Deep Query Ranking for Question Answering over Knowledge Bases

Hamid Zafar[1], Giulio Napolitano[2], Jens Lehmann[1,2]

[1] Computer Science Institute, University of Bonn, Germany;
hzafarta@cs.uni-bonn.de, jens.lehmann@cs.uni-bonn.de
[2] Fraunhofer IAIS, Germany; giulio.napolitano@iais.fraunhofer.de,
jens.lehmann@iais.fraunhofer.de

Abstract. We study question answering systems over knowledge graphs which map an input natural language question into candidate formal queries. Often, a ranking mechanism is used to discern the queries with higher similarity to the given question. Considering the intrinsic complexity of the natural language, finding the most accurate formal counter-part is a challenging task. In our recent paper [1], we leveraged Tree-LSTM to exploit the syntactical structure of input question as well as the candidate formal queries to compute the similarities. An empirical study shows that taking the structural information of the input question and candidate query into account enhances the performance, when compared to the baseline system.

1 Introduction

Question answering (QA) systems provide a convenient interface to enable their users to communicate with the system through natural language questions. QA systems can be seen as advanced information retrieval systems, where a) users are assumed to have no knowledge of the query language or structure of the underlying information system; b) the QA system provides a concise answer, as opposed to search engines where users would be presented with a list of related documents. There are three types of source of information being consumed by QA systems, namely unstructured resources (e.g. Wikipedia pages), structured resources and hybrid sources. Given the extensive progress being made in large scale Knowledge Graphs (KGs), we mainly focus on QA systems using KGs as their source of information, since such systems might be able to yield more precise answers than those using a unstructured sources of information.

Given the complexity of the QA over KGs, there is a proclivity to design QA systems by breaking them into various sequential subtasks such as Named Entity Disambiguation (NED), Relation Extraction (RE) and Query Building (QB) among others [2]. Considering the fact that the system might end up with more than one candidate queries due to uncertainty in the linked entities/relations, ambiguity of the input question or complexity of the KGs, a ranking mechanism in the final stage of the QA system is required to sort the candidate queries based on their semantic similarity in respect to the given natural language question.

Although considerable research has been devoted to QA over KG, rather less attention has been paid to query ranking subtask.

2 Related Work

Bast el al. [3] were inspired by the learning-to-rank approach from the information retrieval community to rank candidate queries using their feature vector, which contains 23 manually crafted features such as *number of entities in the query candidate*. They considered the ranking problem as a preference learning problem where a classifier (e.g. logistic regression) is supposed to pick the better option out of two given options. In a similar line of work, Abujabal et al. [4] hand-picked 16 features and utilized a random forest classifier to learn the preference model. Identifying the feature set requires manual intervention and depends heavily on the dataset at hand. In order to avoid that, Bordes [5] proposed an embedding model, which learns a fixed-size embedding vector representation of the input question and the candidate queries such that a score function produces a high score when the matching question and query are given. Inspired by the success of [5], Yih et al. [6] used deep convolutional neural networks to learn the embeddings and compute semantic similarity of the generated chains of entity/relation with respect to the given question. Despite their advantage to avoid using any manually engineered features, the models introduced by [5,6] failed to exploit the syntactical structure of the input question or the candidate queries. In the next section, we propose to use Tree-LSTM [7] in order to take advantage of the latent information in the structure of question and the candidate queries.

3 Deep Query Ranking

Consider the example question "What are some artists on the show whose opening theme is Send It On?" from [1], the candidate queries of an arbitrary QA pipeline are illustrated in Fig. 3. The candidate queries are similar to each other in the sense that they are made up of a set of entities and relations, which are shared among them. Motivated by the success of embedding models [5,6], we aim to enhance them by considering the structure of input question and candidate queries as well. In this regard, Tai et al. [7] proposed a Tree-LSTM model, which considers the tree representation of the input, as opposed to most RNN based models (e.g. LSTM) which take a sequence of tokens as input. The state of a Tree-LSTM unit depends on the children units (Fig. 2), enabling the model to consume the tree-structure of the input. Consequently, not only the input sequence matters but also how the elements of the input are connected together.

In order to learn the embedding vector we used a similarity function [8] along with two Tree-LSTM models for the input question and the candidate queries. The input to the *Question Tree-LSTM* is the dependency parse tree of the question (Fig. 1), whilst the tree-representation of the candidate queries is fed into the *Query Tree-LSTM* (Fig. 3). The Tree-LSTM models are trained

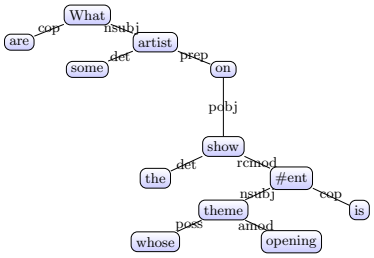


Fig. 1: Dependency parse tree of the running example. (from [1])

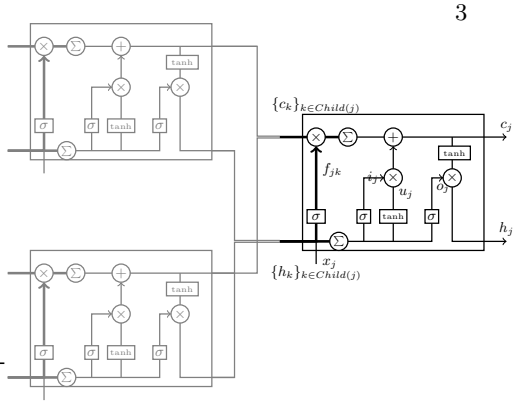
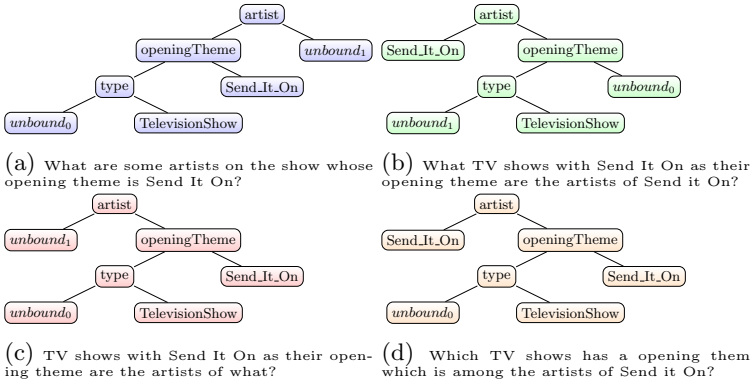


Fig. 2: The architecture of Tree-LSTM

to map their input into a latent vectorized representation such that the pair of question/correct query would have the highest score in respect to the others.



(a) What are some artists on the show whose opening theme is Send It On?

(b) What TV shows with Send It On as their opening theme are the artists of Send it On?

(c) TV shows with Send It On as their opening theme are the artists of what?

(d) Which TV shows has a opening them which is among the artists of Send it On?

Fig. 3: Tree representation of the queries along with their NL meaning. (from [1])

4 Empirical Study

We prepared two datasets for the ranking model based on the LC-QuAD dataset [9] which consists of 5,000 question-answer pairs. Both datasets consist of questions and candidate queries. The first dataset, *DS-Min* is constructed using only the correct entities/relations, while *DS-Noise* is generated using the correct entities/relations plus four noisy ones per each linked item in the question.

The performance of the Tree-LSTM ranking model is reported in Table 1. The Tree-LSTM outperforms vanilla LSTM in both datasets. While Tree-LSTM performs better in *DS-Noise* in comparison to *DS-Min*, LSTM model degrades in *DS-Noise*. Although there are more training data in *DS-Noise* with balanced distribution of correct/incorrect data items, LSTM is not able to benefit from the information laying in the structure of its input, in contrast to Tree-LSTM.

Table 1: The accuracy of Tree-LSTM vs. LSTM (from [1])

Dataset	Size	Distribution(%) Correct/Incorrect	LSTM(F1)	Tree-LSTM(F1)
DS-Min	5,930	0.85/0.15	0.54	0.75
DS-Noise	11,257	0.46/0.54	0.41	0.84

5 Conclusions

We presented the problem of ranking formal queries, with the goal of finding the query that truly captures the intention of a given question. We reviewed the recent attempts to the problem and introduced our findings on using Tree-LSTM from our recent paper [1]. The model learns an embedding vector which captures the dependency parsing structure of the question and tree-representation of the queries to compute the similarity of the pairs for improved ranking.

Acknowledgements This research was supported by EU H2020 grants for the projects HOBBIT (GA no. 688227) and WDAqua (GA no. 642795) as well as by German Federal Ministry of Education and Research (BMBF) funding for the project SOLIDE (no. 13N14456).

References

1. H. Zafar, G. Napolitano, and J. Lehmann, "Formal query generation for question answering over knowledge bases," in *Extended Semantic Web Conference*, 2018.
2. D. Diefenbach, V. Lopez, K. Singh, and P. Maret, "Core techniques of question answering systems over knowledge bases: a survey," *Knowledge and Information systems*, pp. 1–41, 2017.
3. H. Bast and E. Haussmann, "More accurate question answering on freebase," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pp. 1431–1440, ACM, 2015.
4. A. Abujabal, M. Yahya, M. Riedewald, and G. Weikum, "Automated template generation for question answering over knowledge graphs," in *Proceedings of the 26th International Conference on World Wide Web*, pp. 1191–1200, 2017.
5. A. Bordes, S. Chopra, and J. Weston, "Question answering with subgraph embeddings," *arXiv preprint arXiv:1406.3676*, 2014.
6. S. W.-t. Yih, M.-W. Chang, X. He, and J. Gao, "Semantic parsing via staged query graph generation: Question answering with knowledge base," in *Proceedings of the Joint Conference of ACL and AFNLP*, 2015.
7. K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," in *ACL*, 2015.
8. W.-t. Yih, M. Richardson, C. Meek, and M.-W. Chang, "The value of semantic parse labeling for knowledge base question answering.," in *54th Annual Meeting of the Association for Computational Linguistics*, pp. 201–206, 2016.
9. P. Trivedi and M. Dubey, "A corpus for complex question answering over knowledge graphs," in *16th International Semantic Web Conference*, 2017.