Comparison of Symbolic & Sub-Symbolic Approaches for Knowledge Graph Completion

Master Thesis

presented by
Lars Joormann
Matriculation Number 1721931

submitted to the
Data and Web Science Group
Prof. Dr. Stuckenschmidt
University of Mannheim

August 2022

Abstract

Add a brief summary!

Contents

1 Introduction **1**

1.1 Background and Motivation 1

1.2 Research Question 1

2 Knowledge Graphs **2**

2.1 Datasets 4

3 Knowledge Graph Completion **5**

3.1 Symbolic Approaches 6

3.1.1 AnyBURL 6

3.2 Sub-Symbolic Approaches 6

3.2.1 ComplEx 7

3.2.2 RESCAL 7

3.3 Comparison of the Approaches 7

3.4 Model Evaluation 7

4 Experimental Setting **8**

5 Comparison of Symbolic and Sub-Symbolic Performances **9**

6 Test Sets for Vulnerabilities **10**

7 Discussion **11**

8 Conclusion **12**

List of Algorithms

List of Figures

2.1 Example of a Knowledge Graph 2
2.2 Example of a Google Side Window 3
2.3 Applications of Knowledge Graphs 4

3.1 Example of an Embedding for a Knowledge Graph 7

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

1.2 Research Question

Chapter 2

Knowledge Graphs

Knowledge graphs are graph-structured knowledge bases. They store information in the form of relationships between entities. A single information in the graph is referred to as fact. It consist of two entities and a relation between those two. This can be expressed as a triple: $(head, relation, tail)$. Knowledge Graphs are referred to as graphs since their entities can be interpreted as nodes and the relations as labelled and directed edges in graph. The label here indicates which kind of relation the entities share and the direction indicates which entity is the head entity and which the tail entity i.e. an edge points from the head to the tail. An example for a knowledge graph can be seen in figure 2.1. [5]

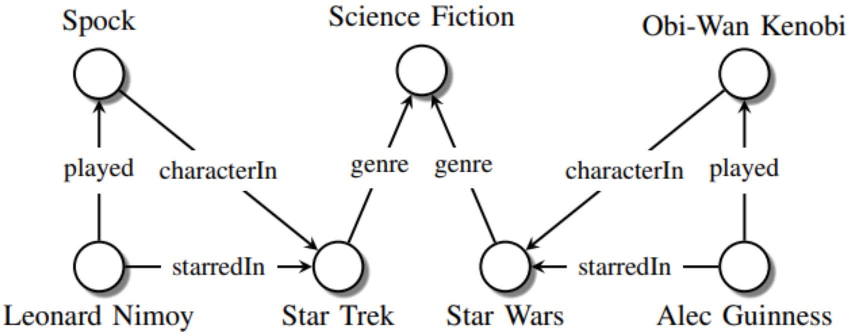
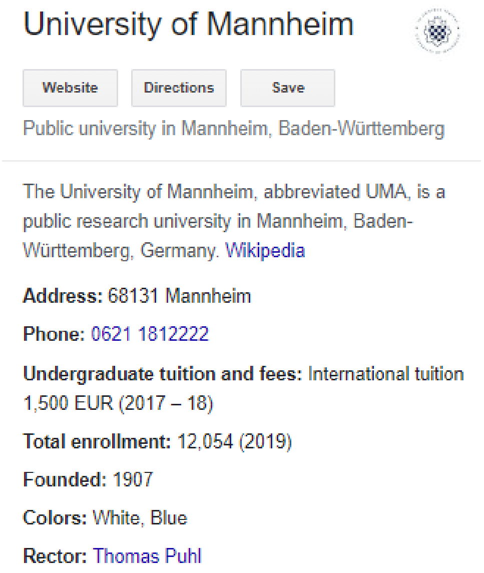


Figure 2.1: Example of a Knowledge Graph

The term 'knowledge graph' was first introduced by Google in 2012 [7]. In their blog they spoke about how they use their knowledge graph to enrich search engine results. The most noticeable part of how they use knowledge graphs are the side windows when searching for an entity with their search engine. An example of such can be seen in figure 2.2. Here we can see what kind of knowledge Google has about the University of Mannheim. For example it seems to be that *(University of Mannheim, founded_in, 1907)* is one of the facts in their Knowledge Graph.


University of Mannheim

Public university in Mannheim, Baden-Württemberg

The University of Mannheim, abbreviated UMA, is a public research university in Mannheim, Baden-Württemberg, Germany. Wikipedia

Address: 68131 Mannheim

Phone: 0621 1812222

Undergraduate tuition and fees: International tuition 1,500 EUR (2017 – 18)

Total enrollment: 12,054 (2019)

Founded: 1907

Colors: White, Blue

Rector: Thomas Puhl

Figure 2.2: Example of a Google Side Window

In the recent years knowledge graphs have become more and more popular and have found their way into further applications than search engines. An overview of applications can be seen in figure 2.3. Question Answering systems use knowledge graphs to enhance their results. Examples here include social chatbots and digital assistance like Siri. Recommender Systems leverage knowledge graphs as side information to improve and diversify their recommendations. Moreover, knowledge graphs are also used in information retrieval, domain-specific applications and more. [9]

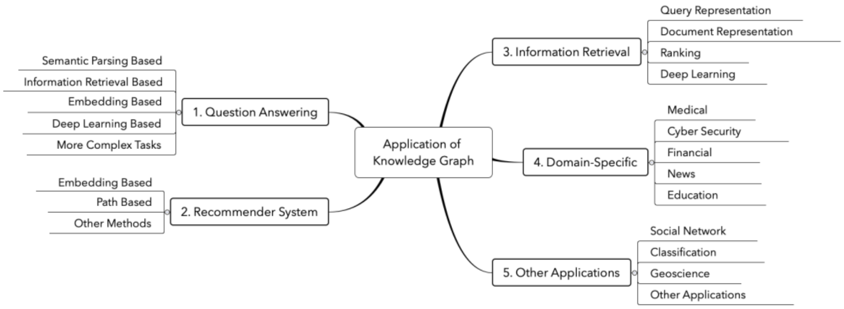


Figure 2.3: Applications of Knowledge Graphs

According to Paulheim [6] a knowledge graph is defined by the following four characteristics:

1. ”mainly describes real world entities and their interrelations, organized in a graph”
2. ”defines possible classes and relations of entities in a schema”
3. ”allows for potentially interrelating arbitrary entities with each other”
4. ”covers various topical domains”

The first characteristic defines that knowledge graphs consist of two kind of instances, entities and relations. An entity can be almost everything from an individual person to any kind of object. These entities are then linked through different kind of relations, which forms our graph.

The schema of the graph plays only a minor role. In most cases the instance-level statements (entities and triples) far outnumber the schema-level statements (entity classes and relations).

With the third characteristic Paulheim opens up the possibility that there are arbitrary relations between entities which are not included in the knowledge graph. The chapter 3 will discuss this further.

Lastly, another characteristic of knowledge graphs is that they do not focus on a single domain but interlink multiple topical domains.

2.1 Datasets

Chapter 3

Knowledge Graph Completion

As discussed in section 2 knowledge graphs are not complete. They contain noisy and incomplete data. It is practically impossible to cover every possible entity and relation existing in the real-world or even in their domain. There might be missing entities and relations or a Knowledge Graph can include two entities/ relations representing the same real-world entity. Knowledge graph completion tries to tackle these and other problems. It can be seen as a way of data cleaning for knowledge graphs. The solutions to the problems are defined into clear tasks, these include: entity resolution and entity and link prediction approaches.

Entity Resolution is according to Talburt ”the process of determining whether two [entities] are referring to the same object or to different objects”. [8]

Entity Prediction is the task of integrating new entities into the knowledge graphs. These entities are are discovered from other external sources and the knowledge graph includes no information about them. The goal is to find all possible relations this new entity has to the entities already existing in the graph. [1, p. 1]

Link Prediction is quiet similar to entity prediction. Instead of finding links for a new entity the goal here is to find all missing relations between already existing entities. [2, p. 125] Link prediction can be approached in two different ways: entity classification and triple classification. ”Entity classification tries to predict the type or the class of an entity [...]” [3] For a triple with a missing tail $(h, r, ?)$ the goal would be to list all entities which fit into the tail along with their confidence. Triple classification on the other hand is a binary tasks. Here the input is a compete triple (h, r, t) and the goal is to predict whether this triple is true or not. [3]

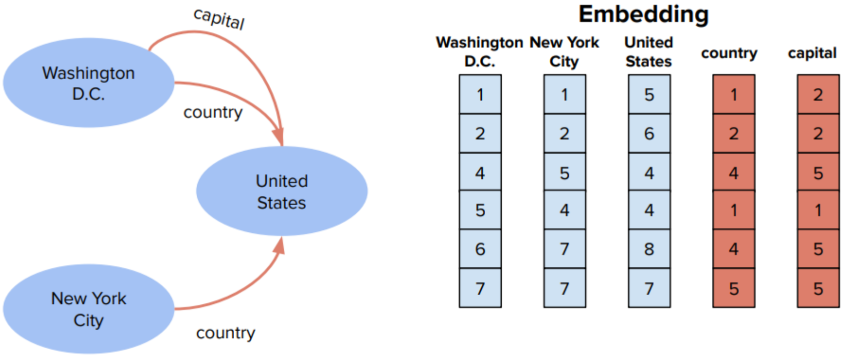
In the following we are going to focus on the task of link prediction. There are various models tackling the problem. They can be categorized into one of the following two categories: symbolic and sub-symbolic approaches. While symbolic methods try to learn an explicit symbolic representation of the patterns found in the knowledge graph, sub-symbolic methods try to solve the problem by learning a latent representation, also often referred to as embeddings, for every entity and relation in the knowledge graph. [4]

3.1 Symbolic Approaches

3.1.1 AnyBURL

3.2 Sub-Symbolic Approaches

Sub-symbolic approaches are based on statistics. They try to learn correlations from the existing triples in the knowledge graph. These can then be expressed as a model which in return allows to predict missing facts. [5] The most prominent models of this approach are embedding-based models. These models learn a vector for each entity and relation. The resulting embedding then represents our entity or relation. In this representation it is then assumed that similar entities and similar relations will have similar vectors. An example for these embeddings can be seen in figure 3.1. On the left side we see a knowledge graph with three entities and two relations. Every entity and relation are represented on the right side as an embedding. Our two entities *Washington D.C* and *New York City* are both cities and therefore we can assume that their semantical meaning are quiet similar. The embeddings of these two entities are also quiet similar which demonstrates that our previous assumption is correct. [**?**] Our embeddings are also called latent features because they can not be directly observed in the data. Instead our model has to infer these features from the data. [5]



Washington D.C.	New York City	United States	country	capital
1	1	5	1	2
2	2	6	2	2
4	5	4	4	5
5	4	4	1	1
6	7	8	4	5
7	7	7	5	5

Figure 3.1: Example of an Embedding for a Knowledge Graph

An embedding-based model is defined by three characteristics [**?**]:

1. representations of entities and relationships
2. the scoring function
3. the loss function

As stated earlier our entities and relations are represented through vectors, their embeddings. Some models vary from this a bit and use complex numbers instead of real ones [**?**] or use matrices to represent relationships [**?**].
The score function $f(h, r, t)$ calculates the distance between the embeddings of two entities relative to their relation. If the triple holds true, its score should be close to 0.
Lastly the loss function defines the objective which is going to be minimized during the training of our model where the embeddings for our entities and relations are learned.

3.2.1 ComplEx

3.2.2 RESCAL

3.3 Comparison of the Approaches

3.4 Model Evaluation

Chapter 4

Experimental Setting

Chapter 5

Comparison of Symbolic and Sub-Symbolic Performances

Chapter 6

Test Sets for Vulnerabilities

Chapter 7

Discussion

Chapter 8

Conclusion

Bibliography

[1] Matthias Baumgartner, Daniele Dell'Aglio, and Abraham Bernstein. Entity Prediction in Knowledge Graphs with Joint Embeddings. In *Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-15)*, pages 22–31, Mexico City, Mexico, June 2021. Association for Computational Linguistics.

[2] Jennifer Golbeck. *Analyzing the Social Web*. Newnes, February 2013. Google-Books-ID: XP8jc2cDNrwC.

[3] Eleni Ilkou and Maria Koutraki. *Symbolic Vs Sub-symbolic AI Methods: Friends or Enemies?* November 2020.

[4] Christian Meilicke, Melisachew Wudage Chekol, Manuel Fink, and Heiner Stuckenschmidt. Reinforced Anytime Bottom Up Rule Learning for Knowledge Graph Completion. *arXiv:2004.04412 [cs]*, April 2020. arXiv: 2004.04412.

[5] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A Review of Relational Machine Learning for Knowledge Graphs. *Proceedings of the IEEE*, 104(1):11–33, January 2016. arXiv:1503.00759 [cs, stat].

[6] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, 8(3):489–508, December 2016.

[7] Amit Singhal. Introducing the Knowledge Graph: things, not strings, May 2012.

[8] John Talburt. *Entity Resolution and Information Quality*. Elsevier, January 2011. Google-Books-ID: tIB0IZYR8V8C.

[9] Xiaohan Zou. A Survey on Application of Knowledge Graph. *Journal of Physics: Conference Series*, 1487(1):012016, March 2020.

Ehrenwörtliche Erklärung

Ich versichere, dass ich die beiliegende Master-/Bachelorarbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen. Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

Mannheim, den 31.08.2022 Unterschrift