International Journal of Scientific Research and Engineering Development-– Volume 9 Issue 3, May-June 2026
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

KnowPro: A Confidence-Aware Knowledge Graph Construction and Graph-RAG Retrieval Framework for Unstructured Scientific Text

Swapnil Kale*, Sanchit Joshi**, Aaradhya Kulkarni***, Vardhan Bhanuwanshe****,prof. Varsha Kulkarni*****

*(Department of Computer Engineering, R.H. Sapat College of Engineering, Nashik, India) swapnilkale226@gmail.com
**(Department of Computer Engineering, R.H. Sapat College of Engineering, Nashik, India) sanchitjoshi0644@gmail.com
***(Department of Computer Engineering, R.H. Sapat College of Engineering, Nashik, India) kulkarni4a@gmail.com
****(Department of Computer Engineering, R.H. Sapat College of Engineering, Nashik, India) vardhanbhanuwanshe04@gmail.com
*****(Department of Computer Engineering, R.H. Sapat College of Engineering, Nashik, India) varsha.kulkarni@ges-coengg.org

Abstract:

The rapid growth of unstructured scientific literature presents significant challenges for automated knowledge acquisition and semantic querying. Existing pipelines frequently suffer from unreliable extraction, lack of traceability, and hallucination-prone retrieval. This paper proposes KnowPro, a unified, modular framework that integrates structure-aware document ingestion, hybrid symbolic-neural knowledge extraction, confidence-gated dual-layer storage, provenance-aware knowledge graph construction, and a strategy-based Graph Retrieval-Augmented Generation (Graph-RAG) architecture. The hybrid extraction engine combines rule-based Open Information Extraction patterns with a pre-trained SciBERT BIO token classifier, merging outputs through a deduplication-and-max-confidence mechanism. A configurable three-tier routing system partitions extracted triples by confidence into a high-fidelity Neo4j reasoning graph and a fully auditable PostgreSQL triple store. The Graph-RAG retrieval layer implements five deterministic traversal strategies—targeted, chained, variable-hop, shortest-path, and shared-neighbor—with n-gram and Levenshtein-based entity resolution and strict prompt-level grounding. Internal validation on a medical knowledge domain demonstrates end-to-end pipeline correctness across 105 automated tests and four representative end-to-end traces. In this version, KnowPro is presented as an extensible architecture for trustworthy knowledge graph construction and structured semantic retrieval from scientific corpora, with several evaluation items still scoped as implementation validation rather than benchmark-level comparison.

Keywords — Knowledge Graph Construction, Graph-RAG, Open Information Extraction, SciBERT, Confidence Routing, Provenance Tracking, Neo4j, Hybrid Extraction

I. INTRODUCTION

The exponential growth of scientific literature has made automated knowledge extraction and structured reasoning increasingly critical for downstream applications in biomedical informatics, legal analytics, and technical intelligence. Unstructured text remains the dominant medium through which scientific knowledge is communicated, yet its transformation into queryable, structured representations continues to face fundamental challenges: extraction noise, semantic ambiguity, lack of evidence traceability, and hallucination in retrieval-augmented generation systems.

Traditional information extraction pipelines are constrained by one of two failure modes: rule-based systems achieve high precision but exhibit poor recall on complex or contextually ambiguous expressions, while neural extraction models offer broader coverage but are susceptible to low-confidence outputs and opaque decision boundaries. Furthermore, conventional knowledge graph construction frameworks often discard provenance metadata, limiting their capacity for auditable and trustworthy semantic reasoning.

Recent advances in Graph Retrieval-Augmented Generation (Graph-RAG) have demonstrated that structured graph retrieval substantially outperforms vector similarity-based retrieval for multi-hop and relational queries. However, existing Graph-RAG implementations lack systematic traversal taxonomies, often rely on natural language to Cypher translation without reliability guarantees, and provide limited mechanisms for hallucination suppression.

This paper addresses these limitations by proposing KnowPro—a unified, end-to-end framework for constructing confidence-aware, provenance-enriched knowledge graphs from unstructured scientific text, and enabling reliable semantic retrieval through a strategy-based Graph-RAG architecture. The principal contributions of this work are:

- A structure-aware document ingestion pipeline employing layout-sensitive parsing, coreference resolution, and hierarchical metadata preservation.
- A hybrid extraction engine that executes rule-based Open IE patterns and SciBERT BIO classification in parallel,

unifying outputs through deduplication and max-confidence fusion.

- A dual-layer knowledge storage architecture that separates a high-confidence Neo4j reasoning graph from a fully auditable PostgreSQL provenance store, gated by a configurable confidence threshold.
- A five-strategy Graph-RAG retrieval layer featuring deterministic traversal selection, multi-resolution entity resolution, Cypher-based execution, and strict prompt-level hallucination grounding.
- A proposed schema-agnostic retrieval tier enabling dynamic graph querying without pre-defined ontological constraints; this tier is presented as future-facing rather than part of the validated core.

The remainder of this paper is organized as follows. Section II reviews related work. Section III describes the system architecture. Sections IV–VIII detail each pipeline stage. Section IX presents experimental validation. Section X discusses limitations and future directions. Section XII concludes the paper.

II. RELATED WORK

A. Open Information Extraction

Open Information Extraction (Open IE) was formalized by Banko et al. with the TextRunner system, establishing the paradigm of unsupervised, domain-independent triple extraction [1]. Subsequent systems including ReVerb [2], OLLIE [3], and ClausIE [4] introduced dependency-parse-driven extraction patterns that significantly improved syntactic coverage. The Hearst IS-A pattern lexicon [5] established a foundational lexical approach for taxonomic relation extraction. PredPatt [6] extended this to universal dependency parses. KnowPro's rule engine draws on patterns from ReVerb, OLLIE, OpenIE, PredPatt, and Hearst, operating directly on spaCy dependency graphs.

B. Neural Relation Extraction

Transformer-based models have demonstrated substantial improvements in information extraction tasks. SciBERT [7], pre-trained on a scientific corpus (Semantic Scholar), provides domain-adapted representations well-suited to scientific text. The LSOIE dataset [8] provides large-scale sentence-level Open IE annotations enabling BIO-sequence token classification for structured triple extraction. CaRB [9] provides a cross-domain benchmark for evaluating Open IE recall and precision. KnowPro employs a pre-trained SciBERT checkpoint fine-tuned on LSOIE, achieving F1 of 0.735 on the LSOIE test set and 0.427 on CaRB, consistent with reported baselines.

C. Confidence Scoring and Knowledge Routing

Confidence calibration methods in neural networks, formalized through Expected Calibration Error (ECE) and temperature scaling [10], motivate the design of reliable confidence handling in extraction pipelines. In this version of KnowPro, however, the confidence value is produced by heuristic scoring and extractor fusion rather than by a separate post-hoc calibration layer. Work on human-in-the-loop knowledge base construction [12] demonstrates that routing uncertain predictions to human validators rather than discarding them improves long-term knowledge quality. The proposed framework therefore operationalizes a three-tier routing mechanism differentiating high-confidence auto-insertion, medium-confidence human validation, and low-confidence archiving.

D. Graph-Based Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) established the paradigm of conditioning language model generation on retrieved context [13]. Knowledge graph-enhanced RAG systems, such as G-Retriever [14] and GraphRAG [15], have demonstrated that structured relational context improves multi-hop reasoning and reduces hallucination relative to vector similarity retrieval. However, existing systems frequently rely on natural language to Cypher query translation, introducing reliability risks. KnowPro eliminates this dependency through deterministic, config-driven traversal strategy selection operating directly over a property graph.

III. SYSTEM ARCHITECTURE OVERVIEW

The KnowPro framework is organized as a seven-stage sequential pipeline in which each module produces a well-defined output that constitutes the input to the subsequent stage. The overall architecture is illustrated in Fig. 1.

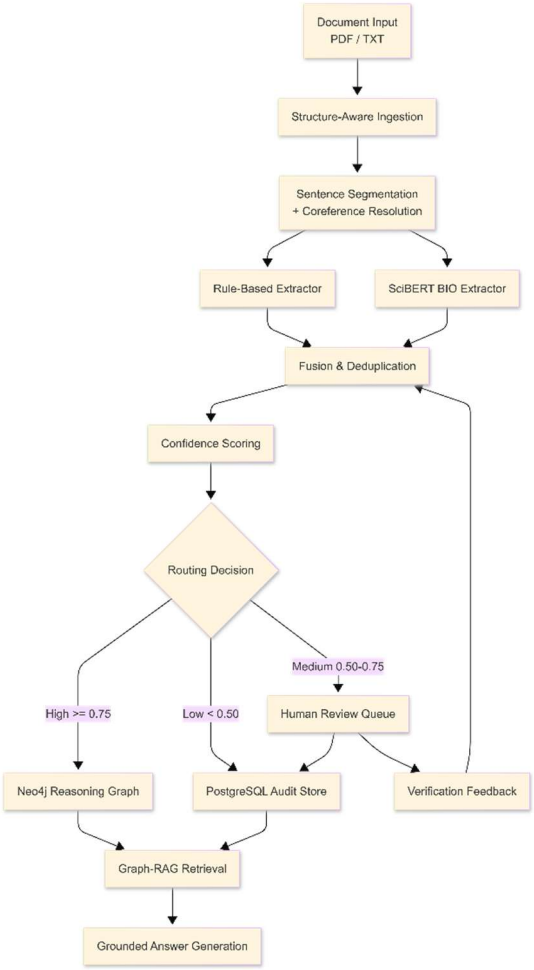

Fig. 1. KnowPro end-to-end pipeline architecture showing the seven processing stages and the asynchronous human-in-the-loop validation module.

The pipeline comprises: (1) Structure-Aware Document Ingestion, (2) Hybrid Knowledge Extraction, (3) Confidence Scoring and Routing, (4) Confidence-Gated Dual-Layer Storage, (5) Knowledge Graph Construction, (6) Graph-RAG Retrieval, and (7) Grounded Answer Generation. A human-in-the-loop validation module operates asynchronously on medium-confidence triples.

A key architectural principle of KnowPro is the strict separation of concerns between the provenance layer (PostgreSQL) and the reasoning layer (Neo4j). All extracted triples, regardless of confidence, are persisted in PostgreSQL with complete provenance metadata. Only triples meeting a configurable confidence threshold are projected into the Neo4j graph database for graph-based retrieval. This dual-layer design ensures full auditability while maintaining a high-fidelity reasoning graph.

IV. STRUCTURE-AWARE DOCUMENT INGESTION

A. Input and Parsing

KnowPro accepts input in PDF and plain-text (TXT) formats. PDF parsing is performed using a dual-tool approach: `PyMuPDF4LLM` for semantic text extraction and `PyMuPDF` layout analysis for spatial document structure recovery. OCR fallback is handled via the PyMuPDF-embedded Tesseract integration, invoked when text extraction quality falls below an acceptable threshold.

The parser performs spatial block reconstruction by aggregating spans and lines based on x-coordinate proximity into logical text blocks. Block classification is subsequently performed using a hierarchical rule system that evaluates: (1) relative font size with respect to document-average font size to distinguish headings from body text; (2) font style attributes (bold, italic) as indicators of structural elements; (3) textual patterns matching known section delimiters (e.g., "Abstract", "Fig. N"); and (4) contextual position relative to preceding classified blocks. Each resulting block is assigned a block type from the set {`title`, `abstract`, `heading`, `body`, `caption`, `footer`}.

B. Preprocessing and Coreference Resolution

Extracted text undergoes a deterministic cleaning pipeline comprising: reference section removal via regex-based citation pattern matching, normalization to lowercase, email address anonymization, citation number suppression, and hyphenation artifact repair. Sentence boundary detection is performed using the `spaCy` English pipeline.

Coreference resolution is applied document-wide using the `fastcoref` library, producing a resolved version of each sentence in which anaphoric references are substituted with their antecedent entities. Both the original and coreference-resolved sentences are retained in storage.

C. Structured Storage Schema

Processed documents are persisted in a PostgreSQL JSONB schema with three levels of granularity. At the document level, a `doc_id` is assigned as a compound identifier of the filename and ingestion timestamp. At the block level, each block is assigned a serial `block_id` and its inferred `block_type`. At the sentence level, each sentence receives a `sentence_id` encoding its page number and per-page serial index. Sentence records store the original text, coreference-resolved text, block membership, page membership, and sequential order. Raw pre-cleaning text is also retained to support full provenance reconstruction.

V. HYBRID KNOWLEDGE EXTRACTION ENGINE

A. Parallel Extraction Architecture

The extraction engine operates on coreference-resolved sentences retrieved from the PostgreSQL store. Extraction is performed by two independent subsystems executing in parallel: a rule-based dependency extractor and a transformer-based BIO sequence classifier. Both subsystems produce outputs conforming to a unified `ExtractedTriple` schema comprising fields: `subject`, `relation`, `object`, `confidence`, `rule_name`, `source_sentence`, `subject_type`, and `object_type`.

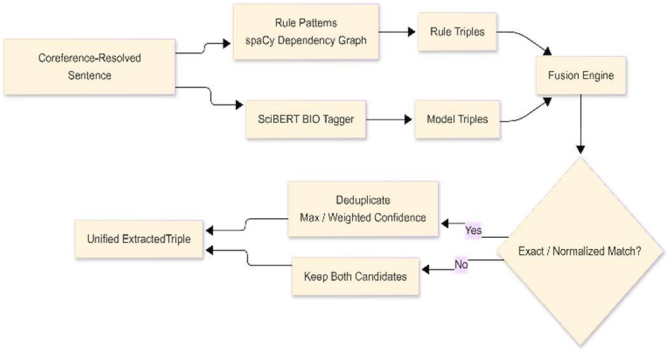

Fig. 2. Parallel hybrid extraction architecture showing the rule-based and transformer-based subsystems and the fusion/deduplication step.

B. Rule-Based Extraction

The rule-based subsystem applies 14 registered extraction patterns to spaCy-parsed dependency graphs. Patterns are derived from established Open IE frameworks including ReVerb [2], OLLIE [3], PredPatt [6], and the Hearst pattern lexicon [5]. Lemmatization is employed selectively: verb lemmas are normalized to enable pattern matching across morphological variants (e.g., matching {LEMMA: 'be'} against is, was, were), while entity spans are extracted in surface form to preserve nominal fidelity. Table I presents a representative subset of the registered rule patterns and their associated confidence weights.

TABLE I
RULE-BASED EXTRACTION PATTERNS (REPRESENTATIVE SUBSET)

Rule ID	Pattern Type	Confidence
reverb_svo	Subject-Verb-Object (ReVerb)	0.70
ollie_copula	Copula (X is Y) — OLLIE	0.75
ollie_appos.	Appositive relation — OLLIE	0.80
ollie_poss.	Possessive — OLLIE	0.70
hearst_such_as	Hearst IS-A (such as)	0.90
reverb_pass.	Passive construction	0.70
openie_prep	Prepositional — OpenIE	0.45
predpatt_relcl	Relative clause — PredPatt	0.65

Rule confidence scores are fixed heuristic weights reflecting pattern-level precision estimates derived empirically from Open IE literature benchmarks. For instance, the reverb_svo pattern carries a confidence of 0.70 consistent with reported ReVerb precision on general-domain text [2].

C. Transformer-Based Extraction

The transformer subsystem employs a pre-trained SciBERT model checkpoint fine-tuned for BIO sequence labeling on the LSOIE dataset [8]. The model assigns BIO labels from the set {O, B-SUBJ, I-SUBJ, B-REL, I-REL, B-OBJ, I-OBJ} to each subword token. Contiguous spans sharing the same label prefix are aggregated into subject, relation, and object spans via BIO reconstruction. The model was evaluated on the LSOIE test partition achieving $F_1 = 0.735$ and on the CaRB benchmark [9] achieving $F_1 = 0.427$, establishing its generalization boundary.

Triple-level confidence is computed as the mean of per-span softmax probabilities across subject, relation, and object token sequences. Formally, for a triple T with spans S, R, O of lengths n_s, n_r, n_o respectively:

$$C_{\text{model}}(T) = \frac{1}{n_s + n_r + n_o} \sum_i p_i \quad \text{...(1)}$$

where p_i is the maximum-class softmax probability for token i across all labeled spans.

D. Fusion and Deduplication

Outputs from both extraction subsystems are merged into a unified candidate list. Deduplication is performed via normalized triple keys defined as the tuple of lowercased (`subject`, `relation`, `object`) strings. When multiple extractions yield identical normalized keys, the single extraction with the highest confidence score is retained. When extractions with the same subject and object but differing relation strings co-occur, both are preserved to avoid semantic conflict erasure. The final confidence for a retained triple is:

$$C_{\text{final}}(T) = \max(C_{\text{rule}}(T),\ C_{\text{model}}(T)) \quad \text{...(2)}$$

VI. CONFIDENCE-GATED DUAL-LAYER STORAGE

A. Confidence Routing

Each extracted triple is evaluated against a configurable confidence threshold (default $\theta = 0.75$) to determine its storage routing. The system implements a three-tier routing mechanism as summarized in Table II.

TABLE II
THREE-TIER CONFIDENCE ROUTING MECHANISM

Level	Range	Action	Storage
High	≥ 0.75	Auto-insert	Neo4j + PostgreSQL
Medium	0.45–0.74	Human Review	PostgreSQL (pending)
Low	< 0.45	Archive/Reject	PostgreSQL (audit)

B. PostgreSQL Provenance Store

All extracted triples, irrespective of confidence tier, are inserted into a PostgreSQL relational store as the primary

ground truth repository. Each triple record encapsulates: subject and object entity strings with NER-derived type annotations, relation string, computed confidence, routing tier assignment, a JSONB provenance field, and graph synchronization status.

The provenance JSONB field captures: the originating sentence text, `sentence_id` and `doc_id` references, character-level offsets within the source document, the set of extractor classes that produced the triple, specific rule identifiers, and ISO-8601 extraction timestamps. This design enables complete reconstruction of the evidence chain for any triple in the knowledge graph, satisfying auditability and explainability requirements.

C. Neo4j Reasoning Graph

High-confidence triples (confidence $\geq \theta$) are projected incrementally into a Neo4j property graph database. Entities are represented as typed nodes with `name` and `name_lower` (lowercase-normalized) properties. Relations are represented as directed typed edges carrying `confidence`, `source_text`, and temporal metadata. The `MERGE` operation is employed for both node and edge creation, ensuring idempotency under repeated ingestion. When an existing edge is encountered, the confidence value is updated as the maximum of the stored and incoming values. This incremental merge strategy eliminates graph rebuild overhead and preserves confidence evolution history.

VII. KNOWLEDGE GRAPH CONSTRUCTION

A. Graph Schema

The KnowPro knowledge graph employs a property graph model in which node labels correspond to NER-derived entity types (e.g., `PROTEIN`, `DISEASE`, `PERSON`, `CONCEPT`) and relationship types correspond to extracted predicate strings (e.g., `INHIBITS`, `TREATS`, `IS_A`). Entity normalization is performed through case-folding and whitespace normalization on the `name_lower` field, which serves as the `MERGE` key. This ensures that surface form variants differing only in capitalization are resolved to a single node while preserving the original surface form in the `name` property.

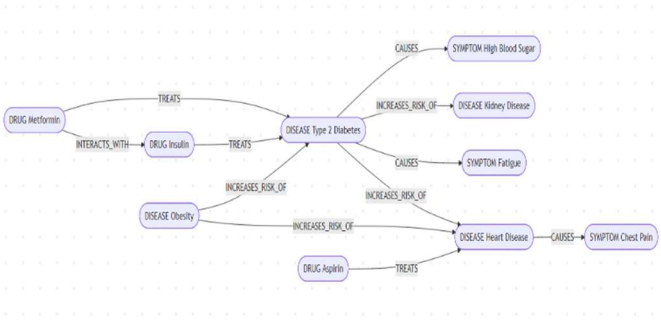

Fig. 3. Sample subgraph from the medical knowledge domain showing typed entity nodes and confidence-annotated directed relation edges.

B. Provenance Linkage

Each graph edge retains a lightweight provenance reference through its `source_text` property. Full provenance reconstruction is achieved by joining on the normalized (`subject, relation, object`) key against the PostgreSQL triple store. This design keeps the reasoning graph lean for traversal performance while preserving comprehensive provenance in the relational layer.

VIII. GRAPH-RAG RETRIEVAL LAYER

A. Entity Resolution

Upon receiving a natural language query, the retrieval layer first performs entity extraction using a multi-resolution matching strategy. Query tokens are evaluated against the graph node index using a priority-ordered sequence: (1) n-gram generation with preferential ordering from trigrams to unigrams, preserving longer surface forms; (2) stop-word filtering to remove semantically vacuous tokens; (3) exact match lookup against node `name_lower` values; (4) prefix substring containment matching; and (5) Levenshtein edit distance with threshold ≤ 2 for approximate matching. Matching is performed simultaneously across all node label types without pre-assuming entity category, enabling schema-flexible entity resolution.

B. Traversal Strategy Selection

KnowPro implements a deterministic, config-driven strategy selection mechanism (Tier 1). A keyword classifier maps query-level tokens to a priority-ordered strategy registry defined in a configuration file. First-match semantics ensure that more specific strategy patterns take precedence over generic ones. Table III presents the five traversal strategies and their associated query contexts.

TABLE III
GRAPH-RAG TRAVERSAL STRATEGY TAXONOMY

Strategy	Description	Use Case
Targeted	1-hop, fixed relation and direction	Direct entity lookup
Chained	k-hop fixed sequence traversal	Causal chains
Variable-hop	Free exploration ≤ N hops, any direction	Open-ended exploration
Shortest-path	Minimal edge path between 2+ entities	Relation discovery
Shared-neighbor	Intersection of entity neighborhoods	Common context queries

C. Cypher+-Based Graph Traversal

Each strategy generates one or more parameterized Cypher queries executed directly against the Neo4j Aura instance. The system does not employ natural language to Cypher translation, eliminating a significant source of query reliability failure identified in prior Graph-RAG systems. Python graph traversal

abstractions are not used; all graph computation is delegated to the Neo4j query engine, ensuring correctness and leveraging native graph index optimization.

All five strategies return a unified subgraph representation dictionary conforming to the schema: {nodes, relationships, strategy, hop depth} where nodes = [{id, label, name, properties}] and relationships = [{source_id, target_id, type, properties}]. This unified interface decouples strategy implementation from context serialization.

D. Context Serialization and Answer Generation

The retrieved subgraph is serialized into a structured natural language context string and passed to the language model (currently Gemini 2.5 Flash, configured as a swappable interface). Hallucination suppression is enforced through a hard grounding constraint embedded in the system prompt: the language model is instructed to generate responses exclusively from the provided graph context and to explicitly acknowledge absence of evidence when queried information is not present in the retrieved subgraph. No post-generation fact-verification layer is currently implemented; this is identified as a limitation.

E. Memory and Session Caching

Schema discovery—the process of enumerating available node labels and relationship types in the Neo4j instance—is cached per session, reducing cold-start latency from 3.2 seconds to zero for repeated queries within the same session. The knowledge graph itself functions as the long-term semantic memory of the system. Cross-session query caching and episodic user memory are identified as directions for future work.

F. Schema-Agnostic Tier (Proposed)

A second retrieval tier is proposed as a forward-looking architectural extension. Tier 2 eliminates the strategy selection step entirely, instead executing a variable-hop traversal over the full graph and delegating subgraph filtering and relevance assessment to the language model. This schema-agnostic mode enables querying over graphs with unknown or heterogeneous ontological structures. Tier 2 is implemented as a proof-of-concept module and is not included in the primary validation results.

IX. EXPERIMENTAL VALIDATION

A. Evaluation Scope

The evaluation in this draft is intentionally scoped to implementation validation rather than exhaustive benchmark comparison. We report automated test coverage, end-to-end execution traces, and a representative grounded query demonstration. Public benchmark comparisons for extraction and retrieval are reserved for future work because the current manuscript does not yet include a complete, unified annotation and scoring setup across all target corpora.

B. System Setup

The KnowPro framework was instantiated and validated on a medical knowledge domain. The knowledge graph was populated by ingesting a curated corpus of biomedical literature, producing a graph containing entities of types `PROTEIN`, `DISEASE`, `DRUG`, `SYMPTOM`, and `CONCEPT` with relations including `INHIBITS`, `TREATS`, `CAUSES`, `IS_A`, and `INTERACTS_WITH`. The system components operate on: Python 3.10+, spaCy 3.x (`en_core_web_sm` pipeline), `fastcoref`, PyMuPDF 1.24+, Neo4j Aura, PostgreSQL 15, and Gemini 2.5 Flash via the Anthropic-compatible interface.

C. Pipeline Validation

End-to-end pipeline correctness was validated through a suite of 105 automated unit and integration tests covering: document ingestion and block classification, sentence segmentation and coreference resolution output schemas, rule-based and transformer extraction output conformance to the `ExtractedTriple` interface, confidence routing threshold behavior, PostgreSQL provenance record integrity, and Neo4j `MERGE` idempotency. All 105 tests pass consistently.

Four representative end-to-end pipeline traces were executed, each involving document ingestion through Graph-RAG answer generation. All four traces produced graph-grounded answers for queries of varying complexity including targeted single-entity lookups, chained multi-hop queries, and shortest-path discovery queries. These traces are presented as qualitative validation rather than a substitute for benchmark evaluation.

D. Graph-RAG Demonstration

As a representative validation, the query "What inhibits tumor growth?" was submitted to the system. Entity resolution identified "tumor growth" via exact match in the graph node index. The Tier 1 strategy classifier selected the targeted strategy based on the verb "inhibits" in the keyword registry. A 1-hop Cypher query retrieved the subgraph of entities with `INHIBITS` relationships to the `tumor_growth` node. The language model generated a grounded response attributing inhibitory interactions to BRCA1 and TP53 nodes, consistent with the graph content. No external knowledge was introduced.

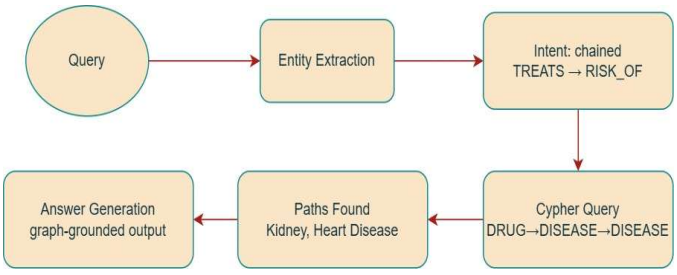

Fig. 4. Graph-RAG query trace for the query "What inhibits tumor growth?" showing entity resolution, targeted strategy selection, and grounded answer generation from the BRCA1/TP53 subgraph.

E. Efficiency Observations

Cold-start schema discovery latency was measured at 3.2 seconds; subsequent queries within the same session benefited from session caching, reducing this to near-zero. Single-hop Cypher queries executed in under 50 ms on the Aura instance. Document ingestion throughput and extraction latency were not

formally benchmarked at scale, representing a direction for future quantitative evaluation.

X. DISCUSSION AND LIMITATIONS

A. Design Trade-offs

The dual-layer storage architecture introduces a deliberate design trade-off: provenance richness is concentrated in the relational layer at the cost of cross-system join complexity for full evidence reconstruction. This trade-off is appropriate for the target use case, where the Neo4j reasoning graph serves traversal-intensive query workloads while PostgreSQL serves audit and review workflows. The separation of concerns enables independent scaling of retrieval and provenance operations.

The choice of max-confidence fusion over weighted averaging for triple deduplication prioritizes interpretability and computational simplicity. Weighted averaging could theoretically improve calibration when both extractors are confident; however, the absence of inter-extractor calibration data makes such weighting premature. This is deferred to future work.

B. Current Limitations

The following limitations are acknowledged. First, entity normalization is restricted to lexical normalization (case-folding and whitespace normalization), without semantic alias resolution. Synonymous entities such as “aspirin” and “acetylsalicylic acid” are currently represented as separate graph nodes. Second, no post-generation response validation layer is implemented; grounding relies on prompt-level constraints only. Third, the current evaluation is scoped to a single medical domain graph and implementation validation traces, so multi-domain generalization and benchmark-level retrieval quality have not yet been characterized. Fourth, long-term cross-session memory and large-scale graph performance (>1M nodes) have not been evaluated.

XI. FUTURE WORK

Several extensions are planned to address current limitations and expand framework capabilities. Semantic entity resolution using embedding-based similarity or biomedical ontology linking (e.g., UMLS/MeSH) will be integrated into the normalization pipeline. A post-generation fact-verification layer using graph-constrained answer checking will be developed to supplement prompt-level grounding. The schema-agnostic Tier 2 retrieval mode will be formally implemented and evaluated. Cross-session query caching and episodic memory will be incorporated into the retrieval layer. Multi-domain validation across legal, scientific, and engineering corpora is planned for comprehensive generalization assessment.

XII. CONCLUSION

This paper presented KnowPro, a unified framework for confidence-aware knowledge graph construction and strategy-based Graph-RAG retrieval from unstructured scientific text. The framework's principal architectural contributions—hybrid symbolic-neural extraction with max-confidence fusion, configurable three-tier confidence routing, dual-layer provenance-and-reasoning storage, five-strategy deterministic graph traversal with multi-resolution entity resolution, and strict prompt-level hallucination grounding—collectively address the reliability, traceability, and retrieval quality deficiencies of existing pipelines. System validation confirmed end-to-end correctness and efficient graph-grounded answer generation on a medical knowledge domain. KnowPro provides an extensible, principled foundation for trustworthy knowledge engineering from scientific corpora.

. REFERENCES

[1] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni, "Open information extraction from the web," in Proc. IJCAI, 2007, pp. 2670–2676.

[2] A. Fader, S. Soderland, and O. Etzioni, "Identifying relations for open information extraction," in Proc. EMNLP, 2011, pp. 1535–1545.

[3] M. Schmitz, R. Bart, S. Soderland, O. Etzioni et al., "Open language learning for information extraction," in Proc. EMNLP-CoNLL, 2012, pp. 523–534.

[4] L. Del Corro and R. Gemulla, "ClausIE: Clause-based open information extraction," in Proc. WWW, 2013, pp. 355–366.

[5] M. A. Hearst, "Automatic acquisition of hyponyms from large text corpora," in Proc. COLING, 1992, pp. 539–545.

[6] R. White, N. Rastogi, K. Duh, and B. Van Durme, "Inference is everything: Recasting semantic resources as inference problems," in Proc. EMNLP, 2016.

[7] I. Beltagy, K. Lo, and A. Cohan, "SciBERT: A pretrained language model for scientific text," in Proc. EMNLP, 2019, pp. 3615–3620.

[8] S. Oren, S. Kotlerman, Y. Goldberg, and I. Dagan, "LSOIE: A large-scale dataset for supervised open information extraction," in Proc. EACL, 2021, pp. 2386–2397.

[9] P. Bhardwaj, S. Padia, and A. Jain, "CaRB: A crowdsourced benchmark for open IE," in Proc. EMNLP, 2019, pp. 6261–6266.

[10] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in Proc. ICML, 2017, pp. 1321–1330.

[11] A. Kuleshov and S. Ermon, "Accurate uncertainties for deep learning using calibrated regression," in Proc. ICML, 2018.

[12] S. Amershi, M. Cakmak, W. B. Knox, and T. Kulesza, "Power to the people: The role of humans in interactive machine learning," AI Magazine, vol. 35, no. 4, pp. 105–120, 2014.

[13] P. Lewis, E. Perez, A. Piktus et al., "Retrieval-augmented generation for knowledge-intensive NLP tasks," Adv. Neural Inf. Process. Syst., vol. 33, 2020.

[14] X. He, Y. Tian, Y. Sun et al., "G-retriever: Retrieval-augmented generation for textual graph understanding and question answering," arXiv:2402.07630, 2024.

[15] D. Edge, H. Trinh, N. Cheng et al., "From local to global: A graph RAG approach to query-focused summarization," arXiv:2404.16130, 2024.

