Quantifying Retrieval Quality in GraphRAG: A Schema-Agnostic Approach

EURA NOVA

Presented by

Thibaud Vanmechelen
AI Engineer

Alexandre Achten
AI Engineer

Zaineb Gabsi
R&D Engineer

Sabri Skhiri
CTO

Presentation Overview

01 **Problem Statement**

02 **Methodology**

03 **Experimental Results**

04 **Key Insights**

05 **Conclusion & Future Work**

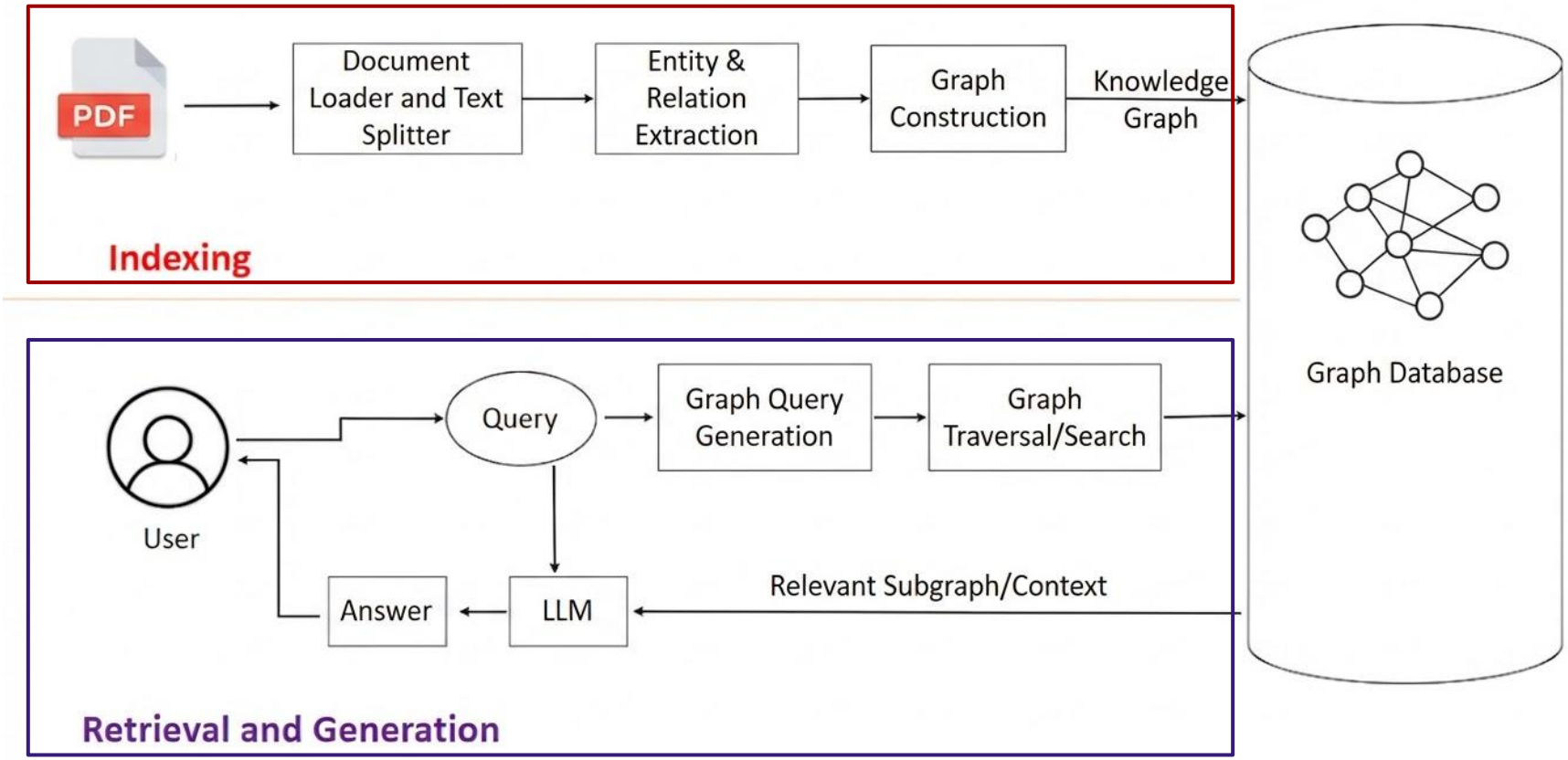
RESEARCH PROBLEM

Retrieval-Evaluation Gap in GraphRAG Architectures

Standard RAG Falls Short: relies on semantic similarity, ignoring structural relationships

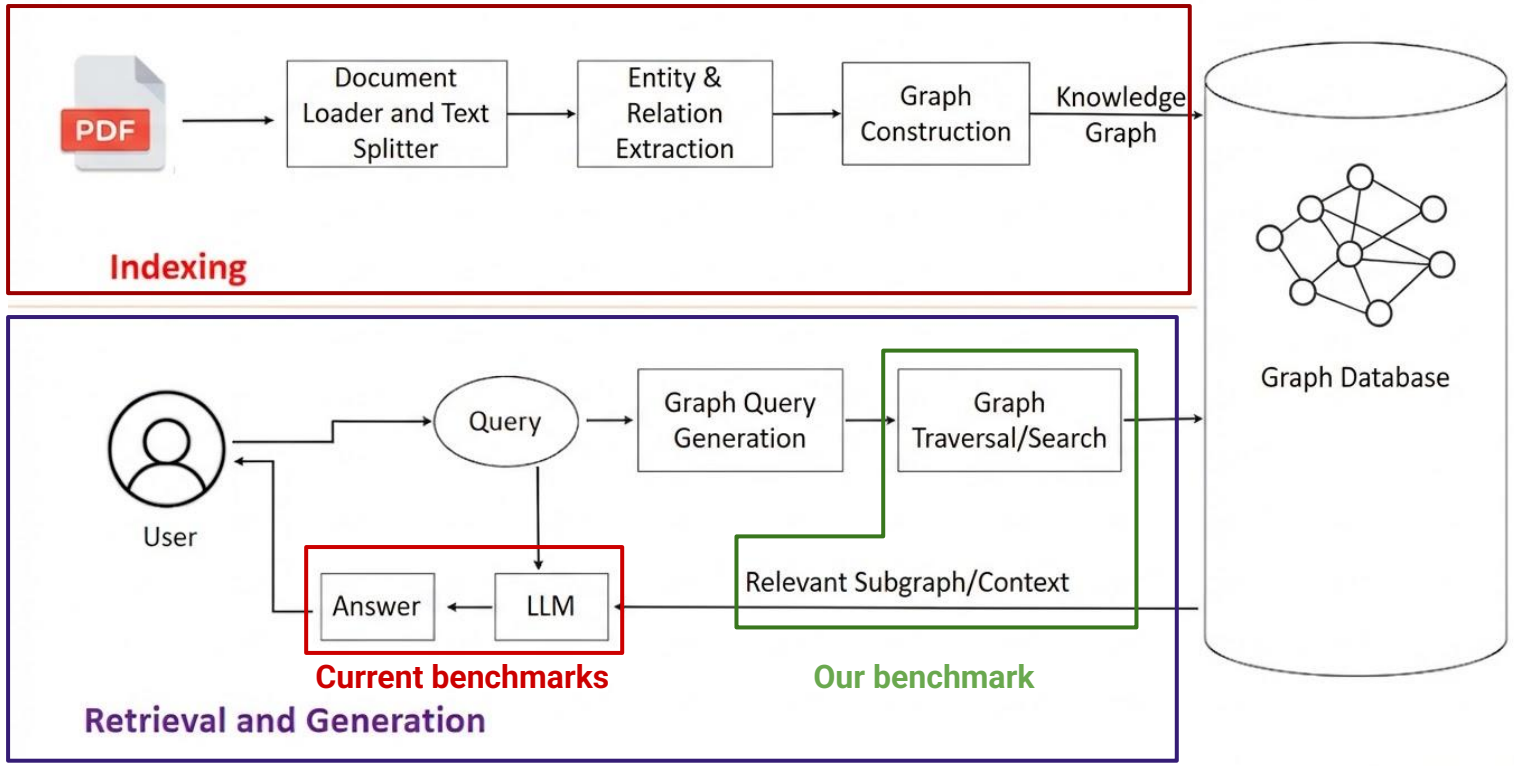
vs.

GraphRAG: explicitly models data relationships to enable path-traversal reasoning (for complex queries)



The Critical Gap

Current benchmarks ignore internal retrieval accuracy, and focus only on final LLM responses.



Garbage in, Garbage out: LLM output quality is strictly capped by the completeness of the retrieved context

RESEARCH GAP

What is Missing in Current Approaches

1 Diagnostic Conflation

Most benchmarks use metrics that focus only on final answers

> Problem: Cannot distinguish between retrieval failure and generation failure

2 Topological Blindness

Current benchmarks (RAGAS, ARES, …) focus on semantic similarity over structural logic

> Problem: Cannot check if the model followed the correct data links/paths

3 Ground Truth Incompleteness

Benchmarks reward “partial” matches

> Problem: Fails to measure if the entire necessary graph structure was retrieved

4 Lack of Structural Ground Truth

No rigorous “gold standard” for the retrieval stage

> Problem: Impossible to calculate Precision/Recall on the graph itself

Presentation Overview

01 **Problem Statement**

02 **Methodology**

03 **Experimental Results**

04 **Key Insights**

05 **Conclusion & Future Work**

OUR CONTRIBUTION

Schema-Agnostic Framework for Evaluating Retrieval Quality in GraphRAG

Novel Approach

Automated synthetic dataset generation with deterministic ground truth

Schema-Agnostic Design

Uses generalized Cypher templates (works on any graph database)

Deterministic Ground Truth

Uses direct Cypher queries to obtain ground truth (no approximations)

Performance Tracking

Pinpoints specific retriever weaknesses by category

METHODOLOGY

Dataset Generation Pipeline

1 Taxonomy & Templates

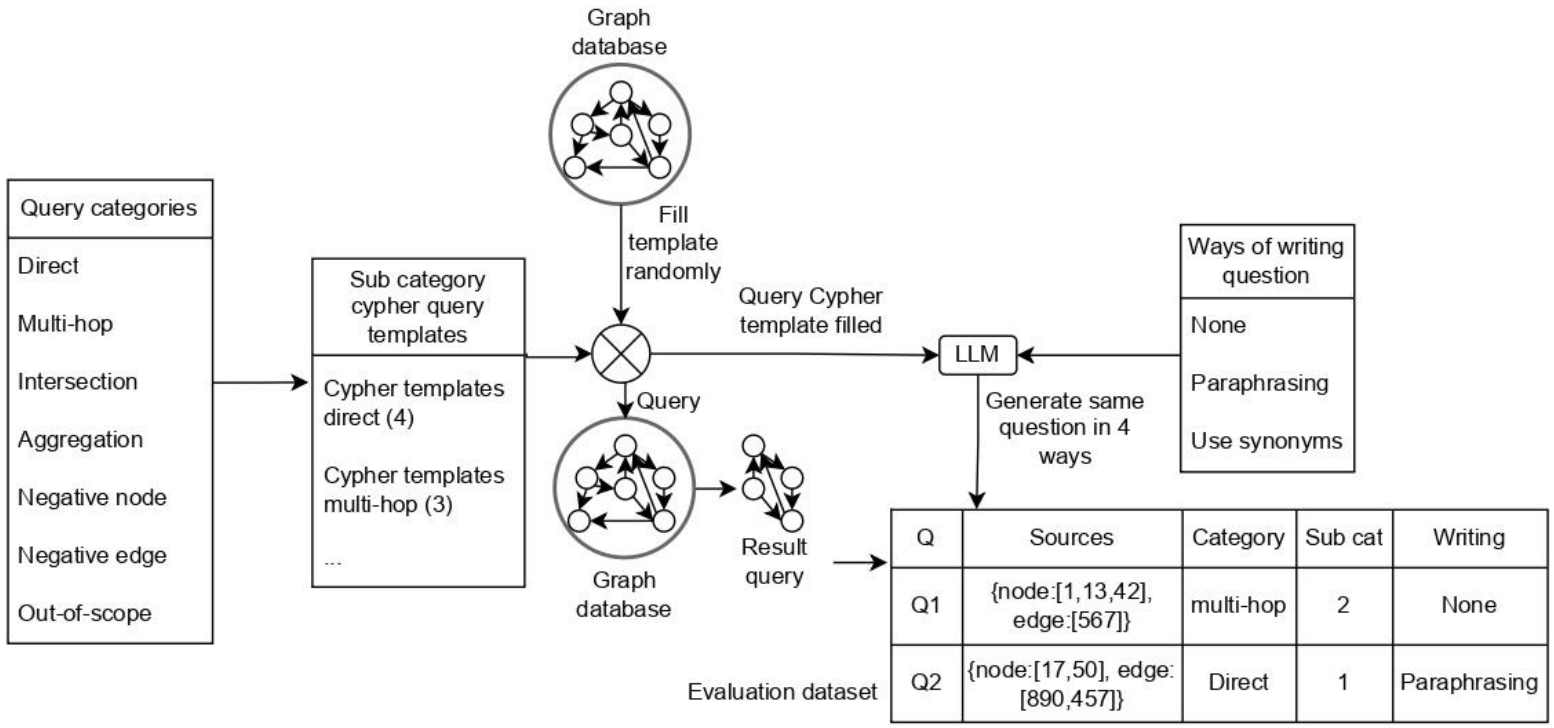
Establish 9 query categories with generalized Cypher templates targeting nodes or edges.

2 Query Population

Populate templates with random data from KG. Execute queries to ensure exhaustive ground truth.

3 Question Generation

Translate Cypher to natural language question using an LLM with varied writing styles



METHODOLOGY

Cypher Template Examples

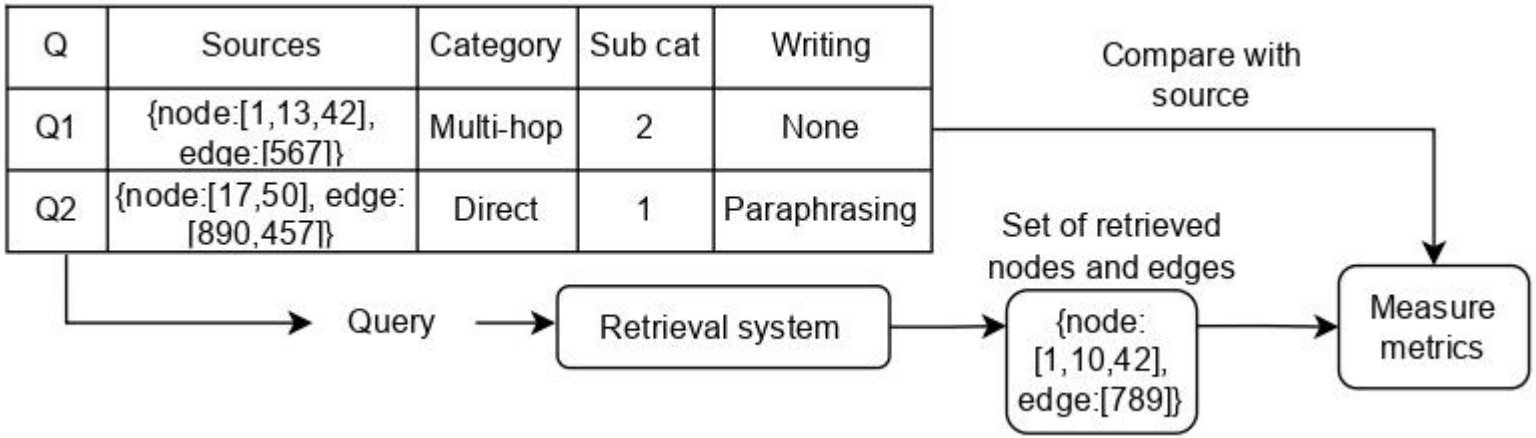
```
aggregation-edge:
  1:
    template: "MATCH (n1{
        node_type1})-[r{rel_type
        }]->(n2{node_type2}) WHERE
         ALL(key IN keys($params)
        WHERE n1[key] = $params[
        key]) RETURN COUNT(r) AS
        count"
    type: "known-source-count"
```

```
direct-edge-retrieval:
  1:
    template: "MATCH (n1{
        node_type1})-[r{rel_type
        }]->(n2{node_type2}) WHERE
         ALL(key IN keys($params)
        WHERE n1[key] = $params[
        key]) RETURN r"
    type: "known-source-matching"
```

Methodology & Metrics

Evaluation Workflow

1. Retrieval system processes each natural language query
2. System retrieves element identifiers (nodes, edges)
3. Retrieved elements compared against deterministic ground truth
4. Metrics computed per category, template, and writing style



EVALUATION

Methodology & Metrics

Standard Metrics

Precision

$$P = \frac{|G_{retrieved} \cap G_{expected}|}{|G_{retrieved}|}$$

Recall

$$R = \frac{|G_{retrieved} \cap G_{expected}|}{|G_{expected}|}$$

F1-Score

$$F_1 = \frac{2 \times P \times R}{P + R}$$

Topological Metrics

$$\text{Path Hit Rate}(G, P) = \begin{cases} 1 & \text{if } G \subseteq P \\ 0 & \text{if } G \not\subseteq P \end{cases}$$

$$\text{ROUGE-L} = \frac{2 \cdot P_{lcs} \cdot R_{lcs}}{P_{lcs} + R_{lcs}}$$

$$R_{lcs} = \frac{LCS(G, P)}{m}$$

$$P_{lcs} = \frac{LCS(G, P)}{n}$$

Efficiency Metrics (values to minimize)

Retrieval Time	API Calls
Seconds	Count

Aggregation Metrics

Proxy Precision:

$$P_{proxy} = \min\left(\frac{C_{expected}}{C_{retrieved}}, 1.0\right)$$

Proxy Recall:

$$R_{proxy} = \min\left(\frac{C_{retrieved}}{C_{expected}}, 1.0\right)$$

RETRIEVAL STRATEGIES

Four Architectures Evaluated

Random Baseline

Lower-bound reference

Selects k random nodes/edges from graph schema. Sets the minimum performance threshold.

Graph Traversal

Text2Cypher approach

LLM translates natural language queries into executable, schema-aware Cypher statements.

Semantic Agent

LLM-driven with semantic seeding

Uses semantic search to find entry points, then crawls through neighbors

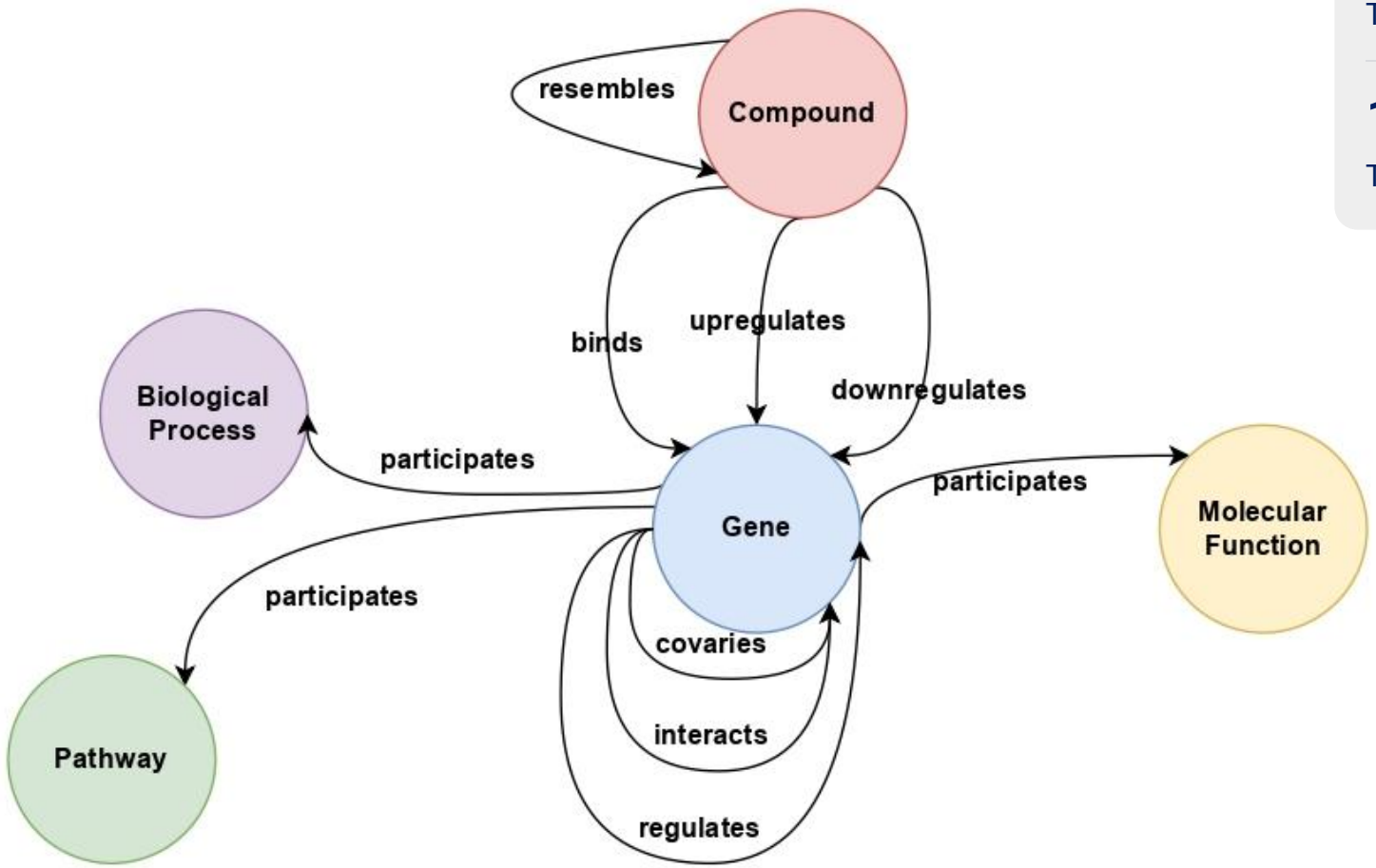
CoT Agent

Chain-of-Thought extension

Extends Semantic Agent with broader toolset: counting neighbors, finding common connections, specific node/edge retrieval.

DATASET

Hetionet Subset



Key Metrics

38,584 Total Nodes	82% of full Hetionet
1,488,879 Total Edges	66% of full Hetionet

A Rich Resource for Biomedical Experimentation derived from Hetionet https://het.io/

Presentation Overview

01 **Problem Statement**

02 **Methodology**

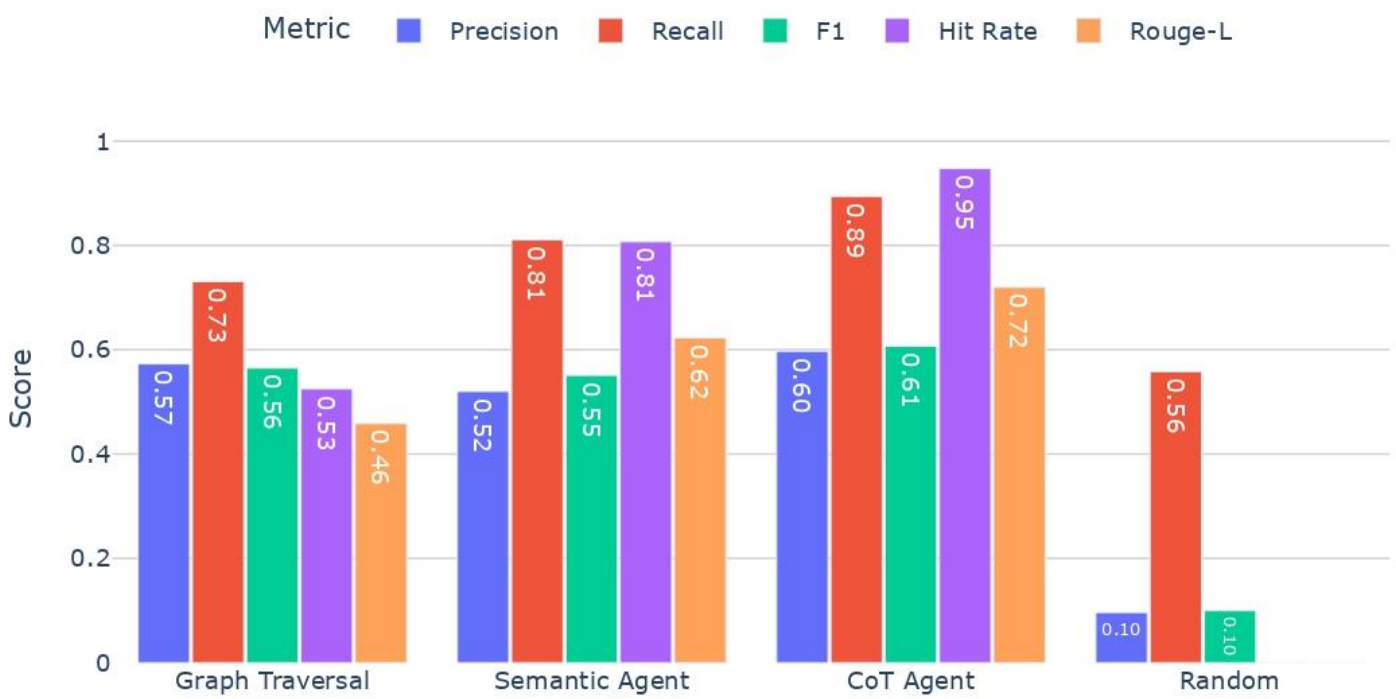
03 **Experimental Results**

04 **Key Insights**

05 **Conclusion & Future Work**

RESULTS

Performance Results



Key Findings

CoT Agent

Highest recall (0.89) and Hit Rate (0.95)

Semantic Agent

Recall (0.81) and Hit Rate (0.81)

Graph Traversal

Higher precision than **Semantic Agent** but lower recall

Random Baseline

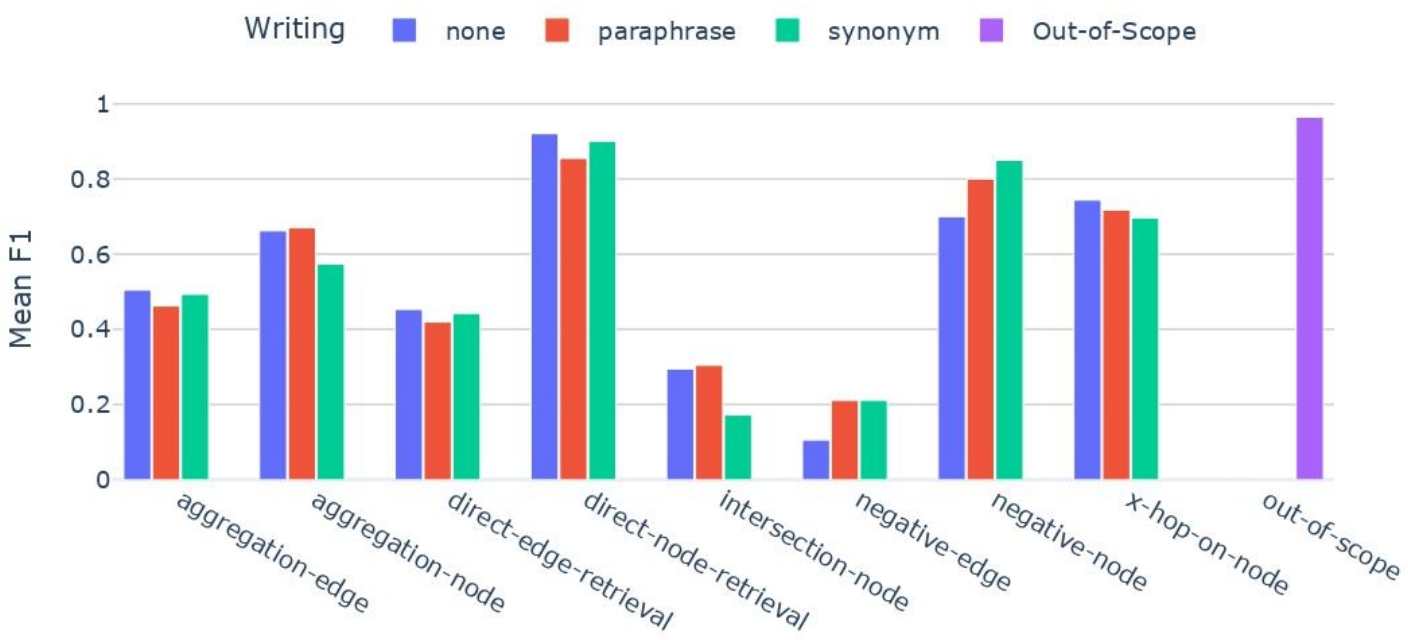
Worst in every metric

"Over-Inclusion" Effect

Recall better suited, because LLM can filter relevant context during answer generation

RESULTS

Category-Level Analysis: CoT Retriever



CoT Agent Strengths

Out-of-Scope	**0.96**
Direct Node	**0.92**
Negative Node	**0.85**
Multi-Hop	**0.74**

Challenging Categories

Intersection-node (0.3) and negative-edge (0.21) show lowest performance.

KG contains hundreds of millions of intersections → difficult to navigate.

Writing Style Impact

Minimal effect. F1 scores consistent across "None", "Paraphrasing", and "Synonyms" styles.

Resource Efficiency: Time & API Costs

Execution Time

Retriever	Time (s)
Random	2.683
GraphTraversal	4.256
Semantic Agent	6.893
CoT Agent	7.560

API Calls per Query

Retriever	API Calls
Random	0
GraphTraversal	1.078
Semantic Agent	2.096
CoT Agent	4.018

Presentation Overview

01 **Problem Statement**

02 **Methodology**

03 **Experimental Results**

04 **Key Insights**

05 **Conclusion & Future Work**

SUMMARY

Key Insights & Implications

1. Agentic Retrievers Excel

LLM-driven agentic retrievers offer the highest recall and reasoning, effectively navigating complex topologies.

- CoT Agent's dynamic tool selection enables sophisticated multi-hop reasoning

2. The "Over-Inclusion"

Strict retrieval benchmarks may over-penalize "over-inclusion", retrieving technically accurate but non-target elements that could actually enhance final answer quality.

- Recommendation: Hybrid approach balancing deterministic metrics with LLM-as-judge

3. Category Dominates Performance

Query category is the principal performance factor, not writing style. F1 scores vary drastically (0.9 to 0.1) across categories but remain consistent across phrasing variations.

- Validates framework's focus on structural reasoning over semantic variation

Presentation Overview

01 **Problem Statement**

02 **Methodology**

03 **Experimental Results**

04 **Key Insights**

05 **Conclusion & Future Work**

CONCLUSION

Contributions & Future Directions

Primary Contributions

- Novel methodology for automated schema-agnostic dataset generation from KGs
- Rigorous deterministic ground truth establishment through query execution
- Comprehensive 9-category taxonomy covering fundamental graph operations
- Empirical evidence of agentic retriever superiority for complex topologies

Impact

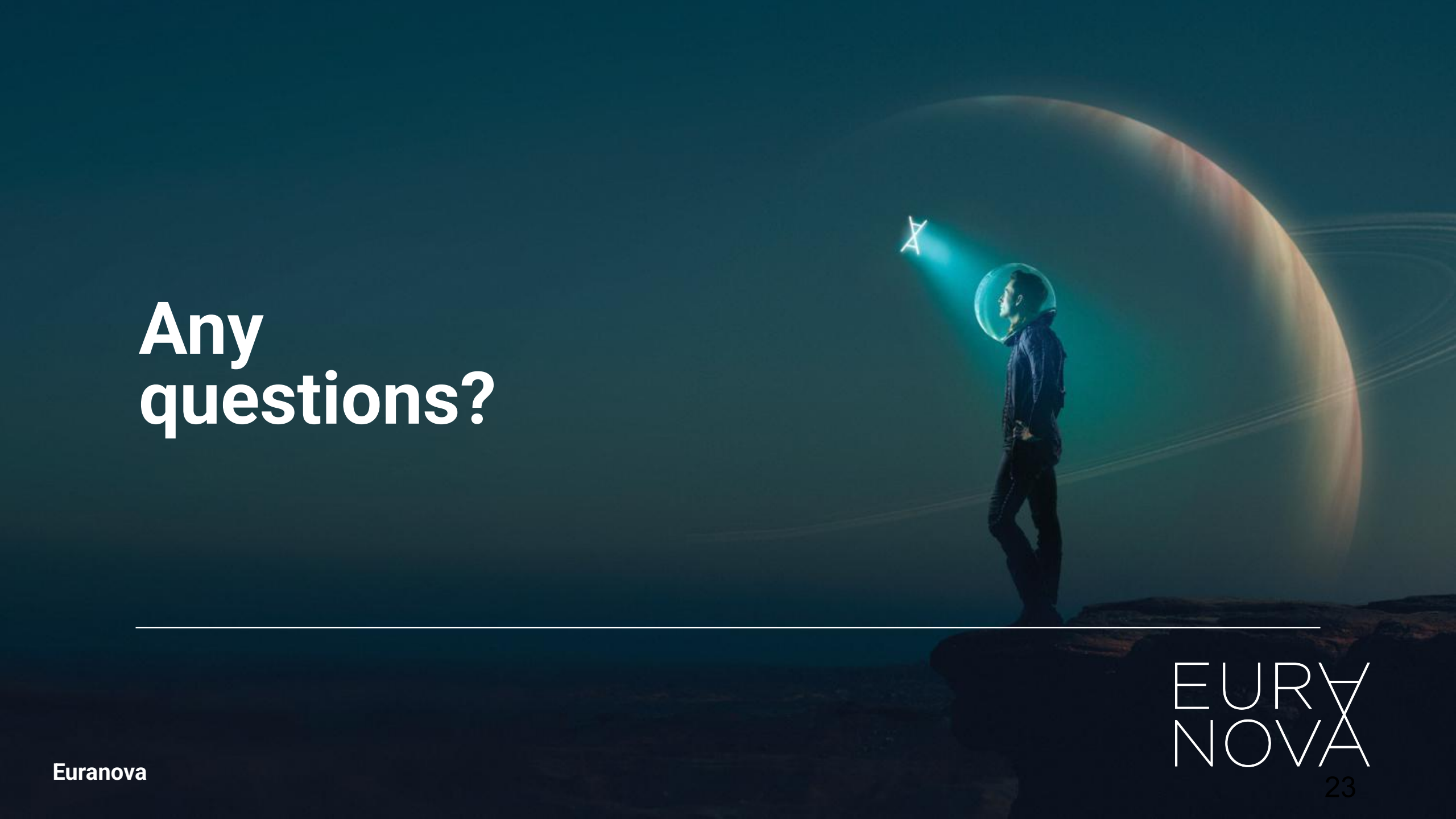
This work enables systematic comparison of GraphRAG retrieval strategies, providing researchers and practitioners with quantitative benchmarks for architectural decisions

Future Work

Cross-Domain Validation
Apply methodology to diverse domain KGs to validate generalisability

Hybrid Evaluation
Combine strict metrics with LLM-as-judge for complete quality spectrum

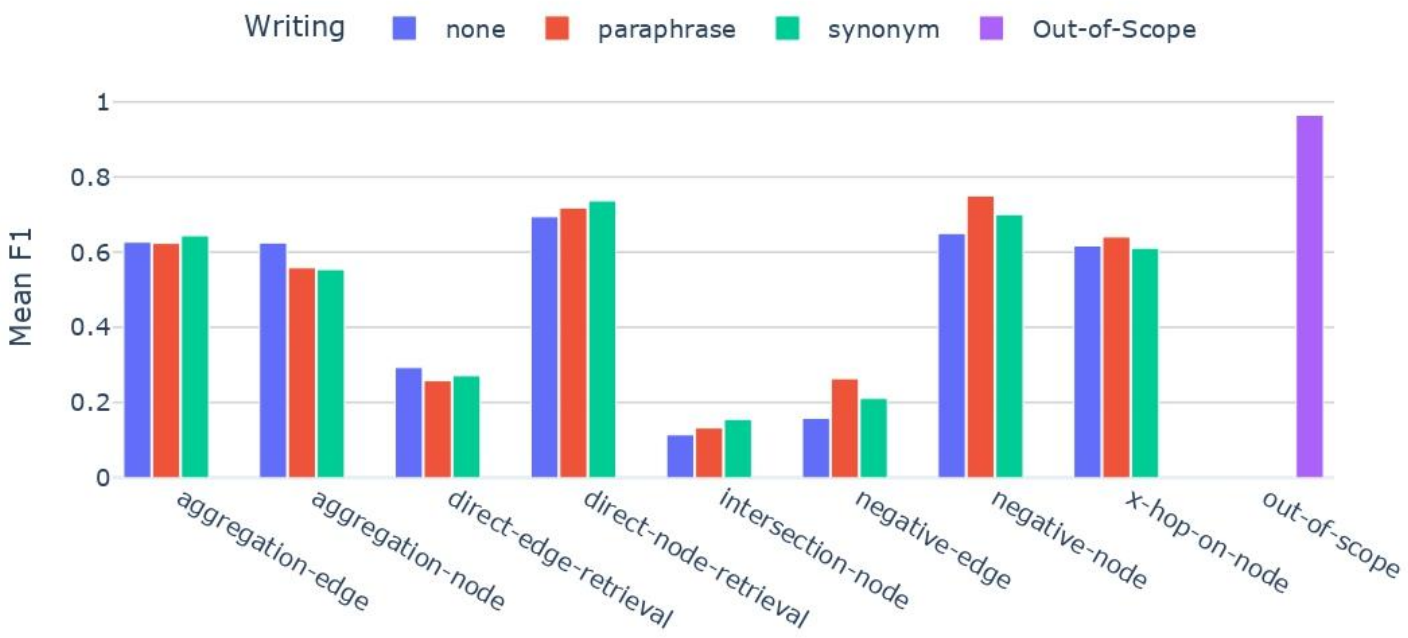
Any questions?



Appendix

RESULTS

Category-Level Analysis: Semantic Retriever



Semantic First Agent Strengths

Out-of-Scope	**0.96**
Direct Node	**0.73**
Negative Node	**0.75**
Multi-Hop	**0.64**

Challenging Categories

Intersection-node (0.15), direct-edge (0.29), and negative-edge (0.26) show lowest performance.

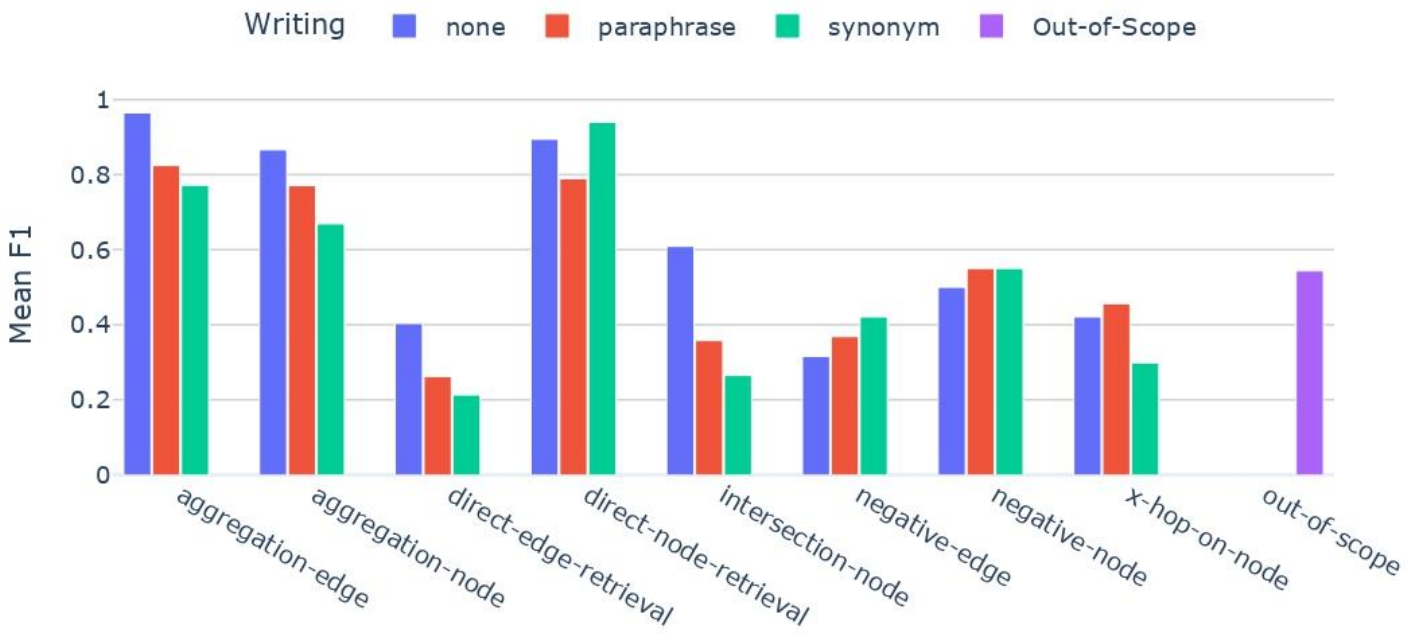
Performs similarly to COT Retriever but underperforms in some categories because it is not given an extensive list of tools

Writing Style Impact

Minimal effect. F1 scores consistent across "None", "Paraphrasing", and "Synonyms" styles.

RESULTS

Category-Level Analysis: GraphTraversal Retriever



GraphTraversal Strengths

Aggregation Edge	**0.96**
Direct Node	**0.94**
Aggregation Node	**0.87**
Intersection Node	**0.61**

Challenging Categories

Direct-edge (0.4), and negative-edge (0.42) show lowest performance.

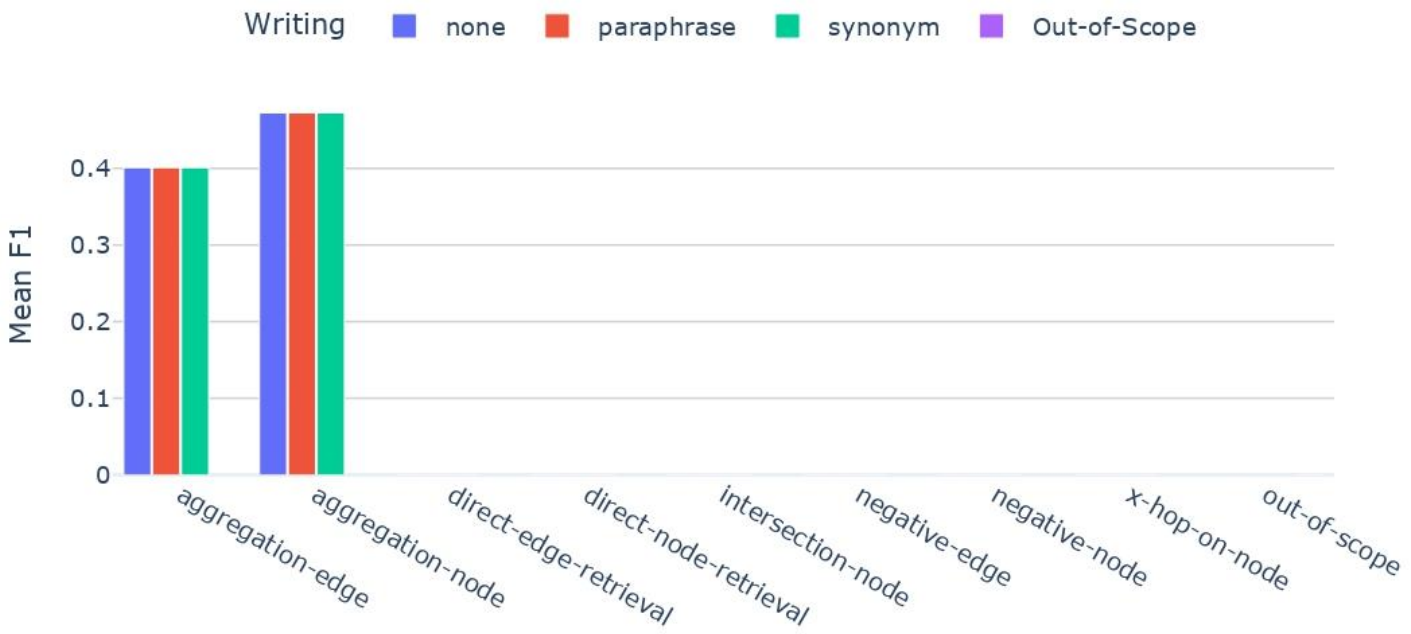
Performs badly in the negative edges and negative nodes because the process times out.

Writing Style Impact

Minimal effect (but more noticeable). F1 scores consistent across "None", "Paraphrasing", and "Synonyms" styles.

RESULTS

Category-Level Analysis: Random Retriever



Random Baseline Strengths

Aggregation Node	**0.47**
Aggregation Edge	**0.4**

The non zero numbers are resulting from the proxy metrics that gives credit to the number of retrieved items even if they are random

Writing Style Impact

No Effect