FROM SCENE GRAPHS TO ANSWERS: SELECTIVE NEURO-SYMBOLIC REASONING FOR AUTONOMOUS DRIVING

Anonymous authors
Paper under double-blind review

ABSTRACT

Autonomous-driving question answering requires reasoning over structured scene information, yet existing vision-language approaches largely delegate heterogeneous reasoning operations to a single neural inference process. We argue that this uniform strategy overlooks a fundamental distinction: some queries admit exact symbolic solutions, while others require semantic interpretation. We introduce a *query-adaptive neuro-symbolic reasoning framework* that explicitly allocates computation according to the nature of the query. At its core is a hierarchical Spatiotemporal Scene Graph (STSG) that separates persistent object identities from frame-specific states and represents spatial relations and temporal transitions as explicit directed structures. Given a query, a symbolic executor first attempts to resolve it through exact graph operations; only when symbolic execution abstains is an LLM invoked for semantic reasoning. For these unresolved queries, query-conditioned graph retrieval and evidence filtering preserve relation direction, temporal locality, and object semantics, providing the LLM with compact and verified task-relevant evidence. This design shifts the role of the LLM from a universal reasoning engine to a targeted semantic reasoner, while allowing deterministic computation to be handled exactly and efficiently. We evaluate the framework on 5,916 NuScenes-QA questions across all ten scenes of nuScenes v1.0-mini under an oracle-perception setting. The complete system achieves 80.63% overall accuracy with GPT-5.4-mini, improving over the corresponding LLM-only configuration by 5.48 percentage points; with DeepSeek-V4-Flash, the improvement reaches 6.64 points. The largest gains occur on counting questions, with improvements of 10.20 and 12.61 points, respectively. These results demonstrate that selective allocation of reasoning can improve both accuracy and inference efficiency, providing a principled alternative to treating the LLM as a universal executor for structured reasoning in autonomous driving.

1 INTRODUCTION

Autonomous driving increasingly requires systems to move beyond perception toward structured reasoning about the surrounding world. Given a driving scene, an intelligent agent should not only recognize vehicles, pedestrians, and their attributes, but also answer relational questions such as whether an object exists in a particular region, how many objects satisfy a spatial constraint, or which object has a specific relation to the ego vehicle. Recent datasets such as nuScenes Caesar et al. (2020) and NuScenes-QA Qian et al. (2024) have made such reasoning increasingly accessible. However, these questions are fundamentally different from conventional visual recognition: they require identifying relevant entities, composing multiple relations, preserving relational direction, and executing exact operations such as filtering and counting. The central challenge is therefore not simply how to make a language model reason over driving scenes, but rather how to allocate different reasoning operations to the computational mechanisms that are best suited for them.

Large language models (LLMs) have emerged as powerful general-purpose reasoners and have demonstrated impressive capabilities in multi-step inference Wei et al. (2022); Yao et al. (2023). In autonomous-driving QA, this naturally suggests using an LLM as a universal reasoning engine over scene information. Yet this design implicitly assumes that every query should be solved through

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.

neural language reasoning. This assumption is problematic for structured environments. Many questions have deterministic semantics: existence can be verified directly, spatial constraints can be evaluated by graph operations, and counting can be performed exactly once the relevant entities have been identified. Forcing an LLM to reproduce these operations through free-form generation introduces an unnecessary source of uncertainty and computation. Conversely, questions involving semantic ambiguity or compositional interpretation may not be naturally captured by a fixed symbolic rule system. This reveals a fundamental asymmetry: some reasoning is computable, while other reasoning is interpretive. Treating both uniformly as neural generation leaves this distinction unexplored.

This observation motivates a different view of LLM-based reasoning: the key problem is not choosing between symbolic and neural reasoning, but deciding when each should be used. Rather than using an LLM as the executor of every reasoning step, we formulate reasoning as a query-adaptive computation process. A query that admits an unambiguous symbolic solution should be resolved exactly and directly, whereas a query that exceeds the expressiveness of the symbolic executor should be delegated to a neural reasoner. This perspective is related to the broader neuro-symbolic paradigm Yang et al. (2025); Fang et al. (2024), but differs in emphasis: instead of tightly coupling symbolic and neural modules throughout inference, we explicitly introduce selective reasoning through an abstention mechanism. The symbolic component is therefore not merely an auxiliary tool for the LLM; it serves as a gate that determines whether neural reasoning is necessary at all.

To realize this principle, we introduce a hierarchical Spatiotemporal Scene Graph (STSG) that provides the structured substrate for selective reasoning. The STSG separates persistent physical entities from their frame-specific states, while explicitly encoding directed spatial relations and temporal transitions. This representation allows the system to preserve object identity across observations while supporting exact relational operations at the state level. On top of the STSG, our framework first attempts symbolic execution over the complete structured scene. Only when symbolic execution abstains does the system invoke an LLM. For these unresolved queries, we further perform query-conditioned graph retrieval and evidence filtering, restricting neural reasoning to relevant entities, relations, and frame-specific evidence. This design also complements recent retrieval-augmented approaches Lewis et al. (2020); Han et al. (2025); Guo et al. (2025): instead of treating retrieval merely as a mechanism for supplying more context to an LLM, we use structured retrieval as part of a reasoning allocation pipeline that separates exact computation from semantic inference.

We evaluate this framework on 5,916 NuScenes-QA questions across all ten scenes of nuScenes v1.0-mini. Under the oracle-perception setting, our full system achieves 80.63% accuracy with GPT-5.4-mini, improving over the corresponding LLM-only configuration by 5.48 percentage points. With DeepSeek-V4-Flash, the improvement reaches 6.64 percentage points. The largest improvements occur on counting questions, where symbolic execution provides gains of 10.20 and 12.61 percentage points for GPT-5.4-mini and DeepSeek-V4-Flash, respectively. Selective reasoning also reduces total inference time by bypassing neural inference for deterministically solvable queries. These results indicate that the benefit of neuro-symbolic reasoning in structured driving scenes does not necessarily come from combining two reasoning modules everywhere, but from selectively assigning computation according to the nature of the query. Our contributions are summarized as follows:

- We formulate LLM-based driving-scene QA as a query-adaptive reasoning problem, arguing that deterministic and semantic reasoning should not be uniformly delegated to a neural language model.
- We introduce a hierarchical Spatiotemporal Scene Graph that separates persistent entities from frame-specific states and explicitly represents spatial and temporal relations, enabling exact structured reasoning over driving scenes.
- We propose a selective neuro-symbolic reasoning architecture with an abstention-based routing mechanism, allowing symbolic execution to handle deterministically solvable queries while delegating unresolved cases to an LLM.
- We demonstrate that this selective allocation of reasoning improves both accuracy and inference efficiency across two LLM backends on NuScenes-QA, with particularly substantial gains on exact counting queries.

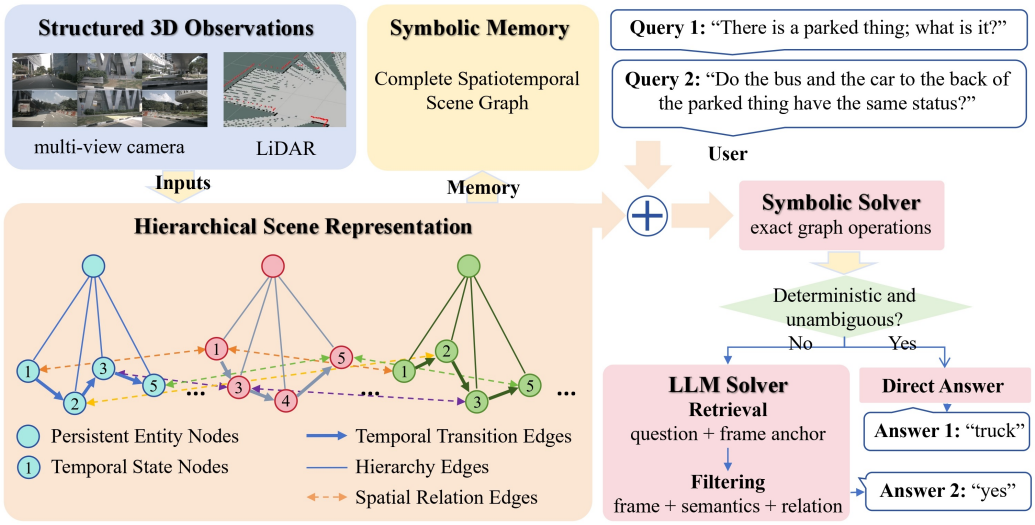

Figure 1: **Overview of the framework.** Structured 3D observations form a hierarchical spatiotemporal scene graph that serves as memory. Unambiguous queries are answered directly by the symbolic solver; otherwise, graph retrieval and evidence filtering provide context for LLM reasoning.

2 Related Work

Vision-language models connect driving-scene understanding with language-based decisions. DriveGPT4 generates driving explanations and control predictions Xu et al. (2024). DriveVLM combines scene analysis with hierarchical planning, while its dual variant incorporates conventional perception and planning Tian et al. (2024). DriveLM organizes question–answer pairs into a graph of logical dependencies Sima et al. (2025).

Scene graphs provide structured representations for robotic perception and reasoning. SceneGraphFusion incrementally predicts and fuses object and relation labels Wu et al. (2021), while Hydra constructs and optimizes hierarchical scene graphs Hughes et al. (2022). ConceptGraphs builds open-vocabulary, object-centric maps for language-guided queries and planning Gu et al. (2023).

Graph-based RAG exploits relational structure to organize evidence. RoG learns relation-path plans and retrieves paths for LLM reasoning Luo et al. (2024). G-Retriever combines connected-subgraph retrieval with graph soft prompts and textualized evidence He et al. (2024). KG^{2}RAG expands and organizes document chunks through graph connections Zhu et al. (2025), while GraphRAG uses hierarchical community summaries for global questions Edge et al. (2025). ToG-3 iteratively refines queries and evidence subgraphs through multi-agent collaboration Wu et al. (2026).

Neuro-symbolic execution and adaptive computation motivate selective reasoning. VISPROG uses an LLM to generate programs that compose pretrained modules and symbolic operations Gupta & Kembhavi (2023). Adaptive-RAG learns to select retrieval strategies according to question complexity Jeong et al. (2024); RouteLLM learns model selection from preference data Ong et al. (2025). Self-RAG learns reflection tokens for retrieval control and output evaluation Asai et al. (2024).

Our framework integrates symbolic-first execution with retrieval-augmented reasoning over a spatiotemporal scene graph. A conservative symbolic solver directly answers supported, unambiguous queries and invokes the LLM only upon abstention, avoiding unnecessary neural computation. This selective design requires no task-specific model fine-tuning and improves both QA accuracy and inference efficiency under the evaluated oracle-perception setting.

3 Methodology

In this section, we present a spatiotemporal retrieval-augmented framework for multi-frame question answering in autonomous driving scenarios. The framework takes structured 3D object ob-

servations from an upstream perception module and decouples deterministic geometric reasoning from language-based inference. In our experiments, the perception interface is instantiated with ground-truth 3D annotations in order to isolate retrieval and reasoning errors from perception errors. The interface can be connected to learned 3D vision-language perception modules that produce the required observations. As illustrated in Figure 1, the framework consists of three components:

1. **Hierarchical Spatiotemporal Graph Construction**: organizing persistent object identities, frame-specific states, spatial relations, and temporal transitions into an explicit, multi-layered spatiotemporal graph.
2. **Structured Graph Projection and Indexing**: projecting task-relevant graph facts into a graph-vector retrieval index while retaining the complete graph as symbolic memory;
3. **Neuro-Symbolic Dual-Engine Reasoning**: first attempting deterministic symbolic execution and, when the symbolic solver abstains, invoking a retrieval-augmented LLM with query-conditioned evidence filtering.

3.1 Spatiotemporal Scene Graph Representation

To represent a dynamic driving scene across multiple frames, we introduce a hierarchical graph that links persistent physical identities to their frame-specific realizations. We refer to this representation as a Spatiotemporal Scene Graph (STSG), denoted by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Unlike a collection of independent frame-level scene graphs, the proposed representation provides a shared identity layer across the sequence while preserving the state and spatial configuration of each timestamp.

The node set $\mathcal{V}$ contains persistent entity nodes $\mathcal{V}_{ent}$, temporal state nodes $\mathcal{V}_{state}$, and an ego-reference node v_{ego}. The edge set $\mathcal{E}$ contains spatial relations $\mathcal{E}_{spat}$, entity–state hierarchy relations $\mathcal{E}_{hier}$, and temporal transitions $\mathcal{E}_{temp}$:

$$\mathcal{V} = \mathcal{V}_{ent} \cup \mathcal{V}_{state} \cup \{v_{\text{ego}}\}, \qquad \mathcal{E} = \mathcal{E}_{spat} \cup \mathcal{E}_{hier} \cup \mathcal{E}_{temp}. \tag{1}$$

Figure 4 in Appendix A.2 illustrates the entity–state hierarchy, temporal transitions, and spatial relations.

3.1.1 Persistent Entity Nodes ($\mathcal{V}_{ent}$)

Persistent entity nodes represent unique instance-level physical objects whose identities are maintained across the scene sequence. Each entity node $v_i^{ent} \in \mathcal{V}_{ent}$ provides a stable identifier that links observations of the same physical object across frames.

3.1.2 Temporal State Nodes ($\mathcal{V}_{state}$)

For each observation of entity i at timestamp t, we create a temporal state node $v_{i,t}^{state} \in \mathcal{V}_{state}$. It represents the realization of that entity at the corresponding frame:

$$v_{i,t}^{state} = \big(i, t, \mathbf{a}_{i,t}\big), \tag{2}$$

where $\mathbf{a}_{i,t}$ denotes the frame-specific semantic and physical attributes, including the object category and motion status. Separating persistent identity from temporal state prevents repeated observations of the same object from being treated as independent physical entities.

3.1.3 Spatial Relation Edges ($\mathcal{E}_{spat}$)

Spatial edges encode directed relations between state nodes observed at the same timestamp, as well as relations between a state node and the ego vehicle. Let $\mathbf{p}_{i,t}^{xy}$ and $\mathbf{p}_{j,t}^{xy}$ be the BEV centers of a subject object i and a reference object j, respectively. Their spatial relations are defined as

$$\Delta\mathbf{p}_{i|j,t} = \mathbf{p}_{i,t}^{xy} - \mathbf{p}_{j,t}^{xy}, \qquad \theta_{i|j,t} = \operatorname{atan2}\left(\Delta y_{i|j,t}, \Delta x_{i|j,t}\right). \tag{3}$$

Following the directional taxonomy of NuScenes-QA Qian et al. (2024), we use its six ego-aligned spatial predicates and denote the corresponding directional mapping by $\phi(\theta)$. The resulting directed relation is

$$e_{i|j,t}^{spat} = \left(v_{i,t}^{state}, \phi(\theta_{i|j,t}), v_{j,t}^{state}\right) \in \mathcal{E}_{spat}. \tag{4}$$

When the reference is the ego vehicle, $v_{j,t}^{state}$ is replaced by v_{ego}. This subject-reference convention makes the direction of each edge explicit and reduces the risk of reversing a spatial relation during reasoning.

3.1.4 HIERARCHY EDGES ($\mathcal{E}_{hier}$)

Hierarchy edges establish structural links between persistent global entities and their frame-specific state instances. A directed hierarchy edge $e_{i,t}^{hier} \in \mathcal{E}_{hier}$ connects a global entity v_i^{ent} to its temporal state $v_{i,t}^{state}$:

$$e_{i,t}^{hier} = \left(v_i^{ent}, \texttt{HAS_STATE}, v_{i,t}^{state}\right) \tag{5}$$

The two-level representation allows an object to be referenced either as a persistent instance or as a frame-specific state.

3.1.5 TEMPORAL TRANSITION EDGES ($\mathcal{E}_{temp}$)

Let $t_i^{(1)} < t_i^{(2)} < \cdots < t_i^{(K_i)}$ denote the ordered timestamps at which entity i is observed. We connect each pair of successive observations by

$$e_{i,k}^{temp} = \left(v_{i,t_i^{(k)}}^{state}, \texttt{NEXT_STATE}, v_{i,t_i^{(k+1)}}^{state}\right) \in \mathcal{E}_{temp}. \tag{6}$$

These links form an explicit state trajectory for each persistent entity and provide a structural basis for reasoning about motion evolution, temporal ordering, and cross-frame comparisons.

3.2 STRUCTURED GRAPH PROJECTION AND INDEXING

The complete spatiotemporal graph $\mathcal{G}$ serves as the structured memory of the symbolic reasoning engine. For retrieval-augmented inference, we derive a retrieval-oriented graph view

$$\mathcal{G}_R = \Pi_R(\mathcal{G}), \tag{7}$$

where Π_R maps relevant graph elements into canonical node and relation records while preserving entity identifiers, relation directionality, and temporal references.

The resulting records are inserted into a LightRAG-based graph-vector index Guo et al. (2025). This structured insertion bypasses the intermediate conversion of the graph into documents and avoids LLM-based re-extraction of entities and relations. In contrast to arbitrary document chunking, each indexed record retains the atomic subject–predicate–object semantics of the original graph.

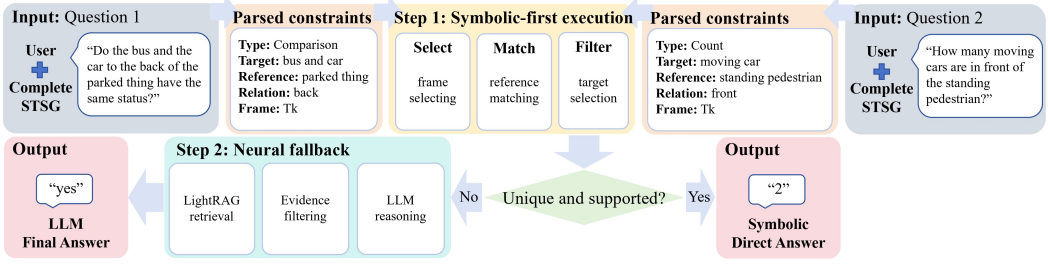

Figure 2: **Neuro-Symbolic Question Answering.** The symbolic solver applies frame selection, reference matching, and target filtering to the complete scene graph. The counting example is answered directly as "2", while the unsupported comparison query triggers abstention and proceeds through LightRAG retrieval, evidence filtering, and LLM reasoning to produce "yes".

3.3 NEURO-SYMBOLIC DUAL-ENGINE QUESTION ANSWERING

Given a natural language query Q, directly providing a large graph $\mathcal{G}$ to a Large Language Model (LLM) is often impractical due to context-length and computational constraints Beltagy et al. (2020); Liu et al. (2024). Dense retrieval reduces the amount of input but may confuse relations that appear in similar contexts while differing in polarity, direction, or other semantic qualifiers, potentially

leading to the retrieval of semantically incompatible facts Weller et al. (2024). To address these limitations, we propose a Neuro-Symbolic Dual-Engine architecture that decouples deterministic symbolic filtering from LLM-based semantic reasoning.

3.3.1 CONSERVATIVE SYMBOLIC-FIRST ROUTING

We define the symbolic solver as a selective solver that may abstain when a question Q cannot be resolved deterministically. Let $\mathcal{Y}$ denote the task-specific answer space, including object categories, motion-status labels, counts, and binary answers. Let Φ_{sym} denote the symbolic solver, Φ_{llm} denote the retrieval-augmented LLM reasoning engine, and $\perp$ denote abstention.

The symbolic solver first evaluates the question against the complete spatiotemporal graph:

$$a_{\mathrm{sym}} = \Phi_{\mathrm{sym}}\left(Q, \mathcal{G}, t_q\right) \in \mathcal{Y} \cup \{\perp\}. \tag{8}$$

For supported and unambiguous questions, the solver performs exact set operations over frame-specific state nodes. It determines existence, counts unique persistent identities for counting questions, and returns an object category or motion status when the answer is unique. When the query pattern is unsupported or multiple answers remain plausible, the solver returns $\perp$ rather than forcing a prediction.

Comparison questions and other compositionally ambiguous questions are therefore delegated to the neural fallback Φ_{llm}.

3.3.2 QUERY-CONDITIONED RETRIEVAL FOR THE NEURAL FALLBACK

When the symbolic solver abstains, the neural fallback retrieves and refines evidence in two steps. Let t_q denote the frame associated with question Q. We use $h(t_q)$ to denote a textual frame anchor, and let $\mathcal{C}_q$ and $\mathcal{R}_q$ summarize the semantic categories and spatial relationship keywords expressed by the question. The retrieval backend first returns a candidate evidence set from the indexed graph view:

$$\mathcal{H}_Q^0 = \mathcal{R}\left(Q \oplus h(t_q); \mathcal{G}_R\right), \tag{9}$$

where $\mathcal{R}$ denotes the LightRAG retrieval operator and $\oplus$ denotes query augmentation.

The retrieved records are subsequently refined by a deterministic query-conditioned filter:

$$\mathcal{H}_Q = \mathcal{F}_{\mathrm{filter}}\left(\mathcal{H}_Q^0; t_q, \mathcal{C}_q, \mathcal{R}_q, Q\right). \tag{10}$$

The filter removes evidence that is inconsistent with the queried frame, object semantics, or relation orientation. Its concrete implementation details are provided in Appendix A.4.

3.3.3 NEURAL REASONING AND FINAL ROUTING

In conclusion, if the symbolic solver returns an answer, that answer is used directly. Otherwise, the filtered evidence $\mathcal{H}_Q$ is serialized into a question-type-specific prompt and passed to the LLM reasoning engine. The final prediction is

$$\hat{a} = \begin{cases} a_{\mathrm{sym}}, & a_{\mathrm{sym}} \neq \perp, \\ \Phi_{\mathrm{llm}}\left(Q, \mathcal{H}_Q\right), & a_{\mathrm{sym}} = \perp. \end{cases} \tag{11}$$

This design reserves neural reasoning for cases that require semantic disambiguation, while deterministic cases are handled directly by the symbolic engine.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Dataset and evaluation protocol. We evaluate on 5,916 NuScenes-QA questions associated with all 404 keyframes from the ten nuScenes v1.0-mini scene sequences Caesar et al. (2020); Qian et al. (2024). We select these questions from the union of the released training and validation question files using their sample tokens. Each question is evaluated at its associated keyframe using the corresponding scene-level graph. Per-scene and per-type statistics are provided in Appendices A.1 and A.7.

Table 1: Answer accuracy (%) on all ten nuScenes v1.0-mini scene sequences. The best result in each column is shown in bold. Symbolic-only is a standalone template-aware symbolic baseline and is not a strict ablation of Full.

Configuration	Exist	Count	Object	Status	Comparison	Overall
Symbolic-only	76.22	51.40	43.21	65.76	65.57	61.43
LLM-only (DeepSeek)	82.42	62.56	59.76	81.63	66.44	71.25
Full (DeepSeek)	85.52	75.17	69.32	89.02	68.67	77.89
LLM-only (GPT)	84.33	65.45	61.77	86.30	**76.21**	75.15
Full (GPT)	**88.80**	**75.65**	**69.88**	**93.26**	74.08	**80.63**

Oracle-perception setting. We use ground-truth 3D bounding boxes, object categories, instance identities, object attributes, and ego poses to construct the spatiotemporal scene graphs. This oracle-perception setting isolates graph construction, retrieval, and reasoning behavior from errors introduced by learned perception modules. Thus, the experiments measure reasoning under structured 3D observations rather than end-to-end perception accuracy.

LLM backends and compared configurations. We evaluate two LLM backends: DeepSeek-V4-Flash and GPT-5.4-mini. No task-specific model fine-tuning is performed. For the neural configurations, we keep the graph facts, embedding model, retrieval configuration, query-conditioned filtering rules, prompting strategy, and answer normalization procedure fixed across the two LLM backends.

We compare the following configurations:

- **Full**: the complete framework. The selective symbolic solver first attempts to answer each question directly from the structured scene graph. When the symbolic solver abstains, the question is processed by the graph retrieval, evidence filtering, and LLM reasoning pipeline.
- **LLM-only**: a direct ablation of the selective symbolic route. The symbolic solver is disabled, and every question is processed by graph retrieval, query-conditioned evidence filtering, and LLM reasoning.
- **Symbolic-only**: a template-aware symbolic baseline. This baseline operates directly on the same unfiltered spatiotemporal scene graph and uses an expanded deterministic parser and graph-execution rule set. It attempts all five question types, without querying the graph retrieval or invoking an LLM.

The Symbolic-only configuration is intended to measure the capability of deterministic graph reasoning without neural inference. Its rule set is broader than the selective symbolic solver used in the Full framework, particularly because it contains additional rules for comparison questions. Therefore, Symbolic-only is treated as a standalone baseline rather than a strict ablation of the Full configuration. Since it does not use an LLM, its result is independent of the selected LLM backend and is reported only once.

Evaluation metric. Following NuScenes-QA Qian et al. (2024), we report the averaged accuracy for each question type over the complete evaluation set. Predictions are canonicalized to the corresponding answer space before evaluation. For the Symbolic-only baseline, unresolved object or status queries are marked as invalid. For the neural configurations, API failures, unparsable responses, and outputs outside the valid answer space are marked as invalid. Invalid cases remain in the evaluation denominator and are scored as incorrect.

4.2 Main Results

Table 1 reports the results of the complete framework, the LLM-only ablation, and the standalone Symbolic-only baseline. The values in the table are accuracies in percent.

The Full framework with GPT-5.4-mini achieves the highest overall accuracy, reaching 80.63% and correctly answering 4,770 of the 5,916 questions. The corresponding Full configuration with

DeepSeek-V4-Flash achieves 77.89%. Among all evaluated configurations, Full GPT obtains the best results on existence, counting, object, and status questions, while GPT LLM-only obtains the best result on comparison questions. When comparing the two Full configurations directly, GPT-5.4-mini outperforms DeepSeek-V4-Flash on all five question types.

Effect of selective symbolic execution. We measure the contribution of the selective symbolic route by comparing Full with LLM-only under the same LLM backend. As shown in Figure 3(a), adding symbolic execution improves the overall accuracy of DeepSeek-V4-Flash from 71.25% to 77.89%, corresponding to a gain of 6.64 percentage points. Under GPT-5.4-mini, the overall accuracy increases from 75.15% to 80.63%, corresponding to a gain of 5.48 points.

The largest gains occur on counting questions. The DeepSeek-based system improves from 62.56% to 75.17%, a gain of 12.61 points, whereas the GPT-based system improves from 65.45% to 75.65%, a gain of 10.20 points. These results indicate that exact candidate-set operations and identity filtering complement neural reasoning, particularly for questions requiring enumeration or precise object selection.

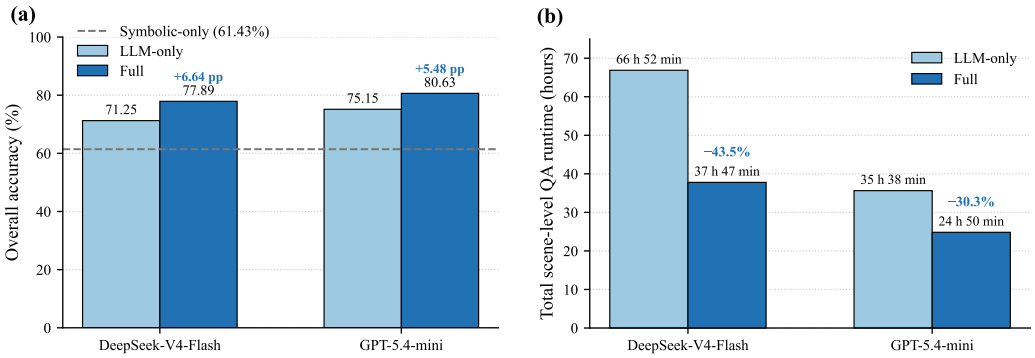

Figure 3: **Accuracy and scene-level QA runtime of the evaluated configurations under two LLM backends.** (a) Overall accuracy on the 5,916-question evaluation set. (b) Total scene-level QA wall-clock time summed over the ten scene sequences for the LLM-only and Full configurations. The Symbolic-only baseline is omitted from this hour-scale runtime panel because it uses only local deterministic graph operations, involves no remote API calls, and completes in negligible time relative to the neural configurations. Blue annotations indicate accuracy gains in percentage points (pp) in (a) and relative runtime reductions in (b).

Accuracy–efficiency trade-off. In addition to accuracy, we measure the wall-clock time of the scene-level QA stage. Timing begins after constructing the scene graph, and ends after all questions in the scene have been answered and parsed. The measured interval includes symbolic execution, graph retrieval, evidence filtering, neural answer generation when required, response parsing, and API retry delays. Graph construction and index preparation are excluded.

As shown in Figure 3(b), across the ten scene sequences, the Full configuration reduces the total QA runtime from 66 h 52 min to 37 h 47 min with the DeepSeek backend, corresponding to a 43.5% reduction. With GPT-5.4-mini, the runtime decreases from 35 h 38 min to 24 h 50 min, corresponding to a 30.3% reduction. Dividing the total runtime by the 5,916 questions gives an amortized runtime per question of 40.69 s versus 22.99 s for DeepSeek, and 21.68 s versus 15.11 s for GPT-5.4-mini, for LLM-only and Full, respectively. Together with the accuracy gains in Figure 3(a), these measurements show that Full improves both accuracy and runtime under the evaluated conditions. This result is consistent with the intended benefit of answering supported questions through symbolic execution before invoking retrieval and neural inference.

Standalone Symbolic-only baseline. The standalone Symbolic-only baseline achieves 61.43% overall accuracy without querying LightRAG or invoking an LLM. It obtains 76.22% on existence questions and 65.57% on comparison questions, showing that deterministic template parsing combined with explicit graph execution can solve a substantial subset of the benchmark.

However, the baseline remains below both neural configurations, especially on counting and object questions. This gap reflects the limited coverage of fixed symbolic rules when questions contain linguistic variation, ambiguous references, or multiple possible candidate objects. The result motivates the combination of symbolic execution with a retrieval-augmented neural fallback.

Because Symbolic-only uses an expanded rule set, its result is not interpreted as the direct performance obtained by merely deleting the LLM from Full. Instead, it provides an independent reference for the upper range of deterministic reasoning implemented in this study.

Effect of the LLM backend. GPT-5.4-mini achieves higher overall accuracy than DeepSeek-V4-Flash in both Full and LLM-only configurations. Detailed statistics are provided in Appendix A.7.

Comparison questions require separate interpretation when comparing Full with LLM-only under the same LLM backend. In Full, the symbolic solver abstains on comparison questions, so both configurations answer these questions through the neural path. Therefore, the observed difference between Full and LLM-only on comparison questions under the same LLM backend cannot be attributed to symbolic answer execution. It may instead reflect run-to-run stochasticity or differences in neural retrieval and answer generation behavior.

5 Discussion

The improvements across two LLM backends support selective symbolic execution as a useful strategy for structured driving-scene QA. Supported queries bypass retrieval and neural generation, while unresolved questions retain access to the neural fallback. Explicit graph facts and filtered evidence also make intermediate information inspectable. Additional discussion of these architectural properties is provided in Appendix A.9.

The evaluation is limited to frame-specific questions from ten nuScenes v1.0-mini scenes under an oracle-perception setting. It does not establish robustness to learned perception errors or the benefit of temporal links for explicit cross-frame reasoning. Symbolic answers depend on query parsing and graph facts, while the neural fallback remains sensitive to missing retrieval evidence and overly restrictive filtering. Future work will evaluate larger datasets, learned perception, and questions requiring reasoning across multiple frames.

6 Conclusions

We presented a neuro-symbolic question-answering framework for autonomous driving scenarios that combines an explicit spatiotemporal scene graph with selective symbolic execution and retrieval-augmented language reasoning. The hierarchical graph separates persistent object identities from frame-specific states and explicitly represents spatial relations and temporal transitions. The symbolic solver answers unambiguous queries through deterministic graph operations and delegates unresolved questions to an LLM.

Experiments on NuScenes-QA questions across all ten nuScenes v1.0-mini scenes demonstrate improvements in both accuracy and QA efficiency under an oracle-perception setting. These results support selective symbolic execution as an effective way to improve answer accuracy while reducing reliance on retrieval and neural inference.

The current evaluation focuses on frame-specific questions over ground-truth scene observations. Future work will integrate learned perception, evaluate robustness to incomplete or noisy graph evidence, and extend the evaluation to larger datasets and questions that explicitly require reasoning across multiple frames. These extensions will help assess the framework's scalability and the contribution of its temporal structure to more demanding driving-scene reasoning tasks.

AI Assistance

Large language models were used to assist with coding, language polishing, drafting manuscript sections, suggesting relevant literature, and discussing methodological choices. The use of LLMs as experimental reasoning backends is described in the methodology and experimental setup.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 9112–9141, 2024.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer, 2020. URL https://arxiv.org/abs/2004.05150. arXiv preprint arXiv:2004.05150.

Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11618–11628. IEEE, 2020.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph rag approach to query-focused summarization, 2025. URL https://arxiv.org/abs/2404.16130.

Meng Fang, Shilong Deng, Yudi Zhang, Zijing Shi, Ling Chen, Mykola Pechenizkiy, and Jun Wang. Large language models are neurosymbolic reasoners. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 17985–17993, 2024.

Qiao Gu, Alihusein Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, Chuang Gan, Celso Miguel de Melo, Joshua B. Tenenbaum, Antonio Torralba, Florian Shkurti, and Liam Paull. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning, 2023. URL https://arxiv.org/abs/2309.16650.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. LightRAG: Simple and fast retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 10746–10761, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.568.

Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14953–14962, 2023. doi: 10.1109/CVPR52729.2023.01436.

Haoyu Han, Yu Wang, Harry Shomer, Kai Guo, Jiayuan Ding, Yongjia Lei, Mahantesh Halappanavar, Ryan A. Rossi, Subhabrata Mukherjee, Xianfeng Tang, Qi He, Zhigang Hua, Bo Long, Tong Zhao, Neil Shah, Amin Javari, Yinglong Xia, and Jiliang Tang. Retrieval-augmented generation with graphs (graphrag). 2025. URL https://arxiv.org/abs/2501.00309.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. 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. 132876–132907. Curran Associates, Inc., 2024. doi: 10.52202/079017-4224.

Nathan Hughes, Yun Chang, and Luca Carlone. Hydra: A real-time spatial perception system for 3d scene graph construction and optimization, 2022. URL https://arxiv.org/abs/2201.13360.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong Park. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7036–7050, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.389.

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

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638.

Linhao Luo, Yuan-Fang Li, Reza Haffari, and Shirui Pan. Reasoning on graphs: Faithful and interpretable large language model reasoning. In *International Conference on Learning Representations*, volume 2024, pp. 14400–14423, 2024.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. Routellm: Learning to route llms with preference data, 2025. URL https://arxiv.org/abs/2406.18665.

Tianwen Qian, Jingjing Chen, Linhai Zhuo, Yang Jiao, and Yu-Gang Jiang. Nuscenes-qa: A multi-modal visual question answering benchmark for autonomous driving scenario, 2024. URL https://arxiv.org/abs/2305.14836.

Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Jens Beißwenger, Ping Luo, Andreas Geiger, and Hongyang Li. Drivelm: Driving with graph visual question answering, 2025. URL https://arxiv.org/abs/2312.14150.

Xiaoyu Tian, Junru Gu, Bailin Li, Yicheng Liu, Yang Wang, Zhiyong Zhao, Kun Zhan, Peng Jia, Xianpeng Lang, and Hang Zhao. Drivevlm: The convergence of autonomous driving and large vision-language models, 2024. URL https://arxiv.org/abs/2402.12289.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Orion Weller, Dawn Lawrie, and Benjamin Van Durme. NevIR: Negation in neural information retrieval. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2274–2287, St. Julian's, Malta, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.eacl-long.139.

Shun-Cheng Wu, Johanna Wald, Keisuke Tateno, Nassir Navab, and Federico Tombari. SceneGraphFusion: Incremental 3D Scene Graph Prediction from RGB-D Sequences. In *Conference on Computer Vision and Pattern Recognition*, pp. 7515–7525, 2021. doi: 10.1109/CVPR46437.2021.00743.

Xiaojun Wu, Cehao Yang, Xueyuan Lin, Chengjin Xu, Xuhui Jiang, Yuanliang Sun, Hui Xiong, Jia Li, and Jian Guo. Think-on-graph 3.0: Efficient and adaptive llm reasoning on heterogeneous graphs via multi-agent dual-evolving context retrieval, 2026. URL https://arxiv.org/abs/2509.21710.

Zhenhua Xu, Yujia Zhang, Enze Xie, Zhen Zhao, Yong Guo, Kwan-Yee K. Wong, Zhenguo Li, and Hengshuang Zhao. Drivegpt4: Interpretable end-to-end autonomous driving via large language model. *IEEE Robotics and Automation Letters*, 9(10):8186–8193, 2024. doi: 10.1109/LRA.2024.3440097.

Xiao-Wen Yang, Jie-Jing Shao, Lan-Zhe Guo, Bo-Wen Zhang, Zhi Zhou, Lin-Han Jia, Wang-Zhou Dai, and Yu-Feng Li. Neuro-symbolic artificial intelligence: Towards improving the reasoning abilities of large language models. In James Kwok (ed.), *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pp. 10770–10778. International Joint Conferences on Artificial Intelligence Organization, 8 2025. doi: 10.24963/ijcai.2025/1195. Survey Track.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023. URL https://arxiv.org/abs/2210.03629.

Xiangrong Zhu, Yuexiang Xie, Yi Liu, Yaliang Li, and Wei Hu. Knowledge graph-guided retrieval augmented generation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 8912–8924, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.449.

Table 2: Per-scene frame and question statistics for the nuScenes v1.0-mini evaluation subset.

Scene index	Scene name	Frames	Questions
0	scene-0061	39	645
1	scene-0103	40	478
2	scene-0655	41	550
3	scene-0553	41	751
4	scene-0757	41	842
5	scene-0796	40	695
6	scene-0916	41	608
7	scene-1077	41	335
8	scene-1094	40	584
9	scene-1100	40	428
Total	**10 sequences**	**404**	**5,916**

A Appendix: Additional Implementation and Details

A.1 Per-Scene Evaluation Statistics

Table 2 lists the ten scene sequences used in the evaluation. The number of frames is obtained by following the sample chain from each scene's first sample to its last sample. The question count includes all retained NuScenes-QA questions whose sample tokens belong to the corresponding scene sequence.

The sequence length limit is set to 41, which covers the longest scene in the v1.0-mini release. Scenes with fewer than 41 keyframes use all of their available keyframes. In the present evaluation, the ten sequences contain 39–41 keyframes, so no scene requires padding or truncation. Questions are selected using their original sample tokens, ensuring that each question remains associated with its original keyframe.

A.2 Ground-Truth Spatiotemporal Graph Construction

For every keyframe, we read the ground-truth object annotations from nuScenes. Each annotation provides an object category, an instance identity, an attribute description, and a 3D bounding box. The bounding box is transformed into the ego-vehicle coordinate system using the ego pose associated with the front camera. The transformed BEV center is used to determine the ego-centric spatial relation.

A persistent entity is created for each instance identity appearing in the scene sequence. For every observed instance at frame t, the graph contains a corresponding temporal state node with a unique frame-specific identifier. The state node stores the frame association, object category, motion status, and other attributes required by the downstream reasoning modules.

The graph contains the following edge types:

- **Ego-centric spatial edges**: each state node is connected to the ego-reference node using one of the six directional predicates defined by NuScenes-QA;
- **Object-to-object spatial edges**: directed spatial edges are generated between ordered pairs of objects observed at the same frame;
- **Hierarchy edges**: a persistent entity is connected to each of its frame-specific state nodes through a `HAS_STATE` edge;
- **Temporal edges**: successive observations of the same persistent instance are connected through a `NEXT_STATE` edge.

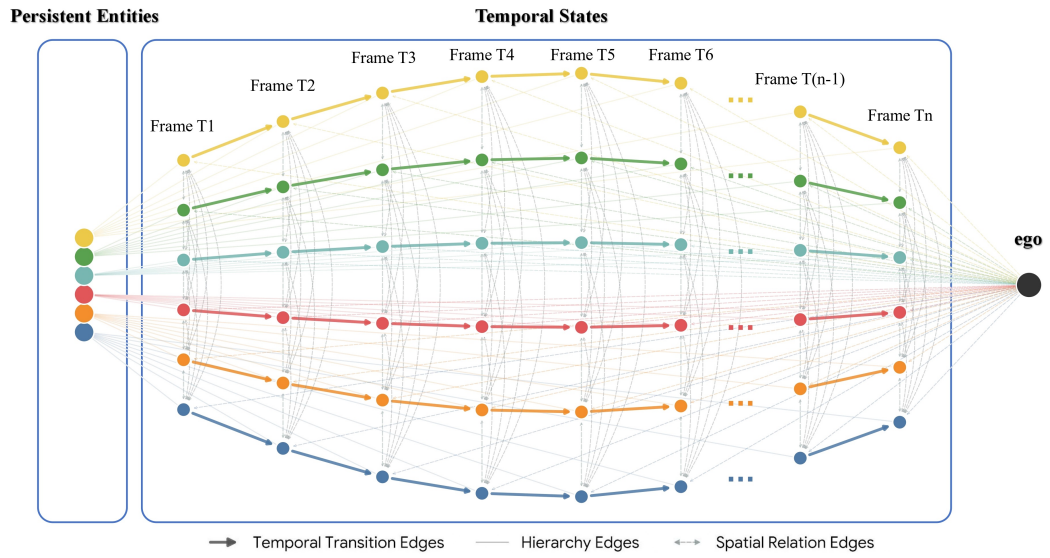

Figure 4: **Hierarchical spatiotemporal scene graph.** Each color denotes a persistent entity and its frame-specific states. Hierarchy edges link entities to their states, thick arrows connect successive observations of the same entity, and dashed edges represent directed spatial relations between states within each frame and between states and the ego vehicle.

The spatial predicate is computed from the subject-relative displacement between two object centers in the ego coordinate system. We use the six-direction taxonomy and angular convention defined by NuScenes-QA Qian et al. (2024); the exact angular intervals are therefore not repeated here.

Object-to-object spatial relations are generated without an explicit distance threshold. To avoid uninformative background-background connections, relations whose two endpoints are both traffic cones or barriers are excluded. The graph therefore retains long-range object relations while controlling redundant relations between static traffic-control objects.

For motion status, the implementation uses the following category-aware defaults when no explicit attribute is available: vehicles are initialized as `stopped`, pedestrians as `standing`, bicycles and motorcycles as `withoutrider`, and barriers and traffic cones as `none`. When an attribute is present, the corresponding attribute-derived status overrides the default.

A.3 Graph Projection and LightRAG Configuration

The complete graph is saved as a scene-level JSON object and remains available to the symbolic reasoning module. For retrieval, the graph is projected into a task-oriented structured representation and inserted through LightRAG's custom-KG interface Guo et al. (2025). The main indexed view contains a scene-summary record, frame-specific state records, and spatial-relation records.

Each indexed relation is represented by its source identifier, target identifier, directional predicate, and frame association. Its canonical description preserves the subject–predicate–reference ordering, for example: `Fact: A IS FRONT_OF B. frame_idx=5`. This representation keeps the direction of each spatial fact explicit.

In the reported main implementation, the persistent entity layer, `HAS_STATE` edges, and `NEXT_STATE` edges remain in the complete scene graph used by symbolic reasoning. The retrieval projection focuses on the records required by the current frame-specific QA tasks. This separation allows the graph to retain longer-term structural information without requiring every graph element to be retrieved for every question.

The graph is inserted directly as structured custom knowledge. The implementation does not call the ordinary document insertion path for the scene text and does not ask an LLM to re-extract the ground-truth entities and relations. The embedding model is `text-embedding-3-small` with embedding dimension 1536. Each scene is assigned an independent LightRAG working directory.

Table 3: Main retrieval and indexing settings.

Component	Setting
Retrieval mode	local
Reranking	disabled
Embedding model	text-embedding-3-small
Embedding dimension	1536
Chunk token size	8192
Embedding batch size	48
Entity extraction gleaning	0

The neural configurations use LightRAG local retrieval, and the retrieval query consists of the natural language question together with a textual anchor for the question's sample-specific frame. The main retrieval settings are summarized in Table 3.

A.4 Query-Conditioned Retrieval and Evidence Filtering

The target frame is obtained from the question's sample token through the scene-level token-to-frame mapping. It is not inferred from the natural language question. For questions that reach the neural fallback, the system executes the following two-step process:

$$\mathcal{H}_Q^0 = \mathcal{R}_{\mathrm{LR}}\left(Q \oplus h(t_q); \mathcal{G}_R\right), \tag{12}$$

$$\mathcal{H}_Q = \mathcal{F}_{\mathrm{filter}}\left(\mathcal{H}_Q^0; t_q, \mathcal{C}_q, \mathcal{R}_q, Q\right). \tag{13}$$

First, LightRAG retrieves candidate entities and relations using the question and the frame anchor. Second, the returned serialized evidence is filtered using deterministic compatibility rules. The filtering procedure applies four practical constraints:

1. **Frame constraint**: records associated with frames other than the queried frame are removed when an exact frame is available;
2. **Semantic constraint**: records belonging to irrelevant object categories are removed. For queries containing generic terms such as *thing* or *object*, we relax category filtering to retain retrieved evidence across traffic-participant categories. Barriers and traffic cones are retained only when explicitly mentioned in the query, either as targets or reference objects;
3. **Directional constraint**: spatial relations are retained only when their directional predicate matches a spatial expression in the question. If a question contains no spatial constraint, spatial relation records are removed;
4. **Ego-reference constraint**: ego-relative relation records are retained only when the question refers to the ego vehicle through expressions such as *me*, *ego*, or *my car*.

The filtering step operates on the serialized retrieval result before it is inserted into the LLM prompt. The resulting evidence is passed to a question-type specific prompt.

A.5 Symbolic Routing and Baseline Details

The Full configuration uses a conservative symbolic-first route. The symbolic solver directly reads the complete Python scene graph and attempts to resolve the question before LightRAG retrieval is performed. Its behavior can be summarized as:

$$\hat{a} = \begin{cases} \Phi_{\mathrm{sym}}(Q, \mathcal{G}, t_q), & \Phi_{\mathrm{sym}}(Q, \mathcal{G}, t_q) \neq \bot, \\ \Phi_{\mathrm{llm}}(Q, \mathcal{H}_Q), & \Phi_{\mathrm{sym}}(Q, \mathcal{G}, t_q) = \bot. \end{cases} \tag{14}$$

The symbolic solver supports deterministic existence and counting queries, as well as object and status queries when the remaining candidates yield a unique answer. It counts persistent global identifiers rather than frame-specific state identifiers. Comparison questions and unsupported multi-hop patterns are delegated to the neural fallback in the Full configuration.

The Symbolic-only baseline uses a different, expanded solver. It directly loads the unfiltered scene graph saved by the main experiment and does not query LightRAG. Its parser recognizes object categories, motion statuses, distance bands, ego-relative spatial expressions, and nested spatial descriptions. It also contains comparison-specific rules, which allows it to attempt comparison questions without an LLM.

The Symbolic-only baseline applies the following answer conditions:

- existence questions return `yes` or `no` according to whether the constrained candidate set is empty;
- counting questions count unique persistent entity identifiers;
- object questions return a category only when the candidate categories are unique;
- status questions return a status only when one persistent entity and one status are identified;
- comparison questions resolve two descriptors and compare the selected attribute, such as motion status, category, direction, distance band, or color.

When an object or status answer cannot be uniquely determined, the Symbolic-only implementation returns an invalid result. For comparison questions, unresolved or non-unique descriptors are handled by the deterministic comparison rule used by the baseline. These behaviors are included to make the reported baseline reproducible.

A.6 Prompting and Answer Normalization

The neural fallback uses question-type-specific prompts for existence, counting, object, status, and comparison questions. The prompt contains the original question, the filtered evidence, and the corresponding output constraints. The required final format is a JSON object of the form {`"answer": "value"`}.

The answer spaces are:

- **Existence**: `yes` or `no`;
- **Counting**: a non-negative integer;
- **Object**: one of the supported object categories;
- **Status**: one of the supported motion-status labels;
- **Comparison**: `yes` or `no`.

Before scoring, capitalization, punctuation, spaces, underscores, and known aliases are normalized. For example, *traffic cone* is mapped to `trafficcone`, *construction vehicle* is mapped to `constructionvehicle`, and *person* is mapped to `pedestrian`. Object names with trailing instance numbers are reduced to their category.

The implementation also applies category-aware status conventions. For example, `withrider` and `withoutrider` are used for bicycles and motorcycles, while `moving`, `parked`, and `stopped` are used for vehicles. These conventions prevent status labels from being assigned across incompatible object categories.

A.7 Detailed Prediction Statistics

Table 4 reports the number of correct and invalid predictions for every configuration. Each cell is formatted as *correct/invalid*. Valid but incorrect predictions are not included in the invalid count.

Table 4: Detailed prediction statistics on the complete v1.0-mini evaluation subset. Each entry reports *correct/invalid*; invalid cases remain in the denominator and are scored as incorrect.

Question type	Total	Symbolic-only	LLM-only (DeepSeek)	Full (DeepSeek)	LLM-only (GPT)	Full (GPT)
Exist	1,678	1,279/0	1,383/9	1,435/1	1,415/6	1,490/9
Count	1,039	534/0	650/35	781/20	680/0	786/6
Object	1,245	538/518	744/394	863/295	769/3	870/17
Status	920	605/253	751/4	819/2	794/0	858/3
Comparison	1,034	678/0	687/54	710/32	788/0	766/17
Overall	5,916	3,634/771	4,215/496	4,608/350	4,446/9	4,770/52

Table 5: Scene-level QA wall-clock runtime for the ten nuScenes v1.0-mini scene sequences. Runtime is reported as hours:minutes (h:mm). Reduction denotes the relative runtime decrease of Full compared with LLM-only under the same LLM backend.

	GPT			DeepSeek		
Scene	**Full**	**LLM-only**	**Reduction**	**Full**	**LLM-only**	**Reduction**
scene0	4:36	8:08	43.4%	8:03	12:49	37.2%
scene1	2:45	2:54	5.2%	3:05	4:59	38.1%
scene2	3:22	4:19	22.0%	2:41	6:38	59.5%
scene3	3:35	4:24	18.6%	5:15	10:12	48.5%
scene4	1:22	2:20	41.4%	4:11	8:09	48.7%
scene5	1:35	2:02	22.1%	3:12	5:11	38.3%
scene6	3:25	4:49	29.1%	4:24	6:50	35.6%
scene7	0:59	1:02	4.8%	1:08	2:07	46.5%
scene8	2:32	4:19	41.3%	3:57	5:35	29.3%
scene9	0:39	1:21	51.9%	1:51	4:22	57.6%
Total	**24:50**	**35:38**	**30.3%**	**37:47**	**66:52**	**43.5%**

A.8 Scene-level Runtime Measurement

We additionally measure the wall-clock time required to answer all questions associated with each scene sequence. The evaluation is performed sequentially in the original question order. For each scene, the timer starts after the scene graph and the prebuilt LightRAG index have been loaded and initialized, and stops after the last question has been answered and parsed. The measured interval includes symbolic execution, LightRAG retrieval, query-conditioned evidence filtering, neural answer generation, response parsing, network waiting time, and API retry delays.

The timer excludes nuScenes database initialization, scene-graph construction, graph indexing, embedding computation during index construction, API preflight checks, and final result-file serialization. Therefore, the reported quantity is scene-level QA inference time with prebuilt graph and retrieval indexes rather than end-to-end perception or simulator execution time.

For each question, we also record the individual wall-clock latency and the execution route. A question is assigned to the symbolic route when the solver returns a valid answer; otherwise, it is assigned to the neural fallback route. Failed and invalid queries remain included in the total runtime and in the accuracy denominator.

The ten scenes contain 5,916 questions in total and are processed without inter-question parallelism. Consequently, the reported scene-level time reflects sequential evaluation throughput. Because the neural path includes remote API calls, the absolute runtime depends on backend serving latency, network conditions, queueing, and retry behavior. The paired Full-versus-LLM-only comparison under the same backend is therefore the primary comparison for measuring the efficiency contribution of selective symbolic execution.

Across all ten scenes, GPT-5.4-mini requires 24 h 50 min in the Full configuration and 35 h 38 min in the LLM-only configuration. DeepSeek-V4-Flash requires 37 h 47 min and 66 h 52 min, re-

spectively. Thus, the Full configuration saves 10 h 48 min with GPT-5.4-mini and 29 h 05 min with DeepSeek-V4-Flash. The reduction is observed in every scene for both backends, although its magnitude varies with the retrieval cost and backend serving latency.

A.9 Additional Discussion of Architectural Properties

Driving vision-language models such as DriveLM use neural inference for driving-scene question answering Sima et al. (2025), while VISPROG uses an LLM to generate executable visual programs Gupta & Kembhavi (2023). Graph-based RAG frameworks such as LightRAG retrieve structured evidence for neural answer generation Guo et al. (2025). Our framework places a conservative symbolic executor before both retrieval and LLM inference, allowing supported, unambiguous queries to be answered directly from the scene graph. This selective design reduces reliance on neural computation for supported queries and provides explicit graph evidence for inspecting the answering process. Its main architectural properties are discussed below:

1. **Deterministic Graph Operations with Selective Execution:** For supported queries, the symbolic solver evaluates explicit spatial relations, filters candidate objects, and performs set operations over persistent identities. This provides deterministic results for the parsed constraints on the available graph. Answer correctness nevertheless depends on the graph facts and the interpretation of the question. The solver therefore abstains on unsupported patterns or unresolved ambiguities and delegates these cases to the neural fallback.
2. **Reasoning without Task-Specific Fine-Tuning:** The framework uses pretrained LLMs without task-specific parameter updates. Task adaptation is implemented through the graph schema, symbolic rules, evidence filtering, and question-type-specific prompts. The experiments demonstrate this approach with two LLM backends.
3. **Inspectable Evidence and Intermediate Results:** Explicit entity identifiers, frame associations, and directed relations make the evidence used for answering questions inspectable. Symbolic candidate sets and filtered retrieval records provide intermediate results that can support error analysis.
4. **Explicit Identity and Temporal Structure:** Persistent entity nodes connect observations of the same object across frames, while `NEXT_STATE` edges record successive observed states. This representation organizes scene information across a sequence and provides a basis for temporal queries. The current evaluation focuses on frame-specific questions, so the benefit of temporal transitions for cross-frame reasoning has not yet been established.
5. **Modular Perception and Reasoning Interfaces:** Structured object observations provide an explicit interface between perception, graph construction, and reasoning. The reasoning backend can therefore be changed while retaining the graph representation and retrieval pipeline, as illustrated by the two evaluated LLM backends. This modularity does not prevent perception errors from affecting downstream answers; its robustness with learned perception requires further study.

A.9.1 Accuracy and QA Efficiency

The observed accuracy gains are accompanied by reductions in QA wall-clock time of 30.3% with GPT-5.4-mini and 43.5% with DeepSeek-V4-Flash relative to their corresponding LLM-only configurations. These results are consistent with the routing design: questions answered symbolically bypass retrieval and neural generation, while unresolved questions retain access to the LLM fallback. The measurements cover only the QA stage and exclude graph construction and index preparation. They therefore demonstrate improved QA efficiency under the evaluated conditions.

A.9.2 Limitations and Future Work

The evaluation uses ground-truth observations from ten nuScenes v1.0-mini scenes. Future work will integrate an upstream perception module through the structured observation interface to evaluate the framework on practical driving tasks. We will also introduce questions that explicitly require cross-frame reasoning to assess the effectiveness of the hierarchical structure in handling more complex reasoning tasks.

In addition, both reasoning routes retain potential sources of error. Symbolic execution depends on the coverage and correctness of the query parser. For the neural fallback, deterministic filtering can remove incompatible records but cannot recover relevant facts absent from the retrieved candidate set. Overly restrictive filters may also discard useful evidence.