# Integrating Knowledge Graph and Large Language Models for Defining Business Strategies

**Eleonora Ghizzota***, **Alex Jordan**†, **Alessandro Petruzzelli***, **Lucia Siciliani***
**Giuseppe Spillo***, **Pierpaolo Basile***, **Davide Sola**†
**Giovanni Scarso Borioli**†, **Giovanni Semeraro***
*Department of Computer Science, University of Bari Aldo Moro
†ESCP Business School, †3HORIZONS
e.ghizzota@phd.uniba.it, ajordan@3horizons.com, dsola@3horizons.com, gsb@3horizons.com
{alessandro.petruzzelli, lucia.siciliani, giuseppe.spillo, pierpaolo.basile, giovanni.semeraro}@uniba.it

**Abstract**

Effective business strategy formulation requires synthesising diverse, often conflicting information sources into coherent action plans. While Large Language Models (LLMs) show potential for processing textual information at scale, their application is limited by hallucinations and a lack of grounding in proprietary data. This paper proposes a methodology that integrates a domain-specific Knowledge Graph (KG) with a GraphRAG pipeline to generate strategic briefing documents, or Primers, which provide a structured overview of a company's competitive environment. Our approach utilizes an ontology-first framework and Cypher-based graph traversal to capture the relational nature of strategic knowledge beyond simple vector retrieval. Experimental results on a Q&A dataset demonstrate that the Vector + Cypher retrieval strategy significantly improves grounding over LLM-only baselines and outperforms naive vector retrieval in terms of completeness and usefulness. These findings suggest that the synergy of LLMs and structured KGs provides a robust foundation for automated strategic analysis in real-world business scenarios.

**Keywords:** Knowledge Graph, Large Language Models, Business Strategies

## 1. Background and Motivation

The formulation and execution of business strategy require the synthesis of diverse, often conflicting information sources (market data, competitive intelligence, industry trends, organizational capabilities and more) into coherent action plans. Despite decades of academic research producing thousands of strategic frameworks and billions spent annually on strategy consulting, most organizations still struggle to turn strategy into action effectively (Vigfússon et al., 2021). Established frameworks such as PESTEL, SWOT (Puyt et al., 2023), and Porter's Five Forces (Porter, 1979) are, by design, simplifications: each captures only a subset of strategic reality, and their effective application requires significant domain expertise to re-contextualize abstract models with industry- and company-specific knowledge (Kim et al., 2026). As the pace and complexity of the business environment accelerate, the gap between available information and an organization's capacity to process it into strategic insight continues to widen, motivating the development of AI-assisted approaches to strategic analysis (Scarso-Borioli et al., 2024).

Large Language Models (LLMs) have demonstrated considerable potential for processing and synthesizing textual information at scale (Brown et al., 2020). However, their application to business strategy faces critical limitations: LLMs lack access to proprietary or real-time organizational data, are prone to generating plausible but unverified content (hallucination), and offer no inherent guarantee that their outputs are grounded in authoritative sources (Kim et al., 2026). Recently, (Kim et al., 2026) introduced the CIAIC framework, which orchestrates multiple specialized LLM agents to collaboratively generate strategic analyses through a five-stage process combining human-guided objectives, retrieval-augmented drafting, and collective revision. While effective for multi-agent collaboration, their approach relies on web-based retrieval and does not leverage structured knowledge representations for context grounding.

Knowledge Graphs (KGs) offer a complementary approach by providing structured, queryable representations of domain knowledge that preserve entity relationships and support traceable reasoning (Hogan et al., 2021). A growing body of work has explored the use of LLMs for KG construction and ontology engineering: (Kommineni et al., 2024) proposed an LLM-supported approach to ontology and KG construction that reduces reliance on manual expert curation; (Abolhasani and Pan, 2024) demonstrated automated ontology extraction and KG generation through an interactive pipeline; (Lippolis et al., 2025) evaluated prompting techniques for LLM-driven ontology generation across multiple domains; and (Oyewale and Soru, 2026) addressed LLM-driven ontology construction specifically for enterprise KGs. Complementary work on ontology engineering methodologies includes LLM-assisted ontology development (Saeedizade and Blomqvist, 2024) and conversational ontology en-



*Proceedings of the Knowledge Graphs and Large Language Models Workshop 2026 (KG-LLM@LREC26)*, pages 53–62
May 16, 2026. ©2026 European Language Resource Association

gineering frameworks (Zhang et al., 2024). These studies demonstrate the growing maturity of LLM-KG integration, yet few have applied this synergy to business strategy.

Retrieval-Augmented Generation (RAG) (Lewis et al., 2020) bridges LLMs and external knowledge sources by retrieving relevant context at inference time. However, conventional RAG approaches exhibit several shortcomings (Peng et al., 2024), including degraded performance with long contexts (Liu et al., 2023) and limited capability to capture complex relationships and global context due to their reliance on vector-based retrieval (Edge et al., 2024). Graph Retrieval-Augmented Generation (Graph RAG) (Edge et al., 2024; Hu et al., 2024; Zhang et al., 2025) addresses these issues by integrating RAG with graph-structured data. This approach leverages knowledge graphs to better represent interconnected information, so that the LLM can generate responses that are both linguistically fluent and grounded in structured domain knowledge. In this paper, we propose a methodology that integrates a domain-specific KG with a GraphRAG pipeline to generate strategic briefing documents, referred to as *Primers*. A Primer provides a structured, outside-in overview of a company's competitive environment, covering its market positioning, customer segments, product offerings, industry trends, and competitive dynamics. Primers are foundational artifacts in strategy consulting, consolidating desk-research intelligence into a coherent narrative that supports subsequent strategic decision-making. In the current work, we focus exclusively on this outside-in analytical perspective, leaving extensions to strategy execution planning for future work.

Our approach differs from prior work in two key respects. First, rather than using LLMs to construct the KG itself, we adopt an ontology-first approach: a purpose-built strategy ontology, developed collaboratively by domain experts and knowledge engineers, serves as the structured backbone that grounds LLM generation in validated domain knowledge. This ensures that AI-generated strategic content is anchored in a coherent conceptual schema rather than relying solely on the model's parametric knowledge. Second, our GraphRAG pipeline exploits the graph structure through Cypher-based traversal beyond simple vector retrieval, enabling the recovery of structurally connected entities that capture the relational nature of strategic knowledge.

In the current work, because we have not included knowledge about strategy execution planning, we cannot exploit the KG to build the entire Primer. For those reasons, we validate the contribution of KG and GraphRAG by analyzing the performance on a dataset of Q&A pairs extracted automatically from the Primer. Our research goal is to demonstrate that the combination of an LLM and a KG can be effective in a real-world application scenario where the main objective is to define business strategies.

## 2. Methodology

As stated in the previous sections, transforming heterogeneous market intelligence into structured knowledge that can be used by LLMs or other AI-based applications in a controlled, grounded manner is not a trivial task. The proposed pipeline connects ontology engineering, graph construction, and retrieval-augmented generation into a single, coherent workflow for producing strategic briefing documents.

Figure 1 depicts the overall architecture, organized into two macro phases:

- Offline phase: definition of the KG schema, graph construction, and semantic indexing;
- Online phase: execution of a GraphRAG pipeline that retrieves relevant graph context and supports grounded LLM generation.

The overall process follows four connected stages:

1 KG Schema Definition: A purpose-built market intelligence ontology formalizes the conceptual backbone of outside-in strategic analysis. It defines the entities, attributes, and relationships necessary to represent companies, markets, trends, competitors, and value-chain dynamics in a structured and interoperable way.

2 KG Construction: The ontology is instantiated within a Neo4j graph database by transforming raw business documents and structured sources into labeled nodes and typed relationships. This step operationalizes the conceptual schema into a queryable knowledge base.

3 GraphRAG Pipeline: The knowledge graph becomes the external memory of the LLM. Through vector embeddings and Cypher-based graph traversal, the Retriever module identifies relevant nodes and structurally connected entities. The retrieved context is then formatted and injected into the LLM prompt, enabling generation that is grounded in validated domain knowledge rather than relying solely on parametric memory.

4 Evaluation: The methodology is assessed through a controlled Question&Answer dataset derived from domain documents. Multiple configurations, including LLM-only and alternative retrieval strategies, are compared to measure accuracy, completeness, usefulness, and grounding to source material.

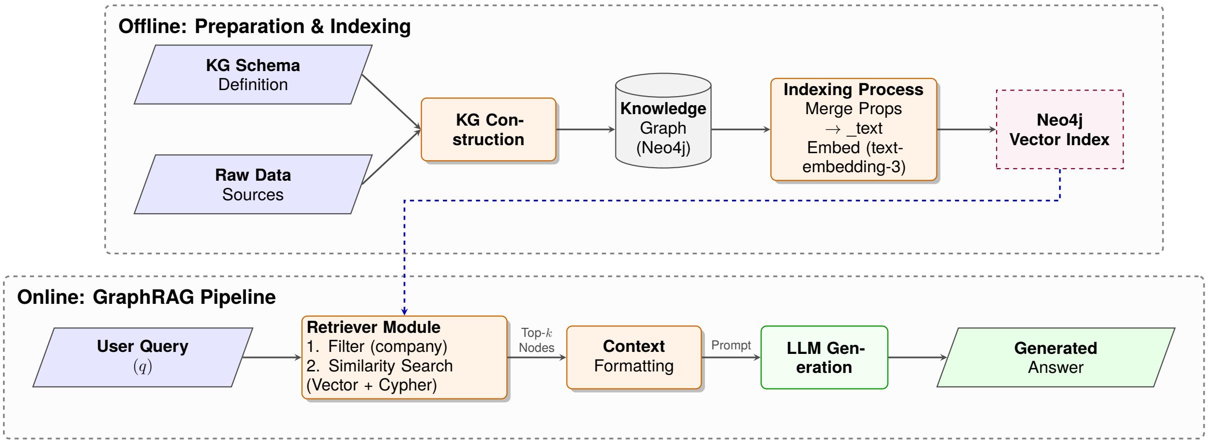


Figure 1: The proposed methodology pipeline. The upper dashed box indicates the offline phase, where data is structured according to the KG Schema into Neo4j, and node properties are merged and embedded into a Vector Index. The lower dashed box shows the online GraphRAG process.

As illustrated in Figure 1, the offline layer prepares the knowledge infrastructure: raw data is structured according to the ontology, node properties are consolidated into unified textual representations, embedded in vector space, and indexed. The online layer activates this infrastructure at inference time: a user query triggers filtering and similarity search, optionally enriched by graph traversal, and the selected subgraph is provided to the LLM for answer generation.

This modular organization separates knowledge modeling from generative reasoning while preserving their interaction. It also enables testing the systematic evaluation of the graph component in the system. The following subsections detail each stage of the pipeline.

## 2.1. KG Schema Definition

A comprehensive strategy ontology spans the full scope of strategic decision-making, from an organization's aspirational goals and competitive arena, through its value proposition and capability configuration, to its execution architecture. For the purposes of this paper, we focus on the *competitive arena*, the outside-in analysis of markets, competitors, trends, and customer dynamics, which constitutes the analytical foundation for generating strategic briefing documents (Primers). Extensions to the remaining ontological dimensions (value proposition design, capability planning, execution orchestration) are left for future work.

The knowledge graph is accordingly built upon a market intelligence ontology designed to capture the entities, attributes, and relationships relevant to outside-in business strategy analysis. The ontology was developed from scratch, without reusing existing domain ontologies, following the iterative methodology outlined in (Noy and McGuinness, 2001). Its theoretical foundations were established through an integrative synthesis of five decades of strategic management literature (Scarso-Borioli et al., 2024), which has been independently peer-reviewed and accepted for presentation at the EURAM 2026 conference. The development team comprised four domain experts: four strategy practitioners (from a strategy consulting firm) who are also strategy academics (three professors and one doctoral researcher). The ontology has undergone two major iterations and has been validated through extensive application in real-world consulting engagements across multiple industries.

The schema comprises over 15 node types organized into four conceptual groups:

**Core Business Entities.** The **Company** node is the central entity, characterized by its identity, structure, and strategic relationships. Each company participates in one or more **Market Spaces**, multidimensional competitive arenas defined by the intersection of customer segments, geographies, and product categories. The *participates_in* relationship between Company and Market Space carries strategic properties: *role_qualifier* (e.g., leader, challenger, niche player) and *distinctive_choices* (e.g., premium positioning, modular design), capturing not only *where* a company competes but *how* it differentiates. Companies further link to **Customer Segments**, **Offerings** (products or services), and **Geographies**.

**Market Dynamics.** Temporal and causal dimensions of market behavior are modeled through a chain of **Trends** (PESTEL-classified macro forces), **Drivers** (operational factors that translate trends into market effects), and **Sizing** nodes (time-stamped market size estimates). This Trend → Driver → Sizing chain enables traceable market forecasting, where projections are explicitly linked to underlying causal assumptions. **Attractiveness** nodes synthesize competitive conditions into evaluative scores informed by **Porter Force** and **Porter Driver** nodes, which decompose industry dynamics into granular, scoreable factors.

**Value Chain.** The schema models how value flows from producers to consumers through **Channels** (distribution routes), **End Users** (ultimate product consumers, distinct from purchasing customers), **Customer Needs** (functional requirements driving purchase decisions), and **Product/Service Categories**. The path Customer Segment → Customer Need → Product Category → Offering captures the full demand-to-delivery chain, supporting portfolio analysis and value proposition alignment.

**Competitive Landscape.** Competition is modeled through **Competitor** nodes with associated offerings, segments, and geographies, enabling side-by-side comparison of strategic footprints. **Deep-Dive** nodes capture structured competitor dossiers (strengths, weaknesses, market insights, pricing strategy), while **Summary Table** nodes provide comparative overlap analyses for direct competitive positioning.

The ontology is instantiated in Neo4j, where entities are stored as labeled nodes with typed properties and relationships as labeled directed edges. This representation supports both the vector-based semantic retrieval and the structural graph traversal central to the GraphRAG pipeline described in Section 2.3.

## 2.2. KG Construction

In this step, the entities and relationships defined in the KG Schema are instantiated and populated. Crucially, the ontology serves as a normative schema governing the entire population process: the LLM can produce instances only that conform to the predefined types and relationships, ensuring an ontology-first discipline in which the conceptual structure is designed by domain experts *prior to* any automated extraction. The current implementation relies on three complementary steps that jointly support this constrained graph construction.

The first step encodes the ontological backbone as a formal specification of the admissible node types, their attributes, and relationship types. This specification guarantees that entities such as Company, Market Space, Trend, Driver, Competitor, and related concepts are structured coherently in accordance with the domain experts' knowledge. By separating the conceptual structure from the population process, the schema can evolve in a controlled manner without requiring modifications to the ingestion logic, and ensures stability across different industries and use cases.

The second step consists of the prompts used in the generative phase. These prompts embed domain-specific knowledge and contextual constraints, emulating the steps domain experts follow to gather the information needed to write the primers. The output of this generative step, obtained using GPT-4o-mini, is formalized as Pydantic[1] classes, a Python library for data validation based on type annotations. These classes serve as typed, machine-readable representations of ontological classes. Each class encapsulates the attributes of an entity together with references to related entities, in accordance with the schema constraints. This intermediate layer functions as a validation mechanism: it enforces structural consistency and type correctness before knowledge is injected into the graph, ensuring that the LLM acts as a guided extraction engine rather than a schema designer.

In the final step, the Pydantic objects are programmatically converted into structures expressed using Cypher, the declarative query language used by Neo4j, so that they can be ingested into our database. Entities are instantiated as labeled nodes and relationships as typed, directed edges, optionally enriched with properties that capture strategic qualifiers or temporal dimensions. The result is a coherent and extensible graph-based representation of market intelligence, designed to serve as the structured memory component for the subsequent parts of the framework.

## 2.3. GraphRAG

The Custom GraphRAG pipeline defined for the task leverages the Neo4j Graph-RAG package[2]. The main components of this pipeline are the Retriever and the LLM. The Retriever gathers relevant information from external sources, such as documents or databases, in response to the user's query. Various methods, such as similarity searches or database queries, are used to find the most relevant data. The retrieved results are then ranked and scored according to how closely they match the query. In this pipeline, the knowledge graph acts as the external source. On the other hand, the purpose of the LLM is to generate an answer to the user query, leveraging the information obtained by the Retriever.

[1] https://docs.pydantic.dev/

[2] https://neo4j.com/docs/neo4j-graphrag-python/current/user_guide_rag.html

#### 2.3.1. Indexing

For each node, text properties are merged into a single new text property, namely _TEXT, in the form of PROPERTY NAME: PROPERTY VALUE. This step is fundamental for obtaining a unique embedding for each node's properties, rather than a separate embedding for each property. Embeddings are computed with **text-embedding-3-small** model with dimension 1,536 and stored within the node in the _EMBEDDING property. Leveraging Neo4j built-in functionalities, _EMBEDDING properties are indexed in a unique index. An index is the cornerstone of the retrieval step, since it is used to compute semantic similarity between its content and the user query.

#### 2.3.2. Retrieval

Two retrieval strategies are implemented. Both strategies filter nodes based on the COMPANY_NAME property and perform semantic-similarity searches on the aforementioned index. Both strategies compute the cosine similarity between the user query and the information in the _TEXT node property.

**Vector retrieval.** Given a user query $q$, the top-$k$ most similar nodes are retrieved from the knowledge graph. This approach corresponds to the Neo4j Vector Retriever implementation[3].

**Vector + Cypher retrieval.** Given a user query $q$, the top-$k$ most similar nodes (hereinafter, *seeds*) are retrieved from the knowledge graph. Then, for each seed, a predefined Cypher query is executed: neighboring nodes within 1-hop distance are retrieved and ranked by cosine similarity with $q$, then the top-$n$ are retained.

#### 2.3.3. Generation

The retrieved nodes are used as context in the LLM prompt. In case of **Vector Retrieval + Cypher retrieval**, the context is formatted as follows for each seed node $S$:

**Context formatting for generation**

Node $S_{label}$ has the following properties $S_{properties}$.
It is linked via relationship $R_1$ to node $N_{1,label}$ that has the following properties $N_{1,properties}$.
⋮
It is linked via relationship $R_n$ to node $N_{n,label}$ that has the following properties $N_{n,properties}$.

[3]https://neo4j.com/docs/neo4j-graphrag-python/current/user_guide_rag.html#vector-retriever

Regardless of the retrieval method, the LLM is instructed to avoid answering when no context is available, nor is it enough to answer the user query. The response fallback is "*I can not answer this question because I have no relevant context.*".

## 3. Evaluation

### 3.1. Dataset construction

To assess the capabilities of our designed methodology, we build a dataset composed of a set of Question&Answer (Q&A) pairs based on textual documents of our domain. The constructed dataset serves as the ground truth, and we compare the performance of the different configurations of our methodology (introduced in the previous sections) on it.

We build our dataset in a two-stage process: first, starting from a corpus of textual documents, we extract a set of *statements* (i.e., information that is considered to be true); then, for each statement, we obtain a set of questions for which the statement serves as the answer. A similar strategy is proposed in (Li and Zhang, 2024).

We exploit LLMs to perform both stages. In particular, given the straightforward nature of the extraction and generation tasks, we use GPT-4o-mini as our LLM, which allows for high performance while maintaining significantly lower computational costs and API overhead.

The need to employ a two-stage strategy rather than a single-stage approach stems from the requirement for data precision and comprehensive coverage. By first factualizing the documents into discrete statements, we ensure that the LLM captures *all* reported semantics and categorizes them into factual, comparative, or edge-case knowledge. Specifically, factualization transforms dense, multi-layered prose into atomic units of truth, ensuring that every discrete piece of information is isolated and preserved before being translated into a query. This intermediate step acts as a semantic anchor, preventing the loss of information that often occurs in a single-stage process where an LLM might prioritize common questions over niche but important facts. Furthermore, this decomposition enables the generation of multiple, diverse queries for each specific fact, ensuring that the resulting ground truth is both robust and strictly grounded in the source text. Both stages are detailed in the following sections.

#### 3.1.1. Statement Generation

At this stage, we aim to obtain, from a corpus of textual documents, a set of statements that capture the semantics encoded in the documents, expressed as simple text passages.

The process begins by extracting raw text from the source files and partitioning it into segments of approximately 800 words to ensure optimal processing by the model. Each chunk is processed by an LLM to generate a comprehensive list of factual, comparative, and edge-case statements. These statements are designed to be simple yet semantically complete, capturing the core knowledge of the source material in a structured JSON format. We use the following system prompt to carry out this step:

**Statement Generation Prompt**

```
You are a data generation assistant.
Your task is to read the provided text and
generate as many diverse and meaningful
statements as possible.
- Keep statements simple, but ensure they
capture all reported semantics.
- Cover three categories: factual, compara-
tive, and edge-case knowledge.

Output requirements:
- Respond ONLY with valid JSON.
- The response must be a single JSON array.
- Each element must be an object with exactly
two fields:
- "statement": <string>
- "type": one of ["factual", "comparative",
"edge-case"]
```

Here, we provide an example from our real data of the first stage. In particular, we show how raw textual information can be factualized in the form of a set of statements.

**First Stage Example**

**Text in raw file:** The company has a long history dating back to 1885 and has evolved through various strategic transformations, acquisitions, and technological advancements to become a leading provider of critical protection solutions.
**Generated statement:**
- "statement": "The company was founded in 1885 and has undergone various transformations since then.". "type": "factual"

#### 3.1.2. Q&A Pairs Generation

After generating statements, the second stage transforms these facts into formal ground-truth Q&A pairs. For each statement extracted in the previous step, the LLM is prompted to generate a variety of clear and unambiguous questions.

The primary constraint of this stage is that the provided statement must serve as the exact and complete answer to the generated question. This ensures high fidelity between the source documents and the final evaluation dataset. By generating multiple questions for a single statement, we create a robust set of queries that test the methodology's ability to retrieve the same underlying fact through different linguistic formulations. The system prompt for generating these pairs is defined as follows:

**Q&A Pairs Generation Prompt**

```
You are a question generation assistant.
Your task is to read the statement and gener-
ate all the possible questions whose answers
are exactly the statement derived from it.
- The same statement can be associated with
more questions.
- Each question must directly correspond to
the provided statement.
- Keep the question clear, concise, and unam-
biguous.
- Cover factual, comparative, and edge-case
types of knowledge.

Output requirements:
- Respond ONLY with valid JSON.
- The response must be a single JSON array.
- Each element must be an object with exactly
two fields:
- "question": <string>
- "answer": <string>, it is the statement
that has been provided
- "type": one of ["factual", "comparative",
"edge-case"]
```

Following the previous example, we report the set of Q&A pairs generated from the statement reported.

**Second Stage Example**

**Statement:**
- "statement": "The company was founded in 1885 and has undergone various transformations since then.". "type": "factual"
**Set of Q&A pairs:**
- "question": "When was the company founded?". "answer": "The company was founded in 1885 and has undergone various transformations since then."
- "question": "What year marks the founding of the company?". "answer": "The company was founded in 1885 and has undergone various transformations since then.",

As shown in the example, our methodology generates a robust set of queries that test how effectively it identifies the same core information despite differing phrasing.

The resulting dataset has been manually assessed by a domain expert, in order to guarantee completeness and correctness of the Q&A pairs.

### 3.2. Experimental Setup

To evaluate the effectiveness of the proposed methodology, we conducted experiments on a proprietary dataset covering 9 distinct companies across various industries. The evaluation aims to measure not only the correctness of the answers but also the completeness of the retrieved context and the adherence to the source documents.

#### 3.2.1. Baselines and Configurations

We compared three configurations:

1. **LLM Only (Baseline):** The LLM (GPT-4o) answers the user query directly using only its parametric knowledge, without any retrieved context.

2. **Vector Retrieval (Naive):** Corresponds to the standard RAG approach described in Section 2.3.2, where nodes are retrieved solely based on vector similarity.

3. **Vector + Cypher Retrieval (Ours):** The proposed methodology, where vector retrieval is followed by a 1-hop expansion and similarity-based ranking to capture structural dependencies.

#### 3.2.2. Metrics

We adopted a multi-faceted evaluation strategy combining reference-free LLM evaluation and reference-based lexical metrics:

- **LLM-based Metrics (1-5 Scale):** An independent LLM evaluator scored the generated answers against the ground truth on a Likert scale (1-5) for:
  - *Accuracy:* The correctness of the information relative to the ground truth.
  - *Completeness:* How thoroughly the answer covers the question's requirements.
  - *Usefulness:* The general helpfulness and relevance of the answer.
- **Grounding Metrics:** To measure adherence to the specific phrasing and content of the source business documents, we calculated:
  - *ROUGE-L & BLEU:* Lexical overlap with the ground truth answer.
  - *BERTScore:* Semantic similarity at the sentence level.

### 3.3. Results and Discussion

We evaluated the performance of the three proposed configurations on the filtered dataset, from which questions where the LLM lacks sufficient knowledge to answer were excluded. This limitation is due to the different nature of the KG and the questions: while the former was generated without any additional information, the generation of the latter was based on the information in the primers.

Table 1 presents the aggregated results across all 9 companies.

#### 3.3.1. The Trade-off: Parametric Knowledge vs. Document Grounding

A striking observation from the results is the high performance of the **LLM Only** baseline on the LLM-evaluated metrics (Accuracy: 4.17, Usefulness: 4.68). This can be attributed to the nature of the dataset; companies are sufficiently public that the model (GPT-4o) likely possesses strong parametric knowledge about them. Consequently, it generates fluent answers based on its pre-training. Moreover, the dataset was created with the same family model (GPT-4o-mini) of the LLM used to answer questions.

However, the **Grounding Metrics** reveal a critical limitation of the baseline. The LLM-only approach achieves a very low **ROUGE-L score of 0.07** compared to $\approx 0.21$ for the RAG methods. This indicates that the LLM is answering from memory rather than utilizing the specific syntax and details of the domain knowledge. In the context of defining business strategies, where adherence to a specific internal document is mandatory, the RAG approach is superior despite lower "fluency" scores, as it guarantees that the answer is grounded in the retrieved context (as evidenced by the 3x higher lexical overlap).

#### 3.3.2. Effectiveness of Graph Structure

Comparing the two retrieval strategies, the proposed **GraphRAG (Vector + Cypher)** demonstrates consistent improvements over the **Naive GraphRAG (Vector)** baseline across the LLM-evaluated qualitative metrics, while maintaining equivalent performance in traditional non-LLM metrics:

- **Completeness (+0.15):** The Vector + Cypher method achieves a score of 3.88 compared to 3.73 for the Naive approach. This suggests that expanding the context via graph traversal retrieves necessary auxiliary information (e.g., linked risks or strategies) that simple vector similarity misses.

- **Usefulness (+0.15):** The structural context translates into more useful answers (4.26 vs. 4.11), suggesting the graph context helps the LLM structure its response better.

- **Accuracy (+0.06):** We observe a modest improvement in accuracy (3.91 vs. 3.85).

- **BERTScore & ROUGE-L (Parity):** Both approaches achieve identical scores (0.88 and 0.21, respectively). Because both methods share the same underlying vector search, they retrieve the same primary text chunks, resulting in similar lexical and semantic overlap with the ground truth.

#### 3.3.3. Company-Specific Performance

The performance varies significantly across different companies, as shown in Table 2.

While the LLM dominates in cases with strong public presence (e.g., $Company_2$), the proposed GraphRAG method shows resilience in specific

| **Method** | **Accuracy** | **Completeness** | **Usefulness** | **BERTScore** | **ROUGE-L** |
|---|---|---|---|---|---|
| LLM Only (Baseline) | **4.17** | **4.35** | **4.68** | 0.84 | 0.07 |
| Naive GraphRAG (Vector) | 3.85 | 3.73 | 4.11 | 0.88 | **0.21** |
| **GraphRAG (Vector + Cypher)** | 3.91 | 3.88 | 4.26 | **0.88** | **0.21** |

Table 1: Mean performance metrics across the evaluation set. Accuracy, Completeness, and Usefulness are reported on a 1-5 Likert scale (higher is better). BERTScore and ROUGE-L measure semantic and lexical similarity to the ground truth, respectively.

| **Company** | **LLM** | **Naive** | **GraphRAG** |
|---|---|---|---|
| $Company_1$ | 3.83 | 3.80 | **3.93** |
| $Company_2$ | **4.37** | 4.01 | 4.09 |
| $Company_3$ | **4.04** | 3.37 | 3.34 |
| $Company_4$ | **3.82** | 3.60 | 3.72 |

Table 2: Mean Accuracy scores for selected companies.

cases. Notably, for $Company_1$, the GraphRAG approach outperforms the LLM baseline (3.93 vs. 3.83). Conversely, cases like $Company_3$ show a significant drop for RAG methods (Accuracy $\approx$ 3.34 vs LLM 4.04), highlighting challenges in retrieval for certain document structures where the graph construction may have been sparse or fragmented.

In conclusion, while a powerful LLM can hallucinate correct answers for known entities, the **GraphRAG** pipeline ensures the necessary grounding in the source text (ROUGE-L 0.21) and provides a richer, more complete context than naive vector retrieval. Moreover, it is important to note that the current ontology covers only a portion of the information needed to build the primer, whereas the LLM can leverage its full knowledge. Q&A pairs are extracted from the whole primer, and it can happen that some questions are partially or fully related to knowledge not already modelled in the KG.

## 4. Conclusions and future works

The integration of Knowledge Graphs (KGs) and Large Language Models (LLMs) through a specialized GraphRAG pipeline represents a significant step toward automating complex strategic analysis. This research demonstrates that while powerful LLMs possess substantial parametric knowledge regarding public companies, they lack the precision and grounding required for professional strategy consulting. By adopting an ontology-first approach, in which domain experts have defined a rigorous schema, the proposed methodology ensures that generated content remains anchored in a validated conceptual framework, effectively mitigating the risks of hallucination and lack of traceability.

Our experimental evaluation confirms the technical advantages of leveraging graph structures. The Vector + Cypher retrieval strategy consistently outperformed naive vector-only methods in completeness and usefulness, proving that 1-hop graph traversals successfully capture the interconnected nature of strategic risks, drivers, and market dynamics. Furthermore, the methodology achieved a ROUGE-L score three times higher than the LLM-only baseline, highlighting its necessity in environments where strict adherence to source documents is mandatory. While these automated metrics offer valuable directional insights, we recognize the inherent complexities of relying solely on LLM-based evaluation for qualitative assessment. To further contextualize these results, as for the next steps, we plan on including an *in vivo* evaluation with business strategy experts.

Future work will expand the system's scope beyond the current "outside-in" analytical perspective. We plan to incorporate strategy execution planning into the framework, which was excluded from the current study. This expansion involves extending the ontology to include additional strategic dimensions, specifically value proposition design, capability planning, and execution orchestration. Ultimately, these enhancements are intended to enable the Knowledge Graph to support the construction of entire strategic Primers.

## 5. Acknowledgements

This work is supported by the Apulia Region through the project SIA LABS funded by the PR FESR-FSE+ 2021-2027 "Applying Generative AI and Knowledge Graphs to Business Strategy" (CUP: B96I25000090007). Eleonora Ghizzota acknowledges the Ph.D. fellowship within the framework of the Italian Ministerial Decree (D.M.) n. 630, date of D.M.: 24.04.2024 - under the National Recovery and Resilience Plan, Mission 4, Component 2, Investment 3.3 - Ph.D. Project 'Development and application of Generative Artificial Intelligence models based on the Symbiotic AI paradigm to support the Public Administration', co-supported by company InnovaPuglia S.p.A. (CUP B91I24000240007).

## 6. Bibliographical References

Mohammad Sadeq Abolhasani and Rong Pan. 2024. Leveraging LLM for automated ontology extraction and knowledge graph generation. In *Proceedings of the Annual Reliability and Maintainability Symposium (RAMS)*. IEEE.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. 2021. Knowledge graphs. *ACM Computing Surveys*, 54(4):1–37.

Yuntong Hu, Zhihan Lei, Zheng Zhang, Bo Pan, Chen Ling, and Liang Zhao. 2024. Grag: Graph retrieval-augmented generation.

Sangyeop Kim, Junguk Ha, Hangyeul Lee, Sohhyung Park, and Sungzoon Cho. 2026. Human-guided collective llm intelligence for strategic planning via two-stage information retrieval. *Information Processing and Management*, 63:104288.

Vamsi Krishna Kommineni, Birgitta König-Ries, and Sheeba Samuel. 2024. From human experts to machines: An LLM supported approach to ontology and knowledge graph construction. In *Proceedings of the AAAI 2024 Spring Symposium on Empowering Machine Learning and Large Language Models with Domain and Commonsense Knowledge*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Kunze Li and Yu Zhang. 2024. Planning first, question second: An llm-guided method for controllable question generation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4715–4729.

Anna Sofia Lippolis, Mohammad Javad Saeedizade, Robin Keskïsärkkä, Sara Zuppiroli, Miguel Ceriani, Aldo Gangemi, Eva Blomqvist, and Andrea Giovanni Nuzzolese. 2025. Ontology generation using large language models. In *Proceedings of the Extended Semantic Web Conference (ESWC)*.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*.

Natalya F. Noy and Deborah L. McGuinness. 2001. Ontology development 101: A guide to creating your first ontology. Technical Report KSL-01-05, Stanford Knowledge Systems Laboratory.

Abdulsobur Oyewale and Tommaso Soru. 2026. LLM-driven ontology construction for enterprise knowledge graphs. In *Proceedings of the IEEE International Conference on Semantic Computing*.

Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, and Siliang Tang. 2024. Graph retrieval-augmented generation: A survey.

Michael E. Porter. 1979. How competitive forces shape strategy. *Harvard Business Review*, 57(2):137–145.

Richard W. Puyt, Finn Birger Lie, and Celeste P.M. Wilderom. 2023. The origins of SWOT analysis. *Long Range Planning*, 56(3):102304.

Mohammad Javad Saeedizade and Eva Blomqvist. 2024. Navigating ontology development with large language models. In *Proceedings of the Extended Semantic Web Conference (ESWC)*. Springer.

Giovanni Scarso-Borioli, Davide Sola, Roberto Quaglia, and Alex Jordan. 2024. Towards a machine-readable standard ontology for business strategy. ESCP Business School Impact Paper. Available at SSRN: `https://ssrn.com/abstract=5467414`.

Kristján Vigfússon, Lára Jóhannsdóttir, and Snjólfur Ólafsson. 2021. Obstacles to strategy implementation and success factors: A review of empirical literature. *Strategic Management*, 26(2):12–30.

Bohui Zhang, Valentina Anita de Berardinis, Valentina Anita Carriero, Katrin Schreiberhuber, Stefani Tsaneva, Lucía Sánchez González, Jongmo Kim, and Jacopo de Giacomo. 2024. OntoChat: a framework for conversational ontology engineering using language models. In *Proceedings of the Extended Semantic Web Conference (ESWC)*. Springer.

Qinggang Zhang, Shengyuan Chen, Yuanchen Bei, Zheng Yuan, Huachi Zhou, Zijin Hong, Hao Chen, Yilin Xiao, Chuang Zhou, Junnan Dong, Yi Chang, and Xiao Huang. 2025. A survey of graph retrieval-augmented generation for customized large language models.