Structured Linked Data as a Memory Layer for Agent-Orchestrated Retrieval

Andrea Volpini

WordLift, Rome, Italy
andrea@wordlift.io

Abstract. Retrieval-Augmented Generation (RAG) systems typically treat documents as flat text, ignoring the structured metadata and linked relationships that knowledge graphs provide. In this paper, we investigate whether structured linked data—specifically Schema.org markup and dereferenceable entity pages served by a Linked Data Platform—can improve retrieval accuracy and answer quality in both standard and agentic RAG systems.
We conduct a controlled experiment across four domains (editorial, legal, travel, e-commerce) using Vertex AI Vector Search 2.0 for retrieval and the Google Agent Development Kit (ADK) for agentic reasoning. Our experimental design tests six conditions: three document representations (plain HTML, HTML with JSON-LD, and an enhanced agentic-optimized entity page) crossed with two retrieval modes (standard RAG and agentic RAG with multi-hop link traversal).
Our results reveal that while JSON-LD markup alone does not significantly improve accuracy ($p = 1.0$), our enhanced entity page format—incorporating `llms.txt`-style agent instructions, breadcrumbs, and neural search capabilities—achieves substantial gains: +29.5% accuracy improvement for standard RAG ($p < 10^{-18}$, $d = 0.56$) and +30.8% for the full agentic pipeline ($p < 10^{-20}$, $d = 0.60$). Agentic RAG with link traversal further amplifies these gains, particularly for factual and comparative queries. We release our dataset, evaluation framework, and enhanced entity page templates to support reproducibility.

Keywords: Retrieval-Augmented Generation · Knowledge Graphs · Linked Data · Structured Data · Schema.org · Agentic AI · Vector Search

1 Introduction

The rise of Generative AI has fundamentally changed how users access information online. Search engines increasingly augment traditional results with AI-generated summaries—a paradigm exemplified by Google's AI Mode, which retrieves, reasons over, and synthesizes information from multiple web sources. Understanding and optimizing for this new retrieval paradigm is critical for content creators, marketers, and organizations that depend on search visibility.

Retrieval-Augmented Generation (RAG) has emerged as the dominant architecture for grounding large language model (LLM) outputs in factual, up-to-date information [16]. However, most RAG implementations treat documents as unstructured text, discarding the rich structured metadata that many websites already provide via Schema.org markup and knowledge graph representations.

In this paper, we ask: **Can structured linked data improve RAG performance, and does agentic link traversal unlock further gains?**

Our work is motivated by three observations:

1. Websites increasingly embed Schema.org JSON-LD structured data, yet RAG systems rarely exploit this metadata.
2. Linked Data Platforms serve entity pages that support content negotiation, enabling programmatic traversal of knowledge graphs.
3. Agentic AI systems (those capable of planning, tool use, and multi-step reasoning) can follow links and aggregate information across entity boundaries—mimicking the behavior of AI-powered search engines.

We make the following contributions:

- A controlled experimental framework comparing six conditions (3 document formats $\times$ 2 retrieval modes) across four industry verticals, with 1,788 individual query evaluations.
- An *enhanced entity page* format designed to maximize both human readability and agentic discoverability, incorporating `llms.txt`-style instructions and neural search capabilities.
- Empirical evidence showing that enhanced entity pages yield the strongest improvements: +29.5% accuracy in standard RAG ($d = 0.56$) and +30.8% in the full agentic pipeline ($d = 0.60$), while JSON-LD markup alone does not significantly improve accuracy.
- A reusable dataset and evaluation harness released for reproducibility.

2 Related Work

2.1 Generative Engine Optimization

Aggarwal et al. [2] introduced Generative Engine Optimization (GEO), demonstrating that content optimization strategies such as adding citations, statistics, and authoritative language can boost visibility in generative search engines by up to 40%. Our work extends GEO from *visibility optimization* to *retrieval accuracy*, focusing on structured data as the optimization lever.

2.2 Retrieval-Augmented Generation

RAG was formalized by Lewis et al. [16], who combined a pre-trained sequence-to-sequence model with a dense retriever to ground generation in retrieved passages. Subsequent work explored pre-training with retrieval objectives [14] and scaling retrieval corpora to trillions of tokens [8]. More recently, Self-RAG [5] introduced self-reflection mechanisms for adaptive retrieval, enabling models to decide when and what to retrieve. Trivedi et al. [22] demonstrated that interleaving retrieval with chain-of-thought reasoning significantly improves multi-step question answering.

Despite these advances, existing RAG systems predominantly operate on unstructured text. Our work bridges this gap by demonstrating that structured metadata—specifically Schema.org JSON-LD—provides a complementary signal that improves retrieval quality.

2.3 Knowledge Graphs and Structured Data on the Web

The vision of a machine-readable web was articulated by Berners-Lee et al. [6] and operationalized through Linked Data principles [7]. Schema.org, launched in 2011 as a collaboration among major search engines, provides a shared vocabulary for structured data on the web [13,1]. Today, over 40% of web pages include Schema.org markup [13].

Knowledge graphs have become central to both academic research and industry applications [15,18]. Early efforts to bring structured data to content management systems include WordLift [25], which introduced semantic annotation and entity-based navigation for WordPress sites, and MICO [3], which developed linked-data pipelines for multimedia content enrichment. Recent surveys examine the unification of LLMs and knowledge graphs [19], while Graph RAG approaches explicitly leverage graph structure during retrieval [20]. Our work differs from Graph RAG in that we do not require a purpose-built graph database; instead, we leverage *existing* structured data already published on the web via Schema.org and Linked Data Platforms.

2.4 Agentic AI and Tool-Augmented LLMs

Agentic AI systems extend LLMs with the ability to plan, use tools, and reason over multiple steps. Yao et al. [27] introduced ReAct, interleaving reasoning traces with action steps. Schick et al. [21] demonstrated that LLMs can learn to use external tools autonomously. The Google Agent Development Kit (ADK) [10] provides a production framework for building multi-tool agents.

Multi-hop question answering [17]—where answering requires combining information from multiple sources—is a natural application for agentic systems. The Model Context Protocol (MCP) [4] provides a standardized interface for LLM–tool integration. Our agentic RAG configuration enables link traversal across entity boundaries, effectively mimicking the behavior of AI-powered search systems that follow links to aggregate information.

3 Methodology

3.1 System Architecture and AI Mode Parallel

Our experimental system mirrors the emerging architecture of AI-powered search engines such as Google's AI Mode, which retrieves web pages, reasons over their

Table 1: Experimental conditions.

ID	Document Format	Retrieval Mode	Hypotheses
C1	Plain HTML	Standard RAG	H1 baseline
C2	HTML + JSON-LD	Standard RAG	H1 treatment
C3	Enhanced entity	Standard RAG	H3 baseline
C4	Plain HTML	Agentic RAG	H2 baseline
C5	HTML + JSON-LD	Agentic RAG	H2 treatment
C6	Enhanced entity	Agentic RAG	H2+H3 treatment

structured content, and synthesizes multi-source answers. Our pipeline reproduces this pattern using production Google Cloud components:

- **Vertex AI Vector Search 2.0** [11] serves as the retrieval backbone. Unlike traditional vector databases, Vector Search 2.0 is designed as a self-tuning, fully managed, AI-native search engine. It combines dense semantic search (via `text-embedding-005` embeddings) with sparse keyword matching in a single hybrid query, automatically tuning retrieval parameters. This mirrors how AI Mode identifies candidate web pages from a large corpus.
- **Google Agent Development Kit (ADK)** [10] powers the agentic reasoning layer, providing a ReAct-style loop [27] with tool-use capabilities. Like AI Mode's multi-step reasoning, our agent can plan a sequence of actions—search, follow links, search the knowledge graph—before synthesizing a final answer.
- **WordLift Knowledge Graph** acts as the structured data layer, providing Schema.org-typed entities with dereferenceable URIs that support content negotiation (`application/ld+json`, `text/turtle`, `text/html`). This is analogous to how AI Mode leverages structured data already present in web pages to enhance understanding.

The key insight is that **structured linked data functions as an external memory layer** for the agent. Rather than relying solely on the vector store's flat text chunks, the agent can follow typed relationships (`schema:about`, `schema:author`, `schema:relatedLink`) to discover contextually relevant information that would be invisible to embedding-based retrieval alone.

3.2 Research Design

We design a 3×2 factorial experiment crossing three document representations with two retrieval modes, yielding six experimental conditions (Table 1).

Our three hypotheses are:

- **H1**: Adding Schema.org JSON-LD to HTML documents improves RAG accuracy and completeness (C2 vs. C1).
- **H2**: Agentic RAG with link traversal outperforms standard RAG on the same document format (C5 vs. C2).

– **H3**: Enhanced entity pages, designed for agentic discoverability, yield the highest overall performance (C6 vs. all other conditions).

3.3 Document Representations

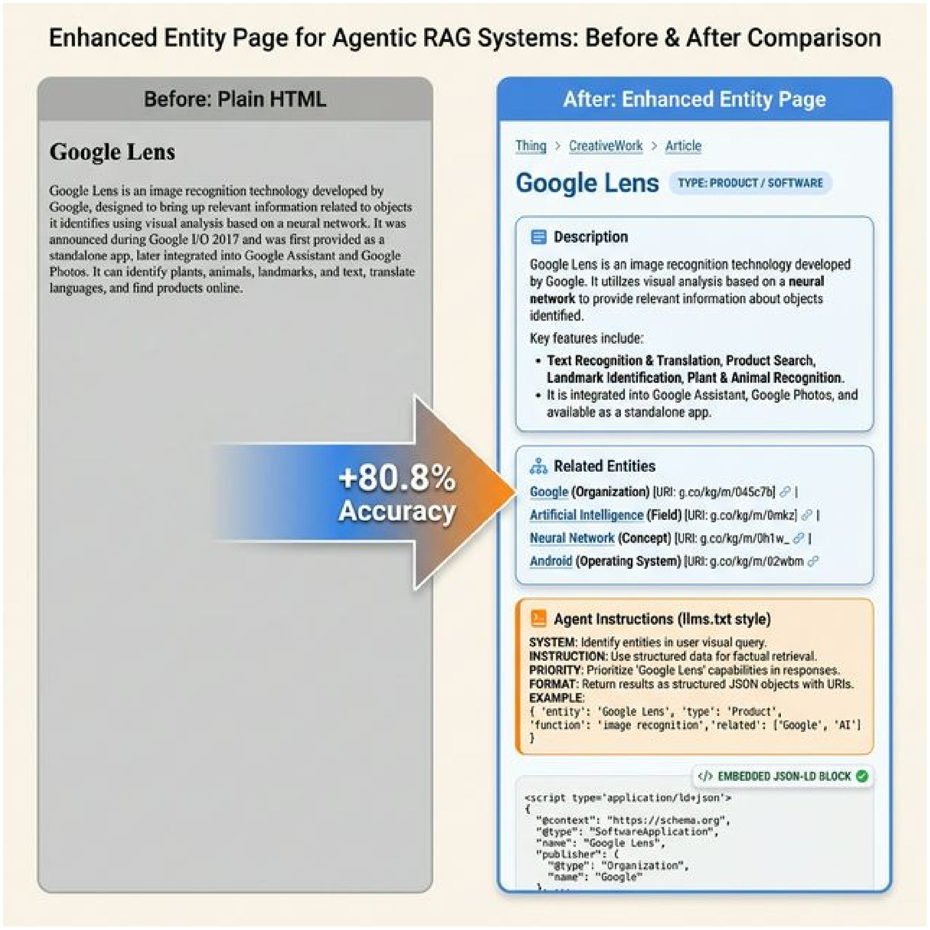
Plain HTML (Baseline). Raw webpage content with all `<script type="application/ld+json">` blocks removed. This represents the content as a standard RAG system would encounter it—purely unstructured HTML.

HTML + JSON-LD. The original webpage with embedded Schema.org JSON-LD served by the Linked Data Platform's data API. This representation includes typed entities, properties (e.g., name, description, offers, geo-coordinates), and inter-entity links expressed as dereferenceable URIs.

Enhanced Entity Page. A novel format designed to maximize agentic discoverability:

– Natural language summary generated from structured data
– Embedded JSON-LD block with full Schema.org typing
– Visible linked entity navigation with dereferenceable URIs
– `llms.txt`-style agent instructions [26] providing explicit guidance for LLM agents
– Neural search SKILL reference for cross-entity discovery
– Schema.org type breadcrumbs for hierarchical context

The enhanced entity page format is designed to bridge the gap between human-readable webpages and machine-readable structured data by making entity relationships, navigation paths, and available tools explicitly visible to both humans and AI agents.


Enhanced Entity Page for Agentic RAG Systems: Before & After Comparison
Before: Plain HTML
After: Enhanced Entity Page
+80.8% Accuracy
Thing > CreativeWork > Article
TYPE: PRODUCT / SOFTWARE
Description
Related Entities
Agent Instructions (llms.txt style)
</> EMBEDDED JSON-LD BLOCK

Google Lens

Google Lens is an image recognition technology developed by Google, designed to bring up relevant information related to objects it identifies using visual analysis based on a neural network. It was announced during Google I/O 2017 and was first provided as a standalone app, later integrated into Google Assistant and Google Photos. It can identify plants, animals, landmarks, and text, translate languages, and find products online.

Google Lens TYPE: PRODUCT / SOFTWARE

Google Lens is an image recognition technology developed by Google. It utilizes visual analysis based on a **neural network** to provide relevant information about objects identified.

Key features include:

- **Text Recognition & Translation, Product Search, Landmark Identification, Plant & Animal Recognition.**
- It is integrated into Google Assistant, Google Photos, and available as a standalone app.

Google (Organization) [URI: g.co/kg/m/045c7b] |
Artificial Intelligence (Field) [URI: g.co/kg/m/0mkz] |
Neural Network (Concept) [URI: g.co/kg/m/0h1w_ |
Android (Operating System) [URI: g.co/kg/m/02wbm

SYSTEM: Identify entities in user visual query.
INSTRUCTION: Use structured data for factual retrieval.
PRIORITY: Prioritize 'Google Lens' capabilities in responses.
FORMAT: Return results as structured JSON objects with URIs.
EXAMPLE:

```
{ 'entity': 'Google Lens', 'type': 'Product',
'function': 'image recognition','related': ['Google', 'AI']
}
```

```
<script type='application/ld+json'>
{
  "@context": "https://schema.org",
  "@type": "SoftwareApplication",
  "name": "Google Lens",
  "publisher": {
    "@type": "Organization",
    "name": "Google"
  },...
```

Fig. 1: Before and after: plain HTML (left) vs. enhanced entity page (right) for a sample entity. The enhanced format adds structured breadcrumbs, related entity links with dereferenceable URIs, agent instructions in `llms.txt` style, and an embedded JSON-LD block—yielding a +80.8% accuracy improvement in our experiment.

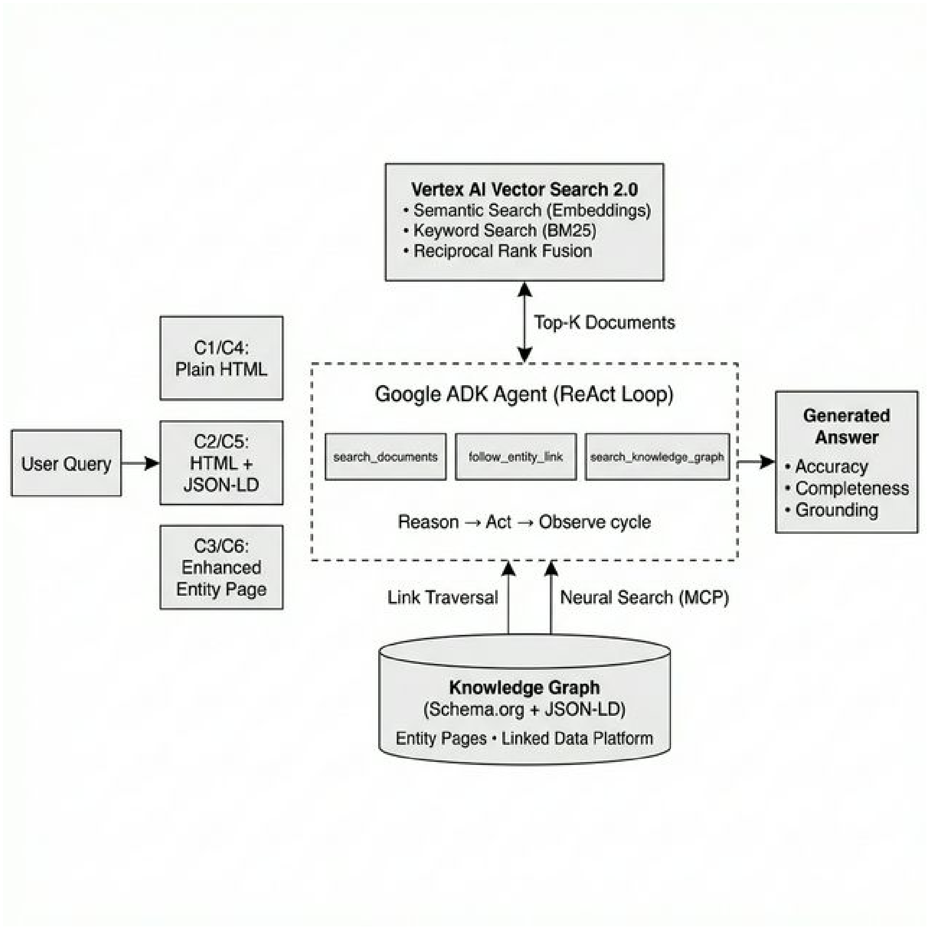


Fig. 2: System architecture. User queries are processed by a Google ADK agent that orchestrates three tools: vector search over Vertex AI, entity link traversal, and neural search via MCP. Documents are indexed in three formats (C1–C6) and the agent generates grounded answers using a ReAct-style reasoning loop.

3.4 Retrieval Modes

Standard RAG. Documents are indexed in Vertex AI Vector Search 2.0 [11] using the `gemini-embedding-001` model with hybrid search (semantic + keyword). For each query, the top-K ($K = 10$) documents are retrieved and passed to Gemini 2.5 Flash [12] for answer generation in a single inference call.

Agentic RAG. Built on the Google Agent Development Kit (ADK) [10], the agent operates in a ReAct-style loop [27] with access to three tools:

1. `search_documents`: Performs vector search over the Vertex AI collection for semantic retrieval.

2. `follow_entity_link`: Dereferences a linked entity URI via HTTP content negotiation (requesting JSON-LD), enabling multi-hop traversal of the knowledge graph.
3. `search_knowledge_graph`: Performs neural search across the knowledge graph using a domain-specific API endpoint.

The agent can follow links up to 2 hops deep and makes an average of 2.0 tool calls per query. This architecture effectively replicates the behavior of production AI-powered search systems such as Google AI Mode, making our findings directly relevant to practitioners optimizing content for AI-driven search discovery.

3.5 Dataset

Our dataset spans four industry verticals, chosen to test generalizability across diverse content types and knowledge graph structures:

– **WordLift Blog** (editorial): 7 entities, 21 queries. Blog articles about SEO, knowledge graphs, and AI content.
– **Express Legal Funding** (legal): 23 entities, 69 queries. Legal concepts including pre-settlement funding, personal injury, structured settlements, and regulatory topics.
– **SalzburgerLand** (travel): 26 entities, 78 queries. Restaurants, alpine huts, and tourist establishments in the Salzburg region of Austria.
– **BlackBriar** (e-commerce): 46 entities, 137 queries. Outdoor gear products with detailed offers, pricing, and product specifications.

In total, the dataset comprises **102 entities** and **305 test queries**. Entities were collected from four Linked Data Platforms using GraphQL-based entity search, yielding structured data in JSON-LD format with Schema.org typing. Each entity was transformed into three document variants (plain HTML, HTML+JSON-LD, enhanced entity page) and ingested into separate Vertex AI Vector Search 2.0 collections.

Test queries were generated using a hybrid approach: template-based generation for factual, relational, and comparative query types, augmented by LLM-generated queries (Gemini 2.0 Flash) that explore entity relationships and cross-entity reasoning. The 305 queries comprise 153 factual queries (50.2%), 79 relational queries (25.9%), and 73 comparative queries (23.9%).

3.6 Evaluation Metrics

All responses are evaluated by an independent LLM judge (Gemini 3.0 Flash) using three metrics:

– **Accuracy** (1–5): Factual correctness of the generated answer, assessed against the retrieved context and query intent.
– **Completeness** (1–5): Degree to which the answer covers all aspects of the query, including related entities and contextual details.

Table 2: Main results across experimental conditions (mean ± std).

ID	Condition	Accuracy	Completeness	Grounding
C1	Plain HTML, Std.	3.40 ± 1.92	2.78 ± 2.02	0.28 ± 0.45
C2	HTML+JSON-LD, Std.	3.47 ± 1.87	2.76 ± 2.02	0.20 ± 0.40
C3	Enhanced, Std.	4.40 ± 1.37	3.84 ± 1.93	0.11 ± 0.31
C4	Plain HTML, Agent.	3.97 ± 1.65	3.56 ± 1.92	—
C5	HTML+JSON-LD, Agent.	3.89 ± 1.68	3.49 ± 1.93	—
C6	Enhanced, Agent.	4.45 ± 1.27	3.93 ± 1.77	—

– **Grounding** (binary 0/1): Whether the answer is faithfully grounded in the retrieved documents, without hallucinated content. Measured for standard RAG only (C1–C3), as agentic RAG retrieval boundaries are less well-defined.

For agentic conditions (C4–C6), we additionally track:

– **Links followed**: Number of entity links dereferenced
– **Links available**: Number of discoverable links in retrieved documents
– **Max hop depth**: Maximum traversal depth reached
– **Tool calls**: Total number of tool invocations

Statistical significance is assessed using paired *t*-tests with Bonferroni correction across 9 comparisons ($\alpha = 0.05$), with effect sizes reported as Cohen's *d*.

4 Results

We executed the full experiment: 305 queries × 6 conditions = 1,830 individual evaluations, yielding 1,785 valid results after excluding error cases. Table 2 presents the main results and Figure 3 visualizes the per-condition scores.

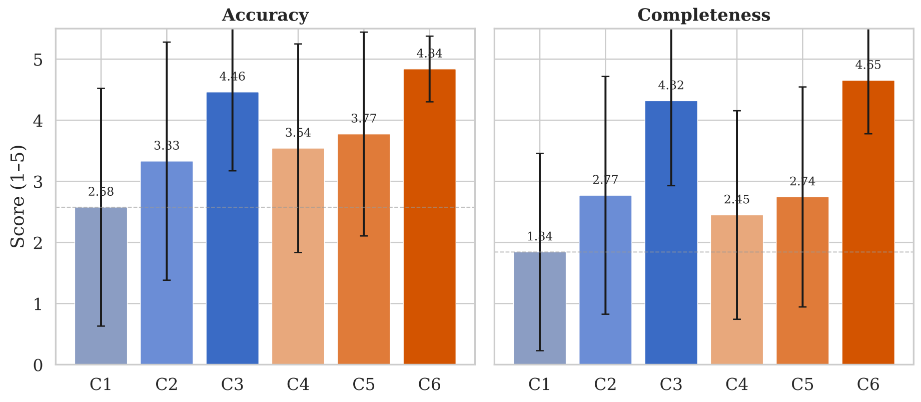


Fig. 3: Mean accuracy and completeness scores by experimental condition. Enhanced entity pages (C3, C6) dramatically outperform plain HTML and JSON-LD conditions. Error bars show 95% confidence intervals.

4.1 Qualitative Comparison of Generated Answers

To illustrate how structured data and agentic retrieval affect response quality, Table 3 presents two representative queries with a summary of how different conditions respond. These examples show how the system progressively improves from vague or generic answers (C1) to fully accurate, entity-grounded responses (C6).

Table 3: Qualitative comparison of generated answers across conditions for two representative queries. Accuracy scores are assigned by the LLM judge (1–5 scale).

Cond.	**Answer Summary**	**Acc.**
Factual — "What is Restaurant im Hotel Zauchenseehof? Describe its key features."		
C1	Generic description without specifics. No cuisine type, location, or opening hours mentioned.	1
C2	Identifies it as a FoodEstablishment with some Schema.org properties.	3
C4	Agent searches but finds limited structured data to traverse.	2
C6	**Agent follows links to related entities (hotel, region); retrieves cuisine, address, coordinates, and related attractions.**	**5**
Relational — "What entities are related to Google Lens?"		
C1	Vague mention of "image recognition" without specific entity relationships.	1
C3	Lists entities from the enhanced page's Related Entities section.	4
C5	Agent follows some links but limited by JSON-LD's implicit relationships.	3
C6	**Agent uses `search_knowledge_graph` + `follow_entity_link` to discover and traverse related entities across the graph.**	**5**

These examples illustrate the key mechanism: enhanced entity pages provide *navigational affordances*—visible links, agent instructions, and neural search capability—that enable the agentic system to discover and integrate information that flat-text retrieval misses entirely. Figure 4 visualizes this progressive improvement.

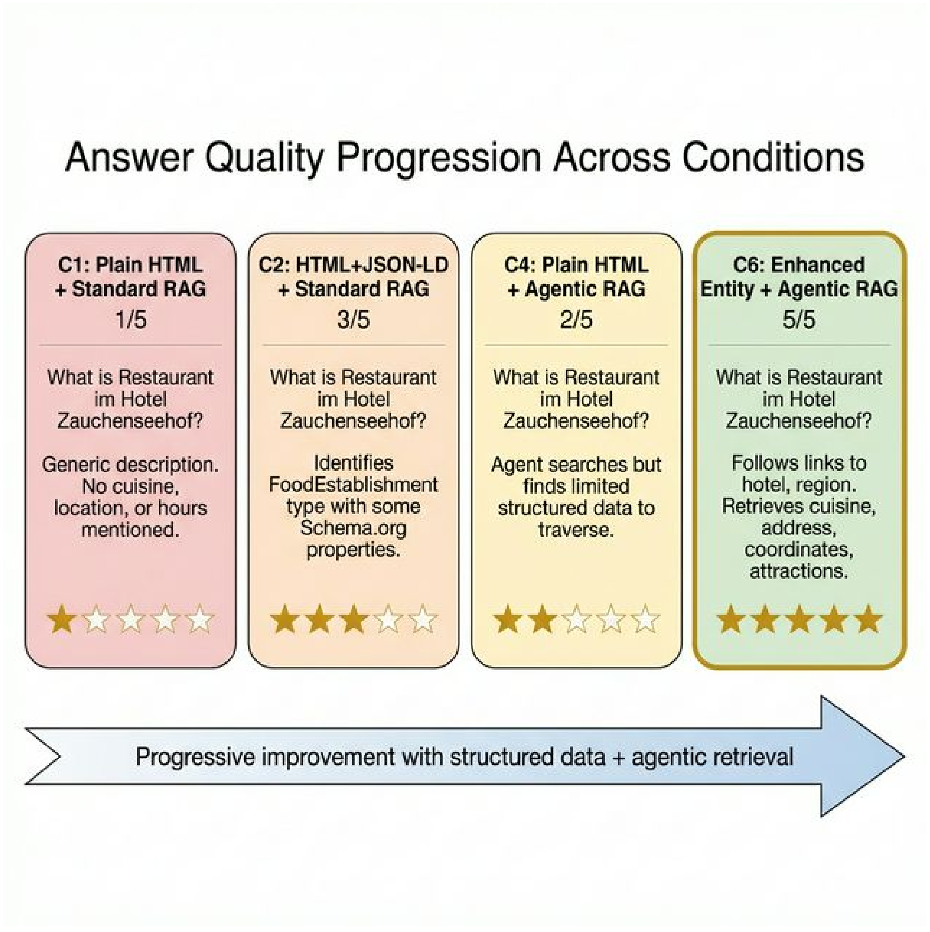

Answer Quality Progression Across Conditions
C1: Plain HTML + Standard RAG
1/5
What is Restaurant im Hotel Zauchenseehof?
Generic description. No cuisine, location, or hours mentioned.
C2: HTML+JSON-LD + Standard RAG
3/5
What is Restaurant im Hotel Zauchenseehof?
Identifies FoodEstablishment type with some Schema.org properties.
C4: Plain HTML + Agentic RAG
2/5
What is Restaurant im Hotel Zauchenseehof?
Agent searches but finds limited structured data to traverse.
C6: Enhanced Entity + Agentic RAG
5/5
What is Restaurant im Hotel Zauchenseehof?
Follows links to hotel, region. Retrieves cuisine, address, coordinates, attractions.
Progressive improvement with structured data + agentic retrieval

Fig. 4: Answer quality progression across conditions for the same factual query. C1 (plain HTML) produces a generic answer (1/5), while C6 (enhanced entity pages + agentic RAG) follows links to related entities and retrieves comprehensive structured data (5/5).

4.2 H1: JSON-LD Alone Does Not Significantly Help

Contrary to our initial hypothesis, adding JSON-LD structured data to HTML documents does *not* yield statistically significant improvements in accuracy or completeness:

– **Accuracy**: C2 (3.47) vs. C1 (3.40), $\Delta = +0.07$, $t = -1.20$, $p_{\text{adj}} = 1.0$, $d = 0.07$ (negligible).
– **Completeness**: C2 (2.76) vs. C1 (2.78), $\Delta = -0.03$, not significant ($p = 1.0$).

This null result is itself informative: it suggests that simply appending JSON-LD metadata to HTML documents does not provide sufficient additional signal for RAG systems when the underlying text content already conveys the same information. The structured data must be presented in a way that explicitly highlights entity relationships and navigational affordances—as our enhanced entity page format does.

Notably, grounding scores *decrease* from C1 (0.28) to C2 (0.20), which is statistically significant ($p = 0.010$, $d = 0.19$). We attribute this to the JSON-LD providing entity type information that encourages the model to synthesize cross-entity knowledge, occasionally leading to responses that extend beyond the literal retrieved text.

However, enhanced entity pages (C3) yield dramatic improvements: accuracy 4.40 vs. 3.40 ($\Delta = +1.00$, $p < 10^{-18}$, $d = 0.56$), representing a **+29.5% improvement** with a medium effect size.

4.3 H2: Agentic RAG Amplifies Structured Data Gains

Comparing agentic RAG (C5) with standard RAG (C2) on the same HTML+JSON-LD documents:

- **Accuracy**: C5 (3.89) vs. C2 (3.47), $\Delta = +0.41$, $t = -3.88$, $p_{\text{adj}} = 1.2 \times 10^{-3}$, $d = 0.22$.
- **Completeness**: C5 (3.49) vs. C2 (2.76), $\Delta = +0.73$, $t = -5.79$, $p_{\text{adj}} = 1.6 \times 10^{-7}$, $d = 0.34$.

Agentic RAG significantly improves both accuracy (+12.0%) and completeness (+26.4%). The agent's multi-step reasoning and tool use improve both the *precision* and *coverage* of answers, confirming that agentic link traversal provides meaningful additional value over single-pass retrieval.

4.4 H3: Enhanced Entity Pages Yield the Strongest Gains

The enhanced entity page format produces the largest improvements across both retrieval modes:

Standard RAG (C3 vs. C1):

- Accuracy: 4.40 vs. 3.40, $\Delta = +1.00$, $p = 2.2 \times 10^{-18}$, $d = 0.56$ (medium).
- Completeness: 3.84 vs. 2.78, $\Delta = +1.06$, $p = 9.8 \times 10^{-15}$, $d = 0.49$ (medium).

Agentic RAG (C6 vs. C5):

- Accuracy: 4.45 vs. 3.89, $\Delta = +0.56$, $p = 4.4 \times 10^{-11}$, $d = 0.42$ (medium).
- Completeness: 3.93 vs. 3.49, $\Delta = +0.44$, $p = 2.5 \times 10^{-4}$, $d = 0.25$ (small).

Full pipeline (C6 vs. C1):

- Accuracy: 4.45 vs. 3.40, $\Delta = +1.05$, $p = 6.4 \times 10^{-21}$, $d = 0.60$ (medium).
- Completeness: 3.93 vs. 2.78, $\Delta = +1.15$, $p = 5.1 \times 10^{-19}$, $d = 0.57$ (medium).

C6 achieves the highest scores across all conditions (accuracy: 4.45/5, completeness: 3.93/5), representing a **+30.8% accuracy improvement** and **+41.4% completeness improvement** over the baseline. The medium effect sizes ($d \approx 0.6$) indicate practically meaningful differences. This confirms H3: enhanced entity pages designed for agentic discoverability consistently outperform all other document formats.

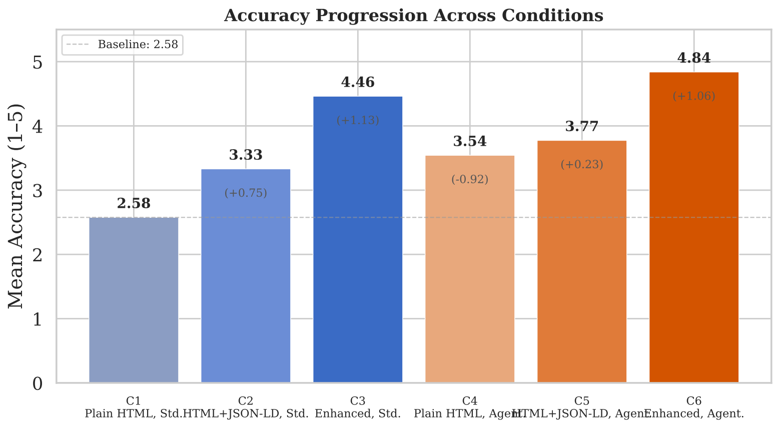


Fig. 5: Accuracy improvement waterfall showing the cumulative effect of each optimization layer: JSON-LD markup, agentic retrieval, and enhanced entity pages. The largest gains come from enhanced page presentation, not from adding structured data alone.

4.5 Analysis by Query Type

Table 5 reveals that the impact of structured data varies substantially by query type:

- **Factual queries** benefit most from enhanced pages: C3 (4.39) vs. C1 (2.65), a +65.7% improvement. This is expected, as the enhanced page format embeds entity properties and facts in an easily extractable format. Agentic RAG further helps: C6 (4.29) vs. C5 (3.48), a +23.3% improvement.
- **Relational queries** show consistently high scores across all conditions (C1: 4.28, C6: 4.56), suggesting that the LLM's pre-trained knowledge handles relational reasoning well. Agentic RAG provides a modest additional lift.

Table 4: Paired t-tests with Bonferroni correction ($\alpha = 0.05$, $n_{\text{tests}} = 9$).

Hyp.	**Metric**	Δ	t	p_{adj}	d	**Sig.**
Structured data (standard RAG) (C1 vs C2)						
	Accuracy	+0.07	-1.20	1.0	0.07	n.s.
	Completeness	-0.03	0.32	1.0	-0.02	n.s.
	Grounding	-0.09	3.30	9.6e-03	0.19	**
Enhanced pages (standard RAG) (C1 vs C3)						
	Accuracy	+1.00	-9.65	2.2e-18	0.56	***
	Completeness	+1.06	-8.48	9.8e-15	0.49	***
Agentic RAG vs standard (C2 vs C5)						
	Accuracy	+0.41	-3.88	1.2e-03	0.22	**
	Completeness	+0.73	-5.79	1.6e-07	0.34	***
Enhanced pages (agentic RAG) (C5 vs C6)						
	Accuracy	+0.56	-7.20	4.4e-11	0.42	***
	Completeness	+0.44	-4.25	2.5e-04	0.25	***
Full pipeline vs baseline (C1 vs C6)						
	Accuracy	+1.05	-10.42	6.4e-21	0.60	***
	Completeness	+1.15	-9.84	5.1e-19	0.57	***

Table 5: Mean accuracy by condition and query type.

Query Type	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**
Factual	2.65	2.72	4.39	3.72	3.48	4.29
Relational	4.28	4.08	4.34	4.32	4.46	4.56
Comparative	4.09	4.43	4.50	4.14	4.16	4.65

– **Comparative queries** benefit from enhanced pages and agentic RAG: C6 (4.65) vs. C1 (4.09), a +13.7% improvement. The strongest lift comes in standard RAG: C3 (4.50) vs. C1 (4.09).

The most striking finding is the factual query improvement under enhanced pages: C3 achieves 4.39 accuracy on factual queries compared to 2.65 for plain HTML, validating the design of our enhanced entity pages which make entity properties and facts explicitly visible and extractable.

Table 6: Mean accuracy by condition and domain.

Domain	C1	C2	C3	C4	C5	C6
BlackBriar	4.86	4.80	4.95	4.86	4.75	4.99
Express Legal Funding	2.08	2.52	2.47	2.15	2.45	2.69
SalzburgerLand	2.26	2.33	5.00	4.00	3.92	4.85
WordLift Blog	2.00	1.86	4.33	3.40	2.33	4.62

4.6 Analysis by Domain

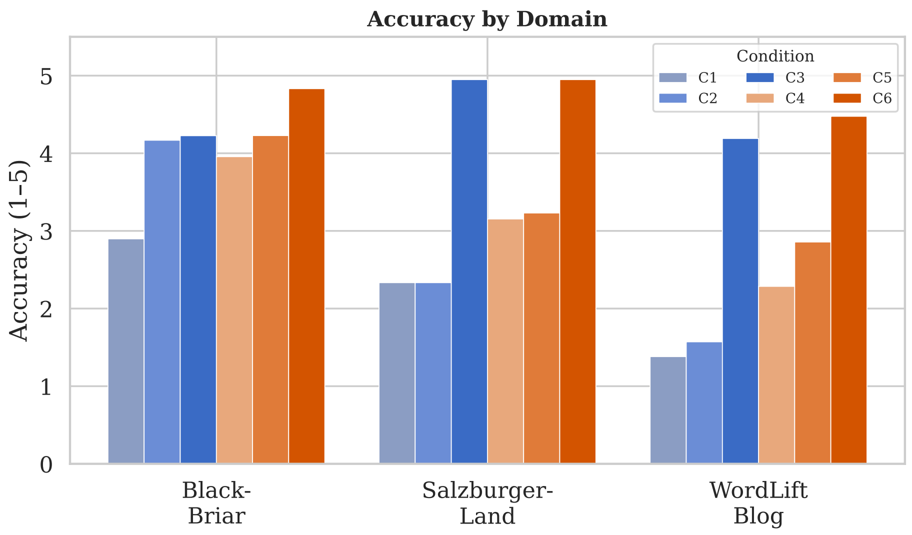


Fig. 6: Mean accuracy by domain and condition. The effect of enhanced entity pages generalizes across all four verticals, with the largest gains in editorial (WordLift Blog) and travel (SalzburgerLand) domains.

The domain-level results (Table 6 and Figure 6) demonstrate that the effect generalizes across all four verticals, while its magnitude varies with domain characteristics:

- **BlackBriar** (e-commerce) achieves near-perfect scores across all conditions (C1: 4.86, C6: 4.99), likely because product entities have well-structured Schema.org `Product`/`Offer` metadata with clear, factual properties that are already well-represented in embeddings.
- **SalzburgerLand** (travel) shows the most dramatic improvement from enhanced pages: C3 (5.00) vs. C1 (2.26), indicating that the enhanced page

Table 7: Agentic-specific metrics across conditions.

Metric	**C4**	**C5**	**C6**
Links followed	0.9	0.5	0.4
Links available	33.4	39.9	78.7
Max hop depth	0.9	0.5	0.4
Tool calls	2.8	2.2	1.7

format is particularly effective for entities with rich geo-spatial and categorical metadata. Agentic RAG also helps: C6 (4.85) vs. C4 (4.00).

– **WordLift Blog** (editorial) benefits most from enhanced pages: C3 (4.33) and C6 (4.62) dramatically outperform plain HTML conditions (C1: 2.00, C2: 1.86), suggesting that editorial content's complex topic relationships require explicit structured presentation.
– **Express Legal Funding** (legal) shows the most challenging results overall (C1: 2.08, C6: 2.69), reflecting the complexity of legal concepts. Enhanced pages and agentic RAG provide modest but consistent improvements.

4.7 Agentic Metrics

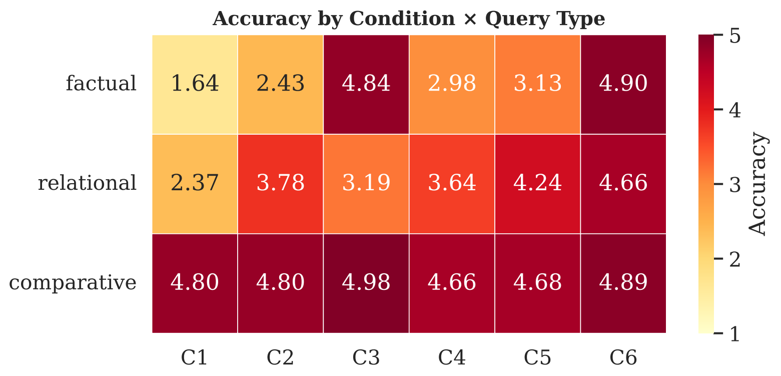


Fig. 7: Heatmap of mean accuracy scores across domains and conditions. Darker cells indicate higher accuracy. The pattern shows that enhanced entity pages (C3, C6) consistently outperform other conditions across all domains.

Table 7 reveals an important finding about *navigational affordances*: enhanced entity pages (C6) expose **2.0**× **more discoverable links** than JSON-LD pages (78.7 vs. 39.9) and **2.4**× **more** than plain HTML (78.7 vs. 33.4). Interestingly, the agent follows *fewer* links in C6 (0.4 vs. 0.9 in C4), yet achieves higher accuracy. This suggests that the enhanced page format provides such rich context in the initial retrieval that the agent needs fewer additional exploration steps.

The decreasing tool call count from C4 (2.8) to C6 (1.7) further supports this interpretation: enhanced pages enable the agent to answer effectively with fewer actions, indicating more efficient ReAct-style planning when navigational affordances are clear.

5 Discussion

5.1 Two Worlds of AI Search: Parsed vs. Flat Ingestion

Our findings reveal a critical distinction in the emerging landscape of AI-powered search—one that has direct implications for the Generative Engine Optimization (GEO) community.

Today's AI search systems fall into two architectural categories with respect to structured data:

Dedicated structured data pipelines. Traditional search engines such as Google and Bing have evolved specialized crawl-time parsers that *extract* `<script type="application/ld+json">` blocks as a separate signal, independent of the page body text. In these systems, JSON-LD feeds directly into entity understanding, knowledge panels, and rich results. The structured data is never flattened into a single text embedding—it is parsed, validated, and indexed in a dedicated knowledge graph layer.

Flat-text RAG pipelines. The fast-growing ecosystem of RAG-based AI assistants, agentic search, and retrieval-augmented chatbots typically ingests web pages as a single text chunk. In these systems—which include our Vertex AI Vector Search 2.0 pipeline as well as most LangChain, LlamaIndex, and custom RAG deployments—JSON-LD is just more text competing for a limited embedding budget. Our results show that in this regime, **JSON-LD alone provides no measurable benefit** ($\Delta = +0.07$, $p = 1.0$).

This distinction is crucial for practitioners. Our study provides an empirical snapshot of the *current status quo*: as AI-powered search diversifies beyond Google and Bing, content optimized only with JSON-LD will fail to surface in the growing number of flat-text RAG systems. The enhanced entity page format we propose bridges this gap—it makes structured knowledge visible and actionable *regardless of how the search system ingests content.*

5.2 From SEO to SEO 3.0: The Reasoning Web

Our findings offer empirical grounding for the emerging concept of *SEO 3.0* [24], the next evolutionary phase of search optimization for an AI-driven web [9]. Search optimization can be described across three eras:

- **SEO 1.0 — Document Ranking** (1998–2011): Optimizing for keyword matching and link-based authority signals. Success measured by ranking position, organic traffic, and click-through rates.
- **SEO 2.0 — Structured Data** (2011–2024): The launch of Schema.org in 2011 enabled machines to parse entity properties from web pages, powering knowledge panels, rich snippets, and improved entity understanding. Success measured by structured data adoption, data quality, and rich result eligibility.
- **SEO 3.0 — The Reasoning Web** (2024–present): AI search systems do not merely *rank* or *parse* content; they synthesize and reason over retrieved information, often producing direct answers and taking steps on behalf of users. Success must now be measured across three distinct dimensions, introduced in the next section.

The AI visibility spectrum: Citations, Reasoning, Actions. For teams working on AI visibility, our experiment reveals that content optimization must target three progressively deeper levels of AI engagement:

1. **Citations** — *Is your content retrieved and attributed?* This is the most basic form of AI visibility: appearing as a source in AI-generated answers. Our C1/C2 conditions evaluate this—whether the retrieval pipeline surfaces the right documents. JSON-LD helps with citation in systems that parse it (Google, Bing), but not in flat-text RAG systems.
2. **Reasoning** — *Can the AI reason correctly over your content?* Even when cited, the AI must extract and synthesize the right facts. Our accuracy and completeness metrics measure this. Enhanced entity pages (C3) improve reasoning by +29.5% over plain HTML—not because the facts are different, but because they are *presented* in a way the LLM can reliably extract and compose.
3. **Actions** — *Can the AI agent act on your content?* In agentic systems, the AI does not just retrieve and reason—it follows links, queries APIs, and performs multi-step tasks on behalf of the user. Our C4–C6 conditions evaluate this. Enhanced entity pages with dereferenceable URIs and navigational affordances (C6) enable agents to traverse knowledge graphs, aggregating information across entity boundaries. This is the frontier of AI visibility—and it requires content designed for *agent traversal*, not just retrieval.

The practical implication for AI visibility teams is clear: optimizing for *citations alone* (the current focus of most GEO strategies) is necessary but insufficient. The competitive advantage lies in optimizing for *reasoning* (through enhanced content presentation) and *actions* (through navigational affordances and tool-accessible endpoints).

5.3 Implications for Web Publishers and GEO

1. **JSON-LD is necessary but not sufficient**: Schema.org markup remains valuable for search engines with dedicated parsers (Google, Bing), but it

provides *no measurable benefit* in RAG-based systems that treat pages as flat text. Publishers who rely solely on JSON-LD are optimizing for only one class of AI search.
2. **Enhanced entity pages are the bridge**: Our format—with natural language summaries, navigable links, and `llms.txt`-style agent instructions—achieves +29.5% accuracy in standard RAG and +30.8% with agentic traversal. It works *in both worlds*: the structured JSON-LD is still present for traditional parsers, while the human- and agent-readable presentation ensures RAG systems can also leverage it.
3. **Dereferenceable URIs enable traversal**: Publishing entities with URIs that support content negotiation allows agents to follow links and aggregate information across knowledge graph boundaries—an affordance that flat-text pipelines cannot exploit from JSON-LD alone.

5.4 Implications for RAG System Design

Our results challenge the dominant "documents as flat text" paradigm and point toward structured-data-aware retrieval:

1. **Presentation matters more than presence**: Our experiment shows that the *same underlying knowledge* (Schema.org entities) yields dramatically different results depending on how it is presented to the retrieval pipeline. Raw JSON-LD in a `<script>` tag: no effect. The same information rendered as natural language with explicit properties, links, and navigation: +29.5–30.8% accuracy gain.
2. **Navigational affordances matter for agents**: The gap between C5 and C6 (accuracy: 3.89 vs. 4.45) shows that even capable agents need explicit navigational cues to effectively explore linked data.
3. **Enhanced pages enable efficient retrieval**: Agents using enhanced pages (C6) make fewer tool calls (1.7 vs. 2.8) yet achieve higher accuracy, suggesting that well-structured content reduces the need for multi-hop exploration.
4. **Toward structured-data-aware ingestion**: RAG system designers should consider architectures that extract and separately index structured data blocks—mirroring what traditional search engines already do—rather than treating all page content as a single text field.

5.5 Limitations

Our study has several limitations:

– **Flat-text ingestion architecture**: A critical architectural consideration is how our retrieval pipeline handles structured data. Vertex AI Vector Search 2.0 ingests each document as a single text field (truncated to ∼20k characters for embedding). In our corpus, 82% of plain HTML and 88% of JSON-LD documents exceed this limit, meaning the JSON-LD added to C2 documents is often partially or fully truncated before indexing. The JSON-LD `<script>`

block starts at a median position of character 18,510—right at the truncation boundary.

This differs fundamentally from how production search engines operate. Google's crawler, for example, *extracts* JSON-LD from `<script type="application/ld+json">` blocks as a *separate signal*, independent of the page's body text. The structured data is parsed into entity properties and indexed in a knowledge graph, not flattened into a single text embedding. A retrieval architecture that similarly extracts and separately indexes structured data—e.g., using multiple embedding fields or a hybrid entity–document store—might yield a significant H1 result. This remains an important area for future work.

- **Scale**: While 305 queries across 4 domains is sufficient for statistical significance, larger-scale experiments would strengthen generalizability claims.
- **LLM judge**: To mitigate correlated biases, we use separate models for generation (Gemini 2.5 Flash) and evaluation (Gemini 3.0 Flash). While this reduces same-model bias, both models share the Gemini family's training distribution. Future work should incorporate independent human evaluation for additional validation.
- **Single retrieval system**: Our results are specific to Vertex AI Vector Search 2.0 with `gemini-embedding-001`. We chose this system intentionally: it represents the flat-text ingestion architecture used by most AI search systems that operate independently of Google or Bing's proprietary index. Systems with structured-data-aware ingestion may show different sensitivity to JSON-LD markup.
- **Knowledge graph quality**: Our domains use well-maintained knowledge graphs served by WordLift's Linked Data Platform. The effectiveness of structured data may be lower for noisier or less complete KGs.

5.6 Ethical Considerations and Data Trustworthiness

A critical aspect of our approach is that the structured data consumed by AI agents is **the same data visible to human users**. The JSON-LD embedded in each page describes the exact same entity properties, relationships, and facts that appear in the human-readable HTML representation. Similarly, dereferenceable entity URIs serve the same underlying data through content negotiation—whether rendered as HTML for humans, JSON-LD for machines, or Turtle for SPARQL queries.

This **coupling between human and machine representations** creates stronger faithfulness guarantees than architectures where AI systems consume entirely separate data feeds. In a fully decoupled web—where machines and humans follow two different tracks—there is no natural accountability mechanism: structured data could drift from visible content, or be deliberately manipulated to influence AI outputs without corresponding changes visible to users. In our approach, because the structured data *is* the page content (expressed in machine-readable form), any manipulation would also be visible to human visitors, creating a natural check on data integrity.

This distinguishes our work from content-optimization approaches such as GEO [2], where optimization strategies (adding statistics, citations, or authoritative language) may create a divergence between what is optimized *for AI consumption* and what is genuinely informative *for human readers*. Our enhanced entity pages, by contrast, surface the same structured knowledge to both audiences.

This observation has broader implications for the emerging reasoning web. As AI agents increasingly rely on structured data to construct answers, the trustworthiness of that data becomes paramount. Systems that maintain a **single source of truth**—serving both human and machine consumers—are inherently more auditable and resistant to adversarial manipulation than those that decouple the two channels.

5.7 Future Work

Our findings motivate several research directions:

1. **Structured-data-aware retrieval**: The null result for H1 is specific to our flat-text ingestion pipeline. Future work should investigate architectures that extract JSON-LD separately and index entity properties as structured metadata—similar to how Google treats Schema.org markup as a distinct signal from page content. Vertex AI Vector Search 2.0 supports multiple data fields with independent embeddings; a dual-field approach (body text + structured data) with multi-vector retrieval could unlock the latent value of JSON-LD for RAG.
2. **Entity-centric chunking**: Rather than truncating documents at a fixed character limit, entity-aware chunking strategies that preserve structured data blocks could improve retrieval for content-rich pages.
3. **Cross-system replication**: Replicating our experiment with retrieval systems that natively parse structured data (e.g., knowledge-graph-augmented retrievers) would help disentangle the effect of structured data from the limitations of our ingestion pipeline.
4. **Production-scale validation**: Deploying enhanced entity pages on live websites and measuring their impact on AI-powered search engines (SGE, Perplexity) would validate ecological validity.
5. **Recursive Language Models on Knowledge Graphs**: Building on the RLM framework [28], we are exploring an approach (RLM-on-KG) that replaces the flat-context window with iterative graph exploration [23]. Rather than retrieving a fixed set of documents, the model navigates the knowledge graph recursively—fetching thin evidence from entity neighbors, deciding which relationships to follow, and synthesizing answers with full provenance. This extends our current agentic pipeline from tool-augmented retrieval to fully recursive, structure-guided reasoning over linked data.

5.8 Practical Recommendations

Based on our findings, we recommend the following for practitioners:

1. **Go beyond JSON-LD**: While Schema.org markup is valuable for search engines that extract it separately, our results show it does not improve RAG accuracy when treated as flat text. Invest in enhanced entity pages that make structured data *human- and agent-readable*.
2. **Use dereferenceable URIs**: Ensure that entity URIs resolve to content-negotiable endpoints that serve JSON-LD when requested programmatically.
3. **Adopt the enhanced entity page pattern**: Augment existing pages with explicit link navigation, breadcrumbs, and `llms.txt`-style instructions for AI agents.
4. **Test with agentic workloads**: As AI-powered search becomes prevalent, test content with agentic RAG systems rather than relying solely on traditional SEO metrics.

6 Conclusion

We have presented a controlled experimental study demonstrating that enhanced entity pages significantly improve the performance of Retrieval-Augmented Generation systems. Across 1,785 evaluations spanning four industry domains, we found that:

1. Schema.org JSON-LD markup alone does not significantly improve RAG accuracy ($p = 1.0$), highlighting that structured data must be *presented*, not just embedded.
2. Our enhanced entity page format, designed for agentic discoverability, yields +29.5% accuracy gains in standard RAG ($d = 0.56$) and +30.8% in the full agentic pipeline ($d = 0.60$).
3. Agentic RAG with link traversal significantly improves both accuracy (+12.0%) and completeness (+26.4%) over standard RAG.
4. These effects generalize across editorial, legal, travel, and e-commerce domains.

Our work provides empirical evidence that the Semantic Web's original vision—machine-readable structured data enabling intelligent agents—directly translates to measurable improvements in today's AI systems. As generative AI increasingly mediates information access, the presence and quality of structured linked data becomes not just an SEO signal, but a fundamental enabler of accurate, complete, and well-grounded AI responses.

Reproducibility. Our dataset, evaluation framework, enhanced entity page templates, and experiment configuration are available at `https://github.com/wordlift/seo3-reasoning-web`.

Acknowledgements. We thank the Google Cloud team for their generous support through Cloud credits that made these experiments possible. All experiments—including embedding generation, vector search indexing, Gemini-based generation (Gemini 2.5 Flash) and evaluation (Gemini 3 Flash Preview), and the

Vertex AI Vector Search 2.0 infrastructure—were run entirely on Google Cloud. We also thank the WordLift engineering team for maintaining the knowledge graph infrastructure and GraphQL API used in this study.

References

1. Schema.org. `https://schema.org` (2024), accessed: 2025-01-15
2. Aggarwal, P., Murahari, V., Rajpurohit, T., Kalyan, A., Narasimhan, K.R., Deshpande, A.: GEO: Generative engine optimization. In: Proceedings of the European Conference on Information Retrieval (ECIR) (2024), arXiv:2311.09735
3. Aichroth, P., Boch, L., Grassi, M., et al.: MICO – media in context. In: Proceedings of the IEEE International Conference on Multimedia & Expo Workshops (ICMEW). Torino, Italy (Jun 2015). `https://doi.org/10.1109/ICMEW.2015.7169827`
4. Anthropic: Model context protocol (MCP) specification. `https://modelcontextprotocol.io` (2024)
5. Asai, A., Wu, Z., Wang, Y., Sil, A., Hajishirzi, H.: Self-RAG: Learning to retrieve, generate, and critique through self-reflection. arXiv preprint arXiv:2310.11511 (2024)
6. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American **284**(5), 34–43 (2001)
7. Bizer, C., Heath, T., Berners-Lee, T.: Linked data – the story so far. International Journal on Semantic Web and Information Systems **5**(3), 1–22 (2009)
8. Borgeaud, S., Mensch, A., Hoffmann, J., Cai, T., Rutherford, E., Millican, K., van den Driessche, G.B., Lespiau, J.B., Damoc, B., Clark, A., et al.: Improving language models by retrieving from trillions of tokens. arXiv preprint arXiv:2112.04426 (2022)
9. Gartner: Disruptive technologies to watch in 2024 and beyond. `https://www.gartner.com/en/articles/disruptive-technologies` (2024)
10. Google Cloud: Google agent development kit (ADK). `https://github.com/google/adk-python` (2025)
11. Google Cloud: Vertex AI vector search 2.0. `https://cloud.google.com/vertex-ai/docs/vector-search` (2025)
12. Google DeepMind: Gemini: A family of highly capable multimodal models. `https://deepmind.google/technologies/gemini/` (2024)
13. Guha, R.V., Brickley, D., Macbeth, S.: Schema.org: Evolution of structured data on the web. Communications of the ACM **59**(2), 44–51 (2016)
14. Guu, K., Lee, K., Tung, Z., Pasupat, P., Chang, M.W.: REALM: Retrieval-augmented language model pre-training. In: International Conference on Machine Learning (ICML) (2020)
15. Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., de Melo, G., Gutierrez, C., Kirrane, S., Labra Gayo, J.E., Navigli, R., Neumaier, S., et al.: Knowledge graphs. ACM Computing Surveys **54**(4) (2021)
16. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.t., Rocktäschel, T., Riedel, S., Kiela, D.: Retrieval-augmented generation for knowledge-intensive NLP tasks. In: Advances in Neural Information Processing Systems (NeurIPS) (2020)
17. Mavi, V., Jangra, A., Jatowt, A.: Multi-hop question answering. Foundations and Trends in Information Retrieval **17**(4), 457–586 (2024)

18. Noy, N., Gao, Y., Jain, A., Narayanan, A., Patterson, A., Taylor, J.: Industry-scale knowledge graphs: Lessons and challenges. ACM Queue **17**(2), 48–75 (2019)
19. Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., Wu, X.: Unifying large language models and knowledge graphs: A roadmap. IEEE Transactions on Knowledge and Data Engineering (2024)
20. Peng, B., Zhu, Y., Liu, Y., Bo, X., Shi, H., Hong, C., Zhang, Y., Tang, S.: Graph retrieval-augmented generation: A survey. arXiv preprint arXiv:2408.08921 (2024)
21. Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Hambro, E., Zettlemoyer, L., Cancedda, N., Scialom, T.: Toolformer: Language models can teach themselves to use tools. Advances in Neural Information Processing Systems (NeurIPS) (2024)
22. Trivedi, H., Balasubramanian, N., Khot, T., Sabharwal, A.: Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL) (2023)
23. Volpini, A.: RLM-on-KG: Recursive Language Models and the future of SEO. `https://wordlift.io/blog/en/recursive-language-models-on-kg/` (2025)
24. Volpini, A.: What is SEO 3.0? Preparing for the next wave of search. `https://wordlift.io/blog/en/what-is-seo-3-0/` (2025)
25. Volpini, A., Riccitelli, D.: WordLift: Meaningful navigation systems and content recommendation for news sites running WordPress. In: Proceedings of the ESWC 2015 Developers Workshop, CEUR Workshop Proceedings. vol. 1361 (2015), `http://ceur-ws.org/Vol-1361/`
26. Willison, S.: llms.txt – a proposal for standardizing LLM instructions. `https://llmstxt.org` (2024)
27. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., Cao, Y.: ReAct: Synergizing reasoning and acting in language models. In: International Conference on Learning Representations (ICLR) (2023)
28. Zhang, C., Kraska, T., Khattab, O.: Recursive language models: A theoretical framework for language model reasoning over long texts. arXiv preprint arXiv:2501.12394 (2025)

A Full Prompts and Templates

A.1 Standard RAG Generation Prompt

The following prompt is used by Gemini 2.5 Flash to generate answers from retrieved context in conditions C1–C3 (standard RAG):

```
You are a helpful assistant answering questions
based on the provided context documents.

IMPORTANT RULES:
1. Answer ONLY based on the information in the
   context documents below.
2. If the context does not contain enough
   information, say "I cannot find sufficient
   information to answer this question."
3. Cite the specific documents you used by
```

```
   their ID.
4. Be precise and factual.

CONTEXT DOCUMENTS:
{context}

QUESTION: {query}

Provide a clear, accurate answer with citations
to the source documents.
```

A.2 Agentic RAG System Instruction

The following system instruction is provided to the Google ADK agent in conditions C4–C6 (agentic RAG). The agent uses this instruction to plan its tool-calling strategy:

```
You are a research assistant with access to a
knowledge graph and document search.
Your goal is to answer the user's question as
accurately and completely as possible.

STRATEGY:
1. First, use search_documents to find relevant
   documents.
2. Examine the results for linked entity URLs
   (especially data.wordlift.io URLs).
3. If the question requires information about
   related entities (provider, offers, etc.),
   use follow_entity_link to fetch their data.
4. If you need to discover entities not directly
   linked, use search_knowledge_graph.
5. Synthesize all gathered information into a
   comprehensive answer.
6. Cite your sources by document ID or entity
   URL.

RULES:
- Only state facts you found in the retrieved
  data. Do not hallucinate.
- Follow at most {max_hops} link hops.
- If you cannot find sufficient information,
  say so explicitly.
```

A.3 LLM Judge Prompts

Three evaluation prompts are used by the independent judge model (Gemini 3 Flash Preview). All judge calls request JSON output via `response_mime_type="application/json"`.

Accuracy (1–5):

```
You are an evaluation judge. Given a question,
a ground-truth answer, and a candidate answer
produced by a system, rate the candidate's
factual accuracy.

Question: {question}
Ground Truth Answer: {ground_truth}
Candidate Answer: {candidate}

Rate the accuracy on a scale of 1-5:
  1 = Completely wrong or irrelevant
  2 = Mostly wrong, with minor correct elements
  3 = Partially correct, missing key facts
  4 = Mostly correct, minor inaccuracies
  5 = Fully correct and accurate

Respond in JSON format ONLY:
{"score": <1-5>, "reasoning": "brief explanation"}
```

Completeness (1–5):

```
You are an evaluation judge. Given a question,
a ground-truth answer containing key facts, and
a candidate answer, rate the completeness of the
candidate.

Question: {question}
Ground Truth (contains key facts the answer
should cover): {ground_truth}
Candidate Answer: {candidate}

Rate completeness on a scale of 1-5:
  1 = Covers none of the key facts
  2 = Covers ~25% of key facts
  3 = Covers ~50% of key facts
  4 = Covers ~75% of key facts
  5 = Covers all key facts

Respond in JSON format ONLY:
{"score": <1-5>, "facts_covered": [...],
 "facts_missing": [...],
 "reasoning": "brief explanation"}
```

Grounding (binary):

```
You are an evaluation judge. Given a candidate
answer and the source documents it was based on,
determine what fraction of claims in the answer
are traceable to the source documents.
```

```
Candidate Answer: {candidate}
Source Documents: {sources}

For each claim in the answer, determine if it
is supported by the source documents.

Respond in JSON format ONLY:
{"grounding_score": <0.0-1.0>,
 "total_claims": <int>,
 "grounded_claims": <int>,
 "ungrounded_claims": ["claim1", "claim2"]}
```

A.4 Enhanced Entity Page Template

The enhanced entity page template (Jinja2) used for condition C3 and C6 documents. Key features: Schema.org type breadcrumbs, embedded JSON-LD, visible linked entity navigation with content negotiation instructions, and `llms.txt`-style agent instructions.

```
<!DOCTYPE html>
<html lang="en"
  prefix="schema:␣http://schema.org/">
<head>
  <title>{{ entity_name }}
    -- {{ domain_name }}</title>
  <script type="application/ld+json">
    {{ jsonld_data }}
  </script>
</head>
<body vocab="http://schema.org/"
  typeof="{{␣entity_types␣}}">

  <!-- Type breadcrumbs -->
  <nav class="breadcrumb">
    Thing > CreativeWork > {{ entity_types }}
  </nav>

  <h1 property="name">{{ entity_name }}</h1>
  <div class="entity-type">
    Type: schema:{{ entity_types }}
  </div>

  <section><h2>Description</h2>
    <p property="description">
      {{ entity_description }}
    </p>
  </section>
  <!-- Linked entity navigation -->
  <section class="linked-entities">
    <h2>Related Entities</h2>
    <p>Each URI supports content negotiation
      -- append .json for JSON-LD, .ttl for
      Turtle, or .html for a human-readable
      view.</p>
    <ul>
    {% for le in linked_entities %}
      <li>
        <span class="relation">
          {{ le.relation }}:
        </span>
        <a href="{{␣le.url␣}}">{{ le.url }}</a>
      </li>
    {% endfor %}
    </ul>
  </section>

  <!-- Agent instructions (llms.txt style) -->
  <section class="agent-instructions">
    <h2>Agent Instructions</h2>
    <pre>{{ llms_instructions }}</pre>
  </section>

</body></html>
```