Journal of Discrete Mathematical Sciences & Cryptography
ISSN 0972-0529 (Print), ISSN 2169-0065 (Online)
Vol. 27 (2024), No. 6, pp. 1963–1975
DOI : 10.47974/JDMSC-2047

Semantic search framework over knowledge bases using embeddings-based similarity

Aatif Ahmad Khan
Sanjay Kumar Malik
University School of Information, Communication and Technology
Guru Gobind Singh Indraprastha University
New Delhi
India

Vanita Jain *
Department of Electronic Science
University of Delhi
New Delhi
India

Abstract

With tremendous progress attained towards AI, understanding context from user queries has become a focus area to provide precise answers. In this direction, recent machine learning-based approaches are trying to extract semantic features from queries. Being keywords driven, these approaches underperform to cater semantics in the queries. To overcome this reliance on syntactic representations, semantics-based methods utilizing Knowledge Bases are being augmented to the query processing systems. In this paper, a novel semantic search framework over Knowledge Bases has been proposed and implemented. Semantic similarity between query and predicates is computed using BERT-based embeddings. Additionally, a dataset of semantically similar sentence variations is generated using ChatGPT for analysing the accuracy of the implemented system over YAGO Knowledge Base.

Subject Classification: *(2010) 68T30, 68T35, 68T50.*

Keywords: *Semantic search, Knowledge base, Semantic similarity, SPARQL, Sentence embeddings.*

* *E-mail:* vjain@electronics.du.ac.in

1. Introduction

Day to day web usage by users contains some sort of searching activity, may it be a simple definition check or more complex research about a new biological enzyme. Search queries often contain words or phrases concerned with the type of entity being searched and optionally some intent about nature of target results. As an example, searching weather of a particular city may use a query such as "what is the temperature of Boston in Degree Celsius". In the example query, user intends to see results formatted in "Degree Celsius" notation.

Difficulties in searching includes scenarios such as not finding the concerned information, or finding vague results etc. Such scenarios are faced primarily due to inability of search systems to understand word meanings and to decode the intent and context of the query. String matching-based algorithms are primarily employed for searching tasks thus making it a syntax-oriented problem with less focus on the semantics. Information overload, redundancy and inconsistency further add to this problem, degrading the precision and recall of results. These issues highlight the need of incorporating semantics into the search algorithms.

With tremendous progress in ongoing Artificial Intelligence (AI) based solutions, various machine learning (ML) based approaches are trying to extract semantic features from queries utilizing token embeddings along with Natural Language Processing (NLP) techniques [**1**]. Various state of the art models such as BERT, GLOVE, ChatGPT are assisting the query answering systems and are extensively used in the Chatbots as well. BERT and GLOVE utilizes token embedding to represent the words in a long vector format. These models are pre-trained over large amount of real-world data to understand the context of words as they appear in real world literature. BERT is a well-known transformer extensively used in the literature as well as in the industry. For question answering (QA) approaches and search systems, it is crucial to extract additional knowledge from the KB for deriving context & answering the query [**2**].

Although these models have a lot of advantages and usability, but there are certain disadvantages and considerations which often overweight the pros of these approaches. These approaches are keywords driven and work with probabilities. Thus, these approaches underperform to cater semantics and word variations in the queries. Also, these are heavily dependent on the correctness, scale, and scope of data that was used during their training phase. Thus, proper semantics-based methods

utilizing Knowledge Bases (KB) are being augmented to the query processing systems [**3**].

Semantic Web is the next generation of web with vision of providing structure to data and exploiting relationships among entities to realize a connected ecosystem of structured knowledge. To provide universal accessibility, Universal Resource Identifiers (URI) are used to represent entities, their values and their relationships with other entities. Key technologies of Semantic Web that are standardized for this goal are Resource Description Framework (RDF), Ontology and SPARQL Protocol and RDF Query Language (SPARQL). RDF provides the base support to represent structured knowledge that too in a machine understandable manner. RDF represents entities (subjects) and their attributes (values) in terms of relationships (predicates) in a single RDF triple statement. Different entities are connected by relations among them resulting in a directed graph mesh like structure concerned with a domain.

KB are structured repositories containing connected facts about various entities and relationships. These contain verified facts and are used as trusted knowledge repositories. Large-scale, cross-domain KBs available for real world search applications include DbPedia, WikiData, and YAGO etc. containing multi-millions to billions of RDF triples facts in various serialization formats. SPARQL is a pattern matching based query language to fetch information out of structured KB. As KBs often contains knowledge in RDF Triples, SPARQL queries can identify related information by specifying any combination either subject, predicate or object on large datasets, thus making it an efficient structured information retrieval mechanism. For utilizing RDF KBs, various tools are available such as Apache Jena, Jena Fuseki Server. Their key application areas are recommendation systems, QA chatbots, and sentiment analysis etc. Knowledge-based models provide recommendations based on the semantic similarity with the queried items and considering the constraints defined in the KB [**4**].

Subject identification and relation extraction are two key concerns for effectively processing a query. NLP techniques such as Named Entity Recognition (NER) are heavily utilized for subject identification. This research direction is often referred as Question Answering over Knowledge Bases in literature [**5**].

In this paper firstly, a novel semantic search framework has been proposed and implemented. Semantic similarity between query and predicates is computed using BERT-based embeddings. Secondly, a dataset of semantically similar sentence variations is generated using ChatGPT for

analysing the accuracy on YAGO Date Facts KB. Section 2 presents the literature survey in this direction. Section 3 presents our approach along with key aspects which are discussed in various subsections. Section 4 presents the implementation details for key steps and the generation of dataset. Section 5 presents the results and discussion. Section 6 concludes the paper with future scope.

1.1 *Contributions of the paper*

Contributions of this papers are as following:

- A novel semantic search framework over Knowledge Bases has been proposed and implemented utilizing sentence embeddings and semantic similarity.
- The proposed approach is implemented and tested with YAGO Date Facts with ChatGPT-generated dataset of semantically similar sentence variations, demonstrating effectiveness of the approach.

2. Literature Survey

In this section, various state of the art techniques is explored & summarized with respect to various processing steps for QA over KBs and their key issues [**2, 6**]. At high level, there are six major concerns which are: translating natural language (NL) queries into structured SPARQL queries, preparation of SPARQL query template for representing the questions, approaches for relation extraction using predicate similarity, coping with incomplete information in NL query, query formulation complexity for large KBs and finally, extracting additional knowledge from the KB.

Karim et. al. has proposed a translation approach for NL Job Search queries into SPARQL [**7**]. SPARQL query generation takes place with SELECT and placement of desired information and job skills filters under various conditional clauses. Fatima et. al. has proposed another approach for translating NL free text to formal SPARQL queries [**8**]. Firstly, they have used keyword mapping to identify query keywords and then classified them into formal classes and instances. Based on identified relationships, a query is formalized into SPARQL representation utilizing a set of transformation rules.

Next, for using an effective SPARQL query template, Cocco et. al. has proposed a learning-based approach [**9**]. Training models with a training set comprising of user natural language queries and their corresponding SPARQL formulations, may generate a generalized SPARQL template for

the QA task. For, predicate similarity approaches, Gayathri et. al. has proposed a partitioning approach for RDF summarization [**10**]. They are using clustering on RDF triples which are having greater semantic similarity among the predicates.

For dealing with incomplete information from NL query, Ge et. al. proposed to fetch approximate information out from the KB [**3**]. In another approach proposed by Rafes et. al., they utilize the query history for suggesting inputs or the new queries [**11**]. Using similarity calculations with previously executed queries, autocomplete suggestions are provided for SPARQL queries in the form of snippets.

For resolution of query formulation complexity due to unknown structure or nature of the target KB, Campinas et. al. has proposed an assisted query formulation mechanism [**12**]. In another approach in this direction Kadir et. al. has proposed formulation of formal SPARQL queries using ML [**13**]. It can automatically disambiguate unknown keywords in the query using the inference engine for specified KB. Ali et. al. has proposed an approach to infer new knowledge using Construct SPARQL query [**14**]. Results are returned in the form of a RDF graph, which can be used as a unit for further querying.

3. Proposed Approach & Key Aspects

This section presents the proposed semantic search approach using sentence embeddings and semantic similarity. Figure 1 depicts the key modules, aspects and process flow. The user provided natural language query Q is processed with tokenization and part of speech tokens which are later transformed into sentence embeddings. These are semantically compared against the embeddings generated from RDF KB predicates. Finally, the RDF predicate is selected based on maximum semantic similarity score.

Another key aspect is the RDF Subject *S* identification from the natural language query. NLP techniques such as Named Entity Recognition (NER) are employed for this. These aspects along with an end-to-end approach for KB subject entity lookup is discussed in [**15**]. Also, some logic based on processing the noun tokens in the query often assists in proper subject identification [**16**].

Once both the RDF subject and RDF predicate are identified, a parameterized SPARQL query is framed and executed on the SPARQL endpoint of the KB to extract the concerned object(s) of interest. The response RDF object is processed and formatted to provide a human

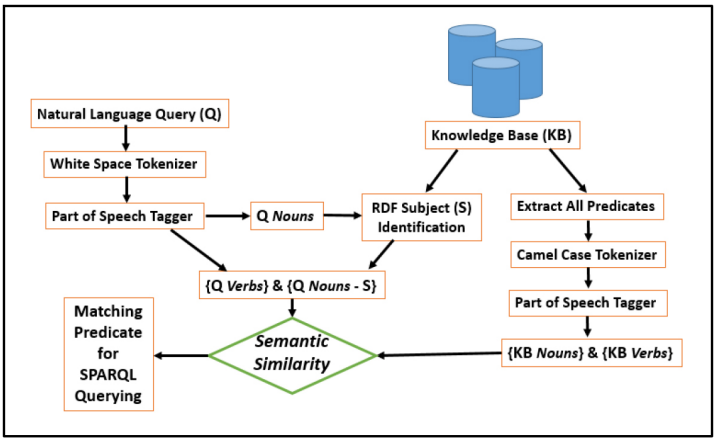

Figure 1

Proposed Semantic Search approach using sentence embeddings and semantic similarity

readable output response for the original natural language query. Thus, making this complete flow for QA system over KB.

3.1 *Processing Workflow*

Query workflow: Processing input query and Subject Identification

1. Input query *Q* is pre-processed: Tokenization and Part of Speech Tagging is performed.
2. Tokens tagged as noun are processed and later matched with subjects available in the KB to identify the RDF subject arguments of the SPARQL query.
3. A set of query tokens containing verbs and nouns (excluding the Subject identified) are stored for further processing to calculate similarity comparison with predicate tokens embeddings.

KB Predicate workflow: Fetching and processing RDF Predicates from KB

4. RDF KB is setup in graph database format and its SPARQL endpoint is exposed for querying.
5. KB is queried to fetch available predicates. A set of predicate tokens is stored for further similarity comparison with query tokens embeddings.

Semantic Similarity comparison for RDF predicate matching and SPARQL query formulation

6. Sentence embeddings are generated for both the sets (from step 3 & 5) using BERT transformer model.
7. Semantic Similarity measures such as Cosine Similarity between embedding vectors are used for quantitatively identifying suitable predicates matching with respect to the query.
8. Based on the identified subject and matched predicate, parameterized SPARQL query is constructed and executed to fetch RDF object values concerned with the query.

Generating response for the query

9. Result set is processed in human readable format and query is answered.

3.2. *Sentence Embeddings and Semantic Similarity*

BERT model-based sentence transformers are used for generating the embedding for the sentence variations. The model generates the embedding in the form of a 768-element vector as shown below.

```
# Sample embedding for the phrase "was born on date" using "bert-base-cased" transformer
[[ 3.68885726e-01 -2.99447149e-01 7.45447651e-02 -2.24752367e-01
2.59226233e-01 4.63026375e-01 9.34034362e-02 5.90059273e-02
...
-2.47543395e-01 9.08255577e-02 2.54286140e-01 2.89659590e-01]]
```

Embedding for the whole sentence can be generated by averaging the embedding for individual participating tokens. Thus, by retaining key meaningful tokens and filtering out others, the sentence embeddings reflect a vector representation of the approximate meaning of the phrase. Cosine similarity is one of the pairwise semantic similarity metrics which is extensively used for computing the relatedness among the pair of embedding vectors. Calculation for cosine similarity for two embedding vectors U and V is given below (T represents the transpose of vector and $|x|$ is the absolute scalar value of the vector).

$$\text{cosine\_similarity}(\underset{u}{\rightarrow}, \underset{v}{\rightarrow}) = \frac{\underset{u}{\rightarrow} \cdot \underset{v}{\rightarrow} T}{|\underset{u}{\rightarrow}| \, |\underset{v}{\rightarrow}|}$$

3.3 *Parameterised SPARQL Querying*

Once the subject has been identified and predicate matching is done, KB is queried via its SPARQL endpoint to extract the respective RDF objects of interest. The response object could be another entity or the literal value such as a date, a number or a string for answering the question-based queries. Following is a sample SPARQL query for querying date of birth for a person of interest over YAGO Date Facts KB.

```
@base <http://yago-knowledge.org/resource/>
SELECT ?dob_obj
WHERE {
<:Roger_Federer> <:wasBornOnDate> ?dob_obj .
}
```

The resulting object fetched using above query needs to be further processed to provide human readable response to the query. Following is the raw response object, which may be further processed to generate "08 August 1981" as the formatted answer for the date of birth query.

```
"1981-08-08"^^<http://www.w3.org/2001/XMLSchema#date>
```

4. Implementation

This section details the implementation aspects along with generation of dataset using ChatGPT. Proposed system is implemented using Java and Python. YAGO Date Facts KB is used for analysing the accuracy. First subsection 4.1 discuss the implementation details. Setting up the graph KB, BERT model and semantic similarity computation is discussed within subsections. Subsection 4.2 details the generation of novel dataset for semantically similar sentence variations using ChatGPT.

4.1 *Implementation Setup*

4.1.1 YAGO Date Facts KB Setup using Jena Fuseki Server

YAGO Date Facts dataset which is available in RDF Turtle file format is converted to graph database structure and is loaded to Jena Fuseki Server. Jena programming API is used for sending queries to Jena Fuseki Server. To implement the logic for querying and retrieval programmatically, we are using Java (JDK 8) with Jena Libraries (method call QueryExecutionFactory.sparqlService() with SPARQL endpoint and the query). All the RDF predicates present in the KB are extracted using Java-

based logic by executing following SPARQL query through Fuseki Server web interface.

```
# SPARQL query for getting all unique predicates (relationships) present in dataset
PREFIX : <http://yago-knowledge.org/resource/>
SELECT DISTINCT ?pred WHERE { ?subj ?pred ?obj . }
```

4.1.2 BERT and Semantic Similarity Setup

Sentence Transformer "bert-base-cased" is used for generating the sentence embeddings. Logic for embedding generation is implemented in python using the sentence_transformers package. For computing the semantic similarity between embedding vectors, cosine similarity is used from the sklearn library.

Figure 2 depicts a sample code snapshot for illustration on BERT embedding and the computation of semantic similarity using cosine similarity metric. Two sample phrases are used which are semantically similar. Firstly, their embedding vectors are generated using the BERT model. Secondly, the pairwise cosine similarity is computed for embedding vector pairs.

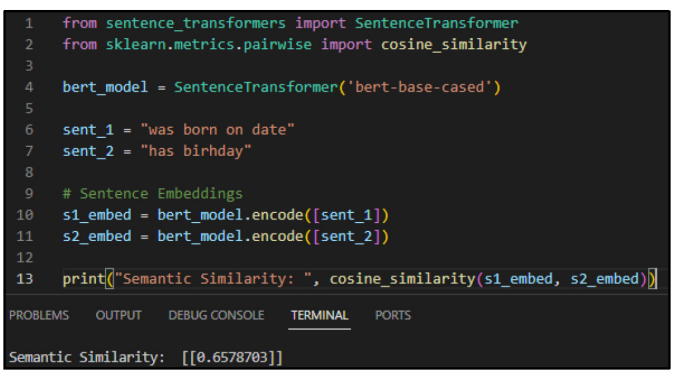

Figure 2

Illustration on semantic similarity computation using BERT embeddings and cosine similarity

4.2 *Generation of dataset of Semantically Similar Sentence Variations using ChatGPT*

ChatGPT has been used for generation of semantically similar query variations [**17**]. Starting with the prompt in Figure 3 asking ChatGPT to

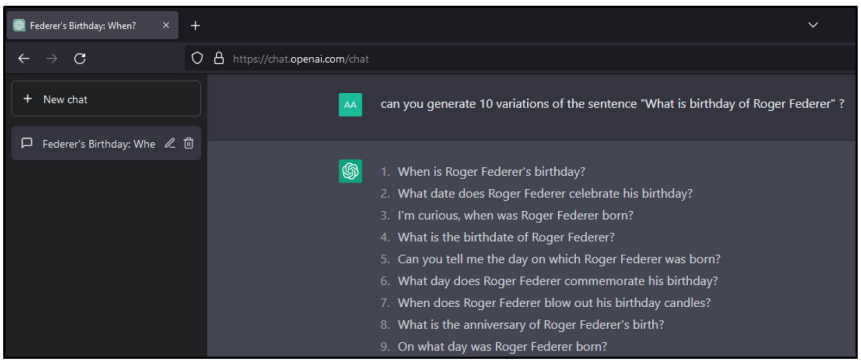

Figure 3

Prompting ChatGPT to generate semantically similar sentence variations for a given query

generate variations of the sentence "What is birthday of Roger Federer", a dataset of 110 semantically similar sentence variations was generated for evaluation of the proposed approach. The generated dataset was pre-processed and verified for syntactical correctness.

4. Results & Discussion

This section details the results for predicate matching and accuracy of proposed method compared to the matching accuracy of using sentence embeddings only for the query and predicates.

Analysis Approach:

- Load BERT Sentence Transformers for generating Sentence Embeddings.
- For All the sentence variations to be queried:
  - Subject is identified and excluded from processing.
  - Generate sentence embedding
- Extract all RDF predicates from the KB and generate their embeddings.
- If the RDF subject is identified:
  - Compute semantic similarity between embeddings of the KB Predicates and embeddings for the query tokens embeddings.
  - Select the RDF predicate with maximum similarity score.

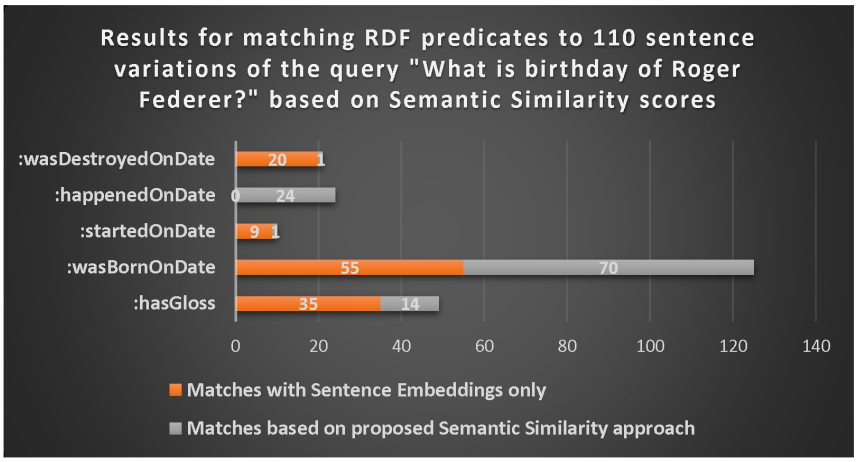

Figure 4

Results based on semantic similarity scores

- If the selected predicate matches with the known correct RDF predicate:
  - Execute the SPARQL query (with RDF subject and predicate parameters) to fetch the response object which will be formatted and returned as the answer from KB.
  - Increment the correct match counter.

Result metrics for RDF predicate matching for the 110 semantically similar queries are presented in Figure 4. Along the Y-axis are RDF predicates from YAGO Date Facts KB. X-axis has the frequency of predicate matched.

For queries related to the birthdate of Roger Federer, the known correct predicate URI is <http://yago-knowledge.org/resource/wasBornOnDate>. Thus, the accuracy of correct match with proposed approach has increased to 63.64% from 50% which was in the case of matching with sentence embeddings only.

6. Conclusion & Future Scope

Identification of the subject and predicate in a natural query and its precise matching to a RDF predicate in Knowledge Base is a key aspect for semantic search systems. Towards this direction, a semantic search approach is proposed and implemented utilizing the sentence embeddings.

The RDF predicate along with the subject is then queried on the KB for extracting the answer using SPARQL. For evaluation, a novel dataset for semantically similar question variations is generated using ChatGPT for analysis. Testing results over YAGO Date Facts KB demonstrates the effectiveness of the proposed approach with 13% increase in matching accuracy as compared to matching with sentence embeddings only.

As part of the future work, proposed system may benefit by incorporation of relation extraction subsystems for improving RDF predicate matching accuracy. Also, comparative analysis among part of speech combinations will be conducted to retain the best set of query tokens for predicate matching.

Acknowledgements

This publication is an outcome of research supported by Visvesvaraya Ph.D. Scheme, MeitY, Govt. of India. VISPHD-MEITY-2800.

References

[1] V. Yadav and S. Bethard, "A Survey on Recent Advances in Named Entity Recognition from Deep Learning models," arXiv preprint arXiv:1910.11470 (2019).

[2] C. Antoniou and N. Bassiliades, "A survey on semantic question answering systems," *The Knowledge Engineering Review*, 37, pp. 1-42 (2022).

[3] Z. Ge, Y. Wang, H. Yan, and X. Xu, "A Learning-Based Semantic Approximate Query over RDF Knowledge Graph," In *2018 6th International Conference on Advanced Cloud and Big Data (CBD)*, IEEE, pp. 135-141 (2018).

[4] M. Chhikara, and S.K. Malik, "A recommendation system for online social semantic network using knowledge based, content based and collaborative filtering," *Journal of Information and Optimization Sciences*, 44(4), pp. 795-806 (2023).

[5] H. Buzaaba and T. Amagasa, "Question answering over knowledge base: a scheme for integrating subject and the identified relation to answer simple questions," *SN Computer Science*, 2(1), pp 1-13 (2021).

[6] A. Pereira, A. Trifan, R. Lopes and J. Oliveira, "Systematic review of question answering over knowledge bases," IET Software, 16(1), pp. 1-13 (2022).

[7] N. Karim, K. Latif, N. Ahmed, M. Fatima, and A. Mumtaz, "Mapping natural language questions to SPARQL queries for job search,"

In *2013 IEEE Seventh International Conference on Semantic Computing*, IEEE, pp. 150-153 September (2013).

[8] A. Fatima, C. Luca, and M. Hobbs, "Free-text user queries for semantic search," In 2015 IEEE 13th International Conference on Industrial Informatics (INDIN), IEEE, pp. 838-843, July (2015).

[9] R. Cocco, M. Atzori, and C. Zaniolo, "Machine learning of SPARQL templates for question answering over linked spending," In 2019 IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), IEEE, pp. 156-161, June (2019).

[10] P. Gayathri, and V. V. Rajendran, "Semantic search on summarized RDF triples," In 2017 International Conference on Intelligent Computing and Control (I2C2), IEEE, pp. 1-6, June (2017).

[11] K. Rafes, S. Abiteboul, S. Cohen-Boulakia, and B. Rance, "Designing scientific SPARQL queries using autocompletion by snippets," In 2018 IEEE 14th International Conference on e-Science (e-Science), IEEE, pp. 234-244, October (2018).

[12] S. Campinas, T. E. Perry, D. Ceccarelli, R. Delbru, and G. Tummarello, "Introducing RDF graph summary with application to assisted SPARQL formulation," In 2012 23rd International Workshop on Database and Expert Systems Applications, IEEE, pp. 261-266, September (2012).

[13] R. A. Kadir, A. R. Yauri, and A. Azman, "Automated Semantic Query Formulation for Document Retrieval," In 4th International Conference on Information Retrieval & Knowledge Management, IEEE, pp. 1-8 (2018).

[14] A. Ali, and O. Qayyum, "Inference New Knowledge Using Sparql Construct Query," In 2nd International Conference on Computing, Mathematics & Engineering Technologies (iCoMET), IEEE, pp. 1-4 (2019).

[15] A. A. Khan and S. K. Malik, "Knowledge Base Entity Lookup using Named Entity Recognition: a case study on YAGO", 2022 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), Greater Noida, India, IEEE, pp. 429-434 (2022).

[16] S. Shelake and V. Honmane, "Entity Recognition by Natural Language Processing and Machine Learning," *International Research Journal of Engineering and Technology,* vol. 7, issue 7, pp. 1990-1994 (2020).

[17] OpenAI. (2023). ChatGPT (Feb 13 version) [Large language model]. https://chat.openai.com/chat

Received December, 2023