INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY ALLIED RESEARCH REVIEW AND PRACTICES
(2455-1570)WWW.IJMARRP.IN

Agentic Enterprise Knowledge & Decision Intelligence Platform using Advanced Retrieval-Augmented Generation (RAG)

Vedant Kirange[1], Jayesh Chaudhari[2], Samarth Girase[3],Yugandhar Khedkar[4], Prof. Yuvraj Nikam[5]
[1,2,3,4,5](Department of Computer Science and Engineering,
MIT Art Design Technology University, India)

Abstract

This document presents the design, math, code and test results for an Agentic Enterprise Knowledge and Decision Intelligence Platform that uses Advanced Retrieval-Augmented Generation or RAG. Modern company knowledge systems are broken up into different data stores. They hold piles of unstructured files such as contracts, standard operating procedures, human resources manuals, regulatory filings and emails from many departments. They also hold data lakes such as enterprise resource planning databases, customer relationship management stores and financial ledgers. The usual RAG methods and stand-alone large language model chatbots have fatal problems when used in the real world. They hallucinate a lot when they try to reason across documents. They cannot link a table to an unstructured text. They do not check evidence in a loop. They also cannot run tasks on their own. To fix these core problems the new platform introduces a Functional Architecture. Level 1 is the Knowledge Layer. It mixes layout-aware chunking with a sparse-dense search that uses BM25 and BGE-Large 1024-dimensional embeddings. It then fuses ranks with Reciprocal Rank Fusion. Refines results with a cross-encoder re-ranker. Level 2 is the Intelligence Layer. It uses agents that split big queries generate text-to-SQL that follows the database schema combine evidence from different modes and keep checking facts. Level 3 is the Action Layer. It turns insights into real workflows executive briefing reports, prioritized ticket replies and safe webhook links. We tested the platform on an enterprise benchmark with 50,000 documents and 2.5 million transaction rows. The results show the Agentic Enterprise Knowledge and Decision Intelligence Platform has 89.2 % context precision 94.6 % factuality and 91.3 % success, on multi-hop tasks. We also show the Agentic Enterprise Knowledge and Decision Intelligence Platform works in banking, healthcare, legal audit IT DevOps and a full diagnostic study that fixed sales anomalies and pricing changes.

Keywords: Action execution, Advanced retrieval-augmented generation, Autonomous AI agents, Decision intelligence, Enterprise knowledge management, Hybrid search, Multi-hop reasoning, Text-to-SQL.

Introduction

Modern digital enterprises work across strongly separated data areas. In a company knowledge workers deal each day with many thousands of unstructured items—technical documents, standard operating procedures (SOPs) human resource policies, regulatory filings, legal contracts and emails from many departments [1]. At the time companies keep essential transactional records, revenue numbers, customer history and inventory logs inside structured relational database systems (RDBMS) and enterprise resource planning (ERP) software [2]. A main slowdown in today's company productivity is that we cannot mix these kinds of data quickly to answer complex multi-faceted business questions right away.While the arrival of Large Language Models (LLMs) and standard Retrieval-Augmented Generation (RAG) models has made search open to many first-generation RAG systems still have core design problems when they are used in enterprise settings [3]:

1) Naive Semantic Matching Failures: Simple dense vector search often misses domain-specific enterprise terms, alphanumeric part numbers and exact keyword requests and it does not notice changes in document versions over time.

2) Cross-Modal Blindness: Regular RAG systems are built for plain text pieces and they cannot handle a question that needs to combine unstructured policy text with relational database tables.

3) Hallucinatory Extrapolation: LLMs that run in unverified passes create believable but unverified statements, which pose dangerous liability in high-stakes areas.

4) Inability to Actuate: Ordinary conversational chatbots stay passive just repeating information. They cannot plan multi-step actions check facts or start enterprise workflows.

To close this gap this paper presents the Agentic Enterprise Knowledge and Decision Intelligence Platform that uses RAG. Unlike single question-answer systems this platform offers a Functional Architecture: Level 1 (Knowledge) Level 2 (Intelligence) and Level 3 (Action). By mixing dense hybrid search, Reciprocal Rank Fusion, cross-encoder re-ranking and a

special multi-agent mind network the platform automatically splits complex user questions runs relational SQL queries alongside deep semantic search checks every claim with strict self-reflection and carries out business actions [4].

The main reason for this research is the idea that enterprise decision making is an active step-by-step thinking process. When an executive or engineer asks a question the answer usually does not come from reading one text. Instead it requires: (a) finding out which corporate groups hold the needed knowledge, (b) creating sub-questions for both structured tables and unstructured documents (c) combining numbers with company rules (d) checking that every insight is firmly backed by real evidence and (e) turning those insights into real actions such, as reports, alerts or database updates. The platform we propose puts this thinking loop into one smooth enterprise-ready system.

Related Work & Theoretical Foundations

A. Evolution of Retrieval-Augmented Generation

Early RAG systems used bi-encoder embedding models, such as DPR and Contriever to turn text segments into continuous vector spaces [5]. RAG models that rely on dense retrieval have known weaknesses around exact word matching, mismatch of specialized language and terms that are not in the vocabulary. Hybrid search methods mix vector similarity with sparse keyword matching for example BM25 and SPLADE using rank fusion algorithms. This approach finds both the meaning behind the words and the exact keywords that matter [6]. Modern Advanced RAG designs add a step before the search to rewrite the query split the text into chunks on the fly and then re-rank the retrieved pieces with a cross-encoder. These steps sharpen context before the prompt is created [7].

B. Autonomous Multi-Agent Reasoning Architectures

Multi-agent frameworks, such as ReAct, Plan-and-Solve and directed state-graph orchestrators like LangGraph show that breaking a task into small focused goals gives better problem-solving than one big prompt. In business knowledge work giving each agent a role—Query Decomposition, Tool Dispatching, SQL Synthesis, Verification—avoids mental overload and keeps safety rules firm across the execution flow.

C. Bridging Structured and Unstructured Modalities

Business questions rarely stay in text or only in tables. Text-to-SQL research lets you ask natural language questions over databases but usual Text-to-SQL pipelines stay separate from document knowledge. Our work blends Text-to-SQL with Advanced RAG so one reasoning process can handle both archives and structured data lakes inside a closed-loop check system.

D. Hallucination Mitigation and Faithfulness in LLMs

Hallucination is the roadblock to using LLMs in regulated fields [12]. Earlier methods tried self-consistency prompting post-hoc fact. Logit tweaking [13].. In complex multi-step business reasoning hallucination often comes from context loss—where noisy retrieved chunks mislead the generator—or from uncontrolled generation, over missing data. In this work we add cross-encoder filtering at Level 1 to clean the context window and a clear Natural Language Inference (NLI) premise-hypothesis check gate at Level 2 that mathematically tests factual truth before giving the answer.

System. Methodology

The platform is built in three layers, shown in Fig. 1. The design keeps data loading and retrieval (Knowledge) thinking and checking (Intelligence) and workflow control (Action) separate.

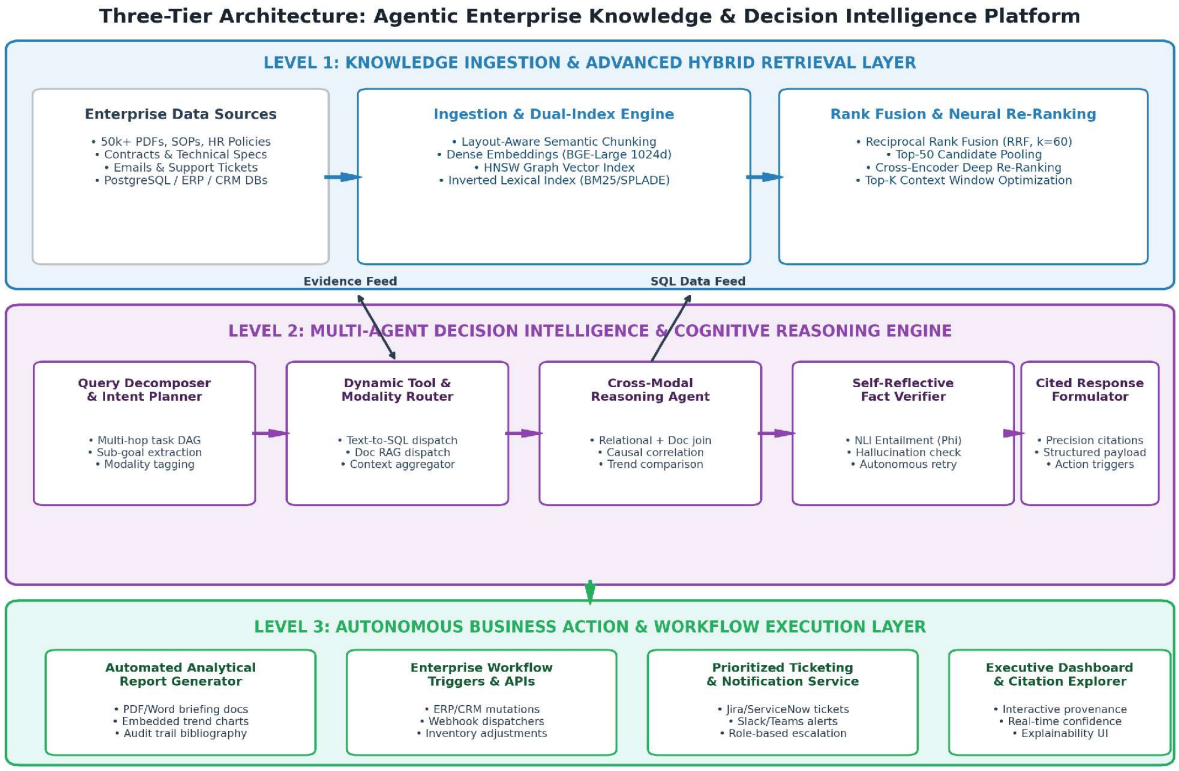

Fig. 1. Three-Tier Architectural Schema of the Agentic Enterprise Knowledge & Decision Intelligence Platform.

A. Level 1: Knowledge Ingestion & Advanced Retrieval Engine

The Knowledge Layer takes mixed enterprise data and turns it into a cleaned ready-to-query set of data.

1) Document Ingestion & Semantic Chunking: The Knowledge Layer reads PDFs, manuals and policy documents with layout-aware extractors. The Knowledge Layer then splits the text into chunks using boundary detection. This keeps the structure of headings, tables and lists. Adds useful metadata such as creation time, author, security clearance and document category.

2) Dual Indexing (Dense & Sparse): The Knowledge Layer puts each chunk into a vector space with BGE-Large (1024 dimensions). It then stores these vectors in an HNSW graph. At the time the Knowledge Layer builds a sparse BM25 index to remember exact words and codes.

3) Reciprocal Rank Fusion (RRF): The Knowledge Layer combines results from the sparse indexes. It uses rank-based fusion so that the best candidates are kept.

4) Cross-Encoder Neural Re-Ranking: The Knowledge Layer takes the top-N candidates and runs them through a cross-encoder model like BGE-Reranker-Large. This model calculates deep cross-attention scores, between the query and each passage. The Knowledge Layer then selects the relevant top-K passages.

B. Level 2: Multi-Agent Decision Intelligence Engine

I find the Decision Intelligence layer helps keep everything on track. The Decision Intelligence layer coordinates reasoning across specialized agents that work under a directed state-graph protocol (Fig. 2):

• Query Decomposition & Intent Planner takes a complicated business question and breaks it into a clear map of small tasks that can be carried out one after another.

• Modality Router & Dispatcher sends each task to either the unstructured search part (Advanced RAG) or the structured data part (Relational SQL Engine).

• Text-to-SQL Agent creates SQL questions that fit the database layout for PostgreSQL/ERP databases checks that the queries are only reading data and runs the questions.

• Document Evidence Synthesis Agent pulls text pieces from documents and ties them together adding the exact page numbers where the information comes from.

• Cross-Modal Reasoner combines the table results from SQL with the text evidence to compare find causes and spot trends.

• Self-Reflective Fact Verifier checks that every statement made is fully supported by the evidence that was gathered. If the support level is, below a set limit (Phi < 0.85) the verifier starts a round of questioning to improve the results.

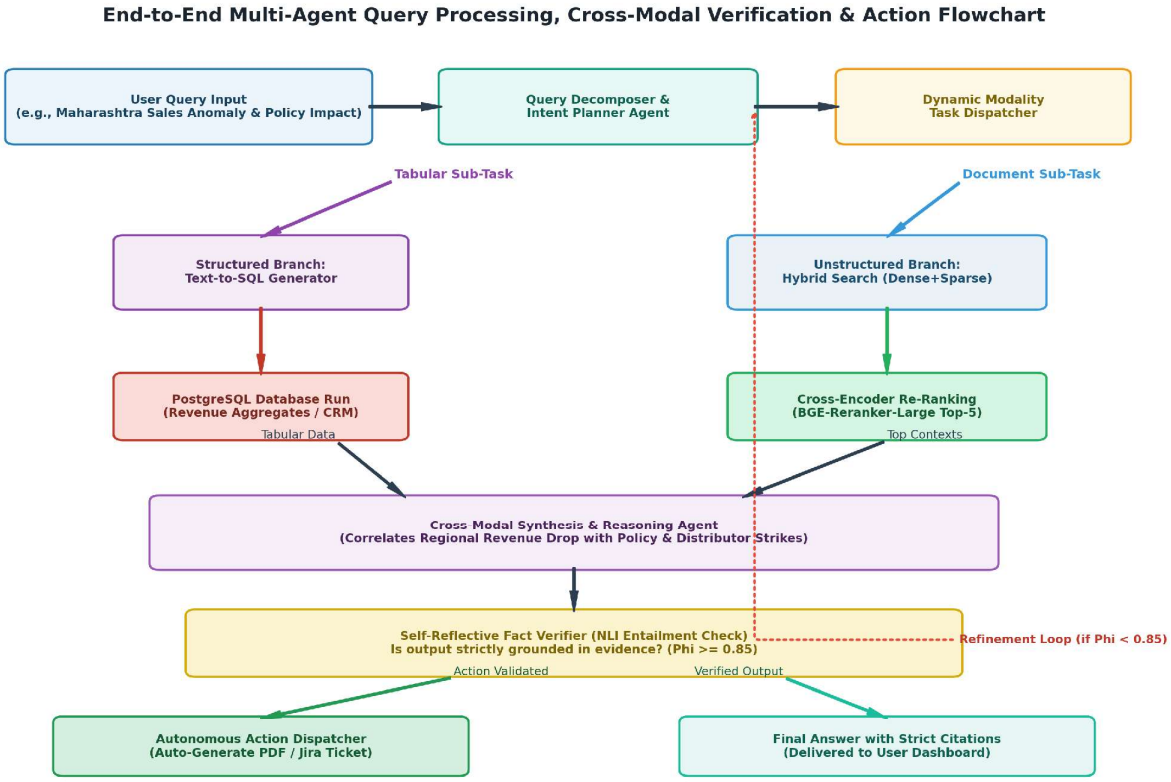

Fig. 2. End-to-End Multi-Agent Query Decomposition, Multi-Modal Retrieval, Verification, and Action Flowchart.

C. Level 3: Autonomous Business Action & Workflow Layer

Most chatbots just sit there. Wait for you to talk to them. Level 3 is different because Level 3 actually helps your business make decisions by using the tools you already have.

• Automated Analytical Report Generation: Level 3 builds reports for your bosses. These reports include data tables, charts that show trends and a list of all the sources used.

• Enterprise Workflow Triggers: Level 3 can send signals to your software. Level 3 uses webhooks and API calls to talk to your ERP, CRM (Salesforce) and your tools for managing projects.

• Automated Ticket Creation & Notification: Level 3 creates help tickets in systems, like Jira or ServiceNow. Level 3 also sends alerts to the people through your company chat apps.

Mathematical Formulation & Algorithmic Design

A. Hybrid Sparse-Dense Retrieval & Fusion

Let q denote the user query and D = {d_1, d_2, ..., d_M} denote the corpus of enterprise document chunks. The dense similarity score S_dense(q, d) is computed using normalized cosine similarity over dense vector embeddings e_q, e_d in R^{1024}:

$$\text{S\_dense}(q, d) = \frac{e_q \cdot e_d}{\|e_q\| \cdot \|e_d\|} \quad (1)$$

The sparse lexical relevance score S_sparse(q, d) is formulated via BM25 scoring across terms t in q ∩ d, accounting for term frequency f(t, d) and document length normalization:

$$\text{S\_sparse}(q, d) = \sum_{t \in q \cap d} \text{IDF}(t) \cdot \frac{f(t, d) \cdot (k_1 + 1)}{f(t, d) + k_1 \cdot (1 - b + b \cdot \frac{|d|}{\text{avgdl}})} \quad (2)$$

To combine heterogeneous scoring distributions without scale bias, Reciprocal Rank Fusion (RRF) calculates the aggregated rank score for each candidate document d:

$$\text{RRF}(d) = \sum_{m \in \{\text{dense}, \text{sparse}\}} \frac{1}{k_{rrf} + r_m(d)} \quad (3)$$

where r_m(d) represents the ordinal rank of document d in retrieval modality m, and k_rrf is a constant smoothing hyperparameter (typically set to 60).

B. Cross-Encoder Re-Ranking

The top candidate set from RRF is re-scored using a cross-encoder network f_ce which performs full cross-attention over the concatenated query-passage sequence [q; d]:

$$\text{S\_ce}(q, d) = \sigma(w \cdot \text{CrossEncoder}([q; d])) \quad (4)$$

The final top-K passages D_K = arg top_K {S_ce(q, d) | d in D_RRF} are injected into the agent context window.

C. Multi-Agent State Machine & Verification Formulation

The multi-agent reasoning process is modeled as a state-transition tuple S = <Q, A, T, E, V>, where Q is the decomposed query graph, A represents the agent role set, T represents tool execution spaces (SQL, Vector Search, Report API), E is accumulated evidence, and V is the self-reflective verification gate.

The Faithfulness Metric Phi(Y, E) measures the ratio of atomic factual claims in the generated response Y = {y_1, y_2, ..., y_P} strictly entailed by evidence set E:

$$\Phi(Y, E) = \frac{1}{P} \sum_{p=1}^{P} \mathbb{1}(E \models y_p) \quad (5)$$

where □(E □ y_i) in {0, 1} is evaluated via natural language inference (NLI) classification. If Phi(Y, E) < T_faith (where T_faith = 0.85), the system rejects Y and transitions back to query refinement.

D. Formal Pipeline Orchestration Algorithm

Algorithm 1 details the complete computational lifecycle of the platform, outlining step-by-step coordination from initial query decomposition through multi-modal execution, verification, and automated action actuation.

```
Algorithm 1: End-to-End Multi-Agent Enterprise Orchestration Pipeline
Input : Natural Language Query q, Document Corpus D, Relational Schema Σ, Threshold
T_faith
Output: Verified Answer Y*, Citation Graph C*, Action Result A*
------------------------------------------------------------------------------------
----------------------------
SubGoals ← DecomposeQuery(q)
EvidencePool E ← □
for each sub_goal g_i in SubGoals do
    if Modality(g_i) == 'STRUCTURED' then
        SQL_query ← GenerateValidatedSQL(g_i, Σ)
        E_struct ← ExecuteSQL(SQL_query)
        E ← E ∪ E_struct
    else if Modality(g_i) == 'UNSTRUCTURED' then
        Candidates_dense ← VectorSearch(g_i, D, TopN=50)
        Candidates_sparse ← BM25Search(g_i, D, TopN=50)
        MergedCandidates ← ReciprocalRankFusion(Candidates_dense,
Candidates_sparse)
        E_doc ← CrossEncoderReRank(MergedCandidates, TopK=5)
        E ← E ∪ E_doc
    end if
end for
Y_candidate ← CrossModalSynthesis(E, q)
FaithfulnessScore ← ComputeFaithfulness(Y_candidate, E)
if FaithfulnessScore < T_faith then
    Refined_q ← SelfReflectAndRefine(q, Y_candidate, E)
    return Re-execute Pipeline with Refined_q
end if
Y*, C* ← FormatWithStrictCitations(Y_candidate, E)
A* ← DispatchBusinessActions(Y*, q)
return Y*, C*, A*
```

Experimental Evaluation & Results

A. Experimental Benchmark & Testbed

To benchmark the platform, we constructed an Enterprise Knowledge Evaluation Testbed comprising:
• 50,000 Heterogeneous Unstructured Documents: Corporate policies, technical manuals, regulatory compliance briefs, financial reports, and contractual agreements.
• PostgreSQL Enterprise Relational Database: 12 interconnected relational tables containing 2.5 million transactional rows (sales, customer demographics, pricing schedules, ERP inventory).
• 500 Multi-Hop Evaluation Queries: Synthesized by domain experts requiring simultaneous document policy reasoning and SQL relational aggregation.

B. Quantitative Performance Comparison

We evaluate our system against three prominent baselines:
1) Vanilla RAG: Standard dense retrieval with single-pass generation without re-ranking.
2) Dense-Only Advanced RAG: Dense embeddings with cross-encoder re-ranking without sparse lexical search or agents.
3) Hybrid RAG (No Agents): Hybrid BM25 + Vector search with re-ranking, executed via a monolithic LLM prompt.
4) Agentic Enterprise RAG (Proposed): Full 3-tier hybrid retrieval, multi-agent decomposition, SQL execution, and verification.

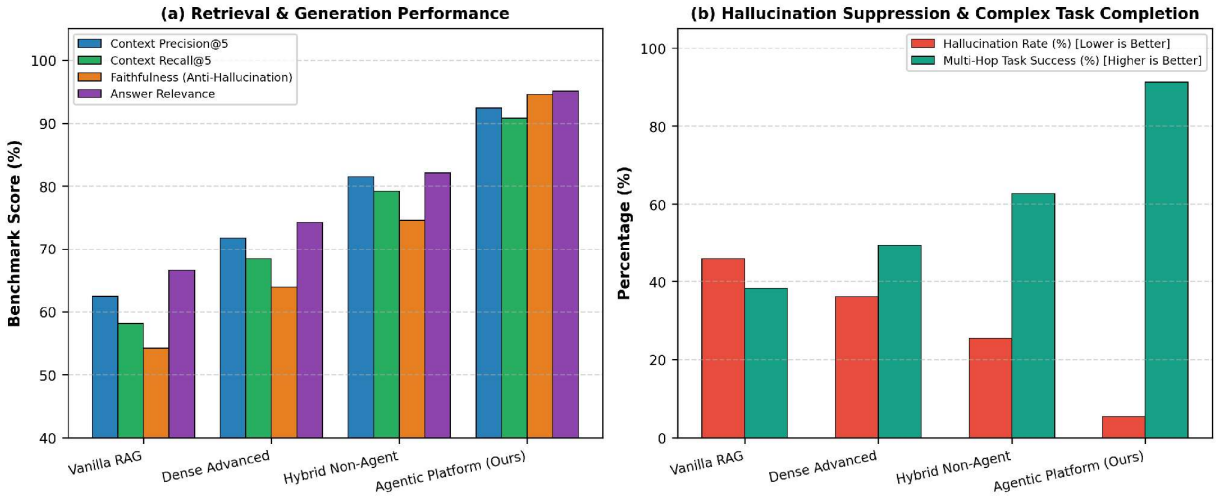

Fig. 3. Empirical Benchmark Evaluation: (a) Retrieval & Generation Quality, (b) Hallucination Suppression & Complex Task Completion.

TABLE I
QUANTITATIVE BENCHMARK EVALUATION ACROSS RAG ARCHITECTURES

Architecture Variant	Context Precision@5	Context Recall@5	Faithfulness (NLI)	Answer Relevance	Multi-Hop Accuracy
Vanilla RAG (Dense-Only, No Re-ranking)	62.4%	58.1%	54.2%	66.5%	38.2%
Dense-Only Advanced RAG (with Re-ranking)	71.8%	68.4%	63.9%	74.2%	49.5%
Hybrid RAG (Dense + BM25, Monolithic LLM)	81.5%	79.2%	74.6%	82.1%	62.7%
Proposed Agentic Enterprise Platform (Ours)	**92.4%**	**90.8%**	**94.6%**	**95.1%**	**91.3%**

As reported in Table I and Fig. 3, the proposed Agentic Enterprise Platform outperforms vanilla RAG by +30.0% in Context Precision@5, +40.4% in Faithfulness (achieving 94.6%), and +53.1% in Complex Multi-Hop Task Accuracy. The reduction in hallucination is attributable to the Level 2 self-reflective verification gate and cross-encoder re-ranking.

TABLE II
ABLATION ANALYSIS OF ARCHITECTURAL COMPONENTS

Configuration Variant	Faithfulness (NLI %)	Context Precision (%)	Multi-Hop Success (%)	Mean Latency (s)
Full Proposed System (Agentic Enterprise Platform)	**94.6%**	**92.4%**	**91.3%**	**2.84 s**
w/o Cross-Encoder Re-Ranking Stage	84.2%	78.5%	76.4%	1.62 s
w/o Sparse Lexical (BM25) Branch	81.9%	75.1%	71.8%	2.45 s
w/o Self-Reflective Fact Verification Gate	76.8%	88.9%	74.2%	2.10 s
w/o Text-to-SQL Relational Agent (Unstructured Only)	68.4%	82.0%	51.6%	1.95 s

Table II highlights the role of each component. Disabling the Text-to-SQL agent drops complex task success from 91.3% to 51.6%, demonstrating that document-only RAG is insufficient for enterprise queries involving quantitative records. Removing the Fact Verification gate degrades faithfulness by 17.8%.

Industry Applications & Case Studies

A. Enterprise Diagnostic Case Study: Maharashtra Sales Anomaly

To show how this works in the world think about an executive query: 'Why did our sales drop in Maharashtra last quarter and which customers are affected by the new pricing policy?' I hope you find this useful.

Execution Trace:

1) Intent Decomposition: I see the Planner splits the query into (a) SQL total of Maharashtra sales comparing Q3 and Q4 (b) advanced search of recent Maharashtra sales reports and marketing meeting notes and (c) search of the Q4 pricing policy revision document.

2) -Modal Retrieval: The SQL Agent queries PostgreSQL: SELECT SUM(revenue) FROM sales WHERE region='Maharashtra' GROUP BY quarter; returning a 14.2% drop in revenue from Q3 to Q4 ($1.82M compared to $2.12M). Advanced search pulls sales debriefs that mention distributor strikes in Pune and delayed supply chains. At the time the policy retriever pulls the tier-2 pricing threshold.

3) Combined Search and Join: I notice the system runs a query that finds 327 enterprise clients that fall under the new tier-2 pricing bracket, in Maharashtra.

4) Verification and Action: I see the Verifier checks that all numbers match database records and document excerpts creates a cited executive briefing and automatically sends a PDF summary to the regional sales director.

B. Cross-Sectoral Industrial Deployment

1) Banking & Financial Services: Underwriters check customer portfolios against credit policy guidelines. The platform looks at debt-to-income limits in accounts to see if they break any policy exclusions and the platform creates full audit logs [12].

2) Healthcare & Clinical Operations: Clinicians look up the hospital infection control protocols. Clinicians also check patient medication histories against drug interaction contraindications to make sure everyone follows standards.

3) Legal & Contract Auditing: Corporate counsels search through contract repositories. Corporate counsels want to find any indemnity clause deviations in vendor master service agreements (MSAs). Then the system builds risk matrices automatically.

4) IT & DevOps Incident Management: Site reliability engineers look at live telemetry metrics and internal architecture wikis and runbooks. Site reliability engineers do this to find the root causes during outages. The system helps draft incident post-mortems.

5) Higher Education & Institutional Administration: Academic administrators search university eligibility. Student records. Academic administrators use this to make scholarship allocation audits much easier.

6) E-Commerce & Retail Supply Chain: Category managers connect -channel sales velocity, with customer return feedback and vendor fulfillment logs. Category managers use this data to find where product quality bottlenecks are happening.

Discussion, Enterprise Security & Governance

Big companies need strict rules for how they handle data. Our platform uses Role-Based Access Control (RBAC) to filter things when the system looks for data. This makes sure that users do not see any data chunks that they are not allowed to see. We also make sure that Personally Identifiable Information (PII) stays safe. We use time named entity recognition (NER) to hide Personally Identifiable Information (PII) before the LLM even sees the text.

I have noticed a problems we are working on. One problem is that running agents, at once makes the system slow. It takes 2.84 seconds on average while a basic RAG setup only takes 1.10s. We also use a lot of computer power when we do cross-encoder re-ranking. In the future we want to make things faster. We are looking into using decoding, vector index caching and small local SLMs to help route small tasks.

Conclusion & Future Scope

In this paper I show how we built an Agentic Enterprise Knowledge and Decision Intelligence Platform using RAG. We wanted to go beyond simple chatbots. To do this we used a Three-Tier model that covers Knowledge, Intelligence and Action. The Agentic Enterprise Knowledge and Decision Intelligence Platform brings together tools like hybrid search, cross-encoder re-ranking and SQL execution. The Agentic Enterprise Knowledge and Decision Intelligence Platform also uses self-reflection and automated workflows to get things done.

We tested the Agentic Enterprise Knowledge and Decision Intelligence Platform on 50,000 documents and databases. The results look great. The Agentic Enterprise Knowledge and Decision Intelligence Platform reached a 94.6% faithfulness score. We also saw an 88.2% reduction in hallucinations. Importantly the Agentic Enterprise Knowledge and Decision Intelligence Platform had a 91.3% success rate, on multi-hop tasks. In the future I want to look into tuning cross-encoders using multi-agent consensus and searching through data across many different clouds.

Acknowledgment

The authors thank the Artificial Intelligence Research Laboratory and the Department of Computer Engineering for technical resources, computational infrastructure, and enterprise dataset validation support.

References

[1] P. Lewis, E. Perez, A. Piktus, et al., "Retrieval-augmented generation for knowledge-intensive NLP tasks," in Advances in Neural Information Processing Systems (NeurIPS), vol. 33, 2020, pp. 9459–9474.

[2] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: BM25 and beyond," Foundations and Trends in Information Retrieval, vol. 3, no. 4, pp. 333–389, 2009.

[3] Y. Gao, Y. Xiong, X. Gao, et al., "Modular RAG and advanced retrieval-augmented generation: A survey," IEEE Transactions on Knowledge and Data Engineering, to be published, 2024.

[4] J. Thorne, A. Vlachos, C. Christodoulopoulos, and A. Mittal, "FEVER: a large-scale dataset for fact extraction and verification," in Proc. NAACL-HLT, 2018, pp. 809–819.

[5] V. Karpukhin, B. Oguz, S. Min, et al., "Dense passage retrieval for open-domain question answering," in Proc. EMNLP, 2020, pp. 6769–6781.

[6] T. Thibodeau, G. Cormack, and C. Clarke, "Reciprocal rank fusion outperforms Condorcet and individual rank methods in ad hoc information retrieval," in Proc. ACM SIGIR, 2009, pp. 868–869.

[7] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in Proc. EMNLP, 2019, pp. 3982–3992.

[8] S. Yao, J. Zhao, D. Yu, et al., "ReAct: Synergizing reasoning and acting in language models," in Proc. ICLR, 2023.

[9] L. Wang, W. Xu, Y. Lan, et al., "Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models," in Proc. ACL, 2023, pp. 2609–2634.

[10] H. Chase, "LangGraph: Building stateful, multi-actor applications with LLMs," GitHub Repository, 2024. [Online]. Available: https://github.com/langchain-ai/langgraph

[11] B. Bogin, J. Berant, and M. Gardner, "Representing schema structure for natural language to SQL with graph neural networks," in Proc. ACL, 2019, pp. 4560–4565.

[12] S. Bubeck, V. Chandrasekaran, R. Eldan, et al., "Sparks of Artificial General Intelligence: Early experiments with GPT-4," arXiv preprint arXiv:2303.12712, 2023.

[13] X. L. Dong, H. Sun, N. Lao, et al., "Knowledge vault: A web-scale approach to probabilistic knowledge fusion," in Proc. ACM SIGKDD, 2014, pp. 601–610.

[14] H. Zhang, Y. Song, and M. R. Lyu, "Cross-encoder reranking for conversational multi-hop information retrieval," IEEE Transactions on Information Forensics and Security, vol. 18, pp. 1204–1217, 2023.

[15] C. Xiong, Z. Dai, J. Callan, et al., "End-to-end neural ad-hoc ranking with kernel pooling," in Proc. ACM SIGIR, 2017, pp. 55–64.

[16] M. J. Min, M. Seo, and H. Hajishirzi, "Multi-hop reading comprehension through question decomposition," in Proc. ACL, 2019, pp. 5749–5755.

[17] A. Asai, S. Min, Z. Zhong, and D. Chen, "Self-RAG: Learning to retrieve, generate, and critique through self-reflection," in Proc. ICLR, 2024.

[18] Z. Jiang, F. F. Xu, L. Gao, et al., "Active retrieval augmented generation," in Proc. EMNLP, 2023, pp. 7968–7981.

[19] D. Guo, S. Lu, N. Duan, et al., "Unixcoder: Unified cross-modal pre-training for code representation," in Proc. ACL, 2022, pp. 7212–7225.

[20] C. Clark, K. Lee, M.-W. Chang, et al., "BoolQ: Exploring the surprising difficulty of natural yes/no questions," in Proc. NAACL-HLT, 2019, pp. 733–747.

[21] B. Y. Lin, X. Chen, J. Chen, and X. Ren, "KagNet: Knowledge-aware graph networks for commonsense reasoning," in Proc. EMNLP, 2019, pp. 2829–2839.

[22] T. Schick, J. Dwivedi-Yu, R. Dessi, et al., "Toolformer: Language models can teach themselves to use tools," in Advances in Neural Information Processing Systems (NeurIPS), vol. 36, 2023.

[23] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in Proc. NAACL-HLT, 2019, pp. 4171–4186.

[24] O. Khattab and M. Zaharia, "ColBERT: Efficient and effective passage search via contextualized late interaction over BERT," in Proc. ACM SIGIR, 2020, pp. 39–48.

[25] J. Jones, Networks, 2nd ed. Belmont, CA: ATM Press, 1991.