TCRMQ: Question Answering by Multi-Hop SPARQL Queries over Event-Based Knowledge Graphs*

Michalis Mountantonakis[1,2][0000−0002−1951−0241] and Yannis Tzitzikas[1,2][0000−0001−8847−2130]

[1] Information Systems Laboratory, FORTH-ICS, Heraklion, Greece
[2] Computer Science Department, University of Crete, Heraklion, Greece
mountant@ics.forth.gr, tzitzik@ics.forth.gr

Abstract. There is an increasing trend of combining Knowledge Graphs (KGs) and LLMs for several tasks, including the generation of structured queries from natural questions. Indeed, it is quite challenging to answer natural questions over KGs, especially for a) sophisticated ontologies, since in most cases one has to derive multi-hop queries with property path expressions, and b) questions requiring extra background information that is not included in such KGs. Towards this direction, we present a research prototype, called `TCRMQ` (Text-2-CIDOC-CRM Query), that enables users to express their questions in natural language and to retrieve the desired answer over KGs using the event-based CIDOC-CRM model (an ISO standard used from many Cultural Heritage organizations). `TCRMQ` is based on a novel two-stage method combining Ontology Path Patterns and Knowledge from Large Language Models (LLMs) and can generate multi-hop SPARQL queries from natural language multilingual questions, that can even require external background knowledge to be answered. Its current version supports two CIDOC-CRM KGs containing artworks, and has been evaluated for English and Greek language through a dedicated benchmark. By applying this method over GPT-4 we achieve accuracy 83%, while the baseline has only 32% accuracy.

Keywords: LLMs, CIDOC-CRM, Text to SPARQL, Event Based KGs

1 Introduction

There is an increasing trend of combining Knowledge Graphs (KGs) and Large Language Models for several tasks and domains, e.g., Fact Validation [9], Question Answering [11,13], and many others. A challenging direction is the LLM to generate structured queries from natural questions over KGs, i.e., Text-2-SPARQL approaches [2,7,3], such as for DBpedia [6] and Wikidata [5]. The task

* This is a preprint of the demo paper: Michalis Mountantonakis and Yannis Tzitzikas, TCRMQ: Question Answering by Multi-Hop SPARQL Queries over Event-Based Knowledge Graphs, Accepted for Publication in the demo track of the 14th International Joint Conference on Knowledge Graphs (IJCKG 2025)

is more challenging for KGs using sophisticated ontologies, since in most cases one has to derive *multi-hop* queries with *property path* expressions (Challenge A). Also, a user can often express natural questions that require additional background information that is not included in the KG (Challenge B), e.g., missing facts or additional reasoning.

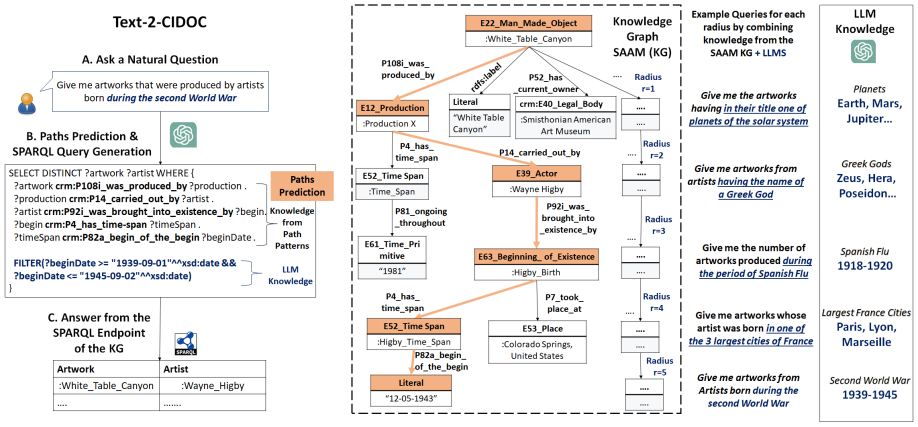

Fig. 1. A natural question requiring knowledge from SAAM KG and from the LLM

Toward this direction, we present a research prototype, called `TCRMQ` (from Text-to-CIDOC CRM Query), which enables users to express their questions in natural language (even in more languages apart from English) and to retrieve the answer over KGs that use the CIDOC-CRM model [4]. CIDOC-CRM is an event-based model and an ISO standard that is currently used by more than 400 Cultural Heritage (CH) organizations [8]. In Fig. 1, a real part of the SAAM (Smithsonian American Art Museum) KG [14] is shown. We can see long paths that we need to follow due to the event-nature of CIDOC-CRM. In the left side of Fig. 1, the user expressed the question "`Give me artworks that were produced by artists born during the second World War`". This question needs to be converted to SPARQL and it is required to follow a long path with radius (or path length) $r = 5$, for finding the artist of an artwork and his/her birth date (Challenge A). Additionally, we need external knowledge (e.g., see the right side) to answer the question (Challenge B). Specifically, the KG contains information about the artworks and artists of the museum, including their birth date. On the contrary, the KG does not contain information about the dates of Second World War, however, LLMs such as ChatGPT contain the mentioned dates.

To answer such questions and tackling both challenges, `TCRMQ` i) uses a *novel two-stage method* combining Ontology Path Patterns and Knowledge from LLMs [10], to predict the most relevant ontology paths and to formulate a multi-hop SPARQL query for a given natural question (not only in English but also in other languages), ii) evaluates the generated query by sending it to the SPARQL endpoint of the KG and iii) retrieves and presents the results to the user. For the question in Fig. 1 (Step A), `TCRMQ` predicted the most relevant paths (see

```
You are given the following triple patterns of the form  Class->Property->Class or Literal
#Patterns
E12_Production->P14_carried_out_by->E39_Actor
...
E52_Time_Span->P82a_begin_of_the_begin->Literal
#End of Patterns
Give me the classes and properties that will be used to answer the question:
I want artworks from artists born during the second world war
I strictly want the output in the following format: Classes: {} Properties:{}

LLM Output - Predictions
Classes:{E22_Man_Made_Object, E39_Actor, E52_Time_Span}
Properties:{P14_carried_out_by, P82a_begin_of_the_begin}
```

Fig. 2. Predictions Prompt and LLM Output

the orange parts on the right side) and generated a multi-hop SPARQL query from ChatGPT (Step B), that includes both the relevant ontology paths and the desired values for the dates in the FILTER (external knowledge from the LLM). Finally, the query was sent to the endpoint to obtain the response (Step C).

Regarding the contribution, we a) present the research prototype `TCRMQ` (https://demos.isl.ics.forth.gr/Text2CIDOC), by using the methods in [10], b) provide use cases and scenarios, and c) present evaluation results for 120 questions (by extending the benchmark in [10]) for three ChatGPT models, in English and Greek. Concerning the novelty, this is the first application for query formulation over CIDOC-CRM KGs by using LLMs, that can also yield queries with the right filtering conditions, even if such information is not present in the KG.

The rest of this paper is organized as follows: §2 describes the methods and the functionality of `TCRMQ`, §3 presents the use cases, §4 shows the evaluation results and §5 concludes the paper.

2 Methods and Functionality

As a pre-processing step, ontology path patterns up to a given radius r (or path length) are created once (e.g., $1 \leq r \leq 5$), by sending SPARQL queries to the endpoint of each desired KG. For instance, in Fig. 1, a path pattern of $r = 2$ is the following: ⟨E22_Man_Made_Object, P108_was_produced_by, E12_Production, P14_carried_out_by, E39_Actor⟩. Afterwards, the path patterns for each KG are stored to be used in the LLM prompts.

Methods. The demo supports several methods using such ontology path patterns and knowledge from the LLM. Also, a zero shot method is available where no further input is given to the LLM. For each method a single prompt template has been created, apart from the two-stage path patterns method, which sends two different prompts. All the details for the methods are given in [10]. The two-stage method, which is the most effective one, is presented below.

Two stage ontology path patterns method. The key objective is to decrease the number of possible candidate path patterns, because a) it is expensive (in terms of efficiency and monetary cost) to feed a large number of patterns for the LLM, and b) if we feed the LLM with several irrelevant path patterns,

```
You are given the following path patterns of the form
{Class}->Property->{Class}->...->{Class} or (Literal)
### Patterns
{E22_Man_Made_Object} -> P108i_was_produced_by  -> {E12_Production} -> P14_carried_out_by
-> {E39_Actor} -> P92i_was_brought_into_existence_by  -> {E63_Beginning_of_Existence}
-> P4_has_time_span -> {E52_Time_Span} -> P82a_begin_of_the_begin->(Literal)
### End of Patterns

By using properties and classes from the above patterns,  and by never using a class as
a property (or the  opposite), please generate only a SPARQL query (without  explanation)
for answering the question:  I want  artworks from artists born during the second world war
```

Fig. 3. Query Generation Prompt with the predicted path

it can be confused, which can result in a wrong pattern (and query). We first send an LLM prompt using only the path patterns with $r = 1$ to predict the most relevant classes and properties of the KG to answer a question (for the running example, see Fig. 2 and the predicted paths in orange color in Fig. 1). Afterwards, we select candidate patterns based on the predictions and 3 rules.

• **Rule A**. It keeps only path patterns of any radius r including at least all the predicted classes and all the predicted properties.

• **Rule B**. If Rule A fails, it keeps the path patterns of any r with either all the predicted properties or all the predicted classes.

• **Rule C**. If both rules fail, it keeps all the path patterns of any radius r including at least one predicted class or property.

The final step is to create a prompt (i.e., see Fig. 3) using only the filtered path patterns based on the 3 rules and to ask the LLM to generate the SPARQL query. In our example, Rule A was executed since the path in orange in Fig. 1 includes the 3 predicted classes and the 2 predicted properties.

`TCRMQ` **Functionality**. The user can a) write a question in a natural language (evaluated in English and Greek, but can work with other languages, too), by selecting the desired KG, the path patterns method and the ChatGPT version. Afterwards, B) based on the user selections, the prompt(s) that correspond to the desired method and KG is sent to ChatGPT through its API and a SPARQL query is generated. The SPARQL query can either be sent directly to the SPARQL endpoint, or the user can edit the query, e.g., for asking for something different. Finally, C) the results of the SPARQL query are retrieved and presented, where the user can click on the URLs to browse more information for the entities.

3 Use Cases

The current version of `TCRMQ` exploits two RDF KGs containing artworks, from the CH domain with millions of triples, SAAM [14] and Kerameikos [1]. Below, we present use cases (UC) and scenarios (see a video in the following link: https://www.youtube.com/watch?v=v2zo_xFOaY8).

UC1. SPARQL Query Generation over a given KG. We can ask a question (even in several languages using the same methods and prompts) to

generate the SPARQL query. This concerns users that are not familiar with RDF and the ontology (e.g., CIDOC-CRM), i.e., they can ask natural questions and browse both the SPARQL query and the results (see the steps of Fig. 1).

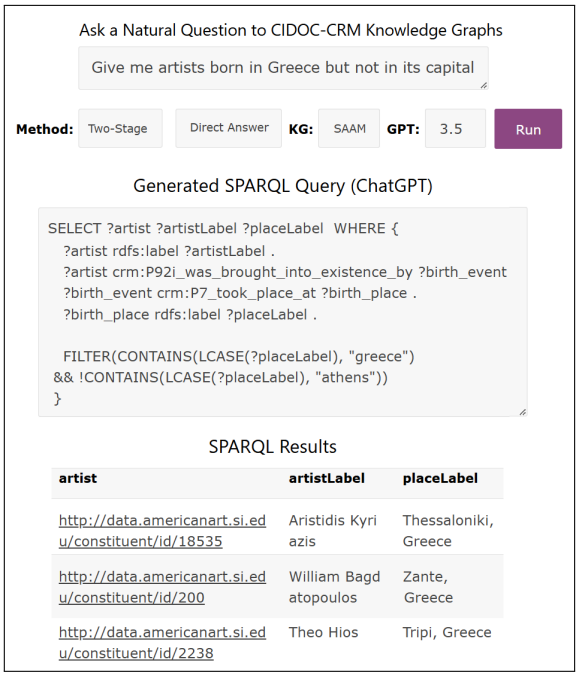

Fig. 4. Screenshots from TCRMQ

UC2. Query Enrichment with missing facts. The user can ask questions, such as those on the right side of Fig. 1, requiring external knowledge. In Fig. 1, the LLM was able to generate the SPARQL query by also adding in the FILTER statement the dates of the Second World War. It is worth mentioning that except for adding information in the filter statement of the query (such as dates), there are cases where the LLM needs extra reasoning, e.g., for questions like “Give me artists born in Greece but not in its capital" (see Fig. 4).

UC3. Ontology Learning and Query Editing. The user (e.g., an expert) can learn how an ontology is used in the given KG, e.g., which path patterns are used for expressing an event, and has the ability to edit the generated query, e.g., to restrict the results and to correct possible errors.

4 Experimental Evaluation

We use an extension of the benchmark that is available online in the following page: https://github.com/mountanton/CIDOC-QA-using-LLMs. That page also includes the code and all the results. The benchmark contains 60 natural

Questions Type	GPT-3.5	GPT-4 Mini	GPT-4
$r = 1$ questions	0.75	0.90	0.95
$r = 2$ questions	0.65	0.80	0.90
$r = 3$ questions	0.75	0.55	0.80
$r = 4$ questions	0.50	0.40	0.80
Mixed r questions	0.65	0.50	0.85
Questions with enriched facts	0.45	0.30	0.70
Average	0.62	0.57	0.83

Table 1. Average Accuracy using the two-stage method

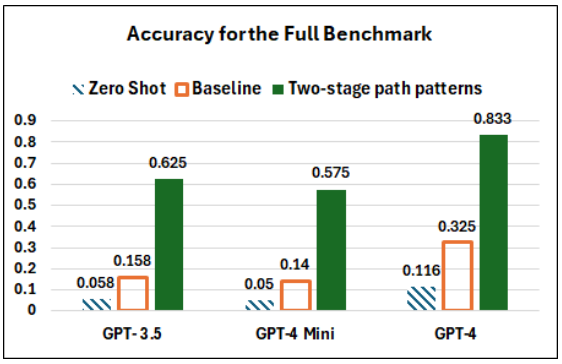

Fig. 5. Accuracy of ChatGPT models

questions for each KG (in total 120 questions). The questions are divided in 6 categories, each one having 10 questions, i.e., questions requiring to follow a path i) with radius $r = 1$, ii) with $r = 2$, iii) with $r = 3$, iv) with $r = 4$, v) a mixed path (combining paths of different r) and vi) questions that need enriched data from external knowledge (e.g., see questions in the right part of Fig. 1). The last category is the new part of the benchmark. Moreover, we use the same set of questions translated into Greek. We provide experimental results for 3 ChatGPT models (versions v3.5, v4-mini and v4) for both languages by using the zero-shot method, where we give the following prompt "By using the CIDOC-CRM ontology, please produce only a SPARQL query (without explanation) for the question Q"; the baseline method, where we give to the LLM the list of all the properties and classes of the KG; and the two-stage path patterns method. For these methods, we provide the accuracy; a number in the range [0,1] showing the fraction of questions that we managed to produce the correct SPARQL query. Each generated query was annotated as correct, if it produced the same results as the SPARQL query of the gold standard.

Comparison between the models. Fig. 5 shows that the zero-shot method achieved low accuracy, i.e., less than 0.12 for all models, while the baseline achieved 0.325 for ChatGPT v4 and approximately 0.15 for the other two versions. The highest accuracy (i.e., 0.833) observed by using ChatGPT v4, i.e., it provided the correct SPARQL query for the 83.3% of the 120 queries (100 /120), while the corresponding percentage for the second best, i.e., v3.5, was 62.5%. Re-

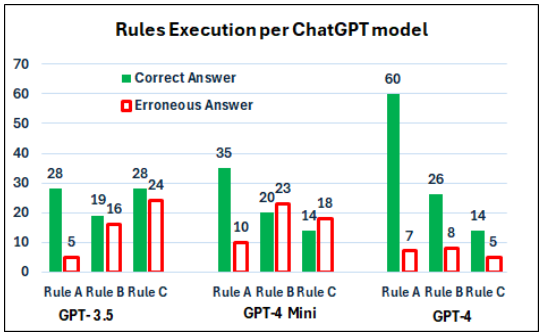

Fig. 6. Rules Execution (two-stage method)

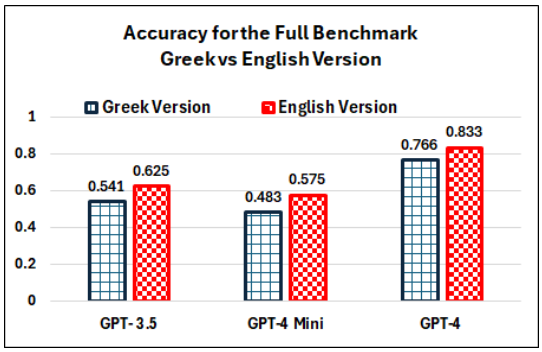

Fig. 7. Results of Greek vs English Benchmark

garding questions' categories (for the two-stage method), Table 1 shows that the GPT-4 outperformed the other models in all categories, and especially for the questions requiring either to follow a long path, or enriched facts from the LLM.

Analysis of Results/Errors. In Fig. 6 we can see a kind of ablation study, i.e., the rules executed for each ChatGPT model. Generally, the accuracy when Rule A is executed is much higher compared to those of Rules B and C. Specifically, we can see in Fig. 6, that for GPT-4, the predictions of the first prompt resulted in Rule A 67 times (with accuracy 60/67) whereas for the other models less than 45 times. Therefore, the predictions of the first prompt for GPT-4 were more accurate. Apart from the rules execution, the GPT-3.5 and GPT-4mini models were more vulnerable to i) SPARQL syntax errors or/and hallucination problems (e.g., for the questions needed missing facts), and ii) the selection of totally wrong paths for each question (see an analysis for GPT-3.5 in [10]). For GPT-4, the main erroneous cases were for questions that the selected path from the LLM was a superset or a subset of the correct one, or filter statements that were syntactically correct but not the desired ones.

English vs Greek Benchmark. The accuracy remains high even by sending the questions in Greek (see Fig. 7), but it is lower compared to English, e.g., 0.766 vs 0.833 for GPT-4. The main reason is that in some questions, greek

words in the filter statements were missing and ambiguous greek words resulted in the wrong path.

Cost and Execution Time. We needed on average ~ 0.01\$ for each question by using GPT-4 (the cost was much lower when Rule A executed), ~ 0.001\$ through GPT-4mini, and ~ 0.02\$ for GPT-3.5 (the most expensive in our case). Finally, the execution time is on average 2-3 seconds for the whole process (query generation and answer from the endpoint), regardless of the used GPT model.

5 Concluding Remarks

We shall demonstrate `TCRMQ`, a tool that offers QA over CIDOC-CRM based KGs, by translating the incoming natural question to a multi-hop SPARQL query, and by retrieving the results from the SPARQL endpoint. The tool is based on a *two-stage ontology path patterns method* that can be used in event-based models, and apart from questions requiring data from the KG, it leverages the LLM to generate queries with the right filtering conditions, even for information that is not present in the KG (e.g., dates). For a benchmark with 120 natural questions, we achieved an accuracy of 0.833 through the two-stage method and GPT-4, compared to 0.325 of the baseline. In future work, we plan to i) enrich the benchmark with questions for named entities, ii) use other event-based models, and iii) investigate RAG techniques [12] to exploit external knowledge.

References

1. Kerameikos.org. http://kerameikos.org/ (2021), accessed: August 2025
2. Allemang, D., Sequeda, J.: Increasing the accuracy of llm question-answering systems with ontologies. In: ISWC. pp. 324–339. Springer (2024)
3. Diallo, P.A.K.K., Reyd, S., Zouaq, A.: A comprehensive evaluation of neural sparql query generation from natural language questions. IEEE Access (2024)
4. Doerr, M.: The CIDOC CRM, an ontological approach to schema heterogeneity. In: Dagstuhl Seminar Proceedings. Schloss Dagstuhl (2005)
5. Feng, Y., Ding, L., Xiao, G.: GeoQAMap-geographic question answering with maps leveraging LLM and open knowledge base (short paper). In: GIScience 2023. Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2023)
6. Kovriguina, L., Teucher, R., Radyush, D., Mouromtsev, D.: SPARQLGEN: One-shot prompt-based approach for SPARQL query generation (2023)
7. Mecharnia, T., d'Aquin, M.: Performance and limitations of fine-tuned llms in sparql query generation. In: Proceedings of GenAIK Workshop. pp. 69–77 (2025)
8. Mountantonakis, M., Theocharakis, I., Tzitzikas, Y.: Why we need ontology-specific data portals: A case study for cidoc-crm. In: SWODCH (2023)
9. Mountantonakis, M., Tzitzikas, Y.: Real-time validation of ChatGPT facts using RDF knowledge graphs. ISWC Demo Paper (2023)
10. Mountantonakis, M., Tzitzikas, Y.: Generating sparql queries over cidoc-crm using a two-stage ontology path patterns method in llm prompts. ACM Journal on Computing and Cultural Heritage **18**(1), 1–20 (2025)

11. Pride, D., Cancellieri, M., Knoth, P.: Check for updates CORE-GPT: Combining open access research and large language models for credible, trustworthy question answering. In: TPDL 2023, Zadar, Croatia, September 26–29, 2023, Proceedings. vol. 14241, p. 146. Springer Nature (2023)
12. Sapidis, I., Zervos, V., Mountantonakis, M., Tzitzikas, Y.: Interactive and provenance-aware search and QA over documents using LLMs, RAG and knowledge graph verbalization. In: Proceedings of TPDL. Tampere, Finland (2025)
13. Sun, K., Xu, Y.E., Zha, H., Liu, Y., Dong, X.L.: Head-to-tail: How knowledgeable are large language models (llm)? AKA will llms replace knowledge graphs? arXiv preprint arXiv:2308.10168 (2023)
14. Szekely, P., et al.: Connecting the smithsonian american art museum to the linked data cloud. In: ESWC 2013. pp. 593–607. Springer (2013)