From Graph to Graph: AMR to SPARQL

Kanchan Shivashankar, Khaoula Benmaarouf, and Nadine Steinmetz

Technische Universität Ilmenau, Germany
firstname.lastname@tu-ilmenau.de

Abstract. We propose a graph to graph based transformation for KBQA systems. AMR graphs have proven promising for Question Answering (QA) systems and generating SPARQL queries. In this paper, we discuss using AMR graph for multilingual QA systems to generate SPARQL queries for Wikidata. The approach shows promising results and has scope for further improvement.

Keywords: Question Answering · MultilingualQA · Semantic Web · AMR · SPARQL · Wikidata

1 Introduction

The field of Knowledge Base Question Answering (KBQA) has seen a huge influx in research with many benchmark datasets being introduced. Question Answering over Linked Data (QALD) has been one such prominent dataset. The QALD 10 challenge deals with multilingual QA for Wikidata. With this paper, we propose a generalized solution for generating SPARQL queries from natural language questions using the Abstract Meaning Representation (AMR) combined with rules to generate the SPARQL query.

Question answering system poses challenges such as handling complex questions and multilingual questions. QA systems built on knowledge bases add the complexities of natural language to query mapping, entity and relation mapping and knowledge base generalizations. A solution that is robust and flexible to handle these complexities is the current challenge.

AMR is a graph-based representation of the semantic information of a language. Its ability to abstract makes it a simple yet powerful form of natural language representation. It is widely used in simplifying NLP tasks and for multilingual data. It captures the semantic representation of a language ignoring the syntactic information. Thus, a sentence with the same meaning which can be worded in multiple ways in natural language, will have a single AMR representation. AMR has been used previously in KBQA tasks for DBpedia knowledge base [3]. Although AMRs favor and are designed for the English language, it has shown promising results for processing multilingual data [1].

Our approach takes into account the AMR graphs of the English and German language version of a question and extracts the SPARQL graph for the query.

Graph based transformation methods and rule based approaches are widely and successfully used in query generation and question answering systems. With our limited understanding of AMR syntax and predicted answer types, we attempt to transform the AMR graph into query paths and define rules to generate SPARQL queries.

2 Data Analysis

Data analysis is performed to extract and visualize different features of the data. It provides insights into the data and design an approach that can easily be interfaced with the data. Our dataset consists of multilingual question and answer (QA) pairs. The data analysis step was performed on the training (QALD9 Plus) and test (QALD10) data. The training data consists of 412 question-answer pairs across 9 languages (English, German, Russian, French, Armenian, Belarusian, Lithuanian, Bashkir, and Ukrainian). Test data contains 394 QA pairs across 4 different languages (English, German, Russian and Chinese). The datasets are provided in json format and contains language - question pairs, SPARQL query for Wikidata and Answers.

3 From Text to Graph to Graph

Our approach consists of two main steps: (1)generation of the AMR graph and (2)deduction of the SPARQL query from the graph. Our pipeline for dataflow is depicted in Figure 1.

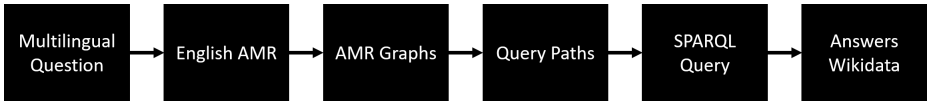


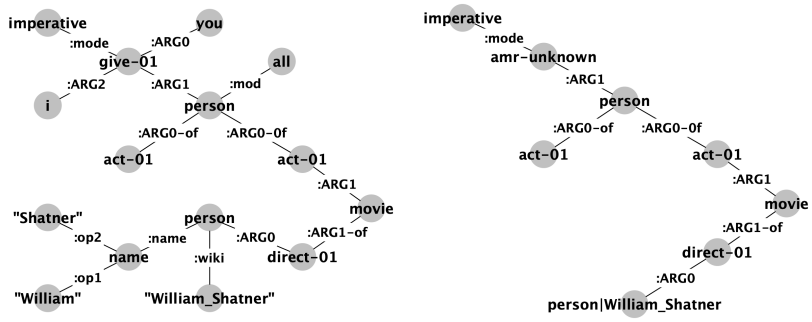
Fig. 1: Dataflow using our approach

3.1 AMR Graph Generation

We use a pre-trained multilingual AMR parser[1], trained on 4 different languages (English, German, Italian, Spanish and Chinese), to generate AMRs for English and German sentences from the QALD10 test dataset. This step is followed by generating alignments to the AMR using JAMR alignment[2]. The JAMR model annotates alignment information to the output of the AMR parser from the previous step. Alignment provides edge and node information, which can be used to plot AMR graphs. The AMR graphs act as the first step in generating SPARQL queries.

3.2 SPARQL Query Generation

In this step, we generate the SPARQL query graph from the AMR graph. In theory, the AMR graphs for the same input question in different languages should look the same. But, in fact, they often differ. In many cases, these differences stem from erroneous entity detection. Thereby, parts of an actual entity surface form are detected as part of the question, which results in incorrect dependencies and nodes. Therefore, we decided to take into account the AMR graphs for the English and German language version of the input question. The subsequent preprocessing steps are performed on both AMR graphs. The procedure is described in detail in the following sections.



(a) Initial graph.

(b) Simplified graph.

Fig. 2: AMR graph for the question `Give me all actors starring in movies directed by William Shatner` (a) initial state and (b) after simplification

AMR Graph Simplification Figure 2a shows the initial AMR graph for the question *Give me all actors starring in movies directed by William Shatner*. Firstly, we simplify the graph by removing unnecessary nodes and merging nodes that belong together. We remove nodes that are empty or contain stop words as well as accompanying `:name` edges, if a `:wiki` mapping is given. References of relations are often split into several nodes in the AMR graph. For instance, for the question *How many grand-children did Jacques Cousteau have?*, the relation reference *have grand-children* is split into two nodes. We merge those nodes to get a complete label for the property mapping. And we identify the `amr-unknown` node in the graph. Figure 2b shows the simplified AMR graph.

Path Extraction We extract all paths from the AMR graph starting at the `amr-unknown` node to all ending nodes. We split a path at an *unknown entity* node, such as *movie* in our sample graph. At the position of the split, we introduce a new variable for the SPARQL query. For each path, the beginning node

(`amr-unknown`) constitutes the subject and the ending node constitutes the object of a triple. All edge and node labels on the path between start and end are concatenated as property label.

Query Generation Each path is transferred to two RDF triples: both options of using the first node as subject or object and the last node as object and subject respectively. For `n` triples, we generate 2^n different queries per question. For entity and property identification, we utilize the linkings from the AMR graph generation, fuzzy search on properties of the train dataset and the Falcon 2.0 API[1]. In addition, we utilize the predicted answer category [4] to add a `COUNT` operator if required, and identify `ASK` questions. Finally, we use `:quant` edges from the AMR graph to add quantification restrictions in a `FILTER` clause to the query.

Query Execution All queries per question are executed on a local instance of the Wikidata SPARQL endpoint[2]. If a query produces results, the categories of these results are compared with the predicted answer type category and accepted only if they match.

4 Evaluation

We evaluate our approach on the QALD 10 test dataset using the GERBIL framework[3] as shown in Table 1. The test dataset contains 394 questions. Our algorithm provides queries and answers for 146 questions. The remaining questions are cases that we do not take into account resp. cannot handle at this stage of our approach, such as comparative questions, questions containing a superlative, boolean questions with either one or more than two entities, and questions that require a property path in the SPARQL query.

	Micro F1	Micro Precision	Micro Recall	Macro F1	Macro Precision	Macro Recall	Macro F1 QALD
Baseline	0.0385	0.0204	0.3293	0.507	0.5068	0.5238	0.5776
Our Approach							
	0.3839	0.7153	0.2624	0.3215	0.3206	0.3312	0.4909
C2KB	0.3706	0.7594	0.2451	0.3252	0.3411	0.3362	
P2KB	0.3568	0.75	0.2341	0.4006	0.417	0.4006	
RE2KB	0.2739	0.5756	0.1797	0.3478	0.3594	0.3498	

Table 1: Evaluation results of our approach on the QALD 10 test dataset.

[1] https://labs.tib.eu/falcon/falcon2/api-use
[2] as provided by: https://hub.docker.com/r/qacompany/hdt-query-service
[3] https://gerbil-qa.aksw.org/gerbil/

5 Conclusion

Using AMR graphs for query generation in QA systems has provided promising results with our approach. We have achieved good results on the questions our system is able to handle (approx. 40 % of the questions). AMR graphs use numerous edges and node labels to represent different aspects of natural language. Understanding these keywords can help prepare and create rules to generate more complex queries. Especially additional operators, such as `LIMIT`, `ORDER`, `GROUP BY`, or `FILTER` are required in many complex questions. Future work includes the comprehension of the AMR graphs and transformation to SPARQL query (operators). Another area of focus would be the entity and relation linking processes for the Wikidata knowledge base. Some of the parts of our approach can be performed agnostic from the knowledge base. But, as the representation of facts might be quite different in the various knowledge bases, this information must be involved in the query generation process. This includes information about domain and range of properties and types of entities, among others.

References

1. Cai, D., Li, X., Ho, J.C.S., Bing, L., Lam, W.: Multilingual amr parsing with noisy knowledge distillation (2021), https://arxiv.org/abs/2109.15196
2. Flanigan, J., Thomson, S., Carbonell, J., Dyer, C., Smith, N.A.: A discriminative graph-based parser for the Abstract Meaning Representation. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1426–1436. Association for Computational Linguistics, Baltimore, Maryland (Jun 2014), https://aclanthology.org/P14-1134
3. Kapanipathi, P., Abdelaziz, I., Ravishankar, S., Roukos, S., Gray, A., Astudillo, R., Chang, M., Cornelio, C., Dana, S., Fokoue, A., Garg, D., Gliozzo, A., Gurajada, S., Karanam, H., Khan, N., Khandelwal, D., Lee, Y.S., Li, Y., Luus, F., Makondo, N., Mihindukulasooriya, N., Naseem, T., Neelam, S., Popa, L., Reddy, R., Riegel, R., Rossiello, G., Sharma, U., Bhargav, G.P.S., Yu, M.: Leveraging abstract meaning representation for knowledge base question answering (2020), https://arxiv.org/abs/2012.01707
4. Shivashankar, K., Benmaarouf, K., Steinmetz, N.: Reaching out for the answer: Answer type prediction. In: Proceedings of the SeMantic AnsweR Type prediction task (SMART) co-located with the 20th International Semantic Web Conference (ISWC 2021). CEUR-WS (2021)