GSS: Graph Semantic Summarization Using Answer-Centered Explanatory Subgraphs

Valantis Zervos[2,3], Spyridon Doukeris[2,3], Kiki Miniadou[2,3], Giannis Vassiliou[4], Sophia Sideri[1,2] and Haridimos Kondylakis[2,3]

[1]LIPADE, Université Paris Cité, Paris, France
[2]CSD, University of Crete, Heraklion, Greece
[3]FORTH-ICS, Heraklion, Greece
[4]Department of Electrical and Computer Engineering, HMU, Heraklion, Greece

Abstract

Knowledge graphs support large-scale structured knowledge representation, but their size and semantic complexity make query-driven access and summarization challenging. Public resources such as DBpedia and Wikidata contain billions of triples, complicating the extraction of contextually meaningful information for a given query. We present GSS (Graph Semantic Summarization), a query-driven framework for constructing *answer-centered explanatory subgraphs* (ACE) from large knowledge graphs. Given a natural language query, GSS combines large language models with symbolic SPARQL querying to identify salient entities, retrieve relevant triples, and assemble compact subgraphs centered on answer-bearing entities while preserving explanatory context. We evaluate GSS on the QALD-9-plus benchmark using gold-standard answers to query DBpedia and Wikidata. The evaluation combines answer-containment, ranking, and intrinsic graph-quality metrics to assess both correctness and explanatory structure. Experiments with two LLMs, show that GSS retrieves correct answers and produces semantically coherent summaries that enhance contextual understanding.

1. Introduction

A knowledge graph is a structured representation of real-world entities, such as people, places, objects, and abstract concepts, and the semantic relationships between them. By modeling information as nodes (entities) and edges (relations), knowledge graphs provide contextualized, machine-readable data that supports reasoning, semantic search, and intelligent applications such as question answering systems and AI assistants.

Motivation. Knowledge graphs have become a fundamental infrastructure for organizing and accessing structured knowledge across a wide range of domains, including the Semantic Web, digital libraries, scientific data management, and intelligent assistants. Modern knowledge graphs often contain millions or billions of triples, enabling expressive semantic querying but also making effective information access increasingly challenging. Users are rarely interested in exhaustive query results; instead, they seek concise, interpretable answers accompanied by contextual information that explains and situates those answers within the broader graph. This need is especially pronounced in interactive and query-driven settings, where summaries must be generated on demand and adapted to diverse information needs.

Problem Statement. Knowledge Graph Summarization (KGS) addresses this challenge by producing compact and informative representations of large and complex knowledge graphs. Instead of exposing users to raw triples, KGS aims to reduce information overload by selecting the most relevant entities and relationships while preserving the semantic structure necessary for understanding [1]. This problem is especially critical in scenarios such as question answering, exploratory search, and decision support, where users require not only correct answers but also explanatory context that situates those answers within the broader graph. The challenge becomes even more pronounced when summaries must be generated dynamically in response to diverse natural language queries, without relying on historical usage patterns or predefined templates like previous works.

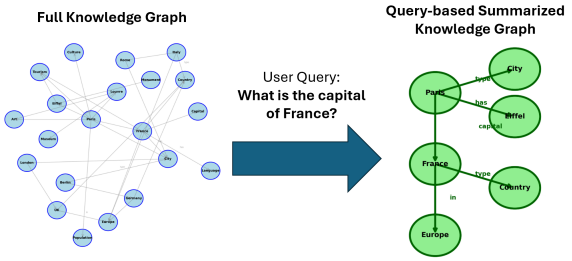

Figure 1: The extraction of an ACE subgraph from a KG in response to a user-defined question.

Formalization. Formally, given a knowledge graph KG with a set of triples T and a user query Q, the objective is not only to retrieve a subset of triples $A \subset T$ that directly answer the query, but also to identify a set of relevant triples $R \subset T$ that together form a subgraph $S \subset KG$. This subgraph should be compact, semantically coherent, and centered around the ACE triples, providing explanatory context that supports interpretation of the answer. For example, given the question $Q =$ *"What is the capital of France?"*, the goal is to construct a subgraph S that includes the ACE triple *<Paris, capitalOf, France>* along with additional contextual triples such as *<Paris, type, City>* or *<France, locatedIn, Europe>*, as illustrated in Figure 1.

Usefullness. From the perspective of large language models (LLMs), answer-centered explanatory subgraphs provide a principled mechanism for grounding natural language generation in explicit symbolic evidence. By constraining the model's input to a compact, query-focused subgraph centered on ACE triples, the reasoning space is reduced to semantically relevant entities and relations, improving controllability and mitigating spurious correlations that may arise from unstructured or excessively large contexts. This structured grounding enables hybrid neuro-symbolic setups in which LLMs are responsible for semantic interpretation and language generation, while correctness and contextual consistency are enforced through graph-based retrieval [2]. As a result, the generated answers can be more faithfully aligned with the underlying knowledge graph, supporting

Published in the Proceedings of the Workshops of the EDBT/ICDT 2026 Joint Conference (March 24-27, 2026), Tampere, Finland
*Corresponding author.

transparent reasoning, reduced hallucination, and explainable outputs that can be traced back to explicit triples and short relational paths within the subgraph

Limitations of Existing Approaches. Existing approaches to knowledge graph summarization typically rely on graph-theoretic measures, handcrafted heuristics, predefined summary templates, or historical query workloads [3]. While effective in constrained settings, these methods often struggle to adapt to diverse natural language queries or to accurately capture user intent. Moreover, approaches that depend on interaction logs or query histories [3] assume the availability of large and representative workloads—an assumption that rarely holds for long-tail entities, emerging knowledge graphs, or specialized domains. As a result, generating meaningful summaries for rare entities or previously unseen information needs remains an open challenge.

Large language models (LLMs) can address these limitations. LLMs can interpret natural language queries and can extract semantically salient queries from user input, without requiring prior query logs. When combined with symbolic querying over knowledge graphs, LLMs can support the construction of query-focused summaries that balance correctness with explanatory richness.

Our Approach. Motivated by these observations, we propose **GSS (Graph Semantic Summarization)**, a query-driven framework for constructing *answer-centered explanatory subgraphs* (ACE) from large KGs. GSS integrates LLM-based keyword extraction with symbolic SPARQL querying to identify ACE triples and organize surrounding contextual information into compact, query-focused summaries. Rather than relying on historical query logs or predefined templates, GSS operates on demand, producing summaries that are anchored around correct answers while enriching them with semantically relevant context.

We demonstrate GSS in real KGs using DBpedia and Wikidata. The system is evaluated with the QALD-9-plus benchmark, leveraging gold-standard answer entities as anchor points to test whether generated summaries contain the correct answers and structure additional explanatory information around them. We report answer-containment and ranking metrics alongside intrinsic graph-quality measures, capturing both correctness and explanatory coherence.

Contributions. The main contributions of this work are summarized as follows:

- We propose GSS, a query-driven knowledge graph summarization framework that constructs ACE subgraphs by integrating large language models with symbolic SPARQL querying over DBpedia and Wikidata.
- We introduce a hybrid scoring and ranking strategy that combines LLM-derived entity importance with embedding-based semantic similarity to prioritize answer-relevant and contextually meaningful triples.
- We present an evaluation methodology that treats summaries as ACE, assessing answer containment and intrinsic graph properties such as path consistency, density, and the presence of bridge entities.
- We empirically evaluate GSS on the QALD-9-plus benchmark, demonstrating that the framework retrieves correct answers while producing compact, semantically coherent subgraphs that enhance contextual understanding.

The remaining part of this paper is structured as follows: Section 2 presents related work, whereas our approach is presented in Section 3. Section 4 evaluates our approach, whereas Section 4.2 discusses the main findings. Finally, Section 5 concludes this paper.

2. Related Work

Following the taxonomy of Cebirić et al. [1], semantic graph summarization methods can be broadly classified into *quotient* and *non-quotient* approaches. Quotient-based methods merge nodes with similar types or structural properties to produce compact, schema-level summaries, but are inherently coarse-grained and unsuitable for query-specific or instance-level information needs. In contrast, non-quotient methods construct summaries as subgraphs of the original knowledge graph, retaining selected entities and relations based on relevance or task-specific criteria. Since GSS generates query-focused subgraphs that preserve original entities, it falls into the class of *structural, non-quotient* summarization approaches. Comprehensive surveys of the area are provided in [1, 4].

Early non-quotient summarization work includes the RDF Sentence Graph model of Zhang et al. [5], which constructs summaries based on overlapping RDF sentences guided by user-defined weights and preferences. While expressive, this approach requires manual parameter tuning and assumes explicitly specified relevance signals, limiting its applicability in open or large-scale settings.

Subsequent work explored personalized ontology summarization. Queiroz-Sousa et al. [6] rank ontology concepts using structural measures and connect them via the Broaden Relevant Paths algorithm, relying on user-defined constraints to guide personalization. However, these methods operate primarily at the schema level and require explicit user input, making them less suitable for ad-hoc or natural language queries.

More recent approaches incorporate usage information. GLIMPSE [7] maximizes inferred user utility based on representative query sets, but depends on users providing such queries and suffers from scalability issues on large graphs. Workload-based methods such as WBSUM [8] and iSummary [3] infer relevance from query logs to construct schema-level or selective summaries. While effective when rich workload data is available, these methods depend on large, representative query logs that are often unavailable or heavily skewed, limiting their effectiveness for non-popular or long-tail entities.

GSS addresses these limitations by adopting an *on-demand, query-driven* approach. Instead of relying on historical workloads or explicit user parameters, GSS leverages large language models to extract salient entities directly from a natural language query and uses symbolic SPARQL querying to assemble a compact, semantically coherent subgraph. This design eliminates the need for query logs, enabling effective summarization for rare entities and previously unseen information needs.

3. The GSS Approach

Problem Setting. Let $KG = (V, E)$ be a knowledge graph, where V is the set of entities, P the set of predicates, and $E \subseteq V \times P \times V$ is the set of RDF triples. Let T denote the set of all triples in a KG. Given a natural language query Q, the goal of GSS is to construct a subgraph $S = (V_S, E_S)$ such that $V_S \subseteq V$, $E_S \subseteq E$, and S is both compact and semantically aligned with the intent of Q. In this work,

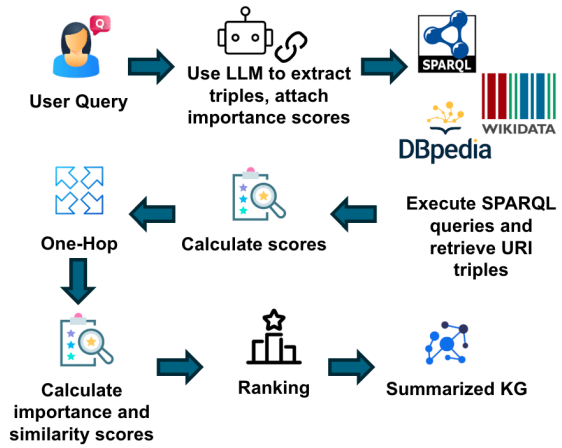

Figure 2: The Graph Semantic Summarizer (GSS) pipeline for constructing ACE subgraphs.

we focus on summaries that are explicitly centered around answer entities and enriched with explanatory context.

Definition 1 (ACE Subgraph). Given a query Q and a knowledge graph KG, an answer-centered explanatory (ACE) subgraph is a subgraph $S \subseteq KG$ that: (i) contains at least one ACE triple for Q, whenever such triples exist in KG, and (ii) includes additional triples that are semantically connected to the answer entities and provide explanatory context for interpreting the answer.

Definition 2 (Entity Importance). Given a query Q, an entity importance function $I_Q : V \to [0, 1]$ assigns to each entity a score reflecting its relevance to Q.

Definition 3 (Triple Importance). Given a triple $t = (s, p, o)$ and an entity importance function I_Q, the importance-based score of t is defined as $w_{\text{imp}}(t) = \max\{I_Q(s), I_Q(o)\}$, capturing the intuition that a triple is important if it involves at least one highly relevant entity.

Definition 4 (Semantic Similarity). Let $\phi(\cdot)$ be a sentence embedding function that maps text to a vector space. The semantic similarity between a query Q and a triple t is defined as $w_{\text{sim}}(t) = \cos(\phi(Q), \phi(t))$, where $\phi(t)$ is computed from a textual serialization of the triple.

Definition 5 (Triple Relevance). The overall relevance of a triple t to a query Q is defined as a convex combination $w(t) = \alpha \cdot w_{\text{imp}}(t) + (1 - \alpha) \cdot w_{\text{sim}}(t)$, where $\alpha \in [0, 1]$ controls the trade-off between entity-centric importance and semantic alignment.

3.1. The GSS Algorithm

Algorithm 1 presents the overall procedure executed by GSS for a single query, producing an ACE subgraph. In the sequel, we explain Algorithm 1 step by step, following the pipeline shown in Figure 2.

Given a natural language query Q, GSS first extracts a set of salient entity and concept mentions E_Q using an LLM (line 1). These mentions are assigned importance scores through the function I_Q (line 2), which reflects their relevance to the query intent as defined above. The output of this step is a mapping from KG entities to normalized importance values in $[0, 1]$. The extracted mentions are then grounded to the knowledge graph by resolving them to KG

Algorithm 1 Graph Semantic Summarization (GSS)

Require: $KG = (V, E)$, natural language query Q, relevance weight $\alpha \in [0, 1]$, size budget $B \geq 1$
Ensure: ACE subgraph $S = (V_S, E_S)$
1: $E_Q \leftarrow \text{ExtractEntities}(Q)$ *// LLM-assisted*
2: $I_Q \leftarrow \text{AssignImportance}(E_Q)$ *// Scores*
3: $U \leftarrow \text{ResolveURIs}(E_Q, KG)$ *// Map entities to IDs*
4: $T_0 \leftarrow \text{RetrieveTriples}(U, KG)$ *// ACE triples*
5: $T_1 \leftarrow \text{OneHopExpand}(T_0, KG)$ *// Expand around answer candidates for explanatory context*
6: $T \leftarrow T_0 \cup T_1$
7: **for all** $t = (s, p, o) \in T$ **do**
8: $\quad w_{\text{imp}}(t) \leftarrow \text{ImportanceScore}(t, I_Q)$
9: $\quad w_{\text{sim}}(t) \leftarrow \text{SemanticSimilarity}(t, Q)$
10: $\quad w(t) \leftarrow \alpha \cdot w_{\text{imp}}(t) + (1 - \alpha) \cdot w_{\text{sim}}(t)$
11: **end for**
12: $T' \leftarrow \text{SelectTop}(T, w, B)$ *// Top-k ACE triples*
13: $E_S \leftarrow \{s, o \mid (s, p, o) \in T'\}$
14: $E_S \leftarrow \text{Deduplicate}(E_S)$
15: **return** $S = (E_S, T')$

identifiers, yielding the set U (line 3). Using these identifiers, GSS retrieves an initial set of triples T_0 that involve the resolved entities (line 4), ensuring that ACE triples are considered early in the process.

To provide explanatory context around answer candidates, GSS performs a controlled one-hop expansion over the neighborhood of T_0, producing an additional set of triples T_1 (line 5). The candidate pool $T = T_0 \cup T_1$ (line 6) balances contextual coverage with compactness by restricting expansion depth.

For each candidate triple $t = (s, p, o) \in T$ (line 7), GSS computes two complementary relevance signals. The importance-based score $w_{\text{imp}}(t)$ (line 8) is derived from the entity importance values $I_Q(s)$ and $I_Q(o)$, capturing the degree to which the triple involves answer-relevant entities. In parallel, the semantic similarity score $w_{\text{sim}}(t)$ (line 9) measures the embedding-based semantic alignment between the query Q and the textual representation of t.

These signals are combined into a single relevance score $w(t) = \alpha \cdot w_{\text{imp}}(t) + (1 - \alpha) \cdot w_{\text{sim}}(t)$ (line 10), where α is a user-configurable parameter controlling the relative emphasis on entity importance versus semantic similarity. In our experiments, α is fixed across queries to ensure deterministic and reproducible behavior.

After scoring all candidate triples (lines 7–11), GSS selects the top-ranked subset T' under the specified size budget B (line 12). The entity set of the output subgraph is defined as the set of subjects and objects appearing in T' (line 13). Finally, duplicate or equivalent triples are removed to reduce redundancy (line 14), and the resulting ACE subgraph $S = (E_S, T')$ is returned (line 15).

Example 1. We illustrate the behavior of GSS using the running example query: $Q =$ "What is the capital of France?". **Entity Extraction and Importance Assignment.** In Line 1 of Algorithm 1, GSS utilizes LLMs to identify salient entities from the query. In this case, the extracted set is $E_Q = \{\textit{France}, \textit{capital}\}$. Then, Line 2 assigns normalized importance scores reflecting query relevance. The explicitly mentioned entity *France* receives the highest importance score $I_Q(\textit{France}) = 1.0$, while the concept *capital* is treated as a strongly implied relation indicator $I_Q(\textit{capital}) = 0.8$. **URI Resolution and Initial Triple Retrieval.** In Line 3, entities are resolved to knowledge graph identifiers. For

DBpedia, *France* is mapped to `dbr:France`. Using this identifier, GSS retrieves an initial set of triples T_0 (Line 4) that involve `dbr:France`:

$$T_0 = \left\{ \begin{array}{l} \langle \texttt{Paris}, \texttt{capitalOf}, \texttt{France} \rangle, \\ \langle \texttt{France}, \texttt{locatedIn}, \texttt{Europe} \rangle, \\ \langle \texttt{France}, \texttt{populationTotal}, 67{,}000{,}000 \rangle, \ \ldots \end{array} \right\}$$

Among these, the triple $\langle \texttt{Paris}, \texttt{capitalOf}, \texttt{France} \rangle$ is an ACE triple.

One-Hop Expansion for Explanatory Context. To enrich the answer with explanatory context, GSS performs a one-hop expansion over T_0, producing an additional set T_1 (Line 5). This expansion retrieves triples connected to entities appearing in T_0, such as:

$$T_1 = \left\{ \begin{array}{l} \langle \texttt{Paris}, \texttt{type}, \texttt{City} \rangle, \\ \langle \texttt{Paris}, \texttt{locatedIn}, \texttt{France} \rangle, \\ \langle \texttt{Europe}, \texttt{type}, \texttt{Continent} \rangle, \ \ldots \end{array} \right\}$$

The candidate triples are formed as $T = T_0 \cup T_1$ (Line 6). We empirically observed that further expansions rapidly increase noise without improving explanatory coherence.

Triple Scoring. For each triple $t \in T$, GSS computes two relevance signals. The importance-based score $w_{\text{imp}}(t)$ (line 8) is derived from the importance values of the entities participating in the triple. For example: $w_{\text{imp}}(\langle \texttt{Paris}, \texttt{capitalOf}, \texttt{France} \rangle) = \max\{I_Q(\texttt{Paris}), I_Q(\texttt{France})\} = 1.0$. In contrast, a peripheral triple such as $\langle \texttt{France}, \texttt{populationTotal}, 67{,}000{,}000 \rangle$ receives a lower importance-based score, as it does not directly support the query intent.

Following, GSS computes the semantic similarity score $w_{\text{sim}}(t)$ (Line 9) by embedding both the query Q and a textual serialization of t into a vector space and computing cosine similarity. The final score $w(t)$ (Line 10) is computed as a combination of importance and semantic similarity.

Ranking and Output Subgraph Construction. After scoring all triples, GSS selects the top-ranked subset T' under the size budget B (Line 12). In this example, T' contains:

$$\left\{ \begin{array}{l} \langle \texttt{Paris}, \texttt{capitalOf}, \texttt{France} \rangle, \\ \langle \texttt{Paris}, \texttt{type}, \texttt{City} \rangle, \\ \langle \texttt{France}, \texttt{locatedIn}, \texttt{Europe} \rangle \end{array} \right\}$$

The resulting entity set V_S (Line 13) includes `Paris`, `France`, and `Europe`. After deduplication (Line 14), the final output is an ACE subgraph that is anchored around the correct answer `Paris` and augmented with minimal yet semantically coherent contextual information explaining its role and location.

3.2. Properties of the GSS Algorithm

Obviously, the explanatory subgraph produced by GSS depends only on the input query Q and the underlying knowledge graph KG, and does not rely on historical query logs. We now discuss key properties of GSS that characterize its behavior and suitability for ACE summarization. These properties follow directly from the design of Algorithm 1

Lemma 1 (Answer Anchoring)**.** *Whenever an ACE triple exists in KG for a query Q, GSS prioritizes its inclusion in the resulting explanatory subgraph.*

Proof Sketch. ACE triples involve entities mentioned or implied by the query. Such entities receive high importance scores , which propagates to an important triple. In addition, ACE triples typically exhibit strong semantic similarity to the query. As a result, their combined relevance score is high, ensuring that they are ranked early and selected during the top-B selection step, provided they exist in KG. □

Lemma 2 (Long-Tail Coverage)**.** *GSS can construct ACE subgraphs for entities that have not appeared in prior queries, as long as they are present in the knowledge graph KG.*

Proof Sketch. Because GSS does not rely on historical query logs or usage statistics, entity relevance is inferred directly from the semantic content of the input query via LLM-based extraction. As long as an entity is present in KG and can be resolved during URI grounding, it can participate in triple retrieval and expansion. Consequently, entities from the long tail of the knowledge graph are treated on equal footing with frequently queried entities. □

Lemma 3 (Bounded Summary Size)**.** *For fixed configuration parameters, the size of the explanatory subgraph produced by GSS is upper-bounded by a constant independent of $|KG|$.*

Proof Sketch. Algorithm 1 enforces explicit limits on both expansion depth (one-hop expansion) and the number of retained triples via the size budget B. These parameters bound the number of triples that can be included in the output regardless of the total size of KG. Therefore, the size of the resulting explanatory subgraph is independent of $|KG|$ and depends only on configuration choices. □

Lemma 4 (Explanatory Coherence)**.** *All non-answer entities in the output subgraph are connected to at least one answer entity by a path of length at most two.*

Proof Sketch. All triples in the output subgraph originate from the initial retrieval step, which includes ACE triples, or from the one-hop expansion around those triples. Consequently, any non-answer entity appears either directly in an ACE triple or in a triple adjacent to such an entity. This guarantees that every non-answer entity is connected to at least one answer entity by a path of length at most two, ensuring explanatory coherence. □

Computational Complexity. Let k be the number of entities extracted from the query, d the average degree of an entity in KG, and m the number of retrieved triples after expansion. Entity extraction and scoring are constant with respect to $|KG|$. Triple retrieval and one-hop expansion require $O(k \cdot d)$ SPARQL pattern matches. Semantic similarity computation requires $O(m)$ embedding comparisons. Ranking and selection run in $O(m \log m)$ time. Overall, the algorithmic complexity of GSS with respect to the knowledge graph is $O(k \cdot d + m \log m)$, excluding the inference latency of the LLM model. Since k, d, and m are bounded by configuration parameters, GSS operates independently of the total size of the knowledge graph, making it suitable for large-scale deployments.

4. Evaluation

The objective of the evaluation is to assess whether GSS produces effective *Answer-Centered Explanatory (ACE) subgraphs*, i.e., compact graph structures that (i) contain correct answer entities for a given query and (ii) organize additional contextual information around them in a semantically coherent and interpretable manner. To this end, we adopt

a multi-layer evaluation framework that jointly assesses answer faithfulness and explanatory structure.

Evaluation Setup. All experiments are conducted on CPU-only hardware using pre-trained transformer models to ensure reproducibility. We evaluate GSS under two LLM configurations, namely `gemini-2.5-flash-lite` and `llama-3.1-8b-instant`. For each query, GSS produces a ranked list of candidate RDF triples. To simulate a realistic summarization setting while maintaining computational feasibility, only the top $K = 15$ ranked triples per query are retained. These candidates are subsequently reranked using a cross-encoder model (`cross-encoder/ms-marco-MiniLM-L-6-v2`), which jointly encodes the natural language query and a textual representation of each triple. The reranked triples constitute the final ACE subgraph evaluated for each query.

Datasets and Benchmarks. We evaluate GSS on DBpedia and Wikidata using the QALD-9-plus benchmark. QALD-9-plus provides natural language questions paired with gold-standard SPARQL queries and answer bindings, but does not define gold-standard summaries or contextual graphs. As a result, QALD cannot be used to directly assess the correctness of a generated summary as a whole. Instead, gold answer entities are treated as *anchor points*, and the evaluation focuses on whether the produced subgraphs (i) contain correct answer entities and (ii) organize additional information around them in a coherent explanatory structure. Gold answer entities are extracted from the QALD JSON files by retaining the corresponding DBpedia and Wikidata resource URIs. Questions whose answers consist exclusively of literals or empty bindings are excluded to ensure compatibility with graph-based analysis.

Evaluation Metrics. Our evaluation metrics are organized into two conceptual layers, reflecting the dual objective of ACE summarization.

Layer 1: Answer Faithfulness. These metrics assess whether the generated subgraphs contain correct answer entities and whether they are ranked early:

- **Precision, Recall, and F1-score**, computed over predicted entities.
- **Mean Reciprocal Rank (MRR)**, based on the rank of the first ACE triple.
- **Hits@k**, measuring whether at least one gold answer entity appears within the top-k ranked triples.

These metrics answer the question: *Does the summary contain the correct answer entities, and are they prioritized early in the ranking?*

Layer 2: Explanatory Structure and Answer Containment. These metrics evaluate how effectively the subgraph organizes contextual information around answer entities:

- **Path Consistency**, capturing whether non-gold entities participate in coherent paths leading to an answer entity.
- **Graph Density**, measuring the structural compactness of the subgraph as the ratio of observed triples to the maximum possible number of triples between its entities; higher values indicate more tightly connected summaries.
- **Bridge Entities**, defined as non-gold entities that lie on at least one path connecting another entity in the subgraph to a gold answer entity, capturing intermediate entities that support explanatory connectivity.
- **AvgDist**, measuring the average shortest-path distance from non-gold entities to the nearest gold answer entity (lower is better). Let $G_s = (V_s, E_s)$ be the generated subgraph for a query, $V_g \subseteq V_s$ the set of gold answer entities, and $V_{ng} = V_s \setminus V_g$ the non-gold entities. AvgDist is defined as:

$$\text{AvgDist} = \frac{1}{|V_{ng}|} \sum_{v \in V_{ng}} \min_{u \in V_g} \text{dist}_{G_s}(v, u)$$

Together, these metrics assess whether the generated subgraphs function as explanations rather than mere containers of answer entities. For summaries consisting of a single triple, several structural metrics become degenerate by construction and should therefore be interpreted with care.

Competitors. We compare GSS against **iSummary** [3], a workload-aware approach that constructs query-focused summaries using historical query logs. Unlike GSS, iSummary is deterministic and always returns a triple patterns appearing in the queries. As a result, iSummary is optimized for answer retrieval rather than explanation, serving as a strong baseline for answer faithfulness but a limited one for explanatory structure.

4.1. Results

The evaluation results are summarized in Table 1, while the ablation study on the relevance weighting parameter α is reported in Table 2.

Comparison. Across both DBpedia and Wikidata, GSS achieves moderate precision and recall, reflecting its design choice to include contextual entities in addition to answer entities. Other metrics such as MRR and Hits@k show that correct answers are prioritized among the top-ranked triples, demonstrating effective alignment between the semantic intent of the query and the extracted graph content. In contrast, iSummary achieves higher Precision, MRR, and Hits@k scores. This is expected, as iSummary deterministically returns one or more triple patterns from an input query, which places the answer at rank one and optimizes ranking-based metrics. However, it significantly limits the and explanatory context, as no additional entities are introduced.

These limitations are reflected in the explanatory metrics. GSS achieves lower AvgDist values, indicating that contextual entities are tightly centered around gold answer entities through short explanatory paths. Moreover, the presence of bridge entities shows that GSS has intermediate nodes that introduce explanations rather than returning single answers. By contrast, iSummary does not have explanatory behavior due to the absence of multi-hop structure, resulting in higher AvgDist values and zero bridge entities.

Ablation Study. To examine GSS with respect to its relevance scoring, we conduct an ablation study by changing the weighting parameter α that controls the trade-off between entity importance and semantic similarity. Results are shown in Table 2 for $\alpha \in \{0, 0.25, 0.5, 0.75, 1\}$. When $\alpha = 0$, it relies exclusively on semantic similarity, producing semantically related triples but weak summaries, as evidenced by higher AvgDist values and a reduced number of bridge entities. In contrast, $\alpha = 1$ prioritizes the importance of the entity alone, leading to smaller summaries with limited explanatory content.

Intermediate values consistently outperform the two corner cases. Values in the range $\alpha \in [0.5, 0.75]$ have the best balance between answer faithfulness and explanatory structure, minimizing AvgDist while maximizing the presence of bridge entities. This behavior confirms that effective ACE

Table 1
Evaluation results.

Metric	DBpedia			Wikidata		
	Ours - Gemini	Ours - LLaMA	iSummary	Ours - Gemini	Ours - LLaMA	iSummary
Precision	0.248	0.205	**0.82**	0.221	0.178	**0.78**
Recall	**0.365**	0.315	0.14	**0.332**	0.282	0.12
F1-score	0.223	0.192	**0.24**	0.201	0.168	**0.21**
MRR	0.466	0.395	**0.78**	0.431	0.361	**0.74**
Hits@1	0.400	0.350	**0.72**	0.365	0.315	**0.69**
Hits@3	0.533	0.480	**0.86**	0.498	0.445	**0.83**
Path Consistency	0.578	0.515	**1.00**	0.542	0.480	**1.00**
Graph Density	0.967	0.905	**1.00**	0.932	0.870	**1.00**
Bridge Entities	**0.600**	0.545	0.00	**0.565**	0.510	0.00
AvgDist (↓ better)	**1.62**	2.05	3.10	**1.88**	2.34	3.45
# Avg. Triples	**3.88**	3.38	1.00	**3.52**	3.05	1.00

Table 2
Ablation study on α (Definition 5) in DBpedia.

α	F1	MRR	Bridge Ent. ↑	AvgDist ↓
0.0	0.22	0.38	0.30	2.46
0.25	0.26	0.42	0.44	2.01
0.5	0.28	0.45	0.56	1.71
0.75	**0.29**	**0.47**	**0.61**	**1.60**
1.0	0.25	0.44	0.28	2.12

summarization requires considering both entity importance and semantic alignment. Based on these observations, we fix $\alpha = 0.6$ for the experiments of Table 1.

Overall, the results highlight a distinction between answer retrieval and answer-centered explanation. While iSummary excels at identifying correct answers, GSS produces compact yet structured subgraphs that contextualize answers within semantically coherent explanations.

4.2. Discussion

The evaluation highlights a clear trade-off between answer retrieval and explanation generation. Methods such as iSummary, which deterministically return a single triple, are highly effective at surfacing correct answers and therefore perform well on ranking-based metrics such as MRR and Hits@k. However, these methods inherently lack the ability to contextualize answers, resulting in poor recall and the absence of explanatory structure.

In contrast, GSS is explicitly designed to produce answer-centered explanatory subgraphs. By incorporating contextual entities and multi-hop connections, GSS sacrifices strict answer compactness in favor of interpretability and semantic coherence. This design choice is reflected in higher recall, lower AvgDist values, and the consistent presence of bridge entities, indicating that supporting information is meaningfully organized around the answer.

Importantly, several structural metrics become degenerate for single-triple summaries and should not be interpreted as indicators of explanation quality in such cases. Metrics such as AvgDist, path consistency, and bridge entities are therefore critical for distinguishing between answer-centric retrieval and explanation-oriented summarization.

5. Conclusions

We introduced GSS, a query-driven approach for generating Answer-Centered Explanatory (ACE) subgraphs over large knowledge graphs. Unlike traditional answer retrieval or extractive summarization methods, GSS produces compact yet structured subgraphs that not only contain correct answers but also provide meaningful contextual explanations. Through a comprehensive evaluation on DBpedia and Wikidata using the QALD-9-plus benchmark, we demonstrated that GSS effectively balances answer faithfulness with explanatory structure. While baseline methods such as iSummary achieve strong performance on ranking-based metrics, they fail to provide explanatory context due to their single-triple output. In contrast, GSS consistently generates multi-hop subgraphs that position contextual entities close to gold answers, improving interpretability and supporting downstream analysis. These results suggest that answer-centered explanation should be treated as a distinct objective from answer retrieval, requiring dedicated evaluation metrics and summarization strategies.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] Š. Čebirić, F. Goasdoué, H. Kondylakis, D. Kotzinos, I. Manolescu, G. Troullinou, M. Zneika, Summarizing semantic graphs: a survey, The VLDB journal 28 (2019) 295–327.

[2] B. Skrlj, B. Koloski, S. Pollak, N. Lavrac, From symbolic to neural and back: Exploring knowledge graph-large language model synergies, CoRR abs/2506.09566 (2025).

[3] G. Vassiliou, F. Alevizakis, N. Papadakis, H. Kondylakis, isummary: Workload-based, personalized summaries for knowledge graphs, ESWC, 2023.

[4] K. Kellou-Menouer, N. Kardoulakis, G. Troullinou, Z. Kedad, D. Plexousakis, H. Kondylakis, A survey on semantic schema discovery, VLDBJ (2022).

[5] X. Zhang, G. Cheng, Y. Qu, Ontology summarization based on rdf sentence graph, in: WWW, 2007.

[6] P. O. Queiroz-Sousa, A. C. Salgado, C. E. S. Pires, A method for building personalized ontology summaries, J. Inf. Data Manag. 4 (2013) 236–250.

[7] T. Safavi, C. Belth, L. Faber, D. Mottin, E. Müller, D. Koutra, Personalized knowledge graph summarization: From the cloud to your pocket, in: ICDM, 2019.

[8] G. Vassiliou, G. Troullinou, N. Papadakis, H. Kondylakis, Wbsum: Workload-based summaries for RDF/S kbs, SSDBM, 2021.