An Extensible Approach for Query-Driven Multimodal Knowledge Graph Completion

Marcelo Machado[1], Guilherme Lima[1], Elton Soares[1], Vítor Nascimento[2], Rafael Brandao[1] and Marcio Moreno[1]

[1]IBM Research, Rio de Janeiro - RJ, Brazil
[2]Universidade Federal Fluminense, Rio de Janeiro - RJ, Brazil

Abstract
The knowledge graph completion task has gained a lot of attention in recent years, especially with the use of machine learning (ML). However, most of the work has focused on the structure of the graph while ignoring the data it describes. In this demo, we present an approach that does the opposite: it leverages the multimodal data described by a knowledge graph for its completion. We use IBM's Hyperlinked Knowledge Graph framework, which allows nodes of the graph to carry arbitrary data content. This content is processed at query time by user-defined functions which are triggered by rules and whose output is used to decide the materialization of new links, completing the original graph. To demonstrate the approach, we use ML models to classify images of paintings and decide the materialization of links describing their semantics. DEMO

Keywords
Hyperknowledge, Hyperlinked Knowledge Graph, Knowledge Graph Completion, Multimodal data

1. Introduction

Knowledge Graphs (KGs) are widely used to enrich applications with factual knowledge about objects in the world. In traditional KG systems, only symbolic content (*e.g.*, concepts, instances) is represented, while non-symbolic content (*e.g.*, images, videos, scripts) and its integration with the knowledge stored in the KG have to be handled by external systems. In contrast, in Multimodal KGs (MMKGs) [1, 2], the relationships between symbolic and non-symbolic content are represented natively, enabling richer knowledge discovery and consumption.

Because of their generality and scope, real-world KGs and MMKGs are large and often incomplete [3, 4]. This means that important nodes or links might be missing; absences which can decrease the accuracy of queries and inhibit important insights from applications. The KG completion task has emerged to address this problem in a scalable manner [5].

Much of the research in KG completion has focused on predicting what is missing based solely on the structure of the graph, usually represented in the form of low dimensional spaces (*i.e.*, embedding space) [6], while fewer works have proposed to incorporate both multimodal data and graph structure embeddings into machine learning (ML) models [2, 4]. Also, completion techniques typically apply inference rules in batches [7] or ML models to the whole graph [8].

ISWC 2022: 21st International Semantic Web Conference, October 23–27, 2022, Hangzhou, China
mmachado@ibm.com (M. Machado); guilherme.lima@ibm.com (G. Lima); eltons@ibm.com (E. Soares); vitorlourenco@id.uff.br (V. Nascimento); rmello@br.ibm.com (R. Brandao); mmoreno@br.ibm.com (M. Moreno)

Our proposal differs from previous approaches in two key aspects: (1) It relies on an extensible rule-based mechanism that completes the graph with new links considering the non-symbolic and symbolic data. (2) It considers human expertise in the loop since the inference of new links is triggered by the execution of user-defined queries. Thus, our work provides an interactive and on-demand MMKG completion solution that can be easily extended by the user.

We instantiate our proposal in IBM's Hyperlinked Knowledge Graph (HKG) [9], an MMKG framework with support for rules, nodes containing executable code, and the capability of representing n-ary relationships among symbolic and non-symbolic data. The figures and videos of this demonstration were made using the Knowledge Explorer System (KES) [10].

2. Proposed Approach

The HKG is composed of traditional graph components such as nodes (*i.e.*, circles in Fig. 1) and links but it also offers specialized components that are particularly interesting for our proposal. We leverage two such specialized components to support extensible query-driven MMKG completion: *possibility links* and *possibility resolvers*. A *possibility link* represents an unasserted link, *i.e.*, a yet-to-be-fulfilled possibility in the KG, which is ignored during regular query evaluation. For example, a *possibility link* can be used to represent the possibility that a painting *X*, initially without authorship, was painted by Salvador Dalí. Meanwhile, a *possibility resolver* specifies how the assertion of a *possibility link* should be resolved. For example, we can use a *possibility resolver* to indicate how to decide whether *X* was painted by Salvador Dalí. The *possibility resolver* is represented by an n-ary link whose components act as parameters to solve a given possibility. Its main component is an executable node, *i.e.*, nodes containing a wrapper code that invokes a function to be executed to perform a task on the graph. These nodes can evaluate whether *possibility links* should be materialized as asserted links.

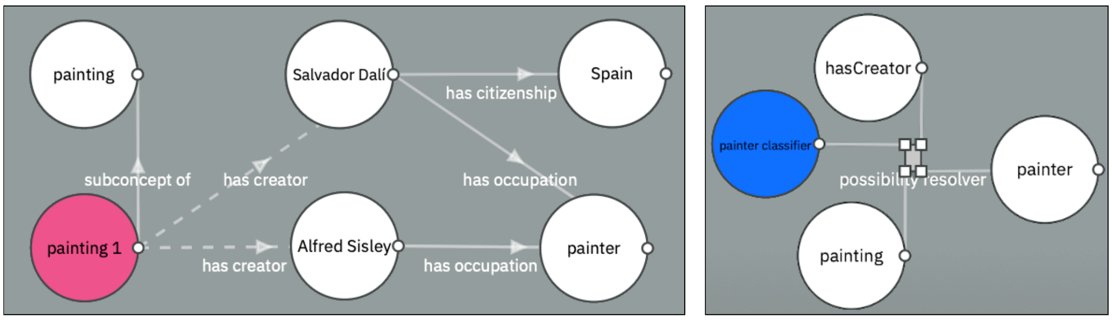

Figure 1: MMKG visualization, in KES, depicting links (solid lines) and *possibility links* (dashed lines) between concept nodes and a content node (pink), and a *possibility resolver* (gray rectangle with four white squares connected to four solid lines) between an executable node (blue) and concept nodes.

To illustrate the usage of these components in our approach, consider Figure 1. In this example, we have a set of images of paintings, an ontology of painters and painting styles, and a set of executable nodes containing functions that calls ML models to classify images by painter or painting style. Here we want to be able to answer queries such as "Give me all impressionist paintings" or "List all paintings created by a Spanish painter".

To answer these queries by leveraging the multimodal knowledge available in this scenario (ontology, images, and ML models), we represent the images as content nodes (*i.e.*, nodes that stand for non-symbolic content) and link them, using *possibility links*, to instances of concepts of the ontology, such as painting, painting style, and painters. We define executable nodes to invoke an ML model passing the non-symbolic data of a content node as a parameter (*e.g.*, an image).[1] Finally, we define *possibility resolvers* to evaluate the possibility of materializing the *possibility links* by executing the adequate ML model for each relation defined in the ontology (*e.g.*, 'hasCreator' corresponds to the painter of an image, therefore the ML model that identifies the painter would be used to evaluate *possibility links* of this relation).

A *possibility resolver* can be formally defined as a statement of the form $\mathbf{R}(f, S, p, T)$ establishing that entity f is a procedure that decides whether an instance or subclass of S is related via predicate p to an instance or subclass of T. The actual representation of this statement depends on the KG system being used. [2] In HKG, this is represented as a 4-ary relationship among f, S, p, and T, as illustrated in Figure 1. Thus, a rule of the form

$$S(x) \land T(y) \land \mathbf{R}(f, S, p, T) \rightarrow \mathbf{P}_f(x, p, y)$$

is used to materialize (to convert from a *possibility link* to a regular link) the *possibility links* from the declared *possibility resolvers.* This rule states that if x is an instance or subclass of S, y is an instance or subclass of T, and f resolves whether p holds from an S to a T, then there is a *possibility link* from x to y with label (predicate) p which can be resolved by f.

During query evaluation, the links inferred by the above rule are used to decide whether a given relationship exists in the graph. For instance, suppose that the query evaluator needs to decide whether $p(x, y)$, that is, whether there is a link with predicate p from x to y. If the link does not exist in the graph and was not inferred, then the result is false. However, if the link exists or was inferred there are two cases. Either the link is a regular link (not a possibility) and in this case, the result is true. Or the link is a *possibility link* in which case the evaluator calls the associated *possibility resolver* f with arguments x, p, and y and the result of this call is the result of the test. If f is deterministic (gives the same result for the same input) then the evaluator can save the trouble of going through this process again by materializing the link.

The entire process starts from a user-determined query execution – hence, human-in-the-loop and query-driven. HKG provides a query language, the Hyperknowledge Query Language (HyQL) [11], that enables the retrieval of the multimodal information described in the graph. For example, a simple HyQL query to retrieve paintings created by Salvador Dali would be:

select Painting **where** Painting hasCreator SalvadorDali[3]

In this case, the relationship $p(x, y)$ was instantiated in the 'where' clause, where p = *hasCreator*, x = *Painting* and y = *SalvadorDali.* If this possibility exists and it has not yet been materialized, then all the aforementioned process is triggered. The output of this process is the query response together with the completion of the graph in case of positive responses of the functions.

[1] We are using ML models here, but any user-defined function could be used to handle multimodal content.

[2] In RDF, this would have to be encoded as a set of triples using some form of reification. If OWL is used and if S and T are the domain and range of p then an assertion linking f to p might be sufficient.

[3] The entities that appear in natural language in the query, such as SalvadorDali, can be aliases for URIs, for example, from Wikidata.

3. Demonstration

The goal of this demo is to show HKG features that support extensible query-driven MMKG completion through KES and HyQL. The demo uses the Kaggle's Best Artworks of All Time dataset[4], two ML models trained over this dataset, and a simple ontology containing painters (Salvador Dalí, Alfred Sisley, etc.), nationalities (Spain, UK, etc.), art movements (Surrealism, Impressionism, Expressionism, etc.), and themes (landscape, portrait, etc.). The images, ML models, and ontology are used to demonstrate how the HKG can be used to infer links from multimodal data at query time while enabling rich semantic queries.

Video 1: https://ibm.box.com/v/iswc2022keg1 – MMKG Representation with HKG and KES.

Video 2: https://ibm.box.com/v/iswc2022keg2 – MMKG Completion with HyQL and KES.

References

[1] Y. Liu, H. Li, A. Garcia-Duran, M. Niepert, D. Onoro-Rubio, D. S. Rosenblum, Mmkg: multi-modal knowledge graphs, in: European Semantic Web Conference, Springer, 2019, pp. 459–474.

[2] S. Liang, A. Zhu, J. Zhang, J. Shao, Hyper-node relational graph attention network for multi-modal knowledge graph completion, ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM) (2022).

[3] B. Shi, T. Weninger, Open-world knowledge graph completion, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 32, 2018.

[4] X. Chen, N. Zhang, L. Li, S. Deng, C. Tan, C. Xu, F. Huang, L. Si, H. Chen, Hybrid transformer with multi-level fusion for multimodal knowledge graph completion, CoRR abs/2205.02357 (2022). doi:10.48550/arXiv.2205.02357. arXiv:2205.02357.

[5] Q. Wang, Z. Mao, B. Wang, L. Guo, Knowledge graph embedding: A survey of approaches and applications, IEEE Transactions on Knowledge and Data Engineering 29 (2017) 2724–2743.

[6] M. Nickel, K. Murphy, V. Tresp, E. Gabrilovich, A review of relational machine learning for knowledge graphs, Proceedings of the IEEE 104 (2015) 11–33.

[7] Q. Wang, B. Wang, L. Guo, Knowledge base completion using embeddings and rules, in: Twenty-Fourth International Joint Conference on Artificial Intelligence, 2015.

[8] Q. Wang, Y. Ji, Y. Hao, J. Cao, Grl: Knowledge graph completion with gan-based reinforcement learning, Knowledge-Based Systems 209 (2020) 106421.

[9] M. F. Moreno, R. Brandao, R. Cerqueira, Extending hypermedia conceptual models to support hyperknowledge specifications, Int. J. Semant. Comput. 11 (2017).

[10] M. F. Moreno, R. C. Santos, W. H. dos Santos, R. Cerqueira, Kes: The knowledge explorer system., in: International Semantic Web Conference (P&D/Industry/BlueSky), 2018.

[11] M. F. Moreno, P. Costa, R. Costa, V. Nascimento, E. F. de Souza Soares, M. Machado, Evaluating semantic queries for dataset engineering on the hyperknowledge platform., in: ISWC (Posters/Demos/Industry), 2021.

[4]https://www.kaggle.com/ikarus777/best-artworks-of-all-time