An Automated Framework to generate pairs of Natural Language Questions and SPARQL Queries from Common RDF Graph Patterns

Julio C. Rangel[1], Tarcisio Mendes de Farias[2], Yamamoto Yasunori[1,3], Ana Claudia Sima[2] and Norio Kobayashi[1]

[1]*RIKEN Information R&D and Strategy Headquarters, 2-1 Hirosawa, 351-0198 Wakoshi, Japan*

[2]*SIB Swiss Institute of Bioinformatics, Lausanne, Switzerland*

[3]*Database Center for Life Science, Joint Support-Center for Data Science Research, Research Organization of Information and Systems, 178-4-4 Wakashiba, Kashiwa, Chiba 277-0871, Japan*

Abstract

This paper presents an approach for automatically generating SPARQL queries from natural language questions against any RDF dataset without requiring example queries or domain-specific training data. Our approach identifies frequently occurring relational paths through a community detection process on the RDF graph, thereby extracting canonical class-level patterns that occur with high frequency. Using these patterns, we produce a rich dataset of natural language questions and automatically generated SPARQL queries. This dataset is then indexed to enable similarity search of relevant natural language questions paired with their SPARQL queries, aligning with user-input questions. During inference, the system uses a semantic index to retrieve existing similar questions and SPARQL queries. If the retrieved question differs within a defined threshold, an entity linking mechanism replaces entities found in the user questions with those in the retrieved SPARQL queries, to ensure flexibility and domain independence.

Keywords

Knowledge Graph, SPARQL Query Generation, Natural Language Querying, Large Language Models, Community Detection

1. Introduction

In many knowledge graphs (KGs), certain relational and class-level patterns appear to occur more frequently than others. By focusing on these prevalent paths, it may be possible to better ground the generation of questions and queries in the most representative aspects of the data. Several recent works have explored methods to produce SPARQL queries from natural language [1, 2] or to leverage LLMs for query construction [3]. While these approaches often rely on annotated data or heuristics to narrow down candidate triples, our framework differs by systematically identifying frequently occurring class-level paths through a community detection algorithm. These paths are then used as a foundation to generate natural language questions and corresponding SPARQL queries, ensuring that the training data for query generation is strongly tied to the actual structural patterns in the graph.

KGQA has progressed with LLMs and retrieval-augmented strategies. Early methods relied on extensive data or engineered pipelines [4], while recent approaches use few-shot prompting [5, 6, 7] or retrieval to align queries with KG schemas [2]. Our framework enhances these by reducing dependence on pre-annotated queries or LLM memorization.

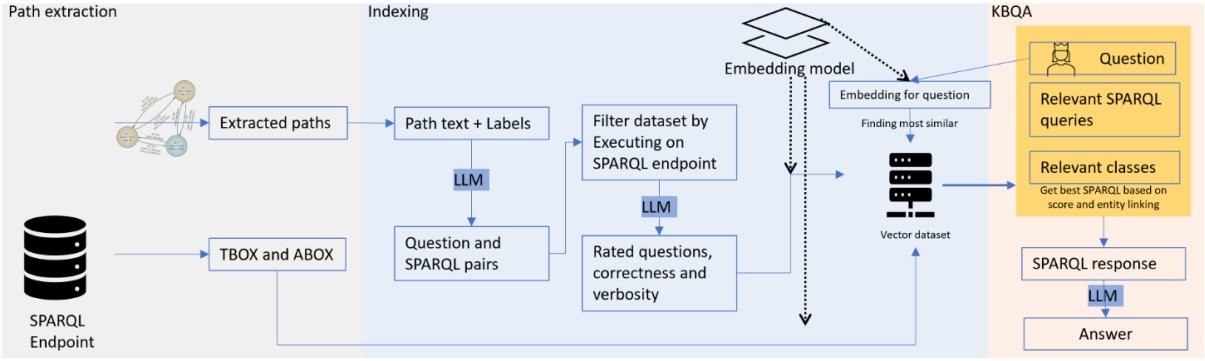
2. Methodology

Graph Preprocessing and Community Detection: A random selection of seed nodes is identified for each class, and from these nodes, we perform a breadth-first path enumeration up to a specified length k, focusing on non-schema relations to ensure the collected paths represent instance-level connections.

SWAT4HCLS 2025: 16th International Conference on Semantic Web Applications and Tools for Health Care and Life Sciences 2025

0000-0002-3175-5372 (T. Mendes de Farias); 0000-0002-6943-6887 (Y. Yasunori); 0000-0003-3213-4495 (A. C. Sima)

Figure 1: Framework architecture.



Collected paths are then abstracted to class-level patterns by replacing instance entities with their parent classes. Frequency analysis of these class-level paths yields a ranked list of common relational structures. These patterns form the backbone of our dataset construction, ensuring that generated queries target frequently encountered graph segments.

Generating Natural Language Questions and SPARQL Queries: Using the top-ranked class-level paths, we select representative instance paths and convert them into human-readable forms. Labels for entities and predicates enrich the readability of these prompts. We then feed these enriched prompts into an LLM (e.g., GPT-4o) with instructions to produce natural language questions closely tied to the provided path structure. The LLM also generates a candidate SPARQL query expected to retrieve data analogous to the provided instance path.

Multiple candidate question-query pairs are generated per path, and these outputs are filtered by executing the SPARQL queries on the KG endpoint. Queries that return non-empty results are retained, ensuring that only data-grounded pairs are included in the final dataset. The retained pairs are then rated using the LLM to identify the most relevant, clear, and well-structured questions. Only highly rated questions and their associated queries are included in the final compilation.

Constructing the Semantic Indexes and Query Processing Pipeline: The curated question-query pairs and their embeddings are stored in a vector index using a library such as Faiss[1]. This index facilitates rapid retrieval of semantically similar queries given a user's question. If no sufficiently similar questions are found, an additional index created from the T-Box and A-Box embeddings is used to identify relevant entities and classes. When the retrieved queries differ only in the entities involved, the entity linking mechanism replaces these entities accordingly in the SPARQL query, ensuring flexibility and domain independence. When a user submits a new query, the pipeline cleans and embeds the user input, finds the closest question-SPARQL pairs, and returns the associated SPARQL queries. The selected SPARQL query is executed, and results are passed to an LLM to produce a coherent and context-sensitive answer see Figure 1.

3. Discussion and Limitations

While LLM-based generation streamlines dataset creation, the approach's effectiveness may vary with model choice and the size of the underlying KG. For instance, smaller models like GPT-3.5 Turbo may struggle to create questions and SPARQL queries that successfully pass execution and data retrieval filtering. In contrast, advanced models like GPT-4o generate more pairs that consistently pass the filtering phase. Further experiments on a broader set of graphs are necessary to evaluate the generalizability of this technique. Additionally, implementing a user rating system would provide valuable feedback for refining the approach. We share the code at https://github.com/RIKEN-DKO/rdf_sensei.

[1] https://github.com/facebookresearch/faiss

Acknowledgments

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] C. V. S. Avila, M. A. Casanova, V. M. P. Vidal, A framework for question answering on knowledge graphs using large language models, in: A. Meroño Peñuela, O. Corcho, P. Groth, E. Simperl, V. Tamma, A. G. Nuzzolese, M. Poveda-Villalón, M. Sabou, V. Presutti, I. Celino, A. Revenko, J. Raad, B. Sartini, P. Lisena (Eds.), The Semantic Web: ESWC 2024 Satellite Events, Springer Nature Switzerland, Cham, 2025, pp. 168–172.

[2] V. Emonet, J. Bolleman, S. Duvaud, T. M. de Farias, A. C. Sima, Llm-based sparql query generation from natural language over federated knowledge graphs, 2025. URL: https://arxiv.org/abs/2410.06062. arXiv:2410.06062.

[3] L. Kovriguina, R. Teucher, D. Radyush, D. Mouromtsev, Sparqlgen: One-shot prompt-based approach for sparql query generation., in: Proceedings of the Posters and Demo Track of the 19th International Conference on Semantic Systems co-located with 19th International Conference on Semantic Systems, 2023.

[4] M. Sander, U. Waltinger, M. Roshchin, T. A. Runkler, Ontology-based translation of natural language queries to sparql, in: AAAI Fall Symposia, 2014. URL: https://api.semanticscholar.org/CorpusID:44247578.

[5] G. Xiong, J. Bao, W. Zhao, Y. Wu, X. He, Autoqgs: Auto-prompt for low-resource knowledge-based question generation from sparql, Proceedings of the 31st ACM International Conference on Information & Knowledge Management (2022). URL: https://api.semanticscholar.org/CorpusID:251881730.

[6] P. Agarwal, N. Kumar, S. Bedathur, SymKGQA: Few-shot knowledge graph question answering via symbolic program generation and execution, in: L.-W. Ku, A. Martins, V. Srikumar (Eds.), Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Bangkok, Thailand, 2024, pp. 10119–10140. URL: https://aclanthology.org/2024.acl-long.545/. doi:10.18653/v1/2024.acl-long.545.

[7] J. D'Abramo, A. Zugarini, P. Torroni, Dynamic few-shot learning for knowledge graph question answering, 2024. URL: https://arxiv.org/abs/2407.01409. arXiv:2407.01409.