Asian Journal of Research in Computer Science

Volume 16, Issue 4, Page 318-326, 2023; Article no.AJRCOS.109603
ISSN: 2581-8260

Exploratory Search Prompt Generation using n-Degree Connection in Knowledge Graph

Sudhanshu Gupta [a*] and Krishna Kumar Tiwari [a*]

[a] *AI Center of Excellence Jio Private Limited Bangalore, India.*

Authors' contributions

This work was carried out in collaboration between both authors. Both authors read and approved the final manuscript.

Article Information

DOI: 10.9734/AJRCOS/2023/v16i4393

Open Peer Review History:
This journal follows the Advanced Open Peer Review policy. Identity of the Reviewers, Editor(s) and additional Reviewers, peer review comments, different versions of the manuscript, comments of the editors, etc are available here: https://www.sdiarticle5.com/review-history/109603

Original Research Article

Received: 01/10/2023
Accepted: 06/12/2023
Published: 09/12/2023

ABSTRACT

Search engines play a vital role in retrieving in- formation, but users often struggle to express their precise information needs, resulting in less-than-optimal search results. Therefore, enhancing search query refinement is crucial to elevate the accuracy and relevance of search outcomes. One particular challenge that existing search engines face is presenting refined results for queries containing two or more unrelated entities.

This paper presents a novel approach for efficient search prompt generation by leveraging connected nodes and attributes in the knowledge graph. We propose a comprehensive exploration technique that explores the n-degrees connections and their at- tributes to generate all possible imaginative prompts. We realized that n-degrees connection exploration is an expensive task, hence we conducted experiments to determine the effectiveness of 2- degree exploration prompts in covering all the user-asked queries within the provided dataset.

In testing with approximately 2000 queries on a related knowl- edge graph dataset, we found out that our proposed methodol- ogy significantly improved query coverage. At n=1 depth, the coverage increased from 58% without the methodology to 84% with it. At n=2 depth, the coverage rose from 92% without the methodology to nearly 99% with it. Additionally, due to our question caching strategy at n=2, we observed faster response times for all questions.

**Corresponding author: E-mail: guptasudhanshu473@gmail.com, 826.Krishna@gmail.com;*

Keywords: Knowledge graph; search.

1. INTRODUCTION

In the rapidly advancing era of Artificial Intelligence, data has emerged as the new oil, driving significant transformations across various industries. Over the past decade, there has been an explosive growth in data generation and consumption, profoundly influencing people's lifestyle preferences. To meet the demands of today's fast-paced lifestyles, businesses are faced with the challenge of extracting valuable insights from high- dimensional data. Traditional data modeling and management techniques have struggled to keep up with the complexities of this data landscape. As a solution, many enterprises have turned to graph-based storage and knowledge graph-based intelligence systems. These systems offer remarkable flexibility in managing and analyzing high-dimensional data, addressing the unique requirements posed by this evolving landscape. Now that the stage is set for high-dimensional data-driven ap- plications, it becomes imperative to explore the impact of data dimensionality on a crucial business component - "Search". Over time, the meaning of search has evolved, encompassing various techniques traditionally employed, including keyword retrieval, PageRank, personalization, natural language processing for user intent queries, and knowledge graph utilization for connected data. These techniques have served as foundational pillars for search functionalities and have played a significant role in information retrieval. However, with the advent of high-dimensional data, it is crucial to examine how these search techniques adapt and perform in the face of complex and interconnected datasets.

Today's digital landscape has expanded the scope of search to encompass a wide range of activities, including web search for information retrieval, e-commerce search for products and reviews, social media search for people, images, and videos, entertainment search for movies and songs, and map search for location-based information. The variety and highly dimensional, interconnected nature of the data against which search is applied are evident in these diverse search domains. Search has evolved beyond its traditional role of finding information on the web. It now plays a crucial role in discovering new insights and establishing connections between people, concepts, and various data entities. The world's connected nature, with its intricate web of relationships and hidden patterns, demands a data representation approach that can support these complex relationships and adapt as new connections are discovered over time. In this context, Knowledge Graphs emerge as a powerful tool for representing highly interconnected data, facilitating comprehensive exploration and analysis of complex relationships.

Knowledge Graphs offer a promising approach to addressing the challenges posed by highly interconnected data.

2. KNOWLEDGE GRAPH

A Knowledge Graph represents knowledge as a graph structure composed of nodes and edges, where nodes represent entities or concepts, and edges represent the relationships between them. This unique way of representing data and its advantages have drawn the attention of academia as well as the industry [1–4]. This graph-based representation enables a semantic understanding of the data, allowing for rich and context-aware data representation and retrieval.

2.1 Features of Knowledge Graph

Knowledge graph has several features that make it one of the best data representation forms and help in uncovering hidden patterns and relationships in evolving data.

Semantic Representation: The semantic nature of Knowledge Graphs provides a powerful mechanism for capturing the meaning and relationships between entities. By encoding the semantics of terms and concepts in a structured and standardized manner, Knowledge Graphs facilitate precise and accurate data representation. This semantic representation is fundamental to the effectiveness of search, as it enables the identification of entities or concepts based on their meaning and allows for the retrieval of relevant information.

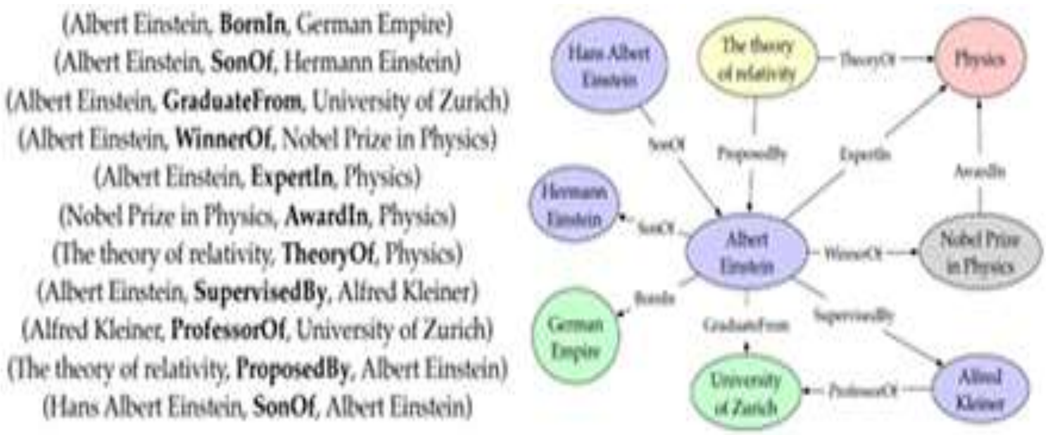

Fig. 1. Example of a General Knowledge Graph [5]

1) Semantic Retrieval: Knowledge Graphs support advanced techniques for data retrieval. When it involves formal semantics, it can be taken as a knowledge base for interpretation and inference over facts [6]. Pattern-based query generation allows the generation of queries based on predefined patterns or templates, capturing specific search patterns or user intents. Query expansion techniques help to broaden the scope of the search by incorporating additional related concepts and entities, enhancing the relevance and coverage of search results. Additionally, entity and relationship extraction techniques leverage natural language processing algorithms to extract entities and relationships from user queries, enabling the generation of structured queries that capture the user's intent accurately [7]. These capabilities of Knowledge Graphs empower search systems to deliver more precise and contextually relevant search results.

In summary, Knowledge Graphs provide an effective approach to representing and retrieving highly interconnected data. By enabling semantic data representation and retrieval, they offer a comprehensive understanding of complex relation- ships and hidden patterns. Leveraging the power of Knowledge Graphs, search systems can deliver more accurate, context- aware, and personalized search results, catering to the evolving needs of businesses and users in the high-dimensional data landscape.

2.2 Types of Nodes

Having laid the groundwork by understanding the funda- mentals of knowledge graphs and their features, we now delve further into their intricacies, examining the potential of nodes—recognized as the fundamental building blocks within a knowledge graph. Nodes form an essential component of Knowledge Graphs as they represent entities, concepts, and

relationships, playing a vital role in organizing and structuring the data. These nodes provide us with a more rich and interconnected representation which enables the knowledge graph to provide a holistic view of the data, allowing for more sophisticated analysis, reasoning, and exploration.

A knowledge Graph usually consists of nodes and edges that represent entities and relationships between them. Two key types of nodes in a knowledge graph are schema nodes and instance nodes. Schema nodes, also known as class nodes or type nodes, represent the high-level categories or concepts in the knowledge graph. They define the schema or ontology of the graph and provide a hierarchical structure for organizing the data. Schema nodes represent the types or classes of entities and define their common attributes and relationships. For example, in a knowledge graph about books, a schema node could be "Book" with attributes such as "Title", "MRP", "Number of pages", "Publication Year" and so on. Instance nodes also referred to as entity nodes, represent specific instances or occurrences of the concepts defined by schema nodes. They represent individual entities or objects in the knowledge graph that possess specific attribute values and participate in relationships. For instance, in a book knowledge graph, an instance node could be "Harry Potter and the Philosopher's Stone" with attribute values like "Title: Harry Potter and the Philosopher's Stone", "price: Rs.500" and "Publication Year: 1997"

2.3 Query Generation Using Different Node Types

Schema nodes and instance nodes utilization in query generation enables powerful and flexible exploration of the knowledge graph. Schema nodes provide a structural framework and define the attributes and relationships that can be queried. They allow for specifying high-level categories or constraints for the desired information. Instance nodes, on the other hand, represent the actual data points in the knowledge graph and allow for specific information retrieval. By leveraging schema nodes, query generation can involve querying for entities that belong to a specific category or class, such as retrieving all books or all authors. It enables broad exploration and filtering of data based on common attributes and relationships defined by the schema. Instance nodes, on the other hand, enable targeted querying by specifying specific attribute values or relationships, allowing for precise information retrieval. The combination of schema nodes and instance nodes in query generation provides a powerful mechanism to explore and retrieve relevant information from the knowledge graph. It allows for both high-level explorations based on categories and detailed querying based on specific attributes and relationships, enabling efficient and effective utilization of the knowledge graph's rich interconnectedness.

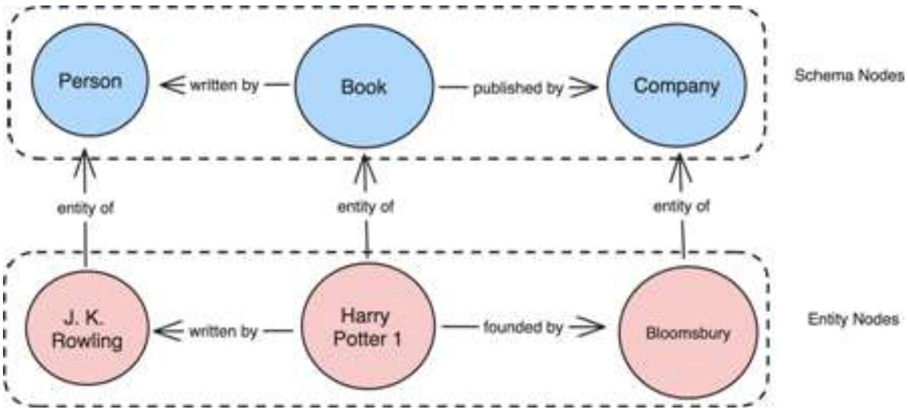

Fig. 2. Different types of nodes in a Knowledge Graph

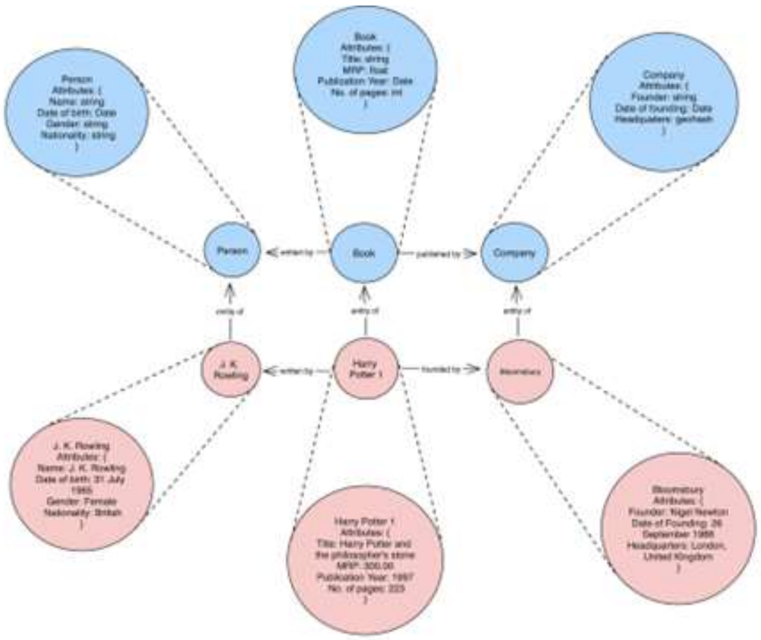

Fig. 3. Example of a Knowledge Graph

3. METHODOLOGY

In the realm of knowledge graph searching, a multitude of strategies have emerged in recent years, offering promising solutions. Factoid searching [8], [9], [10], [11], employs methods such as keyword searching [8] or triplet searching, sometimes enhanced with keywords [12]. However, these approaches are typically confined to one-hop queries, encompassing nodes, attributes, and their immediate neighbors. On the other hand, Semantic searching [13], [14–16] often involves a two-step process: initially transforming a natural language question into an RDF query, followed by executing the query [14–16], or alternatively, converting natural language questions into a semantic query graph and matching the subgraph to retrieve the answer [13]. This methodology, however, encounters limitations when dealing with two unconnected nodes present in the question. Our approach addresses this issue by introducing n-degree prompt generation, thereby expanding the scope and effectiveness of knowledge graph searching.

3.1 N-degree Prompt Generation

User prompts can range from generic queries like ”White Shoes” to more specific queries such as ”White Shoes less than 10K” or ”Hrx White Shoes”. By considering a wide range of prompts, we aim to cover diverse user search intents and requirements. Certain queries, like ”White Shoes” or ”White Shoes less than 10K,” can be easily addressed as they are directly connected to node ”Shoes” within the knowledge graph. These queries can be handled through straightforward template-based queries. However, challenges arise when dealing with queries like ”Hritik Roshan white shoes” or ”Hritik Roshan shoes” where the entity ”Hritik Roshan” is not an attribute directly associated with ”Shoes.”

In the initial phase of our methodology, we focus on generating all possible search prompts that a user may enter in a search box or chatbox. To achieve this we go through all the nodes in a knowledge graph one at a time. For a given node, all its attributes are identified and recorded, which we will refer to as n=0 degree. These attributes represent the different characteristics or properties associated with the node. For instance, if the selected node is a shoe, attributes could include size, color, material, and brand.

Once the attributes of the node are identified, the next step is to generate queries for each attribute. This involves constructing SQL queries based on the attribute values. For example, if the attribute is size, templates like

```
SELECT * FROM shoes WHERE size > [NUMBER];
```

and a corresponding prompt “shoes where size is bigger than 10” is formulated. This process is repeated for all the attributes associated with the node. All such queries and prompts are then stored. To ensure the adequacy of the generated queries, we evaluate them against a given database or dataset. This evaluation process allows us to determine if the queries cover the desired search prompts sufficiently. While the node-based queries provide a solid foundation for search prompt generation, they may not encompass queries connected to the node via edges or predicates. To overcome this limitation, our methodology extends the query generation process to include predicates associated with the selected node (n=1 degree). Predicates typically describe the type of relationship between two entities. They provide information about the connections, properties, or characteristics of the entities within the graph. Therefore, we move beyond the selected node and systematically create queries that consider all possible predicates associated with the chosen node. Continuing with the shoe example, if the connected node is the brand, templates such as

```
SELECT * FROM shoes
WHERE brand = [ORGANISATION];
```

and prompts such as “Nike Shoes” can be generated. This process is repeated for all the connected nodes associated with the given node. All such queries and their corresponding prompts are then stored. Similar to the node-based queries, we evaluate the generated predicate-based queries against the database to ensure their coverage of relevant search prompts. This iterative process allows us to not only check the coverage of the query database but also refine the queries and consider various filters and conditions associated with the predicates, ensuring the completeness and accuracy of the generated search prompts. This step will help us generate templates for the majority of our queries. Yet, queries in which the given entities have no direct connections are yet to be tackled.

An everyday example of such queries is an association of products with brand ambassadors or specific individuals rather than the company or entity itself. Traditionally, these queries are tackled by adding the associated individuals in the name of the product itself. But to generate accurate prompts for such queries requires us to go a step further to enhance the scope of our search prompt generation and go beyond traditional approaches by exploring queries for predicates of predicates. By considering these associations, we generate queries that cover predicates associated with the given predicates (n=2 level). For example, if we have a predicate connecting ”Shoes” and ”Brand,” we further explore predicates that connect the ”Brand” node to individuals like ”Founders” or ”Celebrity Ambassadors” (on schema-level) or connect ”HRX” to “Hritik Roshan” (on instance level). This enables us to address queries such as ”Hritik Roshan shoes” or ”Shoes endorsed by celebrity ambassadors.” Hence, we further explore connections in the knowledge graph by considering nodes that are connected to the previously connected nodes. Queries are generated for these connections with the given node, and the process is repeated for attributes for these nodes. Examples of such templates can include things like

```
SELECT * FROM shoes
WHERE founder.name = [PERSON];
```

and prompts such as “Hrithik Roshan shoes”. By incorporating predicates of the predicate, we expand the range of search prompts we can handle, capturing more nuanced and specific user search intents. By reaching n=2 degree we cover the entire database with comprehensive and contextually relevant search prompts.

3.2 Prompt to Query Mapping and Storage

All the generated queries and prompts from the above steps are saved in a query storage repository. The repository serves as a collection of not only predefined queries, but it includes the retrieval steps for the queries that are required to retrieve the desired information. By incorporating the retrieval steps along with the stored queries, the methodology enhances the query generation and execution process. The retrieval steps provide a detailed guide on how to navigate the knowledge graph to obtain the desired information. This approach en- ables efficient and accurate query execution by following the prescribed steps, leading to effective results presentation. The inclusion of retrieval steps with the stored queries in the query storage repository ensures that the methodology has a comprehensive understanding of how to obtain the desired information from the knowledge graph. It enables the system to execute queries more efficiently, reducing the computational overhead associated with query interpretation and retrieval.

3.3 Query execution and Back Propagation

When a new query is received, instead of immediately converting it to an SQL query, the methodology employs query annotation techniques. The received query is annotated using NLP techniques to identify category (schema) nodes and Named Entity Recognition to recognize instance nodes. Using them, we create a prompt that is compared with the queries stored in the query storage repository. The goal is to identify if there are any existing queries that can satisfy the semantics or intent of the new query. The annotated query is matched against the stored queries by converting it to word embeddings and using cosine similarity to determine its semantic similarity. If a match is found, the corresponding pre-defined query and its associated retrieval steps are retrieved from the query storage. These retrieval steps describe the necessary actions or operations to be performed to obtain the desired information. After the retrieval process is successful we add this prompt to our query storage repository so that instead of deploying semantic similarity to queries, we apply these prompts for efficiency and accuracy. We have observed that 20% of the user search prompts lead to 80% of the traffic hence these 20% prompts/queries are highly significant for us and we store this in our cache layer to make it even faster. Continuing with the shoe example, if a query such as “Show shoes that Hrithik Roshan wears” is received, after employing annotation techniques, we will end up with a prompt “Hrithik Roshan shoes”. This prompt is then compared against all the stored queries using cosine similarity. Once we find an appropriate query, something like

```
SELECT * FROM shoes
WHERE founder.name = [PERSON];
```

we retrieve the execution steps associated with this query and are easily able to navigate the knowledge graph and reach at the brand with which “Hrithik Roshan” is associated, viz ”HRX”. This not only increases the execution time but can also increase the accuracy of search results. This prompt is added to the query storage repository for future use. The methodology anticipates that going two nodes deep in the graph will cover a significant majority of the queries received for the selected node. In case, we come across an annotated query that doesn’t match against the stored queries, the methodology creates an appropriate query and its retrieval steps. This newly generated information is then added to the query storage repository for future use, improving the system’s query-handling capabilities with each new input.

4. RESULTS AND DISCUSSION

Our model was tested on a dataset comprising approxi- mately 2000 suitable queries corresponding to our knowl- edge graph.

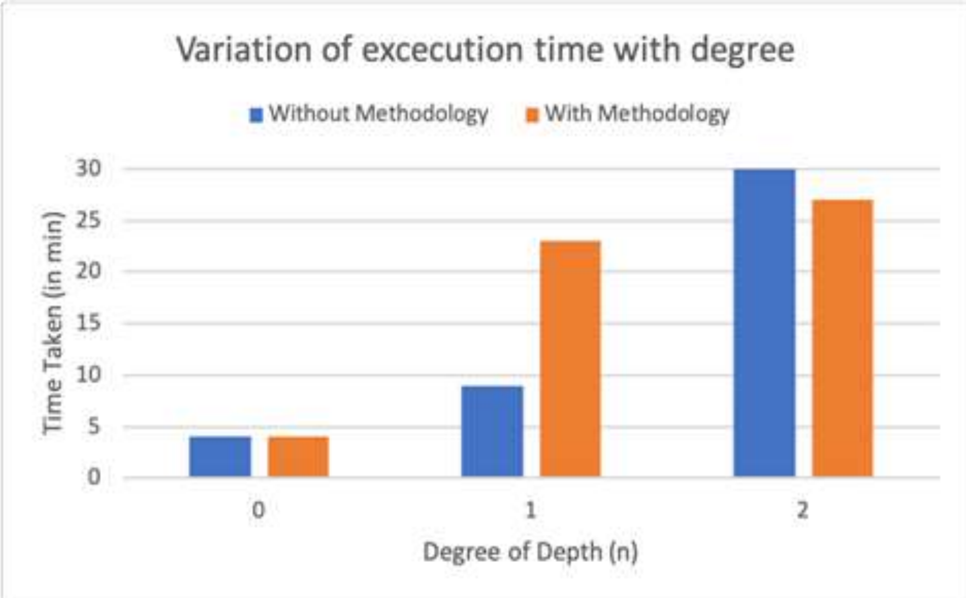

Fig. 4. Variation in execution time with degree of depth (n)

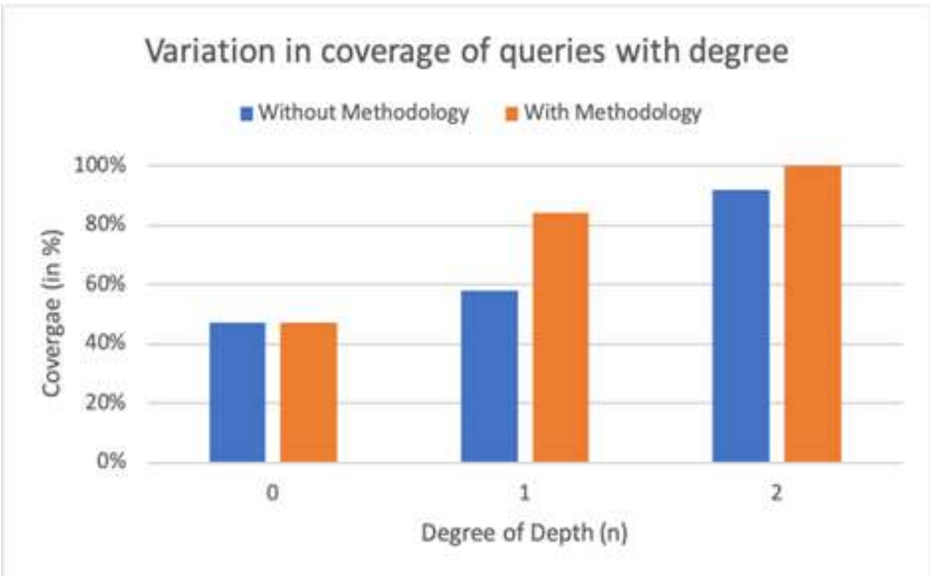

Fig. 5. Variation in coverage of queries with degree of depth (n)

However, to obtain more precise and reliable results, further testing on a larger dataset is necessary, and the outcomes should be updated accordingly. In the initial phase of our approach, we focused on evaluating the cov- erage of queries based on node attributes (n=0 level). The results revealed that our methodology achieved coverage of approximately 47% of the dataset. This indicates that our approach successfully generated search prompts and queries for a significant portion of the entities within the knowledge graph. It's worth noting that this phase, which involved it- erating through attributes, was relatively straightforward in terms of complexity. Next phase of the methodology, we proceeded to test the performance on predicates associated with nodes in the knowledge graph. Combining these results with the queries and prompts generated from attributes, we achieved a significant improvement, covering approximately 84% of the dataset. This represents a substantial increase in coverage compared to our previous testing. However, it is worth noting that this phase required additional time due to the increased complexity per node. In the final phase of our methodology, we conducted testing on predicates of predicates, and added it to our query repository, considering all possible query combinations for each node. The results highlighted the exceptional effectiveness of our methodology, achieving an impressive coverage rate of approximately 99.9% of the dataset. This high level of coverage indicates that our approach successfully generated search prompts and queries for nearly possible queries that can be generated for enti- ties resent in our knowledge graph. It is important to note that the complexity significantly increased during this phase, as each node had to consider its attributes, predicates, and predicates of predicates, adding a higher level of intricacy to the process. In addition, we conducted an evaluation by executing queries from the dataset relevant to our knowledge graph. We compared the results obtained without using our methodology to those obtained with our methodology. The findings revealed a clear difference in performance. When the methodology was not utilized, only approximately 80% of the queries produced relevant results. However, when our methodology was employed, it successfully provided relevant results for approximately 97% of the queries. This stark contrast demonstrates the effectiveness of our approach in enhancing the relevance and accuracy of query results.

5. CONCLUSION

In this paper, we presented an efficient search prompt gen- eration in advance using the knowledge graphs by exploiting the attributes of nodes and exploring two degrees connected nodes, we were able to generate a comprehensive collection of queries. The incorporation of query annotation and matching techniques improved the query execution process by iden- tifying pre-defined queries that can fulfill the semantics of incoming queries. Storing queries along with their retrieval steps allowed for precise and consistent data retrieval. Our methodology significantly reduced the computational overhead of query conversion and retrieval, resulting in faster re- sponse times for query execution. The experiments conducted demonstrate the feasibility and effectiveness of our approach in harnessing the power of adjacent nodes and attributes in knowledge graphs for efficient search query generation. Overall, our work contributes to the advancement of query- generation techniques in knowledge graphs, providing a more streamlined and effective approach for retrieving information from interconnected data. The utilization of adjacent nodes and attributes, along with query annotation and matching, opens up new avenues for enhancing the querying capabilities of knowledge graph systems. Future research can focus on expanding the methodology to handle more complex queries and investigating optimizations to further improve query gen- eration and execution efficiency.

FUTURE WORK

As Stephen Hawking said, the twenty-first century is an era of complexity, we believe this work can be extended to cater to the diverse user prompts which are possible in the near future and we can develop a pseudo-likelihood prompt generation approach that can help to generate prompts in advance with more coverage and lesser exploration time.

COMPETING INTERESTS

Authors have declared that they have no known competing financial interests OR non-financial interests OR personal relationships that could have appeared to influence the work reported in this paper.

REFERENCES

1. Dong X, Gabrilovich E, Heitz G, Horn W, Lao N, Murphy K, Strohmann T, Sun S,

Zhang W, Knowledge vault: A web-scale approach to probabilistic knowledge fusion. in SIGKDD. ACM. 2014;601–610.
2. Nickel M, Murphy K, Tresp V, Gabrilovich E. A review of relational machine learning for knowledge graphs. Proceedings of the IEEE, 2016;104(1):11–33.
3. Wang Q, Mao Z, Wang B, Guo L. Knowledge graph embedding: A survey of approaches and applications. IEEE TKDE, 2017;29(12): 2724–2743.
4. Hogan A, Blomqvist E, Cochez M, d'Amato C, de Melo G, Gutierrez C, Gayo JEL, Kirrane S, Neumaier S, Polleres A et al. Knowledge graphs, arXiv preprint arXiv:2003.02320;2020.
5. Ji, Shaoxiong et al. A survey on knowledge graphs: Representation, acquisition, and applications. IEEE transactions on neural networks and learning systems 2021;33(2): 494-514.
6. Bordes A, Weston J, Collobert R, Bengio Y. Learning structured embeddings of knowledge bases, in AAAI. 2011;301–306.
7. Lin Y, Han X, Xie R, Liu Z, Sun M. Knowledge representation learning: A quantitative review, arXiv preprint arXiv:1812.10901;2018
8. Yang S, Han F, Wu Y, Yan X. Fast top-k search in knowledge graphs. In 2016 IEEE 32nd international conference on data engineering (ICDE). IEEE. 2016; 990-1001.
9. Khan A, Wu Y, Aggarwal CC, Yan X. Nema: Fast graph search with label similarity. Proceedings of the VLDB Endowment. 2013;6(3):181-192.
10. Jin J, Khemmarat S, Gao L, Luo J. Querying web- scale information networks through bounding matching scores. In Proceedings of the 24th International Conference on World Wide Web. 2015;527-537.
11. Zou L, O¨ zsu MT, Chen L, Shen X, Huang R, Zhao D. gStore: a graph-based SPARQL query engine. The VLDB journal. 2014;23:565-590.
12. Hiba Arnaout, Shady Elbassuoni. Effective searching of RDF knowledge graphs, Journal of Web Semantics. 2018;48:66-84. Available:https://doi.org/10.1016/j.websem.2017.12.001 ISSN 1570-8268
13. Zou L, Huang R, Wang H, Yu JX, He W, Zhao D. Natural language question answering over RDF: a graph data driven approach. In Proceedings of the 2014 ACM SIGMOD international con- ference on Management of data. 2014;313-324.
14. Yahya M, Berberich K, Elbassuoni S, Ramanath M, Tresp V, Weikum G. Natural language questions for the web of data. In EMNLP- CoNLL. 2012;379–390.
15. Cimiano P, Lopez V, Unger C, Cabrio E, Ngomo A-CN, Walter S. Multilingual question answering over linked data (qald-3): Lab overview. In CLEF. 2013;321–332.
16. Lopez V, Unger C, Cimiano P, Motta E. Evaluating question answering over linked data. J. Web Sem., 2013;21:3–13.

© 2023 Gupta and Tiwari; This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Peer-review history:
The peer review history for this paper can be accessed here:
https://www.sdiarticle5.com/review-history/109603