An Automated Evaluation Framework for Graph Database Query Generation Leveraging Large Language Models

Bailan He[1,2], Yushan Liu[1,2], Marcel Hildebrandt[1], Zifeng Ding[1,2], Yaomengxi Han[1,3] and Volker Tresp[1,2,*]

[1] *Siemens AG, Otto-Hahn-Ring 6, 81739 Munich, Germany*
[2] *Ludwig Maximilian University of Munich, Geschwister-Scholl-Platz 1, 80539 Munich, Germany*
[3] *Technical University of Munich, Arcisstraße 21, 80333 Munich, Germany*

Abstract

Large language models (LLMs) have attracted considerable attention in academia and industry due to their superior performance compared to classical machine learning models across various applications. In particular, prompt engineering and in-context learning enable LLMs to operate effectively in scenarios with minimal training data, where they demonstrate proficiency with only giving precise instructions or a few examples. The advanced reasoning abilities of LLMs have been instrumental in the development of intelligent assistants. These assistants often rely on accessing information from comprehensive databases such as knowledge graphs (KGs) through natural language. The process of converting natural language requests into query language to retrieve information from databases is known as query generation (QG). One challenge in QG is the evaluation of the LLMs' performance due to the absence of standardized evaluation frameworks and datasets. To tackle this challenge, we introduce an automated evaluation framework tailored for QG, featuring three key metrics: Gold Query Accuracy, Execution Accuracy, and Execution Rate. We focus on the exemplary use case of accessing a graph database in a supply chain management (SCM) setting via a natural language interface. Our results demonstrate the efficacy of our framework and metrics in accurately evaluating model performance for QG tasks.

Keywords

Query generation, Knowledge graph, Large language model, Supply chain management

1. Introduction

Large Language Models (LLMs) have recently gained prominence in natural language processing, demonstrating exceptional proficiency across various tasks [1, 2, 3, 4]. Unlike conventional machine learning models that rely heavily on labeled data, LLMs exhibit reasoning capabilities, allowing them to perform diverse tasks without extensive labeled datasets. For instance, Perez-Beltrachini et al. [5] developed a semantic parser using LLMs with an RDF dataset. This system

Third International Workshop on Linked Data-driven Resilience Research (D2R2'24) co-located with ESWC 2024, May 27th, 2024, Hersonissos, Greece

*Corresponding author.
bailan.he@siemens.com (B. He); yushan.liu@siemens.com (Y. Liu); marcel.hildebrandt@siemens.com (M. Hildebrandt); zifeng.ding@siemens.com (Z. Ding); yaomengxi.han@siemens.com (Y. Han); tresp@dbs.ifi.lmu.de (V. Tresp)
0009-0001-5504-1462 (B. He)

effectively understands natural language questions within a user's dialogue and generates corresponding SPARQL queries to address them. Given the capacity of LLMs to process and generate vast amounts of text data, manual evaluation of LLMs becomes impractical, while automatic frameworks ensure a swift and efficient assessment. Automated evaluation is also preferred due to its ability to mitigate biases and inconsistencies introduced by human evaluators, thereby enhancing reliability and reproducibility. As the utilization of LLMs expands across various domains, scalable evaluation methods are imperative. Objective metrics such as perplexity [6], BLEU score [7], and ROUGE score [8] offer quantifiable performance measures in general scenarios. However, accurately evaluating the performance of LLMs in domain-specific settings remains challenging. In many cases, anecdotal evidence and manual inspection have been used as substitutes for more rigorous evaluation methods. Furthermore, in many industrial contexts, the developers of artificial intelligence tools may not have the necessary domain expertise to evaluate the performance of LLMs, leading to a need for an automatic and systematic approach to evaluate LLMs in the absence of in-depth domain knowledge. In this paper, we focus on an application in the context of supply chain management (SCM), where the lack of standard evaluation datasets poses a significant challenge, complicating the comparison of different LLMs.

Supply chain management (SCM) involves continuously monitoring supply chains to ensure their operability and proactively making them sufficiently resilient to withstand disruptive events such as pandemics, natural disasters, or political and economic crises [9]. With the growing volume and complexity of available data, it is essential for supply chain managers to efficiently manage, store, and retrieve the relevant information. This ensures that relevant data can be accessed and analyzed in a timely manner to make well-informed strategic decisions, identify critical risks, and accurately react to current events [10]. The management across the whole supply chain necessitates extensive databases for comprehensive analysis. Knowledge Graphs (KGs) represent networks of real-world entities, encompassing objects, events, situations, and concepts, and elucidate the connections between them and have emerged as invaluable tools for offering an interconnected perspective on SCM data. For instance, within SCM, entities like suppliers, smelters, and components can be represented as nodes in a KG, while relationships between different types of nodes, such as "located in," serve as the edges within the KG. Several studies have developed SCM-related KGs and applied reasoning techniques to improve supply chain management [11, 12, 13]. For example, the CoyPu KG [1] integrates macroeconomic data and global crisis events to enhance transparency in SCM operations. These KGs are typically processed using graph database management systems like Neo4j [2]. Querying these systems, such as using Cypher queries for Neo4j, assists in extracting relevant information for further analysis. However, mastering the optimal querying of graph databases requires significant time and effort from SCM professionals. Therefore, developing a natural language interface utilizing LLM-based approaches to interpret database schemas and translate requests into graph database queries (e.g., SPARQL or Cypher) can significantly aid SCM professionals in understanding the data and making informed decisions. This translation task is referred to as query generation (QG).

[1]https://coypu.org/ergebnisse/knowledge-graph
[2]https://neo4j.com/

Despite the emergence of QG solutions [14, 15], their real-world performance when using proprietary domain data and schemas is still uncertain because there are no readily available datasets to evaluate model performance. To address this challenge, we present a novel automated evaluation framework [3] tailored for practical applications, exemplified in the industrial use case of SCM QG. Grounded on KGs, our framework first harnesses the power of GPT-3.5 [16] to generate an evaluation dataset. This dataset is then utilized to objectively measure model performance, enabling users to modify prompts according to their specific needs and data requirements.

2. Definitions

Definition 1 (Knowledge Graph). A KG is defined as a collection of triples $\mathcal{G} \subset \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ where $\mathcal{E}$ denotes the set of entities and $\mathcal{R}$ the set of relation types. In our use case, elements in $\mathcal{E}$ correspond to supply chain-related entities, e.g., suppliers, smelters, and components, and are represented as nodes in the graph. Every entity has a unique entity type, which is defined by the mapping $f : \mathcal{E} \rightarrow \mathcal{T}$, where $\mathcal{T}$ stands for the set of entity types. The entities are connected via relations specified in $\mathcal{R}$, represented as typed directed edges in the graph.
Definition 2 (Query Generation). Given a user's natural language request $X = (x_1, x_2, ..., x_m)$ and a corresponding query $Y = (y_1, y_2, ..., y_n)$, where each x_i and y_j represent a token in the request and query, respectively, the aim is to accurately discern the underlying intent of a request X and generate a corresponding query Y [5]. The execution of query Y within the KG database yields retrieval results denoted as R, where R corresponds to the user's desired information from the KG.

3. Framework Overview

Our proposed framework consists of two main processes: (1) A query dataset creation process that generates an evaluation dataset containing diverse natural language requests and queries. (2) A QG process that evaluates the model performance of different prompts based on the generated evaluation dataset.

3.1. Query dataset creation

For the scope of our study, the query dataset consists of structured queries alongside their corresponding natural language requests. Creating such datasets presents notable challenges, particularly due to the potential lack of existing datasets containing relevant queries tailored to specific scenarios or domains. The task of annotating data for specific domains, such as SCM, requires the involvement of domain experts, which incurs significant costs and manual efforts.

To optimize efficiency, we present an innovative framework harnessing the generative potential of language models. This framework aims to automate the creation of query datasets,

[3] Code is released at: `https://github.com/4hebailanc/Automated-Evaluation-Framework-for-Graph-Database-Query-Generation`

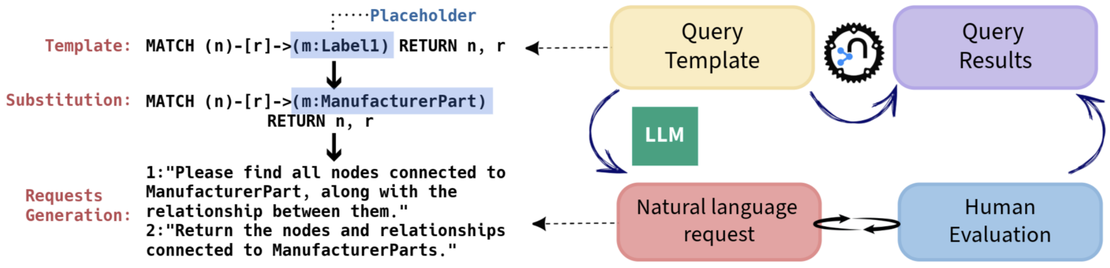

Placeholder
Template: MATCH (n)-[r]->(m:Label1) RETURN n, r
Substitution: MATCH (n)-[r]->(m:ManufacturerPart) RETURN n, r
Requests Generation: 1:"Please find all nodes connected to ManufacturerPart, along with the relationship between them." 2:"Return the nodes and relationships connected to ManufacturerParts."
Query Template
Query Results
LLM
Natural language request
Human Evaluation

Figure 1: Workflow of query dataset generation. Templates with placeholders are created first, i.e., ”m:Label1” in the example. Then placeholders will be replaced with specific information, i.e. ”m:ManufacturerPart”, to compose a gold query. After that, we use an LLM to generate natural language requests. For each query, three distinct requests are generated, resulting in a set of query and request pairs. Finally, a domain expert is involved to ensure the quality of dataset.

thereby facilitating the quantitative evaluation of models. As depicted in Figure 1, the query generation process encompasses four sequential steps:

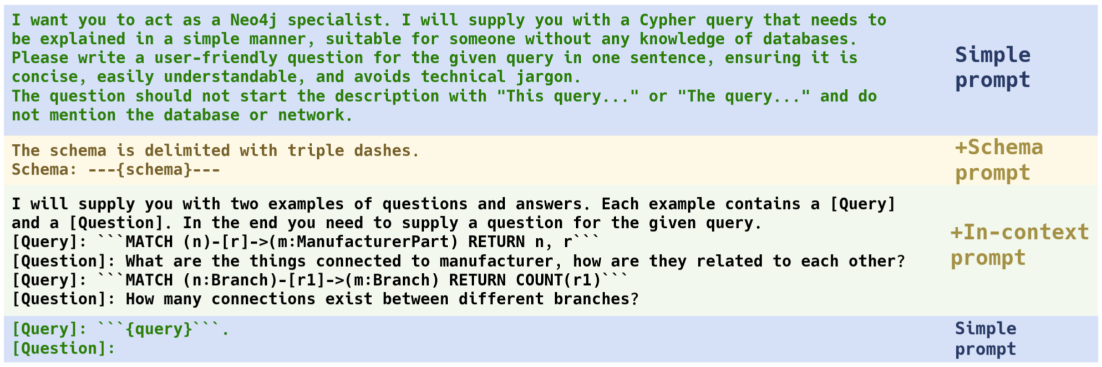
a. Initial Query Template The process begins by creating a general query template *t*, using placeholders for specific information elements like nodes or relations. An example for a query template *t* in the query language Cypher is *MATCH (n) -[r]-> (m:Label1) RETURN n, r*, where *m:Label1* acts as a placeholder for a node labeled with ‘Label1’. In the context of this query template, ‘Label1’ represents a specific label assigned to nodes within the graph database. The placeholder allows for dynamic substitution with actual node labels during query generation.

b. Placeholder Substitution Once the template is crafted, specific information is utilized to replace the designated placeholders, resulting in the formulation of a comprehensive gold query *q*. A gold query *q* is the final query created by substituting specific information into a template, and it acts as the standard against which we evaluate the accuracy of our model’s query generation capabilities. In this context, the placeholder *m:Label1* is replaced with *m:ManufacturerPart*. Consequently, the placeholder substitution produces the following gold query *q*: *MATCH (n) -[r] -> (m:ManufacturerPart) RETURN n, r*. This particular query, *q*, is tailored to retrieve nodes (identified as ’n’) and their associated relationships (referred to as ’r’) connected to nodes labeled as ’ManufacturerPart’ within the database’s graph structure. After substitution, each generated gold query will be executed once to ensure its functionality and accuracy in retrieving pertinent information from the database.

c. Requests Generation The LLM employs diverse prompting formats to generate a range of query requests in natural language. As shown in Figure 2, three distinct prompt templates, i.e., simple prompt, schema prompt, and in-context prompt templates have been devised. The simple prompt directly instructs the model to generate natural language requests. The schema prompt incorporates the KG schema to guide the model. The in-context prompt employs in-context learning [17], integrating multiple query and natural language request pairs as demonstrations within the template. Each query results in the generation of three distinct natural language requests.

d. Human Evaluation The concluding stage necessitates human intervention to validate the quality and precision of the generated queries (see Section 4.2 for details). This step acts as an important quality control measure, confirming that the queries effectively represent the intended information retrieval process.

It is noteworthy that only steps (a) and (d) necessitate human intervention within the proposed methodology. Step (a) consists of creating a query template, a process that is simplified by using the reference base query syntax [4]. The human validation undertaken in Step (d) serves to ensure the quality and accuracy of queries, thereby augmenting the overall reliability of the system.



```
I want you to act as a Neo4j specialist. I will supply you with a Cypher query that needs to
be explained in a simple manner, suitable for someone without any knowledge of databases.
Please write a user-friendly question for the given query in one sentence, ensuring it is
concise, easily understandable, and avoids technical jargon.
The question should not start the description with "This query..." or "The query..." and do
not mention the database or network.
```

```
The schema is delimited with triple dashes.
Schema: ---{schema}---
```

```
I will supply you with two examples of questions and answers. Each example contains a [Query]
and a [Question]. In the end you need to supply a question for the given query.
[Query]: ```MATCH (n)-[r]->(m:ManufacturerPart) RETURN n, r```
[Question]: What are the things connected to manufacturer, how are they related to each other?
[Query]: ```MATCH (n:Branch)-[r1]->(m:Branch) RETURN COUNT(r1)```
[Question]: How many connections exist between different branches?
```

```
[Query]: ```{query}```.
[Question]:
```

Figure 2: The simple prompt template directly instructs the model without additional information. In particular, the schema prompt template also provides the schema of the KG. On this basis, the in-context prompt template includes multiple query and natural language request pairs as demonstrations to guide the model.

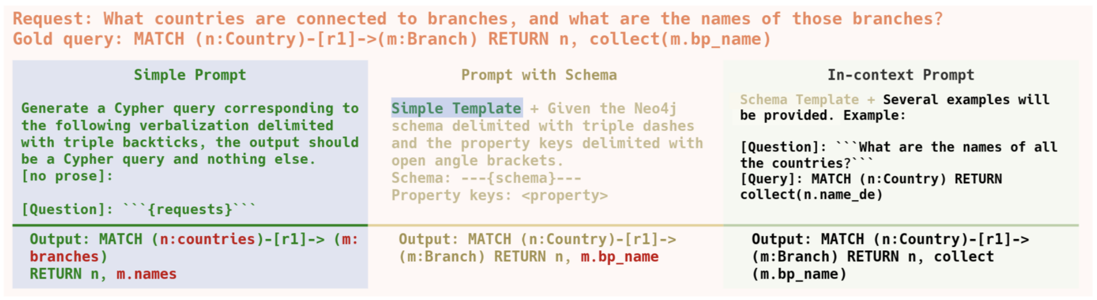
3.2. Query generation with LLMs

Our study centers on the task of query generation within the framework of Neo4j databases. Specifically, our objective is to develop a model capable of producing queries based on natural language requests, enabling the retrieval of pertinent information from the database. To assess the model's performance quantitatively, in line with prior research [18, 5], we employ execution-based automatic metrics. We employ three primary metrics in our evaluation: Execution Rate (ER), which evaluates the executability of the output queries within the database, Gold Query Accuracy (GQA), which assesses the similarity between the output query and the gold query, and Execution Accuracy (EA), which measures the correspondence of retrieval results to those of the gold query. Both GQA and EA are calculated by averaging the scores of all gold and output queries pairs using BERTScore [19], which measures the similarity between two text sequences by computing the cosine similarity between the contextual embeddings of their tokens, as produced by the language model BERT [20]. The key idea behind BERTScore is to not only take exact word matches into account but also semantic similarity and contextual appropriateness [5], which makes it more robust compared to traditional evaluation metrics. Higher values for all three metrics are preferred: a high ER signifies correct execution of output queries in the database, while higher GQA and EA scores indicate strong resemblance between

[4]https://neo4j.com/docs/cypher-manual/current/queries/basic/
[5]In our case, the same cypher queries return a fixed order of requested information,

the output and gold queries, along with accurate retrieval results aligning with those of the gold query.

1. **Execution Rate (ER)**: This metric evaluates the executability of the output queries within the database. It is defined as the ratio of executable queries to the total number of output queries:
$$\text{ER} = \frac{\text{Number of executable queries}}{\text{Total number of output queries}} \tag{1}$$
2. **Gold Query Accuracy (GQA)**: This metric evaluates the similarity between the output query and the gold query. It is calculated using the BERTScore metric [19] as follows:
$$\text{GQA} = \frac{1}{N}\sum_{i=1}^{N} \text{BERTScore}(O_i, G_i) \tag{2}$$
where N is the total number of output and gold queries pairs, and O_i and G_i represent the output query and gold query for the i-th pair, respectively.
3. **Execution Accuracy (EA)**:
$$\text{EA} = \frac{1}{N}\sum_{i=1}^{N} \text{BERTScore}(R_i, R_{G_i}) \tag{3}$$
where N is the total number of output and gold queries pairs, and R_i and R_{G_i} represent the retrieval results for the output query and gold query for the i-th pair, respectively.



```
Request: What countries are connected to branches, and what are the names of those branches?
Gold query: MATCH (n:Country)-[r1]->(m:Branch) RETURN n, collect(m.bp_name)
```

```
Simple Prompt

Generate a Cypher query corresponding to
the following verbalization delimited
with triple backticks, the output should
be a Cypher query and nothing else.
[no prose]:

[Question]: ```{requests}```

Output: MATCH (n:countries)-[r1]-> (m:
branches)
RETURN n, m.names
```

```
Prompt with Schema

Simple Template + Given the Neo4j
schema delimited with triple dashes
and the property keys delimited with
open angle brackets.
Schema: ---{schema}---
Property keys: <property>

Output: MATCH (n:Country)-[r1]->
(m:Branch) RETURN n, m.bp_name
```

```
In-context Prompt

Schema Template + Several examples will
be provided. Example:

[Question]: ```What are the names of all
the countries?```
[Query]: MATCH (n:Country) RETURN
collect(n.name_de)

Output: MATCH (n:Country)-[r1]->
(m:Branch) RETURN n, collect
(m.bp_name)
```

Figure 3: The natural language request and gold query are on the top, and the model query generation output is given under each type of prompt. Wrong or undesired query outputs are marked in red. Without the schema information, the model with simple prompt utilizes ”associated with” as a relation type, which is not defined in the KG schema. The prompt with schema solves this problem but does not use the desired ”collect” function. With in-context demonstrations, the model shows best performance.

4. Experiments

We illustrate the practical application of our framework by utilizing real SCM data [10], emphasizing its effectiveness in assessing the performance of LLMs for QG in real-world scenarios.

4.1. Supply chain knowledge graph

The supply chain knowledge graph consolidates information from internal sources of the company Siemens. This comprehensive dataset includes insights such as tier-1 suppliers, business scopes, and specific parts associated with a company. Additionally, it incorporates external data, such as publicly available information on smelters and substances. Specifics regarding tier-2 and tier-3 suppliers are primarily derived from customs data. This information is further supplemented by a smaller portion obtained from both private customs records and public media sources. In total, there are 16,910 tier-1 suppliers, 43,759 tier-2 suppliers, and 49,775 tier-3 suppliers of Siemens. All entity and relation types and corresponding numbers of nodes and edges are listed in Table 1. It is important to note that suppliers at different tier levels are not mutually exclusive. To facilitate the access to this complex network of information, the graph is structured using the graph database platform Neo4j.

Table 1
Knowledge graph statistics: Eight entity types and eleven relation types are in the knowledge graph.

Entity type	#Nodes	Relation type	#Edges
Supplier	61,234	supplies_to	138,197
Manufacturer Part	1,650	related_to	59,894
Company Part	1,295	belongs_to	56,663
Smelter	340	located_in	30,107
Substance	321	includes	10,088
Component	233	produces	7,831
Country	172	produced_in	4,381
Business Scope	32	same_as	1,847
		manufactured_by	1,564
		contains	764
		refines	340
Total	**65,277**	**Total**	**311,676**

4.2. Supply chain query dataset

We compiled a set of 60 query templates, categorized into six main groups as detailed in Table 2. Each template underwent placeholder substitution, as outlined in Section 3.1, yielding five distinct queries per template. This process resulted in the creation of 300 gold queries. Subsequently, we generated three natural language requests for each gold query, using three different prompts. These query-request pairs were subsequently evaluated by three reviewers within our organization to determine their feasibility. A query-request pair is labeled as reasonable if the reviewers agree that the query is effective in retrieving information from the database to answer the corresponding request; otherwise, it was marked as unreasonable. Any pair receiving unreasonable annotations from the reviewers was filtered out. Following this filtering process, we retained 825 pairs of query-requests to constitute our query dataset. Inter-annotator agreement was measured using Fleiss' kappa [21], which yielded a value of 0.72 in our annotations, indicating substantial agreement among the reviewers regarding the alignment of the generated query and request pairs. The high number of retained pairs underscores the effectiveness of our dataset generation methodology. The time spent annotating by each reviewer is estimated to be 3 hours.

Table 2
Query template statistics: We design 60 query templates across 6 categories.

Query Template Type	Number
Node Matching	10
Relationship Matching	10
Aggregation and Analysis	10
Combining Filters and Aggregation	10
Complex Queries	15
Traversal and Paths	5

4.3. Query generation performance

In this section, we show how the generated dataset can be used to evaluate the QG capabilities of the model and to help modify the model's corresponding task prompts. As shown in Figure 3, we evaluate the performance of three different types of QG prompts using two state-of-the-art language models, GPT-3.5 and GPT-4: simple prompts, prompts with schema, and in-context prompts. In a simple prompt, the model is directed to generate a corresponding query without prior knowledge of schema information within the KG. This may result in the utilization of relation types not present in the KG; for instance, the relation type ”associated with” is not defined in the KG schema. However, when additional schema data is integrated, the model shows improved effectiveness by employing precise schema properties in queries. Nevertheless, without specific user directives, such as indicating the desired output format using the ”collect” function, the generated query may not perfectly match our gold query. The in-context prompt takes a step further by providing multiple request and query pairs, along with the schema, to guide the model in producing correct results. When presented with an in-context demonstration, the model shows the best performance in QG tasks.

Table 3
Query generation results: ”Simple” denotes direct model instruction, ”Schema” indicates prompting with schema, and ”ICL-*k* shot” (in-context learning with *k* examples) involves instructing the model with in-context demonstrations.

GPT-3.5				GPT-4			
Method	**GQA**	**ER**	**EA**	**Method**	**GQA**	**ER**	**EA**
Simple	0.47	0.76	0.31	Simple	0.55	0.81	0.29
Schema	0.62	0.74	0.42	Schema	0.71	0.89	0.43
ICL-1 shot	0.63	0.87	0.44	ICL-1 shot	0.69	0.89	0.47
ICL-3 shot	0.70	0.90	**0.55**	ICL-3 shot	0.73	0.91	0.52
ICL-5 shot	**0.72**	**0.91**	0.52	ICL-5 shot	**0.75**	**0.93**	**0.53**

The results detailed in Table 3 consistently demonstrate GPT-4's superior performance over GPT-3.5 across all three prompt types. Notably, employing schema and in-context demonstrations consistently result in higher GQA , ER, and EA compared to direct instructions to the model. The provision of a schema significantly enhances model performance, with in-context demonstrations exhibiting the highest efficacy in both GPT-3.5 and GPT-4. These results provide valuable insights into the efficacy of various prompting methods and highlight the advancements from GPT-3.5 to GPT-4 in QG tasks. Furthermore, they validate the utility of query datasets generated through the pipeline for prompt tuning in real-world applications.

5. Conclusion

In conclusion, our paper proposed a comprehensive automated evaluation framework for QG, addressing the challenge of assessing the performance of LLMs in specific domains due to the lack of standardized evaluation criteria and datasets. Our framework comprises two main steps: firstly, the creation of an evaluation dataset leveraging the reasoning abilities of LLMs through query templates, and secondly, the evaluation of QG model performance based on the generated evaluation dataset. To illustrate the efficacy of our framework, we apply it to a concrete industry use case: QG in SCM KGs. The creation of a diverse supply chain query dataset, as delineated in Table 2, establishes the groundwork for assessing the model performance of different prompts for QG. Subsequent analysis, summarized in Table 3, consistently demonstrates the superiority of GPT-4 over GPT-3.5 across various conditions. Overall, our framework enhances the comprehension of QG within the realm of SCM graphs, offering practical implications for enhancing query efficiency and accuracy in real-world SCM applications. This use case exemplifies our methodology for enhancing efficiency and accuracy in query generation within real-world scenarios.

6. Future Directions

In future research, expanding the experimental configuration to include other prominent LLMs like Gemini [22] from Google and Llama [1] from Meta would be beneficial. This expansion would broaden the scope of the study, allowing for a more comprehensive evaluation of the proposed framework's effectiveness across a wider range of LLMs. Additionally, the current single-round question answering format is constraining; hence, delving into the exploration of multi-round dialogues represents a promising avenue for further exploration. By incorporating these future directions, the research can continue to advance our understanding of LLMs and their applications in real-world scenarios.

Acknowledgments This work has been supported by the German Federal Ministry for Economic Affairs and Climate Action (BMWK) as part of the project CoyPu under grant number 01MK21007K and has also been supported by the DAAD programme Konrad Zuse Schools of Excellence in Artificial Intelligence, sponsored by the Federal Ministry of Education and Research.

References

[1] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, G. Lample, Llama:

Open and efficient foundation language models, ArXiv abs/2302.13971 (2023). URL: https://api.semanticscholar.org/CorpusID:257219404.

[2] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. de las Casas, L. A. Hendricks, J. Welbl, A. Clark, T. Hennigan, E. Noland, K. Millican, G. van den Driessche, B. Damoc, A. Guy, S. Osindero, K. Simonyan, E. Elsen, O. Vinyals, J. W. Rae, L. Sifre, An empirical analysis of compute-optimal large language model training, in: A. H. Oh, A. Agarwal, D. Belgrave, K. Cho (Eds.), Advances in Neural Information Processing Systems, 2022. URL: https://openreview.net/forum?id=iBBcRUlOAPR.

[3] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, N. Fiedel, Palm: Scaling language modeling with pathways, J. Mach. Learn. Res. 24 (2023) 240:1–240:113. URL: http://jmlr.org/papers/v24/22-1144.html.

[4] P. Liang, R. Bommasani, T. Lee, D. Tsipras, D. Soylu, M. Yasunaga, Y. Zhang, D. Narayanan, Y. Wu, A. Kumar, B. Newman, B. Yuan, B. Yan, C. Zhang, C. Cosgrove, C. D. Manning, C. Ré, D. Acosta-Navas, D. A. Hudson, E. Zelikman, E. Durmus, F. Ladhak, F. Rong, H. Ren, H. Yao, J. Wang, K. Santhanam, L. J. Orr, L. Zheng, M. Yüksekgönül, M. Suzgun, N. Kim, N. Guha, N. S. Chatterji, O. Khattab, P. Henderson, Q. Huang, R. Chi, S. M. Xie, S. Santurkar, S. Ganguli, T. Hashimoto, T. Icard, T. Zhang, V. Chaudhary, W. Wang, X. Li, Y. Mai, Y. Zhang, Y. Koreeda, Holistic evaluation of language models, CoRR abs/2211.09110 (2022). URL: https://doi.org/10.48550/arXiv.2211.09110. doi:10.48550/ARXIV.2211.09110. arXiv:2211.09110.

[5] L. Perez-Beltrachini, P. Jain, E. Monti, M. Lapata, Semantic parsing for conversational question answering over knowledge graphs (2023) 2499–2514. URL: https://doi.org/10.18653/v1/2023.eacl-main.184. doi:10.18653/V1/2023.EACL-MAIN.184.

[6] Perplexity—a measure of the difficulty of speech recognition tasks, The Journal of the Acoustical Society of America 62 (1977) S63–S63.

[7] K. Papineni, S. Roukos, T. Ward, W. Zhu, Bleu: a method for automatic evaluation of machine translation, in: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA, ACL, 2002, pp. 311–318. URL: https://aclanthology.org/P02-1040/. doi:10.3115/1073083.1073135.

[8] C.-Y. Lin, ROUGE: A package for automatic evaluation of summaries, in: Text Summarization Branches Out, Association for Computational Linguistics, Barcelona, Spain, 2004, pp. 74–81. URL: https://aclanthology.org/W04-1013.

[9] A. Cox, Power, value and supply chain management, Supply chain management: An international journal 4 (1999) 167–175.

[10] Y. Liu, B. He, M. Hildebrandt, M. Buchner, D. Inzko, R. Wernert, E. Weigel, D. Beyer, M. Berbalk, V. Tresp, A knowledge graph perspective on supply chain resilience, in: S. Tramp, R. Usbeck, N. Arndt, J. Holze, S. Auer (Eds.), Proceedings of the Second In-

ternational Workshop on Linked Data-driven Resilience Research 2023 co-located with Extended Semantic Web Conference 2023 (ESWC 2023), Hersonissos, Greece, May 28, 2023, volume 3401 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2023. URL: https://ceur-ws.org/Vol-3401/paper3.pdf.

[11] J. Deng, C. Chen, X. Huang, W. Chen, L. Cheng, Research on the construction of event logic knowledge graph of supply chain management, Adv. Eng. Informatics 56 (2023) 101921. URL: https://doi.org/10.1016/j.aei.2023.101921. doi:10.1016/J.AEI.2023.101921.

[12] X. Huang, L. Cheng, J. Deng, T. Wang, Binocular attention-based stacked bilstm NER model for supply chain management event knowledge graph construction, in: Proceedings of the 15th International Conference on Machine Learning and Computing, ICMLC 2023, Zhuhai, China, February 17-20, 2023, ACM, 2023, pp. 40–46. URL: https://doi.org/10.1145/3587716.3587723. doi:10.1145/3587716.3587723.

[13] W. Chen, L. Cheng, T. Wang, J. Deng, Knowledge graph construction for supply chain management in manufacturing industry, in: D. Huang, P. Premaratne, B. Jin, B. Qu, K. Jo, A. Hussain (Eds.), Advanced Intelligent Computing Technology and Applications - 19th International Conference, ICIC 2023, Zhengzhou, China, August 10-13, 2023, Proceedings, Part IV, volume 14089 of *Lecture Notes in Computer Science*, Springer, 2023, pp. 682–693. URL: https://doi.org/10.1007/978-981-99-4752-2_56. doi:10.1007/978-981-99-4752-2_56.

[14] T. Bratanic, Generating cypher queries with chatgpt-4 on any graph schema, 2023. URL: https://neo4j.com/developer-blog/generating-cypher-queries-with-chatgpt-4-on-any-graph-schema/, accessed: [15.10.2023].

[15] X. Li, R. Zhao, Y. K. Chia, B. Ding, L. Bing, S. R. Joty, S. Poria, Chain of knowledge: A framework for grounding large language models with structured knowledge bases, CoRR abs/2305.13269 (2023). URL: https://doi.org/10.48550/arXiv.2305.13269. doi:10.48550/ARXIV.2305.13269. arXiv:2305.13269.

[16] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, Advances in neural information processing systems 33 (2020) 1877–1901.

[17] Q. Dong, L. Li, D. Dai, C. Zheng, Z. Wu, B. Chang, X. Sun, J. Xu, Z. Sui, A survey for in-context learning, arXiv preprint arXiv:2301.00234 (2022).

[18] A. Saha, V. Pahuja, M. Khapra, K. Sankaranarayanan, S. Chandar, Complex sequential question answering: Towards learning to converse over linked question answer pairs with a knowledge graph, in: Proceedings of the AAAI conference on artificial intelligence, volume 32, 2018.

[19] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, Y. Artzi, Bertscore: Evaluating text generation with BERT, in: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020, OpenReview.net, 2020. URL: https://openreview.net/forum?id=SkeHuCVFDr.

[20] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, in: J. Burstein, C. Doran, T. Solorio (Eds.), Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers), Association for

Computational Linguistics, 2019, pp. 4171–4186. URL: https://doi.org/10.18653/v1/n19-1423. doi:10.18653/V1/N19-1423.

[21] J. L. Fleiss, Measuring nominal scale agreement among many raters., Psychological bulletin 76 (1971) 378.

[22] G. Team, R. Anil, S. Borgeaud, Y. Wu, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, et al., Gemini: a family of highly capable multimodal models, arXiv preprint arXiv:2312.11805 (2023).