Finding Rules that imply Facts based on a Knowledge Graph using a Relational Database

Tim Gutberlet and Janik Sauerbier

University of Mannheim, Mannheim, Germany
{tgutberl,jsauerbi}@mail.uni-mannheim.de

Abstract. Retrieving information from knowledge graphs is a highly relevant problem in academia and the industry. This includes among other things the knowledge graph/link prediction completion problem, and rule-based pattern recognition.
Knowledge graph completion (KGC) received a lot of attention in recent years and is of high practical relevance inside and outside of academia. KGs are often incomplete and potentially noisy which hinders information extraction. Many different techniques are used to approach the knowledge graph completion problem including various embeddings and rule-based methods. Prior research indicates that inductive logic programming and embedding models can be used together to improve inference quality. This can be done by combining embedding models and constraints derived from rules as the objective function of an integer linear programming problem or by employing other methods focused on combining the results of different approaches. Another method (could be/is) the use of rules during the training of embedding models. Especially in the latter case, there is a significant need for a time-efficient database architecture which is used to identify rules that implicate the predictions of the embedding models. To contribute to this research area, we compare different database architectures using different indexing, hashing and pre-processing methods.

Keywords: Knowledge graph completion · Knowledge representation and reasoning · Database design and models.

1 Introduction

Retrieving information from Knowledge Graphs has been a much-discussed topic in research over the years. Knowledge Graphs are used in different fields from biomedical applications [OpenBioLink, HETIONET, Bio2RDF] over supply chain risk analysis to semantic search applications. Understanding patterns and rules as well as predicting links based on these patterns are highly relevant problems in the context of Knowledge Graphs in academia and industry. There are already many effective tools available that solve these problems either as black box Knowledge Graph embedding models or through learning interpretable symbolic rules. We want to contribute to the field by analyzing how to effectively

find rules that imply certain target facts, which have not been part of the Knowledge Graph yet. We are not concerned with learning those rules but just with identifying rules that have been learn previously and explain a certain target fact. Here we specifically explore relational database technologies. Our findings might be used to expand current approaches that aim to combine knowledge graph embedding models and rule-based approaches. This might be achieved by quickly finding all rules that imply the predictions by the embedding models. [. . .]

Beyond potentially facilitating faster link prediction in the future our findings can already be helpful to build tools that help understanding the dynamics of rules given large amounts of potential candidates for Knowledge Graphs. Our research might also be helpful to understand the limits of relational database technologies and might help to facilitate the comparison between different database technologies. [. . .]

To achieve those goals we ran a number of experiments aimed to illustrate how diffrent database structures impact the performance in terms of exectution time. Beyond that we also analyszed how certain queries (aka. the target facts) and certain rule types (e.g. by lenght of the rule body) performed in comparison to each other.

2 Related Work

- AnyBURL, specificly the latest IJCAI paper.[1]
- Other work on combining embeddings and rule-based approaches. (e.g. Wang, Quan, Bin Wang, and Li Guo. "Knowledge base completion using embeddings and rules."[3] or Zhang, Wen, et al. "Iteratively learning embeddings and rules for knowledge graph reasoning."[4])
- Work specifically focused on building effective databases (e.g. Indexing, Hashing and preprocessing stratagies) -> What is should we look into her?

3 Preliminaries

Knowledge Graphs and Rules A Knowledge Graph (KG) is a set of triples (subject, relation, object) which describe a directed graph. The edges are represented by the relations going from subject to object. The nodes are the set of entities which appear as subjects and objects of the triples. The individual triples are also called facts. Knowledge Graphs can be used to describe a variety of domains and can store knowledge about them. Possible domains are for example found in the biomedical field or in the context of the Semantic Web. Knowledge Graphs can be used to identify rules and patterns in those domains. This can be very helpful to for example analyze how different biomedical processes correlate with certain genes, diseases or treatments. For our purposes, we are concerned about first-order logic Horn rules. These rules consist of a head and a body where the head consists of one triple and the body consists of one or more triples. The subjects and objects of the triples can either be specific entities

or variables. Here are two example rules. We use the notation relation(subject, object) for the triples and capitalize the variables.

$$citizenOf(X, germany) \leftarrow bornIn(X, mannheim) \quad (1)$$

$$livesIn(X, germany) \leftarrow marriedTo(X, A_1), bornIn(A_1, germany) \quad (2)$$

The grounding of a rule is a set of triples that fulfil the rule. That means the Knowledge Graph contains triples with matching entities for all variables. Rules are not necessarily universally true and there might be sets of triples that match the body but there is no matching triple for the head. Based on the ratio of the number of groundings of a rule (matching head + body) and the number of matching bodies without a matching head we can derive a confidence score for any given rule.

Knowledge Graph Completion As Knowledge Graphs are often based on real-world data, they are prone to be incomplete. Therefore, one very relevant problem in the context of Knowledge Graphs is Link Prediction/Knowledge Graph Completion. The problem is concerned with identifying missing links (aka. missing facts) that might be true in the real world but are not yet part of the Knowledge graph. One example would be the generation of friendship suggestions for users of social media sites based on known connections. A variety of black box and interpretable approaches have been used to tackle the problem. These include Knowledge Graph embedding models and rules-based approaches.

4 Problem statement

The problem we are trying to solve is identifying rules that imply certain target facts that are not part of the Knowledge Graph. There are various rule learners like AnyBURL available that can learn a large number of rules even on very big Knowledge graphs in a short amount of time. That's why we are not concerned with learning those rules based on a given Knowledge Graph. Instead, we want to identify all matching rules for a certain target triple based on a previously learned set of rules. This means that the target triple should match the head of the rule and a subset of triples from the Knowledge Graph should match the body of the rule. We want to specifically explore in how far we can use traditional relational database technology to solve the problem as efficiently as possible.

5 Logical Modelling

The solution was implemented using the open source object-relational database system PostgreSQL. For this solution, PostgreSQL Version PostgreSQL 14.6 was used. - Modelled Rules in 5 groups by their type of head: - subBound: h(c, A) - A triple with relation r R, where the subject is bound to the constant C - objBound: h(A, c) - A triple with R - bothBound h(c, d) - A triple with relation

r R, where both subject and object are bound to a constant - noBoundUnequal: h(A, B) - A triple with relation r R, where both subject and object are variables that appear in the head - NoBoundEqual: h(A, A) - A triple with relation h R, where both subject and object represent the same Variable

6 Experiments

– Comparison with AnyBURL given the limitation of AnyBURL to only find the following rule types:

$$h(X, Y) \leftarrow r(X, Y) \tag{3}$$

$$h(X, Y) \leftarrow r_1(X, Z) \wedge r_2(Z, Y) \tag{4}$$

$$h(X, e_1) \leftarrow r(X, e_2) \tag{5}$$

https://www.overleaf.com/project/6370d1fa93b205423bfca1af

– XXX
– XXX

Our approach is written in Java and requires a PostgreSQL JDBC Driver installed. The source code and datasets, used in the experiments, can be found at https://github.com/timgutberlet/Online-Rule-Search-Database. We conducted all experiments on a Fujitsu Esprimo P957 (construction year 2017) with 32 GB RAM, 512 GB SSD, an Intel i7-7700 @ 3.6 GHz CPU and Ubuntu 22.04.1 LTS as an operating system.

7 Conclusions

Acknowledgements ...

References

1. Betz, Patrick, Christian Meilicke, and Heiner Stuckenschmidt. "Adversarial explanations for knowledge graph embeddings." International Joint Conferences on Artificial Intelligence, 2022.
2. Mayer, Marta Cialdea, and Fiora Pirri. "First order abduction via tableau and sequent calculi." Logic Journal of the IGPL 1.1 (1993): 99-117.
3. Wang, Quan, Bin Wang, and Li Guo. "Knowledge base completion using embeddings and rules." Twenty-fourth international joint conference on artificial intelligence. 2015.
4. Zhang, Wen, et al. "Iteratively learning embeddings and rules for knowledge graph reasoning." The World Wide Web Conference. 2019.