# Knowledge Base Question Answering

## 1 Introduction

Knowledge base plays an important role in online applications like search engine, question answering, chatbot etc. Knowledge Base Question Answering (KBQA) has been a challenging task which requires both natural language understanding and symbolic reasoning capability. Recently, pre-trained language models have brought new opportunities for this task.

KBQA methods fall into two broad cateogies, **semantic parsing**-based (SP-based) methods and **information retrieval**-based (IR-based) methods [2]. The former approach parses a question into a logic form and then executes it against the KB to obtain the answers. The latter approach retrieves a question-specific graph and then generates the final answers based on the extracted graph (by applying some ranking algorithms to select entities or directly generate answers with text decoder). To summarize, the two approaches follow either a *parse-then-execute* paradigm or a *retrieve-and-generate* paradigm.

**Project Goal** Given benchmarks [format: (question, answer) or (question, logic form)], the goal of this project is to implement traditional KBQA methods (SP-based, IR-based) and further improve them with pre-trained language models (You can pick up smaller size model like distilbert to replace some components). You're supposed to learn how to conduct semantic parsing (SP-based) to answer questions based on a knowledge base, how to retrieve precise subgraphs for reasoning (IR-based), and conduct experiments to show how pre-trained language models help improve these methods.

## 2 Research Questions

The following are the research questions we will aim to answer in this project:

**RQ1:** How to transform natual language questionw to executable program (e.g., SPARQL or query graph)?

**RQ2:** How to conduct multi-hop reasoning in a knowledge base to answer factual questions given a topic entity?

**RQ2:** How to combine large language models in the SP-based paradigm and IR-based paradigm?

## 3 Experimental Setup

### 3.1 Benchmark and Methods

#### 3.1.1 SP-based methods and datasets

- Required benchmark: KQA Pro (http://thukeg.gitee.io/kqa-pro/). For ease of experiment, you can split the val.json into two parts, taking the last 5000 pieces of samples as test set, and others as evaluation set. Or, you can choose to do random split based on given original train.json and val.json.
- Optional benchmark: WebQSP and CWQ (weak supervision signals, only final answer)
- Required baseline: RNN seq2seq SPARQL parser (https://github.com/shijx12/KQAPro_Baselines)

#### 3.1.2 IR-based methods and datasets

- Required benchmark: WebQSP (relatively simple KBQA task, small size dataset)
- Optional benchmark: CWQ (Complex KBQA reasoning), MetaQA
- required baseline: GraftNet [3] (code: https://github.com/haitian-sun/GraftNet) and Neural State Machine (NSM) [1] (https://github.com/RichardHGL/WSDM2021_NSM)

## 3.2 Expected Outcomes

You are expected to

1. Implement SP-based methods based on seq2seq framework (https://github.com/shijx12/KQAPro_Baselines): taking questions as input, generate SPARQL (query language to knowledge base) or query graph (designed query).
2. Retrieve question-specific graph with heuristics (e.g., Personalized Pagerank) and using graph neural networks to conduct representation learning [3, 1].
   Code reference: NSM model: https://github.com/RichardHGL/WSDM2021_NSM (also contain code for graph retrieval with heuristics)
3. Replace RNN (LSTM/GRU) in above methods with pre-trained language model (PLM), compare model performance.

**Nice-to-have Outcomes**

1. Retrieve question-specific graph with semantic similarity (based on pre-trained language model), this may require more GPU, but you can still implement with smaller size pre-trained language models. Refer to https://github.com/RUCKBReasoning/SubgraphRetrievalKBQA.
2. Train SP-based methods with only answers (weak supervision signals). When only weak supervision signals are provided, you need to train neural networks with reinforcement learning (e.g, policy gradient). Refer to https://github.com/lanyunshi/Multi-hopComplexKBQA.

**Remarks**

- Refer to literature review [2] for more background about (complex) KBQA task
- BART seq2seq SPARQL parser can be viewed as the PLM improved version of RNN seq2seq SPARQL parser, you can also try other PLM
- This project needs GPUs. You can try your own computers if they come with 16G GPUs. Otherwise contact Jie <j.yang-3@tudelft.nl>. Make sure your code runs successfully on small (sampled) data before you deploy that to run on larger machines : ).

# References

[1] G. He, Y. Lan, J. Jiang, W. X. Zhao, and J.-R. Wen, "Improving multi-hop knowledge base question answering by learning intermediate supervision signals," in *Proceedings of the 14th ACM international conference on web search and data mining*, 2021, pp. 553–561.

[2] Y. Lan, G. He, J. Jiang, J. Jiang, W. X. Zhao, and J.-R. Wen, "Complex knowledge base question answering: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[3] H. Sun, B. Dhingra, M. Zaheer, K. Mazaitis, R. Salakhutdinov, and W. W. Cohen, "Open domain question answering using early fusion of knowledge bases and text," *arXiv preprint arXiv:1809.00782*, 2018.