



HAL
open science

PSYCHIC: A Neuro-Symbolic Framework for Knowledge Graph Question-Answering Grounding

Hanna Abi Akl

► **To cite this version:**

Hanna Abi Akl. PSYCHIC: A Neuro-Symbolic Framework for Knowledge Graph Question-Answering Grounding. ISWC 2023 - International semantic web conference, Juanzi Li; Manolis Koubarakis; Valentina Ivanova; Wen Zhang; Dimitris Plexousakis; Vassilis Christophides; Heng Ji; Haofen Wang; Andrea Nuzzolese; Evgeny Kharlamov; Zoi Kaoudi; Gong Cheng; Heiko Paulheim; Cogan Shimizu; Eleni Tsalapati; Ioannis Chrysakis; George Vouros; Ioannis Karatzanis; Lei Hou; Terry R. Payne; Valentina Presutti; Guilin Qi; María Poveda-Villalón; Bo Fu; Giorgos Stoilos; Laura Hollink; Jose Manuel Gomez-Perez; Daniel Garijo; Claudia d'Amato; Jeff Z. Pan; Iriini Fundulaki; Kozaki Kouji, Nov 2023, Athens (Greece), Greece. hal-04311370

HAL Id: hal-04311370

<https://inria.hal.science/hal-04311370>

Submitted on 28 Nov 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

PSYCHIC: A Neuro-Symbolic Framework for Knowledge Graph Question-Answering Grounding

Hanna Abi Akl^{1,2}

¹Data ScienceTech Institute (DSTI), 4 Rue de la Collégiale 75005 Paris, France

²Université Côte d'Azur, Inria, CNRS, I3S

Abstract

The Scholarly Question Answering over Linked Data (Scholarly QALD) ¹ at The International Semantic Web Conference (ISWC) 2023 challenge presents two sub-tasks to tackle question answering (QA) over knowledge graphs (KGs). We answer the KGQA over DBLP (DBLP-QuAD) task by proposing a neuro-symbolic (NS) framework based on PSYCHIC ², an extractive QA model capable of identifying the query and entities related to a KG question. Our system achieved a F1 score of 00.18% on question answering and came in third place for entity linking (EL) with a score of 71.00%.

Keywords

Question Answering, Knowledge Graphs, Neuro-Symbolic Artificial Intelligence, Entity Linking, Linked Data, Language Models

1. Introduction

Knowledge veracity has always been one of the key topics of the Web [1]. The ability to present users with a body of factual information to refer to is a challenge that continues to defy the growth of the web [1]. In particular, the plethora of available information is plagued by the non-structured nature of this knowledge [1].

One way the Semantic Web community addresses this issue is by constructing KGs which are structured repertoires of entities and relationships [1]. These graphs encapsulate factual knowledge and allow users to retrieve it by navigating their different connections [1]. The DBLP computer science bibliography is an example of such an effort that provides open bibliographic information on major computer science journals and proceedings [2].

Aside from consolidating bodies of information, KGs provide a platform for information retrieval [1]. In artificial intelligence (AI), question answering refers to the task of asking an AI agent a question and receiving an answer in return [3]. Large Language Models (LLMs) are popular agents that fill this role well due to their nature of ingesting huge amounts of

¹<https://kgqa.github.io/scholarly-QALD-challenge/2023/>

²<https://huggingface.co/HannaAbiAkl/psychic>

Scholarly QALD at ISWC, November 6-10, 2023, Athens, Greece

* Scholarly QALD at ISWC, November 6-10, 2023, Athens, Greece

*Corresponding author.

✉ hanna.abi-akl@dsti.institute (H. A. Akl)

ORCID [0000-0001-9829-7401](https://orcid.org/0000-0001-9829-7401) (H. A. Akl)

 © 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0)

 CEUR Workshop Proceedings (CEUR-WS.org)

information [4]. However, this also makes them prone to giving out misinformation based on their inability to disseminate correct from incorrect knowledge [4].

Integrating LLMs and KGs has emerged as a solution to mitigate the shortcomings of large language models [4]. By querying a factual base of information, LLMs gain a way to validate their responses before sending them back to users [4]. It is in this scope that the Scholarly QALD challenge presents its two sub-tasks, DBLP-QuAD and SciQA. We focus on DBLP-QuAD, a QA task over the DBLP¹ KG. We distinguish two parts in the challenge: question answering, which requires participants to propose systems capable of retrieving specific information from the KG to answer a question, and entity linking, which requires participants to retrieve the list of entities related to a question.

In this paper, we show how a NS system can handle the tasks of QA and EL over a KG. The rest of the paper is organized as follows. In section 2, we discuss some of the related work. In section 3, we present the experimental setup. In section 4, we discuss the results. Finally, we present our conclusions in section 5.

2. Related Work

This section reviews some of the proposed systems designed for QA and EL over KGs.

2.1. Question answering over knowledge graphs

Several techniques have emerged to tackle the problem of QA over KGs. Zheng and Zhang make use of structured query patterns which involve identifying query graph candidates in the KG using EL and disambiguation and transforming them to SPARQL queries [5] to answer questions. Pramanik et al. employs a similar approach by proposing a model that draws from relevant RDF triples to generate all possible query context graphs from which queries are created to answer natural language questions. Their findings result in an advancement in graph-based methods but prove their approach to be highly noisy [6]. Sima et al. improve on this approach by leveraging graph algorithms to identify and rank domain-specific query candidates based on the node centrality of the relevant entities [7]. Zheng et al. go a step further in this direction by integrating semantic parsers to their query templates to refine the query-generation process. By aligning both natural language questions and query templates, they prove they can effectively answer complex and detailed questions [8].

Nikas et al. combine graph-based techniques with neural-based methods to create a NS QA system [9]. In their work, they train a DistilBERT model based on an expected answer type to handle different kinds of questions [9]. The neural network also leverages SPARQL queries to gather facts for entity enrichment and boost its performance in extracting the correct answer from the provided context [9]. Cm et al. propose a similar pipeline by replacing the SPARQL endpoint with a supervised BERT model that performs relation extraction to add context information to the system [10].

Diomedi and Hogan propose a different approach by using neural machine translation to map the questions in natural language to SPARQL templates directly. They show that this

¹<https://blog.dblp.org/2022/03/02/dblp-in-rdf/>

method, coupled with tree-based entity disambiguation techniques, turns the QA problem to a slot-filling task and outperforms vanilla deep learning (DL) models [11]. Aghaei et al. leverage a similar slot-filling pipeline on a domain-specific KG to demonstrate that it can reliably learn the pattern structures of the domain queries [12].

In their work, Mavromatis and Karypis and Li et al. leverage graph networks to compute graph embeddings and compare them with the input question embeddings to target the correct answer entities [13, 14]. Dutt et al. utilize graph convolution networks to score questions based on similarity and derive their corresponding graph paths to answer new questions [15].

Saxena et al. and Zuo et al. demonstrate how using KG embeddings can help solve multi-hop questions [16, 17]. Rony et al. propose a system whereby KG embeddings are stored in a vector database for faster retrieval and improved performance in answering complex questions [18].

2.2. Entity linking over knowledge graphs

Shi et al. present a survey on the techniques seen in EL over KGs. They regroup approaches into rule-based, machine learning (ML) and DL EL methods [19]. In their work, Dubey et al. use rule-based methods by leveraging global traveling salesman approximate solver algorithms to disambiguate entities in large KGs [20]. Steinmetz employ abstract meaning representation, a series of text dependency parsing rules, to identify and group sentences having the same meaning but different structures [21]. Using this technique, they perform data augmentation to enrich entity representation and achieve better performance on the EL task [21]. In a similar fashion, Radhakrishnan et al. use co-occurrences from large corpora to enrich existing KGs with entity information and create dense KGs as a basis for EL [22].

Thawani et al. employ ML techniques by combining TF-IDF with Wikidata entries to generate feature vectors and score entity candidates accordingly [23]. Li et al. make use of translating embeddings to encode entity-entity relationships and combine them with entity-relation embeddings to get better performance over KGs [24].

From a DL perspective, Huang et al. show that using BERT to calculate sentence embeddings over entities outperforms rule-based entity enrichment approaches [25]. Banerjee et al. improve on this approach by concatenating entity embeddings with context using FastText embeddings in a pointer network entity linker structure designed to represent entities in a dense vector space and identify the correct one for any input entity [26].

Finally, Ding et al. propose a NS approach based on EL using SpaCy embeddings in a case-based reasoning by computing the cosine score between input entity embeddings and KG entity embeddings [27]. The final entity is computed automatically according to the best matching embedding or manually labeled based on a threshold cosine score [27]. Diomedi and Hogan experiment with a pipeline that integrates rule-based entity matching over DBpedia and Wikidata KGs and neural network models to associate the resulting entities to a fixed set from a designed KG [28].

3. Experiments

This section describes the framework for our experiments in terms of data, system and training process.

3.1. Dataset description

The dataset considered for this shared task is divided in two parts: the DBLP-QuAD² dataset which consists of 10000 question-SPARQL pairs and is answerable over the DBLP KG, and a dataset of 500 questions retaining the same format provided by the task organizers which will be referred to as seed data. We provide details for both datasets in the following subsections.

3.1.1. DBLP-QuAD data

DBLP-QuAD is a scholarly KGQA dataset with 10,000 question-SPARQL query pairs targeting the DBLP KG. DBLP-QuAD was created using the OVERNIGHT approach where logical forms are first generated from a KG. Canonical questions are then generated from these logical forms. The dataset is split into 7,000 training, 1,000 validation and 2,000 test questions. Each question-SPARQL pair consists of the following data fields: the id of the question (*id*), a string containing the question (*question*), a paraphrased version of the question (*paraphrased_question*), a SPARQL query that answers the question (*query*), the type of the query (*query_type*), the template of the query (*template_id*), a list of entities in the question (*entities*), a list of relations in the question (*relations*), a boolean indicating whether the question contains a temporal expression (*temporal*) and a boolean indicating whether the question is held out from the training set (*held_out*). Sample data is shown in Figure 1.

3.1.2. Seed data

Participants are provided with seed data to evaluate the performance of their systems on the QA and EL sub-tasks. This data is curated by the task organizers and consists of 500 random question-SPARQL pairs that comply with the same format as the DBLP-QuAD data. For the final evaluation, the organizers provided an additional set of 500 random questions containing only the question and its paraphrase. This dataset was used exclusively to score the performance of the participant systems in both sub-tasks.

3.2. System description

This section introduces our proposed system. It presents the system architecture and describes the training process in our experiments.

3.2.1. PSYCHIC model

Since the challenge presents itself as a QA task, we selected the DistilBERT-base uncased³ model to tackle it in an extractive QA setting. Extractive QA confines a model to selecting the appropriate answer chunk from a context given as input with the question. It also leverages some control over the model answers as opposed to generative QA which makes models generate the answer.

²<https://huggingface.co/datasets/awalesushil/DBLP-QuAD>

³<https://huggingface.co/distilbert-base-uncased>

```

{
  "id": "Q0001",
  "query_type": "SINGLE_FACT",
  "question": {
    "string": "Show the Wikidata ID of the person Robert Schober."
  },
  "paraphrased_question": {
    "string": "What is the Wikidata identifier of the author Robert S.?"
  },
  "query": {
    "sparql": "SELECT DISTINCT ?answer WHERE {
      <https://dblp.org/pid/95/2265> <https://dblp.org/rdf/schema#wikidata> ?answer
    }"
  },
  "template_id": "TC04",
  "entities": [
    "<https://dblp.org/pid/95/2265>"
  ],
  "relations": [
    "<https://dblp.org/rdf/schema#wikidata>"
  ],
  "temporal": false,
  "held_out": false
}

```

Figure 1: Example of DBLP-QuAD data

Training a QA model involves giving the model an input composed of the question to be answered and a context that should contain the answer. In the scope of this shared task, we targeted two types of answers: the SPARQL query which should directly answer the given question and the list of entities for EL. We constructed the context to include both pieces of information and added information to guide the model regarding the nature of the question asked and the expected SPARQL answer: the query type and the template id provided in the DBLP-QuAD data. We also introduced symbolic information through a symbolic rule engine that inserts the special tokens [CLS] at the start of the context string and [SEP] between the different pieces of information in the context. These markers were added to ground the output of the model and enable it to discriminate between the different pieces of context information. The aim is to help the model learn the types of query structures and entities it will be asked to retrieve.

On the output side, we fine-tuned our PSYCHIC (**P**re-trained **S**ymbolic **C**hecker **I**n **C**ontext) model to return both the SPARQL query and the list of entities. We constructed the output as a string containing both pieces of information separated by the [SEP] token. We also introduced another symbolic rule engine which is a programmatic function designed to split the output string based on the [SEP] symbol to return the query and entities chunks separately. The PSYCHIC model architecture is presented in Figure 2.

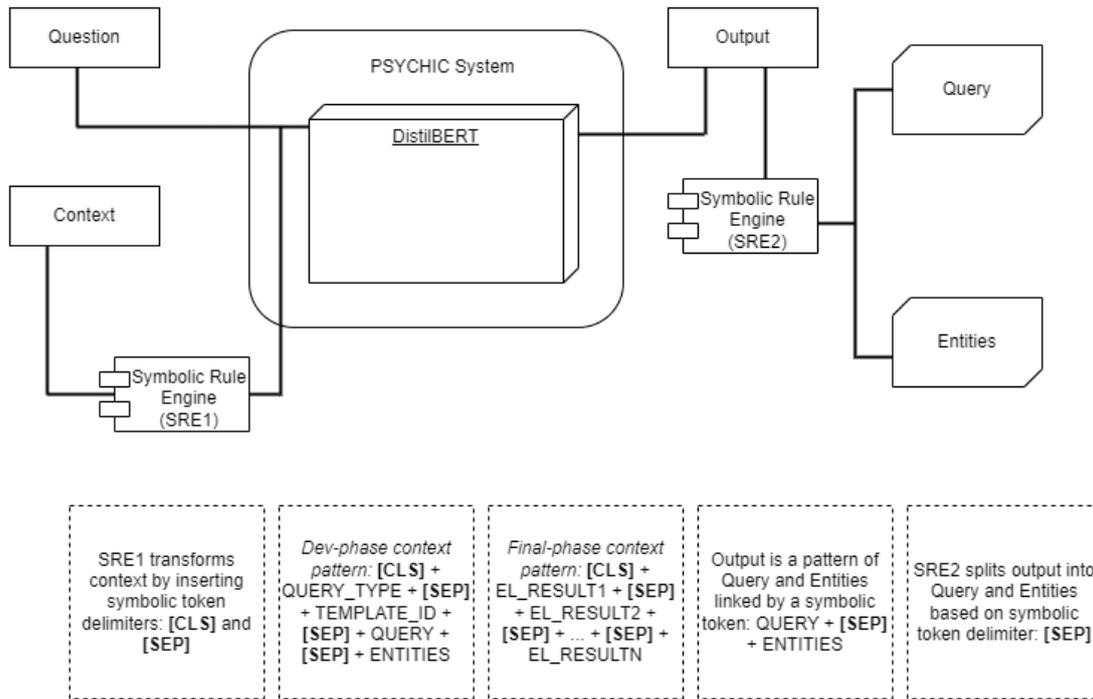


Figure 2: PSYCHIC model architecture

3.2.2. Pipeline architecture

To answer both parts of the shared task, we need to return a result from a SPARQL query for the QA challenge and an entity list for the EL challenge. In that respect, the PSYCHIC model alone is not enough. The QA challenge defines an answer as the result of a SPARQL query, whereas PSYCHIC returns the queries themselves. This is a deliberate design choice since teaching a model to recognize patterned query structures is still an easier problem than teaching it to learn dynamic answers ranging from boolean values to lists of elements. Being able to correctly return the right query is essentially the hard part of the challenge, and the additional step to get the final answer consists in running the query returned by PSYCHIC using the SPARQL endpoint provided by the task organizers.

As such, we built the NS framework shown in Figure 3 to leverage the power of LLMs and symbolic reasoning. It takes as input the question-SPARQL pairs, processes them and prepares the question-context dataset needed for the PSYCHIC model. The model predicts the output in the form of a string containing the query, the separator [SEP] and the entity list. Two additional modules, the query and entity sanitizers, extract the relevant pieces of information from the output, namely the query and the entity list, by splitting the output string and validating the query and entity elements by matching them to their respective patterns. Through this sanitization process, these modules can perform error-correction to account for malformed strings returned by the model, e.g., transforming `'select distinct? answer where {? answer < https://dblp.org/rdf/schema#authoredby> < https://dblp.org/pid/00/2941 >}'` to `'select distinct`

?answer where { ?answer <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/00/2941> }' for the query and ['< https : / / dblp . org / pid / 00 / 2941 >'] to ['<https://dblp.org/pid/00/2941>'] for the entity list.

The resulting output from the query sanitizer module is a correctly-formed SPARQL query that is run using the SPARQL endpoint to return the expected final answer. The output of the entity sanitizer is the correctly-formed entity list.

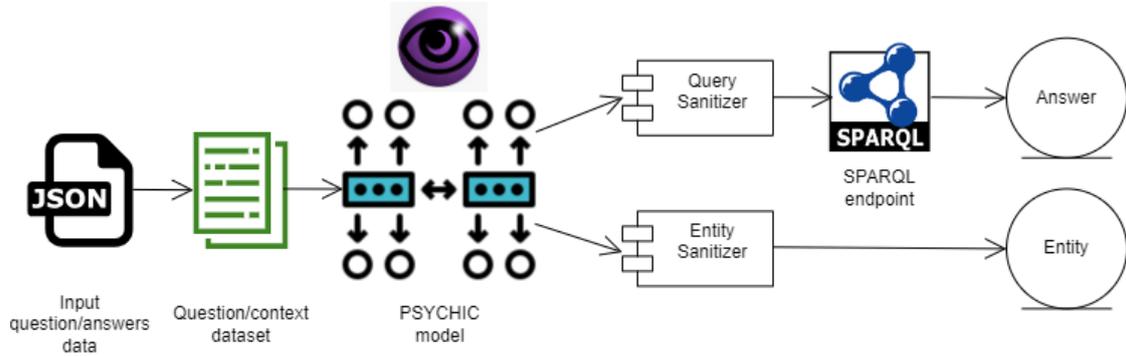


Figure 3: Solution system architecture

3.2.3. Experimental setup

We divided our experiment into two phases: training and inference or evaluation phase.

For the training phase, we treated the question and paraphrase as distinct questions to effectively double our overall training, validation and test sets. We trained the PSYCHIC model on the training and validation splits and evaluated on the test split. We set the following training hyperparameters for the model: learning rate = $2e-05$, training batch size = 16, evaluation batch size = 16, seed = 42, optimizer = Adam with betas = (0.9, 0.999) and epsilon = $1e-08$, and number of epochs = 3.

For the inference step, we distinguished two phases: the dev phase and the final phase. The dev phase corresponds to the phase when the first 500 random question-SPARQL pairs are made available by the organizers. The final phase represents the true system evaluation phase whereby the dataset used is the set of 500 random questions (and their paraphrases). The phases differ by the nature of the input given to the PSYCHIC model. In the dev phase, the context was constructed the same way as for DBLP-QuAD, i.e., following the [CLS] + QUERY_TYPE + [SEP] + TEMPLATE_ID + [SEP] + QUERY + [SEP] + ENTITIES pattern. In the final phase, since the only available information is the question and its paraphrase, we used an entity linker provided by the task organizers that leverages a t5-base language model with translating embeddings to return a list of predicted entities for each provided question. These results were concatenated using the symbolic rule engine to form a different context pattern, i.e., [CLS] + EL_RESULT1 + [SEP] + EL_RESULT2 + [SEP] + ... + [SEP] + EL_RESULTN for N returned entity results.

We used the F1-QA and F1-EL metrics, representing the F1 scores on each of the QA and EL sub-tasks respectively, to evaluate our system. All experiments were performed on a Dell G15

Special Edition 5521 hardware with 14 CPU Cores, 32 GB RAM and NVIDIA GeForce RTX 3070 Ti GPU.

4. Results

Table 1 displays the training results of the PSYCHIC model. Over 3 epochs, the model learns consistently with a perfect loss of 0 by the third epoch. Having integrated symbolic knowledge through the insertion of special tokens in the context proves to be a sound way to represent the different context chunks we want the model to learn. By the end of training, PSYCHIC can reliably return a complex output constituted of a query and an entity list for an input question.

The results of Table 2 seem to corroborate these findings. In the dev phase, the seed input structure is identical to that of DBLP-QuAD which means that PSYCHIC expects a similar context pattern. The model achieves a perfect score on both the QA and EL sub-tasks which suggests it is perfectly capable of discriminating query tokens from entity tokens. The results of the final phase are interesting in the sense that they seem to contradict this theory. The F1-QA score in particular suggests the model fails to predict any question query correctly. The reason for this bad performance lies in the fact that the input structure for the final phase set is radically different from the DBLP-QuAD and dev data as it contains no accompanying context for questions. Since PSYCHIC is an extractive QA model, it becomes very challenging for it to predict any kind of information without relevant context. We were not able to fill this context gap in the scope of this challenge. For the EL sub-task, we were able to fill the gap by using the entity linker predictions as missing context for the input questions. Because these context patterns were unseen by PSYCHIC during the training and dev phases, the F1-EL score achieved by the model is impressive since it reveals it can correctly identify valid entity pattern structures. This performance ranks PSYCHIC at third overall in the EL sub-task final standings as shown in Table 3.

Epoch	Training Loss	Step	Validation Loss
1.0	0.001	1000	0.0001
2.0	0.0005	2000	0.0000
3.0	0.0002	3000	0.0000

Table 1
PSYCHIC training results

Phase	F1-QA	F1-EL
Dev	100.00	100.00
Final	00.18	71.00

Table 2
Final system evaluation performance

Team	F1-EL
nsteinmetz	83.53
ruijie.wang	79.61
PSYCHIC	71.00
Shreyaar12	62.35

Table 3
Final competition standings for the EL sub-task

5. Conclusion

In this shared task, we propose a NS framework to tackle QA and EL sub-tasks over a KG. We explore the effects of including symbolic learning in the context of LLMs and evaluate the overall performance on the sub-tasks for a changing context. In the future, we plan to extend these symbolic mechanisms to generative models such as retrieval augmented generation (RAG) pipelines.

References

- [1] C. Peng, F. Xia, M. Naseriparsa, F. Osborne, Knowledge graphs: Opportunities and challenges, *Artificial Intelligence Review* (2023) 1–32.
- [2] D. Banerjee, S. Awale, R. Usbeck, C. Biemann, Dblp-quad: A question answering dataset over the dblp scholarly knowledge graph, *arXiv preprint arXiv:2303.13351* (2023).
- [3] K. Ishwari, A. Aneeze, S. Sudheesan, H. Karunaratne, A. Nugaliyadde, Y. Mallawarachchi, *Advances in natural language question answering: A review*, *arXiv preprint arXiv:1904.05276* (2019).
- [4] J. Z. Pan, S. Razniewski, J.-C. Kalo, S. Singhanian, J. Chen, S. Dietze, H. Jabeen, J. Omeliyanenko, W. Zhang, M. Lissandrini, et al., Large language models and knowledge graphs: Opportunities and challenges, *arXiv preprint arXiv:2308.06374* (2023).
- [5] W. Zheng, M. Zhang, Question answering over knowledge graphs via structural query patterns, *arXiv preprint arXiv:1910.09760* (2019).
- [6] S. Pramanik, J. Alabi, R. S. Roy, G. Weikum, Uniqorn: unified question answering over rdf knowledge graphs and natural language text, *arXiv preprint arXiv:2108.08614* (2021).
- [7] A. C. Sima, T. Mendes de Farias, M. Anisimova, C. Dessimoz, M. Robinson-Rechavi, E. Zbinden, K. Stockinger, Bio-soda ux: enabling natural language question answering over knowledge graphs with user disambiguation, *Distributed and Parallel Databases* 40 (2022) 409–440.
- [8] W. Zheng, J. X. Yu, L. Zou, H. Cheng, Question answering over knowledge graphs: question understanding via template decomposition, *Proceedings of the VLDB Endowment* 11 (2018) 1373–1386.
- [9] C. Nikas, P. Fafalios, Y. Tzitzikas, Open domain question answering over knowledge graphs using keyword search, answer type prediction, sparql and pre-trained neural models, in:

- The Semantic Web–ISWC 2021: 20th International Semantic Web Conference, ISWC 2021, Virtual Event, October 24–28, 2021, Proceedings 20, Springer, 2021, pp. 235–251.
- [10] S. Cm, J. Prakash, P. K. Singh, Question answering over knowledge graphs using bert based relation mapping, *Expert Systems* (2022) e13456.
 - [11] D. Diomedi, A. Hogan, Question answering over knowledge graphs with neural machine translation and entity linking, 2021. doi: 10.48550, arXiv preprint arXiv:2107.02865 (2021).
 - [12] S. Aghaei, E. Raad, A. Fensel, Question answering over knowledge graphs: A case study in tourism, *IEEE Access* 10 (2022) 69788–69801.
 - [13] C. Mavromatis, G. Karypis, Rearev: Adaptive reasoning for question answering over knowledge graphs, arXiv preprint arXiv:2210.13650 (2022).
 - [14] S. Li, K. W. Wong, C. C. Fung, D. Zhu, Improving question answering over knowledge graphs using graph summarization, in: *Neural Information Processing: 28th International Conference, ICONIP 2021, Sanur, Bali, Indonesia, December 8–12, 2021, Proceedings, Part IV 28*, Springer, 2021, pp. 489–500.
 - [15] R. Dutt, K. Bhattacharjee, R. Gangadharaiyah, D. Roth, C. Rose, Perkgqa: Question answering over personalized knowledge graphs, in: *Findings of the Association for Computational Linguistics: NAACL 2022*, 2022, pp. 253–268.
 - [16] A. Saxena, A. Tripathi, P. Talukdar, Improving multi-hop question answering over knowledge graphs using knowledge base embeddings, in: *Proceedings of the 58th annual meeting of the association for computational linguistics*, 2020, pp. 4498–4507.
 - [17] Z. Zuo, Z. Zhu, W. Wu, W. Wang, J. Qi, L. Zhong, Improving question answering over knowledge graphs with a chunked learning network, *Electronics* 12 (2023) 3363.
 - [18] M. R. A. H. Rony, D. Chaudhuri, R. Usbeck, J. Lehmann, Tree-kgqa: an unsupervised approach for question answering over knowledge graphs, *IEEE Access* 10 (2022) 50467–50478.
 - [19] J. Shi, Z. Yuan, W. Guo, C. Ma, J. Chen, M. Zhang, Knowledge-graph-enabled biomedical entity linking: a survey, *World Wide Web* (2023) 1–30.
 - [20] M. Dubey, D. Banerjee, D. Chaudhuri, J. Lehmann, Earl: joint entity and relation linking for question answering over knowledge graphs, in: *The Semantic Web–ISWC 2018: 17th International Semantic Web Conference, Monterey, CA, USA, October 8–12, 2018, Proceedings, Part I 17*, Springer, 2018, pp. 108–126.
 - [21] N. Steinmetz, Entity linking for kgqa using amr graphs, in: *European Semantic Web Conference*, Springer, 2023, pp. 122–138.
 - [22] P. Radhakrishnan, P. Talukdar, V. Varma, Elden: Improved entity linking using densified knowledge graphs, in: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018, pp. 1844–1853.
 - [23] A. Thawani, M. Hu, E. Hu, H. Zafar, N. Divvala, A. Singh, E. Qasemi, P. Szekely, J. Pujara, Entity linking to knowledge graphs to infer column types and properties, 2019, semantic web challenge on tabular data to knowledge graph matching, iswc, 2019.
 - [24] Q. Li, F. Li, S. Li, X. Li, K. Liu, Q. Liu, P. Dong, Improving entity linking by introducing knowledge graph structure information, *Applied Sciences* 12 (2022) 2702.
 - [25] B. Huang, H. Wang, T. Wang, Y. Liu, Y. Liu, Entity linking for short text using structured knowledge graph via multi-grained text matching (2020).

- [26] D. Banerjee, D. Chaudhuri, M. Dubey, J. Lehmann, Pnel: Pointer network based end-to-end entity linking over knowledge graphs, in: *The Semantic Web–ISWC 2020: 19th International Semantic Web Conference, Athens, Greece, November 2–6, 2020, Proceedings, Part I* 19, Springer, 2020, pp. 21–38.
- [27] W. Ding, V. K. Chaudhri, N. Chittar, K. Konakanchi, Jel: applying end-to-end neural entity linking in jpmorgan chase, in: *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 2021, pp. 15301–15308.
- [28] D. Diomedi, A. Hogan, Entity linking and filling for question answering over knowledge graphs, in: *Natural Language Interfaces for the Web of Data (NLIWOD) Workshop*, 2022.