A Question Answering Framework Based on Hybridization of Deep Learning and Semantic Web Techniques

*[1]Sikirat K. Aina, [2]Afolayan A. Obiniyi, and [2]Donfack A. F. Kana
[1]Department of Computer Science, Federal University Gashua, Nigeria
[2]Department of Computer Science, Ahmadu Bello University, Zaria, Nigeria
{kennystaina | aaobiniyi | donfackkana}@gmail.com

Received: 26-MAR-2022; Reviewed: 27-APR-2022; Accepted: 12-MAY-2022
https://doi.org/10.46792/fuoyejet.v7i2.827

ORIGINAL RESEARCH ARTICLE

Abstract- The question answering (QA) system has been existing for several years. QA systems are divided into different processes such as question processing, document processing, paragraph extraction, answer extraction, question analysis, phrase mapping, disambiguation, query construction, querying the Knowledge Base (KB), and result ordering on user response respectively. Based on these processes, many models have been developed using approaches ranging from linguistic, statistical, and pattern matching. Popular models are Feedback, Refinement and Extended VocabularY Aggregation (FREyA), PowerAqua, SemSek, Semantic Interpretation of User Queries for QA on Interlinked Data (SINA), DEep Answers for maNy Naturally Asked questions (DEANNA), gAnswer, SemGraph, OKBQA (Open Knowledge Base and Question Answering) and Semantic Question Answering (SQA), for performance evaluation, these mostly focus on higher precision, recall, and/or F-measure. However, most of these models are constrained in the following operations: the combination of knowledge bases from different sources, formalism for knowledge representation to support interoperability, optimization of query construction and generation, ranking of responses, and support for set operations (union, sorting, comparison, and aggregation on user query) during query generation. This research proposes a hybrid of recurrent neural network and semantic-web-based question answering as the (RNNSQA) framework that combines heterogeneous knowledge sources and improves on state-of-the-art query generation mechanisms to allow for integration of comprehensive question-type operations, and set-based operations listed above. First, there would be a combination of techniques that is Natural Language Processing (NLP) and Recurrent Neural Network (RNN) techniques in classifying questions into types. Secondly, the semantic web technique is then employed in generating (SPARQL Protocol and RDF (Resource Description Framework) Query Language) SPARQL-based candidate queries. The result of this study is an enhanced QA framework with improved query translation and construction capability.

Keywords- Complex question, Knowledge base, Natural Language Processing, Question Answering System, SPARQL.

1 INTRODUCTION

The world analysis result reported by Datareportal, (2019), as shown that 5.11 billion are mobile users and 4.39 billion among them have access to the internet which means over half of the world's population has mobile phones with internet. There is a lot of information online related to various fields such as Technology, Education, Health, e-commerce, and industries. The more user accessibility to the internet the more user seeking for information in form of questions on their interest area and hope to get desirable answers.

QA to the user, is to retrieve straightforward relevant information as an answer to their presented question in natural language (Antoniou and Bassiliades, 2022). Question answering systems are categorized into two types, which are closed and open domain system, QA system are sub divided into three parts, they are Question Processing, Document Processing and Answer Processing (Allam and Haggag, 2012; Derici et al., 2015). Researchers have used several approaches in an attempt to provide a better QA system that will give users a concise but detailed response to the natural language questions the users seek to know about (Abujabal et al., 2017).

*Corresponding Author

Section B- ELECTRICAL/ COMPUTER ENGINEERING & RELATED SCIENCES
Can be cited as:
Aina S.K., Obiniyi A.A., and Kana D.A.F. (2022): A Question Answering Framework Based on Hybridization of Deep Learning and Semantic Web Techniques, *FUOYE Journal of Engineering and Technology* (FUOYEJET), 7(2), 126-132. http://doi.org/10.46792/fuoyejet.v7i2.827

Several models of QA system has been developed relying on different tools ranging from natural language (Xu, et al., 2014), neutral networks, deep learning (Zhang et al., 2016), ontology and semantic web (Vargas-Vera et al., 2003), some of these models are AquaLog (Lopez et al., 2007); Natural Language Interfaces for Databases (NLIDB) (Lopez et al., 2011); SemGraph (Beaumont et al., 2015); gAnswer (Zou et al., 2014); AQUA and SINA (Shekarpour et al., 2015) are in support of the pipeline of QA system. However, review from researches have also shown that some of these systems pipeline are not reusable, have problem of combination of knowledge bases from difference sources (Lopez et al., 2011), issues with formalism of knowledge representation to support interoperability (Pundge et al., 2016; Deifenbach et al., 2017), optimization of query construction and generation, ranking of responses (Pundge et al., 2016) and support for set operations (union, sorting, comparison and aggregation on user query) during query generation (Pundge et al., 2016; Zafar et al., 2018; Diefenbach et al., 2018; Abdelkawi et al., 2019; Ding et al., 2019). In this line, formal query generation has been seen as an approach to strengthen the QA system such that complex questions are well supported in Knowledge Based over Question Answering (KBQA). Many QA systems have seen the importance of formal query generation in answering their complex questions (Cui et al., 2017; Chen et al., 2020; Luo et al., 2018) which use semantic parser to form the SPARQL in generating queries to enhance user's natural language questions over Knowledge base such as Freebase (Bollacker et al., 2008), and DBpedia (Auer et al., 2007).

The QA system needs to have the ability to source information simultaneously from different knowledge bases and combine the information to respond to users' questions so that it makes usefulness out of the billions of webs of data available (McCrae et al., 2018). To achieve this, support for interoperability is essential because most of these knowledge bases (KB) operate in different data formats and modelling languages. In this work, we are looking at the option of using ontology to align the knowledge bases, to have them in a robust single KB.

Complex questions are complex due to their semantic structure. An example is "Who was the president when Boko haram started in Nigeria?" to answer the given example one needs to know "when did the Boko haram start?": When (?DATE) and "who was the president?": Who (?PERSON), and some complex questions could be difficult to answer due to the requirement of quantitative analysis over the answer space. This study proposes to present an enhanced hybridized model to enhance the current research on QA systems by largely focusing on formal query generation. The following are the contributions of this paper:

i. Leverage NLP technique to tokenize, repair, reconstruct and classify questions into a type of known/defined question-types

ii. A RNN model is then applied to predict the category of the classified question.

iii. Design an improved query generation mechanism to allow for integration of comprehensive question-type and set operations.

iv. Also aim at using semantic web technique in generating SPARQL-based candidate queries.

The remaining part of this study is arranged as follows. In session 2, related works were discussed, session 3 discussed the proposed model, session 4 question type and the proposed framework were discussed and lastly, the conclusions of the study were represented.

2 Related Works

Knowledge Base Question Answering (KBQA) has been among the latest study people are working on because they believe that it improves human/machine relation and help human have access to abundant information easily. Currently, studies into KBQA have shown that the formal query generation is either using semantic parser (Xu *et al*., 2014; Berant and Liang, 2014; Xu *et al*., 2015; Yih *et al*., 2015; Reddy *et al*., 2016) or template decomposition of query, (Unger *et al*., 2012; Abujabal *et al*., 2017; Cui *et al*., 2017; Zafar *et al*., 2018; Ding *et al*., 2019; (Abdelkawi *et al*., 2019) and (Liang *et al*., 2021) to support complex question (Lan et al., 2019).

Abujabal et al. (2017) proposed a method in which a complex question is made up of multiple simple sub-questions, each of which has only one relation that is mapped to a predicate in the knowledge graph. They were able to obtain the dependency representation by first using the existing syntactic parser and then performing simple automated rewriting to obtain two separate interrogative propositions following some predefined rules, and each sub-question is answered separately. Because it is heavily reliant on the syntactic parser and manually defined rewriting rules, the approach has limited ability to tackle complex problems. KBQA decomposes the question q into a succession of binary factoid questions (BFQ) q1..., qk and then answers each BFQ serially in Cui et al. (2017) rather than relying on the dependency representation provided by syntactic parsers (Abujabal et al. 2017; Zheng et al. 2018). Even though the study was too difficult to capture the essential examples, in the end, it lacks a distinct entity.

In the work of Diefenbach *et al*. (2018) their QA system supported five knowledge bases (Wikidata, DBpedia, MusicBrainz, DBLP, and Freebase) and was multilingual (English, German, French, Italian, and Spanish), the system was evaluated using the Question Answering over Linked Data (QALD) benchmarks. The work has no consideration for response time when querying multiple KB, not accessible to more than five KB, and there is no support for aggregation and function during query.

Zafar *et al*. (2018) focused on query generation using a subgraph to cover multiple candidates walks related to SPARQL query and the subgraph algorithm was represented; Tree-structured Long Short-term Memory (Tree LSTM) was used to rank for better accuracy. Support Vector Machine (SVM) and Naive Bayes models were trained to classify the questions into different types such as Boolean, count, and list depending on their Term Frequency-Inverse Dense Frequency (TFIDF) representation, the benchmark used was the Large-scale Complex Question Answering Dataset (LC-QuAD) dataset and the system outperform the baseline system. The work mainly supported simple and compound questions.

Ding *et al*. (2019) proposed Sub Query Generation (SubQG), their method was built on frequently used query substructures in both online and offline modes and Bidirectional LTSM was used for predicting subquery structure, then all the sub-queries relating to question are ranked either for existing query structure or merge query substructure and lastly, the output query was generated. Their work focuses on questions type that has to do with COUNT. For the implementation and evaluation, the system uses LC-QuAD and QALD-5 benchmark datasets and the time taken for the training was 1102s and 272s respectively and the system outperform the previous work. There was no support for question type involving UNION, GROUP BY, or numerical comparison.

Abdelkawi *et al*. (2019) proposed a query generation component, the work is built on SQG the work of Zafar *et al*. (2018) where the SPARQL for query generation was more elaborate. The work proposed Extended SQG (ExSQG) where it modified the existing question classifier and added an augmentation component in the new model. The job of the augmentation component in the new model was to complete the SPARQL queries chosen by the ranking model by including the necessary restrictions and parameters to construct the final query that corresponded to the input question. The system was evaluated using QALD 4 – 7 for Ordinal question type even though for QALD 5&7 did not yield a better

http://doi.org/10.46792/fuoyejet.v7i2.827 http://journal.engineering.fuoye.edu.ng/

performance due to misclassification or incorrect query generation while QALD 4&6 for Filter question type did not have a performance like Ordinal because there are much fewer questions of the type Filter that were supported in the datasets. There is no support for aggregation, union and sorting during query in the work.

Liang *et al*. (2021) proposed a QA system which translated NL question to SPARQL query, the translation process was subdivided into 5 steps namely, question analysis, question type classification, phrase mapping, query generation and query ranking. The work extended the query generation algorithm of Zafar *et al*. (2018) to add more complex queries, the question type classification phase used a random forest classifier to perform the classification, and then training occurred during question type classification and at the query ranking stage. Tree-LSTM was used to sort the candidate queries in the query ranking phase. QALD 7 and LC-QuAD were used to evaluate the system, it showed an outstanding performance compared with state of art systems but the system only focuses on three question types (Boolean, Count and List), while FILTER, LIMIT, ORDER, MIN, MAX, UNION are not considered and the system was built to source data from a single knowledge graph meanwhile a complex question could require sourcing information from multiple knowledge graphs.

3 Proposed QA Model

In this section, there would be a presentation and discussion about the question types, query generator model, and a comprehensive QA framework.

3.1 Preliminary

The Knowledge graph, Knowledge bases and the query generation as defined by Zafar *et al*. (2018) are modified below:

Knowledge graph $K = (E, R, T)$
Where E is define as a set of Entities,
Consist of a relation R as a set of relation labels
And T is defined as a set of ordered triples,
mathematically represented as follow: $T \subseteq V \times R \times V$

Definition 1 Walk: A Knowledge graph Walk $K = (E, R, T)$ is a series of edges that connects the nodes: $W = (e_0, r_0, e_1, r_1, e_2, \ldots e_{k-1}, r_{k-1}, e_k)$ with $(e_i, r_i, e_i + 1) \in T$ for $0 \leq i \leq k - 1$.

Definition 2 (Valid Work). If and only if a walk W contains all set of entities E' and relations R', that is: $\forall e \in E' : e \in W$ and $\forall r \in R' : r \in W$, it is valid.

A node e with $e \in W$ and $e \in E'$ is unbounded. An unbounded node is an abstract node that is used to connect the other nodes in the walk. Meanwhile, examples of established question types will be shown later such as Boolean, Count, Sorting, Comparison, Union and Aggregation. A number of valid walks are extracted from the KG depending on the types of questions, but the majority of them may be an incorrect mapping of the input question, in which case it will not capture the correct intention behind the question. The candidate walks are then sorted according to their similarity to the input question.

3.2 Knowledge-Based (KB) and Knowledge Graph

The Knowledge base (KB) consists of many KBs which are combined to form a single KB by applying similarities calculation in ontology alignment for different source Knowledge bases. The knowledge graph works to interlinked descriptions of entities; the entities are defined in section 3.1. A foundation for data integration and unification is provided by the knowledge graph, which adds data in context via linking and semantic information.

3.3 Query Generation

As discussed in section 3.1 and detailed in Zafar *et al*. (2019) query generation is constructed in triples and since the SPARQL query is in a graph pattern where variables may be denoted in the form of subject, predicate and object, the triple output would be generated as shown in the example questions below.

An example question "Who are the children of Olusegun Obasanjo whom was the former president of Nigeria" the triple pattern would be shown as <dbr: children dbp: Olusegun Obasanjo ?child > and in the question "Is Lagos a city in Nigeria" the triple pattern would be shown as < dbr:Nigeria dbo:city dbr:Lagos ; rdf:type dbo:Place >.

To locate candidate walks in the KG from KB_1 to KB_n, one must begin with a linked entity $e \in E$ and traverse the KG. It will take a long time to list all of the valid walks.

Algorithm 1: For the subgraph

Data: E', R', O_{KBs} $\emptyset$
Result: SPARQL – Queries
$K \leftarrow \emptyset$
$G \leftarrow \emptyset$
$align \leftarrow \emptyset$
$qtype \leftarrow train\ RNN\ (\emptyset)$

```
foreach o in O_KBs
    align ← alignment (o, align)
end
K ← align
Add ∀ e ∈ E' to G as notes;
for each e ∈ E', r ∈ R' do
    if (e, r, ?) ∈ K then
        add (e, r, ?) to G;
    else if (?, r, e) ∈ K then
        add (?, r, e) to G;
end
```


http://doi.org/10.46792/fuoyejet.v7i2.827 http://journal.engineering.fuoye.edu.ng/

```
foreach (e2, r, e2) ∈ G do
    foreach r' ∈ R', r' ≠ r do
        if (e2, r', ?) ∈ K then
            add (e2, r', ?) to G;
        else if (?, r', e2) ∈ K then
            add (?, r', e2) to G;
        else if (e1, r', ?) ∈ K then
            add (e1, r', ?) to G;
        else if (?, r', e1) ∈ K then
            add (?, r', e1) to G;
    end
end
```

```
foreach g in G
    if (qtype = boolean)
        SPARQL − queries ← format (q, boolean)
    else if (qtype = list)
        SPARQL − queries ← format (q, list)
    else if (qtype = count)
        SPARQL − queries ← format (q, count)
    else if (qtype = sorting)
        SPARQL − queries ← format (q, sorting)
    else if (qtype = union)
        SPARQL − queries ← format (q, union)
    else if (qtype = comparison)
        SPARQL − queries ← format (q, comparison)
    else if (qtype = aggregation)
        SPARQL − queries ← format (q, aggregation)
end
return SPARQL − queries;
```

4 QUESTION / QUERY TYPES

This work supported simple, compound, and complex question types such as List, Boolean, Count, Comparison, Sorting, Union, and Aggregation.

The question types listed in Table 1 are classified into their categories using an improved question classifier proposed in this study. Figure 1 shows the question classifier. The approach applied to this task is using a recurrent neural network (RNN) for the prediction of the category of a question asked in natural language by the user. This classification's outcome will be presented to the SPARQL-based query generation module.

Table 1. List of Question types, sample and their Queries form

Category/Type	Question Explanation /Sample/ Query form
List	This question is called factoid and it could be in a single or multiple association. An example is "Who are the Children of Chief Olusegun Obasanjo whom was the formal president of Nigeria?" SELECT ?child where{ dbr:Olusegun Obasanjo dbp:children ?child }
Boolean	The Boolean query is a direct question which requires a yes or no as an answer for the query or question presented. An example "Is Lagos a city in Nigeria?" SELECT WHERE {dbr:Nigeria dbo:city dbr:Lagos ; rdf:type dbo:Place}
Count	The purpose of this question or query is to count the number of all the likely outcomes as an answer from the presented queries. An example "How many towns are in Nigeria?" SELECT COUNT(?city) WHERE{?city dbo:country dbr:Nigeria ; rdf:type dbo:Town}
Comparison	This query or question compares two or more variables and its outcomes requires conditional clause in items of measures. An example "List all cities with more than five hundred thousand population in Nigeria?" SELECT ?city WHERE{?city dbo:country dbr:Nigeria ; dbo:populationTotal ?p;rdf:type dbo:City. FILTER (?p> 500000)}
Sorting	The purpose of the question is to order the results over certain c. An example "Which is tis he mosonditionst populous University in Nigeria?" SELECT ?university WHERE{?university dbo:country dbr:Nigeria ; dbo:populationTotal ?population ; rdf:type dbo:City} ORDER BY DESC(?population) LIMIT1
Union	The purpose of the question is to count the number and arrange the result orderly. An example "What is the population of Oyo, Kano and Lagos State?" SELECT COUNT (?Population) List Where (?State)
Aggregation	The aggregate functions are count, sum, min, max, rank, threshold etc. when more than one of the stated functions occurred in a question/query, it is said to be aggregate and also classified as a complex question. An example "What is the population of Nigeria at the end of General Ibrahim Babandiga's tenure?" SELECT COUNT (?Population) When (?Date)

http://doi.org/10.46792/fuoyejet.v7i2.827 http://journal.engineering.fuoye.edu.ng/

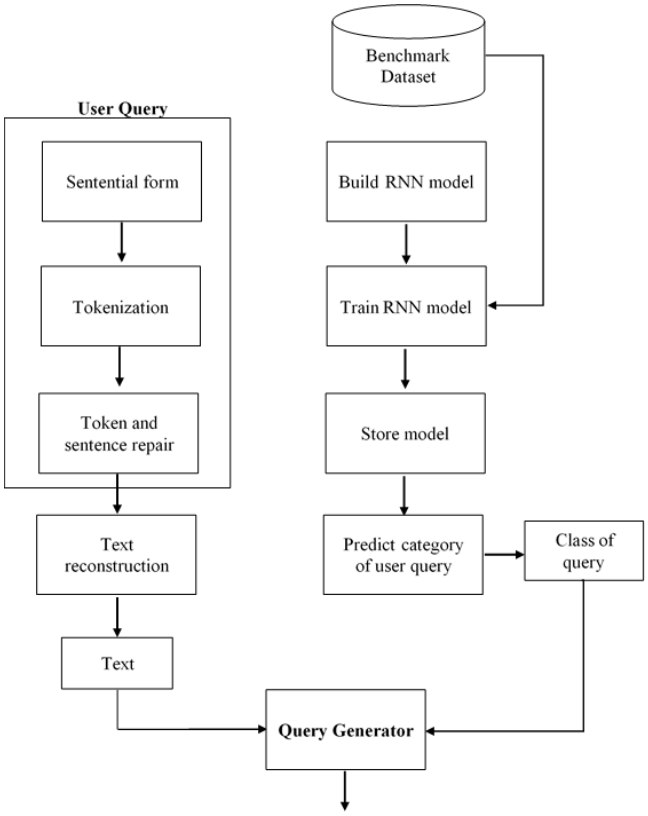

Fig. 1: Architecture of the Question Classifier

The components of the Question classifier processes are described as follows.

User query/question: it is a formulation of a user's utterance.

Sentential form: the query could either be in form of a single keyword or in combinations of many words (phrase) which is usually processed in form of a document.

Tokenization: here the strings from the user query would be breaking into pieces and some characters such as punctuation marks would be removed.

Token and sentence repair: here it repairs an incomplete/ incorrect sentence/spelling error and unknown word.

Text reconstruction: here the incomplete sentence or words are reconstructed

Text: here it displays the complete and corrected sentence for processing

Benchmark Dataset: here the dataset for this research would be uploaded.

Build RNN model: here RNN model would build using python and the other necessary libraries would be imported.

Train RNN model: here RNN model trains some percentage of the chosen dataset.

Store model: here the build model and trained model would all be stored.

Predict of a category of user query: here the prediction took place.

Class of query: here the predicted user query would be displayed.

4.1 SPARQL Query Generator (SQG)

The query generating module is a SPARQL-based mechanism that allows for automating the building of queries in the SPARQL language. The proposed QA system acquires its knowledge from multiple sources to boost its hit-value and accuracy though at the cost of computational time. Users' input is passed into the system in a natural language pattern and then parsed using the Parse Tree component of the architecture. Before the SPARQL-based queries are generated, the RNN question classifier is applied to the parsed questions to classify the type(s) of questions inherent in the user's question. Once the question type(s) are established, queries are generated.

The auto-generated queries are ordered according to their relevance to the user's natural language-based queries. This is to allow for the display of the outcome of the queries according to their relevance. This mechanism was so proposed to pattern after Google's search engine which also takes the relevance of searched results in cognizance with the query of the user. The ordered queries are then augmented with important meta-data and operators to improve and optimize performance on the query engine, and as well enhance the outcome received by the user. We contend that the question classifier has a significant influence in determining what the user sees. This determines the depth and semantic richness of what is presented to the user about his/her request – search query. Finally, the ordered queries are passed on to the query engine for processing and then output the result to the user. Figure 2 describes the approach adopted for query generation.

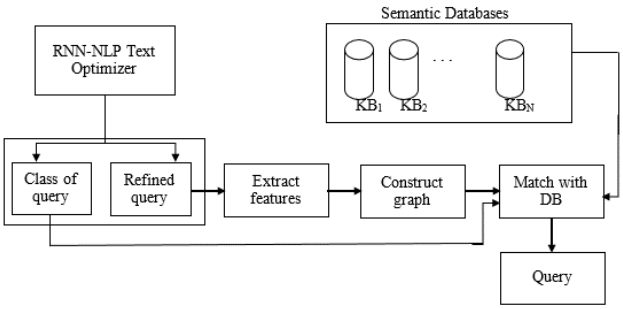

Fig. 2: Query Generating Approach

The query generation would be achieved by implementing the following processes, they are RNN-NLP test optimizer, Class of query, Refined query, Extract features, Construct graph, and then, the DB would be Matched to generate the query.

4.2 The QA Framework

The proposed framework of question answering system displayed in figure 3 which have several components ranging from the Knowledge base (where multiple knowledge bases are coupled together as briefly explained in section 3), question classifier (discussed in section 4.1), query generation (discussed in section 4.1), D Parse Tree (described in section 4.3) and finally the ranking model (discussed in section 4.4).

http://doi.org/10.46792/fuoyejet.v7i2.827 http://journal.engineering.fuoye.edu.ng/

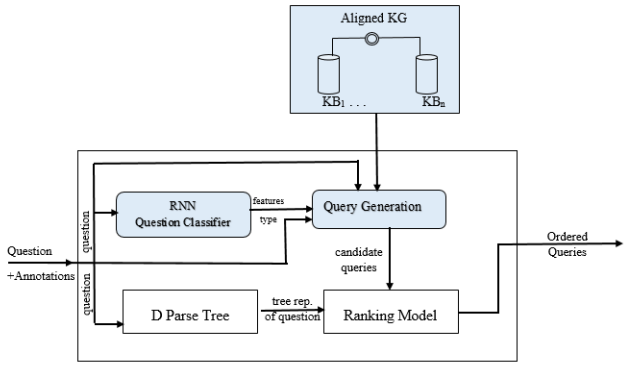

Fig. 3: Proposed Framework

4.3 D Parse Tree

Here it would recognize and analyse the grammatical structure of any sentence provided to the model as it was done (Zafar et al., 2019). Such as "which is the most populous University in Nigeria" distinguishing the keywords from the sentence, "most populous University in Nigeria"

4.4 Ranking Model

In this work, we propose using Tree LTSMs and LTSM for the ranking so that we would be able to find the similarity score of the candidate queries using the similarity function as it was done (Zafar et al., 2019). Tree LTSMs are good at integrating data from child nodes, whereas LTSMs can only allow sequential propagation.

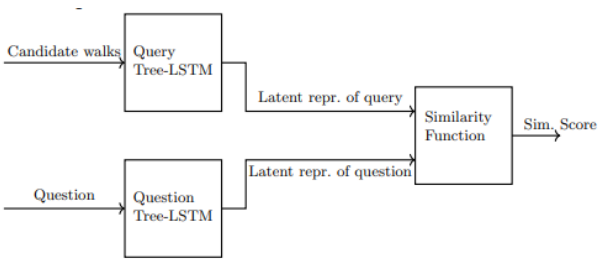

Fig. 4: Ranking Model Architecture (Zafar et al., 2019)

5 Conclusion

An improved question answering system based on Semantic Web technologies is proposed in this paper. The approach adopted in this study is the use of SPARQL-generated queries to answer the questions of the users. The proposed framework described a query generation mechanism and its corresponding algorithm. The framework consists of four items component namely D Parse Tree, question type, SQG, and ranking of response, and this work focuses on query generation and question classification into types. In the next work, the implementation will be displayed along with the evaluation results.

Reference

Abdelkawi; A., Zafar, H., Maleshkova, M. and Lehmann, J. (2019). Complex Query Augmentation for Question Answering Over Knowledge Graphs. Pp 1 – 17.

Abujabal, A. and Roy, R. S. (2018). Never-Ending Learning for Open–Domain Question Answering over Knowledge Bases. Web Content Analysis, Semantics and Knowledge WWW 2018, April 23-27, 2018, Lyon, France. Pp 1053 – 1062.

Abujabal, A., Yahya, M., Riedewald, M. and Weikum, G. (2017). Automated template generation for question answering over knowledge graphs. In Proceedings of the 26th International Conference on World Wide Web, WWW 2017, pp 1191–1200.

Allam, A. M. N. and Haggag, M. H. (2012). "The Question Answering Systems: A Survey", International Journal of Research and Reviews in Information Sciences (IJRRIS), September 2012 Science Academy Publisher, United Kingdom.

Antoniou, C. and Bassiliades, N. (2022). A Survey on Semantic Question Answering Systems. Published online by Cambridge University Press: 2022. The Knowledge Engineering Review (2022), 37, e2, pp. 1 – 42.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z. (2007). Dbpedia: A nucleus for a web of open data. In: The semantic web, pp. 722–735. Springer (2007).

Beaumont, R., Grau, B. and Ligozat, A. L. (2015). SemGraphQA@QALD-5: LIMSI participation at QALD-5@CLEF. CLEF.

Berant, J. and Liang, P. (2014). Semantic parsing via paraphrasing. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1415–1425.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD international conference on Management of data. pp. 1247–1250.

Chen, Y., Li, H., Hua, Y. and Qi, G. (2020). Formal Query Building with Query Structure Prediction for Complex Question Answering over Knowledge Base. Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20) pp. 3751 – 3758.

Cui, W., Xiao, Y., Wang, H., Song, Y., Hwang, S.W. and Wang, W. (2017). KBQA: learning question answering over QA corpora and knowledge bases. Proceedings of the VLDB Endowment, 10(5):565–576.

Derici, C., Çelik, K., Kutbay, E. & Aydın, Y., Gungor, T., Ozgur, A. and Kartal, G. (2015). Question Analysis for a Closed Domain Question Answering System. 10.1007/978-3-319-18117-2_35.

Diefenbach, D., Both, A., Singh, K. and Maret, P. (2018). Towards a Question Answering System over the Semantic Web. Semantic Web 0 (0) 1 IOS Press Pp 1 – 15.

Diefenbach, D., Lopez, V., Singh, K. and Maret, P. (2017). Core Techniques of Question Answering Systems over Knowledge Bases: a Survey. Knowledge and Information Systems (KAIS), Springer, pp. 1 – 42.

Digital, (2019). Global Digital Overview: https://datareportal.com/reports/digital-2019-global-digital-overview.

Ding, J., Hu, W., Xu, Q. and Qu, Y. (2019). Leveraging frequent query substructures to generate formal queries for complex question answering. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, pages 2614–2622. Association for Computational Linguistics, 2019.

Lan, Y., Wang, S. and Jiang, J. (2019). Knowledge Base Question Answering with Topic Units. Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19). Pp 5046 – 5052.

Liang, S., Stockinger, K., Mendes de Farias, T., Anisimova, M. and Gil, M. (2021). Querying knowledge graphs in natural language. Journal of Big Data, 8:3 https://doi.org/10.1186/s40537-020-00383-w.

Lopez, V. (2011). PowerAqua: Open Question Answering on the

Semantic Web. Ph. D Thesis Submitted to Semantic Web and Knowledge Services, the Knowledge Media Institute the Open University England.

Lopez, V., Motta, E., Sabou, M. and Fernandez, M. (2007). PowerAqua: A Multi-Ontology Based Question Answering System–v1. Open Knowledge Deliverable D8.4.

Luo, K., Lin, F., Luo, X. and Zhu, K. Q. (2018). Knowledge base question answering via encoding of complex query graphs. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018, pages 2185–2194.

McCrae, J. P., Abele, A., Buitelaar, P., Cyganiak, R., Jentzsch, A., and Andryushechkin, V. (2018). The Linked Open Data Cloud, 2018. URLhttp://lod-cloud.net/.

Pundge, A. M.; Khillare S. A. and Mahender C.N., (2016). Question Answering System, Approaches and Techniques: A Review. International Journal of Computer Applications (0975 – 8887) Volume 141 – No.3, May 2016.

Reddy, S., Tackstrom, O., Collins, M., Kwiatkowski, T., Das, D., Steedman, M. and Lapata, M. (2016). Transforming dependency structures to logical forms for semantic parsing. TACL, 4:127–140.

Shekarpour, S., Marx, E., Ngomo, A. C. N. and Auer, S. (2015). Sina: Semantic interpretation of user queries for question answering on interlinked data. Web Semantics: Science, Services and Agents on the World Wide Web 30 (2015).

Unger, C., Bühmann, L., Lehmann, J., Ngomo, A.-C. N., Gerber, D. and Cimiano, P. (2012). "Template-based question answering over RDF data," in Proceedings of the 21st international conference on World Wide Web, 2012, pp. 639–648.

Vargas-Vera, M., Motta, E. and Domingue, J. (2003). 'AQUA: An ontology-driven question answering system', AAAI Spring Symposium, New Directions in Question Answering, Stanford University, 24–26 March.

Xu, K., Feng, Y. and Zhao, D. (2014). Answering natural language questions via phrasal semantic parsing. In Working Notes for CLEF, pages 1260–1274.

Xu, K., Feng, Y., Huang, S. and Zhao, D. (2015). Question answering via phrasal semantic parsing. In Yih et al., 2015 Experimental IR Meets Multilinguality, Multimodality, and Interaction - 6th International Conference of the CLEF Association, pages 414–426.

Zafar, H., Napolitano, G. and Lehmann, J. (2018). Formal query generation for question answering over knowledge bases. In Proceedings of the 15th Extended Semantic Web Conference, ESWC 2018, pages 714–728.

Zhang, Y., Liu, K., He, S. Ji, G., Liu, Z., Wu, H. and Zhao, J. (2016). Question Answering over Knowledge Base with Neural Attention Combining Global Knowledge Information. arXiv preprint arXiv:1606.00979 (2016).

Zheng, W., Yu, J. X., Zou, L. and Cheng, H. (2018). Question Answering Over Knowledge Graphs: Question Understanding Via Template Decomposition. Proceedings of the VLDB Endowment, Vol. 11, No. 11 Copyright 2018 VLDB Endowment 2150-8097/18/07. Pp 1373 – 1386.

Zou, L., Huang, R., Wang, H., Yu, J. X., He, W. and Zhao, D. (2014). Natural language question answering over RDF: a graph data driven approach. In Proceedings of the 2014 ACM SIGMOD international conference on Management of data.