Mapana – Journal of Sciences
2024, Vol. 23, No. 3, 45-60
ISSN 0975-3303|https://doi.org/10.12723/mjs.70.3

Knowledge Graph Generation for Research Articles

Dhanalakshmi Teekaraman* and Tamizhselvi S P†

Abstract

The increasing amount of web data has made accessing and processing information efficiently challenging. One solution is to transform the unstructured data into a machine-readable structured format like a knowledge graph. This paper addresses an information retrieval system that enables users to formulate queries in natural language and obtain pertinent information from a knowledge graph specific to a particular domain. Understanding any unstructured data is tougher than structured data. Inferring knowledge from any research article is difficult for naïve users. To resolve this, we propose to create a knowledge graph for the same. Our system utilizes natural language processing techniques to analyze user queries, generating SPARQL queries to retrieve pertinent data from the knowledge graph. We leverage state-of-the-art knowledge graph models to assess the system's precision, recall, and F1 score. The result shows that the proposed model can effectively retrieve relevant information with an average precision of approximately 95%.

Keywords: Query System · Triples · Natural Language Processing. SPARQL · RDF

* Department of Computer Science and Engineering, Jerusalem College of Engineering Chennai, India – 600100; dhanalakshmi.t@jerusalemengg.ac.in

† School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, India – 632014; tamizhselvi.sp@vit.ac.in

1. Introduction

The rapid expansion of data on the internet has led to the challenge for users to access and process information efficiently. The exponential surge in data has created a demand for sophisticated tools capable of efficiently handling and processing extensive datasets. Knowledge graphs have surfaced as a potent tool for encapsulating structured knowledge in a format interpretable by machines. A knowledge graph constitutes a knowledge representation system structured as a graph, capturing information in a semantic format. It provides a means for organizing and storing data and representing the relationships between different pieces of information. Over the past few years, knowledge graphs have gained popularity and found applications in diverse fields, including search engines, recommendation systems, and chatbots.

One of the most significant advantages of knowledge graphs is their ability to facilitate natural language processing. Offering a semantic representation of information, knowledge graphs serve various applications, including question-answering systems. Through the utilization of a knowledge graph, a question-answering system allows users to pose questions in natural language and access pertinent information. The contribution of the proposed model is given below.

- Triples Generation
- Knowledge Graph Creation
- Knowledge Extraction
- SPARQL Query Conversion

The organization of the paper is as follows. The various existing knowledge graph representation models are described in Section 2. The proposed work and methodology used in knowledge graph generation in elaborated in Section 3. Section 4 succinctly examines the results and discussion, while Section 5 concludes by summarizing key findings and suggesting avenues for future research.

1. Related Work

Over the past few years, there has been a significant increase in the research and development of knowledge graphs and their applications in different areas. Knowledge graphs are an invaluable tool for presenting and organizing complex information in a structured way, which enables efficient and accurate information retrieval and analysis. The paper explores a survey on the fundamental techniques employed in question-answering systems that operate over knowledge bases [6]. An effective technique for enhancing the performance of Graph Neural Networks (GNNs) [16] is self-supervised pre-training. This method has proven to be valuable across a range of applications involving GNNs. The paper titled "Self-Supervised Message Passing Transformer on Large-Scale Molecular Data" introduces the GROVER model, which is an example of a Graph Neural Network (GNN) demonstrating promising outcomes even in the absence of supervision. Another area where knowledge graphs have been extensively used is medical report generation. The paper in [11] aims to create medical reports without supervision by using knowledge graphs to bridge the visual and textual domains. However, this approach has limitations, including the need for independent sets of medical images and medical reports, which can be difficult to obtain. Gad-Elrab et.al. built a framework for presenting facts through knowledge graphs and text [7]. The author in [8] provides an overview of graph embedding and discusses its problems, techniques, and applications. Microsoft has released an academic graph [9], when experts are not enough to build the graph.

Moreover, there have been efforts to leverage knowledge graphs for COVID-19-relatedresearch, such as the paper Deep Learning-based KG for COVID-19 which uses a COVID-19 perspective language model by fine-tuning the BERT. However, this approach is limited to the COVID-19 perspective, as it analyses METEOR scores. Furthermore, the paper [15] discusses techniques for pre-training graph neural networks, incorporating both node-level and graph-level pre-training along with a powerful Graph Neural Network (GNN). However, acknowledges the necessity of improving GNN architectures, as well as using pretraining and fine-tuning approaches to enhance generalization. Another area of research that has gained significant attention is the use of natural language to query knowledge graphs. The paper [10] highlights the introduction of an automated Tree-LSTM-based neural network model tailored for querying knowledge graphs in natural language. This model is designed to facilitate the translation of natural-language questions into SPARQL queries. And, it has the potential to elongate the time required for computing the similarity between natural language questions and SPARQL queries. Furthermore, explainable AI methods have been proposed to interpret the predictions obtained via knowledge graph embeddings, as demonstrated in the paper Knowledge Graph Embeddings and Explainable AI [3]. However, making comparisons between different methods is often difficult and not directly possible. The paper [12] offers a concise overview of a comparative analysis of Knowledge Graph Embedding for Link Prediction. The author talked about a survey on Knowledge graphs, which encompassed the concepts of representation, acquisition, and applications [13]. Lastly, DGL-KE [5] is an open-source package that offers efficient computing for knowledge graph embeddings, but it recognizes the limitations of mini-batch training.

The knowledge graphs are widely applied in industries such as Healthcare, IoT, Finance & Banking, Media & Entertainment, Search Engines, Cybersecurity, E-commerce & Retail. Gartner (https://www.gartner.com) predicts that knowledge graphs will be a part of 80% of data and analytics innovations by 2025. As per this literature, we found that the knowledge graph for the research article is not touched upon yet and also it is indeed useful for naïve or inexperienced researchers. Hence, we aim to develop a knowledge graph and query system.

3. Proposed Work and Methodology

Our system is built around a knowledge graph that has structured knowledge about a specific domain. By creating a knowledge graph, the system can better capture the complex relationships between entities and make it easier for users to explore the information. A triple-store database is used to store the knowledge graph and query it using the SPARQL query language. The system uses the Stanford CoreNLP library that supports natural language processing techniques for analyzing the input user question and generating a corresponding SPARQL query. The query further retrieves the relevant information from the knowledge graph.

3.1. Architecture

The workflow diagram is shown in Fig. 1 which describes the flow of data and processing steps in order to generate a knowledge graph and how natural language questions are converted into SPARQL queries to retrieve relevant information for the user.

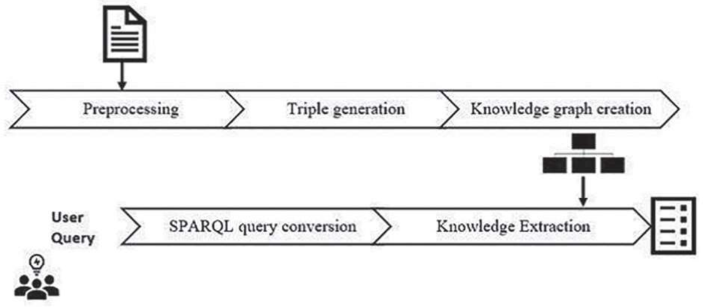
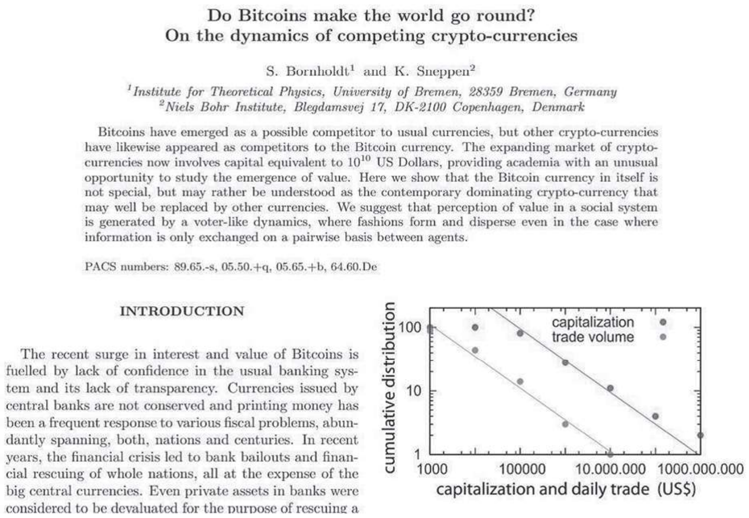


Fig. 1. Work Flow Diagram

1. The unstructured data, which is our raw text data is processed through a pre-processing step, which typically involves techniques such as tokenization, stop word removal, and stemming to clean the data and prepare it for further processing.
2. The data undergoes a post-pre-processing phase where it is subjected to triple extraction and cleaning. This procedure involves obtaining relevant text and converting it into triples (subject-predicate-object statements) that can be utilized. This step also involves cleaning the extracted triples to remove any noise or irrelevant information.
3. Following the triple extraction and cleaning step, the subsequent phase involves constructing a knowledge graph based on the refined triples. In this step, relationships between entities in the data are represented in a structured and interconnected manner. The knowledge graph can answer natural language questions about the data.
4. The final step involves employing a neural network model to translate natural language inquiries into structured query language (SPARQL) queries. This model serves as a bridge between human-readable queries and the structured format necessary for querying the knowledge graph.

3.2. Data Collection and Pre-Processing

Collect the user input, which is a set of research papers related to the cryptocurrency domain in our case. The module accepts input _les in the PDF format and applies specific operations to extract text from the PDF files. The sample input file content is shown in Fig. 2. The pre-processing module plays a crucial role in the overall architecture of our project. This takes place in two steps (i) Text extraction and (ii) Pre-processing.



Do Bitcoins make the world go round?
On the dynamics of competing crypto-currencies

S. Bornholdt[1] and K. Sneppen[2]

[1]*Institute for Theoretical Physics, University of Bremen, 28359 Bremen, Germany*
[2]*Niels Bohr Institute, Blegdamsvej 17, DK-2100 Copenhagen, Denmark*

Bitcoins have emerged as a possible competitor to usual currencies, but other crypto-currencies have likewise appeared as competitors to the Bitcoin currency. The expanding market of crypto-currencies now involves capital equivalent to 10^{10} US Dollars, providing academia with an unusual opportunity to study the emergence of value. Here we show that the Bitcoin currency in itself is not special, but may rather be understood as the contemporary dominating crypto-currency that may well be replaced by other currencies. We suggest that perception of value in a social system is generated by a voter-like dynamics, where fashions form and disperse even in the case where information is only exchanged on a pairwise basis between agents.

PACS numbers: 89.65.-s, 05.50.+q, 05.65.+b, 64.60.De

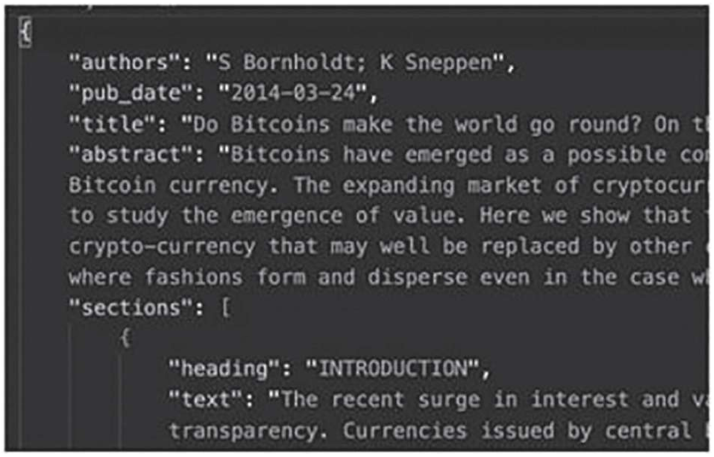
INTRODUCTION

The recent surge in interest and value of Bitcoins is fuelled by lack of confidence in the usual banking system and its lack of transparency. Currencies issued by central banks are not conserved and printing money has been a frequent response to various fiscal problems, abundantly spanning, both, nations and centuries. In recent years, the financial crisis led to bank bailouts and financial rescuing of whole nations, all at the expense of the big central currencies. Even private assets in banks were considered to be devaluated for the purpose of rescuing a

Fig. 2. Sample Research Article

Text Extraction: In this module, we leveraged an ML-based parser, GROBID, to extract headers and parse articles in PDF format. By using GROBID, we were able to generate bibliographical information such as the title, abstract, authors, affiliations, and keywords, as well as the sections of the articles. To further enhance our data processing capabilities, we used the scipdf-parser to extract essential information from the articles, specifically the abstract and conclusion. This allowed us to focus on the most important sections of the articles, which are critical for generating accurate and relevant knowledge graphs.

Pre-Processing: Once the text is extracted, the pre-processing steps are applied to clean and transform the text data. These pre-processing steps include removing stop words, stemming, and lemmatization, which are necessary to improve the accuracy of the system. After pre-processing, the module generates a JSON file that contains the extracted information in a structured format as shown in Fig. 3.



```
{
    "authors": "S Bornholdt; K Sneppen",
    "pub_date": "2014-03-24",
    "title": "Do Bitcoins make the world go round? On t
    "abstract": "Bitcoins have emerged as a possible co
    Bitcoin currency. The expanding market of cryptocur
    to study the emergence of value. Here we show that
    crypto-currency that may well be replaced by other
    where fashions form and disperse even in the case w
    "sections": [
        {
            "heading": "INTRODUCTION",
            "text": "The recent surge in interest and v
            transparency. Currencies issued by central
```

Fig. 3. Pre-Processed Research Article

Overall, the pre-processing module provides a robust and efficient way to handle unstructured data and extract valuable information from it. By leveraging advanced parsing techniques and tools, we can streamline the data processing workflow and ensure the accuracy and completeness of our knowledge graphs.

3.3. Triples Extraction

The triples extraction and cleaning module is an important part of our architecture as it helps to extract meaningful information from the pre-processed input text. In this module, we pre-process the input text to remove special characters, lemmatize, and remove sentences without named entity recognition (NER). We then leverage ML-based parsers such as AllenNLP and StanfordOPENIE to process the paragraphs and perform coreference resolution on a paragraph level. This helps to improve the accuracy of our triple extraction process by ensuring that all relevant information is extracted and correctly linked. Fig. 4 shows the triple extraction process. We also use SRLBERT to label semantic roles for each token of the sentence.

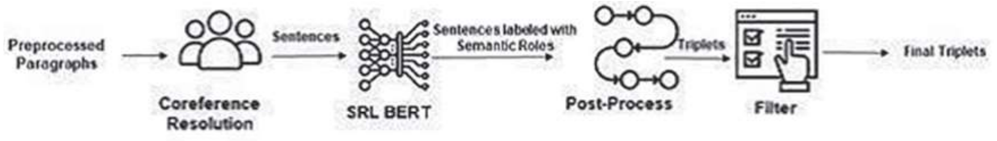
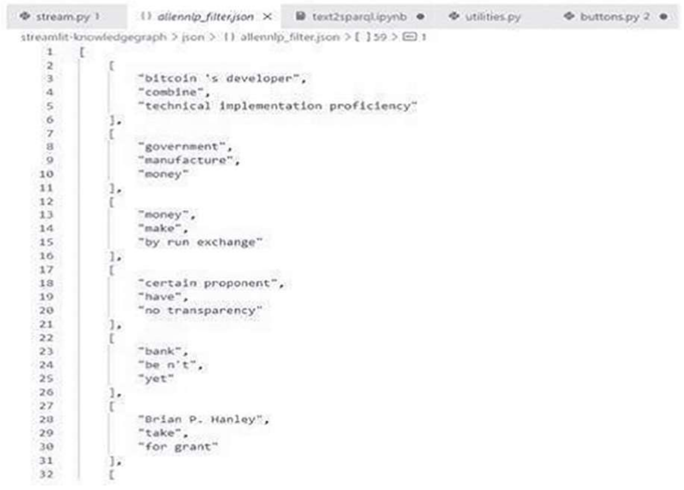


Fig. 4. Triple Extraction Process

For example, a sentence like Bitcoin is a cryptocurrency will be labeled as [ARG0] Bitcoin [V] is [ARG1] a cryptocurrency. We then post-process this data to convert semantic roles to subject, verb, and object, and create triplets.

For example, (ARG0, V, ARG1) --> (S, V, O) --> (Bitcoin, is, a cryptocurrency).


stream.py 1
{} allennlp_filter.json ×
text2sparql.ipynb ●
utilities.py
buttons.py 2 ●
streamlit-knowledgegraph > json > {} allennlp_filter.json > [ ] 59 > ▭ 1

```
[
    [
        "bitcoin 's developer",
        "combine",
        "technical implementation proficiency"
    ],
    [
        "government",
        "manufacture",
        "money"
    ],
    [
        "money",
        "make",
        "by run exchange"
    ],
    [
        "certain proponent",
        "have",
        "no transparency"
    ],
    [
        "bank",
        "be n't",
        "yet"
    ],
    [
        "Brian P. Hanley",
        "take",
        "for grant"
    ],
    [
```

Fig. 5. Sample Extracted Triple

To ensure the quality of our extracted triplets, we apply KnowText filtering to filter out triplets that do not meet our criteria. The cleaned triples are then stored in a Blazegraph server, which is a high-performance graph database that can be used for querying and analyzing the data. Finally, we obtain cleaned and structured triplets that are ready for graph generation and subsequent natural language questions to SPARQL query conversion. The sample extracted triples are shown in Fig. 5.

3.4 Natural Language Question to SPARQL Query Conversion

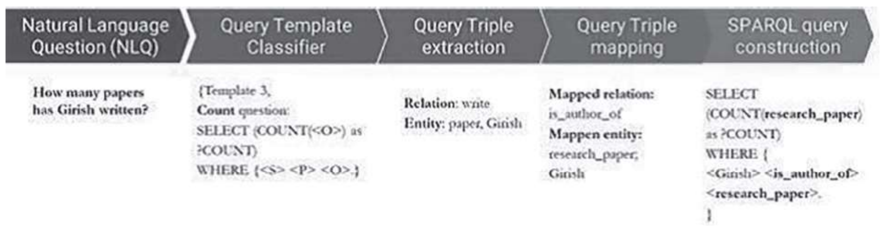
The described module assumes the critical role of taking in a natural language question as input and converting it into a SPARQL query suitable for extracting information from the knowledge graph. Its operation involves leveraging natural language processing techniques to understand the intent embedded in the user's question and subsequently generate a pertinent SPARQL query. To achieve this, the module incorporates several techniques, including:

Named Entity Recognition (NER): This technique identifies and categorizes entities, such as specific names, locations, or organizations, within the user's question.

Part-of-Speech Tagging (POS): POS tagging assigns grammatical categories (like nouns, verbs, adjectives) to each word in the question, aiding in understanding the syntactic structure.

Dependency Parsing: Dependency parsing analyses the grammatical structure of the sentence, establishing relationships between words and determining the hierarchical structure.

By employing these techniques, the module extracts relevant concepts and entities from the user's question, facilitating the construction of a meaningful and contextually appropriate SPARQL query for querying the knowledge graph effectively. Figure 6 shows the NLQ to SPARQL Query Conversion process and is described below.


Natural Language Question (NLQ)
Query Template Classifier
Query Triple extraction
Query Triple mapping
SPARQL query construction
How many papers has Girish written?
Relation: write
Entity: paper, Girish
Mapped relation: is_author_of
Mappen entity: research_paper, Girish

```
{Template 3,
Count question:
SELECT (COUNT(<O>) as
?COUNT)
WHERE {<S> <P> <O>.}
```

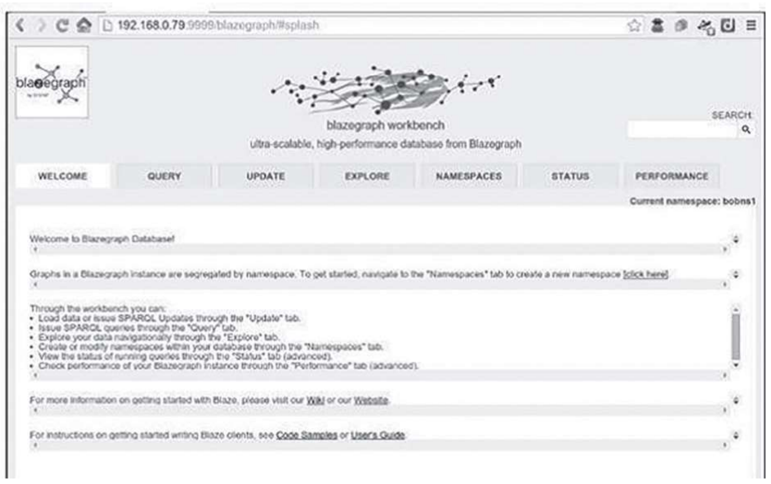
```
SELECT
(COUNT(research_paper)
as ?COUNT)
WHERE {
<Girish> <is_author_of>
<research_paper>.
}
```

Fig. 6. Natural Language to Query Conversion

1. Identify the components of a query which involves recognizing the subject (s), predicate (p), and object or objects (o) in the context of a triple pattern.
2. By analyzing the structure and keywords in the user's question, you can infer the type of SPARQL query that would best address their query intent.
3. Select an appropriate template for generating the SPARQL query.
4. Capturing common patterns in Natural Language Queries (NLQ) and mapping them to the underlying structure of the knowledge graph is a crucial step in natural language processing and knowledge graph querying.
5. Verifying the existence of relationships between entities in the knowledge graph is a crucial step to ensure the accuracy and comprehensiveness of the generated queries.
6. Execute the SPARQL query against a knowledge graph and retrieve the required Information

7. Return the result of the query to the user in a human-readable format, such as a table or list of entities.

Connecting to Blazegraph: We use Streamlit [1], a Python library for building interactive web applications. Here, we ask questions in natural language which is converted into SPARQL queries using NLP techniques. The SPARQL queries fetch answers from the Blazegraph server and the answers are presented to the user as a knowledge graph. Blazegraph is a graph database that enables users to store and query billions of nodes and edges, which makes it ideal for a diverse range of use cases. Fig. 7 shows the simple connection to the Blazegraph and Fig. 8 shows the model selection page created in the proposed work.



blazegraph workbench

ultra-scalable, high-performance database from Blazegraph

Current namespace: bobns1

Welcome to Blazegraph Database!

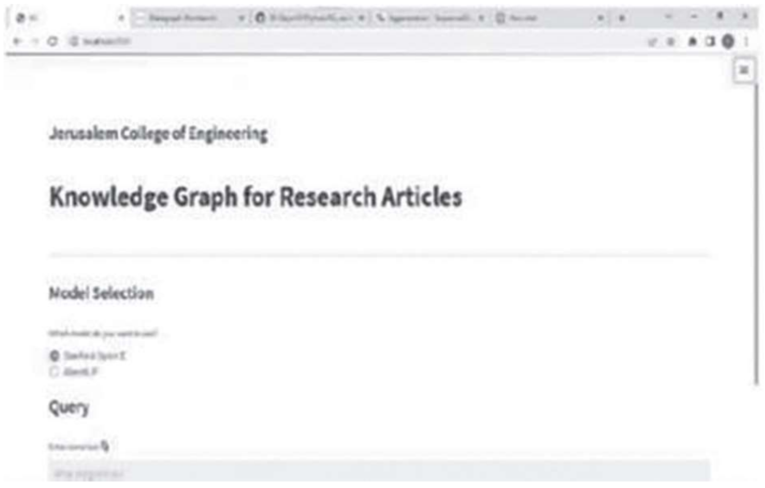
Graphs in a Blazegraph instance are segregated by namespace. To get started, navigate to the "Namespaces" tab to create a new namespace [click here]

Through the workbench you can:
- Load data or issue SPARQL Updates through the "Update" tab.
- Issue SPARQL queries through the "Query" tab.
- Explore your data navigationally through the "Explore" tab.
- Create or modify namespaces within your database through the "Namespaces" tab.
- View the status of running queries through the "Status" tab (advanced).
- Check performance of your Blazegraph instance through the "Performance" tab (advanced).

For more information on getting started with Blaze, please visit our Wiki or our Website.

For instructions on getting started writing Blaze clients, see Code Samples or User's Guide.

Fig. 7. Connectivity with Blazegraph



Jerusalem College of Engineering

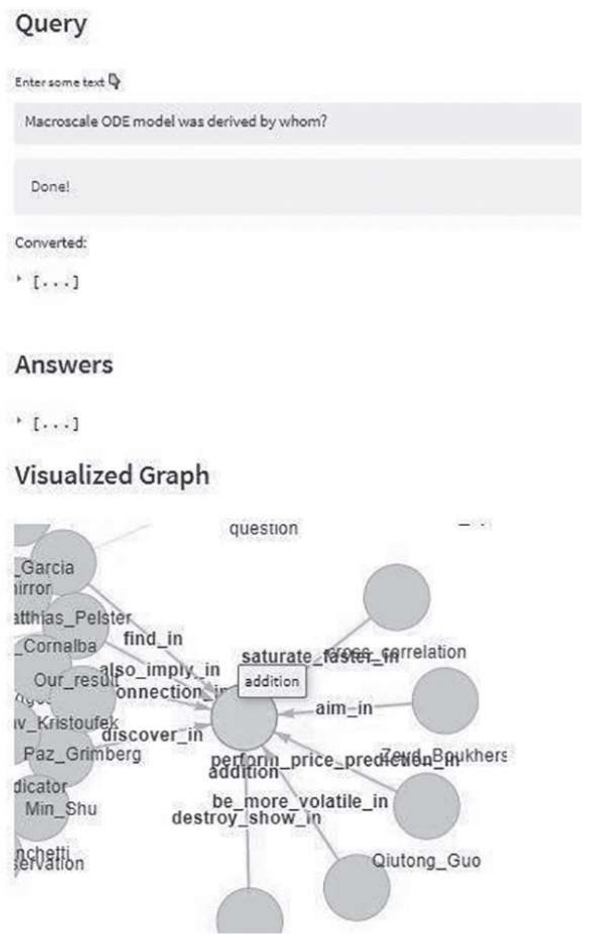
Knowledge Graph for Research Articles

Model Selection

Query

Fig. 8. Triple Extraction Process

Visualization of Knowledge Graph: A visualized graph can be a powerful tool for representing the answer to a query, particularly when dealing with complex or large datasets. By visualizing the data in a graph, patterns, and trends can be quickly identified, providing insights into the relationships and connections between different entities.



Query

Enter some text

Macroscale ODE model was derived by whom?

Done!

Converted:

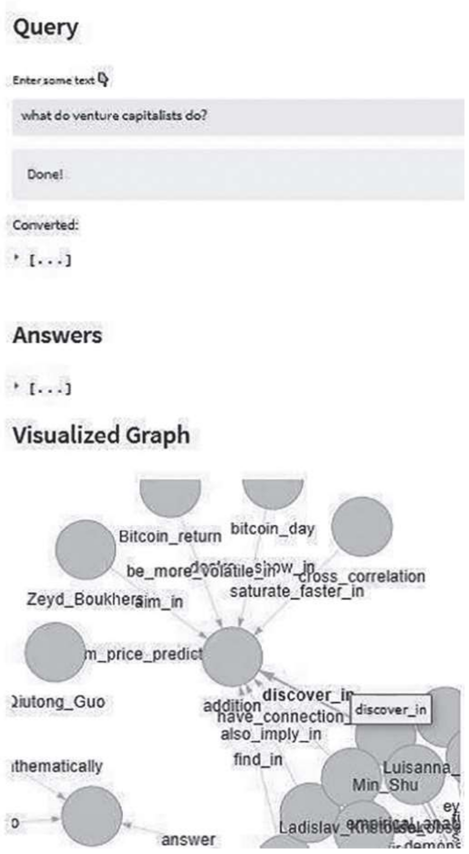
```
▸ [...]
```

Answers

```
▸ [...]
```

Visualized Graph

Fig. 9. Knowledge Graph for Query 1


Query

Enter some text

what do venture capitalists do?

Done!

Converted:

[...]

Answers

[...]

Visualized Graph

Fig. 10. Knowledge Graph for Query 2

For example, a social network could be represented by a visualized graph that shows the relationships between different actors, with each actor being represented as an edge. The various queries raised and the corresponding answers are visualized in the Fig. 9 and Fig. 10.

4. Results and Discussion

We evaluate our system by evaluating it through a series of natural language queries. Our results show that our system can effectively retrieve relevant information from the knowledge graph. We assess the performance of our system by comparing it to a baseline system that employs keyword matching for retrieving information from the knowledge graph. Our system outperforms the baseline system in terms of precision and recall as shown in Fig. 11. The F1-Score is displayed in Fig. 12. From this it is observed that till the 0.2 value of recall the precision value fluctuates in all three cases. But From 0.2 onwards the values of precision in the proposed method are slowly increasing and are also higher than the rule-based method and neighbourhood information method.

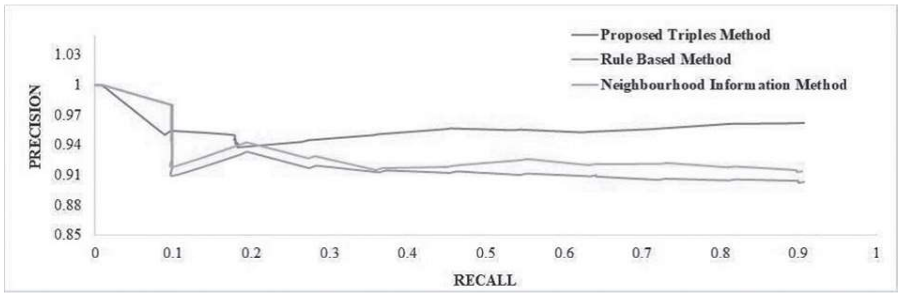


Fig. 11. Precision Vs Recall

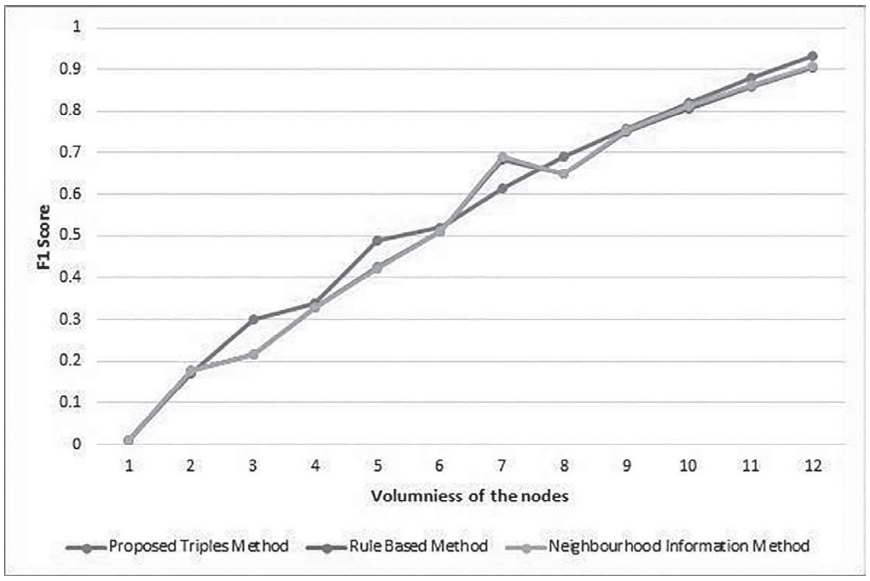


Fig. 12. F1 Score

Illustrative Example Our system identifies the matching triplets based on the query. The precision defines the percentage of relevant triplets from the matching triplets. The recall defines the relevant triplets from total triplets. For example, assume that the total triplets in a knowledge graph is 35. The query is "What is Bitcoin?". The number of matching triplets for the keyword "Bitcoin" that exists in the knowledge graph is 23 and the number of relevant triplets identified by our model is 21. Then the precision is calculated as 21/23 = 0.91. The recall is calculated as 21/35 = 0.6. Similarly, for each query precision and recall is calculated and plotted.

Since the number of samples is smaller, the model can be predicted effectively by precision and recall rather than accuracy. To conclude, the proposed model has greater accuracy in predicting matching triplets than the other two approaches.

In summary, we used Stanford OpenIE [14] and Blazegraph [4] RDF to construct a knowledge graph, AllenNLP [2], and SparqlWrapper to develop a query module. We evaluated the system's performance by employing metrics such as precision, recall, and F1-score on a test dataset comprising research articles. The combination of these methods enabled us to develop an effective query system for research articles based on a knowledge graph.

5. Conclusion and Future Work

Our system exhibits the capability of constructing a query system that can retrieve information from structured data by using a knowledge graph. The proposed query system performs better than the rule-based and neighbourhood information methods. However, in a few cases, our system is unable to extract the matching entities. It can be improved by implementing advanced NLP approaches such as deep learning and deep translation. If the knowledge graph is incomplete or inaccurate, the system's performance will be limited.

There are several potential future enhancements that could be made to the question-answering system based on the knowledge graph for research articles. Some of these include:

- Adding more advanced natural language processing techniques: - Currently, the system uses basic techniques for entity recognition and relationship extraction.
- Knowledge Graph Expansion: - The system can be enhanced to expand the underlying knowledge graph, either by incorporating new data sources or by generating new relationships and entities. This can help to provide more complete and accurate answers to a wider range of questions.
- Integrating with other tools: - The system could be integrated with other research tools, such as reference managers or literature search engines, to provide a more comprehensive research experience for users.

- Expanding the scope of the system: - Currently, the system is focused on research articles in a particular domain. However, the same approach could be applied to other types of text, such as news articles or social media posts. This would require adapting the system to different types of language and text structures.

References

[1]. A faster way to build and share data apps, https://streamlit.io/

[2]. AllenNLP, https://allenai.org/allennlp

[3]. Bianchi, F., Rossiello, G., Costabello, L., Palmonari, M., Minervini, P.: KnowledgeGraph Embeddings and Explainable AI. Knowledge Graphs for eXplainable Artificial Intelligence, Computer Science. (2020)

[4]. Blazegraph Database, https://blazegraph.com/

[5]. Da Zheng, Xiang Song, Chao Ma, Zeyuan Tan, Zihao Ye, Jin Dong, Hao Xiong, Zheng Zhang, and George Karypis.: DGL-KE: Training Knowledge Graph Embeddings at Scale. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR' 20). Association for Computing Machinery, New York, NY, USA, 739-748. https://doi.org/10.1145/3397271.3401172

[6]. Diefenbach, D., Lopez, V., Singh, K., Maret,P. : Core techniques of question answering systems over knowledge bases: a survey. Knowledge and Information Systems 55(2), 529-569 (2018)

[7]. Gad-Elrab, M. H., Urbani, J., Stepanova, D., Weikum, G.: ExFaKT: A framework for explaining facts over knowledge graphs and text. In WSDM 2019 - Proceedings of the 12th ACM International Conference on Web Search and Data Mining, pp.87-95. Association for Computing Machinery, Inc. Melbourne VIC Australia (2019)

[8]. H. Cai, V. Zheng, and K. Chang: A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. IEEE Transactions on Knowledge & Data Engineering, 30(9), 616-1637 (2018)

[9]. Kuansan Wang, Zhihong Shen, Chiyuan Huang, Chieh-Han Wu, Yuxiao Dong, and Anshul Kanakia: Microsoft academic graph: When experts are not enough. Quantitative Science Studies 1(1), 396-413 (2020)

[10]. Liang, Shiqi & Stockinger, Kurt & Mendes de Farias, Tarcisio & Anisimova, Maria & Gil, Manuel: Querying Knowledge Graphs in Natural Language. Journal of Big Data 8(1):3 (2021)

[11]. Liu, Fenglin & You, Chenyu & Wu, Xian & Ge, Shen & Wang, Sheng & Sun, Xu.: Auto-Encoding Knowledge Graph for Unsupervised Medical Report Generation. arXiv:2111.04318 [cs. LG]. (2021)

[12]. Rossi, Andrea & Barbosa, Denilson & Firmani, Donatella & Matinata, Antonio & Merialdo, Paolo.: Knowledge Graph Embedding for Link Prediction: A Comparative Analysis. ACM Transactions on Knowledge Discovery from Data. 2(15), 1-49 (2021)

[13]. S. Ji, S. Pan, E. Cambria, P. Marttinen and P. S. Yu: A Survey on Knowledge Graphs: Representation, Acquisition, and Applications. IEEE Transactions on Neural Networks and Learning Systems, 33(2), 494-514 (2022)

[14]. The Stanford NLP Group, https://nlp.stanford.edu/software/openie.html

[15]. Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, Jure Leskovec: Strategies for pre-training graph neural networks. In International Conference on Learning Representations (ICLR) (2020)

[16]. Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang: Self-supervised graph transformer on large-scale molecular data. In Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS'20), Article 1053, pp.12559-12571. Curran Associates Inc., Red Hook, NY, USA (2020). https://doi.org/10.5555/3495724.3496777