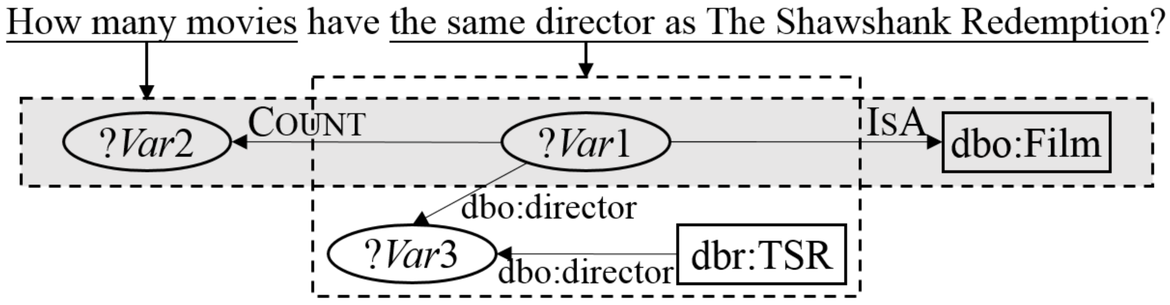
Leveraging Frequent Query Substructures to Generate Formal Queries for Complex Question Answering

Jiwei Ding, Wei Hu, Qixin Xu and **Yuzhong Qu**
National Key Laboratory for Novel Software Technology, Nanjing University, China
{jwdingnju,qxxunju}@outlook.com, {whu,yzqu}@nju.edu.cn

1. Introduction

Background

- Knowledge-based question answering (KBQA) systems transform natural language questions to formal queries (e.g., SPARQL).
- Formal query generation aims to generate correct executable queries over knowledge bases, given entity and relation linking results.



- Formal query generation is expected to have the capabilities of:
  (1) Recognize and paraphrase differnet kinds of constraints, e.g., "movie" stands for a type constraint <dbo:Film>; "the same … as" stands for a subgraph-level constraint in the dashed box;
  (2) Recognize and paraphrase aggregations, e.g., "How many"→COUNT;
  (3) Organize all the above to generate an executable query.
- Current approaches may suffer from the lack of training data, especially for long-tail questions with rarely appeared structures. Furthermore, current approaches cannot handle questions with unseen query structures.

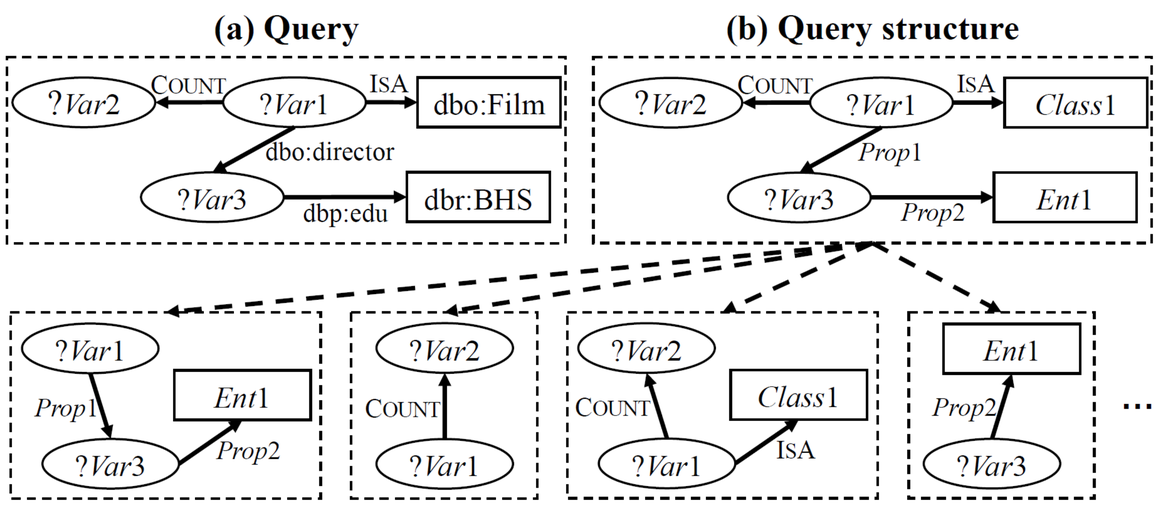
Main idea

- **Observation:** the query structure for a complex question may rarely appear, but it usually contains some query substructures that frequently appeared in other questions.
- Instead of predicting the query structure for the whole question, we predict (all) query substructures contained in the question.

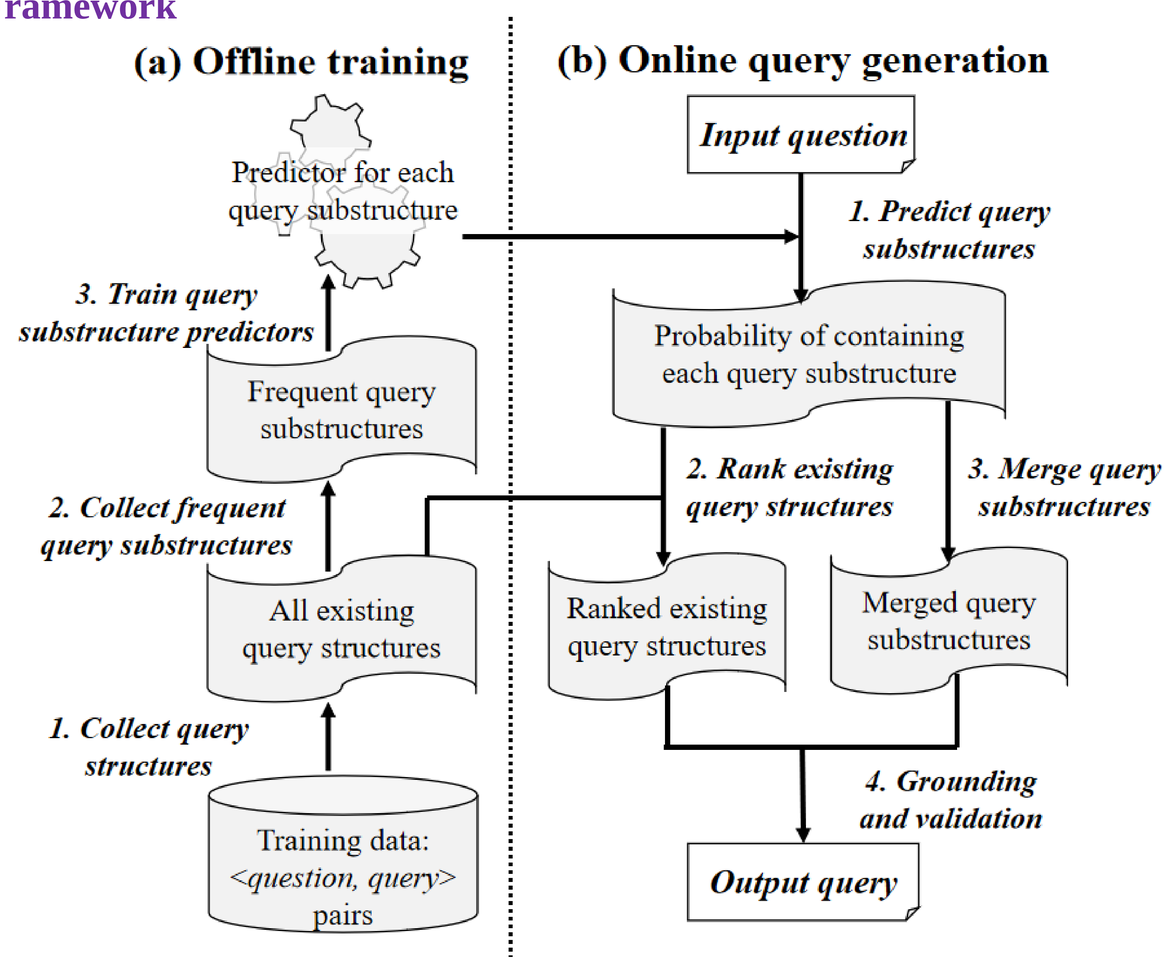
2. The Proposed Approach (SubQG)

Preliminaries

- **Query structure** is defined as a set for all structurally-equivalent queries.
- For two query structures S_a and S_b, if the representative query of S_a has a subgraph which is structurally-equivalent with the representative query of S_b, we say S_b is a **query substructure** of S_a.
- Illustration of a query, a query structure and query substructures:



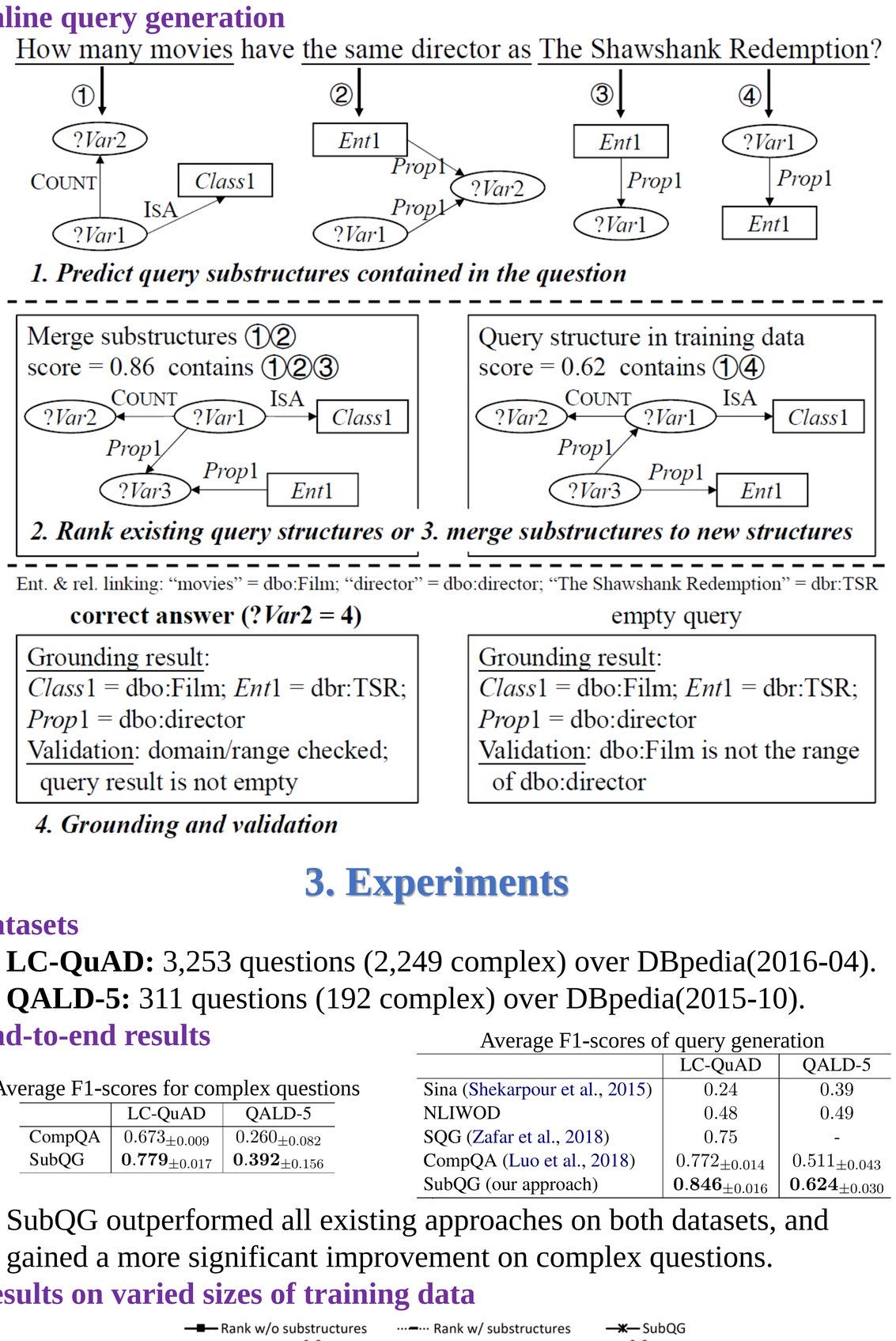
Framework



Offline training

- **Collect query structures.** We first discover the structurally equivalent queries in the training data, and then extract all query structures.
- **Collect frequent query substructures.** We decompose each query structure to collect query substructures. A query substructure is considered as a frequent query substructure if it appears more than γ times.
- **Train query substructure predictors.** We train an Attention-based BiLSTM network for each frequent query substructure, to predict whether the target query of the input question contains the substructure or not.

Online query generation



3. Experiments

Datasets

- **LC-QuAD:** 3,253 questions (2,249 complex) over DBpedia(2016-04).
- **QALD-5:** 311 questions (192 complex) over DBpedia(2015-10).

End-to-end results

Average F1-scores for complex questions

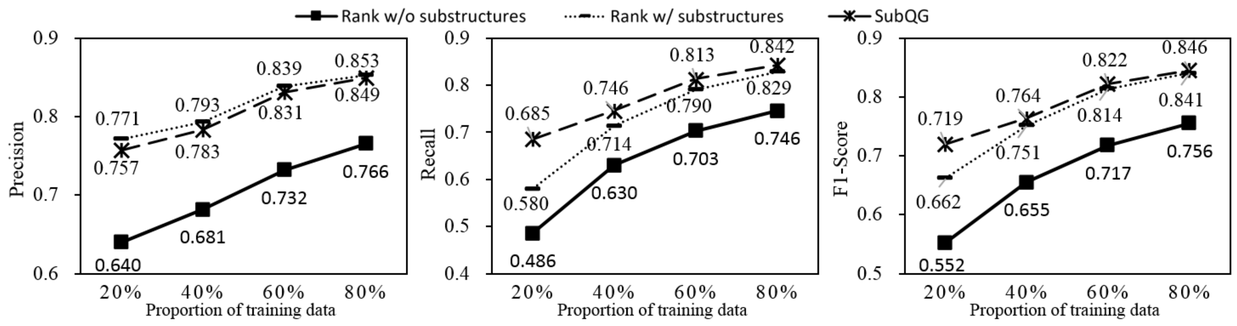
	LC-QuAD	QALD-5
CompQA	$0.673_{\pm 0.009}$	$0.260_{\pm 0.082}$
SubQG	$\mathbf{0.779}_{\pm 0.017}$	$\mathbf{0.392}_{\pm 0.156}$

Average F1-scores of query generation

	LC-QuAD	QALD-5
Sina (Shekarpour et al., 2015)	0.24	0.39
NLIWOD	0.48	0.49
SQG (Zafar et al., 2018)	0.75	-
CompQA (Luo et al., 2018)	$0.772_{\pm 0.014}$	$0.511_{\pm 0.043}$
SubQG (our approach)	$\mathbf{0.846}_{\pm 0.016}$	$\mathbf{0.624}_{\pm 0.030}$

- SubQG outperformed all existing approaches on both datasets, and gained a more significant improvement on complex questions.

Results on varied sizes of training data



- SubQG achieved a stable improvement (9% ~16%) compared with the approach which directly predicts the appropriate query structure for the whole input question.
- Although the merging method impaired the overall precision a little bit, it shows a bigger improvement on recall, especially when there is few training data (since more test questions have unseen query structures).

4. Conclusion

We introduced SubQG, a formal query generation approach based on frequent query substructures.

- SubQG firstly utilizes multiple neural networks to predict query substructures contained in the question, and then ranks existing query structures using a combinational function.
- SubQG merges query substructures to build new query structures for questions without appropriate query structures in the training data.
- SubQG achieved better results than the existing approaches in QALD-5 and LC-QuAD, especially for complex questions.

// EMNLP 2019