonomy.

AI USE STATEMENT

In this work, we used generative AI tools for generating synthetic data sets, assisting in the writing of proofs, assisting with translation, and implementing methods. We have not used generative AI tools for helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, designing or providing feedback on research methodology or experiments, cleaning and reformating dataset, supporting qualitative and thematic data analysis, and interpreting results. Proposing or refining hypotheses is not applicable to this work. Additionally, we used generative AI tools for creation of artifacts. We have reviewed all AI-assisted work. For example, apart from LLM annotation in Section 3.2, we also have a human expert to independently annotate as a reliable anchor to examine the quality of LLM annotation. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Appendix A, B, and C document taxonomy annotation, tool interfaces, and data synthesis. Section 6.1, Appendix D and E describe experimental settings, backend configurations, and baseline reproduction. We further provide the synthesized dataset in the supplementary files. Code will be released upon acceptance.

ACKNOWLEDGMENTS

OpenStreetMap data is copyrighted OpenStreetMap contributors and available from https://www.openstreetmap.org.

REFERENCES

Gregory D. Abowd, Anind K. Dey, Peter J. Brown, Nigel Davies, Mark Smith, and Pete Steggles. Towards a Better Understanding of Context and Context-Awareness. In *Handheld and Ubiquitous Computing*, pp. 304–307, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg. ISBN 978-3-540-48157-7. doi: https://doi.org/10.1007/3-540-48157-5_29.

Temitope Akinboyewa, Zhenlong Li, Huan Ning, and M. Naser Lessani. GIS Copilot: towards an autonomous GIS agent for spatial analysis. *International Journal of Digital Earth*, 18 (1):2497489, 2025. doi: 10.1080/17538947.2025.2497489. URL https://doi.org/10.1080/17538947.2025.2497489.

Akari Asai, Sewon Min, Zexuan Zhong, and Danqi Chen. Retrieval-based Language Models and Applications. In Yun-Nung (Vivian) Chen, Margot Mieskes, and Siva Reddy (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 6: Tutorial Abstracts)*, pp. 41–46, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-tutorials.6. URL https://aclanthology.org/2023.acl-tutorials.6/.

Riyang Bao, Cheng Yang, Dazhou Yu, Zhexiang Tang, Gengchen Mai, and Liang Zhao. SpatialAgent: Agentic Geo-spatial Reasoning with Scientific Core Concepts. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14896–14911, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.679. URL https://aclanthology.org/2026.acl-long.679/.

R. Bayer and E. McCreight. Organization and maintenance of large ordered indices. In *Proceedings of the 1970 ACM SIGFIDET (Now SIGMOD) Workshop on Data Description, Access and Control*, SIGFIDET '70, pp. 107–141, New York, NY, USA, 1970. Association for Computing Machinery. ISBN 9781450379410. doi: 10.1145/1734663.1734671. URL https://doi.org/10.1145/1734663.1734671.

Anderson Chaves Carniel. Defining and designing spatial queries: the role of spatial relationships. *Geo-spatial Information Science*, 27(6):1868–1892, 2024. doi: 10.1080/10095020.2022.2163924. URL https://doi.org/10.1080/10095020.2022.2163924.

Eliseo Clementini, Paolino Di Felice, and Peter van Oosterom. A small set of formal topological relationships suitable for end-user interaction. In *Advances in Spatial Databases*, pp. 77–295, Berlin, Heidelberg, 1993. Springer Berlin Heidelberg. ISBN 978-3-540-47765-5. doi: 10.1007/3-540-56869-7_16. URL https://doi.org/10.1007/3-540-56869-7_16.

Eliseo Clementini, Jayant Sharma, and Max J. Egenhofer. Modelling topological spatial relations: Strategies for query processing. *Computers & Graphics*, 18(6):815–822, 1994. ISSN 0097-8493. doi: https://doi.org/10.1016/0097-8493(94)90007-8. URL https://www.sciencedirect.com/science/article/pii/0097849394900078.

Mahir Labib Dihan, MD Tanvir Hassan, MD TANVIR PARVEZ, Md Hasebul Hasan, Md Almash Alam, Muhammad Aamir Cheema, Mohammed Eunus Ali, and Md Rizwan Parvez. MapEval: A Map-Based Evaluation of Geo-Spatial Reasoning in Foundation Models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=hS2Ed5XYRq.

Alishiba Dsouza, Nicolas Tempelmeier, Ran Yu, Simon Gottschalk, and Elena Demidova. WorldKG: A World-Scale Geographic Knowledge Graph. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, pp. 4475–4484, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450384469. doi: 10.1145/3459637.3482023. URL https://doi.org/10.1145/3459637.3482023.

Max J. Egenhofer and Robert D. Franzosa. Point-set topological spatial relations. *International journal of geographical information systems*, 5(2):161–174, 1991. doi: 10.1080/02693799108927841. URL https://doi.org/10.1080/02693799108927841.

Ben Eyal, Moran Mahabi, Ophir Haroche, Amir Bachar, and Michael Elhadad. Semantic Decomposition of Question and SQL for Text-to-SQL Parsing. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 13629–13645, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.910. URL https://aclanthology.org/2023.findings-emnlp.910/.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, Mayank Chaturvedi, Aditya Chawla, Victor Cotruta, Alice Coucke, Phil Culliton, Robert Dadashi, Lucas Dixon, Mohamed Elhawaty, Utku Evci, Clément Farabet, et al. Gemma 4 technical report, 2026. URL https://arxiv.org/abs/2607.02770.

Karim Ghonim, Pere-Lluís Huguet Cabot, Riccardo Orlando, and Roberto Navigli. RAED: Retrieval-augmented entity description generation for emerging entity linking and disambiguation. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 34439–34452, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1746. URL https://aclanthology.org/2025.emnlp-main.1746/.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, et al. The Llama 3 Herd of Models, 2024. URL https://arxiv.org/abs/2407.21783.

Milan Gritta, Mohammad Taher Pilehvar, and Nigel Collier. Which Melbourne? Augmenting Geocoding with Maps. In Iryna Gurevych and Yusuke Miyao (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1285–1296, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1119. URL https://aclanthology.org/P18-1119/.

Kilem Li Gwet. Computing inter-rater reliability and its variance in the presence of high agreement. *British Journal of Mathematical and Statistical Psychology*, 61(1):29–48, 2008. doi: https://doi.org/10.1348/000711006X126600. URL https://bpspsychub.onlinelibrary.wiley.com/doi/abs/10.1348/000711006X126600.

Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968. doi: 10.1109/TSSC.1968.300136.

Md Hasebul Hasan, Mahir Labib Dihan, Tanzima Hashem, Mohammed Eunus Ali, and Md Rizwan Parvez. MapAgent: A Hierarchical Agent for Geospatial Reasoning with Dynamic Map Tool Integration. In Vera Demberg, Kentaro Inui, and Lluís Marquez (eds.), *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 1296–1322, Rabat, Morocco, March 2026. Association for Computational Linguistics. ISBN 979-8-89176-386-9. doi: 10.18653/v1/2026.findings-eacl.67. URL https://aclanthology.org/2026.findings-eacl.67/.

Joseph M. Hellerstein, Jeffrey F. Naughton, and Avi Pfeffer. Generalized Search Trees for Database Systems. In *Proceedings of the 21st International Conference on Very Large Data Bases*, VLDB '95, pp. 562–573, Zurich, Switzerland, 1995. Morgan Kaufmann Publishers Inc. URL https://dl.acm.org/doi/10.5555/645921.673145.

Linda L. Hill. Core Elements of Digital Gazetteers: Placenames, Categories, and Footprints. In *Research and Advanced Technology for Digital Libraries*, pp. 280–290, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg. ISBN 978-3-540-45268-3. doi: https://doi.org/10.1007/3-540-45268-0_26.

Chenghua Huang, Shisong Chen, Zhixu Li, Jianfeng Qu, Yanghua Xiao, Jiaxin Liu, and Zhigang Chen. GeoAgent: To Empower LLMs using Geospatial Tools for Address Standardization. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 6048–6063, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.362. URL https://aclanthology.org/2024.findings-acl.362/.

Paul Jaccard. Étude comparative de la distribution florale dans une portion des Alpes et du Jura. *Bulletin de la Société Vaudoise des Sciences Naturelles*, 37(142):547, 1901. ISSN 0037-9603. doi: 10.5169/seals-266450. URL https://doi.org/10.5169/seals-266450.

C. B. Jones, R. S. Purves, P. D. Clough, and H. Joho. Modelling vague places with knowledge from the web. *International Journal of Geographical Information Science*, 22(10):1045–1065, 2008a. doi: 10.1080/13658810701850547. URL https://doi.org/10.1080/13658810701850547.

Rosie Jones, Wei V. Zhang, Benjamin Rey, Pradhuman Jhala, and Eugene Stipp. Geographic intention and modification in web search. *International Journal of Geographical Information Science*, 22(3):229–246, 2008b. doi: 10.1080/13658810701626186. URL https://doi.org/10.1080/13658810701626186.

Sergios-Anestis Kefalidis, Dharmen Punjani, Eleni Tsalapati, Konstantinos Plas, Mariangela Pollali, Michail Mitsios, Myrto Tsokanaridou, Manolis Koubarakis, and Pierre Maret. Benchmarking Geospatial Question Answering Engines Using the Dataset GeoQuestions1089. In *The Semantic Web – ISWC 2023*, pp. 266–284. Springer Nature Switzerland, 2023. ISBN 978-3-031-47243-5. doi: 10.1007/978-3-031-47243-5_15. URL https://doi.org/10.1007/978-3-031-47243-5_15.

Lynn Khellaf, Ipek Baris Schlicht, Tilman Mirass, Julia Bayer, Tilman Wagner, and Ruben Bouwmeester. SPOT: Bridging Natural Language and Geospatial Search for Investigative Journalism. In Pushkar Mishra, Smaranda Muresan, and Tao Yu (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 71–81, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-253-4. doi: 10.18653/v1/2025.acl-demo.8. URL https://aclanthology.org/2025.acl-demo.8/.

Alexander Klippel, Heike Tappe, Lars Kulik, and Paul U. Lee. Wayfinding choremes—a language for modeling conceptual route knowledge. *Journal of Visual Languages & Computing*, 16(4): 311–329, 2005. ISSN 1045-926X. doi: https://doi.org/10.1016/j.jvlc.2004.11.004. URL https://www.sciencedirect.com/science/article/pii/S1045926X0400059X.

Klaus Krippendorff. *Content Analysis: An Introduction to Its Methodology*. SAGE Publications, Inc., Thousand Oaks, California, Fourth edition, 2019. doi: 10.4135/9781071878781. URL https://doi.org/10.4135/9781071878781.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, pp. 611–626, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400702297. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Carolin Lawrence and Stefan Riezler. NLmaps: A Natural Language Interface to Query OpenStreetMap. In Hideo Watanabe (ed.), *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, pp. 6–10, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee. URL https://aclanthology.org/C16-2002/.

Carolin Lawrence and Stefan Riezler. Improving a Neural Semantic Parser by Counterfactual Learning from Human Bandit Feedback. In Iryna Gurevych and Yusuke Miyao (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1820–1830, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1169. URL https://aclanthology.org/P18-1169/.

Chaehong Lee, Varatheepan Paramanayakam, Andreas Karatzas, Yanan Jian, Michael Fore, Heming Liao, Fuxun Yu, Ruopu Li, Iraklis Anagnostopoulos, and Dimitrios Stamoulis. Multi-Agent Geospatial Copilots for Remote Sensing Workflows. In *IGARSS 2025 - 2025 IEEE International Geoscience and Remote Sensing Symposium*, pp. 1084–1089, 2025. doi: 10.1109/IGARSS55030.2025.11243915.

Philip L. Lehman and S. Bing Yao. Efficient locking for concurrent operations on B-trees. *ACM Trans. Database Syst.*, 6(4):650–670, December 1981. ISSN 0362-5915. doi: 10.1145/319628.319663. URL https://doi.org/10.1145/319628.319663.

Stephen C. Levinson. *Pragmatics*. Cambridge Textbooks in Linguistics. Cambridge University Press, 1983. doi: https://doi.org/10.1017/CBO9780511813313.

Patrick 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. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf.

Zekun Li, Malcolm Grossman, Ehsan Qasemi, Mihir Kulkarni, Muhao Chen, and Yao-Yi Chiang. Benchmarking Geospatial Question Answering with MapQA. In *Proceedings of the 33rd ACM International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '25, pp. 1042–1045, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400720864. doi: 10.1145/3748636.3764174. URL https://doi.org/10.1145/3748636.3764174.

Chao Liang, Yuanzheng Cui, Run Shi, Guixiang Zha, Xin Yin, Mingzhong Xiao, Dong Xu, Xuejun Duan, and Bo Huang. GeoAgentic-RAG: A Multi-Agent framework for autonomous geospatial reasoning and visual insight generation with LLM. *International Journal of Applied Earth Observation and Geoinformation*, 147:105195, 2026. ISSN 1569-8432. doi: https://doi.org/10.1016/j.jag.2026.105195. URL https://www.sciencedirect.com/science/article/pii/S1569843226001111.

Jianyuan Liang, Shuyang Hou, Haoyue Jiao, Yaxian Qing, Anqi Zhao, Zhangxiao Shen, Longgang Xiang, and Huayi Wu. GeoGraphRAG: A graph-based retrieval-augmented generation approach for empowering large language models in automated geospatial modeling. *International Journal of Applied Earth Observation and Geoinformation*, 142:104712, 2025. ISSN 1569-8432. doi: https://doi.org/10.1016/j.jag.2025.104712. URL https://www.sciencedirect.com/science/article/pii/S1569843225003590.

Qingming Lin, Rui Hu, Huaxia Li, Sensen Wu, Yadong Li, Kai Fang, Hailin Feng, Zhenhong Du, and Liuchang Xu. ShapefileGPT: a multi-agent large language model framework for automated shapefile processing. *International Journal of Digital Earth*, 18(2):2577884, 2025. doi: 10.1080/17538947.2025.2577884. URL https://doi.org/10.1080/17538947.2025.2577884.

Alexander H. Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, Alexandre Sablayrolles, Amélie Héliou, Amos You, Andy Ehrenberg, Andy Lo, Anton Eliseev, Antonia Calvi, Avinash Sooriyarachchi, Baptiste Bout, Baptiste Rozière, et al. Ministral 3, 2026. URL https://arxiv.org/abs/2601.08584.

Mengyi Liu, Xieyang Wang, Jianqiu Xu, and Hua Lu. NALSpatial: An Effective Natural Language Transformation Framework for Queries over Spatial Data. In *Proceedings of the 31st ACM International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '23, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701689. doi: 10.1145/3589132.3625600. URL https://doi.org/10.1145/3589132.3625600.

Mengyi Liu, Xieyang Wang, Jianqiu Xu, Hua Lu, and Yongxin Tong. NALSpatial: A Natural Language Interface for Spatial Databases. *IEEE Transactions on Knowledge and Data Engineering*, 37(4):2056–2070, 2025. doi: 10.1109/TKDE.2025.3525587.

Paul A Longley and James A Cheshire. Geographical information systems. In *The Routledge Handbook of Mapping and Cartography*, pp. 251–258. Routledge, 2017. ISBN 9781315736822. doi: 10.4324/9781315736822.

Pan Lu, Bowen Chen, Sheng Liu, Rahul Thapa, Joseph Boen, and James Zou. OctoTools: A Multi-Agent Framework with Extensible Tools for Complex Reasoning. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1–86, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1. URL https://aclanthology.org/2026.acl-long.1/.

Dennis Luxen and Christian Vetter. Real-time routing with OpenStreetMap data. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '11, pp. 513–516, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450310314. doi: 10.1145/2093973.2094062. URL https://doi.org/10.1145/2093973.2094062.

Kai Ma, Qingzhong Zhan, Tongyun Zhang, Yuzheng Shi, Zhenxi Fang, Syed Yasir Ali Shah, Or Aimon Brou Koffi Kablan, and Baoyi Zhang. Geo-MAG: A Knowledge Graph (KG)-enhanced Multimodal Retrieval-Augmented Generation (RAG) Framework for Geological Map Understanding. *Transactions in GIS*, 30(2), 2026. doi: 10.1111/tgis.70226.

Gengchen Mai, Krzysztof Janowicz, Rui Zhu, Ling Cai, and Ni Lao. Geographic Question Answering: Challenges, Uniqueness, Classification, and Future Directions. *AGILE: GIScience Series*, 2: 8, 2021. doi: 10.5194/agile-giss-2-8-2021. URL https://agile-giss.copernicus.org/articles/2/8/2021/.

Thomas Mandl, Fredric Gey, Giorgio Di Nunzio, Nicola Ferro, Mark Sanderson, Diana Santos, and Christa Womser-Hacker. An Evaluation Resource for Geographic Information Retrieval. In Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, and Daniel Tapias (eds.), *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, May 2008. European Language Resources Association (ELRA). URL https://aclanthology.org/L08-1285/.

Rohin Manvi, Samar Khanna, Marshall Burke, David B. Lobell, and Stefano Ermon. Large Language Models are Geographically Biased. In *Forty-first International Conference on Machine Learning*, 2024a. URL https://openreview.net/forum?id=sHtIStlg0v.

Rohin Manvi, Samar Khanna, Gengchen Mai, Marshall Burke, David Lobell, and Stefano Ermon. GeoLLM: Extracting Geospatial Knowledge from Large Language Models. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 38791–38807, 2024b. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/a87f2df7c4ab0213c6ea228e7b7f0a4d-Paper-Conference.pdf.

Dirk Merkel. Docker: Lightweight Linux Containers for Consistent Development and Deployment. *Linux Journal*, 2014(239), March 2014. ISSN 1075-3583. URL https://dl.acm.org/doi/10.5555/2600239.2600241.

Huan Ning, Zhenlong Li, Temitope Akinboyewa, and M. Naser Lessani. An autonomous GIS agent framework for geospatial data retrieval. *International Journal of Digital Earth*, 18(1): 2458688, 2025. doi: 10.1080/17538947.2025.2458688. URL https://doi.org/10.1080/17538947.2025.2458688.

Nominatim contributors. Nominatim: Open Source Geocoding with OpenStreetMap Data. https://nominatim.org/, 2025. Version 5.2.0.

Roland M. Olbricht. Data Retrieval for Small Spatial Regions in OpenStreetMap. In Jamal Jokar Arsanjani, Alexander Zipf, Peter Mooney, and Marco Helbich (eds.), *OpenStreetMap in GIScience: Experiences, Research, and Applications*, pp. 101–122. Springer International Publishing, Cham, 2015. ISBN 978-3-319-14280-7. doi: 10.1007/978-3-319-14280-7_6. URL https://doi.org/10.1007/978-3-319-14280-7_6.

Open Geospatial Consortium. OpenGIS® Implementation Standard for Geographic information - Simple feature access - Part 1: Common architecture, 2011. URL https://docs.ogc.org/is/06-103r4/06-103r4.pdf.

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

osm2pgsql contributors. osm2pgsql: An open source tool for importing OpenStreetMap (OSM) data into a PostgreSQL/PostGIS database. https://osm2pgsql.org, source at https://github.com/osm2pgsql-dev/osm2pgsql, 2025. Version 2.2.0.

Rebecca Passonneau. Measuring Agreement on Set-valued Items (MASI) for Semantic and Pragmatic Annotation. In Nicoletta Calzolari, Khalid Choukri, Aldo Gangemi, Bente Maegaard, Joseph Mariani, Jan Odijk, and Daniel Tapias (eds.), *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy, May 2006. European Language Resources Association (ELRA). URL https://aclanthology.org/L06-1392/.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large Language Model Connected with Massive APIs. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 126544–126565. Curran Associates, Inc., 2024. doi: 10.52202/079017-4020. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/e4c61f578ff07830f5c37378dd3ecb0d-Paper-Conference.pdf.

PostGIS contributors. PostGIS spatial database extension to PostgreSQL. https://postgis.net/, 2023. Version 3.4.0.

PostgreSQL Global Development Group. PostgreSQL Documentation: Chapter 70. GIN Indexes, 2026a. URL https://www.postgresql.org/docs/16/gin.html. Version 16. Accessed: 2026-04-30.

PostgreSQL Global Development Group. PostgreSQL Documentation: F.18. hstore — hstore key/value datatype, 2026b. URL https://www.postgresql.org/docs/16/hstore.html. Version 16. Accessed: 2026-04-30.

Ross S. Purves, Paul Clough, Christopher B. Jones, Mark H. Hall, and Vanessa Murdock. Geographic Information Retrieval: Progress and Challenges in Spatial Search of Text. *Foundations and Trends in Information Retrieval*, 12(2-3):164–318, 02 2018. ISSN 1554-0669. doi: 10.1561/1500000034. URL https://doi.org/10.1561/1500000034.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, dahai li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating Large Language Models to Master 16000+ Real-world APIs. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=dHng2O0Jjr.

Vipula Rawte, Swagata Chakraborty, Agnibh Pathak, Anubhav Sarkar, S.M Towhidul Islam Tonmoy, Aman Chadha, Amit Sheth, and Amitava Das. The Troubling Emergence of Hallucination in Large Language Models - An Extensive Definition, Quantification, and Prescriptive Remediations. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2541–2573, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.155. URL https://aclanthology.org/2023.emnlp-main.155/.

B. Schilit, N. Adams, and R. Want. Context-Aware Computing Applications. In *1994 First Workshop on Mobile Computing Systems and Applications*, pp. 85–90, 1994. doi: 10.1109/WMCSA.1994.16.

Jochen Schiller and Agnès Voisard. *Location-Based Services*. Morgan Kaufmann, San Francisco, 2004. ISBN 978-1-55860-929-7. doi: https://doi.org/10.1016/B978-1-55860-929-7.X5000-6. URL https://www.sciencedirect.com/science/article/pii/B9781558609297500005.

S. Shekhar, S. Chawla, S. Ravada, A. Fetterer, Xuan Liu, and Chang-Tien Lu. Spatial Databases–Accomplishments and Research Needs. *IEEE Transactions on Knowledge and Data Engineering*, 11(1):45–55, 1999. doi: 10.1109/69.755614.

Simranjit Singh, Michael Fore, and Dimitrios Stamoulis. GeoLLM-Engine: A Realistic Environment for Building Geospatial Copilots. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 585–594, Los Alamitos, CA, USA, June 2024. IEEE Computer Society. doi: 10.1109/CVPRW63382.2024.00063. URL https://doi.ieeecomputersociety.org/10.1109/CVPRW63382.2024.00063.

Michael Staniek, Raphael Schumann, Maike Züfle, and Stefan Riezler. Text-to-OverpassQL: A Natural Language Interface for Complex Geodata Querying of OpenStreetMap. *Transactions of the Association for Computational Linguistics*, 12:562–575, 04 2024. ISSN 2307-387X. doi: 10.1162/tacl_a_00654. URL https://doi.org/10.1162/tacl_a_00654.

Valhalla contributors. Valhalla: Open Source Routing Engine for OpenStreetMap. https://github.com/valhalla/valhalla, 2026. Version 3.6.3.

Maria Vasardani, Stephan Winter, and Kai-Florian Richter. Locating place names from place descriptions. *International Journal of Geographical Information Science*, 27(12):2509–2532, 2013. doi: 10.1080/13658816.2013.785550. URL https://doi.org/10.1080/13658816.2013.785550.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. RAT-SQL: Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7567–7578, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.677. URL https://aclanthology.org/2020.acl-main.677/.

Bing Wang, Changyu Ren, Jian Yang, Xinnian Liang, Jiaqi Bai, LinZheng Chai, Zhao Yan, Qian-Wen Zhang, Di Yin, Xing Sun, and Zhoujun Li. MAC-SQL: A Multi-Agent Collaborative Framework for Text-to-SQL. In Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, and Steven Schockaert (eds.), *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 540–557, Abu Dhabi, UAE, January 2025a.

Association for Computational Linguistics. URL https://aclanthology.org/2025.coling-main.36/.

Shengyuan Wang, Jie Feng, Tianhui Liu, Dan Pei, and Yong Li. Mitigating Geospatial Knowledge Hallucination in Large Language Models: Benchmarking and Dynamic Factuality Aligning. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 870–888, Suzhou, China, November 2025b. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.45. URL https://aclanthology.org/2025.findings-emnlp.45/.

Martin Werner. Mining Meaningful Facets in Spatial Information Retrieval with Spatial Relevance Feedback. *AGILE: GIScience Series*, 6:12, 2025. doi: 10.5194/agile-giss-6-12-2025. URL https://agile-giss.copernicus.org/articles/6/12/2025/.

Simeon Wetzel, Auriol Degbelo, and Stephan Maes. Large Language Models for Conversational Geodata Search, June 2025. URL https://github.com/simeonwetzel/agile2025-llm-geo-search. AGILE 2025 Tutorial, Dresden, Germany.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, et al. Qwen2 Technical Report, 2024. URL https://arxiv.org/abs/2407.10671.

Dazhou Yu, Riyang Bao, Ruiyu Ning, Jinghong Peng, Gengchen Mai, and Liang Zhao. SpatialRAG: Spatial Retrieval Augmented Generation for Real-World Geospatial Reasoning Questions. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 11094–11112, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.539. URL https://aclanthology.org/2026.findings-acl.539/.

Zeyu Zhang and Steven Bethard. A survey on geocoding: algorithms and datasets for toponym resolution. *Language Resources and Evaluation*, 59(2):1775–1796, 2024. ISSN 1574-0218. doi: 10.1007/s10579-024-09730-2. URL https://doi.org/10.1007/s10579-024-09730-2.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph Gonzalez, and Ion Stoica. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 46595–46623. Curran Associates, Inc., 2023. doi: 10.52202/075280-2020. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/91f18a1287b398d378ef22505bf41832-Paper-Datasets_and_Benchmarks.pdf.

A TAXONOMY AND EMPIRICAL VALIDATION

This appendix provides more details of the proposed faceted taxonomy of capabilities for map queries, empirical validation, and more corresponding results.

Table 7: Definitions of leaf capabilities of the proposed faceted taxonomy, as illustrated in Figure 2.

Leaf	Name	Definition	Example
C1	spatial context binding‡	resolve implicit spatial references from session state.	restaurants *near me*
C2	temporal context binding‡	resolve implicit temporal references, e.g., now, tonight, from session state.	places *open now*
G1	geocoding	map a textual address to coordinates.	*AB Str. 123, City C* → *coordinates*
G2	reverse geocoding	map a coordinate to a textual address.	*coordinates* → *AB Str. 123, City C*
G3	entity/toponym resolution	resolve named entities in map queries to map entities.	the central station in *City C*
G4	deictic resolution‡	resolve referring expressions, e.g., here or that café, via context binding.	*the park we discussed*
R1	thematic retrieval	retrieve entities by type or category.	all *bakeries*
R2	attribute retrieval	retrieve entities by descriptive attribute.	cafés *with outdoor seating*
R3.1	topological retrieval	retrieve entities under a topological predicate, e.g., within, contains, etc.	shops *within District A*
R3.2	metric retrieval	retrieve entities under a metric predicate, e.g., within distance d.	ATMs *within 500m of here*
R3.3	directional retrieval	retrieve entities under a directional predicate, e.g., north of.	parks *north of the river*
R4	temporal retrieval	retrieve entities under temporal validity predicates.	museums *open on Sundays*
D1.1	measurement	compute a scalar geometric or network quantity, e.g., length, area, bearing, and distance.	*area of this park*
D1.2	aggregation	reduce an entity set to a scalar/-grouped summary, e.g., `COUNT`, `AVG`, `SUM`, `GROUP BY`, etc.	*number of cafés per district*
D2.1	overlay	compute set-algebraic combinations ($\cup$, $\cap$, or $\setminus$) of geometries with attribute propagation.	*intersection of District A and B*
D2.2	transformation	compute a unary geometric derivation, e.g., buffer, centroid, etc.	*500 m buffer around the river*
D3	ranking	return entities in order under a specified criterion: top-k, nearest neighbor, and comparison.	*five nearest* pharmacies to here
D4	projection‡	select returned fields or representations, e.g., geometry, IDs, labels, or attributes.	*return names only, no geometries*
N1	route computation	compute a route between origin and destination under cost or mode constraints.	*drive from A to B avoiding tolls*
N2	stop sequencing	order stops to optimize a routing objective.	*best order for these five errands*
N3	OD/cost matrix	compute pairwise travel cost among origins and destinations.	*drive times among these ten depots*
N4	reachability	compute the area or set reachable under a travel-cost budget.	places *within 15-min walk of here*

A.1 Definitions of leaf capabilities

Definitions of the leaf capabilities in the proposed faceted taxonomy for map queries are presented, along with typical examples, in Table 7.

A.2 Empirical validation

Further details regarding the empirical validation of the proposed taxonomy are provided here.

A.2.1 Dataset

The following three datasets are selected to span the principal formulations of natural-language map queries currently studied in the literature, differing in target representation, underlying spatial data, and source style. Such heterogeneity is essential to the validation in Section 3.2, since reliance on any single data source would conflate taxonomy adequacy with the source conventions.

GeoQuestions1089 contains 1,089 natural-language questions paired with GeoSPARQL targets over geographic linked data, contributing factoid and structured-attribute queries grounded in the semantic web (Kefalidis et al., 2023). A portion of its questions require non-spatial information, e.g., population, which lie outside the scope of the taxonomy. **MapEval** is a benchmark designed to evaluate the geospatial reasoning capabilities of foundation models across textual, API-driven, and visual map contexts, comprising queries that integrate multiple capabilities in a single prompt. Its textual subset (300 queries) is utilized for annotation (Dihan et al., 2025). **MapQA** is an open-domain geospatial QA dataset of 3,154 questions across two study regions, Southern California and Illinois, organized into nine question types covering amenity retrieval, proximity, adjacency, intersection, distance, direction, and nearest-neighbor comparison (Li et al., 2025).

Additionally, two other datasets, NLmaps v2 (Lawrence & Riezler, 2018) and Text-to-OverpassQL (Staniek et al., 2024), were also considered for annotation. However, we subsequently decided not to include these two datasets for the following reasons. **NLmaps v2** contains numerous confusing questions, such as *How many closest Minor Road from Stuttgart in walking distance* (111 analogues). It also contains queries that compel annotators to infer unstated intents. For instance, for *phone closest Hairdresser from Köln in walking distance* (69 analogues), an annotator may wonder whether the query refers to the closest hairdresser to the centroid or to the boundary of Köln, leading to differing leaf capabilities. Such ambiguities introduce meaningless noise into the annotation results and affect validation in an unconstructive manner. **Text-to-OverpassQL** pairs natural-language requests with OverpassQL queries over OpenStreetMap. However, a substantial proportion of these problems are typically coupled with OSM-specific context, such as edit history, contributor information, and geometry IDs, which are derivable from OSM but not from other map data sources. This introduces considerable uncertainty regarding solvability. Although this paper largely utilizes OSM, the data source should not be considered when analyzing the taxonomy, in order to avoid dependence on any specific dataset, as discussed in Section 3.2.

A.2.2 Prompt design

The annotation prompt asks whether a query can be answered using the taxonomy. Leaves used only as possible implementation steps are excluded. Queries requiring external information, such as population or GDP, are marked unsolvable. Grammatical incompleteness or ambiguity alone does not make a query unsolvable. The system prompt provides definitions for all 22 leaves (Table 7) and boundary rules for commonly confused labels, including named-place resolution versus reference to prior conversation. The user prompt supplies 16 author-written examples spanning the five facets (one is shown in Listing 1), followed by the query to annotate. Responses use a fixed JSON schema containing the solvability decision, its reason when unsolvable, and leaf assignments with evidence. The complete prompts and examples are provided with the released code.

A.2.3 LLM annotation settings

LLMs are served locally via vLLM with tensor parallelism across two NVIDIA A100 GPUs in `bfloat16` precision (Kwon et al., 2023). The context window of Qwen2.5-72B-Instruct is set to

Table 8: Pairwise IAA on solvability measured by AC1.

Pair	#queries	AC1↑	95% bootstrap CI
Qwen - Llama	4,543	0.990	[0.986, 0.993]
Llama - Human	120	0.991	[0.972, 1.000]
Qwen - Human	120	1.000	[1.000, 1.000]

Table 9: Pairwise IAA on leaf assignment for solvable queries, measured by Krippendorff's α.

Pair	#queries	α↑	95% bootstrap CI
Qwen - Llama	4,354	0.832	[0.822, 0.843]
Llama - Human	116	0.723	[0.653, 0.793]
Qwen - Human	116	0.786	[0.723, 0.846]

20,480 tokens, and that of Llama-3.3-70B-Instruct is set to 32,768 tokens. All LLMs are configured with a temperature of 0 (greedy decoding) and a maximum output budget of 512 tokens.

Raw model outputs are parsed by attempting direct JSON decoding, followed by searching for a fenced code block, and finally extracting the first legitimate JSON substring. An output is accepted only if it satisfies the following conditions: the three top-level keys (`solvable`, `unsolvable_reason`, and `assignments`) are present, all assignment labels belong to the predefined leaf set, no label appears more than once, and the solvability flag is consistent with the remaining fields. Upon a validation failure, the annotator makes two repair calls, appending a list of violations to the conversation and requesting a corrected JSON object. If the subsequent response still fails, the raw output is written to an error log without further attempts.

A.2.4 Human annotation protocol

The human annotations are produced by a domain expert who was not involved in the design of the taxonomy, thus reflecting an independent application of the taxonomy rather than the designers' own intent. The expert is provided with the same system prompt (Appendix A.2.2) and few-shot examples as LLMs, making annotations by LLMs and the expert directly comparable. Labeling covers a subset of 120 queries, with 40 sampled uniformly at random from each of the three datasets. The random sampling is kept equal across datasets rather than proportional to dataset size, ensuring that no single dataset dominates the agreement estimates.

A.2.5 More results of the empirical validation

We present supplementary evaluation results here, complementing those described in Section 3.2. Apart from the reported three-way IAA on solvability and leaf assignment, Table 8 and Table 9 further provide pairwise results. For both solvability and leaf assignment, `Human - LLM` agreement closely approaches `Qwen - Llama` agreement, indicating that the LLMs align with the human expert rather than merely with each other. Furthermore, pairwise per-leaf IAA is calculated, where per-leaf AC1 scores are no lower than 0.879 (`R3.1`, `Llama - human`) against the expert, indicating that disagreement is not attributable to a single unreliable leaf. Meanwhile, some disagreement is intrinsic, as certain queries admit multiple valid decompositions (e.g., *within walking distance* fits both `R3.1 + N4` and `R3.2`), and thus does not reflect a deficiency of the taxonomy.

Table 10 illustrates the leaf utilization per dataset labeled by LLMs. Leaf utilization is unsatisfactory across the three datasets. While 18 of the 22 leaves are exercised by at least one dataset, four (`G4`, `D2.1`, `N3`, `N4`) appear almost in none, and the network facet is exercised almost entirely by MapEval, which corresponds to established capabilities. Therefore, their absence reflects under-coverage in current map-query benchmarks rather than redundancy in the proposed taxonomy.

Meanwhile, the leaf utilization analysis also addresses a natural question that may arise: why `R` leaves are differentiated in the taxonomy given that they are grouped into a single tool (Table 2). This is because the taxonomy operates at the capability level and supports query and coverage analysis.

Table 10: Leaf utilization per dataset labeled by LLMs. Numbers are the percentages of queries assigning the leaf, averaged across both LLMs. Because utilization records only what the annotated queries require, a capability that a dataset's system assumes to be available in its backend, but that no sampled query requires explicitly, receives a value at or near zero and is not counted as exercised. Utilization is therefore an empirical property of the annotated query sets, complementary to the structural assessment of documented question types and tool inventories in Table 1.

Leaf	GeoQuestions1089	MapEval	MapQA
C1	0.2	31.0	0.3
C2	0.0	1.7	0.0
G1	0.0	0.0	1.7
G2	5.6	0.2	0.0
G3	80.8	97.8	97.7
G4	0.0	0.7	0.0
R1	61.0	40.0	69.9
R2	4.0	13.2	1.4
R3.1	52.9	4.7	8.6
R3.2	2.9	3.8	24.3
R3.3	14.8	4.0	9.8
R4	0.3	12.3	0.0
D1.1	17.6	24.3	9.7
D1.2	19.9	14.2	16.3
D2.1	0.4	0.2	0.0
D2.2	0.0	1.8	0.0
D3	13.4	25.8	50.6
D4	0.5	3.3	0.0
N1	0.0	44.8	0.0
N2	0.0	5.2	0.0
N3	0.0	0.2	0.0
N4	0.0	0.0	0.0

Table 11: Data types utilized in Table 2, with compatible types that can be passed as a valid input.

Type	Description	Also compatible with
text	query, place name, or address records, etc.	
coord	single point (lon, lat)	**entity_set** (representative point)
coords	sequence of points, length ≥ 1	**entity_set** (representative points)
geom	geometries (point, line, polygon, . . .)	**polygon**, **route**, **entity_set** (geometry field)
polygon	closed 2D region	
route	path geometry with distance and duration	
entity_set	map entity set with id, geometry, and attributes	
scalar	numeric value (length, count, duration, . . .)	
matrix	2D array of scalars (e.g., a cost matrix)	

As illustrated in Table 10, R leaves exhibit considerably different utilization patterns across the three datasets. Merging all R leaves in the proposed taxonomy would obscure such a distinction.

B Data Types and Substitution Rules

Each tool in Table 2 has a typed signature, where inputs and outputs are typed values exchanged along DAG edges. Table 11 defines each type and enumerates the types that can be accepted as inputs of that type. For example, an **entity_set** can be adapted to a **coord** by centroid. These adaptations follow standard conventions (Open Geospatial Consortium, 2011), where **polygon** and **route** are both subtypes of geometry and may appear wherever **geom** is expected.

Table 12: Production rules used for backward plan generation. Signatures use the data types and admitted substitutions defined in Table 11.

Production rule	Typed signature	Tool
resolve-place	**text** → **entity_set**	geocode
reachable	**coord** → **polygon**	isochrone
buffer	**geom** → **geom**	geo_transform
intersection	**geom** × **geom** → **geom**	overlay
union	**geom** × **geom** → **geom**	overlay
difference	**geom** × **geom** → **geom**	overlay
find-entities	**geom** → **entity_set**	retrieve
nearest	**entity_set** → **entity_set**	rank
count	**entity_set** → **scalar**	aggregate
area	**geom** → **scalar**	measure
visit-in-order	**coords** → **route**	route
optimize-order	**coords** → **coords**	optimize_stops
visit-optimized	**coords** → **route**	route
route-length	**geom** → **scalar**	measure
travel-costs	**coords** × **coords** → **matrix**	cost_matrix

C Details of Data Synthesis

We provide more details of the entire data synthesis procedure introduced in Section 4.

C.1 Plan generation

Plan generation begins with one of seven answer goals: *entities*, *count*, *region*, *area*, *route*, *route-length*, or *travel-costs*. These goals describe the semantic form of the requested answer. For example, *entities* may be an unranked or nearest entity set, whereas *travel costs* is a matrix of pairwise distances or durations. A *production rule* specifies how a semantic output is obtained from one or more semantic inputs and identifies the tool that realizes the operation. Table 12 lists the complete production rule inventory. Starting from the answer goal, generation proceeds backward: an unresolved semantic input is expanded with a production rule whose output has the required semantic type. For instance, *find-entities* introduces a required geometry, which may in turn be produced by *reachable*. Text, coordinates, geometries, and coordinate sets may remain as external inputs, whereas all other values must be produced by another rule. The graph must be acyclic, contain at most M tool nodes, and have a single answer node to which every other node contributes.

We classify query DAG geometry as SINGLE, CHAIN, BRANCH-JOIN, or SHARED-RESULT. A SINGLE graph contains one node. In a CHAIN, each intermediate result has one consumer. A BRANCH-JOIN graph contains independently produced branches that converge at a downstream node. In a SHARED-RESULT graph, one intermediate result has multiple downstream consumers, which means this class takes precedence when sharing and joining occur together.

For each candidate, we choose a topology τ from the four classes, then a feasible number $m \leq M$ of tool nodes. The weights $\boldsymbol{\pi}$ specify the relative chance of choosing each topology while its target remains unfilled. We generate V alternative plans with the chosen topology and size. A proposed plan that fails the topology, type, or composition checks is redrawn up to a fixed attempt limit. From the valid alternatives, we select the least frequent semantic plan form. The attempt limit keeps generation finite even when a topology and size are difficult to realize.

C.2 Instantiation

Instantiation comprises three steps: grounding the abstract plan, executing the grounded plan to obtain an answer, and replaying the serialized instance to verify that answer. Concretely, for each plan, the underlying map data (snapshot) supplies candidate names, coordinates, geometries, entity categories, and other tool parameters. Related operands are sampled jointly where required by the plan. Each candidate receives at most B grounding attempts. A grounded plan is retained only when every tool precondition is satisfied. If its answer is a set of rows, it must contain at most L rows.

Table 13: Implementation settings of dataset generation used in Section 6. Topology weights are ordered as SINGLE, CHAIN, BRANCH-JOIN, and SHARED-RESULT.

Parameter	Setting
Maximum plan size M	6 tool nodes
Topology proposal weights $\boldsymbol{\pi}$	0.10 / 0.50 / 0.20 / 0.20
Multi-node size mixture	0.80 for feasible sizes up to 4, 0.20 for 5 / 6 nodes
Alternatives per candidate V	8
Maximum grounding attempts B	3
Candidate and rule-proposal limits	1,000 and 10,000
Query timeout	5 seconds
Local anchor radius	2 km
Density radius and count cap	1 km and 64
Density/exploration mixture	0.80 / 0.20; density exponent 0.5
Set-valued answer limit L	5 rows
Maximum empty-answer fraction ρ	0.10
Maximum instances per plan form K	20
Model for verbalization	GPT-5.6 Sol, low reasoning effort
Model for semantic judge	GPT-5.6 Sol, low reasoning effort

Valid empty or zero answers may occupy at most a fraction ρ of the retained set. The complete plan is first executed to produce an answer and then replayed from the exported plan, inputs, and context. A candidate is acceptable only if the re-execution reproduces the complete answer.

C.3 VERBALIZATION AND SEMANTIC REVIEW

The verbalizer receives the plan, bound inputs, regional context, and shared operation definitions **without** the gold answer or intermediate result counts. It is instructed to express every predicate, quantifier, travel mode, ordering constraint, and requested output without introducing unsupported constraints. Each question is then assessed in a separate, tool-free LLM session. The judge receives the question, grounded plan and binding, and shared definitions **without** gold answers. It assesses what the question and solver-visible inputs convey about the plan and records evidence for every plan node. Its checks cover dependencies, anchors, constraints, spatial predicates, answer scope, input provenance, shared parameters, and operand order. Acceptance requires every applicable check to pass and the wording to be understandable and non-redundant. Invalid or incomplete reviews are excluded. Final selection caps each plan form at K instances and balances topology and capability coverage. The published record stores the query, executable plan, required inputs and context, answer schema, gold answer, and provenance.

C.4 DATASETS USED IN THE EXPERIMENTS AND THEIR STATISTICS

We turned the above methodology into practice using OSM snapshots of Bayern (`BY`)[1] and Illinois (`IL`)[2] dated `260101`. Table 13 lists the construction parameters shared by both regions. The OSM snapshots supply named points of interest, place names, coordinates, polygons, and supported tag categories. For a plan that searches for a category of place near a location, we pair a category with a location only if the map contains a matching place within 1 km. In 80% of these draws, locations with more matches are sampled more often. In each region, the final topology distribution is 15% SINGLE, 45% CHAIN, 25% BRANCH-JOIN, and 15% SHARED-RESULT.

We re-executed every instantiated plan against OSM snapshots and compared with its stored gold answer. The total row count must agree exactly. Rows are compared in order when order matters. Identifiers and categorical values must match exactly. Text is compared after case and whitespace normalization, and serialized geometries must match (with two empty geometries treated as equal). For numeric values, the difference may not exceed the larger of 10^{-3} in the column's units and 10^{-3} of the gold value's magnitude. All plans in the synthesized dataset passed this check.

[1] https://download.geofabrik.de/europe/germany/bayern-260101.osm.pbf
[2] https://download.geofabrik.de/north-america/us/illinois-260101.osm.pbf

Table 14: Statistics of the synthesized datasets. #DAGs counts distinct tool-DAG structures, #Empty includes valid empty and zero-valued answers.

Region	#Queries	#DAGs	SINGLE	CHAIN	BRANCH-JOIN	SHARED-RESULT	#Empty
BY	300	83	45	135	75	45	21
IL	300	77	45	135	75	45	20
Union	600	93	90	270	150	90	41

Table 15: Number of queries exercising each taxonomy leaf. One query may require multiple leaves.

Leaf	BY	IL	Leaf	BY	IL
C1	3	3	C2	3	3
G1	3	3	G2	3	3
G3	73	66	G4	3	3
R1	100	93	R2	3	3
R3.1	86	80	R3.2	10	9
R3.3	4	4	R4	3	3
D1.1	65	65	D1.2	39	33
D2.1	117	118	D2.2	98	102
D3	44	38	D4	3	3
N1	34	30	N2	14	12
N3	27	28	N4	200	203

Table 14 reports the structural diversity of the synthesized data. A tool-DAG structure records tool names, typed ports, and ordered dataflow edges, but ignores bound values and wording. Our dataset contains 93 distinct tool-DAG structures, providing substantial compositional diversity despite a restricted set of generation goals. In comparison, MapQA defines nine task types, while MapEval defines four answerable classes and one unanswerable class.

Tables 15 and 16 report the capabilities and tools exercised by the retained queries. Every leaf and tool appears at least three times for each OSM snapshot. Listing 2 shows an example in our dataset.

D On the Choice of Retrieval Backend

This appendix presents benchmark configurations used to compare the retrieval latency between Overpass and PostGIS, thereby substantiating the design decision, as described in Section 6.1, to exclusively adopt PostGIS without incorporating Overpass.

D.1 Backend configurations

The comparison of retrieval latency is conducted across three backend configurations, i.e.,

- `OP`. Overpass with its built-in indexing.
- `PG`. PostGIS with indexes upon `osm2pgsql` import (osm2pgsql contributors, 2025), including a GiST index on the way (geometry) column and a B-tree on `osm_id` per output table (Bayer & McCreight, 1970).
- `*PG`. PostGIS with all above indexes, plus a GIN index on the `hstore` tags column of output tables, and partial B-tree indexes on promoted tag columns, e.g., `amenity`, `shop`, `tourism`, `leisure`, `admin_level`, `landuse`, `natural`, `boundary`, `building`, `name` and `highway`, with planner statistics updated via `ANALYZE`.

All these backends are initialized using the same OSM snapshot[3], deployed as Docker containers on a single host, with the corresponding Docker Compose configuration provided in Listing 3. In this

[3] https://download.geofabrik.de/europe/germany/bayern/oberbayern-260101.osm.pbf.

Table 16: Number of queries using each tool. Counts are not mutually exclusive.

Tool	BY	IL	Tool	BY	IL
geocode	76	69	rev_geocode	3	3
retrieve	100	93	measure	65	65
aggregate	39	33	overlay	117	118
geo_transform	98	102	rank	44	38
route	34	30	optimize_stops	14	12
cost_matrix	27	28	isochrone	200	203

```
Query: How many residential roads intersect the region reachable within
       10 minutes by bicycle from the supplied Lacey Cemetery coordinate?

Input: n0_s0_coord = [40.3494874, -90.2842896]  (Lacey Cemetery)

Plan:
 n0 = isochrone($input(n0_s0_coord), bicycle, 10 min)
 n1 = retrieve(highway=residential, intersects($ref(n0)))
 n2 = aggregate($ref(n1), count)

Gold answer: 14
```

Listing 2: An example from the synthesized dataset. Certain metadata are omitted for readability.

way, Overpass and PostGIS are accessed by the benchmark client via local HTTP within the Docker network, eliminating external network variance. Latency is measured end-to-end from the client's perspective, including request issuance, backend execution, response transfer, and result materialization. Note that the selection of tag columns in *PG is intended only to demonstrate the effect of reasonable indexing on retrieval latency. These columns are empirically selected, corresponding to a subset of tags that appear most frequently in OSM.

D.2 Query workload

Table 17 summarizes 13 query classes formulated for covering a range of patterns for which Overpass is conventionally preferred. Each class specifies equivalent queries for OverpassQL and PostGIS SQL, instantiated with a fixed predicate, e.g., amenity=cafe, cuisine=italian, and, if applicable, one of two bounding boxes (bbox) in Munich, Germany: a large city-wide region and a small city-center region. These settings remain consistent across all backends and configurations.

Class 5/6, 7/8, and 9/10 are paired comparators, designed to observe the effect of the hstore predicate form on index utilization. Within each pair, OverpassQL are identical, whereas PostGIS SQL differ only in the formulation of the tag-equality predicate. Concretely, the arrow form tags -> 'k'='v' extracts the value before comparing, while the containment form tags @> 'k=>v'::hstore tests the hstore directly. These two forms are semantically equivalent but not treated equivalently by the query planner. This is because the GIN of hstore supports @> and ? but not ->, so only the containment form can effectively utilize the GIN. Such pairing isolates the contribution of GIN, as the latency difference within a pair is solely attributable to whether a query can effectively leverage the index.

D.3 Evaluation

All three configurations, OP, PG, and *PG, are evaluated across 13 query classes in Table 17.

D.3.1 Result equivalence

A meaningful latency comparison requires the three backend configurations to return identical result sets for the same queries. However, as Overpass and PostGIS serialize OSM data differently, direct comparison of raw query results is infeasible.

```yaml
services:
  PostGIS:
    image: postgis/postgis:16-3.4
    container_name: postgis
    restart: unless-stopped
    environment:
      POSTGRES_DB: postgres
      POSTGRES_USER: <uname>
      POSTGRES_PASSWORD: <password>
    ports:
      - <your-port>:5432
    healthcheck:
      test: ["CMD-SHELL", "pg_isready -U <uname> -d postgres"]
      interval: 60s
      timeout: 5s
      retries: 20

  Overpass:
    image: wiktorn/overpass-api:0.7.62
    container_name: overpass
    restart: unless-stopped
    ports:
      - <your-port>:80
    volumes:
      - <your-volume>:/osm:ro
    environment:
      OVERPASS_META: "yes"
      OVERPASS_MODE: "init"
      OVERPASS_PLANET_URL: "file:///osm/oberbayern-260101.osm.pbf"
      OVERPASS_RULES_LOAD: 10
      OVERPASS_COMPRESSION: "gz"
      OVERPASS_UPDATE_SLEEP: 3600
      OVERPASS_ALLOW_DUPLICATE_QUERIES: "yes"
      OVERPASS_PLANET_PREPROCESS: "mv /db/planet.osm.bz2 /db/planet.osm.pbf && osmium cat -o /
        db/planet.osm.bz2 /db/planet.osm.pbf && rm /db/planet.osm.pbf && chmod og+rx /db"
    healthcheck:
      test: curl --noproxy '*' -qfg 'http://localhost/api/interpreter?data=[out:json];node(1);
        out;' | jq '.generator' | grep -q Overpass || exit 1
      interval: 10m
      timeout: 60s
      retries: 5
      start_period: 5m
```

Listing 3: The Docker Compose configuration for benchmarking PostGIS and Overpass API performance.

To address this issue, we extract the OSM identifier for each returned element, represented as (`type`, `id`). Overpass directly provides this pair within its returned elements. For PostGIS, the `type` is inferred from the queried tables, e.g., `planet_osm_point` corresponding to nodes and `planet_osm_line` and `planet_osm_polygon` corresponding to ways. Further, since `osm2pgsql` encodes multipolygon relations in polygon tables using negative `osm_id` values, negative identifiers $-r$ are converted to OSM identifiers $(\text{relation}, r)$ before comparison with Overpass outputs. The similarity is quantified by the Jaccard index J over the resulting identifier sets of `OP` and `PG` (Jaccard, 1901). Since indexing does not alter resulting sets, the equivalence check applies to both `PG` and *`PG`.

The equivalence checking demonstrates exact matching ($J = 1$) for #1–4, #9, and #10, and near-perfect matching for the remaining classes, with the smallest $J \approx 0.9659$ (#13). Such deviations are attributable to known representational differences. For instance, compared to Overpass, PostGIS flattens multipolygons and strips empty-valued tags by default. However, these discrepancies are negligible relative to the measured latency gaps reported hereinafter and therefore do not account for the observed performance trends.

D.3.2 Latency analysis

Before measurements, three warm-up cycles are executed to bring both backends to a steady state, ensuring that the measurements reflect query execution without one-time setup costs. Each query is then executed 20 times. Table 18 reports the resulting median warm-cache latencies in milliseconds. `PG` outperforms `OP` across all query classes. *`PG` further reduces latency whenever the query pred-

Table 17: Query classes designed for benchmarking retrieval latency. PostGIS examples use ... to omit a fixed `SELECT` list (`osm_id`, `name`, `ST_AsGeoJSON(way)`, `hstore_to_json(tags)`) and the corresponding `FROM` clause. Only predicates that distinguish each query class are presented.

#	Class description	OverpassQL snippet	PostGIS SQL snippet
1	common tag, no spatial filter	`node["amenity"="cafe"]; out body;`	`SELECT ... FROM planet_osm_point WHERE amenity='cafe'`
2	common tag, large bbox	`node["amenity"="cafe"](bbox); out body;`	`... WHERE amenity='cafe' AND way && bbox`
3	common tag, ∼2 km bbox	`as #2 with smaller bbox`	`as #2 with smaller bbox`
4	predicate over two promoted tags	`node["amenity"="restaurant"] ["tourism"!~".*"](bbox)`	`... WHERE amenity='restaurant' AND tourism IS NULL AND way && bbox`
5	long-tail tag, `hstore` arrow form	`node["cuisine"="italian"]; out body;`	`... WHERE tags->'cuisine' = 'italian'`
6	long-tail tag, `hstore` containment form	`as #5`	`... WHERE tags @> 'cuisine=>italian'::hstore`
7	long-tail + bbox, arrow form	`as #5 with bbox`	`... WHERE tags->'cuisine' = 'italian' AND way && bbox`
8	long-tail + bbox, containment form	`as #5 with bbox`	`... WHERE tags @> 'cuisine=>italian'::hstore AND way && bbox`
9	very rare tag, arrow form	`node["craft"="blacksmith"]; out body;`	`... WHERE tags->'craft' = 'blacksmith'`
10	very rare tag, containment form	`as #9`	`... WHERE tags @> 'craft=>blacksmith'::hstore`
11	polygon retrieval by promoted tag	`way["building"="yes"](bbox); out geom;`	`... FROM planet_osm_polygon WHERE building='yes' AND way && bbox`
12	line retrieval by promoted tag	`way["highway"="residential"] (bbox); out geom;`	`... FROM planet_osm_line WHERE highway='residential' AND way && bbox`
13	tag-key existence	`node["wheelchair"](bbox); out body;`	`... WHERE tags ? 'wheelchair' AND way && bbox`

Table 18: Retrieval latency across query classes, with the same numbering as Table 17.

#	OP (*ms*)	PG (*ms*)	*PG (*ms*)	OP/PG	OP/*PG
1	1212.08	69.33	12.77	17.48×	94.92×
2	395.34	121.66	10.63	3.25×	37.19×
3	189.58	5.74	4.30	33.03×	44.09×
4	514.28	138.45	18.87	3.71×	27.25×
5	1086.40	110.14	109.44	9.86×	9.93×
6	1086.32	108.68	8.95	10.00×	121.38×
7	398.57	121.61	121.55	3.28×	3.28×
8	398.55	102.22	5.28	3.90×	75.48×
9	230.90	100.12	101.61	2.31×	2.27×
10	230.93	100.21	0.50	2.30×	461.86×
11	337.24	10.51	9.75	32.09×	34.59×
12	1151.39	106.74	97.24	10.79×	11.84×
13	856.25	191.29	110.78	4.48×	7.73×

icate can leverage GIN. The largest gains occur for #10. In contrast, the arrow form receives little or no benefit from GIN. Meanwhile, the memory overhead introduced by the additional indexes remains modest. *PG adds 105.75 MiB of indexing, corresponding to 7.57% of the total database size (1.36 GiB). Hence, the observed latency gains of *PG are achieved at a reasonable memory cost.

D.4 Conclusion on the choice of retrieval backend

To summarize, PostGIS functionally subsumes Overpass. An Overpass-based system would still need PostGIS for derivation, complicating cross-tool compatibility and optimization. Meanwhile, experiments demonstrate that augmenting the default GiST index (Hellerstein et al., 1995) with a GIN index on the `hstore` tag column (PostgreSQL Global Development Group, 2026a;b) and partial B-tree indexes on promoted tag columns (Lehman & Yao, 1981), PostGIS is 2.27× – 461.86× faster than Overpass across 13 query classes on state-scale OSM data.

Consequently, Overpass does not provide a latency advantage for OSM-native retrieval, thereby justifying the adoption of PostGIS as the sole retrieval backend. Beyond the latency advantage, a unified PostGIS backend handles both retrieval and derivation, eliminating the need to maintain two systems with distinct query semantics, which substantially simplifies the development.

E Notes on Baseline Reproduction

Experiments in Section 6 involve two baselines, namely MapAgent (Hasan et al., 2026) and Spatial-Agent (Bao et al., 2026). We are grateful to the authors for making their code publicly available, and we regard both works as the closest points of reference for ours. When reusing them, we encountered three obstacles that shape how our numbers should be read, concerning backend support, scoring conventions, and implementation completeness. We document them below in the interest of transparency. None of them reflects on the methodological quality of the two methods.

E.1 Spatial-Agent

We thank the authors of Spatial-Agent for releasing their code[4], enabling us to examine and test the implementation. The released code exposes a retrieval-planning branch, but that branch calls an undefined `_plan_with_retrieval()` method (`src/agent/planner.py`, line 338 - 360). The usable non-retrieval path relies on fixed intent-specific prompts. Functional-role ordering is computed only for logging, while `_build_dag()` passes through the LLM-generated transformations after filtering out unrecognized operator names. There is no separate graph-factorization stage or general graph-wide enforcement of the five paper constraints. Therefore, the released execution path does not implement all of the planning mechanisms described in the paper. These gaps have different consequences in the two evaluation settings, which we discuss in turn.

OSM-backed experiments. The released code supplies Google Maps tools and a MapEval-API-style multiple-choice runner, but no corresponding OSM backend or MapQA loader, evaluation script, or split manifest. Therefore, using it in OSM-based experiments would require reconstructing substantial benchmark-specific functionality whose equivalence to the original setup cannot be verified. We thus rely on the results reported in the original paper for this baseline. Those results cover only six of the nine MapQA question types, and the paper does not state why the remaining three are omitted. To keep the comparison within reported coverage, we additionally report six-type results for our agent and compare against the numbers given in the Spatial-Agent paper[5].

Google-Maps-backed experiments. The README states that the strongest reported results depend on `data/context_cache.db`, yet the release does not distribute this database. While `data/build_cache.py` can reconstruct a cache from MapEval-Textual data, the release provides no checksum, row-level inventory, provenance record, or original snapshot against which the reconstruction can be compared. Equivalence between the claimed cache, a reconstructed cache, and a live Google Maps configuration can therefore not be established from the release alone. In our run of the release with local compatibility adaptations, Qwen2.5-72B-Instruct reached 27.14% macro-accuracy across the four answerable MapEval categories, compared with the paper-reported 53.41%. The full 300-question run recorded 55 failures, including unsupported operator arguments. The unavailable original cache, differences in live API evidence, and our compatibility adaptations prevent us from isolating the causes of this discrepancy. Hence, we report this run only as a reproduction diagnostic and distinguish paper-reported reference results from baselines under our configuration.

[4] https://github.com/ecerybao/Spatial-Agent

[5] https://doi.org/10.18653/v1/2026.acl-long.679

```
# pip install fuzzywuzzy==0.18.0
from fuzzywuzzy import fuzz

def released_string_equal(prediction, reference):
    return fuzz.ratio(prediction.strip().lower(),
                      reference.strip().lower()) >= 30

for pred, gold in [("Turn left", "Turn right"),
                   ("10 mins", "90 mins"),
                   ("Garden", "Backyard")]:
    assert pred != gold
    print(fuzz.ratio(pred.lower(), gold.lower()),
          released_string_equal(pred, gold))

# Output:
# 63 True
# 86 True
# 43 True
```

Listing 4: A minimal example showing the unwanted MapEval-API scoring behaviors in MapAgent's original evaluation pipeline.

Table 19: Comparison on MapEval-API across backbones using the *original* MapAgent scoring.

Model Series	Scale	Ours acc.	MapAgent acc.
Qwen 2.5	72B	85.00	72.00
Qwen 2.5	32B	84.67	79.67
Qwen 2.5	14B	83.00	79.00
Qwen 2.5	7B	85.00	74.33
Ministral 3	14B	83.00	72.00
Ministral 3	8B	83.00	67.33
Gemma 4	31B	83.33	68.33
Gemma 4	12B	87.33	76.67
Gemma 4	E4B	85.00	77.67

E.2 MapAgent

We thank the authors of MapAgent for releasing their implementation and benchmark data. We examined the public repository here[6]. Although the paper reports results on OSM-based MapQA dataset, the released map tools support Google Maps only. Specifically, `src/model.py` invokes `run_conversation()` from `parallel_function_implementation.py`, whose retrieval and routing functions call Google Places, Geocoding, and Directions services. The repository does include MapQA records and split files under `datasets_dir/map_qa_data`, including records with supplied textual spatial context; these do not provide an interface for executing queries against our frozen OSM data. Adding that interface would require a new backend adaptation whose equivalence to the published setup is unverified. We therefore restrict our execution-based comparison with MapAgent to MapEval-API, where both agents use Google Maps evidence.

E.3 MapEval-API scoring

The released evaluator calls `safe_equal(prediction, answer)` to determine correctness (`src/run.py`, line 124 - 136). For string answers, `safe_equal()` function lowercases and trims both strings, then accepts any pair with `fuzz.ratio` of at least 30 on a 0 - 100 scale (`utilities.py`, line 757 - 781). Character similarity at this threshold does not establish answer correctness. Opposite directions or substantially different travel times can share most of their

[6] https://github.com/Hasebul/MapAgent

Table 20: Observed plans with and without merged retrieval options. The gold answer is 884.94 m. The merged plan reuses the ranking reference point P as a route origin and adds a routing operation before measurement. The common prefix and query are condensed for readability.

Query (abridged). What is the pedestrian-route length visiting, in order, the centers of up to four `highway=steps` features nearest to the supplied Fellowship Baptist Church location, selected within the 15-minute bicycle-reachable region from Moulton Methodist Church?

Shared prefix. Geocode Moulton Methodist Church (`limit=1`); compute its bicycle isochrone A (`costs=[15]`). P is the supplied Fellowship Baptist Church coordinate.

Original	Merged

`S = retrieve(`
`thematic: highway=[steps],`
`spatial: within(A))`
	`T = retrieve(`
	`thematic: highway=[steps],`
	`spatial: within(A),`
	`order_by: distance_to(P),`
	`limit: 4)`

`T = rank(handle: S,` `by: distance_to(P), k: 4)`	Ranking is included in retrieval.
`R = optimize_stops(` `stops: T, mode: pedestrian)`	`R = optimize_stops(` `stops: T, mode: pedestrian)`

No additional routing step.	`R2 = route(origin: P,`
	`destination: R,`
	`mode: pedestrian)`

`return measure(` `handle: R, kind: length)`	`return measure(` `handle: R2, kind: length)`
884.94 m: correct	**2,408.56 m: incorrect**

wording. Consequently, even a prediction naming a different, incorrect multiple-choice option can receive credit. This is a correctness check against the reference answer, so the issue directly affects the reported accuracy of runs scored by this evaluator. Listing 4 can minimally reproduce the string-comparison rule using the repository's dependency. All three predictions are incorrect, yet all pass the released threshold. This demonstrates that the scoring rule for MapEval-API used in MapAgent overestimates accuracy. Hence, we report exact-match accuracy in our MapEval-API experiments.

As shown in Table 19, under MapAgent's original evaluator for MapEval-API, our agent still leads in all nine backbone configurations, so the ordering does not depend on the evaluator. We report exact match as the primary metric only because the original evaluator cannot establish answer correctness.

F AN EXAMPLE TRACE FROM THE ABLATION STUDY

Table 20 compares saved final plans from Qwen2.5-Instruct 7B on the same query. Original returns the gold route length; Merged inserts an additional routing step and measures a different route. Both executions succeed and return scalar lengths; the error is numerical.

The paired plans illustrate a downstream composition error observed under Merged, but do not isolate overlap as its cause. Ranking accuracy changes little in aggregate; this case illustrates an individual failure rather than the dominant source of the accuracy difference.