Vrin: From Retrieval to Reasoning — A Hybrid Knowledge Graph Architecture for Enterprise AI

Vedant Patel[a,1]

[a]Vrin, https://vrin.cloud

February 2026

Retrieval-Augmented Generation (RAG) has become the standard approach for grounding large language model responses in factual data. Yet current implementations remain fundamentally limited: they retrieve text chunks through semantic similarity and rely on the LLM to reason over unstructured fragments. This approach fails on queries requiring multi-document reasoning, temporal awareness, or numerical constraints. We present Vrin, a hybrid knowledge graph architecture that transforms RAG from a retrieval mechanism into a reasoning engine. Vrin introduces a multi-stage pipeline comprising entity-centric fact extraction with coreference resolution, temporal fact versioning with conflict detection, confidence-scored multi-hop graph traversal with parallel reasoning strategies, constraint-aware retrieval across five types, hybrid BM25 and vector search with reciprocal rank fusion, multi-dimensional retrieval confidence assessment with adaptive bail-out, and focused chain-of-thought reasoning. On MultiHop-RAG, Vrin achieves 95.1% accuracy, outperforming GPT 5.2 (78.9%) with the same evidence documents. These results demonstrate that the gap between retrieval and reasoning is not a limitation of LLMs, but of the architectures surrounding them.

Retrieval-Augmented Generation | Knowledge Graphs | Multi-Hop Reasoning | Enterprise AI | Hybrid Search

The Retrieval-Augmented Generation paradigm has achieved remarkable adoption since its introduction Lewis et al. (2020). By 2026, RAG underpins most enterprise AI deployments, from customer support chatbots to financial analysis tools. Yet a striking paradox persists: the vast majority of enterprise AI pilots fail to deliver measurable ROI. Organizations invest in vector databases, embedding pipelines, and prompt engineering, only to find that their AI systems cannot reliably answer questions that a junior analyst could handle with a spreadsheet and ten minutes of reading.

The root cause is architectural. Current RAG systems perform one operation exceedingly well (semantic similarity search) and rely on the LLM to do everything else. When a user asks *"How did TechCorp's revenue change after the CEO transition in Q3?"*, a standard RAG system finds the most semantically similar text chunks and feeds them to the model. It does not identify the entities involved, the temporal constraint, the causal relationship, or the comparison being requested. All of these reasoning steps are delegated to the LLM as an implicit, unstructured task.

This is not how reasoning works. Consider how a human analyst would approach the same question through five distinct cognitive subprocesses:

1. **Perceive**: Identify the key entities and relationships
2. **Structure**: Formalize the question as a constrained search
3. **Store**: Know where the relevant information lives
4. **Organize**: Connect related facts across documents
5. **Retrieve**: Pull the specific facts needed

Each subprocess is a distinct engineering challenge. The current RAG paradigm addresses only the last one, and does so imprecisely through semantic matching rather than structured retrieval. The first four subprocesses are entirely absent.

Vrin is built on the thesis that *each of these five cognitive subprocesses must be engineered explicitly.* Rather than treating retrieval as a single undifferentiated step, Vrin implements a multi-stage reasoning pipeline where knowledge is perceived through entity-centric extraction, structured as typed facts in a knowledge graph, stored with temporal metadata and confidence scores, organized through conflict detection and version management, and retrieved through constraint-aware multi-hop traversal fused with hybrid search.

The result is a system that doesn't just find relevant text; it reasons over structured knowledge. Vrin achieves 95.1% on MultiHop-RAG versus 78.9% for GPT 5.2 with the same evidence.

We believe the industry has explored less than 5% of the innovation space in knowledge-augmented AI, and the remaining 95% lies not in better embeddings or larger context windows, but in the engineering of perception, structure, storage, and organization.

1. Related Work

A. Traditional RAG.. The original RAG framework Lewis et al. (2020) established the retrieve-then-generate paradigm. Subsequent work improved individual components, including better embeddings Neelakantan et al. (2022), smarter chunking Anthropic (2024), and more effective reranking Nogueira and Cho

Significance Statement

Enterprise AI systems powered by Retrieval-Augmented Generation (RAG) consistently fail to deliver reliable reasoning. Current RAG architectures retrieve text chunks via semantic similarity and delegate all reasoning to a language model, an approach that breaks on multi-document, temporal, and numerical queries. Vrin introduces a hybrid knowledge graph architecture with entity-centric extraction, temporal fact versioning, constraint-aware retrieval, multi-hop graph traversal fused with vector search, and adaptive retrieval confidence assessment. On the MultiHop-RAG benchmark, Vrin achieves 95.1% accuracy, outperforming GPT 5.2 by 16.2 percentage points even when GPT is given the same evidence documents.

V.P. designed and implemented the Vrin system, conducted all experiments, and wrote the paper.

The author is the founder of Vrin. Benchmark evaluation code is open-source.

[1]To whom correspondence should be addressed. E-mail: vedant@vrin.cloud

(2019), but the fundamental architecture remains unchanged: the query is a bag of semantics, the knowledge base is a bag of chunks, and the LLM bridges the gap.

B. Knowledge Graph Approaches.. Microsoft's GraphRAG Edge et al. (2024) introduced community summaries built from knowledge graphs, enabling global queries over document collections. However, GraphRAG lacks temporal awareness, constraint handling, or real-time hybrid search. T-GRAG Li et al. (2025a) addressed temporal reasoning, concurrent with Vrin's temporal fact versioning. T-GRAG focuses specifically on temporal queries, while Vrin's temporal system is one component of a broader multi-stage pipeline.

C. Reasoning-Enhanced RAG.. Self-RAG Asai et al. (2024) teaches models to self-reflect on retrieval quality. CoT-RAG Li et al. (2025b) integrates chain-of-thought with retrieval decisions. DeepRAG Guan et al. (2025) models retrieval as a Markov Decision Process, making adaptive decisions about when to retrieve. These systems advance reasoning in RAG but typically focus on a single dimension rather than addressing the full perception-to-retrieval pipeline.

D. Entity-Centric Retrieval.. Entity Similarity RAG Wang et al. (2025) introduced entity-centric retrieval using entity similarity for knowledge graph construction. Vrin shares the intuition that entities should be first-class objects but extends it with coreference resolution, temporal versioning, constraint handling, and hybrid graph-vector fusion.

E. Positioning.. Vrin is not the first system to use knowledge graphs for RAG, nor the first to integrate reasoning with retrieval. Its contribution is the *integration* of these capabilities into a unified, production-grade pipeline: entity-centric extraction, temporal versioning, five-type constraint solving, confidence-scored multi-hop traversal, hybrid search with rank fusion, cross-encoder reranking, and focused chain-of-thought, all operating together on every query.

2. Architecture

Vrin's architecture maps to the five cognitive subprocesses identified in Section 1. We describe each in order: the insertion pipeline (perceive and structure), the knowledge graph (store and organize), and the query pipeline (retrieve and reason).

A. Knowledge Insertion Pipeline.. When a document enters Vrin, it passes through a multi-stage extraction pipeline that transforms unstructured text into structured, entity-linked facts.

Two-Phase Entity-Centric Extraction. Standard RAG systems chunk documents and embed the chunks. Vrin takes a fundamentally different approach: it extracts structured facts as subject-predicate-object triples, where subjects are always concrete, named entities.

The extraction operates in two phases. In **Phase 1** (Entity Identification), an LLM identifies the main entities in each text chunk: organizations, people, products, projects, and locations. Each entity is typed and scored for importance. In **Phase 2** (Entity-Linked Fact Extraction), the LLM extracts subject-predicate-object triples using the identified entities as context anchors.

A critical requirement is **coreference resolution**: the system resolves pronouns and indirect references ("it," "the company," "they") to their concrete entity referents before creating facts. This ensures that "TechCorp announced earnings. It reported $245M in revenue" produces the triple `(TechCorp, reported_revenue, $245M)` rather than `(It, reported_revenue, $245M)`.

Attribute vertex prevention ensures literal values such as monetary amounts, percentages, and years are stored as edge properties rather than graph vertices. Without this, "$245M" and "2024" would become entity nodes, creating spurious connections across unrelated facts.

Every extracted fact carries a **confidence score** between 0.0 and 1.0. Facts below a configurable threshold are discarded. The extraction model, timestamp, and source document are recorded for full provenance.

Table-Aware Processing. Financial and technical documents frequently contain tables. Standard chunking destroys table structure, making it impossible for the LLM to reason about row-column relationships. Vrin detects tables in multiple formats (markdown, HTML, JSON, CSV) and extracts them separately, preserving structural information needed for numerical reasoning.

Visual Intelligence. Vrin processes visual content using multimodal vision models. Charts are converted to structured JSON with axis labels and data points, transforming a bar chart into queryable facts. Diagrams are decomposed into nodes and edges. Cross-modal embeddings (1024 dimensions) place images and text in the same vector space, enabling text queries to discover relevant visual content.

B. Knowledge Graph with Temporal Versioning.. Vrin stores extracted knowledge in a property graph (Amazon Neptune) alongside vector embeddings in a search index (Amazon OpenSearch). This dual-store architecture enables both structured graph traversal and semantic similarity search.

Graph Structure. The knowledge graph consists of typed entity vertices connected by fact edges. Each edge carries the predicate, confidence score, temporal validity window (`valid_from`, `valid_to`), status (active, superseded, corrected), numerical metadata (queryable numeric value, unit, and type), and source attribution. This structure enables queries that standard RAG cannot handle: "What was TechCorp's revenue in 2022?" becomes a graph traversal with a temporal filter.

Temporal Fact Versioning. Knowledge evolves. A company's CEO changes, revenue figures are updated quarterly. Standard RAG treats all information as equally current, leading to contradictions. Vrin implements automatic temporal versioning:

- **Conflict detection**: New facts with the same subject-predicate but different objects trigger conflict resolution
- **Supersession**: Older facts are closed with `valid_to` set and status changed to "superseded"; both versions remain linked
- **Adaptive granularity**: Temporal resolution adapts to update frequency: day-level for rapidly changing facts, year-level for infrequent changes
- **Time-travel queries**: Temporal metadata enables point-in-time retrieval

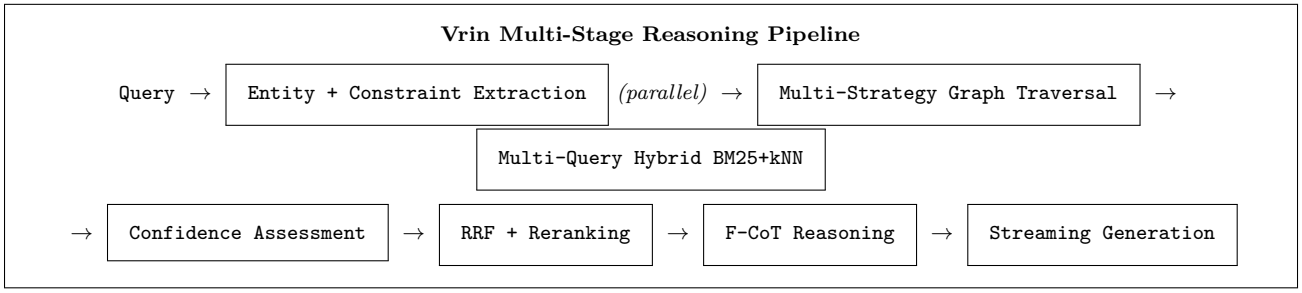

Fig. 1. The Vrin query pipeline. Seven stages transform a natural language query into a reasoned, evidence-grounded response. Stages 1–3 operate in parallel where possible. Stage 4 can short-circuit the pipeline when the knowledge base lacks sufficient coverage.

C. Multi-Stage Query Pipeline.. When a query arrives, Vrin processes it through a seven-stage pipeline where each stage enriches the context for the next (Figure 1).

Stage 1: Parallel Entity and Constraint Extraction. Two operations run in parallel. **Entity extraction** identifies the entities mentioned in the query, which become starting points for graph traversal. **Constraint identification** extracts five constraint types:

1. **Temporal**: years, quarters, date ranges
2. **Numerical**: comparisons, ranges, specific values
3. **Entity**: specific subjects or objects required
4. **Comparison**: requests to compare entities or metrics
5. **Aggregation**: totals, averages, percentages

These constraints actively filter downstream retrieval; a temporal constraint narrows graph traversal to facts within the specified window; a numerical constraint filters by stored numeric values.

Stage 2: Multi-Strategy Graph Traversal. Using extracted entities as seeds, Vrin performs multi-hop beam search:

- **Hop 0**: Rich entity matching (exact, case-insensitive, fuzzy)
- **Hops 1+**: Batched expansion with multiplicative confidence decay per hop
- **Hub detection**: High-degree entities receive restricted fan-out
- **Path confidence floor**: Expansion stops on low-confidence paths

The traversal produces entity chains, temporal chains, and causal chains. A cross-document synthesizer identifies entities appearing across multiple documents, detects temporal overlaps, and flags contradictions.

For complex queries classified as multi-hop or comparative involving two or more entities, Vrin generates three traversal strategies: *focused* (aggressive confidence decay, narrow expansion), *balanced* (default parameters), and *exploratory* (conservative decay, broad expansion), executing them in parallel. Results are merged via reciprocal rank fusion, boosting facts discovered independently by multiple strategies.

Stage 3: Multi-Query Hybrid Search. An LLM selects one of two strategies per query: **Rephrase** (alternative phrasings for narrow queries) or **Decompose** (aspect-specific sub-queries for broad queries). Multiple variants are searched in parallel, each using hybrid BM25 keyword matching plus kNN vector similarity with score normalization and relevance filtering.

Stage 4: Retrieval Confidence Assessment. Before committing to LLM generation, Vrin evaluates retrieval quality across five dimensions: entity coverage, type alignment, temporal alignment, fact density, and topical relevance. Entity coverage measures what fraction of queried entities were found in the knowledge base. Topical relevance matches query terms against fact predicates and objects, detecting cases where an entity exists but the queried topic does not (for example, asking about a company's environmental record when only financial data is available).

When entity coverage is zero and no relevant facts or chunks were retrieved, the system short-circuits LLM generation entirely, returning a structured "insufficient information" response. This *adaptive bail-out* saves 1.5–5.5 seconds per null query while eliminating hallucinated answers about unknown entities. When entity coverage is high but topical relevance is zero, a *soft-miss bail-out* prevents the LLM from generating plausible-sounding but groundless responses.

Stage 5: Result Fusion and Reranking. Reciprocal Rank Fusion (RRF) Cormack et al. (2009) combines graph facts and vector chunks in a score-scale-independent manner. A cross-encoder reranker then re-scores candidates against the original query, producing significant precision improvements.

Stage 6: Focused Chain-of-Thought. Structured reasoning instructions are constructed based on the query type and retrieved evidence. A memory system provides supplementary context from pre-computed domain knowledge packs, adding reasoning hints with minimal token overhead.

Stage 7: Streaming Generation. A configurable LLM (supporting OpenAI, Anthropic, Google, xAI) generates the response, receiving structured graph facts with confidence scores, relevant text chunks with attribution, reasoning chains, visual descriptions, and constraint information, all fundamentally richer context than concatenated chunks.

D. Enterprise Data Sovereignty.. Vrin supports three deployment modes: Vrin Cloud, Hybrid Cloud, and Private VPC. For enterprise customers, the knowledge graph, vector index, document storage, and embedding computation reside entirely within the customer's cloud account. Vrin's compute layer accesses customer data through time-limited, scoped credentials. The API key prefix transparently determines which infrastructure a request uses. Enterprise data never leaves the customer's cloud.

3. Implementation

Vrin is implemented as a serverless system on AWS: Lambda functions with FastAPI for streaming, Neptune for the knowledge graph, OpenSearch for hybrid search, Bedrock for embeddings (Cohere embed-english-v3, 1024 dimensions) and reranking (Cohere Rerank 3.5), S3 for documents, and DynamoDB for metadata. Cost-efficient model selection uses lightweight models for preprocessing and more capable models for generation.

Vrin also operates as a **Model Context Protocol (MCP) server**, enabling integration with Claude, ChatGPT, and custom AI agents; any MCP-compatible assistant can query Vrin's knowledge graph as a reasoning backend.

4. Experimental Evaluation

A. Methodology.. We evaluate on MultiHop-RAG Tang and Yang (2024), a benchmark specifically designed for cross-document multi-hop reasoning, the core capability that distinguishes Vrin from standard RAG systems. Following BetterBench guidelines Reuel et al. (2024), we use stratified sampling (seed=42), report 95% confidence intervals with finite population correction, and make all evaluation code open-source.*

A three-stage evaluation pipeline ensures fair assessment: (1) direct substring matching, (2) LLM-based answer normalization to extract core answers from verbose responses, and (3) semantic pattern matching. The same pipeline evaluates both Vrin and baselines.

B. MultiHop-RAG Results.. MultiHop-RAG Tang and Yang (2024) consists of 2,556 queries requiring reasoning across 2–4 news articles. We evaluate on 384 stratified samples.

Table 1. MultiHop-RAG benchmark results.

System	Accuracy	95% CI
Vrin (Hybrid RAG)	**95.1%**	[90.5, 99.7]
GPT 5.2 (w/ evidence)	78.9%	[74.3, 83.5]
Multi-Meta RAG + GPT-4	63.0%	—
IRCoT + GPT-4	58.2%	—
Standard RAG + GPT-4	47.3%	—

Vrin and GPT 5.2: 384 stratified samples each (seed=42). GPT receives oracle evidence documents. Published baselines from Tang and Yang (2024).

Vrin outperforms GPT 5.2 by **+16.2 percentage points** even when GPT is given the exact same evidence documents. This controls for retrieval quality; the gap is entirely attributable to how evidence is structured and reasoned over.

The per-type breakdown (Table 2) reveals where structured reasoning matters most. **Inference queries**: both systems perform well (Vrin 99.2%, GPT 98.4%), as these are single-hop lookups where entity-centric extraction and oracle context both suffice. **Comparison queries (Vrin +15.5pp)**: Vrin's constraint solver explicitly identifies comparison operations and structures retrieval to find both sides; GPT must infer the comparison structure from raw text. **Temporal queries (Vrin +48.9pp)**: the largest gap. Temporal versioning and constraint extraction ensure the right facts from the right time periods are retrieved, while GPT struggles to reason over temporal relationships in unstructured evidence. **Null queries (Vrin 95.5%, GPT 100%)**: Vrin's retrieval confidence assessment with adaptive bail-out (Section 3, Stage 4) detects zero entity coverage and topical mismatches before LLM generation, correctly identifying 95.5% of queries about absent entities or topics. GPT achieves 100% because oracle context directly signals when information is absent, a condition that does not exist in production retrieval settings.

Table 2. Accuracy by question type on MultiHop-RAG (384 samples).

Type	Vrin	GPT 5.2	n
Inference	**99.2%**	98.4%	123
Comparison	**94.6%**	79.1%	129
Temporal	**89.8%**	40.9%	88
Null	**95.5%**	100.0%	44

Stratified by question type. Vrin leads in all categories except null queries. Largest gaps: temporal (+48.9pp) and comparison (+15.5pp).

C. Analysis.. The results support two conclusions. First, **structured reasoning outperforms context stuffing**. The GPT 5.2 comparison controls for retrieval quality, isolating the value of structured processing. Second, **the gap is largest where structure matters most**: temporal queries show a 48.9 percentage point advantage for Vrin, and comparison queries show 15.5 percentage points. These are precisely the query types that require understanding the *structure* of the question (temporal constraints, comparison operations) rather than simply finding semantically similar text.

D. Discussion: Oracle Context vs. Retrieved Context.. A critical distinction in RAG evaluation is the difference between *oracle context* (where the exact documents needed are placed in the LLM's context window) and *retrieved context*, where the system must find relevant information within a larger corpus containing both signal and noise. This distinction is often overlooked, yet it dominates real-world performance.

Recent work has quantified the gap. The RGB benchmark Chen et al. (2024) found that retrieval noise degrades LLM accuracy by 20–34 percentage points depending on model size. Meta's CRAG benchmark Yang et al. (2024) found that even the best commercial RAG system (Microsoft Copilot Pro) achieves only 62.6% accuracy on realistic retrieval tasks, with 16–25% hallucination rates across all tested systems. Google Research Joren et al. (2025) demonstrated that insufficient retrieved context increases error rates by 6.5× compared to having no context at all; RAG with bad retrieval is worse than no RAG.

In our evaluation, GPT 5.2 received the exact evidence documents for each query directly in its context window, an oracle setup where retrieval quality is perfect by construction. Vrin, by contrast, retrieved relevant facts and passages from the full ingested corpus of 609 news articles, which contains both relevant and irrelevant content for any given query. That Vrin achieves 95.1% through noisy retrieval while GPT 5.2 achieves only 78.9% with oracle context suggests the effective reasoning gap is substantially larger than the headline 16.2 percentage points.

This asymmetry reflects a broader reality: enterprise RAG

*https://github.com/Vrin-cloud/vrin-benchmarks

systems operate on private, noisy, multi-source corpora where retrieval quality varies by query. No major enterprise RAG provider, including those valued at billions of dollars, competes on standardized public benchmarks. The industry has converged on custom evaluations against real enterprise data as more meaningful measures than leaderboard positions on academic benchmarks with artificial retrieval conditions. Vrin's benchmark methodology is designed with this in mind: the system ingests the full document corpus and retrieves under realistic conditions, then is compared against the strongest possible baseline: a frontier LLM with perfect context.

5. Production Readiness

Vrin operates as a production system. Expert-mode queries complete in under 20 seconds; simpler queries in 3–5 seconds. Null queries (questions about entities or topics absent from the knowledge base) are detected and answered in under 500 milliseconds through the adaptive bail-out system, eliminating unnecessary LLM calls. Responses stream in real-time via Server-Sent Events. The serverless architecture scales automatically with demand. All external calls include timeout, retry, and exponential backoff.

The system supports multiple LLM providers (OpenAI, Anthropic, Google, xAI) and operates as an MCP server for integration with Claude, ChatGPT, and custom agents. Current traction includes one finalized enterprise contract, two active pilots (including UC Davis Health), and five to seven additional enterprises in the pipeline, all inbound.

6. Vision: The 95% Unexplored

We believe the RAG industry has explored less than 5% of the available innovation space. The dominant focus has been on improving the *retrieval* subprocess: better embeddings, smarter reranking, larger context windows. The other four subprocesses remain largely unaddressed.

Vrin's five-subprocess framework reveals a vast surface area for future innovation:

Adaptive Retrieval. Vrin's confidence-based bail-out system, inspired by DeepRAG's Guan et al. (2025) modeling of retrieval as a decision process, demonstrates the value of adaptive retrieval by detecting insufficient coverage before LLM generation. The current implementation makes a binary decision: proceed or bail out. Future versions will make finer-grained adaptive decisions about which pipeline stages to invoke. Simple factual queries may need only graph traversal, while questions about general knowledge may not need retrieval at all.

Automatic Domain Specialization. Infrastructure exists for automatic specialization, learning domain expertise from query patterns and feedback. Future versions will detect that a knowledge base is finance-focused or healthcare-focused and adjust extraction, retrieval, and reasoning accordingly.

Iterative Reasoning. For queries requiring five or more reasoning steps, iterative decomposition, where each step uses intermediate answers to formulate the next sub-query, could discover paths that single-pass traversal misses.

Knowledge Graph Pattern Detection and Model Specialization. Over time, usage patterns reveal which subgraphs and entity clusters are most frequently retrieved: specific teams repeatedly query the same financial entities, the same regulatory frameworks, the same product hierarchies. Vrin is building infrastructure to detect these patterns over months of usage and automatically create memory packs from the most heavily-accessed subgraphs. These memory packs then become the foundation for fine-tuning smaller, domain-specialized models. A model trained on a healthcare team's most-queried knowledge subgraph will outperform a general-purpose model on that team's queries while running at a fraction of the cost. This creates a virtuous cycle: structured knowledge in the graph enables precise pattern detection, pattern detection enables targeted memory pack creation, and memory packs enable efficient domain specialization per team and per concept.

The fundamental thesis is that AI systems will eventually be specialized like human employees, not through fine-tuning a single model, but through engineering the cognitive infrastructure surrounding it. Vrin is building that infrastructure.

7. Conclusion

The transition from retrieval to reasoning is not incremental; it is architectural. Vrin demonstrates that this transition is both possible and measurable: 95.1% on multi-hop reasoning versus 78.9% for GPT 5.2 with the same evidence documents.

These results are achieved not by using a more powerful language model, but by engineering the cognitive infrastructure around the model: entity-centric extraction, temporal versioning, constraint-aware retrieval, multi-hop traversal, hybrid search with rank fusion, adaptive confidence assessment, and focused chain-of-thought reasoning.

The gap between retrieval and reasoning represents the largest opportunity in enterprise AI. Our benchmark evaluation code is open-source at https://github.com/Vrin-cloud/vrin-benchmarks.

ACKNOWLEDGMENTS. We thank the early enterprise partners and pilot users who provided feedback that shaped Vrin's architecture. Benchmark datasets are provided by the MultiHop-RAG team.

References

Anthropic (2024). Contextual retrieval. https://www.anthropic.com/news/contextual-retrieval.

Asai, A., Wu, Z., Wang, Y., Sil, A., and Hajishirzi, H. (2024). Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations*.

Chen, J., Lin, H., Han, X., and Sun, L. (2024). Benchmarking large language models in retrieval-augmented generation. *Proceedings of AAAI*.

Cormack, G. V., Clarke, C. L., and Buettcher, S. (2009). Reciprocal rank fusion outperforms Condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference*, pages 758–759.

Edge, D., Trinh, H., Cheng, N., Bradley, J., Chao, A., Mody, A., Truitt, S., Metropolitansky, D., Osazuwa Ness, R., and Larson, J. (2024). From local to global: A graph RAG approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Guan, X., Zeng, J., Meng, F., Xin, C., Lu, Y., Lin, H., Han, X., Sun, L., and Zhou, J. (2025). DeepRAG: Thinking to retrieve step by step for large language models. *arXiv preprint arXiv:2502.01142*.

Joren, H., Zhang, J., Ferng, C.-S., Juan, D.-C., Taly, A., and Rashtchian, C. (2025). Sufficient context: A new lens on retrieval augmented generation systems. *Proceedings of ICLR*.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., and Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Li, D., Niu, Y., Ai, Y., Zou, X., Qi, B., and Liu, J. (2025a). T-GRAG: A dynamic GraphRAG framework for resolving temporal conflicts and redundancy in knowledge retrieval. *arXiv preprint arXiv:2508.01680*.

Li, F., Fang, P., Shi, Z., Khan, A., Wang, F., Wang, W., Zhang, X., and Cui, Y. (2025b). CoT-RAG: Integrating chain of thought and retrieval-augmented generation to enhance reasoning in large language models. *Findings of EMNLP*.

Neelakantan, A., Xu, T., Puri, R., Radford, A., Han, J. M., Tworek, J., Yuan, Q., Tezak, N., Kim, J. W., Hallacy, C., et al. (2022). Text and code embeddings by contrastive pre-training. *arXiv preprint arXiv:2201.10005*.

Nogueira, R. and Cho, K. (2019). Passage re-ranking with BERT. *arXiv preprint arXiv:1901.04085*.

Reuel, A., Hardy, A., Smith, C., Lamparth, M., Hardy, M., and Kochenderfer, M. J. (2024). BetterBench: Assessing AI benchmarks, uncovering issues, and establishing best practices. In *Advances in Neural Information Processing Systems*. https://betterbench.stanford.edu/.

Tang, Y. and Yang, Y. (2024). MultiHop-RAG: Benchmarking retrieval-augmented generation for multi-hop queries. *arXiv preprint arXiv:2401.15391*.

Wang, Z., Peng, B., Tu, H., and Li, X. (2025). Entity similarity RAG: Enhancing LLM answers with precise knowledge graph retrieval. In *Proceedings of ICONIP*. Springer.

Yang, X., Sun, K., Xin, H., Sun, Y., Bhalla, N., Chen, X., Cai, S., Dang, H., Sunkara, K., Jiang, Y., et al. (2024). CRAG – comprehensive RAG benchmark. *arXiv preprint arXiv:2406.04744*.