Graph Serializations for Reasoning LLMs: A Study Across Knowledge Extraction and Structural Reasoning

Julien Yu, Mihaela Bornea, Avirup Sil, Katrin Kirchhoff
Oracle
{julien.yu, mihaela.bornea, avi.sil, katrin.kirchhoff}@oracle.com

Abstract

The way a knowledge graph is serialized as text for inclusion into LLM context can substantially affect downstream LLM performance, with prior work reporting tens of percentage points variations across formats. However, existing studies consider only a limited set of representations and predate frontier reasoning models, which now offer stronger reasoning, instruction-following, and long-context capabilities. We evaluate 15 graph representation formats across nine frontier LLM configurations on ontology-based knowledge extraction and structural graph reasoning tasks. While knowledge extraction is relatively insensitive to different serialization formats, the choice of the optimal format for structural reasoning tasks significantly affects accuracy. We demonstrate that the optimal representation choice depends on model, query, and task and propose to approach graph serialization as a dynamic orchestration decision.

1 Introduction

LLMs increasingly serve as reasoning components in systems that consume symbolic structure, including ontologies, knowledge graphs (KGs), and graph-derived retrieval contexts. Applications include ontology-guided extraction (Mihindukulasooriya et al., 2023), multi-hop reasoning over KGs (Sun et al., 2024; Ma et al., 2025; Zhou et al., 2025), and generation grounded in retrieved graph evidence (Edge et al., 2024). While it is possible to connect LLMs to dedicated graph query engines, it is increasingly common to inject graph representations directly into LLM context, via serialization of nodes, edges, types, relations, and constraints into a sequence of tokens. Prior work shows that the representation choice can substantially affect downstream task accuracy while holding the model and task fixed (Fatemi et al., 2024; Markowitz et al., 2025; Sclar et al., 2024; He et al., 2024; Sui et al., 2024). However, previous studies only investigated a small number of possible graph representations, did not test frontier reasoning models, and did not analyze the relationship between query/task type, graph properties, models, and reasoning level. In particular, it remains unclear whether frontier reasoning models render serialization less important (by being able to infer structure from most reasonable representations), or more important by exploiting semantic, constraint, or layout cues when exposed explicitly.

We organize the study around three research questions: **RQ1: Representation selection.** Which graph representations are the most effective across different tasks, graph types, frontier models, and reasoning effort? **RQ2: Semantic versus structural representation.** Are the most effective representations aligned with the task type, with semantically rich representations favoring semantic extraction and connectivity-focused representations favoring structural reasoning? **RQ3: Reasoning capacity and cost.** As model reasoning strength increases, does serialization become less important, and how does the preferred representation move along the accuracy–token-cost frontier?

We make three contributions: **First, we introduce a broad empirical evaluation of graph representations for frontier LLMs.** We compare 15 representations spanning semantic-web formats, constraint-style renderings, structured natural language, tabular views, diagrammatic formats, compact connectivity encodings, and verbalization of graph structures. **Second, we evaluate representation choices across both ontology-driven semantic knowledge extraction and structural graph reasoning.** We use Text2KGBench (Mihindukulasooriya et al., 2023) for ontology-based extraction and extend GraphQA (Fatemi et al., 2024) with harder graph sizes, structural operators, and topology variation. **Third, we analyze graph representation selection as a model- and cost-aware**

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 11779–11794
October 24-29, 2026 ©2026 Association for Computational Linguistics

orchestration decision. We evaluate nine frontier LLM configurations, including reasoning-enabled models, and measure both task accuracy and token-cost behavior. We show that a query-dependent selector model can increase the average performance by 9.52 pp over the best fixed format. Together, these contributions position graph serialization as an explicit, dynamic system design choice for KG-augmented LLM pipelines rather than a static prompt-formatting detail.

2 Related Work

Studies leveraging KGs include (Sun et al., 2024; Ma et al., 2025), and constrained KG reasoning frameworks (Zhou et al., 2025; Luo et al., 2024a), which have demonstrated improved retrieval, traversal, decoding, and answer generation over graph contexts. Ontology-guided extraction and schema-constrained KG construction methods (Mihindukulasooriya et al., 2023; Zhang and Soh, 2024) require models to use both semantic labels and topology. Since these systems ultimately prompt LLMs with serialized graphs, they can directly benefit from graph representations that are more efficient. Talk-Like-a-Graph (TLAG) (Fatemi et al., 2024) showed that graph representations can affect accuracy by 4.8–61.8 pp on synthetic structural reasoning tasks. GraphArena (Tang et al., 2025) broadens graph reasoning evaluation but largely fixes the input representation. NLGraph (Wang et al., 2023) evaluates graph problems stated in natural language; GraphInstruct (Luo et al., 2024b) targets instruction tuning; and GraphToken (Perozzi et al., 2024) bypasses textual representations with learned graph-token embeddings. LLM-KG-Bench 3.0 (Meyer et al., 2025) evaluates LLMs on RDF and SPARQL generation, but gives limited attention to graph representation format. Closest to our setting, KG-LLM-Bench (Markowitz et al., 2025) shows that textualizing a knowledge graph matters: across KG understanding tasks, language models, and textualization strategies, representation choice produces 17.5 pp performance variation. We view this result as the starting point for studying graph-to-text interfaces; however, KG-LLM-Bench does not answer which representational properties matter when the design space includes semantic-web formats, constraint-oriented ontology views, diagrammatic formats, tabular renderings, compact structural encodings, and prose-like representations. Nor does it test whether textualization sensitivity persists for frontier reasoning LLMs, whose stronger instruction following and long-context reasoning could plausibly make representation choice less important. In addition to addressing these questions, our study approaches graph serialization as a model, task, and cost-dependent orchestration decision and proposes a dynamic selector model.

formal semantics (RDF/OWL)	`turtle`, `jsonld`, `shacl_lite`
structured natural language	`text2kgbench`, `entity_centric` `relation_indexed`
tabular	`markdown_table`, `toon`
visual / diagrammatic	`mermaid`, `dot`
connectivity-focused	`adjacency`, `edge_list`, `structured`
prose	`nl_summary`, `friendship`

Table 1: Graph formats by semantic explicitness.

3 Method

Representations. We evaluate 15 serialization formats, grouped into six descriptive families (Tab. 1): RDF and constraint-oriented Semantic Web formats (`turtle` (Beckett et al., 2014), `jsonld` (Kellogg et al., 2020), and the SHACL-derived `shacl_lite` (Knublauch and Kontokostas, 2017)); structured natural-language formats (`text2kgbench` (Mihindukulasooriya et al., 2023), `entity_centric`, and `relation_indexed`); tabular formats (`markdown_table` and `toon`); textual diagram languages (`mermaid` and `dot`); connectivity-focused encodings (`adjacency`, `edge_list`, and `structured`); and prose formats (`nl_summary` and `friendship` (Fatemi et al., 2024)). These representation formats differ in how they expose graph information to LLMs and range from semantically explicit representations that expose concept types, relations, and constraints to compact encodings that primarily expose graph connectivity.

Tasks and metrics. *Ontology-based knowledge extraction.* We evaluate the 2,014-sentence test split of Text2KGBench (Mihindukulasooriya et al., 2023), drawn from the full `DBPEDIA_WEBNLG` benchmark spanning 19 ontologies. It pairs each sentence with a serialized ontology and asks the model to extract all supported (subject, relation, object) triples. The task requires semantic understanding of the sentence and ontology-aware reasoning over concept types and relations.

Structural graph reasoning. GraphQA (Fatemi et al., 2024) tests whether LLMs can recover graph properties such as edges, degrees, neighbors, counts, paths, and cycles, from textual graph encod-

ings. Such tasks relate to multi-hop KG reasoning, where answers depend on paths and neighborhoods (Chen et al., 2024). We evaluate three regimes. **GraphQA** contains 11 tasks: 9 original structural tasks from TLAG together with 2 summary tasks we add (`num_components`, `connectivity`), over Erdős–Rényi graphs with $n \in \{8, 15, 25, 35, 50\}$. **GraphQA-Hardened** increases graph size to $n \in \{35, 50, 75\}$ and adds seven operators that are harder to infer from local edge inspection: k-core size (Seidman, 1983), graph diameter, k-shortest path length (Yen, 1971), local clustering coefficient (Watts and Strogatz, 1998), high-degree-neighbor count, maximum clique size (Bron and Kerbosch, 1973; Tomita et al., 2006), and top betweenness-centrality node (Brandes, 2001). **GraphQA-Topology** fixes the graph size at $n = 50$ and varies the generator across Erdős–Rényi, Barabási–Albert, stochastic block model, star, complete, and path graphs (Erdős and Rényi, 1959; Barabási and Albert, 1999; Holland et al., 1983). These extensions let us separate representation effects under larger graphs, harder operators, and different connectivity patterns.

Reasoning over retrieved Wikidata subgraphs. We add an experiment over retrieved subgraphs from Wikidata, using the same models, prompting setup, and 15 graph representations. We select 10 Wikidata seed entities spanning diverse domains, including films, scientists, cities, and companies. For each seed, we retrieve its two-hop neighborhood and capped the resulting context at 450 triples. We generated 110 questions covering relation-specific aggregations, typed entity aggregations, and multi-hop reachability. Examples include: *"How many awards has the scientist Rowan received?", "How many films have at least one director?", "How many distinct entities can be reached from the company Meridian?"*. Entity labels are pseudonymized before serialization, while predicate labels are retained. Each gold answer was computed directly from the canonical subgraph before serialization. Thus, all 15 representations contain the same graph facts and share the same gold answer.

Models and controls. We evaluate nine model configurations: `gpt-5.4` at the default and `high` reasoning-effort settings (OpenAI, 2026a); `gpt-5.4-mini` (OpenAI, 2026b); `o3` (OpenAI, 2025); `claude-sonnet-4-6` with and without extended thinking (Anthropic, 2026); `claude-haiku-4-5` (Anthropic, 2025); and the reasoning and non-reasoning variants of `grok-4-fast` (xAI, 2025). Within each model–task instance, all formats use the same instruction, underlying graph facts, question, seed, node identifiers, node and edge ordering, and task wording; only the serialization changes. Temperature is 0 for all configurations except thinking-enabled `claude-sonnet-4-6`, which uses the API default of 1. To assess repeated-run stability, we additionally rerun all 15 formats three times with caches disabled on fixed GraphQA-Hardened and GraphQA-Topology evaluation sets, using `gpt-5.4` and `gpt-5.4-mini`.

Adaptive Format Selection. We design a query-conditioned, instance-specific selector for graph serialization. We show that it improves serialization choice beyond global and task-level accuracy priors. We construct disjoint, independently seeded calibration and test splits of GraphQA-Hardened (105 questions each) and use `gpt-5.4` as both the serialization selector and answer model. Calibration accuracies define per-task, per-format priors and a five-format candidate set. At test time, five parallel fixed-prompt scoring calls (one per candidate) receive the task, format identity, calibration prior, and graph rendered in that format. The highest-scoring rendering is passed to a single answer call; neither selector rationales nor calibration examples are exposed to the answer model. We compare against the best fixed format on the held-out test set, a calibration-only task router, and five-format parallel majority voting, reporting accuracy, latency, and end-to-end inference cost.

4 Results

We compare graph formats on Text2KGBench and GraphQA and evaluate their effects on task performance, token budgets, and inference costs.

Ontology-based knowledge extraction. In this experiment we serialize the ontology into each candidate format; `text2kgbench` denotes the original compact Text2KGBench serialization. All nine configurations obtain 45–55 F1 (Tab. 2, leftmost column; Fig. 1, top-left). Our best extraction result, `o3` with `shacl_lite` (54.9 F1), is ~ 25 pp above the Mihindukulasooriya et al. (2023) baselines. The variation in F1 score due to the graph representation ranges between 1.3–3.9 percentage points. Most configurations perform best using semantically explicit representations. This result aligns with expectations, as semantically rich representations can better support tasks requiring seman-

model	Text2KGBench	GraphQA	GraphQA-Hardened	GraphQA-Topology
gpt-5.4-high-reasoning	turtle 54.7/1.9	structured 100.0/0.6	—	—
gpt-5.4	mermaid 52.7/3.3	shacl_lite 93.3/4.8	nl_summary 64.2/12.6	jsonld 90.3/15.3
gpt-5.4-mini	shacl_lite 50.6/2.6	jsonld 79.4/8.5	jsonld 41.0/8.4	jsonld 58.3/16.7
o3	shacl_lite 54.9/2.1	adjacency 98.2/2.4	nl_summary 87.4/11.6	structured 90.3/9.7
claude-sonnet-4-6-thinking	jsonld 53.0/2.9	text2kgbench 97.0/3.6	text2kgbench 83.2/4.2	jsonld 95.8/5.5
claude-sonnet-4-6	jsonld 52.2/2.3	nl_summary 94.5/2.4	entity_centric 87.4/13.7	relation_indexed 84.7/6.9
claude-haiku-4-5	structured 50.7/3.9	shacl_lite 90.9/7.9	adjacency 72.6/28.4	nl_summary 80.6/16.7
grok-4-fast-reasoning	shacl_lite 48.8/1.4	nl_summary 97.0/4.8	entity_centric 84.2/15.8	adjacency 95.8/16.7
grok-4-fast-non-reasoning	mermaid 49.9/1.3	adjacency 81.8/17.0	shacl_lite 44.2/21.1	nl_summary 66.7/13.9

Table 2: Best format per (model, benchmark) as "score/spread". Text2KGBench: F1; GraphQA: exact-match accuracy (pp). Spread = max − min across formats (pp).

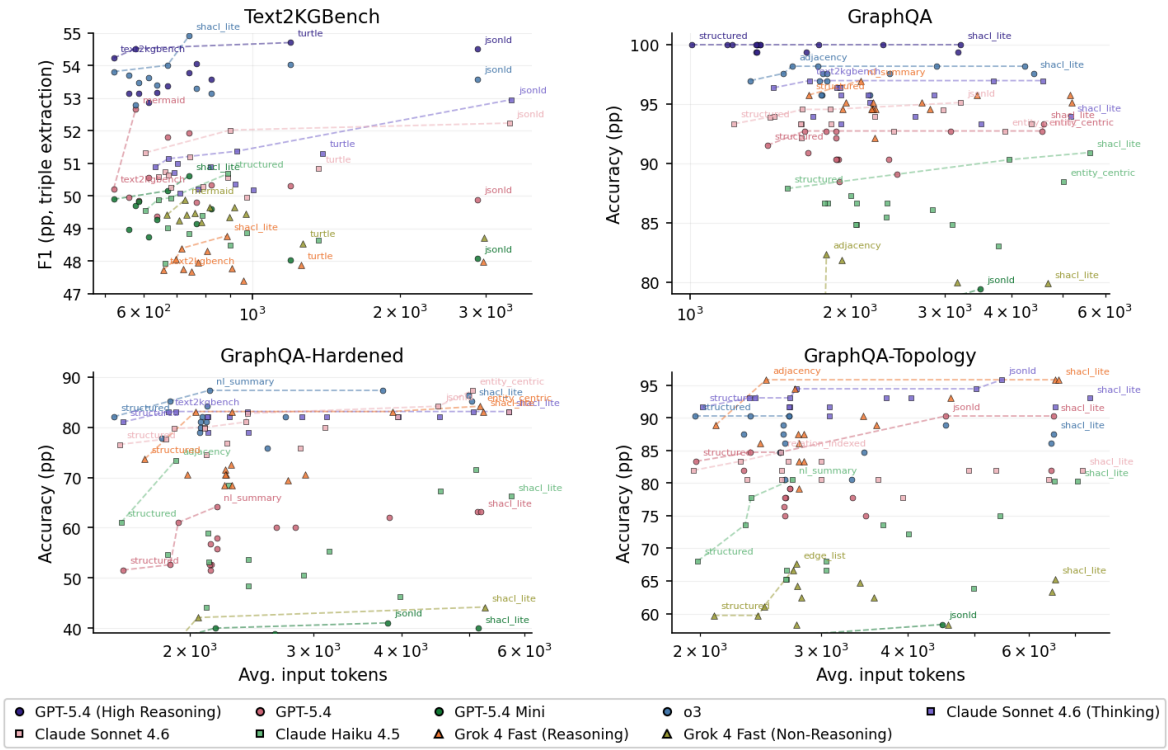

Figure 1: Score vs. input-token cost across the four benchmarks. Per-model Pareto fronts dashed.

tic understanding of concept types and relations. On Text2KGBench's 19 WebNLG domain ontologies, frontier models leave modest room for improvements; gains stem from model capacity rather than serialization choice.

Structural graph reasoning. For structural graph reasoning (Tab. 2) we find that `gpt-5.4-high` solves GraphQA with 100% accuracy using the compact `structured` format, and it is less sensitive to format representation. Format sensitivity increases as LLM reasoning level drops, rising from 4.8 pp on default-tier `gpt-5.4` to 17.0 pp on `grok-4-fast-non-reasoning`. Harder settings further widen the gap for all models: 4.2–28.4 pp on GraphQA-Hardened and 5.5–16.7 pp on GraphQA-Topology. Results in (Tab. 2) show that no single format dominates structural reasoning. The best graph representation is model- and task-dependent. Representation preferences can change with reasoning tier even within a single model family. For the GraphQA task `gpt-5.4-mini` favors `jsonld` (79.4 pp, 3,501 tokens), `gpt-5.4` favors `shacl_lite` (93.3 pp, 4,608 tokens), and `gpt-5.4-high` reaches 100.0 pp with compact `structured` (1,008 tokens). We use a fixed prompt for controlled comparisons; it is not intended to be the optimal prompt for every model family.

Retrieved Wikidata subgraphs. In a targeted evaluation using `gpt-5.4` and `gpt-5.4-mini`, we evaluate 110 questions over 10 Wikidata-derived subgraphs; both configurations rank `jsonld` first and `turtle` second. `gpt-5.4` scores 41.8% and

	LLM calls/query		Resources/query				Outcome
Policy	Selector	Generator	Input tok.	Output tok.	Rel. cost	Latency	Acc. (%)
Global fixed-best	0	1	4,667	296	1.00×	5.3 s	66.67
Task-wise router	0	1	5,124	305	1.08×	5.6 s	64.76
MV-5 (parallel)	0	5	27,231	1,573	5.69×	9.1 s	71.43
Adaptive selector	5	1	17,696	1,034	3.71×	9.4 s	**76.19**

Table 3: End-to-end policy comparison on the same 105 held-out GraphQA-Hardened questions. Counts cover the complete pipeline, and API cost is normalized to global fixed-best. The five selector scores and the five majority voting (MV-5) generations are issued in parallel when measuring latency.

40.0%, compared with 13.6% for `adjacency`, a 28.2 percentage-point spread. `gpt-5.4-mini` scores 30.9% and 29.1%, compared with 6.4% for `nl_summary`, a 24.5 percentage-point spread. The similarly large spreads for `gpt-5.4` and `gpt-5.4-mini` show that representation sensitivity persists beyond synthetic GraphQA and across model capability levels on real-world Wikidata subgraphs.

Representation characteristics and token efficiency. Fig. 1 plots scores against average input tokens; dashed lines show the accuracy vs. token cost tradeoff (per-model Pareto fronts). On Text2KGBench (top-left), scores fall within a narrow F1 band. `shacl_lite`, `text2kgbench` and `mermaid` are token-efficient while achieving high accuracy. In contrast, `turtle` and `jsonld` have a substantially higher token cost. On GraphQA (top-right), reasoning-enabled configurations generally occupy the high-accuracy region across a wide token range, and compact formats such as `adjacency` and `structured` achieve high accuracy at low-cost, even for harder tasks. On harder tasks, GraphQA-Hardened and GraphQA-Topology (bottom row), format choice affects both accuracy and token cost. For LLMs with default or reduced reasoning capabilities, verbose representations may be justified to increase accuracy.

Adaptive Format Selection. The task-wise router using disjoint calibration priors reaches 64.76%, compared with 66.67% for global fixed-best, 71.43% for five-format majority voting, and 76.19% for the fully adaptive selector (Tab. 3). The selector therefore exceeds task-wise routing by 11.43 points, showing that task-level priors alone do not explain its gain, and exceeds majority voting by 4.76 points. Relative to global fixed-best, the selector gains $+\mathbf{9.52}$ percentage points, with a paired-BCa-bootstrap 95% CI of $[+1.90, +16.19]$ and a mid-p McNemar test yielding $p = 0.013$. The post-hoc oracle ceiling over the same five-format candidate slate is 82.86%.

The complete selector pipeline makes five parallel selector calls and one generator call per query, averaging 17,696/1,034 input/output tokens. It incurs 3.71× the API cost of global fixed-best and has an end-to-end latency of 9.4 s. Parallel MV-5 has similar latency at 9.1 s but lower accuracy and higher cost, reaching 71.43% at 5.69× the fixed-best cost. These results demonstrate the feasibility of query-dependent selection on unseen, separately seeded graphs drawn from the same GraphQA-Hardened distribution. Broader transfer across datasets, graph distributions, and selector–generator model families remains future work.

5 Conclusion

We systematically evaluate 15 graph serializations across nine frontier LLM configurations with varied reasoning effort. Serialization effects are modest for ontology-guided extraction (per-format F1 spread up to 3.9 pp) but substantially larger for structural reasoning: up to 28.4 points on controlled GraphQA regimes, and 24.5–28.2 points on retrieved Wikidata contexts.

Graph serialization is therefore a measurable systems decision, not a cosmetic prompt choice. Across the evaluated settings, formats including `shacl_lite`, `mermaid`, and `jsonld` can be strong candidates, but no representation is a universal default; the accuracy–token-cost tradeoff depends on the model, task, and graph regime. Our in-distribution serialization-selector study achieves 9.52 pp improvement over the best fixed format on GraphQA Hardened. More broadly, we suggest that KG-grounded LLM systems should report, evaluate, and, where worthwhile, adapt their graph-to-text interface as a dynamic, model- and task-aware orchestration choice.

Limitations

This study focuses on graph and ontology serialization as text for LLMs; its findings may not generalize to graph-native methods, learned graph tokens, tools, constrained decoding, or end-to-end Graph-RAG systems. Evaluation is limited to Text2KGBench, synthetic GraphQA, and a two-model Wikidata study over fixed retrieved subgraphs. We do not assess retrieval quality, larger contexts, temporal or probabilistic graphs, retrieval errors, or open-ended KGQA.

Our 15 hand-designed serializers are not exhaustive. They vary in explicitness, grouping, ordering, verbosity, and token cost. Paired comparisons control for model, graph facts, question, and prompt. Finally, all evaluated models are closed-weight commercial systems, so we cannot determine the effect of prior exposure to different serialization formats.

Ethical Considerations

The experiments use the publicly released WebNLG split of Text2KGBench (Mihindukulasooriya et al., 2023), synthetic graphs generated for our GraphQA extensions, and cached subgraphs retrieved from Wikidata (Vrandečić and Krötzsch, 2014). We collect no new human-subject data and submit no private, confidential, or organizational data to model endpoints.

Model requests are sent through access-controlled enterprise endpoints. The submitted content consists only of benchmark sentences, serialized ontologies or graphs, and task instructions.

Acknowledgments

The authors used generative AI tools for limited language editing, rewriting suggestions, and assistance in improving exposition. All scientific content, experimental design, code, analyses, claims, citations, and final text were reviewed, verified, and approved by the human authors.

References

Anthropic. 2025. Introducing Claude Haiku 4.5. https://www.anthropic.com/news/claude-haiku-4-5.

Anthropic. 2026. Introducing Claude Sonnet 4.6. https://www.anthropic.com/news/claude-sonnet-4-6.

Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *science*, 286(5439):509–512.

David Beckett, Tim Berners-Lee, Eric Prud'hommeaux, and Gavin Carothers. 2014. Rdf 1.1 turtle: Terse rdf triple language. *W3C recommendation*, 25.

Ulrik Brandes. 2001. A faster algorithm for betweenness centrality. *Journal of mathematical sociology*, 25(2):163–177.

Coen Bron and Joep Kerbosch. 1973. Algorithm 457: finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577.

Ruirui Chen, Weifeng Jiang, Chengwei Qin, Ishaan Singh Rawal, Cheston Tan, Dongkyu Choi, Bo Xiong, and Bo Ai. 2024. Llm-based multi-hop question answering with knowledge graph integration in evolving environments. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 14438–14451.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Bradley Efron. 1987. Better bootstrap confidence intervals. *Journal of the American statistical Association*, 82(397):171–185.

Paul Erdős and Alfréd Rényi. 1959. On random graphs i. *Publ. math. debrecen*, 6(290-297):18.

Morten W Fagerland, Stian Lydersen, and Petter Laake. 2013. The mcnemar test for binary matched-pairs data: mid-p and asymptotic are better than exact conditional. *BMC medical research methodology*, 13(1):91.

Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. 2024. Talk like a graph: Encoding graphs for large language models. In *International conference on learning representations*, volume 2024, pages 43909–43934.

Jia He, Mukund Rungta, David Koleczek, Arshdeep Sekhon, Franklin X Wang, and Sadid Hasan. 2024. Does prompt formatting have any impact on llm performance? *arXiv preprint arXiv:2411.10541*.

Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. 1983. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137.

Sture Holm. 1979. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pages 65–70.

John E Hopcroft and Richard M Karp. 1973. An nˆ5/2 algorithm for maximum matchings in bipartite graphs. *SIAM Journal on computing*, 2(4):225–231.

Gregg Kellogg, Pierre-Antoine Champin, and Dave Longley. 2020. JSON-LD 1.1: A JSON-based serialization for linked data. W3C Recommendation.

Holger Knublauch and Dimitris Kontokostas. 2017. Shapes constraint language (SHACL). W3C Recommendation.

Linhao Luo, Zicheng Zhao, Gholamreza Haffari, Yuan-Fang Li, Chen Gong, and Shirui Pan. 2024a. Graph-constrained reasoning: Faithful reasoning on knowledge graphs with large language models. *arXiv preprint arXiv:2410.13080*.

Zihan Luo, Xiran Song, Hong Huang, Jianxun Lian, Chenhao Zhang, Jinqi Jiang, Xing Xie, and Hai Jin. 2024b. Graphinstruct: Empowering large language models with graph understanding and reasoning capability. *arXiv preprint arXiv:2403.04483*.

Shengjie Ma, Chengjin Xu, Xuhui Jiang, Muzhi Li, Huaren Qu, Cehao Yang, Jiaxin Mao, and Jian Guo. 2025. Think-on-graph 2.0: Deep and faithful large language model reasoning with knowledge-guided retrieval augmented generation. In *International Conference on Learning Representations*, volume 2025, pages 52782–52806.

Elan Markowitz, Krupa Galiya, Greg Ver Steeg, and Aram Galstyan. 2025. Kg-llm-bench: A scalable benchmark for evaluating llm reasoning on textualized knowledge graphs. *arXiv preprint arXiv:2504.07087*.

Lars-Peter Meyer, Johannes Frey, Desiree Heim, Felix Brei, Claus Stadler, Kurt Junghanns, and Michael Martin. 2025. Llm-kg-bench 3.0: A compass for semantic technology capabilities in the ocean of llms. In *European Semantic Web Conference*, pages 280–296. Springer.

Nandana Mihindukulasooriya, Sanju Tiwari, Carlos F Enguix, and Kusum Lata. 2023. Text2kgbench: A benchmark for ontology-driven knowledge graph generation from text. In *International semantic web conference*, pages 247–265. Springer.

OpenAI. 2025. OpenAI o3 and o4-mini System Card. System Card.

OpenAI. 2026a. GPT-5.4 Thinking System Card. OpenAI Deployment Safety Hub.

OpenAI. 2026b. Introducing GPT-5.4 mini and nano. Product Announcement.

Bryan Perozzi, Bahare Fatemi, Dustin Zelle, Anton Tsitsulin, Mehran Kazemi, Rami Al-Rfou, and Jonathan Halcrow. 2024. Let your graph do the talking: Encoding structured data for llms. *arXiv preprint arXiv:2402.05862*.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *International Conference on Learning Representations*, volume 2024, pages 25055–25083.

Stephen B Seidman. 1983. Network structure and minimum degree. *Social networks*, 5(3):269–287.

Yuan Sui, Mengyu Zhou, Mingjie Zhou, Shi Han, and Dongmei Zhang. 2024. Table meets llm: Can large language models understand structured table data? a benchmark and empirical study. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 645–654.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. 2024. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *International Conference on Learning Representations*, volume 2024, pages 3868–3898.

Jianheng Tang, Qifan Zhang, Yuhan Li, Nuo Chen, and Jia Li. 2025. Grapharena: Evaluating and exploring large language models on graph computation. In *International Conference on Learning Representations*, volume 2025, pages 48118–48145.

Etsuji Tomita, Akira Tanaka, and Haruhisa Takahashi. 2006. The worst-case time complexity for generating all maximal cliques and computational experiments. *Theoretical computer science*, 363(1):28–42.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. 2023. Can language models solve graph problems in natural language? *Advances in Neural Information Processing Systems*, 36:30840–30861.

Duncan J Watts and Steven H Strogatz. 1998. Collective dynamics of 'small-world'networks. *nature*, 393(6684):440–442.

xAI. 2025. Grok 4 Fast. `https://x.ai/news/grok-4-fast`.

Jin Y Yen. 1971. Finding the k shortest loopless paths in a network. *management Science*, 17(11):712–716.

Bowen Zhang and Harold Soh. 2024. Extract, define, canonicalize: An llm-based framework for knowledge graph construction. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pages 9820–9836.

Yigeng Zhou, Wu Li, Yifan Lu, Jing Li, Fangming Liu, Meishan Zhang, Yequan Wang, Daojing He, Honghai Liu, and Min Zhang. 2025. Reflection on knowledge graph for large language models reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 23840–23857.

A GraphQA

In this section we provide additional information on the GraphQA tasks, focusing on the extensions we implemented to make this task harder.

Task inventory and the three regimes. The three regimes differ along *task set*, *graph size*, and *graph generator* (Tab. 4).

GraphQA reproduces the 9 Fatemi–Perozzi tasks (Fatemi et al., 2024) and adds two structural summaries (`num_components`, `connectivity`) at $n \in \{8, 15, 25, 35, 50\}$ under $p \sim U[0, 1]$ Erdős–Rényi density.

GraphQA-Hardened layers 7 harder operators on top at $n \in \{35, 50, 75\}$ under a calibrated $p = \min(0.5,\ 2.5 \log_{10} n/n)$ so k-core, diameter, and max-clique stay non-trivial. The added operators cover: k-core peeling (Seidman, 1983), diameter via all-pairs reachability (Watts and Strogatz, 1998), k-shortest paths via Yen (Yen, 1971), max-clique via Bron–Kerbosch / Tomita pivot (Bron and Kerbosch, 1973; Tomita et al., 2006), top-betweenness (Brandes, 2001), local clustering, and high-degree-neighbour count.

GraphQA-Topology fixes $n = 50$ at the calibrated density, restricts to the 4 bounded-answer operators (`kth_shortest_path`, `local_clustering`, `neighbour_degree_ge`, `betweenness_top1`), and varies the generator over six families (Tab. 5). Reference examples are computed on the 5-node graph from App. H.

Topology generators. The six generators at $n = 50$ span 49–1,225 edges (Tab. 5). Erdős–Rényi (ER) and Barabási–Albert (BA) graphs differ by $\sim 15\%$ in mean edge count, so the BA-vs-ER contrast is at approximately matched density; stochastic block model (SBM) and complete confound topology with edge count, and star/path are degenerate by construction.

Full per-(model, format) accuracy. Tabs. 6–8 list the raw accuracy of every (model, format) cell on the three GraphQA regimes. Rows are models (full canonical names); columns are representation formats (header rotated 60°). Bold marks the per-row (per-model) maximum.

B Reasoning over Retrieved Wikidata Subgraphs

To assess whether representation sensitivity extends beyond synthetic GraphQA instances, we construct a controlled evaluation using Wikidata (Vrandečić and Krötzsch, 2014). We select 10 seed entities from heterogeneous domains, including films, scientists, cities, and companies. For each seed, we retrieve its directed two-hop neighborhood and retain at most 450 triples; the resulting subgraph serves as the canonical evaluation context. We replace entity labels with consistent anonymous identifiers while preserving predicate labels. This retains real-KG relational and topological structure while limiting reliance on memorized entity-specific knowledge.

The evaluation comprises 110 questions: 31 relation-specific aggregation questions, 31 type-aggregation questions, and 48 directed-reachability questions. As a targeted extension to the primary evaluation, we test all 15 serializers with two model configurations, `gpt-5.4` and `gpt-5.4-mini`. Gold answers are computed from the canonical subgraphs before rendering, and every serializer receives the same underlying subgraph for each question.

Table 9 reports the results. Both configurations rank `jsonld` first and `turtle` second. Across all 15 formats, the best-to-worst spread is 28.2 percentage points for `gpt-5.4` and 24.5 percentage points for `gpt-5.4-mini`. These results demonstrate representation sensitivity in this targeted two-model evaluation on heterogeneous Wikidata-derived subgraphs. Because each canonical subgraph is fixed before serialization, the experiment evaluates reasoning over retrieved graph contexts rather than retrieval quality or an end-to-end Graph-RAG pipeline.

C Robustness and Representation Properties

Repeated-run stability. We repeated all 15 formats for three independent decodes on fixed held-out sets of 105 GraphQA-Hardened and 120 GraphQA-Topology questions, using `gpt-5.4` and `gpt-5.4-mini`. All runs used temperature 0, and response caching was disabled so that repeated prompts were submitted independently. This gives $15 \times 3 \times (105 + 120) \times 2 = 20{,}250$ model calls. Across the four model–benchmark settings, the mean per-format run-to-run accuracy standard deviation is 1.5 percentage points. The between-format accuracy spread is 7.7× larger on average. Serving-side nondeterminism is therefore substantially smaller than the observed representation ef-

task	question template	ref. example	in
GraphQA (11 tasks; 9 adapted from Fatemi et al., 2024 + 2 added).			
node_count	"How many nodes are in the graph?"	5	G
edge_count	"How many edges are in the graph?"	5	G
node_degree	"What is the degree of node v?"	$\deg(3)=3$	G
edge_existence	"Is there an edge directly between a and b? Yes/No."	$\{0,4\}\rightarrow$ No	G
connectivity	"Is there a path from a to b? Yes/No."	$0\rightarrow 4$ Yes	G
connected_nodes	"List all direct neighbours of v, comma-separated."	$N(3)=1,2,4$	G
cycle_check	"Does the graph contain at least one cycle? Yes/No."	Yes	G
shortest_path	"Length of shortest $a\rightarrow b$ path? -1 if disconnected."	$d(0,4)=3$	G
triangle_counting	"How many unordered triangles?"	0	G
num_components	"How many connected components?" *(added)*	1	G
disconnected_nodes	"List every node *not* reachable from v, comma-separated."	none	G
GraphQA-Hardened (7 tasks; added on top of the GraphQA 11).			
kcore_size	"How many nodes belong to the 2-core?"	4	H
graph_diameter	"Within the largest component, what is the diameter?"	3	H
max_clique_size	"Size of the largest clique?"	2	H
kth_shortest_path	"Length of the second-shortest simple $a\rightarrow b$ path? -1 if none."	$0\rightarrow 4:-1$	H,T
betweenness_top1	"Which node has the HIGHEST betweenness centrality? Ties: smallest id."	3	H,T
local_clustering	"Local clustering coefficient of v? (2 d.p.)"	$C(3)=0.00$	H,T
neighbour_degree_ge	"How many neighbours of v have degree $\geq k$, where $k=\lceil \bar{d} \rceil$?"	$v=3, k=2:2$	H,T

Table 4: The 18 task templates (11 GraphQA + 7 Hardened). "in": **G** = GraphQA, **H** = Hardened, **T** = Topology. Topology keeps only the 4 Hardened operators with a bounded integer / 2-dp answer; reference examples use the 5-node graph of App. H.

generator	parameters	degree distribution	edge count at $n=50$
ER $G(n,p)$	$p=\min(0.5,\ 2.5\log_{10} n/n)\approx 0.085$	binomial, sharp peak	~ 85
BA	$m=2$ edges per new node	power-law $P(k)\sim k^{-3}$	≈ 98
SBM	2 communities, $p_{\text{in}}=0.55$, $p_{\text{out}}=0.03$	bimodal	~ 360
star	1 hub, $n-1$ leaves	$\{1,\ n-1\}$	49
complete	all-to-all	$\{n-1\}$	1,225
path	$0-1-\ldots-(n-1)$	$\{1,\ 2\}$	49

Table 5: GraphQA-Topology generators at $|V|=50$. Density-matched comparisons across all six are not meaningful; the load-bearing contrast (§4) is BA vs. ER at $\sim 15\%$ density mismatch.

model	adjacency	dot	edge_list	entity_centric	friendship	jsonld	markdown_table	mermaid	nl_summary	relation_indexed	shacl_lite	structured	text2kgbench	toon	turtle
gpt-5.4-high-reasoning	**100.0**	**100.0**	**100.0**	99.4	99.4	**100.0**	**100.0**	**100.0**	99.4	**100.0**	**100.0**	**100.0**	**100.0**	99.4	**100.0**
gpt-5.4	90.9	88.5	92.1	92.7	90.3	92.7	90.3	90.3	92.7	90.3	**93.3**	91.5	92.7	92.7	89.1
gpt-5.4-mini	75.1	72.1	73.3	77.6	71.5	**79.4**	75.1	73.9	76.4	75.8	76.4	75.1	72.7	71.5	70.9
o3	**98.2**	97.0	95.8	**98.2**	95.8	**98.2**	97.0	97.6	**98.2**	97.6	97.6	97.0	97.6	95.8	97.6
claude-sonnet-4-6-thinking	93.9	95.2	95.8	**97.0**	95.8	**97.0**	93.9	95.8	96.4	96.4	93.9	96.4	**97.0**	93.3	93.3
claude-sonnet-4-6	93.9	**94.5**	92.1	92.7	92.7	**94.5**	93.9	92.7	**94.5**	93.3	93.3	93.3	93.3	93.3	93.9
claude-haiku-4-5	86.7	86.7	86.7	88.5	84.9	90.3	86.1	85.5	87.3	84.9	**90.9**	87.9	86.7	84.9	83.0
grok-4-fast-reasoning	95.2	92.1	94.5	95.8	95.2	95.8	94.5	94.5	**97.0**	94.5	95.2	95.8	94.5	95.2	94.5
grok-4-fast-non-reasoning	**81.8**	66.7	66.7	77.6	67.9	80.0	69.1	69.1	**81.8**	67.9	79.4	66.7	66.7	64.8	65.5

Table 6: GraphQA per-format accuracy (%; ER, $|V|\leq 50$). Bold: per-model maximum. $n=165$ questions per (model, format).

fect.

Representation properties. The family labels in Tab. 1 are descriptive rather than causal. No individual characteristic determines performance across settings. Rankings vary by task, graph regime, and model configuration; a format that clearly exposes local neighborhoods, for example, need not

model	adjacency	dot	edge_list	entity_centric	friendship	jsonld	markdown_table	mermaid	nl_summary	relation_indexed	shacl_lite	structured	text2kgbench	toon	turtle
gpt-5.4	61.1	55.8	56.8	63.2	60.0	62.1	52.6	57.9	**64.2**	52.6	63.2	51.6	52.6	51.6	60.0
gpt-5.4-mini	37.9	35.8	32.6	35.8	39.0	**41.0**	34.7	34.7	40.0	33.7	40.0	34.7	33.7	33.7	34.7
o3	85.3	81.0	79.0	86.3	75.8	**87.4**	82.1	84.2	**87.4**	80.0	85.3	82.1	77.9	81.0	82.1
claude-sonnet-4-6-thinking	**83.2**	79.0	79.0	**83.2**	82.1	82.1	82.1	**83.2**	82.1	82.1	**83.2**	81.0	**83.2**	82.1	82.1
claude-sonnet-4-6	79.0	81.0	80.0	**87.4**	80.0	84.2	75.8	81.0	76.8	79.0	83.2	75.8	76.8	73.7	82.1
claude-haiku-4-5	**72.6**	53.7	52.6	71.6	54.7	67.4	50.5	48.4	68.4	44.2	66.3	61.1	54.7	59.0	46.3
grok-4-fast-reasoning	83.2	68.4	71.6	**84.2**	69.5	83.2	70.5	72.6	83.2	68.4	83.2	73.7	70.5	70.5	70.5
grok-4-fast-non-reasoning	42.1	28.4	27.4	35.8	23.2	35.8	26.3	24.2	34.7	26.3	**44.2**	30.5	23.2	26.3	28.4

Table 7: GraphQA-Hardened per-format accuracy (%; $|V| \in \{35, 50, 75\}$). Bold: per-model maximum. $n = 95$ questions per (model, format). gpt-5.4-high-reasoning excluded from this full-format comparison.

model	adjacency	dot	edge_list	entity_centric	friendship	jsonld	markdown_table	mermaid	nl_summary	relation_indexed	shacl_lite	structured	text2kgbench	toon	turtle
gpt-5.4	84.7	79.2	75.0	81.9	77.8	**90.3**	77.8	79.2	84.7	76.4	**90.3**	83.3	81.9	77.8	75.0
gpt-5.4-mini	48.6	50.0	52.8	48.6	50.0	**58.3**	54.2	54.2	47.2	52.8	41.7	55.6	52.8	48.6	50.0
o3	**90.3**	**90.3**	80.6	86.1	80.6	88.9	86.1	**90.3**	84.7	87.5	87.5	**90.3**	87.5	88.9	84.7
claude-sonnet-4-6-thinking	93.1	90.3	93.1	91.7	93.1	**95.8**	93.1	91.7	94.4	91.7	93.1	91.7	93.1	91.7	94.4
claude-sonnet-4-6	80.6	83.3	80.6	80.6	77.8	81.9	80.6	80.6	83.3	**84.7**	81.9	81.9	83.3	81.9	81.9
claude-haiku-4-5	77.8	66.7	66.7	79.2	72.2	75.0	73.6	68.1	**80.6**	65.3	79.2	68.1	73.6	65.3	63.9
grok-4-fast-reasoning	**95.8**	83.3	86.1	**95.8**	90.3	93.1	83.3	87.5	94.4	87.5	**95.8**	88.9	86.1	79.2	88.9
grok-4-fast-non-reasoning	61.1	55.6	**66.7**	62.5	63.9	58.3	52.8	62.5	**66.7**	62.5	65.3	59.7	59.7	58.3	62.5

Table 8: GraphQA-Topology per-format accuracy (%; $|V| = 50$, six generators). Bold: per-model maximum. $n = 72$ questions per (model, format). gpt-5.4-high-reasoning excluded from this full-format comparison.

model	best	acc.	second	acc.	worst	acc.	spread
gpt-5.4	jsonld	41.8	turtle	40.0	adj.	13.6	**28.2**
gpt-5.4-mini	jsonld	30.9	turtle	29.1	nl_sum.	6.4	**24.5**

Table 9: Tolerance accuracy (%) on 110 questions over retrieved Wikidata subgraphs. Spread is the within-model maximum minus minimum across all 15 formats.

perform best on a global graph operator. The targeted Wikidata evaluation provides a complementary real-KG case, with both evaluated models favoring the schema-explicit jsonld and turtle formats. We therefore interpret these regularities as associations between information organization and task demands, not as a causal decomposition of representation effects.

D Text2KGBench

In this section we show the comparison to the original baseline and detailed scores for every task, model, and representation.

Comparison with the original baselines. We rescore offline at the paper's threshold 0.60 from the on-disk pred_triples / gold_triples. The single methodological difference from Mihindukulasooriya et al. (2023) is the matcher: Hopcroft–Karp maximum-cardinality bipartite matching (Hopcroft and Karp, 1973) in place of their greedy. Hopcroft–Karp finds a globally optimal 1:1 matching, while greedy matching can produce suboptimal alignments due to early local choices.

Ontology size. The 19 WebNLG ontologies are small and fully connected, so context length is not a limiting factor for any frontier LLM (Tab. 12).

Per-domain heterogeneity. Per-domain F1 spans an order of magnitude across the 19 WebNLG ontologies on every model: head domains (comicscharacter, university, scientist, astronaut) saturate, while the tail (meanoftransportation, airport, monument, city) stays below 50 pp. On non-saturated domains, per-domain cross-format spread reaches up to 6.1 pp (building), exceeding the 3.9 pp Text2KGBench headline. The reported mean F1 therefore averages over an order-of-magnitude difficulty band; any prior-art comparison must hold the domain fixed.

model (year)	best fmt	F1	Δ vs. Vicuna 2023
Vicuna-13B, 2023	text2kgbench	30.0	—
Alpaca-LoRA-13B, 2023	text2kgbench	25.0	−5.0
gpt-5.4-high-reasoning	turtle	54.7	+24.7
gpt-5.4	mermaid	52.7	+22.7
gpt-5.4-mini	shacl_lite	50.6	+20.6
o3	shacl_lite	**54.9**	+24.9
claude-sonnet-4-6-thinking	jsonld	53.0	+23.0
claude-sonnet-4-6	jsonld	52.2	+22.2
claude-haiku-4-5	structured	50.7	+20.7
grok-4-fast-reasoning	shacl_lite	48.8	+18.8
grok-4-fast-non-reasoning	mermaid	49.9	+19.9

Table 10: Fuzzy F1 on DBPEDIA_WEBNLG at threshold 0.60 (Hopcroft–Karp). The 2023 baselines (Mihindukulasooriya et al., 2023, Tab. 2) use a greedy 1:1 matcher; the Δ column is context, not an encoding-choice effect.

model	edge_list	entity_centric	jsonld	markdown_table	mermaid	nl_summary	relation_indexed	shacl_lite	structured	text2kgbench	toon	turtle
gpt-5.4-high-reasoning	52.8	53.4	54.5	53.2	54.5	53.6	53.1	53.8	54.0	54.2	53.1	**54.7**
gpt-5.4	50.6	51.8	49.9	49.4	**52.7**	50.3	50.0	51.9	49.8	50.2	49.9	50.3
gpt-5.4-mini	48.7	50.2	48.1	49.3	49.7	49.6	49.0	**50.6**	49.2	49.9	49.8	48.0
o3	53.6	54.0	53.6	53.4	52.8	53.1	53.7	**54.9**	53.3	53.8	53.5	54.0
claude-sonnet-4-6-thinking	50.1	50.2	**52.9**	50.9	50.7	50.2	51.1	51.4	50.4	50.9	51.0	51.3
claude-sonnet-4-6	50.3	51.2	**52.2**	50.3	50.7	50.0	50.6	52.0	50.6	51.3	50.6	50.8
claude-haiku-4-5	49.9	48.9	46.8	49.4	47.9	48.9	49.9	48.5	**50.7**	49.5	49.0	48.6
grok-4-fast-reasoning	47.7	48.3	48.0	47.9	48.4	47.4	48.1	**48.8**	47.8	47.7	47.8	47.9
grok-4-fast-non-reasoning	49.5	49.7	48.7	49.2	**49.9**	49.5	49.2	49.4	49.6	49.4	49.4	48.5

Table 11: Text2KGBench per-format Fuzzy F1 (%) at threshold 0.60 (Hopcroft–Karp). Bold: per-model maximum. $n = 2{,}014$ questions per (model, format).

Statistic	min	median	mean	max
Concepts	8	14	14	20
Relations	18	38	36	68
Avg. rendered tokens	327	622	602	1,188

Table 12: Text2KGBench ontology size and rendering statistics across the 19 DBPEDIA_WEBNLG ontologies. Tokens averaged over all representation formats per ontology.

E Representation format selector

The per-question ceiling bounds *any* selector at up to +37.9 pp on the paired-BCa-significant slots (Tab. 13). We show that a representation format selector captures an appreciable fraction of that headroom. We construct a format selector model and evaluate end-to-end on GraphQA-Hardened using gpt-5.4 as *both* generator and selector so no privileged information leaks from a stronger model.

Calibration and test are two independently seeded 105-instance samples drawn from the same synthetic generator as GraphQA-Hardened (7 Hardened tasks ×15 ER graphs each at $n \in \{35, 50, 75\}$, calibrated density), with disjoint seed families so the calibration and test graph sets do not overlap. Calibration is not in-context learning: it is an offline per-(task, format) accuracy audit on the generator ($7 \times 15 \times 15 = 1{,}575$ generator calls). The calibration set is used to construct a per-(task, format) prior table including accuracy and sample count. At test time, the selector is given a task and all candidate formats in the set. It makes use of the prior table to include the mean accuracy and sample count in its context. No calibration question, graph, prediction, or reasoning trace ever appears in a test-time prompt.

Candidate set: 5 formats. For every test question, a set containing 5 candidate formats is created using the prior constructed from the calibration set: 2 SAFE succinct anchors (top-2 on the calibration cross-task ranking, restricted to compact formats), 1 ALT semantic anchor (top semantic-heavy format on the same ranking), and 2 GUEST formats from

model	regime	fixed	ceiling	Δ pp	sig
gpt-5.4	**GraphQA**	93.3	**95.8**	**+2.5**	**
gpt-5.4	**Hardened**	64.2	**80.0**	**+15.8**	**
gpt-5.4	**Topology**	90.3	**97.2**	**+6.9**	**
gpt-5.4-mini	**GraphQA**	79.4	**87.9**	**+8.5**	**
gpt-5.4-mini	**Hardened**	41.0	**70.5**	**+29.5**	**
gpt-5.4-mini	**Topology**	58.3	**80.6**	**+22.3**	**
o3	GraphQA	98.2	98.2	0.0	ns
o3	**Hardened**	87.4	**91.6**	**+4.2**	**
o3	Topology	90.3	93.1	+2.8	ns
claude-sonnet-4-6-thinking	GraphQA	97.0	98.8	+1.8	ns
claude-sonnet-4-6-thinking	**Hardened**	83.2	**87.4**	**+4.2**	**
claude-sonnet-4-6-thinking	Topology	95.8	97.2	+1.4	ns
claude-sonnet-4-6	GraphQA	94.5	96.4	+1.9	ns
claude-sonnet-4-6	Hardened	87.4	90.5	+3.1	ns
claude-sonnet-4-6	Topology	84.7	87.5	+2.8	ns
claude-haiku-4-5	**GraphQA**	90.9	**93.9**	**+3.0**	**
claude-haiku-4-5	**Hardened**	72.6	**83.2**	**+10.6**	**
claude-haiku-4-5	**Topology**	80.6	**90.3**	**+9.7**	**
grok-4-fast-reasoning	GraphQA	97.0	98.8	+1.8	ns
grok-4-fast-reasoning	Hardened	84.2	86.3	+2.1	ns
grok-4-fast-reasoning	Topology	95.8	95.8	0.0	ns
grok-4-fast-non-reasoning	**GraphQA**	81.8	**86.7**	**+4.9**	**
grok-4-fast-non-reasoning	**Hardened**	44.2	**82.1**	**+37.9**	**
grok-4-fast-non-reasoning	**Topology**	66.7	**90.3**	**+23.6**	**

Table 13: Oracle headroom per model (paired-BCa 95% CI; $n_{\text{boot}} = 2{,}000$). Bold $**$: lower CI bound > 0.

the calibration per-(task, format) ranking that carry the task-specific signal.

Selector: a fixed-prompt LLM scorer. Per test question the selector issues exactly 5 independent `gpt-5.4` chat completions, one per candidate, with no tool use, no loop, and no retrieval. Each call sees the question, the candidate's name, the candidate's calibration per-task baseline (e.g. "73% on `max_clique_size`"), and the test graph rendered in that candidate format. The system prompt enforces a fixed four-step chain: (1) TASK-NEEDS: one sentence on what the generator must compute; (2) RENDERING: one sentence on whether the format surfaces that substructure directly; (3) PRIOR-DELTA: a signed integer in $[-40, +40]$ shifting the baseline based on (2); (4) SCORE: a final integer 0–100. The highest-scoring candidate is chosen, with a small diversification band b that round-robins through ties within b points of the top to avoid collapsing to a single cross-task winner on every question. The chosen format is then handed to the generator, which runs the standard GraphQA chain-of-thought template (no selector context leaks into the generator call). Total LLM cost per question: 5 selector calls + 1 generator call.

Result. At band $b = 5$, the selector lifts mean accuracy by $+\mathbf{9.52}$ pp over fixed-best (`nl_summary`, the best fixed format on the test set, scores 66.67%); 95% paired-BCa CI $[+1.90, +16.19]$ pp; mid-p McNemar 0.013, $**$ (Tab. 14). This captures $\approx 59\%$ of the $+16.19$ pp slate-restricted headroom. Gains concentrate on the four non-saturated Hardened tasks (Tab. 15); a $b = 2$ vs. $b = 5$ ablation reproduces the qualitative result (Tab. 16).

F Statistical pipeline

Each headline contrast is reported with three pieces: paired bias-corrected-and-accelerated bootstrap (Efron, 1987) (2,000 resamples) over the per-question pairing; Holm–Bonferroni (Holm, 1979) family-wise correction at $\alpha = 0.05$; and a parallel two-sided mid-p McNemar test (Fagerland et al., 2013) on paired binary outcomes. The per-question ceiling is computed on the shared question set where every format has an answer, so ceiling $\geq$ fixed holds row-wise.

G Model Provenance and Inference Configuration

All experiments were run using the exact dated model identifiers reported in Table 17. The OpenAI configurations are pinned to `openai.gpt-5.4-2026-03-05`, `openai.gpt-5.4-mini-2026-03-17`, and `openai.o3-2025-04-16`. The default- and high-reasoning `gpt-5.4` configurations use the same

per-format mean accuracy ($n=105$)	%
entity_centric	68.57
nl_summary	66.67
jsonld / shacl_lite	65.71
adjacency	62.86
friendship	59.05
dot / mermaid	58.10
relation_indexed	57.14
markdown_table / structured / turtle	56.19
toon	54.29
text2kgbench	53.33
edge_list	49.52

selector vs. fixed-best ($b=5$)	value
selector accuracy	**76.19%**
fixed-best test set accuracy	66.67%
in-candidate post-hoc ceiling	82.86%
Δ selector $-$ fixed-best	**+9.52** pp
95% paired-BCa CI	[+1.90, +16.19]
mid-p McNemar	0.013
discordant pairs b/c	13/3
significance	**

Table 14: Selector headline ($n=105$ queries). **Left:** calibration-set per-format mean accuracy (the prior the selector uses at test time). **Right:** selector summary at $b=5$. Breakdowns in Tab. 15–16.

task	betweenness _top1	local _clustering	max_clique _size	neigh_deg _ge	graph _diameter	kcore _size	kth_shortest _path
fixed-best %	60.0	40.0	33.3	73.3	93.3	100.0	66.7
selector %	73.3	60.0	53.3	86.7	93.3	100.0	66.7
in-cand. ceiling %	86.7	73.3	73.3	86.7	93.3	100.0	66.7
Δ pp (sel. $-$ fxd.)	**+13.3**	**+20.0**	**+20.0**	**+13.3**	0.0	0.0	0.0
residual ceiling gap pp	+13.3	+13.3	+20.0	0.0	0.0	0.0	0.0

Table 15: Per-task selector vs. fixed-best (nl_summary) at $b=5$; $n=15$ questions per task. In-candidate ceiling: per-question best within the 5-format slate (upper bound on any slate-restricted selector). Gains concentrate on the four non-saturated tasks; no task regresses.

band b	selector acc.	Δ vs. nl_summary	95% paired-BCa CI	mid-p McNemar; b/c	sig.
$b=2$ (default)	75.24%	+8.57 pp	[+0.95, +15.24]	0.021; 12/3	**
$b=5$ (headline)	**76.19%**	**+9.52** pp	[+1.90, +16.19]	0.013; 13/3	**

Table 16: Diversification-band ablation on the same 105 queries. Both bands are paired-BCa significant; 16/105 picks flip between $b=2$ and $b=5$, the other 89 are decided by the LLM's argmax irrespective of b.

dated checkpoint and differ only in the requested reasoning-effort setting.

For the evaluated Anthropic and xAI models, requests used the provider-specific model identifiers shown in the table; dated checkpoint identifiers were not available. We therefore report their access windows alongside their requested identifiers. The original benchmark panel was evaluated during May 13–25, 2026. The retrieved-Wikidata and repeated-run additions used the same dated gpt-5.4 and gpt-5.4-mini identifiers during July 10–12, 2026.

H Representation format inventory and prompts

Below we list every representation format with its output on a small reference input (5-node graph for GraphQA, 3-concept / 3-relation ontology for Text2KGBench). GraphQA representation formats (§H.2) are applied to the reference graph; Text2KGBench representation formats (§H.3) are applied to the reference ontology.

H.1 Reference inputs

Reference graph (5 nodes, 5 edges, untyped): used for fact-first GraphQA representation formats.

```
G_FACT = {
  "nodes": [0, 1, 2, 3, 4],
  "edges": [(0,1), (0,2), (1,3),
            (2,3), (3,4)]}
```

Reference ontology (3 concepts, 3 relations): used for Text2KGBench representation formats.

```
ONTO = {
  "title": "PersonCompanyCity",
  "concepts": [
    {"qid":"Q1","label":"Person"},
    {"qid":"Q2","label":"Company"},
    {"qid":"Q3","label":"City"}],
  "relations": [
    {"label":"worksAt",
     "domain":"Person",
     "range":"Company"},
    {"label":"headquarteredIn",
     "domain":"Company",
     "range":"City"},
    {"label":"livesIn",
```

paper configuration	model identifier	inference control	temperature	original-panel access window
gpt-5.4	openai.gpt-5.4-2026-03-05	default reasoning	0	May 13–15, 2026
gpt-5.4-high	openai.gpt-5.4-2026-03-05	reasoning_effort=HIGH	0	May 13–23, 2026
gpt-5.4-mini	openai.gpt-5.4-mini-2026-03-17	default reasoning	0	May 13–15, 2026
o3	openai.o3-2025-04-16	default reasoning	0	May 13–19, 2026
claude-sonnet-4-6	anthropic.claude-sonnet-4-6	thinking off	0	May 18–25, 2026
claude-sonnet-4-6-thinking	anthropic.claude-sonnet-4-6	extended thinking	1	May 18–25, 2026
claude-haiku-4-5	anthropic.claude-haiku-4-5	thinking off	0	May 18–25, 2026
grok-4-fast-reasoning	xai.grok-4-fast-reasoning	reasoning endpoint	0	May 13–16, 2026
grok-4-fast-non-reasoning	xai.grok-4-fast-non-reasoning	non-reasoning endpoint	0	May 13–15, 2026

Table 17: Model identifiers, inference controls, and access windows for the original benchmark panel. `gpt-5.4-high` is the same dated checkpoint as `gpt-5.4`, with HIGH reasoning effort enabled.

```
    "domain":"Person",
    "range":"City"}]}
```

H.2 All 15 GraphQA representation formats

Output of every representation format on the 5-node reference graph above.

adjacency.

```
Graph (adjacency list):
  0: [1, 2]
  1: [0, 3]
  2: [0, 3]
  3: [1, 2, 4]
  4: [3]
```

edge_list.

```
Graph (undirected).
Nodes: 0, 1, 2, 3, 4
Edges:
  0 -- 1
  0 -- 2
  1 -- 3
  2 -- 3
  3 -- 4
```

mermaid.

```
graph TD
    N0
    N1
    N2
    N3
    N4
    N0 --- N1
    N0 --- N2
    N1 --- N3
    N2 --- N3
    N3 --- N4
```

markdown_table.

```
**Nodes:** 0, 1, 2, 3, 4

**Edges**

| u | v |
|---|---|
| 0 | 1 |
| 0 | 2 |
| 1 | 3 |
| 2 | 3 |
| 3 | 4 |
```

toon.

```
Nodes[id]:
  0, 1, 2, 3, 4
Edges[u|v]:
  0 | 1
  0 | 2
  1 | 3
  2 | 3
  3 | 4
```

friendship (Fatemi et al., 2024).

```
There are these people: Person0, Person1,
Person2, Person3, Person4.
Person0 is a friend of Person1.
Person0 is a friend of Person2.
Person1 is a friend of Person3.
Person2 is a friend of Person3.
Person3 is a friend of Person4.
```

turtle (Beckett et al., 2014).

```
@prefix : <http://ex.org/> .
:n0 a :Node . :n1 a :Node . :n2 a :Node .
:n3 a :Node . :n4 a :Node .
:n0 :linkedTo :n1 . :n0 :linkedTo :n2 .
:n1 :linkedTo :n3 . :n2 :linkedTo :n3 .
:n3 :linkedTo :n4 .
```

jsonld (Kellogg et al., 2020).

```
{"@context":{"@vocab":"http://ex.org/",
  "adjacent":{"@type":"@id"}},
 "@graph":[
  {"@id":"n0","adjacent":["n1","n2"]},
  {"@id":"n1","adjacent":["n0","n3"]},
  {"@id":"n2","adjacent":["n0","n3"]},
  {"@id":"n3",
   "adjacent":["n1","n2","n4"]},
  {"@id":"n4","adjacent":["n3"]}]}
```

structured. Our shorthand for a compact set-builder form.

```
V = {0, 1, 2, 3, 4}
E = {(0,1), (0,2), (1,3), (2,3), (3,4)}
```

nl_summary.

```
The graph has vertices labeled
  0, 1, 2, 3, 4.
Vertex 0 is connected to 1, 2.
Vertex 1 is connected to 0, 3.
Vertex 2 is connected to 0, 3.
Vertex 3 is connected to 1, 2, 4.
Vertex 4 is connected to 3.
```

relation_indexed.

```
Nodes: 0, 1, 2, 3, 4

linkedTo:
  0 -> 1
  0 -> 2
  1 -> 3
  2 -> 3
  3 -> 4
```

dot. Graphviz DOT.

```
graph G {
    0;
    1;
    2;
    3;
    4;
    0 -- 1;
    0 -- 2;
    1 -- 3;
    2 -- 3;
    3 -- 4;
}
```

text2kgbench (the original Text2KGBench serialization (Mihindukulasooriya et al., 2023), applied to a graph rather than an ontology).

```
Graph Vertices: 0, 1, 2, 3, 4
Graph Edges: edge(0,1), edge(0,2),
  edge(1,3), edge(2,3), edge(3,4)
```

entity_centric. Per-vertex outgoing-edge view.

```
## Vertex 0
  --[edge]--> 1
  --[edge]--> 2
## Vertex 1
  --[edge]--> 0
  --[edge]--> 3
## Vertex 2
  --[edge]--> 0
  --[edge]--> 3
## Vertex 3
  --[edge]--> 1
  --[edge]--> 2
  --[edge]--> 4
## Vertex 4
  --[edge]--> 3
```

shacl_lite. A SHACL-lite shape-constraint view on the graph (no W3C profile; our shorthand for the `sh:minCount`/`sh:maxCount` fragment).

```
# Shape constraints (graph-as-SHACL view)
shape(0) {
  path adjacent -> node(1) ;
  path adjacent -> node(2) ;
}
shape(1) {
  path adjacent -> node(0) ;
  path adjacent -> node(3) ;
}
shape(2) {
  path adjacent -> node(0) ;
  path adjacent -> node(3) ;
}
shape(3) {
  path adjacent -> node(1) ;
  path adjacent -> node(2) ;
  path adjacent -> node(4) ;
}
shape(4) {
  path adjacent -> node(3) ;
}
```

H.3 All 12 Text2KGBench representation formats

Output of every representation format on the 3-concept / 3-relation reference ontology above.

edge_list.

```
CONCEPTS:
Person, Company, City

RELATIONS (domain --[rel]--> range):
Person --[worksAt]--> Company
Company --[headquarteredIn]--> City
Person --[livesIn]--> City
```

entity_centric.

```
## Person
  --[worksAt]--> Company
  --[livesIn]--> City
## Company
  --[headquarteredIn]--> City
## City
  (no outgoing relations)
```

jsonld (Kellogg et al., 2020).

```
{"@context":{
   "ont":"http://example.org/onto#",
   "owl":"http://www.w3.org/2002/07/owl#",
   "rdfs":
 "http://www.w3.org/2000/01/rdf-schema#"},
 "@graph":[
  {"@id":"ont:Person",
   "@type":"owl:Class",
   "rdfs:label":"Person"},
  {"@id":"ont:Company",
   "@type":"owl:Class",
   "rdfs:label":"Company"},
  {"@id":"ont:City",
   "@type":"owl:Class",
   "rdfs:label":"City"},
  {"@id":"ont:worksAt",
   "@type":"owl:ObjectProperty",
   "rdfs:label":"worksAt",
   "rdfs:domain":"ont:Person",
   "rdfs:range":"ont:Company"},
  {"@id":"ont:headquarteredIn",
   "@type":"owl:ObjectProperty",
   "rdfs:label":"headquarteredIn",
   "rdfs:domain":"ont:Company",
   "rdfs:range":"ont:City"},
  {"@id":"ont:livesIn",
   "@type":"owl:ObjectProperty",
   "rdfs:label":"livesIn",
   "rdfs:domain":"ont:Person",
   "rdfs:range":"ont:City"}]}
```

markdown_table.

```
**Concepts**

| # | Label |
|---|-------|
| 1 | Person |
| 2 | Company |
| 3 | City |
```

```
**Relations (typed)**

| Relation | Domain | Range |
|---|---|---|
| worksAt | Person | Company |
| headquarteredIn | Company | City |
| livesIn | Person | City |
```

mermaid.

```
classDiagram
    class Person
    class Company
    class City
    Person --> Company : worksAt
    Company --> City : headquarteredIn
    Person --> City : livesIn
```

nl_summary.

```
This ontology defines the entity types:
  Person, Company, City.
The following typed relations are defined:
 - A Person can be related via
   "worksAt" to a Company.
 - A Company can be related via
   "headquarteredIn" to a City.
 - A Person can be related via
   "livesIn" to a City.
```

relation_indexed.

```
CONCEPTS:
Person, Company, City

RELATIONS (relation -> domain --> range):
  worksAt: Person --> Company
  headquarteredIn: Company --> City
  livesIn: Person --> City
```

shacl_lite (Knublauch and Kontokostas, 2017).

```
# Schema shapes (constraint view)
shape(Person) {
  path worksAt -> class(Company) ;
  path livesIn -> class(City) ;
}
shape(Company) {
  path headquarteredIn -> class(City) ;
}
shape(City) {
  // no outgoing typed paths
}
```

structured. Header-labeled bulleted list.

```
Concepts (Entity Types):
  - Person
  - Company
  - City

Relations (with domain and range):
  - worksAt
      (domain: Person, range: Company)
  - headquarteredIn
      (domain: Company, range: City)
  - livesIn
      (domain: Person, range: City)
```

text2kgbench (compact form used in the original Text2KGBench paper; Mihindukulasooriya et al., 2023).

```
Ontology Concepts: Person, Company, City
Ontology Relations:
  worksAt(Person,Company),
  headquarteredIn(Company,City),
  livesIn(Person,City)
```

toon. TOON-style header-then-pipe columns.

```
Concepts[label]:
  - Person
  - Company
  - City

Relations[label|domain|range]:
  worksAt | Person | Company
  headquarteredIn | Company | City
  livesIn | Person | City
```

turtle (Beckett et al., 2014).

```
@prefix : <http://example.org/onto#> .
@prefix owl:
  <http://www.w3.org/2002/07/owl#> .
@prefix rdfs:
 <http://www.w3.org/2000/01/rdf-schema#> .

:Person a owl:Class .
:Company a owl:Class .
:City a owl:Class .

:worksAt a owl:ObjectProperty ;
  rdfs:domain :Person ;
  rdfs:range :Company .
:headquarteredIn a owl:ObjectProperty ;
  rdfs:domain :Company ;
  rdfs:range :City .
:livesIn a owl:ObjectProperty ;
  rdfs:domain :Person ;
  rdfs:range :City .
```

H.4 Per-task system/user prompt template

```
[SYSTEM]
You are a careful reasoner.  Read the
graph (or ontology), think through the
question step by step, then write your
final answer on a line that begins
with 'FINAL: '.

[USER, GraphQA - task=node_count]
You are given an undirected graph
below.  Answer the question in one
integer.  Show your reasoning, then
write 'FINAL: <int>'.

<rendered graph>

Question: How many nodes does the
graph have?

[USER, Text2KGBench]
You are given an ontology and a
sentence.  Extract every (subject,
relation, object) triple supported by
the sentence, constrained to the
relations in the ontology.  Output one
triple per line in the format
'(<subject>, <relation>, <object>)'.

<rendered ontology>

Sentence: <sentence>
```